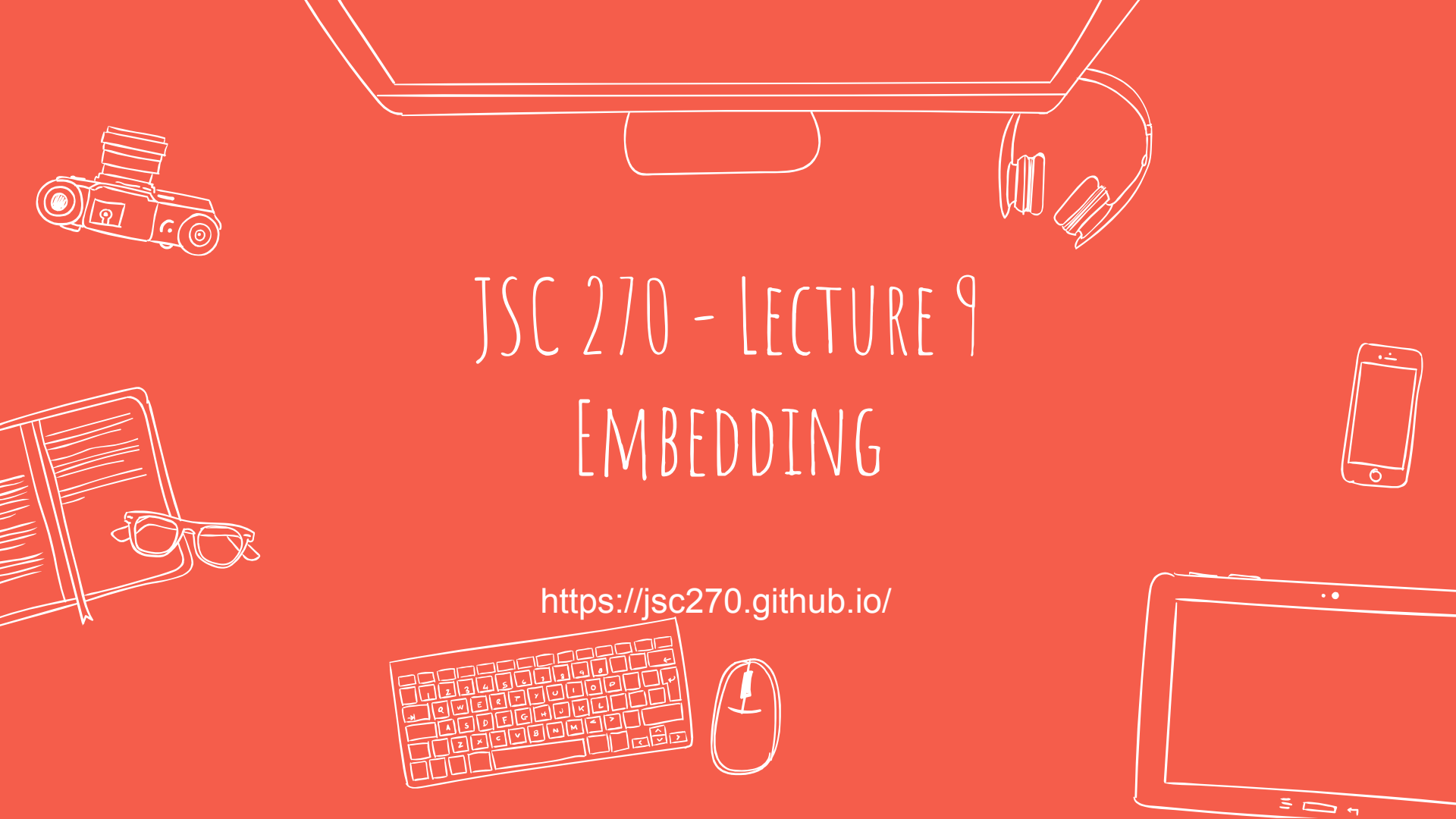


JSC 270 - Lecture 9
Embedding

https://jsc270.github.io/

Announcements

- Perusall paper is posted
- Reflection quiz will be graded by the end of today
- Next week lecture: data science in healthcare

  talk by Devin Singh, MD, MSc (SickKids/HeroAI)
- Paper I mentioned last class by Zack Lipton on NLP: https://arxiv.org/pdf/1909.12434.pdf

Dimensionality Reduction

How would you define it?

Dimensionality Reduction

Dimensionality Reduction is the transformation of data from a *high-dimensional space* into a *low-dimensional space* so that the low-dimensional representation retains some meaningful properties of the original data

Wikipedia

Dimensionality Reduction

What applications can you think of?

Dimensionality Reduction

In any AI and stats applications where

a) There is a reason to believe that the data can be well captured by fewer dimensions
b) There is a need to reduce data, for e.g. for computational reasons

Primary Uses:

- Exploratory data analysis
- Visualization

Applications: ubiquitous across fields (healthcare, industry; text, speech, imaging, etc)

Dimensionality Reduction

Which techniques do you know?

Dimensionality Reduction

- Matrix factorization based
  - Principle Component Analysis (PCA)
    - Graph-based PCA
    - Kernel-based PCA
  - Non-negative Matrix Factorization (NMF)
  - Linear Discriminant Analysis (LDA)
  - Generalized Discriminant Analysis (GDA)
  - Random projections
- Neighborhood based
  - t-SNE
  - UMAP
  - Locally Linear Embedding
  - ISOMAP
- Autoencoders (reconstruction based)

And many more

https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

Dimensionality Reduction vs Embedding

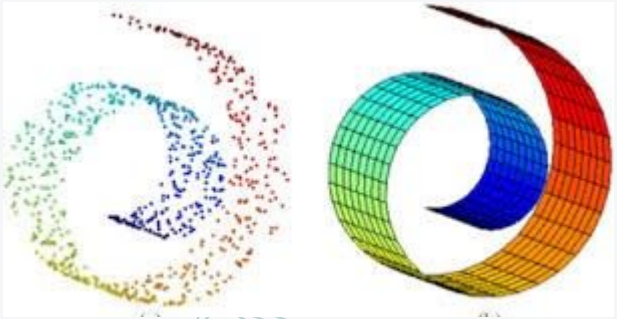
High-dimensional data often lies on or near a much lower dimensional manifold.

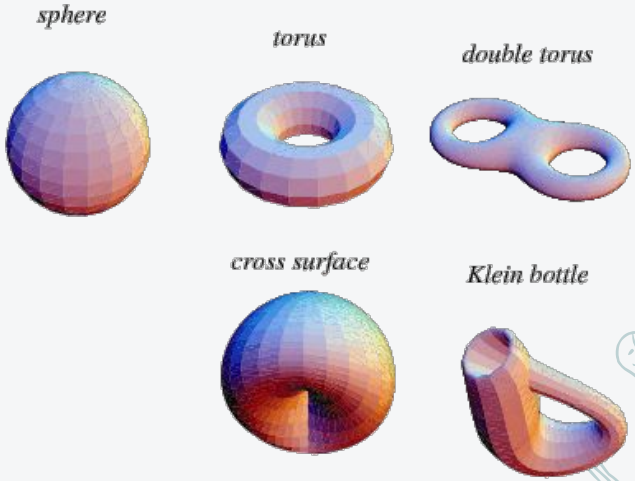
We reduce dimension or *embed* data into the manifold to get the *embedding*, in other words a low dimensional representation

Useful Concepts: MANIFOLD

Manifold - a collection of points forming a certain kind of set, such as those of a topologically closed surface in three or more dimensions (such topological space that locally resembles Euclidean space near each point)

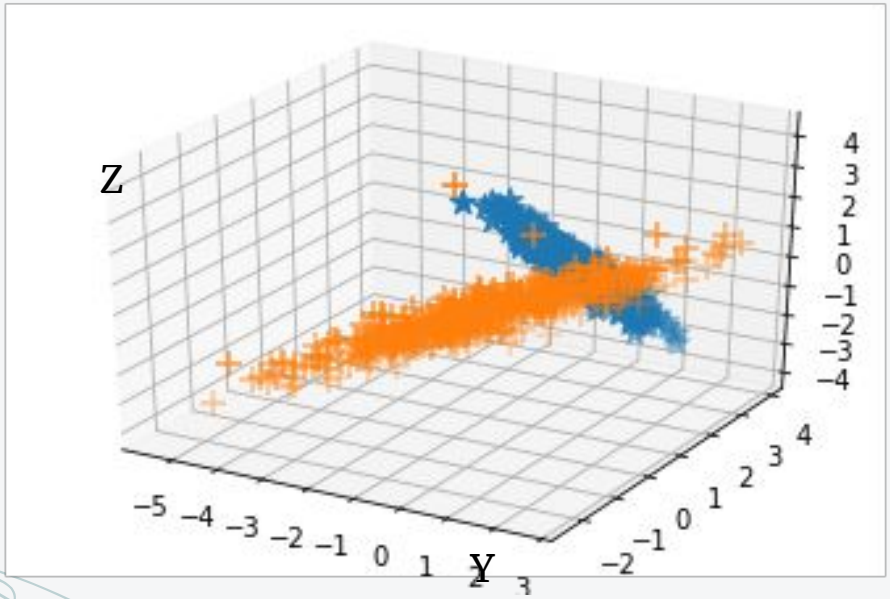
Swiss roll





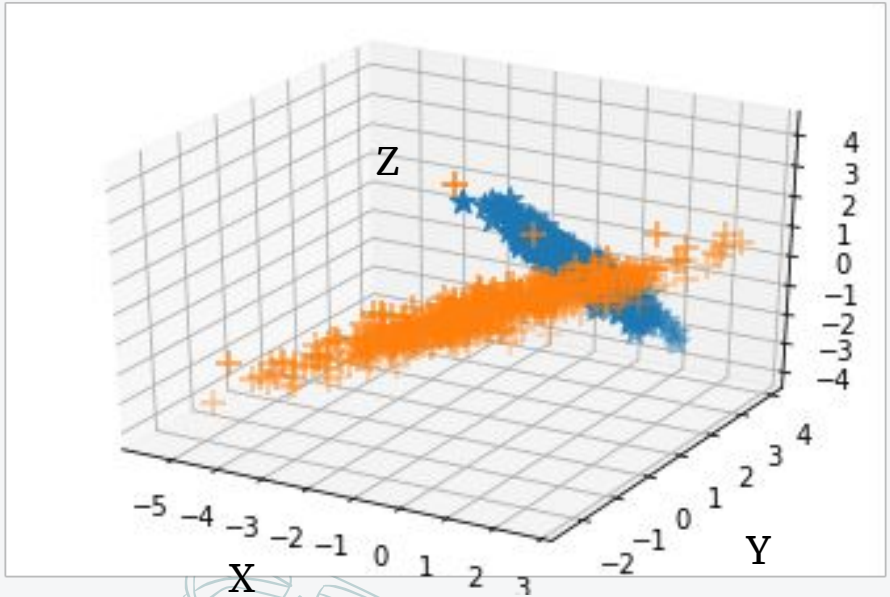
Data

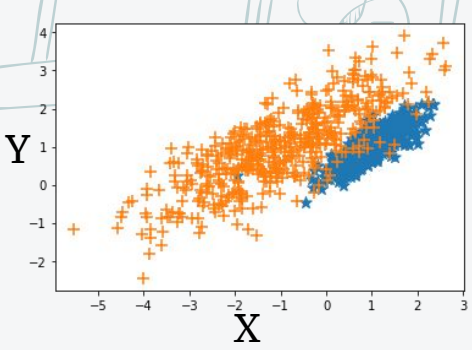
Number of samples N = 1000
Number of features P = 3

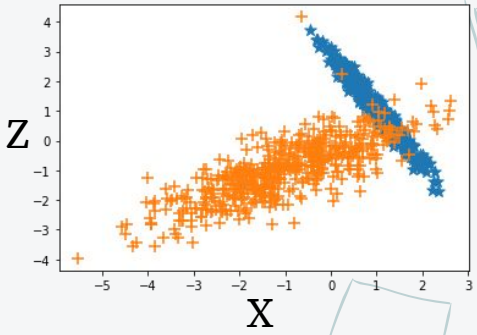


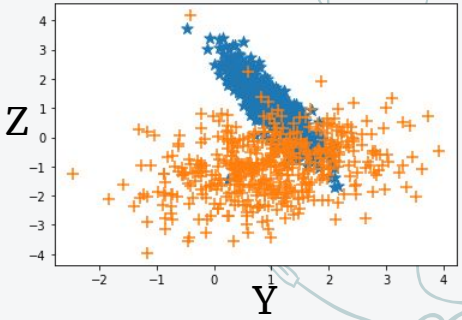
Data

Number of samples N = 1000
Number of features P = 3









Principal Component Analysis

Principle components = axes of greatest variability

When we talk about X1,X2,...,Xp - these features are in cartesian coordinate system

Principle components are a *new* set of p axes in the direction of greatest variability, that are a linear combination of the original

Selecting Principal Components

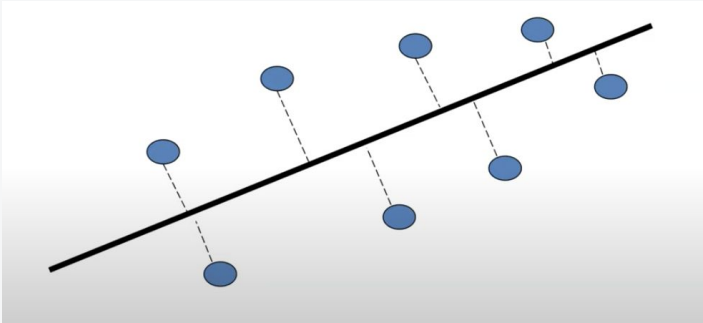
In one dimension, we would projecting points to a line.

How do we do that?

Selecting Principal Components

In one dimension, we would projecting points to a line.

How do we do that? Regression (squared loss)



Selecting Principal Components

In one dimension, we would projecting points to a line.

How do we do that? Regression (squared loss)

How do we do it in many dimensions?

Linear Algebra (review)

- Eigenvectors for a square m x m matrix **S**

$$\boldsymbol{S}\boldsymbol{v} = \lambda \boldsymbol{v}$$

Eigenvector $\boldsymbol{v} \in \mathbb{R}^m, \boldsymbol{v} \neq \mathbf{0}$

Eigenvalue $\lambda \in \mathbb{R}$

E.g

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

eigenvector

eigenvalue

Linear Algebra (Review)

How many eigenvalues are there at most?

Linear Algebra (Review)

How many eigenvalues are there at most?

$$\boldsymbol{S}\boldsymbol{v} = \lambda \boldsymbol{v} \equiv (\boldsymbol{S} - \lambda \boldsymbol{I})\boldsymbol{v} = 0$$

only has non-zero solutions if $|\boldsymbol{S} - \lambda \boldsymbol{I}| = 0$

Can have at most *m* distinct solutions

If $S = S^T \Rightarrow \lambda \in \mathbb{R}$

Linear Algebra (Review)

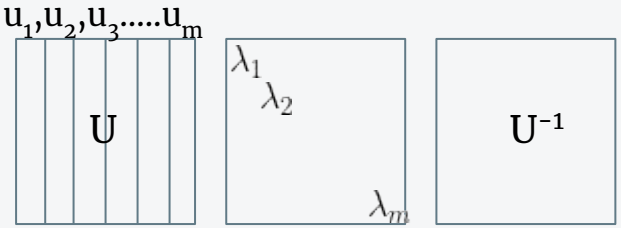
For *symmetric matrices*, eigenvectors for distinct eigenvalues are *orthogonal*

All eigenvalues for a *positive semidefinite matrix* are *non-negative*

Note: S is positive semidefinite if $\forall w \in \mathbb{R}^m, w^T S w \geq 0$

Eigen Decomposition

Eigen decomposition transforms a matrix into its principal vectors of variation

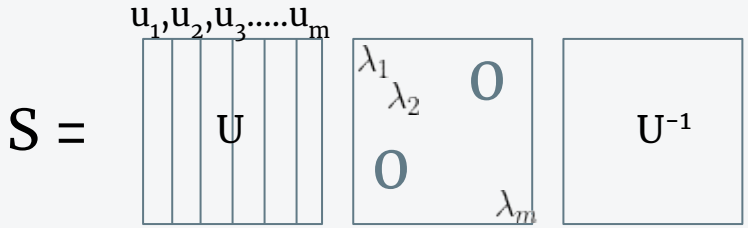
S = 

$$S = U \Lambda U^{-1}$$

U are eigenvectors of S, diagonal values of Lambda are eigenvalues.

Eigen Decomposition

Eigen decomposition transforms a matrix into its principal vectors of variation



Decomposition is unique if eigenvalues are unique

$$S = U \Lambda U^{-1}$$

Columns of U are eigenvectors of S

Diagonal values of Lambda are eigenvalues of S

Singular Value Decomposition (SVD)

For an m x n matrix **A** of rank r there exists a factorization (SVD) as follows:

$$A = U\Sigma V^T$$

m x m m x n n x n

The columns of *U* are orthogonal eigenvectors of $\mathbf{AA}^T$

The columns of V are orthogonal eigenvectors of $\mathbf{A}^T\mathbf{A}$

Eigenvalues $(\sigma_1, \sigma_2, \ldots, \sigma_r)$ of $\mathbf{AA}^T$ are eigenvalues of $\mathbf{A}^T\mathbf{A}$

$$\Sigma = \begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ 0 & & \sigma_r \end{bmatrix}$$

Note: rank r means there are max of r linearly independent rows and columns in A

Singular Value Decomposition (SVD)

For an m x n matrix **A** of rank r there exists a factorization (SVD) as follows:

$$A = U\Sigma V^T$$

m x m m x n n x n

The columns of *U* are orthogonal eigenvectors of $\mathbf{AA}^T$

The columns of V are orthogonal eigenvectors of $\mathbf{A}^T\mathbf{A}$

Eigenvalues $(\sigma_1, \sigma_2, \ldots, \sigma_r)$ of $\mathbf{AA}^T$ are eigenvalues of $\mathbf{A}^T\mathbf{A}$

$$\Sigma = \begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ 0 & & \sigma_r \end{bmatrix}$$

Singular Values

Note: rank r means there are max of r linearly independent rows and columns in A

Singular Value Decomposition (SVD)

For an m x n matrix **A** of rank r there exists a factorization (SVD) as follows:

$$A = U\Sigma V^T$$

m x m m x n n x n

The columns of *U* are orthogonal eigenvectors of $\mathbf{AA}^T$

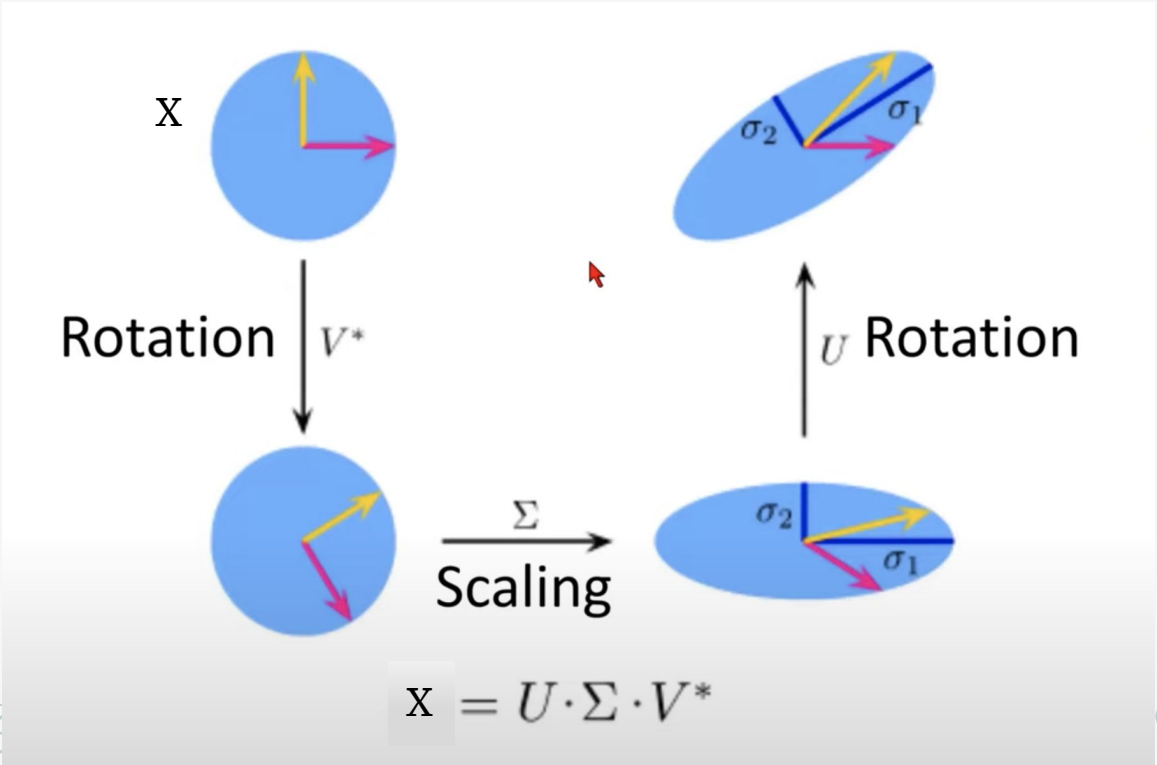
The columns of V are orthogonal eigenvectors of $\mathbf{A}^T\mathbf{A}$

Eigenvalues $(\sigma_1, \sigma_2, \ldots, \sigma_r)$ of $\mathbf{AA}^T$ are eigenvalues of $\mathbf{A}^T\mathbf{A}$

$$\Sigma = \begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ & & \\ 0 & & \sigma_r \end{bmatrix} \quad \lambda_i = \sqrt{\sigma_i}$$

Note: rank r means there are max of r linearly independent rows and columns in A

Geometric interpretation of SVD



Low Rank Approximation

Rank r of matrix A might not be small, but we can find A_k with rank k << r :

$$A_k = \min_{Z:rank(Z)=k} \|A - Z\|_F$$

Frobenius (Euclidean) norm

$$\|X\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|X_{ij}|^2}$$

A and Z are both m x n matrices

Low Rank Approximation

Solution via SVD

$$A_k = U diag(\sigma_1, \ldots, \sigma_k, \underbrace{0, \ldots, 0}) V^T$$

Set smallest r-k singular values to 0

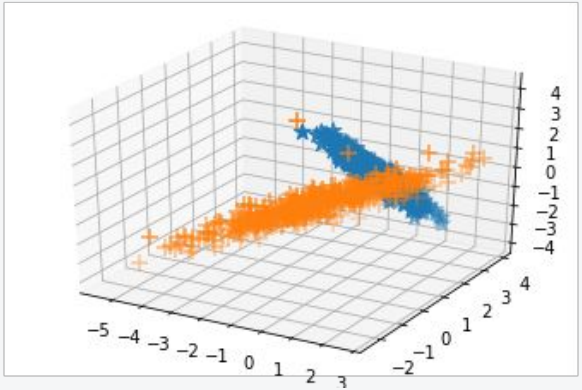
Error: $\|A - A_k\|_F = \sigma_{k+1}$

Back to PCA

- Find eigenvectors and eigenvalues using SVD
- Examine how much variance is explained using k components (k is varied)
- For the purposes of explaining (reconstructing the data), pick the number of components based on variance explained
- For plotting purposes can plot the first 2 PCA or all pairs of interest

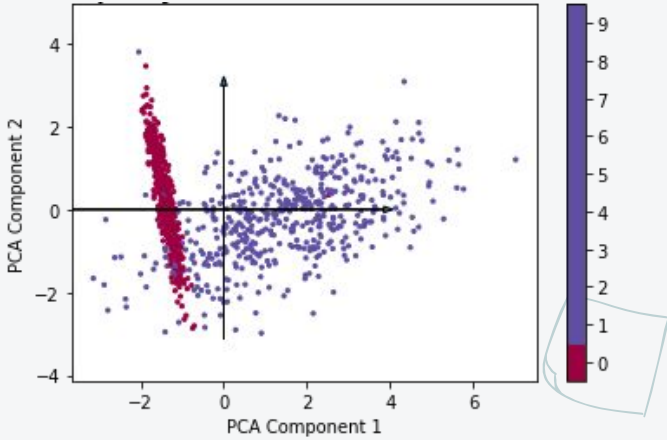
PCA

DATA



Variance explained:

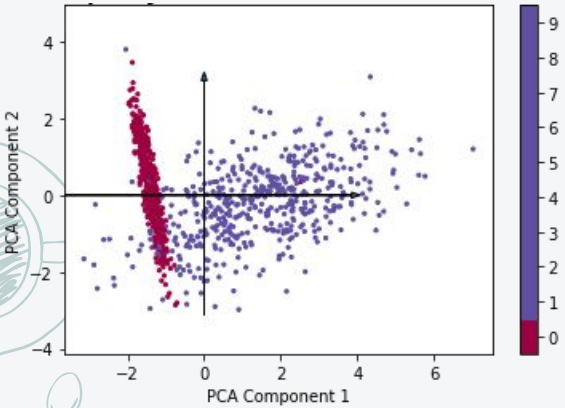
PC1	PC2	PC3
[0.73	0.24	0.03]

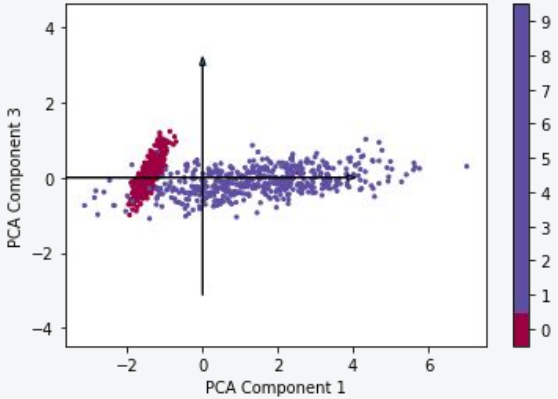


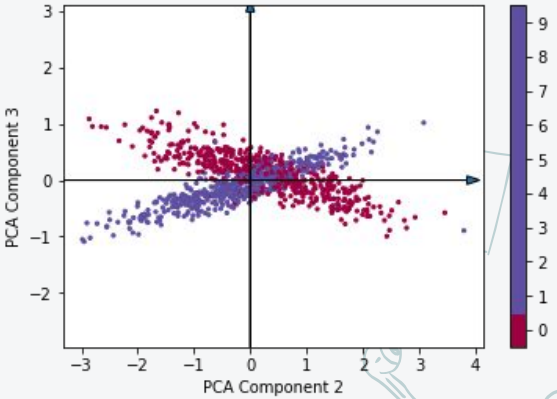
PCA

Variance explained:

PC1	PC2	PC3
[0.73	0.24	0.03]





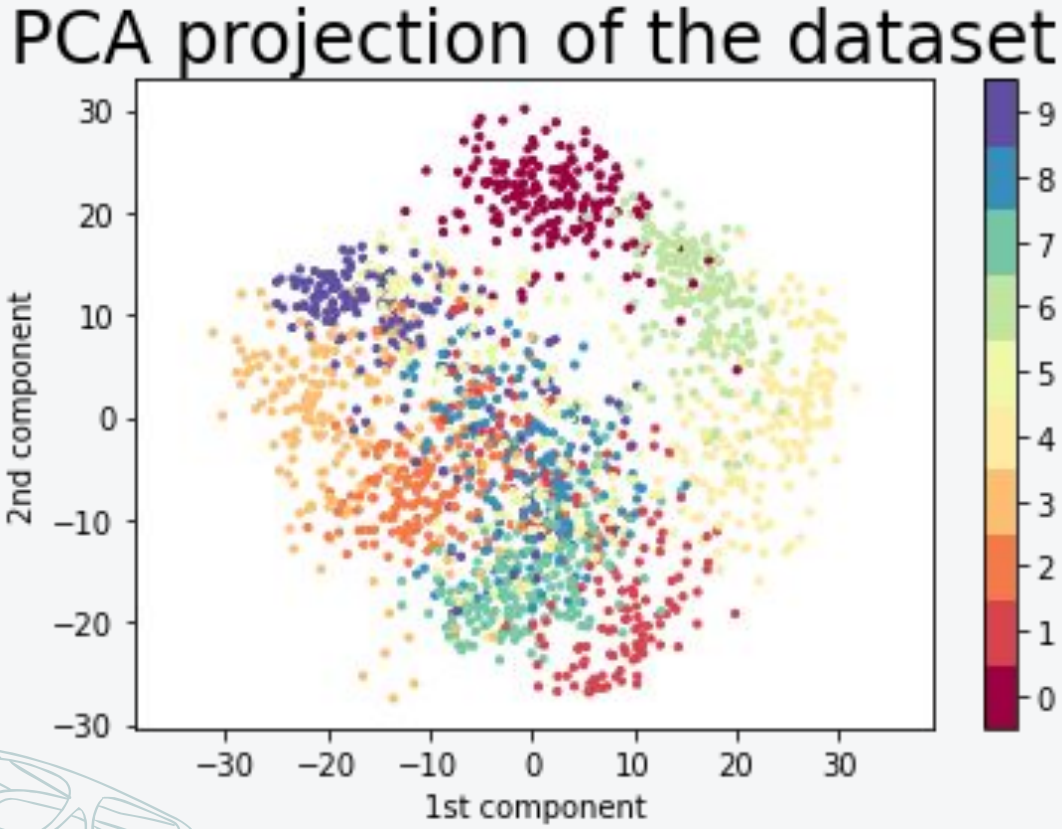


Another dataset: Digits

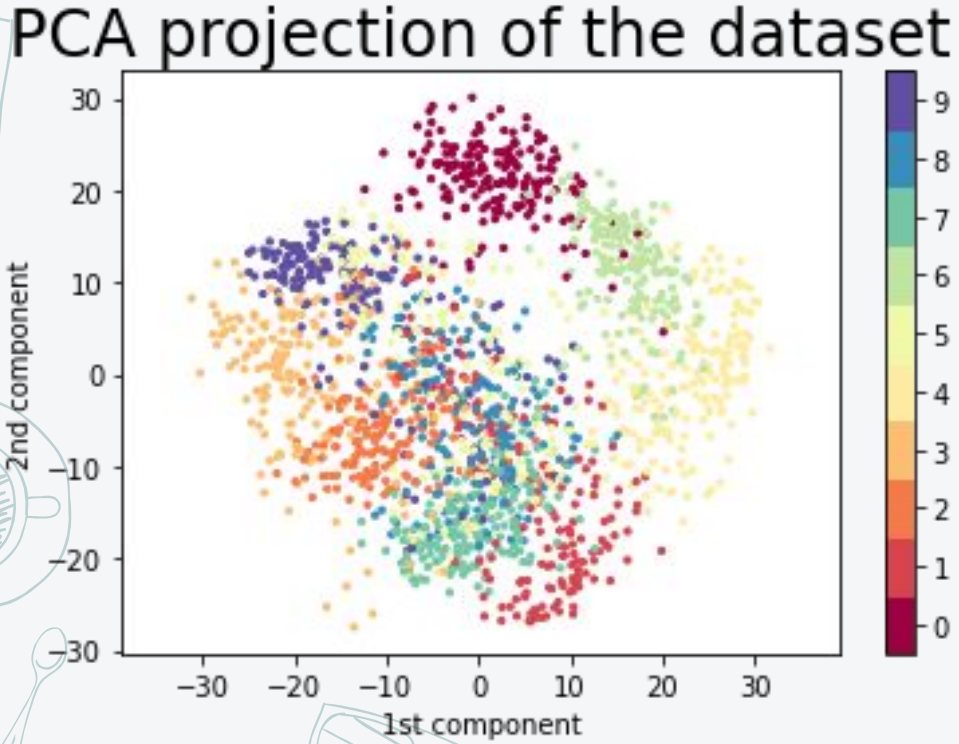
Optical recognition of handwritten digits dataset

Number of Instances: 5620
Number of Attributes: 64
Attribute Information: 8x8 image of integer pixels in the range 0..16
Creator: E. Alpaydin
Date: July, 1998

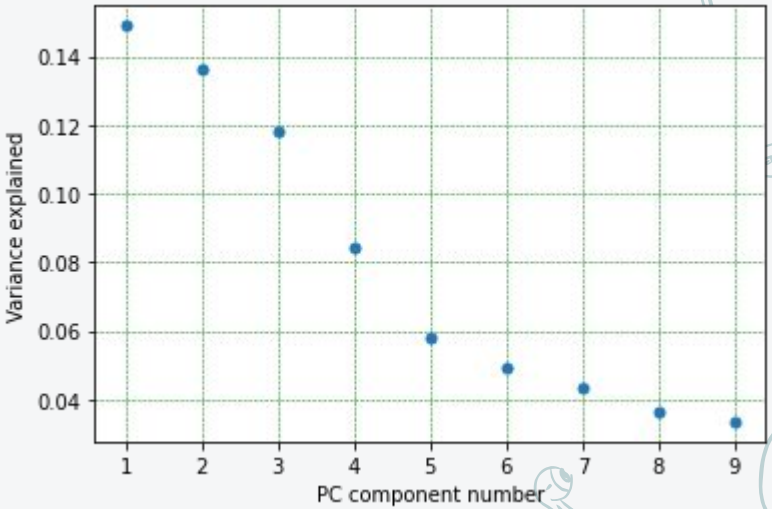
PCA: DIGITS


PCA projection of the dataset
2nd component
1st component
30
20
10
0
−10
−20
−30
−30
−20
−10
0
10
20
30
9
8
7
6
5
4
3
2
1
0

PCA: DIGITS



Variance explained



PCA

Advantages:

- Relatively Fast
- Gives some sense of intrinsic dimensionality
- Can compute variation explained
- Axes are interpretable

Disadvantages:

- Linear reduction in dimensionality
- Orthogonal components (hard to model non-linear data)

Non-linear Lower Dimensional Embedding

Neighborhood based

- T-SNE
- UMAP
- ISOMAP
- LLE

Autoencoder-based

And many more...

t-SNE : t-distributed Stochastic Network Embedding

- *Distance preservation:* when 2 points are close in high dimensional space, they should be close in 2 dimensional space
- *Neighbor preservation:* neighborhoods of a given point in high-d should be neighborhoods of a given point in low-d

  Note1: effective number of neighbors is a parameter called *perplexity*

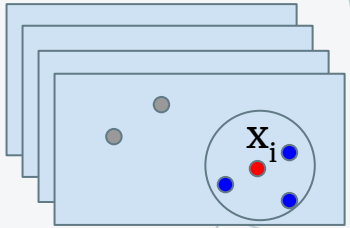
Note2: t-SNE focuses on the order of things, not necessarily on the distance itself

Van der Maaten and Hinton, 2008

T-SNE

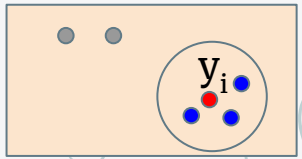
Similarity of data points in High Dimensional space

$$p_{ij} = \frac{exp(-\|x_i - x_j\|^2/2\sigma^2)}{\sum_{k \neq l} exp(-\|x_k - x_l\|^2/2\sigma^2)}$$



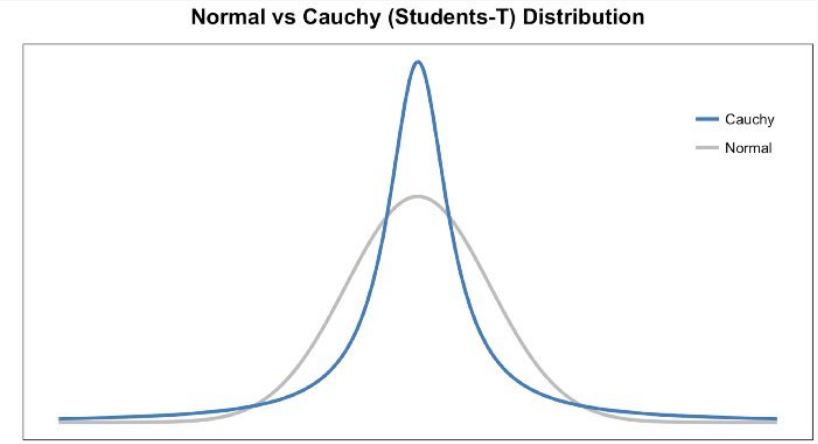
Similarity of points in Low Dimensional space

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}}$$



T-SNE

Reason for t-distribution as opposed to Gaussian as in high d is crowding - need fatter tails to accommodate distant points



t-SNE cost function

Cost function: Kullback Leibler divergence (relative entropy)
- measures distance between two distributions

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

(Kullback and Leibler, 1951)

Properties?

t-SNE cost function

Cost function: Kullback Leibler divergence (relative entropy) - measures distance between two distributions

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

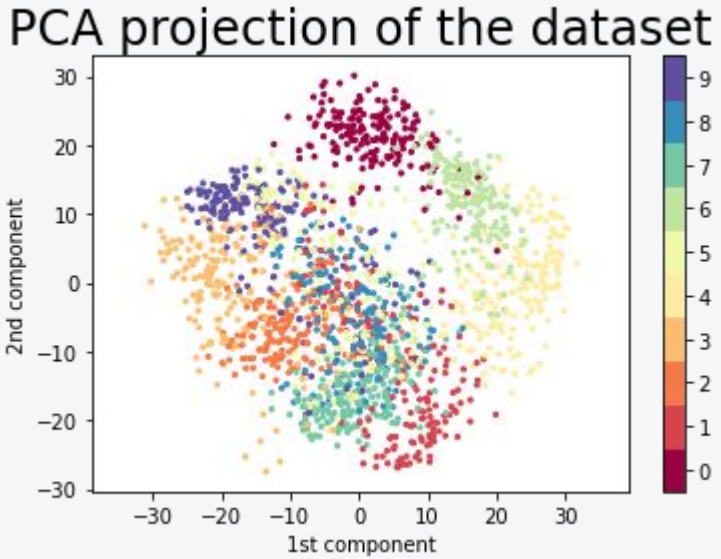
(Kullback and Leibler, 1951)

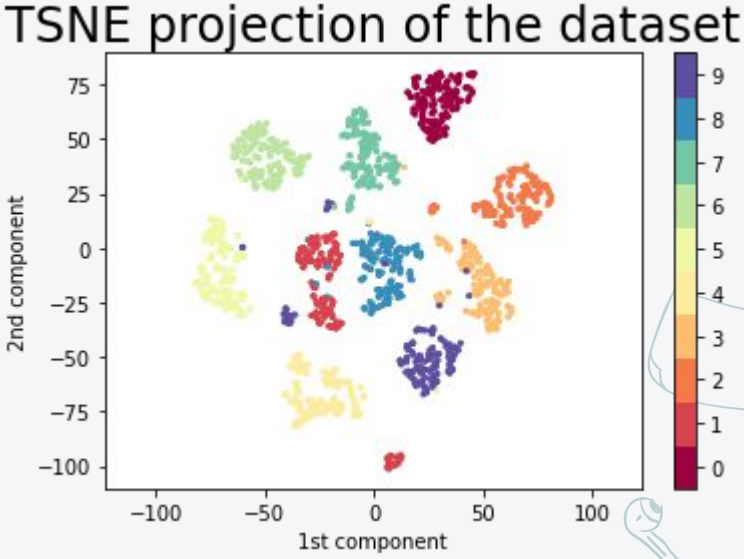
Properties:

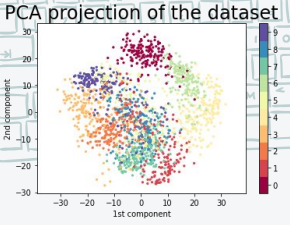
- Large p_{ij} modeled by small q_{ij} - large penalty (distant points shouldn't be close in low d)
- Small p_{ij} modeled by large q_{ij} - small penalty (ok to separate nearby points -most are beyond neighborhood)

Optimize the cost using gradient descent

T-SNE on Digits

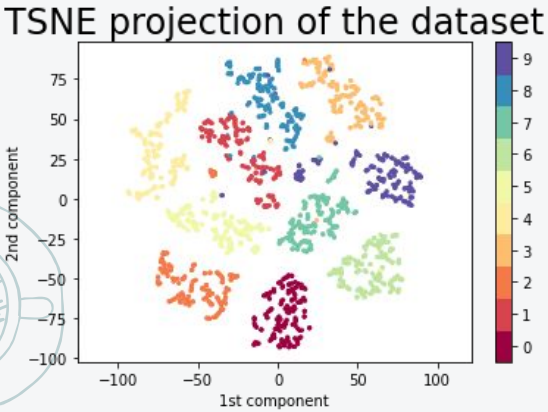




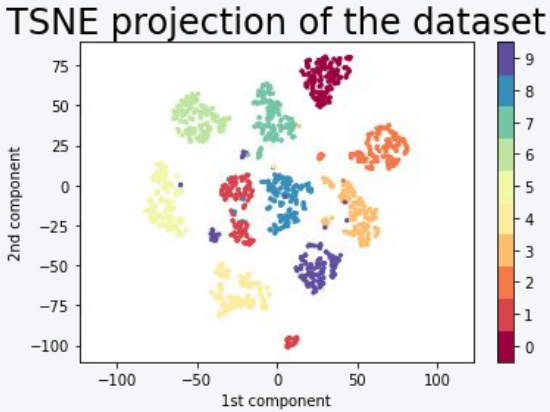


T-SNE sensitivity to parameters: Perplexity

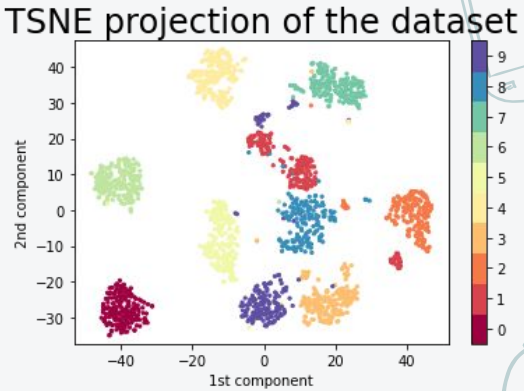
Perplexity = 5



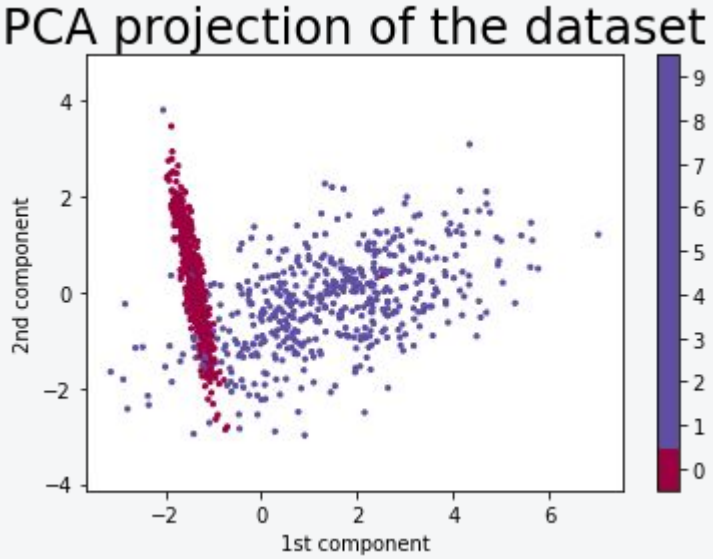
Perplexity = 10

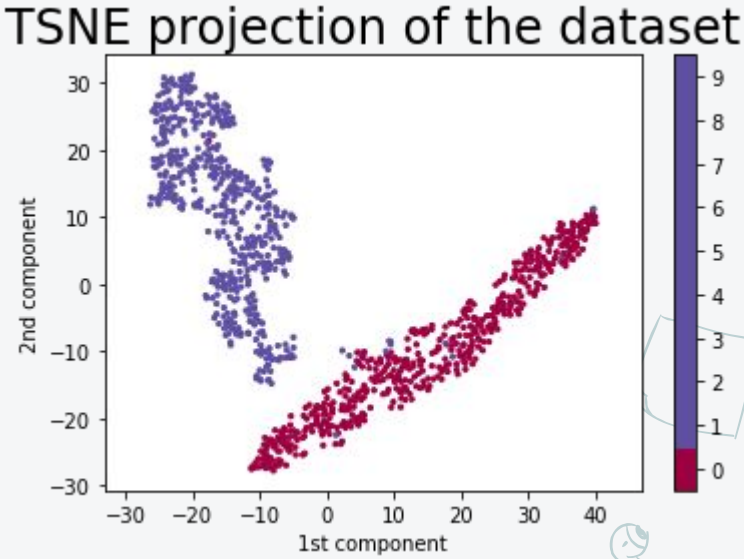


Perplexity = 50



T-SNE ON THE ORIGINAL DATASET





T-SNE

Advantages

- Retains local structure
- Excels at visualizing complex high-d data in 2d

Disadvantages

- Slower
- Doesn't really capture global structure
- Sensitive to params
- Clustering and distance depend on perplexity (shouldn't be interpreted off the plot)
- Stochasticity leads to different results
- Not guaranteed to converge to the global optimum of the cost function

https://distill.pub/2016/misread-tsne/

UMAP: Uniform Manifold Approximation and Projection

UMAP - preserves both local and global structure by

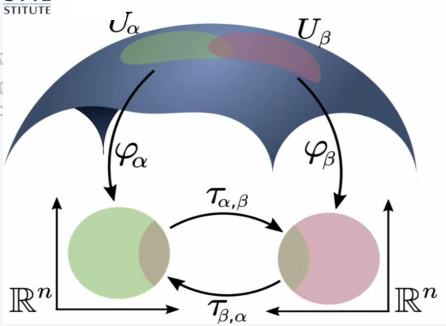
1. :earning the manifold of the data
2. Embedding that manifold into a low dimensional space
   - Builds on neighborhood based approaches but adds mathematical foundations

Key intuition:

While the assumptions may not be true in the original space, they can be true on the manifold

L McInnes, J Healy, J Melville, arXiv, 2018

UMAP



Assumptions:

- The points are covering manifold uniformly
- There is local connectedness between points (confident in the distance to the 1st neighbor)

Using topological theory, construct the neighborhood graph on the low dimensional manifold.

Optimize cross entropy:

Get the clumps right

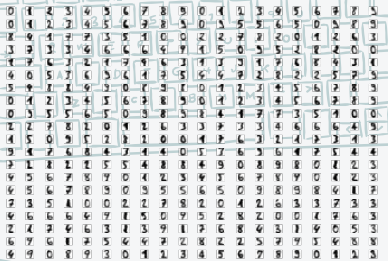
Similar to t-SNE

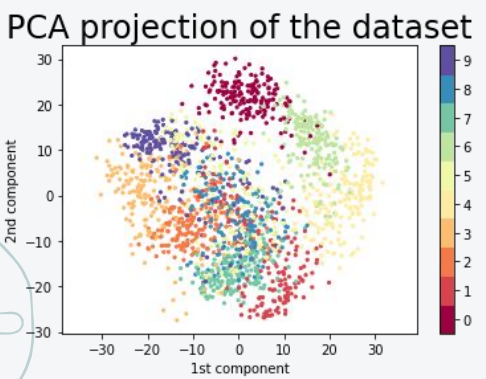
$$\sum_{a \in A} \mu(a) \log\left(\frac{\mu(a)}{\nu(a)}\right) + (1-\mu(a)) \log\left(\frac{1-\mu(a)}{1-\nu(a)}\right)$$

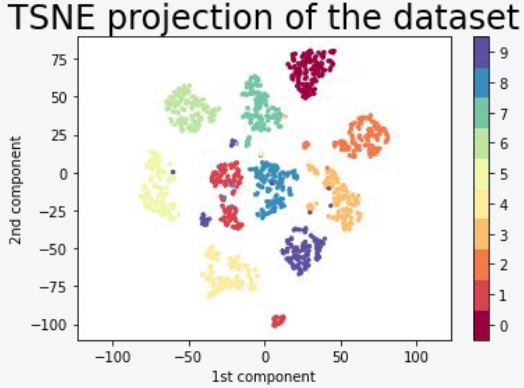
Get the gaps right

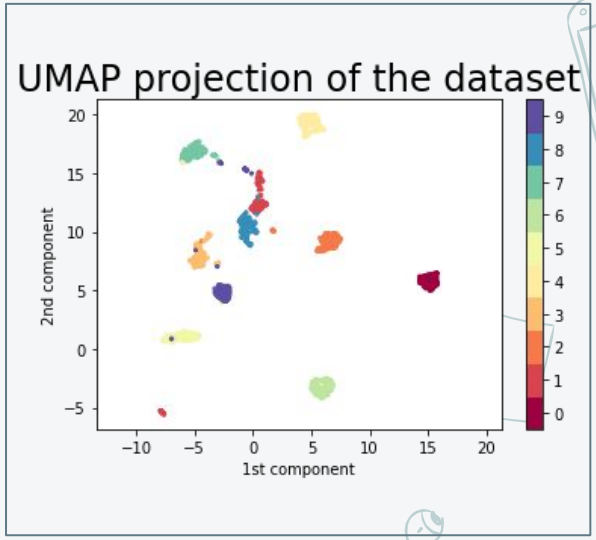
L McInnes, J Healy, J Melville, arXiv, 2018

UMAP on DIGITS





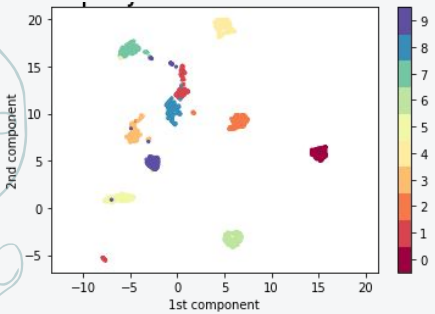




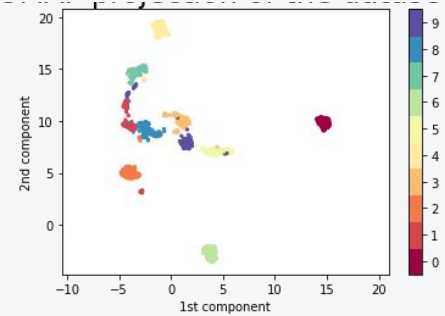
UMAP HYPERPARAMETERS

- Number of neighbors - local vs global structure
- Minimum distance - how tightly points can be packed together
- Number of components (we set 2 for all)
- Distance metric

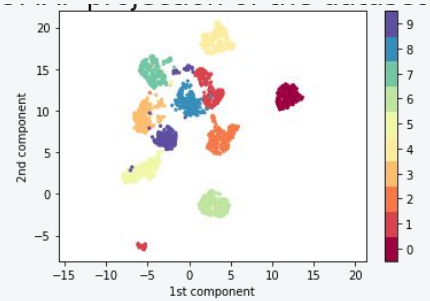
neighbors = 15
metric = euclidian
min_dist = 0.1



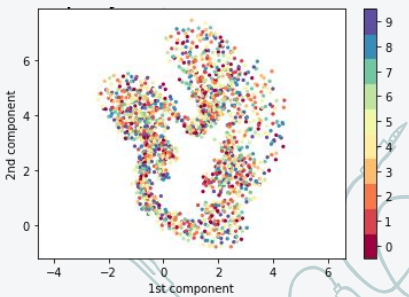
neighbors = 50
metric = euclidian
min_dist =0.1



neighbors = 5
metric = euclidian
min_dist =0.5

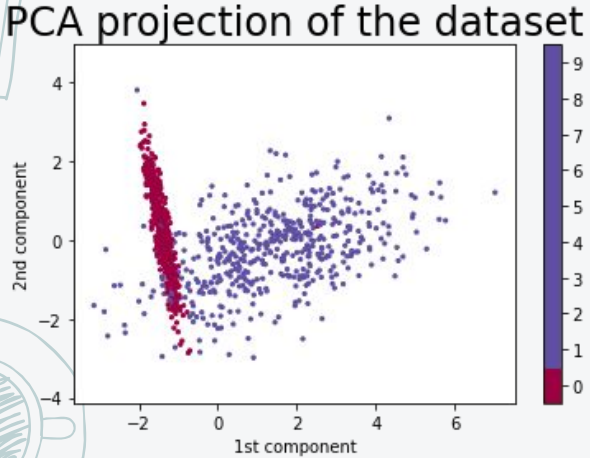


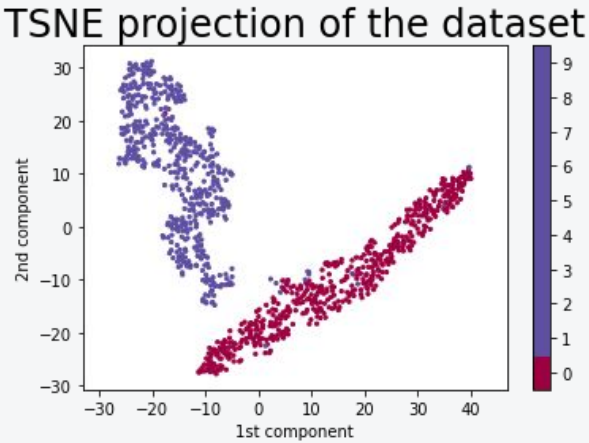
neighbors = 5
metric = mahalanobis
min_dist = 0.1

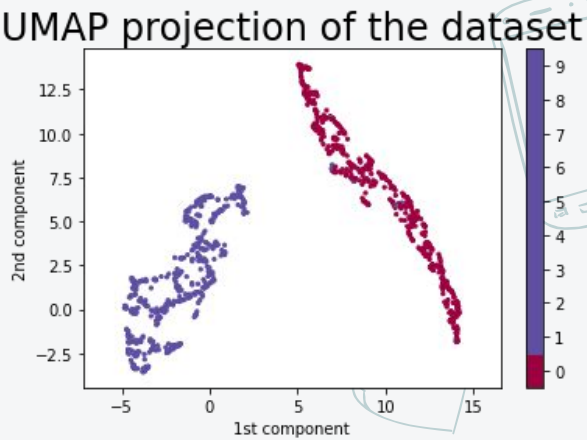


https://umap-learn.readthedocs.io/en/latest/parameters.html - on all parameters and their effects

UMAP ON THE ORIGINAL DATA







UMAP

Advantages

- Preserves global as well as local structure
- Has mathematical guarantees
- More robust to hyperparameter changes (e.g. number of neighbors)

Disadvantages

- Depends on hyperparameters
- Slow-ish

Reading

PCA

- In depth PCA with examples from the Python handbook: https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html

t-SNE

- Playing with t-sne: https://distill.pub/2016/misread-tsne/
- Intro to t-sne with python: https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

UMAP

- Talk (26min): https://www.youtube.com/watch?v=nq6iPZVUxZU
- Examples with code in python: https://umap-learn.readthedocs.io/en/latest/basic_usage.html

t-SNE vs UMAP:
https://towardsdatascience.com/tsne-vs-umap-global-structure-4d8045acba17