Machine Learning I

Lecture 17: Nonlinear Dimensional Reduction

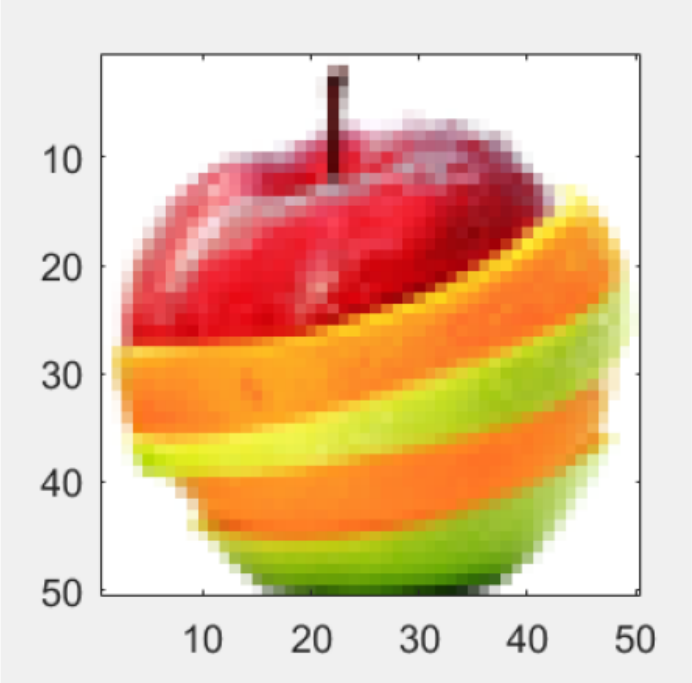
Nathaniel Bade

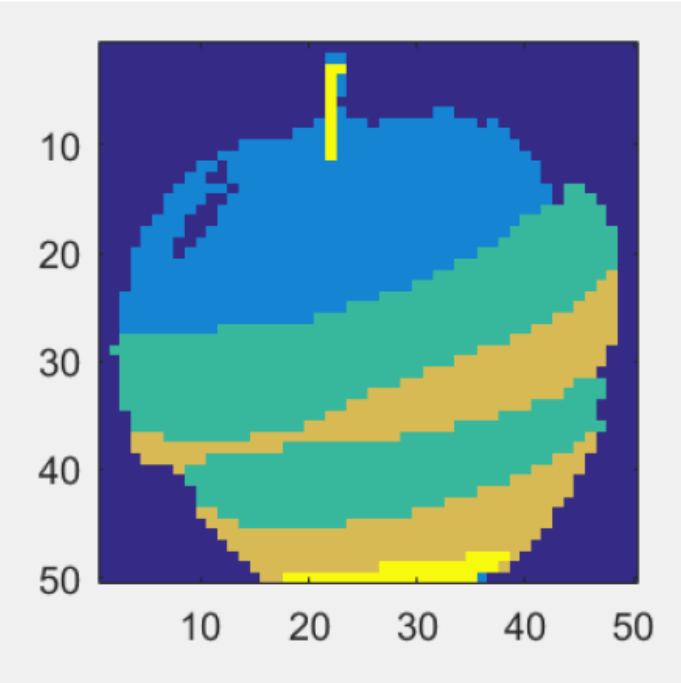
Northeastern University Department of Mathematics

Table of contents

Spectral Clustering

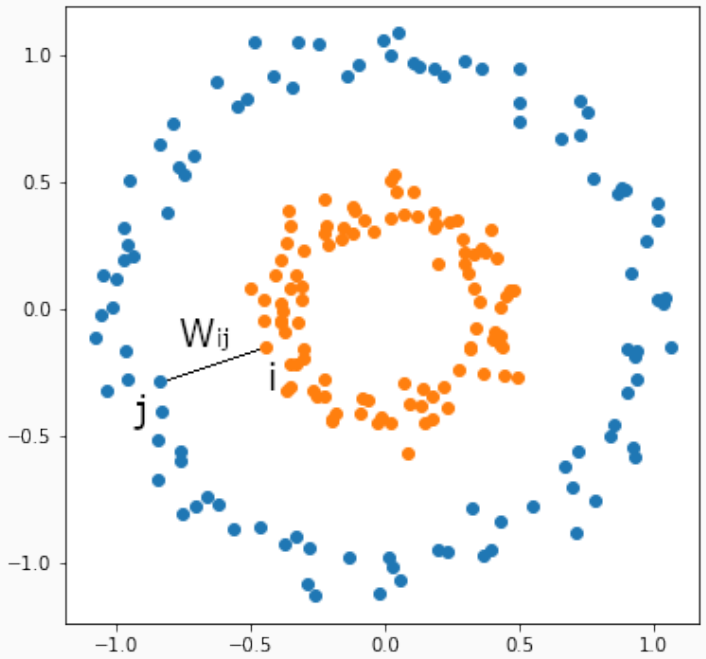
Spectral Clustering

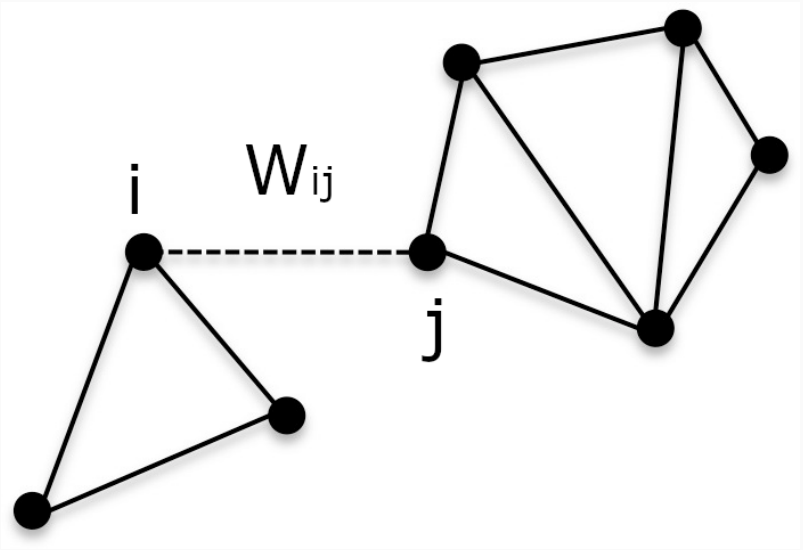




Spectral clustering uses graph theory methods to detect strings of locally close points. Much like DBSCAN and single-linkage hierarchical clustering, spectral clustering is an algorithm that finds clusters by considering only local neighborhoods around a point. Unlike those method however, it is somewhat more statistically stable under resampling.

Similarity Graphs

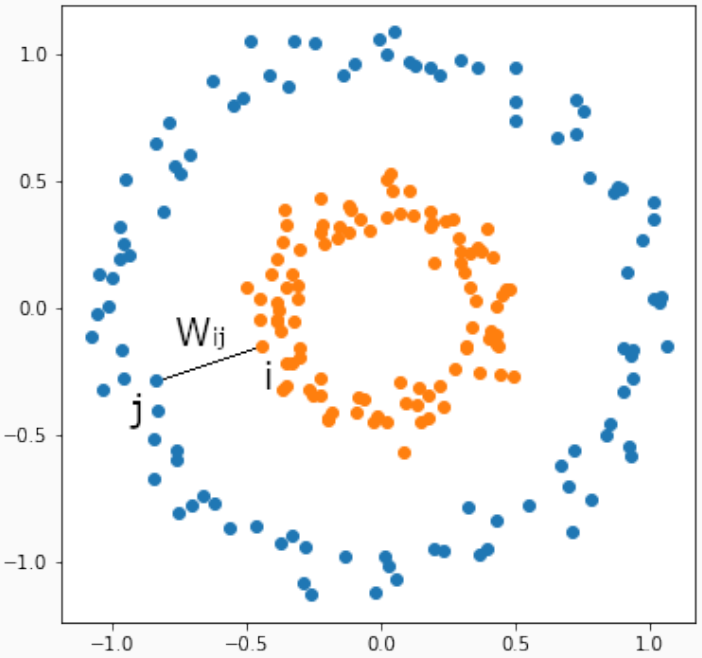


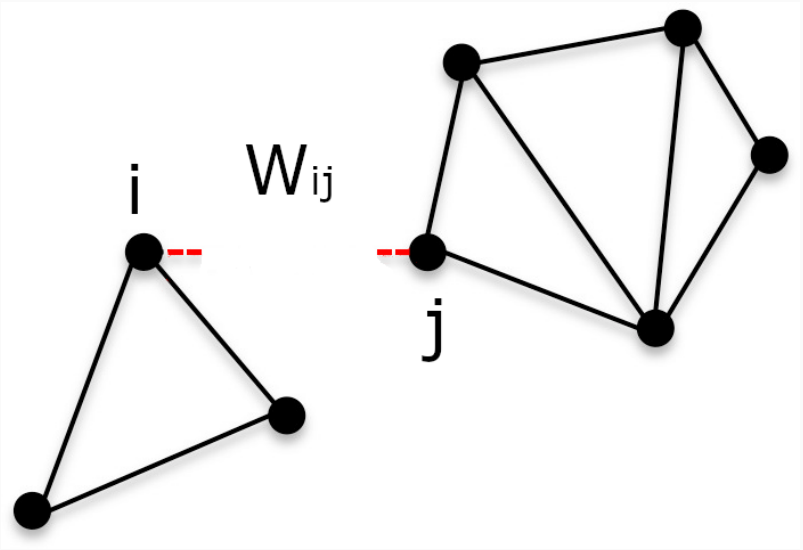


The starting point is an $N \times N$ matrix of pairwise **similarities** or **weights** $W_{ii'} \geq 0$ between observed points. This matrix allows us to form the **similarity graph**.

The similarity graph may be dense if all weights are included, or sparse if not all connections are included.

Similarity Graphs

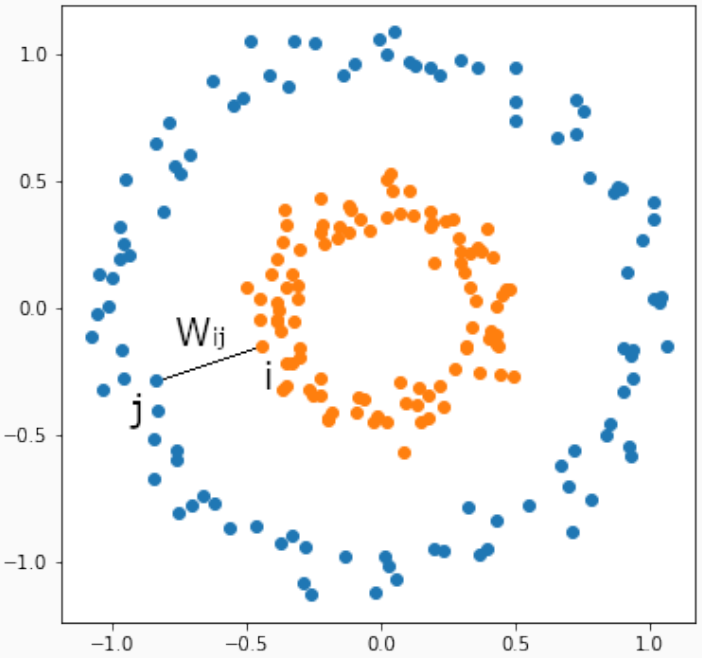


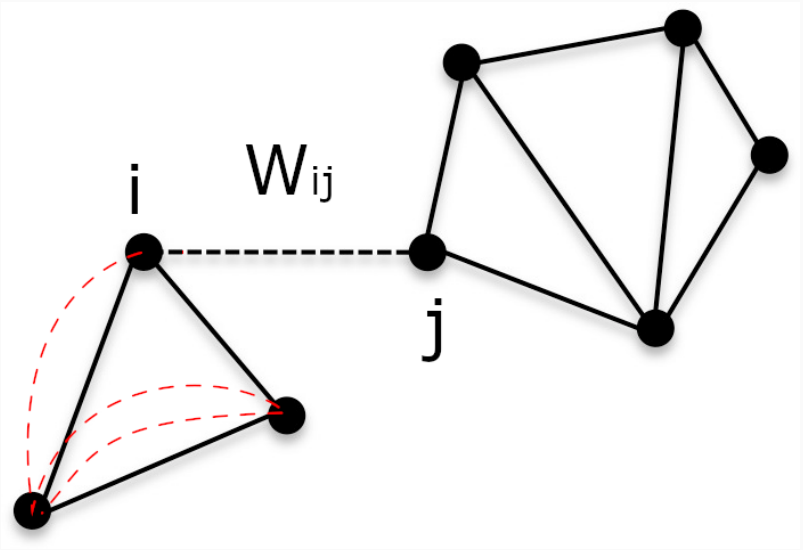


There are three standard (roughly equivalent) ways to think forming clusters from a similarity graph:

Graph partitioning by cutting lowest weight connections.

Similarity Graphs



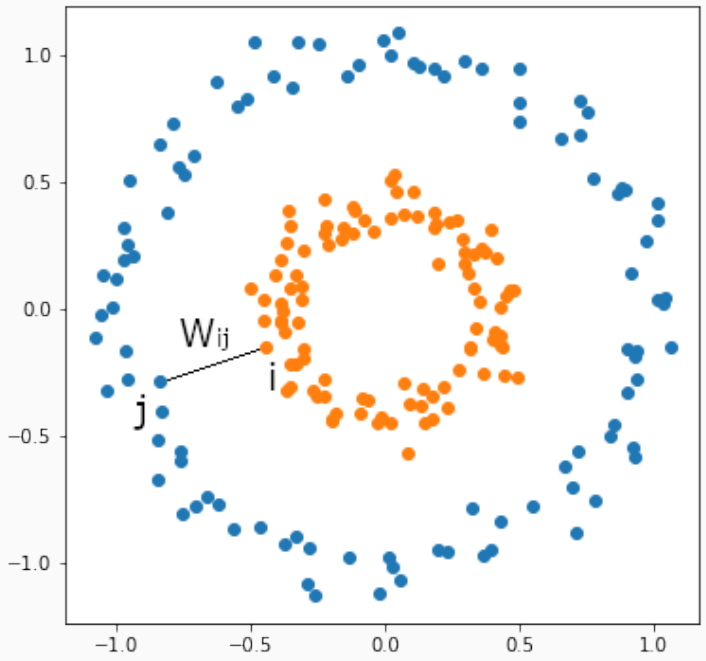


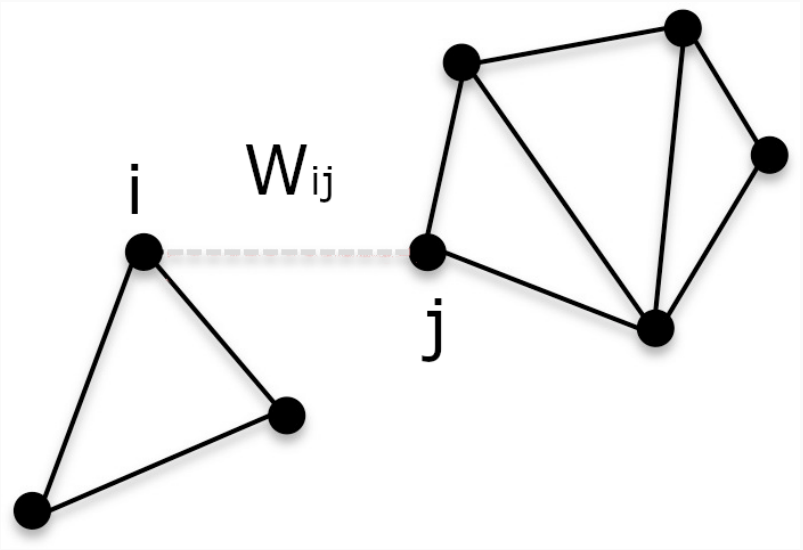
There are three standard (roughly equivalent) ways to think forming clusters from a similarity graph:

Graph partitioning by cutting lowest weight connections.

Random walk with transition threshold T.

Similarity Graphs





There are three standard (roughly equivalent) ways to think forming clusters from a similarity graph:

- Graph partitioning by cutting lowest weight connections.
- Random walk with transition threshold T.
- Perturbing away from the case of an ideal separation where 0 wights separate true connected clusters.

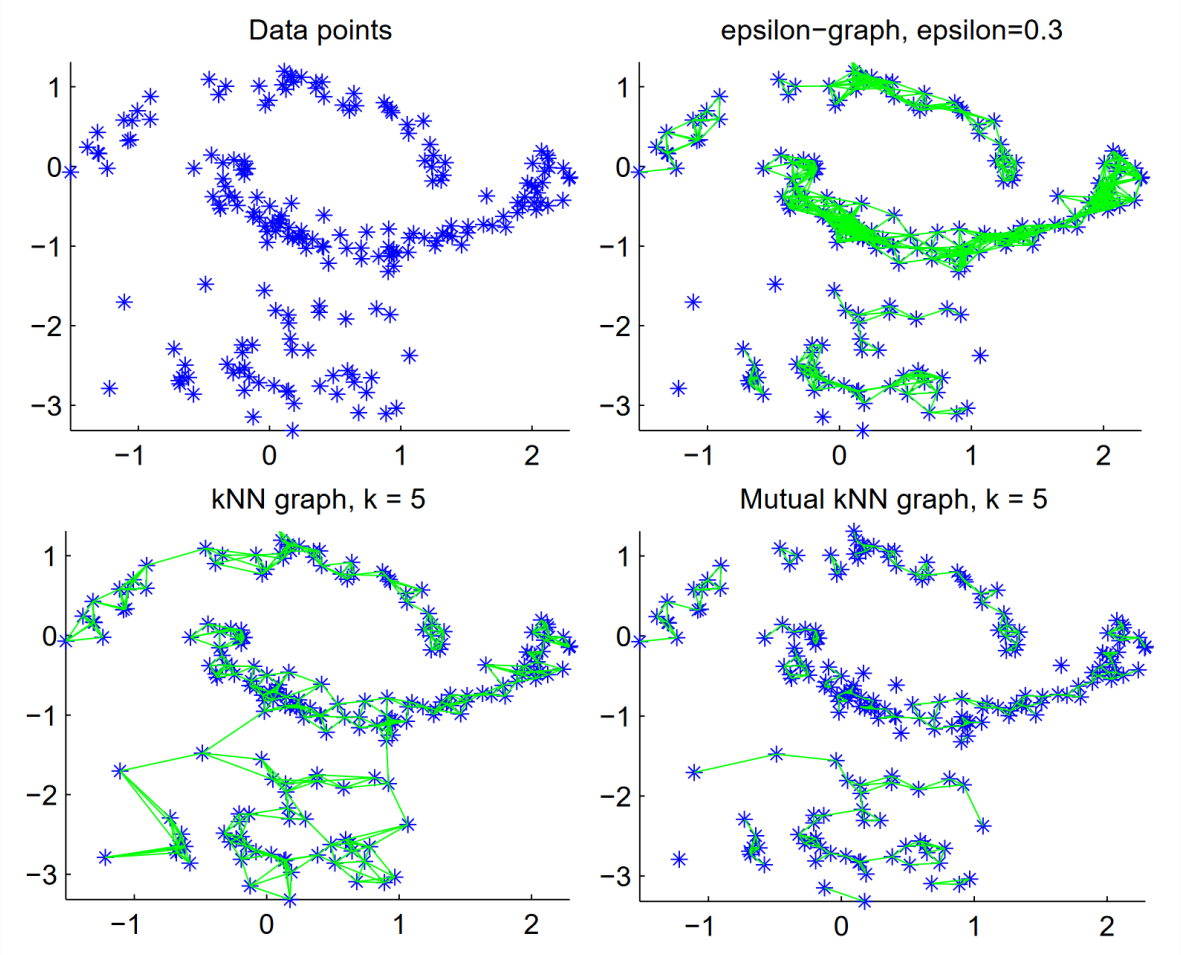
Sparse Graphs

In addition, there are three ways to "sparsify" the graph if we want to trim some weights:

> K-neighbors: Only the K closest points $x_{i'}$ to each point x_i are included in the graph. Edges are weighted by $W_{ii'}$ as before.
>
> K-mutual neighbors: A weight is only nonzero if $x_{i'}$ is one of the K points closest to x_i and $x_{i'}$ is one of the K closest points to $x_{i'}$. This enforces symmetry on the graph. Edges are weighted by $W_{ii'}$ as before.
>
> ϵ-neighborhood: Weights of all points outside an ϵ ball of x_i are set to 0. In this case, weights may be computed as before, but it is also common to set all nonzero weights to 1.

Of course, we can also consider the fully connected graph.

Sparse Graphs



In this example, we see the difference between K-NN, mutual K-NN and neighborhood fitting. Note that the latter two can classify outliers. `https://www.cs.cmu.edu/~aarti/Class/10701/readings/Luxburg06_TR.pdf`

Graph Cut Loss

We will enter spectral clustering through cutting. Consider the partition $C = (C_1, \ldots, C_K)$. We would like to find a cut that minimizes

$$\text{Cut}(C) = \sum_{i=1}^{K} \sum_{\substack{i \in C_k \\ i' \notin C_k}} W_{ii'} .$$

There are two problem: This NP hard, and (at least for $K = 2$) minimizing $\text{Cut}(C)$ often just peals off a single point into one cluster.

Instead, we will solve the normalized **Ratio cut** problem:

$$\text{RCut}(C) = \sum_{i=1}^{K} \frac{1}{|C_i|} \sum_{\substack{i \in C_k \\ i' \notin C_k}} W_{ii'} .$$

Ratiocut

Define the **graph Laplacian** to be the $N \times N$ matrix $L = D - W$, where D is the diagonal matrix $D_{ii} = \sum_{i'=1}^{N} W_{ii'}$.

Let

$$h_k = \frac{1}{\sqrt{|C_k|}} \mathbb{1}_{C_k}$$

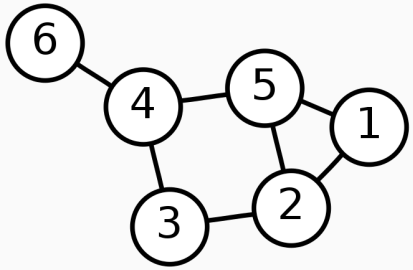
be the N-vector that is $\frac{1}{\sqrt{|C_k|}}$ in the i'th position if $x_i \in C_k$ and 0 otherwise. Let $\mathbf{H}$ be the matrix whose columns are h_i.

Lemma. Ratiocut can be written

$$\mathrm{RCut}(C) = \mathrm{Tr}(\mathbf{H}^T L \mathbf{H}).$$

Graph Laplacian Example

Graph



Laplacian matrix

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

Degree matrix

$$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Adjacency matrix

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Ratiocut as an Eigenvalue Problem

Lemma. Ratiocut can be written

$$\mathrm{RCut}(C) = \mathrm{Tr}(\mathbf{H}^T L \mathbf{H})\,.$$

Proof: For any vector v,

$$\begin{aligned} v^T L v = v^T (D - W) v &= \frac{1}{2} \sum_i D_{ii} v_i^2 - \sum_{i,i'} W_{ii'} v_i v_{i'} + \frac{1}{2} \sum_{i'} D_{i'i'} v_{i'}^2 \\ &= \frac{1}{2} \sum_{i,i'} W_{ii'} (v_i^2 - 2 v_i v_{i'} + v_{i'}^2)\,, \\ &= \frac{1}{2} \sum_{i,i'} W_{ii'} (v_i - v_{i'})^2\,. \end{aligned}$$

Recall in the above that $D_{ii} = \sum_{i'=1}^{N} W_{ii'}$.

Ratiocut as an Eigenvalue Problem

Applying

$$v^T L v = \frac{1}{2} \sum_{i,i'} W_{ii'} (v_i - v_{i'})^2 ,$$

to h_k and noting that $(h_{ki} - h_{ki'})^2$ is only nonzero if $i \in C_k$ and $i' \notin C_k$ we have

$$h_k^T L h_k = \frac{1}{|C_k|} \sum_{\substack{i \in C_k \\ i' \notin C_k}} W_{ii'} .$$

Since

$$\mathrm{Tf}(\mathbf{H}^T L \mathbf{H}) = \sum_{k=1}^{K} h_k^T L h_k = \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{\substack{i \in C_k \\ i' \notin C_k}} W_{ii'} ,$$

we have shown

$$\mathrm{RCut}(C) = \mathrm{Tr}(\mathbf{H}^T L \mathbf{H}) .$$

□

Eigenvalues of Laplacian

Now, since

$$\mathrm{RCut}(C) = \mathrm{Tr}(\mathbf{H}^T L \mathbf{H}),$$

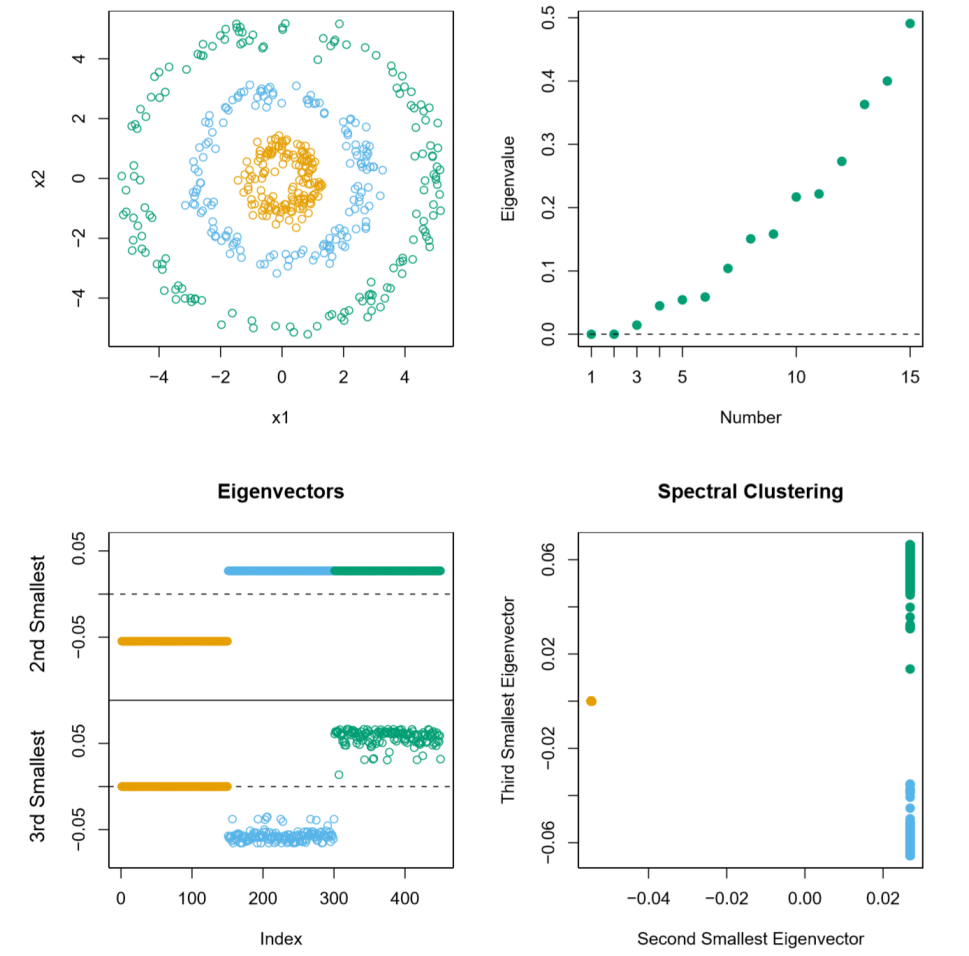
we know how to minimize $\mathrm{RCut}(C)$: the solution that extremize $h_k^T L h_k$ are the eigenvectors. Therefore, $\mathrm{Tr}(\mathbf{H}^T L \mathbf{H})$ will be minimized by a matrix of the smallest eigenvectors of L, just as for PCA

The eigenvectors will not in general be of the form h_k for some clustering k. We can see that

$$h_k^T L h_k = \frac{1}{2} \sum_{i,i'} W_{ii'} (h_{ki} - h_{ki'})^2,$$

will be minimized when h_k is picked to map points with high adjacency near to eachother in the reduced space. Clustering there yields a fundamentally new clustering.

Eigenvalues of Laplacian



Here, we see the projection onto the first nontrivial eigenvectors of L, as well the eigenvalues plotted against eachother.

Eigenvalues of Laplacian

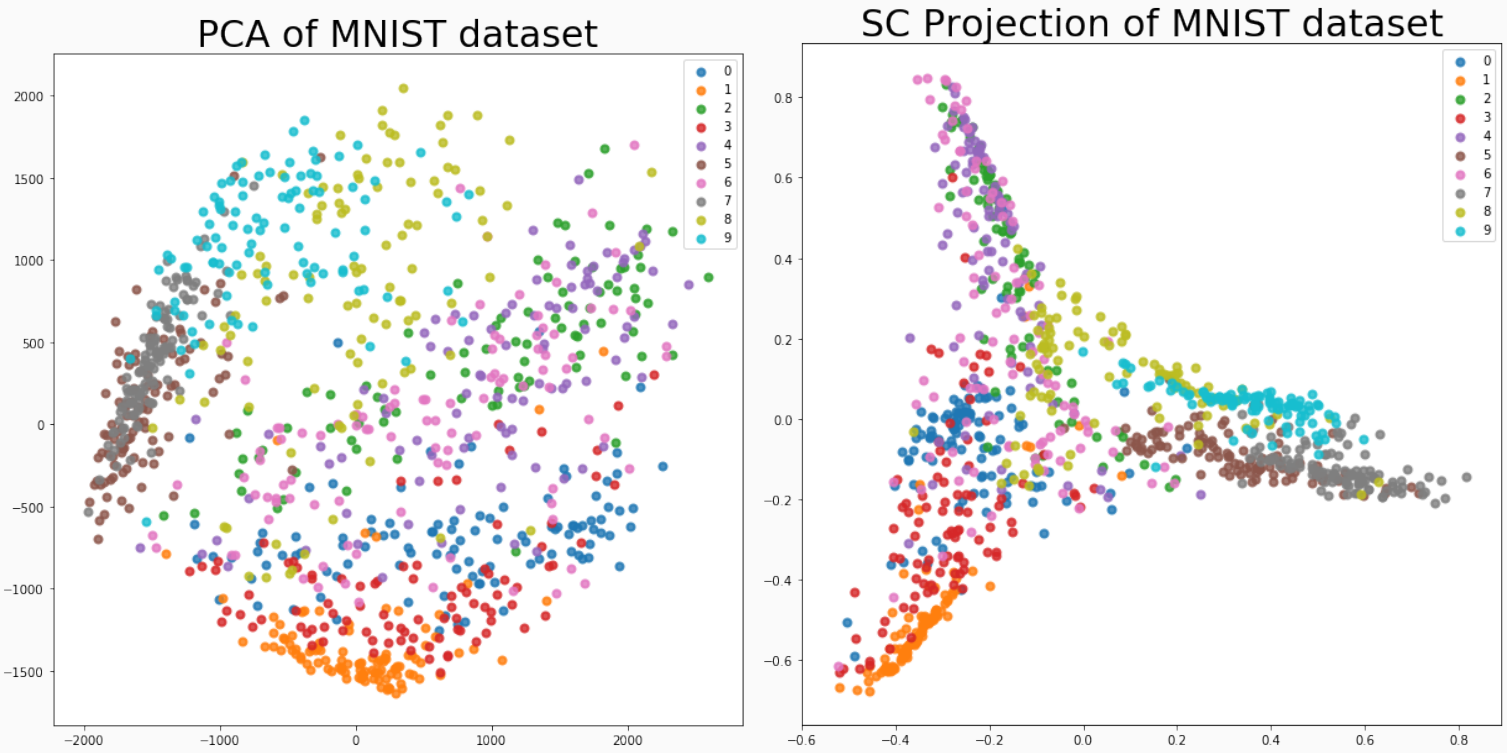
In the previous slide, we only plotted the second and third eigenvectors. Since $D_{ii} = \sum_{i'=1}^{N} W_{ii'}$, the vector of 1s is always an eigenvector of L:

$$1^T L 1 = 1^T D 1 - 1^T W 1 = \sum_i \sum_{i'} W_{ii'} - \sum_{ii'} W_{ii'} = 0$$

One can show that if the graph is connected it is the only zero eigenvalue.

With a bit of work, one can show that if a graph has K connected components, L can be arranged into a block diagonal form, with blocks of connected subgraphs. L then has exactly K zero eigenvectors.

Example: PCA vs Spectral Projection



We compare the principle component projection from the MNIST dataset to the spectral projection. We see a similar structure between the two, although one set is not convex while the other is.

Summary

We should mention the advantages and disadvantages of spectral clustering:

- Works well on hard, nonconvex problems.
- Obtain low dimensional representations as part of process.
- Empirically very successful.
- Flexibly definable for a wide variety of dissimilarity measures.

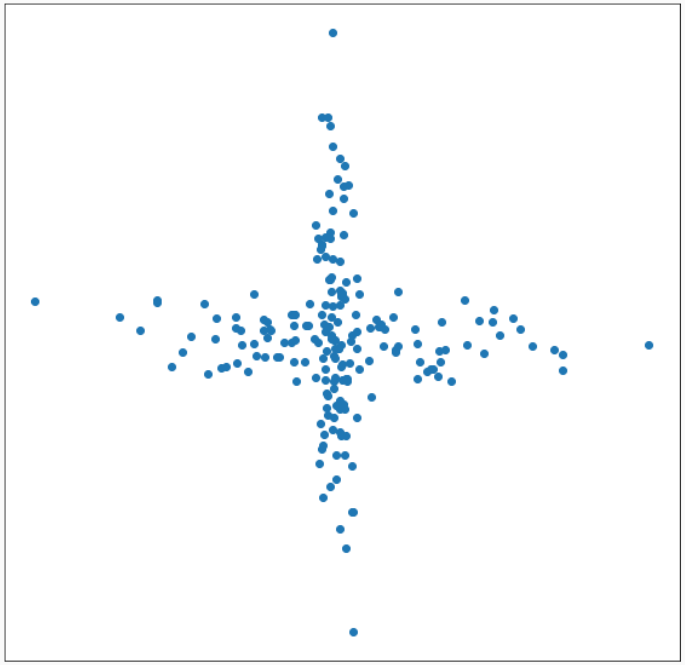
Summary

We should mention the disadvantages of spectral clustering:

- We must choose K.
- We must choose a similarity measure, and Gaussian similarity may not be the best.
- Can be very computationally expensive on large datasets.
- Low dimensional feature space doesn't have an intuitive meaning.

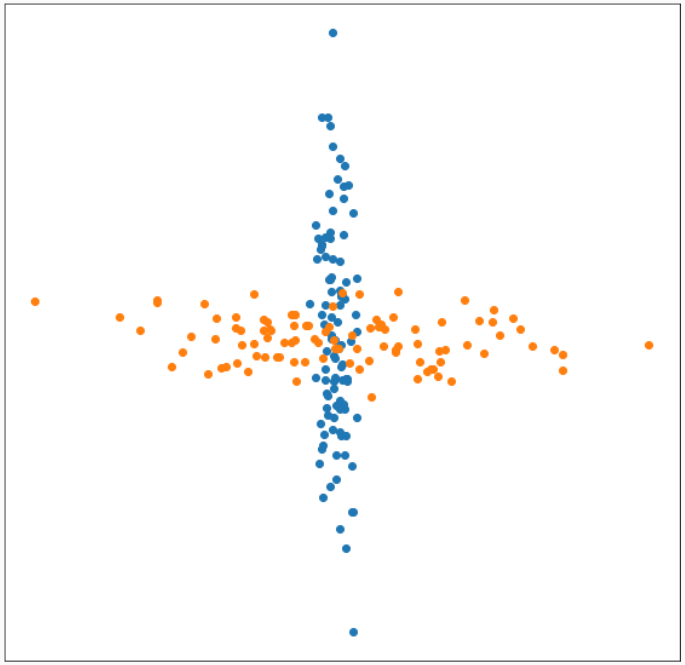
Probabilistic Cluster Analysis: Gaussian Mixtures

Probabilistic Cluster Analysis



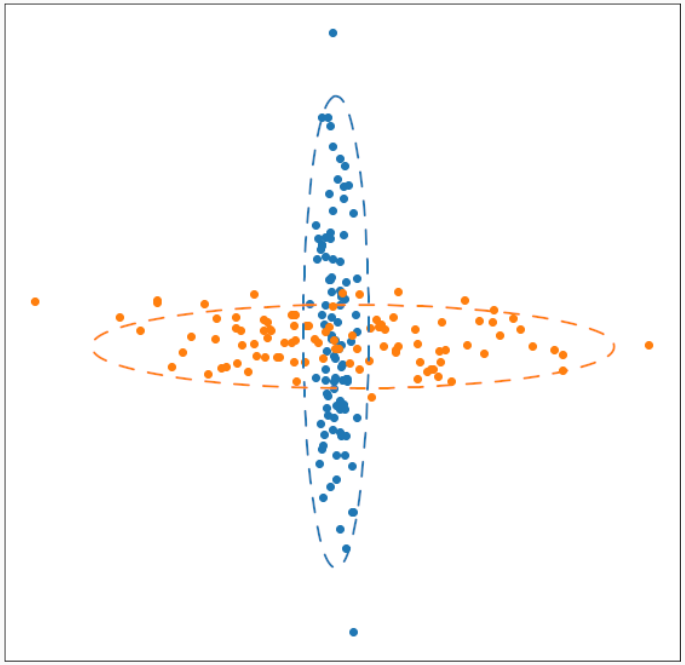
Many times clusters fundamentally overlap, with individual data points not clearly belonging to one cluster or the other even though distinct clusters still clearly exist. Even in high dimensional datasets we might not expect our features to completely separate the data.

Mixture Models



In such cases we should allow ourselves to fit **mixture models** to the data. Mixture models are a form of additive modeling, and we are constrained as usual by the complexity of the constructed features of our assumed distributions.

Mixture Models



In a **Gaussian mixture model** we try to fit Gaussian distributions to K clusters. That is, we assume each label is distributed as a Gaussian:

$$x_i | z_i = k \sim \text{Norm}(\mu_k, \Sigma_k)\,.$$

The probability a point x_i belongs to cluster k is then $p_{ik} = p(z_i = k | x_i)$, where $\sum_{k=1}^{K} p_{ik} = 1\,.$

Mixture Models and Expectation Maximization

Formally, would like to find a probability function

$$p(x|\theta) = \sum_{k=1}^{K} \phi_k \mathcal{N}(x|\mu_k, \Sigma_k)\,,$$

where μ_k denotes the Gaussians mean, and Σ_k denotes it's covariance matrix:

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^K |\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)\,.$$

The $\theta = \{\mu_k, \Sigma_k, \phi_k\}$ collect all parameters, and $\sum \phi_k = 1$.

Mixture Models and Expectation Maximization

Given a data set $\mathcal{T}$, we want to maximize the likelihood that $\mathcal{T}$ was drawn (i.i.d.) from $p(x|\theta)$. The likelihood of getting the data we observe is

$$L(\mathcal{T}, \theta) = \prod_{i=1}^{N} p(x_i|\theta)\,.$$

Our goal then is to find θ that maximize $L(\mathcal{T}, \theta)$. The **expectation maximization** algorithm is a popular tool for solving MLE problems. The EM algorithm attempts to maximize the log likliehood

$$\ell(\mathcal{T}, \theta) = \log L(\mathcal{T}, \theta) = \sum_{i=1}^{N} \log p(x_i|\theta)\,.$$

Unlike for linear methods, direct maximization in this case involves derivatives of a log of a sum which are hard to find an analytic solution for.

Mixture Models and Expectation Maximization

Expectation Maximization uses an iterative method that lowers the loss at each step. The algorithm proceeds in two steps:

> **Expectation Step:** For a set of parameters $\theta^{(i)}$, calculate exceptions p_{ik} of the component assignment for each datapoint x_i.
>
> **Maximization Step:** Then, maximize the expectations calculated the previous step with respect to the model parameters.

Lets run through it.

Mixture Models and Expectation Maximization

Initialization: Randomly assign training samples to clusters C_k, set μ_k, Σ_k to the in cluster sample mean and variance respectively, and set $\phi = \frac{1}{K}$.

Expectation Step: Calculate the probability x_i is generated by component C_k:

$$p_{ik} = \frac{\phi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \phi_j \mathcal{N}(x|\mu_j, \Sigma_j)} .$$

Maximization Step: Using p_{ik} we can calculate the parameters that would maximize the expectation as follows: The number per cluster is approximately

$$\hat{N}_k = \sum_{i=1}^{N} p_{ik} \,, \qquad \text{so} \qquad \hat{\phi} = \frac{\hat{N}_k}{N} \,.$$

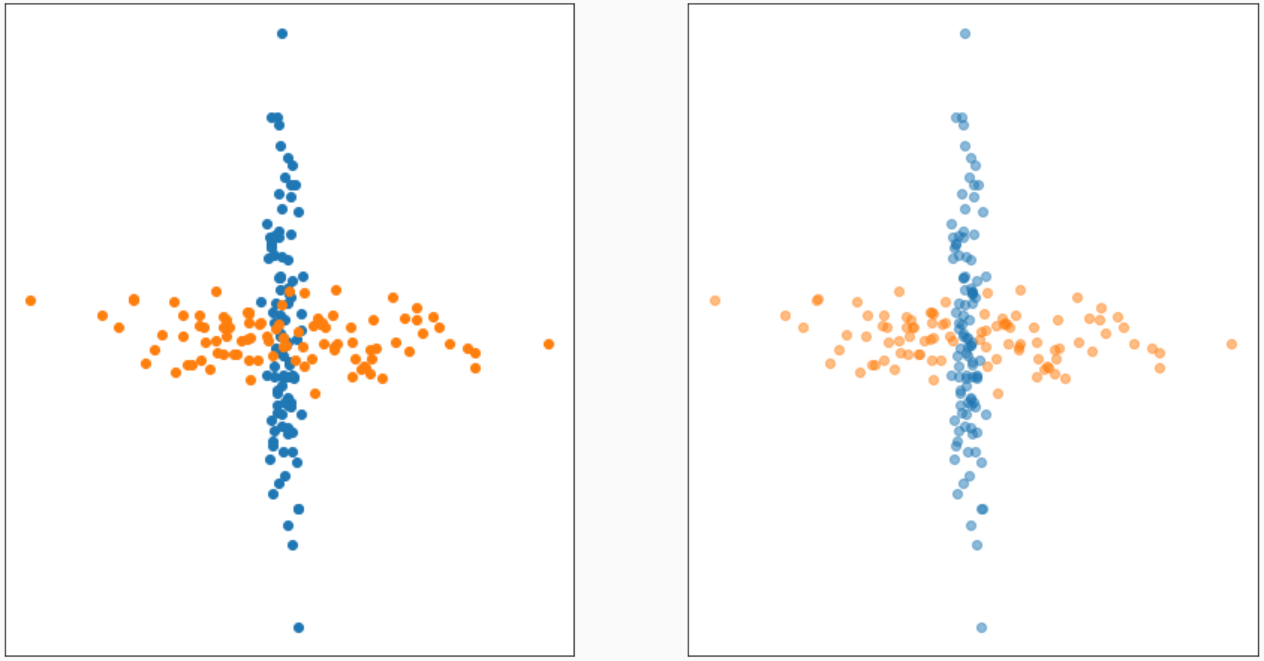
The mean can be computed as the weighted centroid

$$\hat{\mu}_k = \frac{1}{\hat{N}_k} \sum_{i=1}^{N} p_{ik} x_i \,,$$

and the variance is the weighted covariance

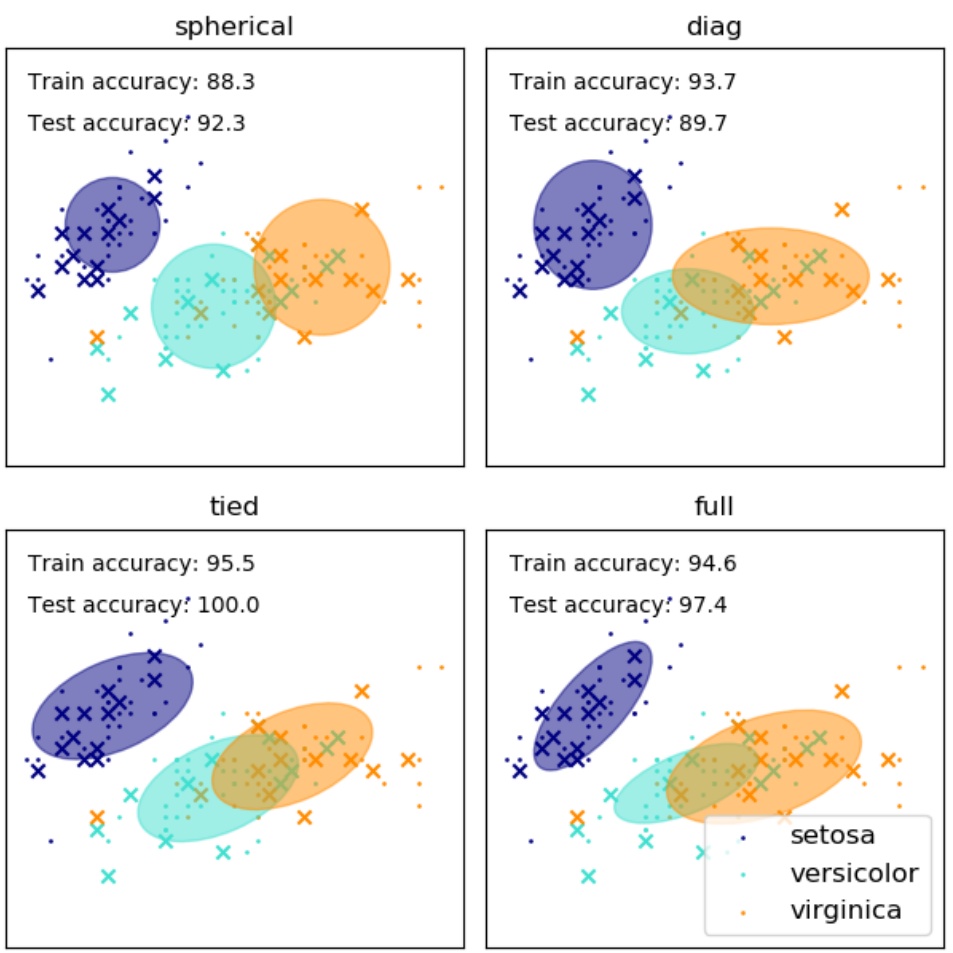
$$\hat{\Sigma}_k = \frac{1}{\hat{N}_k} \sum_{i=1}^{N} p_{ik} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T \,.$$

Mixture Models



Compare the actual labeling on the left with the Guassian labeling on the right. While all the points along the strip are classified as being a part of Cluster 1 this is probably just because Cluster 1 has lower variance, in fact the probabilities are around .5 along the intersecting strip.

Mixture Models



We can see several fittings of the Iris data set using differently shaped clusters. Here, the dots are the training points and the crosses are the test points.

t-Stochastic Neighbor Embedding

Stochastic Neighbor Embedding

Stochastic Neighbor Embedding (SNE) is an amalgamation of probabilistic methods, dimensional reduction, and clustering. SNE tries to convert high dimensional Euclidean data into the conditional probabilities $p_{j|i}$ of whether x_i would pick x_j as its neighbor. Mathematically,

$$p_{j|i} = \frac{\exp\left(-||x_i - x_j||^2/2\sigma_i^2\right)}{\sum_{k\neq i}\exp\left(-||x_i - x_k||^2/2\sigma_i^2\right)},$$

where σ_i is a variance centered at x_i. Our goal then is to find a low dimensional representation y_i that preserves these probabilities, setting the variance to $1/\sqrt{2}$:

$$q_{j|i} = \frac{\exp\left(-||y_i - y_j||^2\right)}{\sum_{k\neq i}\exp\left(-||y_i - y_k||^2\right)}.$$

A natural candidate for the cost is

$$C = \sum_{i,j=1}^{N} p_{j|i}\log\frac{p_{j|i}}{q_{j|i}}.$$

Stochastic Neighbor Embedding

The cost

$$C = \sum_{i,j=1}^{N} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} .$$

can be minimized using gradient decent

$$\frac{\partial C}{\partial y_i} = 2 \sum_{j=1}^{N} (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j) .$$

The authors understand this gradient as a "springing force" connecting y_i to each other point, where the mismatch $(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})$ represents the push or pull.

It turns out this gradient is hard to compute, and even with a severe penalty to the cost function results in overcrowding as all point are pulled towards each other.

t-Stochastic Neighbor Embedding

t-SNEs differs from SNE in a few important ways:

- t-SNE is to works with the symmetrized probabilities $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$. This greatly simplifies the computation of the gradient.
- As we see in the equation for p_{ij} above, t-SNE also provides a minimum probability so that even outliers have some effect on the organization.
- In the dimensionally reduced space, t-SNE uses the student-t distribution with 1 degree of freedom instead of the Gaussian.
- Finally, q_{ij} is always greater than $\frac{2\rho}{n(n-1)}$. This means that for far apart points $q_{ij} > p_{ij}$ creating a slight repulsion.

t-Stochastic Neighbor Embedding

The result is that

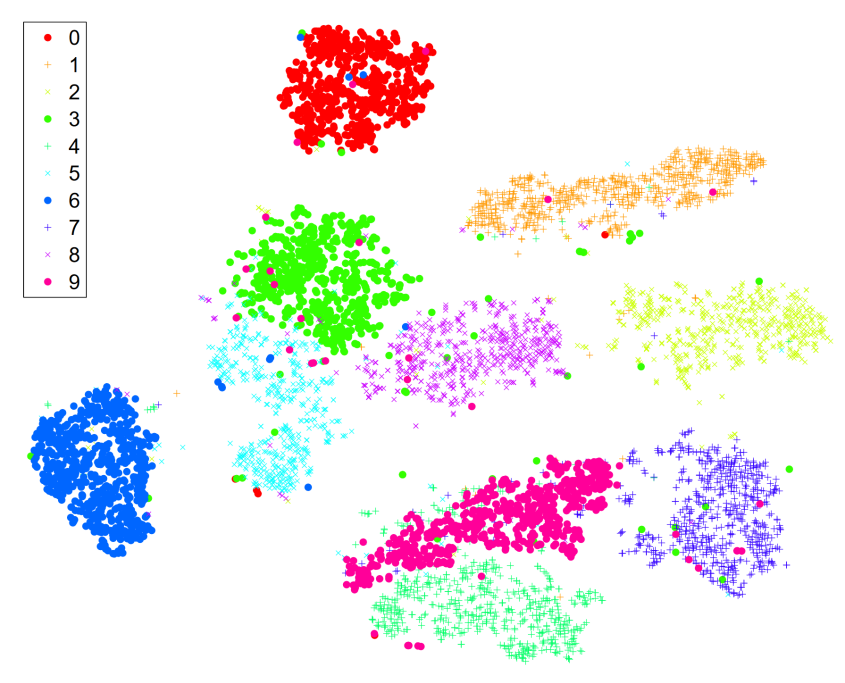
$$q_{ij} = \frac{1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq j}(1 + ||y_i - y_j||^2}$$

and

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i - y_j||^2)\,.$$

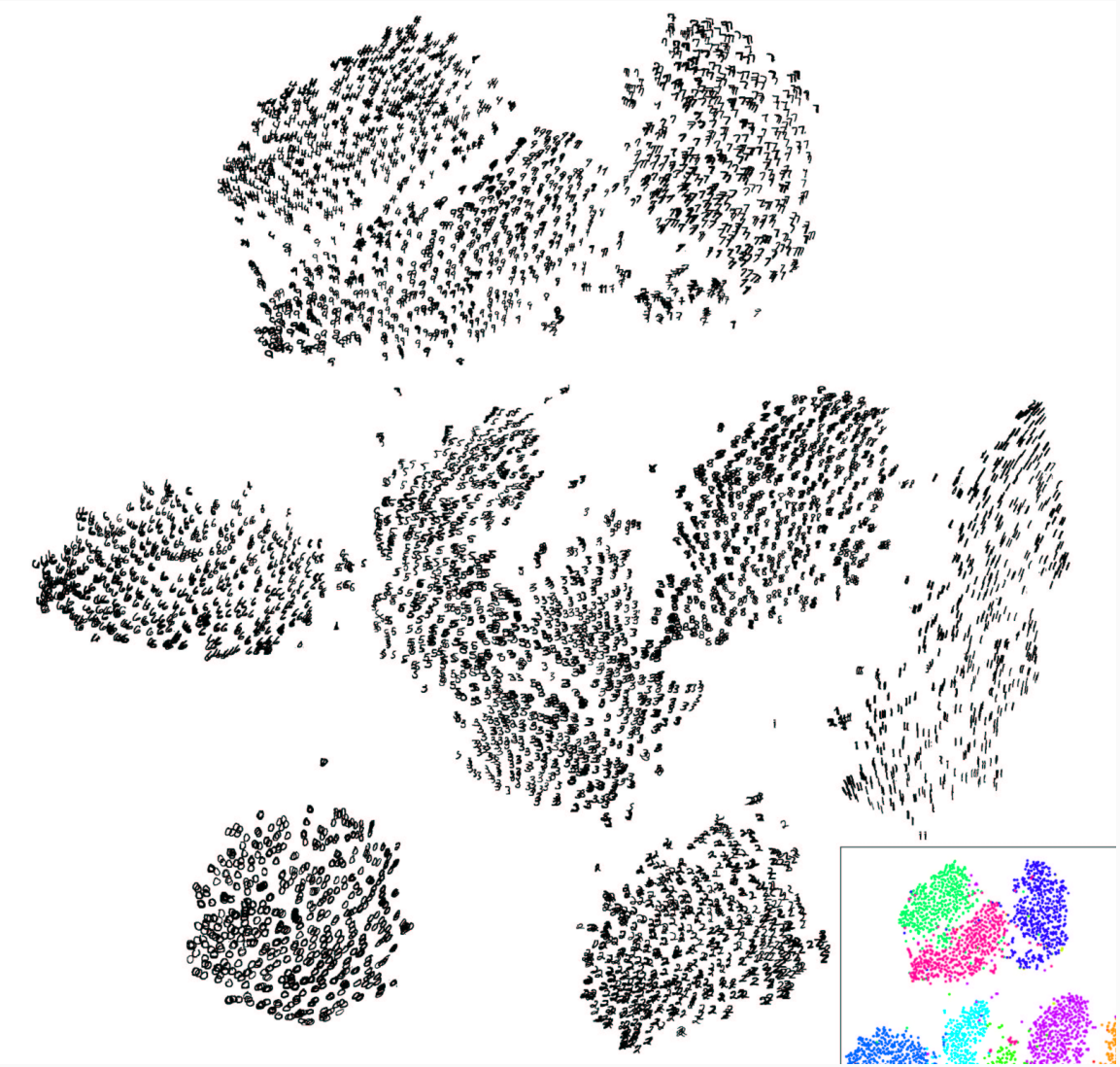
Fitting is then done by gradient decent, using compression and exaggeration to force the points closer together and make the probabilities look larger.

t-Stochastic Neighbor Embedding



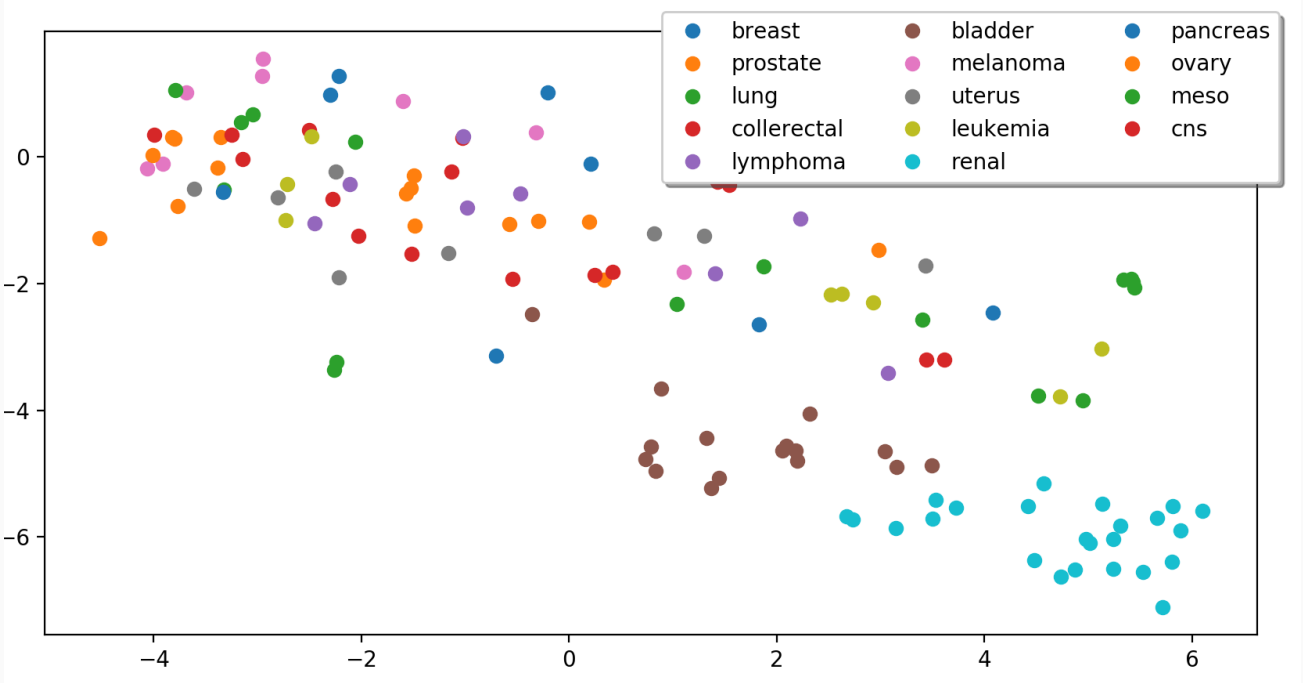
tSNE performs very well on image datasets, even ones with a large number of labels. Although the axes have no intrinsic mean in terms of the feature set, we can try to give meaning to the dimensions by analyzing the organization.

tSNE for MNIST



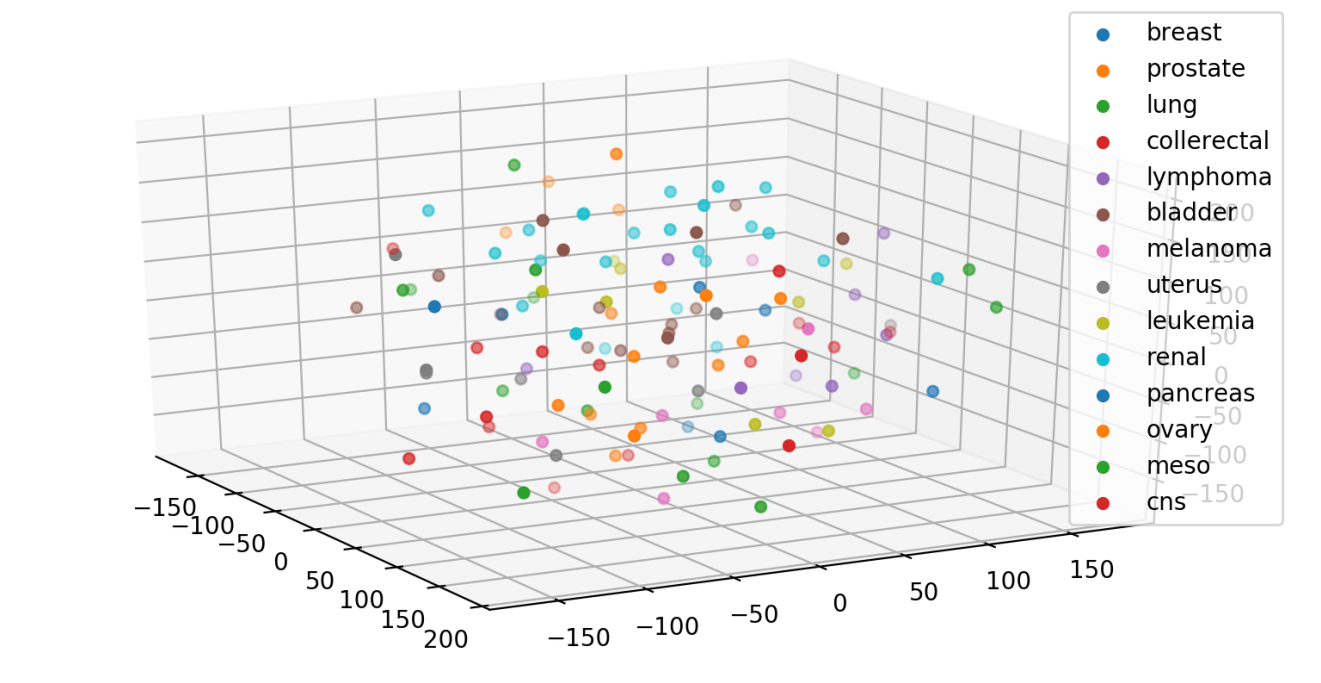
tSNE for MNIST

t-Stochastic Neighbor Embedding



It is an open question as to how well tSNE works on non-image datasets. For example, the cancer microarray dataset is displayed above. While we do see some clustering (and indeed some clusters that we hadn't seen before) the majority of the projection is noise.

t-Stochastic Neighbor Embedding



It is an open question as to how well tSNE works on non-image datasets. For example, the cancer microarray dataset is displayed above. While we do see some clustering (and indeed some clusters that we hadn't seen before) the majority of the projection is noise. The projection to 3d is even worse.

t-Stochastic Neighbor Embedding

tSNE lives int he family of **manifold learning algorithms.** Other algorithms include

Sammon Mapping: Similar to SNE, but relative distance is directly minimized.

Multidimensional Scaling: Project into low dimensions maximally preserving between point distance.

Isomap: An extension of MDS that include the geodesic distance computed from the nearest neighbor graph.

Locally Linear Embedding: LLE tries to embed the nearest neighbor graph directly into the lower dimensional space.

Autoencoders and GANs: Neural networks that try to reproduce their own input after squeezing through a layer with only two or three nodes. The representation in those nodes is the low dimensional representation.

Reference

Spectral Clustering: https://www.cs.cmu.edu/~aarti/Class/10701/readings/Luxburg06_TR.pdf

Gaussian Clustering: http://www.inference.org.uk/mackay/itprnn/ps/300.318.pdf

A decent survey of dimensional reduction techniques: https://arxiv.org/ftp/arxiv/papers/1403/1403.2877.pdf

tSNE: http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf