SLT coding exercise #1

Locally Linear Embedding

https://gitlab.vis.ethz.ch/vwegmayr/slt-coding-exercises

Due on Monday, March 6th, 2017

Michael Schürch

13-916-614

Contents

The Model

The LLE algorithm is a method for dimensionality reduction in an unsupervised manner. Thus for some data matrix $\mathcal{X} \in \mathbf{R}^{nxD}$ with n data points of size D we seek a new matrix $\mathcal{Y} \in \mathbf{R}^{nxd}$ with $d < D$ thus reducing the dimension of each data point.

The objective of LLE is to minimize the following reconstruction error:

$$\mathcal{E}(W) = \sum_i \left| X_i - \sum_j W_{ij} X_j \right|^2$$

where we have the constraint that $\sum_j W_{ij} = 1$ and $W_{ij} = 0$ if X_j isn't one of the K closest neighbors of X_i.

In a final step we use found W and map every high dimensional observation X_i to a low dimensional vector Y_i. This is done by minimizing the embedding cost function:

$$\Phi(Y) = \sum_i \left| Y_i - \sum_j W_{ij} Y_j \right|^2 = \sum_{ij} M_{ij} (Y_i \cdot Y_j)$$

with the constraint that $\sum_i Y_i = 0$ and $\frac{1}{N} \sum_i Y_i Y_i^T = I$

This information was gathered from https://www.cs.nyu.edu/ roweis/lle/papers/lleintro.pdf.

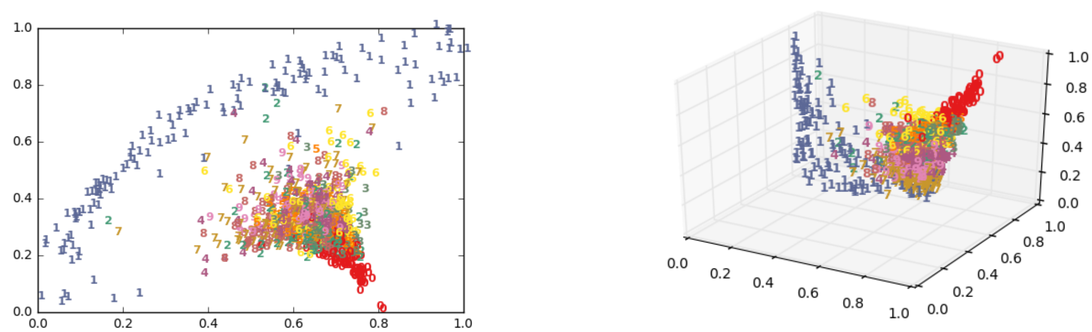
The Questions - solved

(a) Get the data

I got the data from http://yann.lecun.com/exdb/mnist/ and uncompressed it into the folder ./data.

(b) Locally linear embedding

For the 2D embedding space we can observe that the numbers 1 and 0 form a cluster and all other numbers seem to be scrambled into a third cluster in the middle. The 3D embedding space additionally manages to distinguish between numbers 6,2 and 7,4.



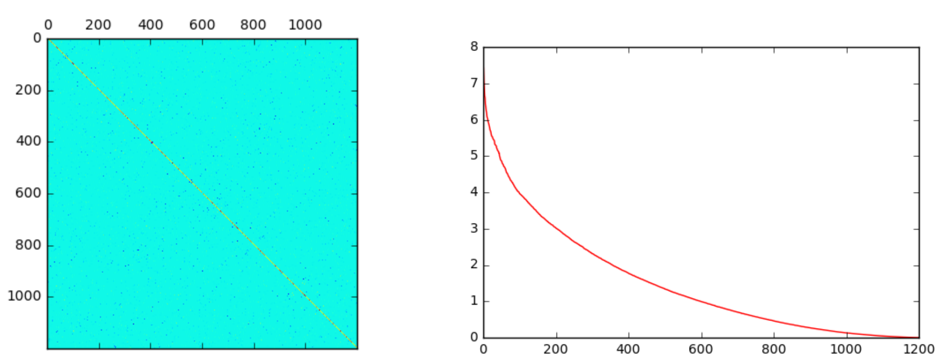
(a) 2D LLE plot

(b) 3D LLE plot

Figure 1

(c) Cluster structure

The matrix M, see Fig. 2a, is very sparse. It also has most of its non-zero entries on its diagonal and is symmetric. But I cannot see any form of a block structure. The singular values of M can be seen in Fig. 2b. As of determining the optimal embedding dimension we can basically read off the reconstruction error by summing up the last d eigenvalues, where d is the dimension of the embedded space. Therefore we can estimate the error easily and choose a certain threshold.



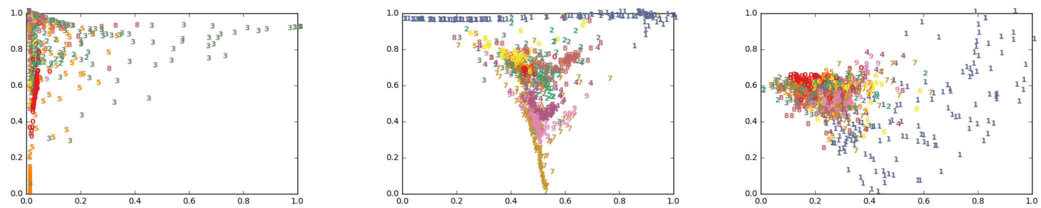
(a) The M matrix

(b) singular values of M

Figure 2

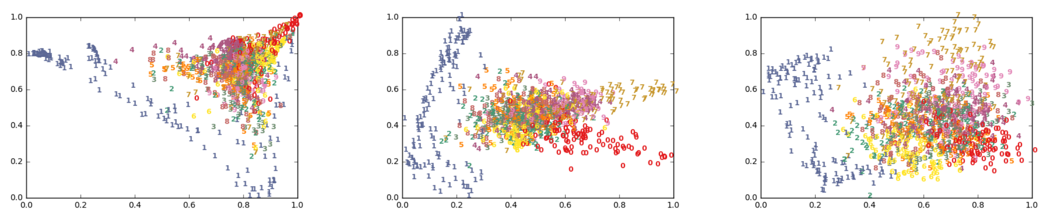
(d) Nearest Neighbors

As can be seen in the following illustrations Fig 3, 4 the choice of the number of neighbors K and the metric can have a big effect on the quality of the solution.



(a) LLE with K = 5 (b) LLE with K = 10 (c) LLE with K = 50

Figure 3: Minkovski metric



(a) LLE with K = 5 (b) LLE with K = 10 (c) LLE with K = 50

Figure 4: Chebyshev metric

(e) Linear manifold interpolation

See the Reconstruction algorithm in the section implementation for details. We can see in Fig. 5 that the reconstruction is possible for some numbers like 0 and can be hard for numbers like 7. Interpolating between these two numbers in the embedded space also translates to an interpolation in the original space.

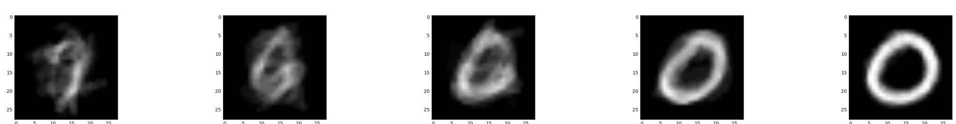


Figure 5: Reconstruction of embedded vectors between 7 and 0 (left to right)

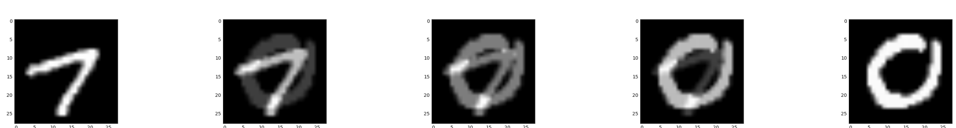


Figure 6: Interpolation in original space from 7 to 0

The Implementation

13-916-614/1_locally_linear_embedding

The implementation can be seen in src/code.ipynb. I mostly used the LocallyLinearEmbedding implementation from scikit learn or adapted it. Further I will describe issues I encountered as I worked through the project:

1. Choices
   b) I chose K = 30 (number of neighbors), as a measure of distance the minkovski norm was used
2. Efficiency problems (task b)
   Because the LLE algorithm has a complexity of $\mathcal{O}(N^2)$ I could not use the full dataset and restricted it to 1200 data points.
3. calculation of M in c)
   M was calculated using $M = (I - W)^T(I - W)$

Reconstruction algorithm

Given original data matrix $\mathcal{X} \in \mathbf{R}^{nxD}$, embedded space $\mathcal{Y} \in \mathbf{R}^{nxd}$ and a new data point $y_{new} \in \mathbf{R}^{1xd}$ in the embedded space. We reconstruct the point in the original space by the following procedure:

1. Find the K nearest neighbors of y_{new}. Let's denote them as $NB \in \mathbf{R}^{Kxd}$
2. Solve the following using a Least Squares approach: $uNB = y_{new}$
3. For all neighbors in NB find the corresponding points $NB_{orig} \in \mathbf{R}^{KxD}$ in the original space
4. Reconstruction: $x_{new} = uNB_{orig}$

Your Page

I want cookie. - cookie monster

Your Answer 13-916-614/1_locally_linear_embedding