Technische Universität Berlin

Problem Set 1: PCA, LLE, Outlier Detection

Report Machine Learning Lab Course
Fachgebiet Maschinelles Lernen
Prof. Dr. Klaus-Robert Müller
Fakultät IV Elektrotechnik und Informatik
Technische Universität Berlin

submitted by
Budi Yanto

Instructor: Daniel Bartz
Felix Brockherde

Matrikelnummer: 308819
Email: budiyanto@mailbox.tu-berlin.de

Contents

List of Figures

Chapter 1

Implementation

This chapter explains the implementation of some algorithms that are used in this problem set. It includes: *PCA*, γ-*index* and *LLE*.

1.1 Assignment 1: PCA

In this assignment the function *pca* has to be implemented, which receives a *d x n* matrix *X* and the number of components *m* as parameters, and returns the principal components as well as the projected data points in a *m x n* matrix *Z*. The principle components should be returned as a *d x d* matrix *U* and a *1 x d* vector *D*. The vector *D* contains the principal values, sorted in descending order $(D_1 \geq D_2 ...)$, whereas the matrix *U* contains the principal directions, which corresponds to the sorted principle values.

The implemented function was tested on the test data and passed the test. Following steps are performed in the implementation of the function:

1. Substract *X* from its mean.
2. Calculate the covariance matrix from the zero-mean *X*.
3. Calculate the eigenvalues and eigenvectors from the covariance matrix.
4. Sort the eigenvalues and eigenvectors in descending order.
5. Form the feature vectors by taking only the first *m* eigenvectors.
6. Project the zero-mean *X* to the feature vectors.
7. Return the projected data points, principal directions and principal values as *Z*, *U* and *D*.

✓

15 / 15

1.2 Assignment 2: γ-Index

The task in this assignment is to implement the γ-index which can be used to detect outliers in data set. In their paper, Harmeling et al. (2006) formulate the formula to calculate the γ-index for each data point as follows:

$$\gamma(x) = \frac{1}{k}\sum_{j=1}^{k} \|x - z_j(x)\| \tag{1.1}$$

where x is a data point, k is the number of nearest neighbours, and $z_1(x), ..., z_k(x)$ are the k nearest neighbours of x.

The implemented function receives a *d x n* matrix *X* containing the data points and a scalar *k* representing the number of neighbours as parameters. It returns the γ-index for each data point in a *1 x n* vector *y*.

The function was tested on the test data and passed the test. Following steps are performed in the implementation of the function:

1. Implement a helper function *distmat* that calculates the distances from the data points to each other and return the distances as a matrix.
2. Get the distance matrix using the function *distmat* mentioned above.
3. Sort the distance matrix in ascending order.
4. Take only the *k*-nearest data points as neighbours for each data point.
5. Calculate the mean from the distances of the *k*-nearest neighbours and set it as the γ-index.
6. Return the calculated γ-index as a *1 x d* vector *y*. ✓

it is not a good idea to put a nan in the distmat!!!! 13 / 15

1.3 Assignment 3: LLE

The last task in the implementation part is to implement the *locally linear embedding* method as described by Saul & Roweis (2000) in their paper. The implemented *lle* function returns a *m x n* matrix *Y* representing the resulting embedding and takes following parameters as inputs:

- A *d x n* matrix *X* containing the data points.
- A scalar *m* representing the dimension of the resulting embedding.
- A string *n_rule* determining the method (*'knn'* or *'eps-ball'*) for building the neighbourhood graph.

✓

- A scalar *param* used as parameter for the *n_rule* (k or ϵ, respectively).
- A scalar *tol* determining the size of the regularization parameter.

The implementation is based on the pseudocode described by Saul & Roweis (2000) on their website[1], which contains of three main parts:

1. Find the nearest neighbours of each data point based on *n_rule*.
2. Solve for reconstruction weights W.
3. Compute embedding coordinates Y using weights W.

what do you think about the test? eps balls look strange...
there seem to be problems for eps ball!
do you check connectedness?
.
comment on the aspects!
.
10 / 20

[1] http://www.cs.nyu.edu/ roweis/lle/algorithm.html

Chapter 2

Application

In this chapter, we are trying to apply the algorithms that are described and implemented in Chapter 1 to various datasets: *usps*, *banana*, *fishbowl*, *swissroll* and *flatroll*.

2.1 Assignment 4: PCA

This assignment asks us to apply *PCA* to the *usps* data set and visualizing the results. The *usps* data set consists of 2007 images with the dimension of *16 x 16*. The images are hand-written digits of zero to nine, which can be viewed as classes. Firstly, i separate the data set according to each digit into ten classes and then applied *PCA* to each class. The *PCA* was applied to the original data set and noisy data set.

why do you separate the digits???

2.1.1 Original Data Set

PCA was applied to each class and then the results were visualized. As depicted in Figure 2.1 and 2.2, the following were visualized:

- All principle values (as bar plot).
- The largest 25 principal values (as bar plot).
- The first 5 principal directions (as images).

From the visualization, it can be obtained that the information of the data set is only represented by a small subset of principal components, about five to ten components.

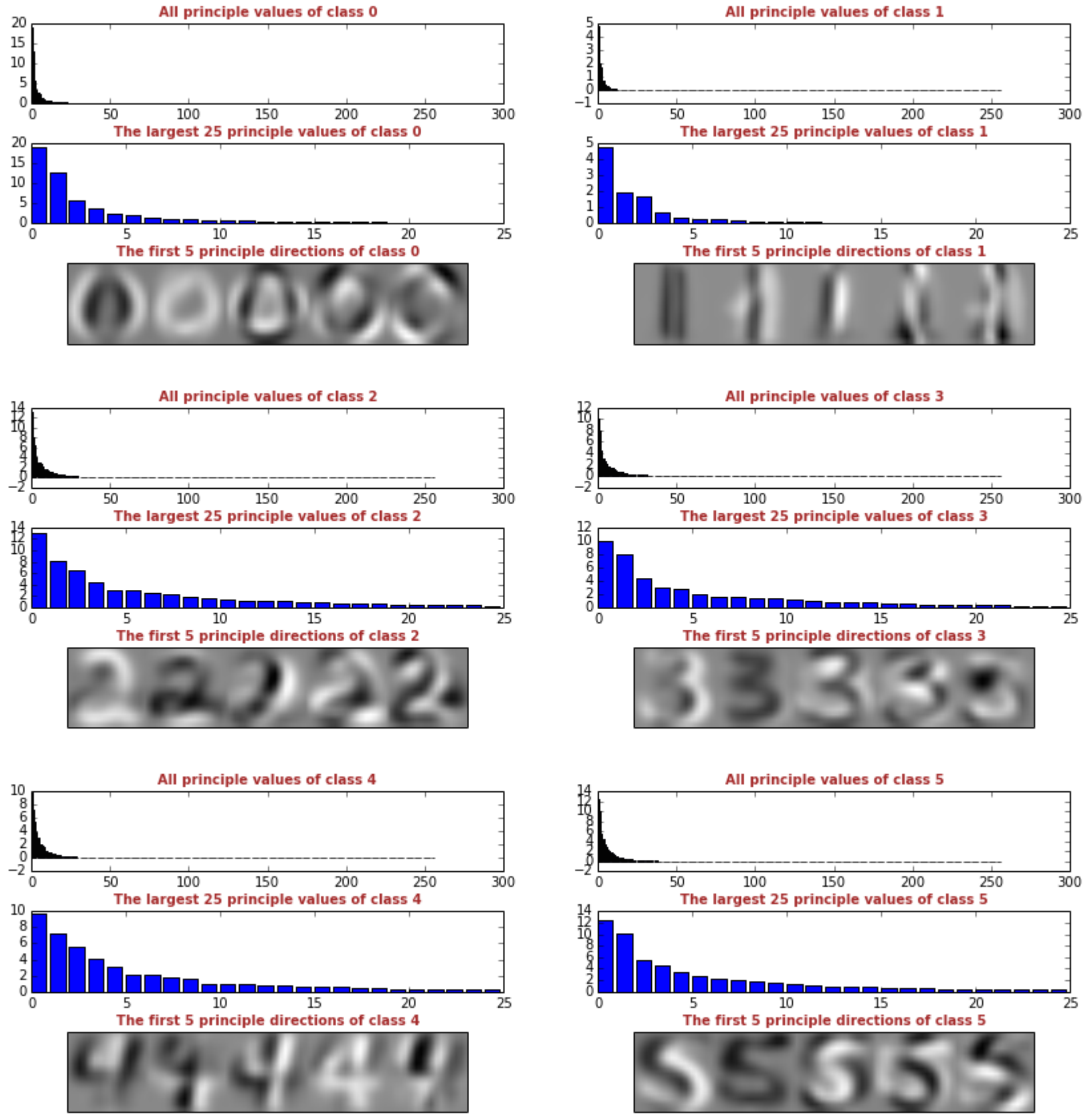
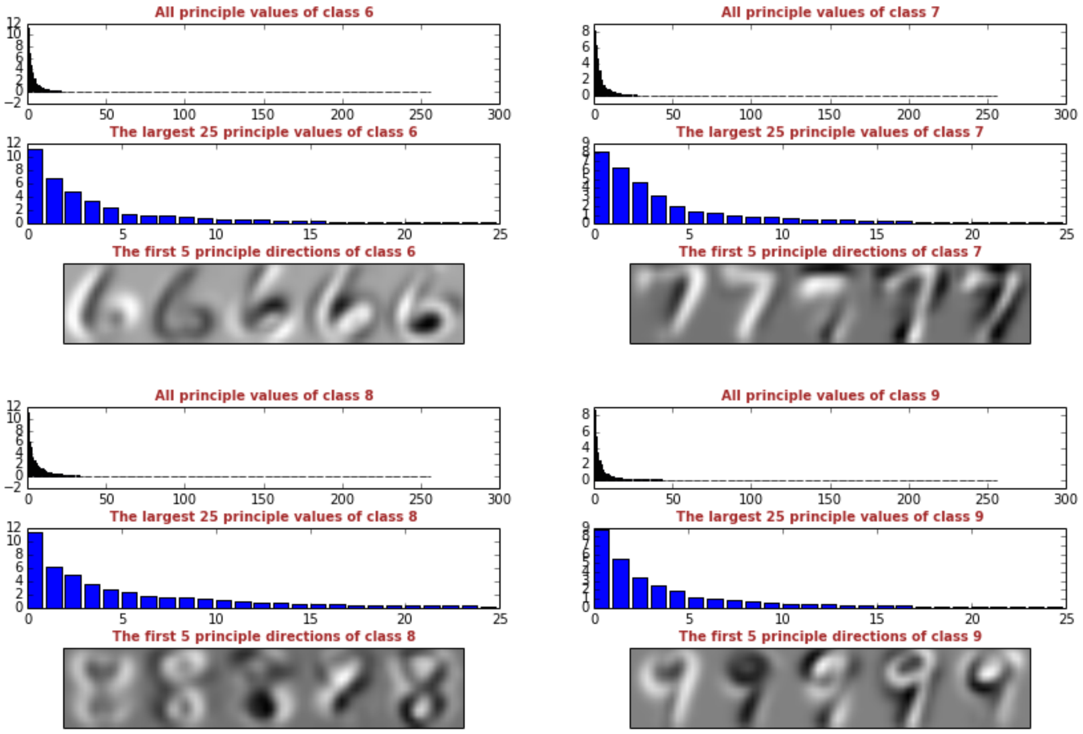


Figure 2.1: Principal components of usps original data set for class 0 - 5

2.1.2 Noisy Data Set

To make it more challenging, some noised are added to the original data set. There are three noise scenarios: *low gaussian noise*, *high gaussian noise* and *outliers*.

For the *low gaussian noise*, standard deviation used to generate the noise is 0.25, whereas for *high gaussian noise* and *outliers*, standard deviations of 0.55 and 1.25 are used, respectively. The same steps as in Subsection 2.1.1 are performed for all noisy data sets. The visualization is depicted in Figure 2.3 to 2.8.


All principle values of class 6

The largest 25 principle values of class 6

The first 5 principle directions of class 6

All principle values of class 7

The largest 25 principle values of class 7

The first 5 principle directions of class 7

All principle values of class 8

The largest 25 principle values of class 8

The first 5 principle directions of class 8

All principle values of class 9

The largest 25 principle values of class 9

The first 5 principle directions of class 9

Figure 2.2: Principal components of usps original data set for class 6 - 9

It can be obtained that the information of the data set is now represented by a bigger number of principal components. The noisier a data set is, the bigger the number of principal components is needed to represent the information of the data set. ✓

Figure 2.9 shows ten images of the original data, noisy data, and their reconstruction (denoised) data using *PCA*. The number of components m used for *moderately* noisy images is 11, for *very* noisy images is 75 and for *extremely* noisy images is 150.

3 / 10

Your report features to little ANALYSIS, mainly description. Does it work???

2.2 Assignment 5: Outlier Detection Using γ-Index

In this assignment, the γ-index method is utilized to detect outliers and applied it to the *banana* data set. The positive class of the data set is used as *inliers*, to which the negative class is added as outliers. The γ-index is then used to detect outliers with contamination rates of 1%, 5%, 10% and 25% relative to the positive class. Figure 2.10 shows the complete original data set, both positive and negative class, whereas Figure 2.11 shows the contaminated data set.

There are three methods that should be used to detect the outliers: (a) the γ-index with $k = 3$, (b) the γ-index with $k = 10$ and (c) the distance to the mean for each data

- calculate_auc is possible without a for-loop
- from the text it is not clear how you chose the thresholds

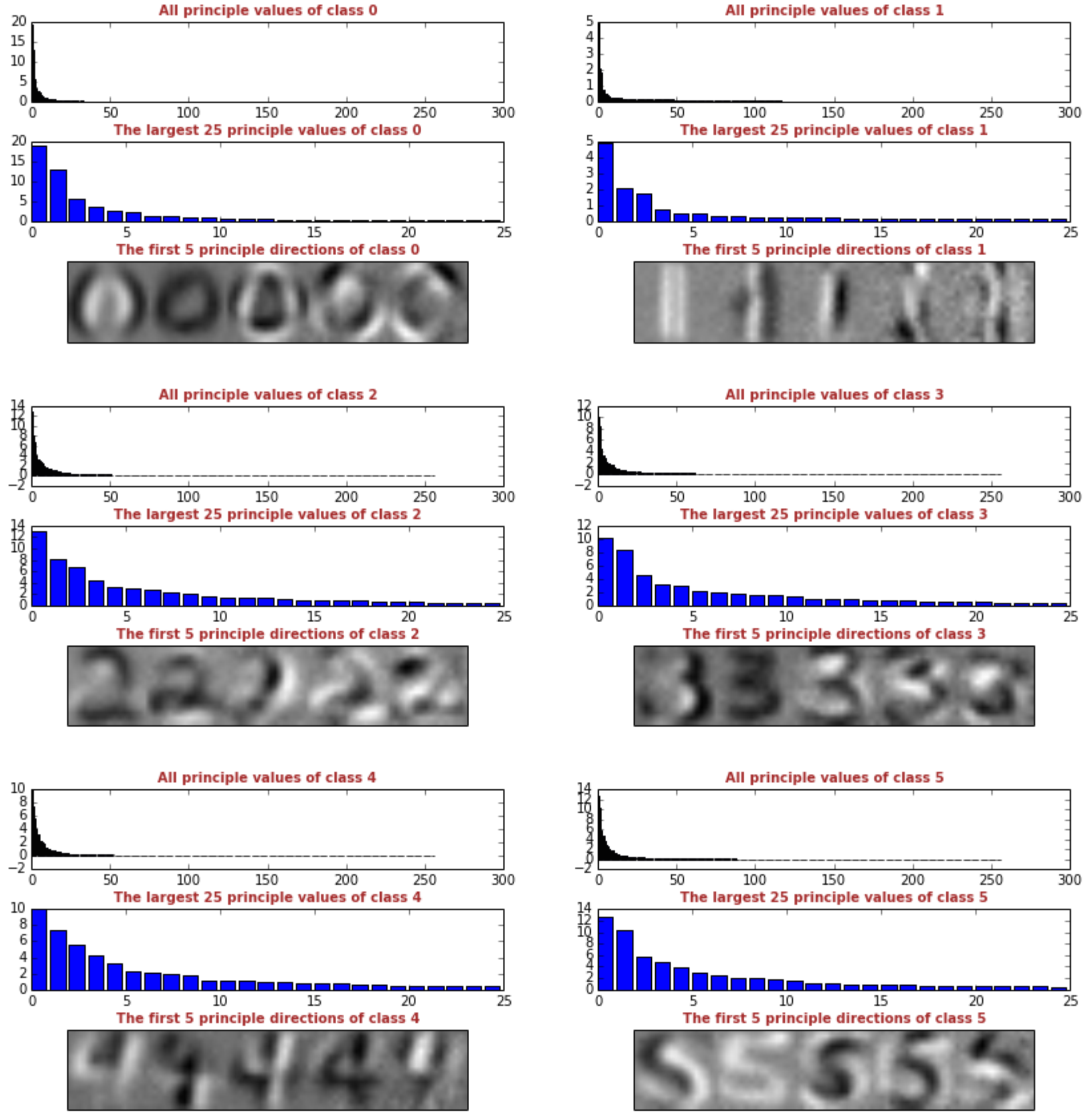


Figure 2.3: Principal components of moderately noisy data set for class 0 - 5

point. All of the methods are then applied to the four contamination rates mentioned above. After that, the *AUC* (area under the *ROC*) should be calculated. Figure 2.12 shows the boxplots that visualize the distribution of the *AUC* values.

The boxplots show that the method using the distance to the mean for each data point performed quite bad, while both of the γ-index methods performed very well, especially for the data set with lower contamination rates. The γ-index with $k = 10$ performed slightly better than the γ-index with $k = 3$.

explain the differences! why do the algorithms behave as they do?

.

more explanations and more discussions needed!

.

6 / 10

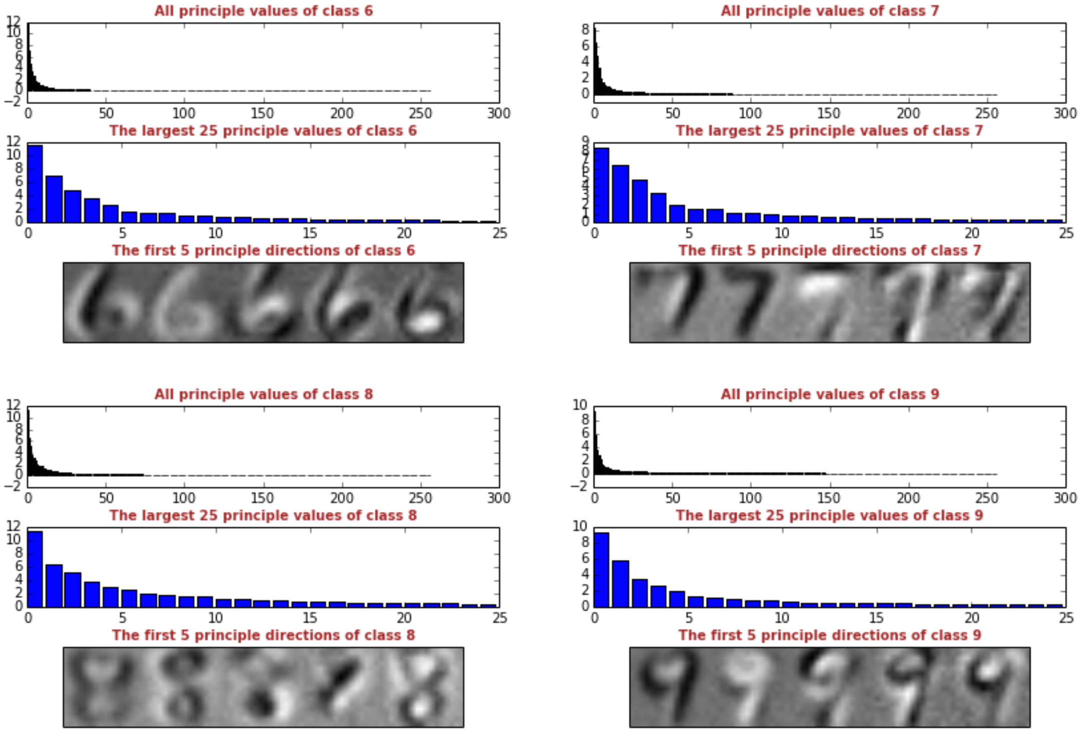


Figure 2.4: Principal components of moderately noisy data set for class 6 - 9

2.3 Assignment 6: LLE

In this assignment *LLE* should be applied to the data sets *fishbowl*, *swissroll* and *flatroll* using appropriate values for the parameters.

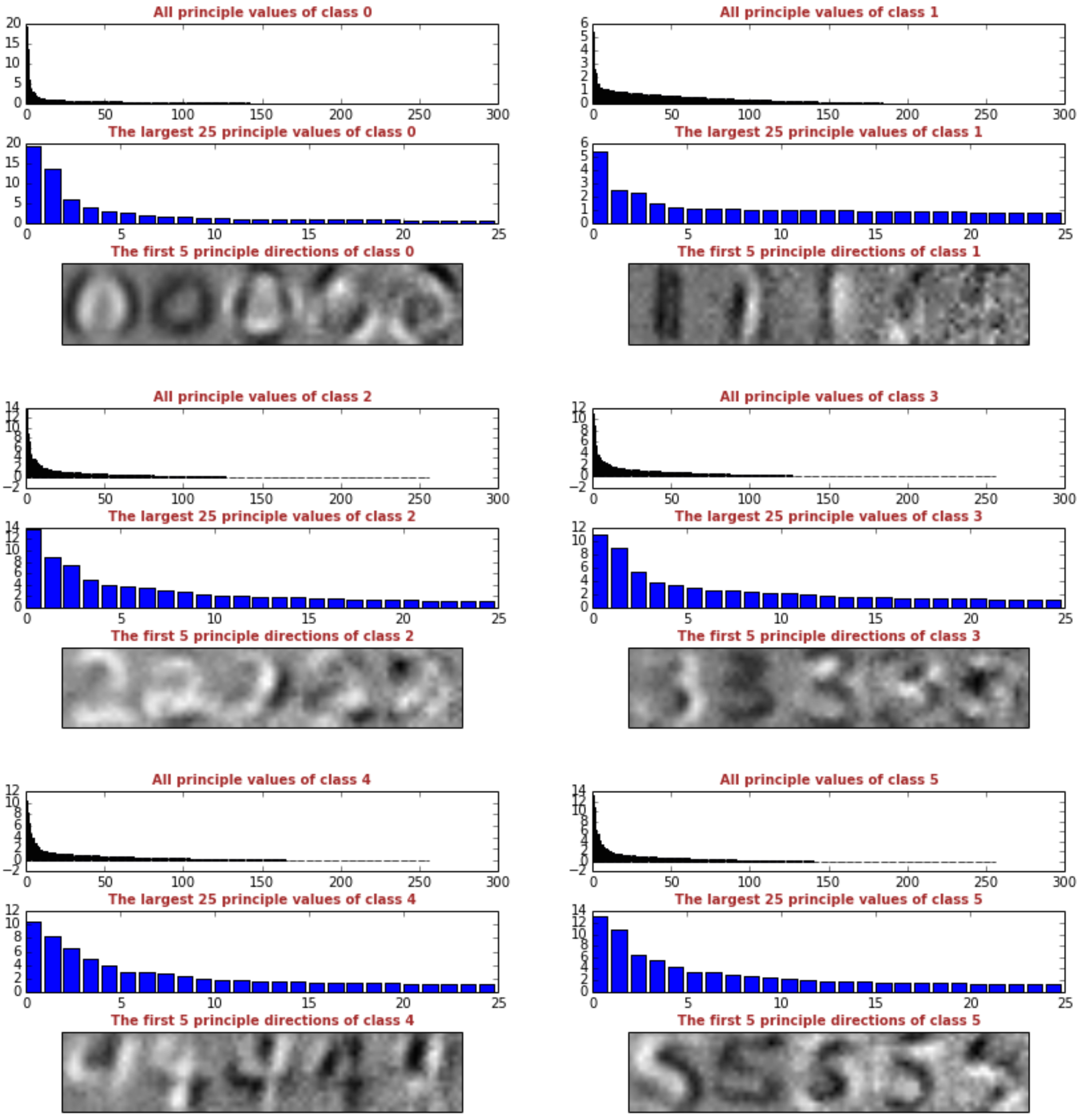
2.3.1 Flatroll Data Set

For the *flatroll* data set, it is quite easy to find the appropriate values for the parameters. Figure 2.13 shows the *flatroll* data set and its resulting embedding using *knn* rule with $k = 12$ and regularization parameter $tol = 1e - 3$. The resulting embedding is quite good.

2D -> 1D. what is shown on the second axis in the 1D plot?

2.3.2 Swissroll Data Set

For the *swissroll* data set, it is already quite hard to find the appropriate values for the parameters. Figure 2.14 shows the visualizationof the *swissroll* data set and its resulting embedding using *eps-ball* with $\epsilon = 5.7$ and regularization parameter $tol = 2.8e - 5$. I cannot find the appropriate parameters.



All principle values of class 0

The largest 25 principle values of class 0

The first 5 principle directions of class 0

All principle values of class 1

The largest 25 principle values of class 1

The first 5 principle directions of class 1

All principle values of class 2

The largest 25 principle values of class 2

The first 5 principle directions of class 2

All principle values of class 3

The largest 25 principle values of class 3

The first 5 principle directions of class 3

All principle values of class 4

The largest 25 principle values of class 4

The first 5 principle directions of class 4

All principle values of class 5

The largest 25 principle values of class 5

The first 5 principle directions of class 5

Figure 2.5: Principal components of very noisy data set for class 0 - 5

2.3.3 Dense Fishbowl Data Set

For the *dense fishbowl* data set, i use *knn* with $k = 7$ and regularization parameter $tol = 1e - 5$. Figure 2.15 shows the visualization and its resulting embedding.

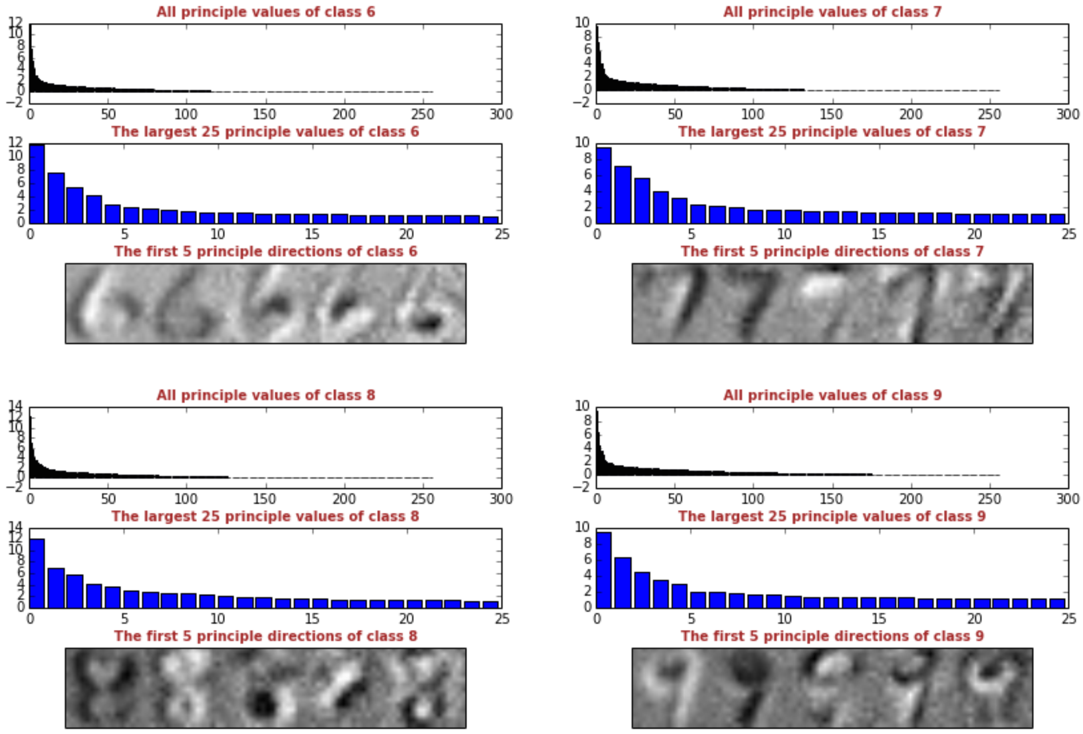


Figure 2.6: Principal components of very noisy data set for class 6 - 9

2.3.4 Fishbowl Data Set

For the *fishbowl* data set, it is also really difficult to find the appropriate parameters. I cannot find them. Figure 2.16 shows the visualization of the *fishbowl* data set and its resulting embedding using *knn* with $k = 14$ and regularization parameter $tol = 4e - 8$.

does the embedding correspond to a correct embedding?

6 / 20

2.4 Assignment 7: LLE With Noise

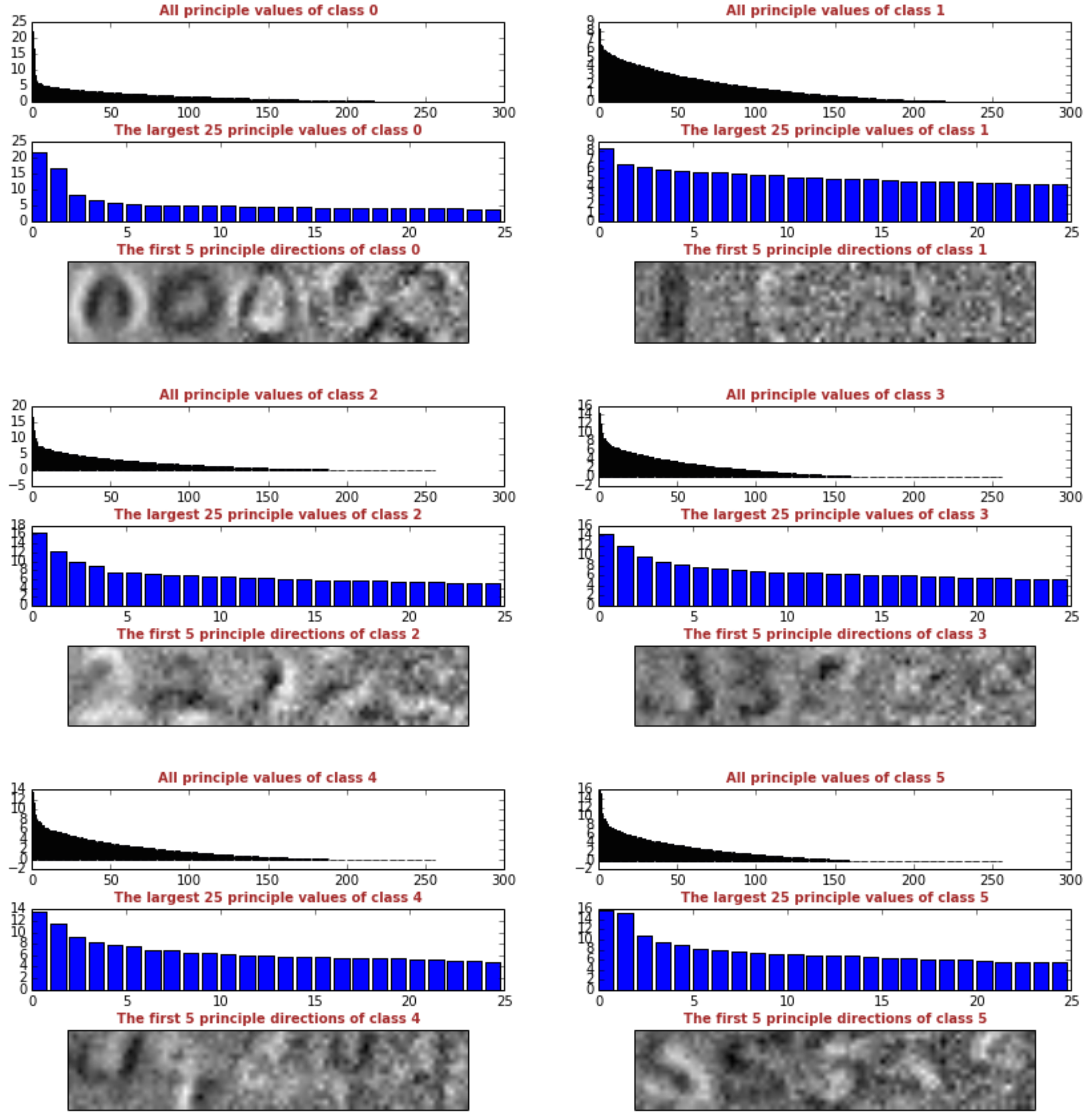
In the last assignment of this problem set, *LLE* has to be applied to noisy *flatroll* data set. Two gaussian noise were added to the data set, with variance 0.2 and 1.8, respectively. After that, both noisy data sets should be unrolled using *knn* with a good value of k and a value which is obviously too large. Figure 2.17 depicts the two noisy images and their resulting embedding. It can be obtained that the noisy data set with big variance(1.8) is not very good unrolled.

you add noise with standard deviation of 0.2 and 1.8, not variance!

actually, only one of the four can be unrolled.

Your visualizations are not very helpful in seeing that!

3 / 10



All principle values of class 0

The largest 25 principle values of class 0

The first 5 principle directions of class 0

All principle values of class 1

The largest 25 principle values of class 1

The first 5 principle directions of class 1

All principle values of class 2

The largest 25 principle values of class 2

The first 5 principle directions of class 2

All principle values of class 3

The largest 25 principle values of class 3

The first 5 principle directions of class 3

All principle values of class 4

The largest 25 principle values of class 4

The first 5 principle directions of class 4

All principle values of class 5

The largest 25 principle values of class 5

The first 5 principle directions of class 5

Figure 2.7: Principal components of extremely noisy data set for class 0 - 5

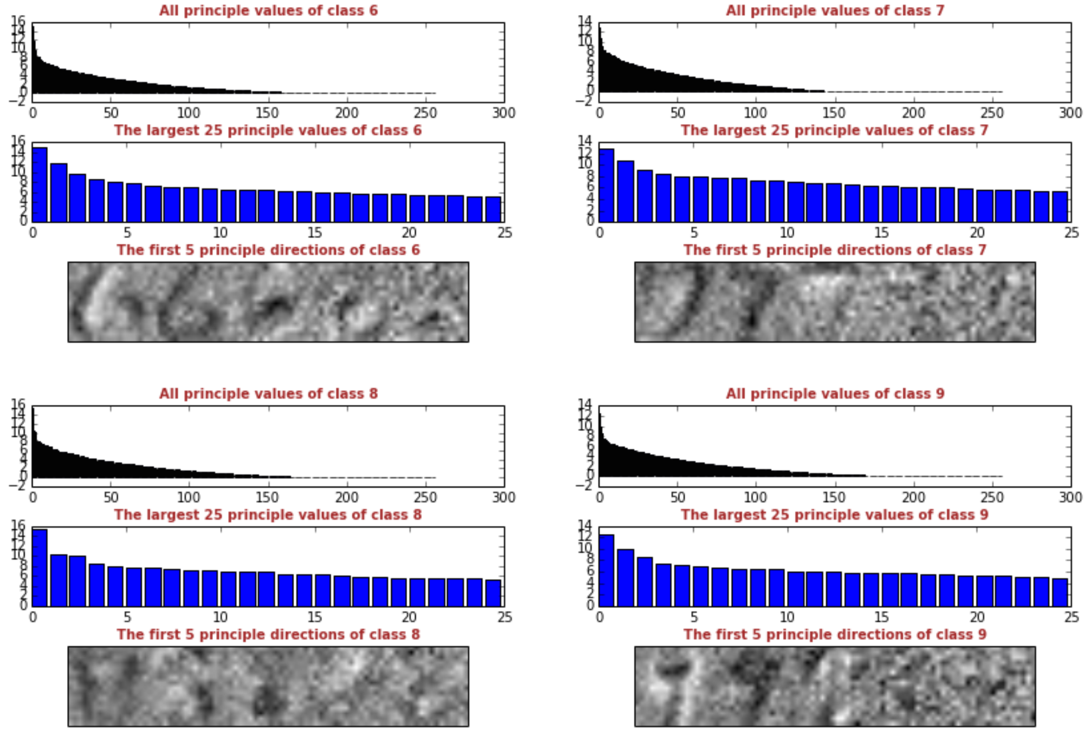


Figure 2.8: Principal components of extremely noisy data set for class 6 - 9

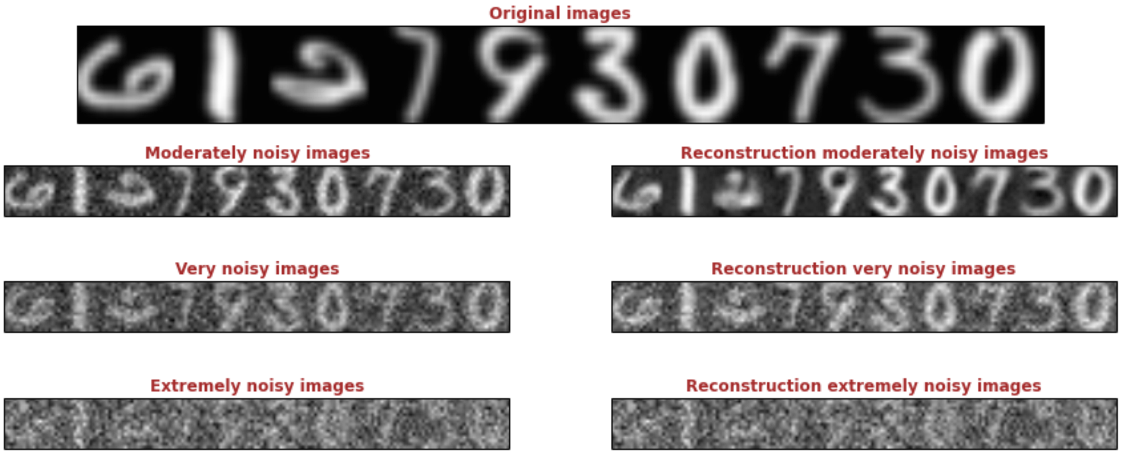

Original images
Moderately noisy images
Reconstruction moderately noisy images
Very noisy images
Reconstruction very noisy images
Extremely noisy images
Reconstruction extremely noisy images

Figure 2.9: Examples of some original, noisy and reconstruction images

you added noise to all images! why that? the task says explicitly that in the outlier scenario you have to add noise to only five images!

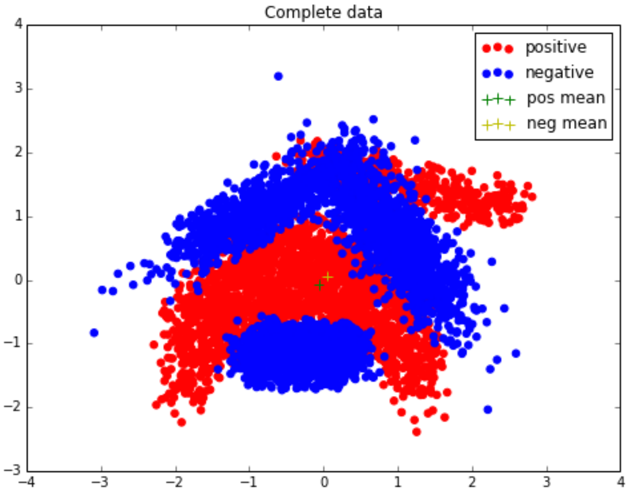


Figure 2.10: Both positive and negative of banana data set, including the mean of both classes

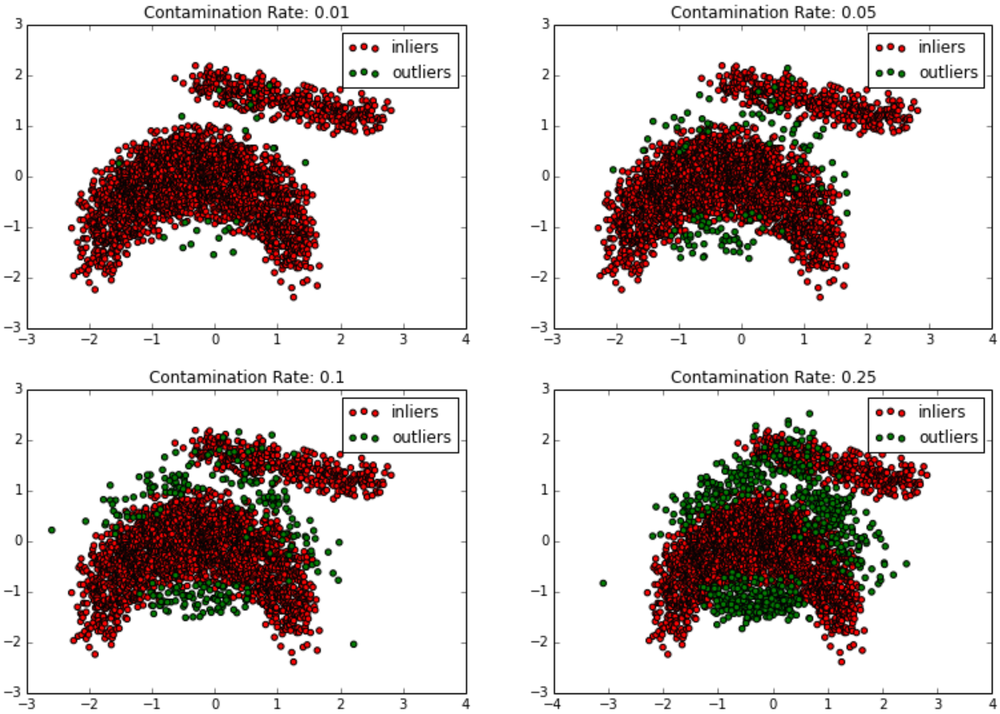


Figure 2.11: Contaminated banana data set with contamination rate of 1%, 5%, 10% and 25%

labels?

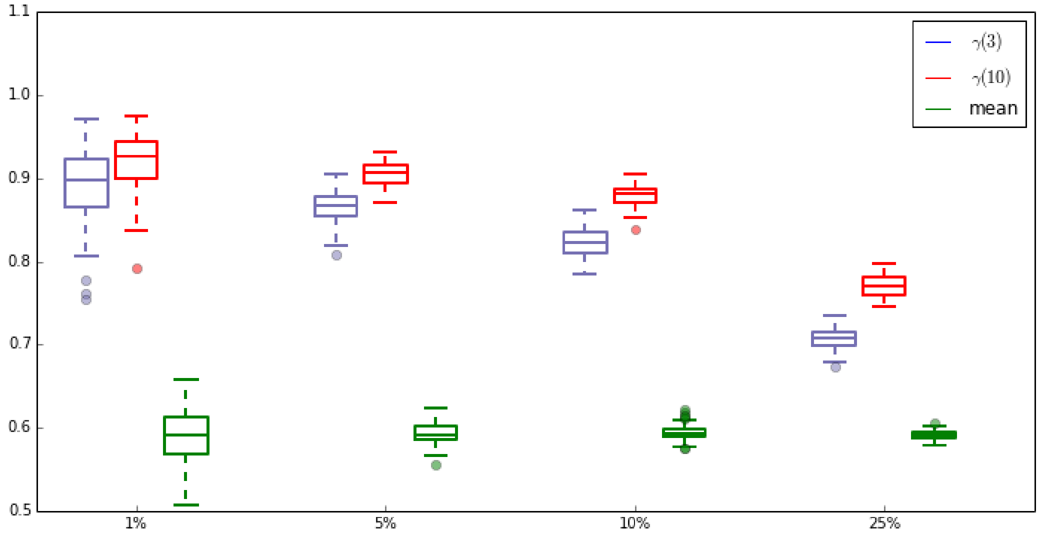
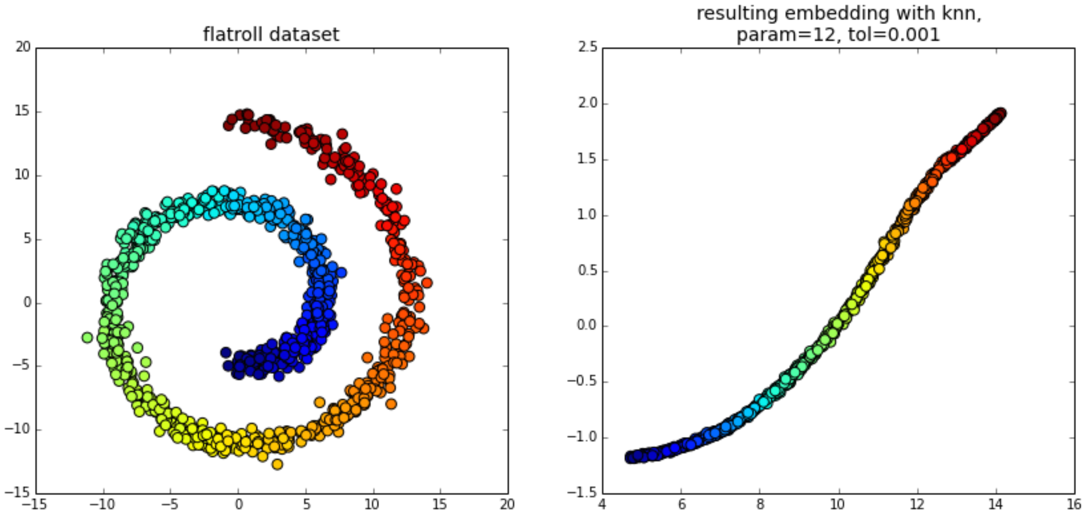


Figure 2.12: Boxplots visualizing the distribution of the *AUC* values



what is depicted on the x-axis? y-axis?

Figure 2.13: Visualization of *flatroll* data set and its resulting embedding

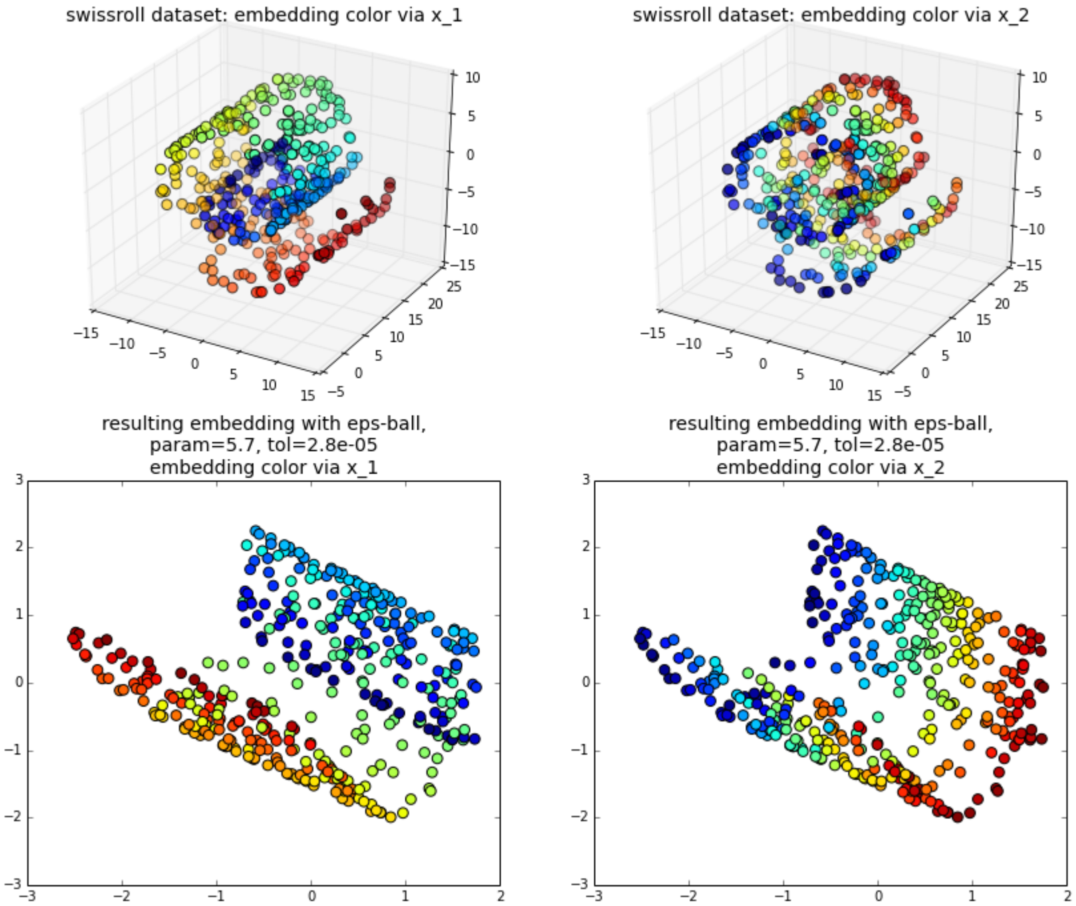


Figure 2.14: Visualization of *swissroll* data set and its resulting embedding

in your emedding, red points are very close to green points. this is not good!

a good embedding can be seen in the paper and the slides!

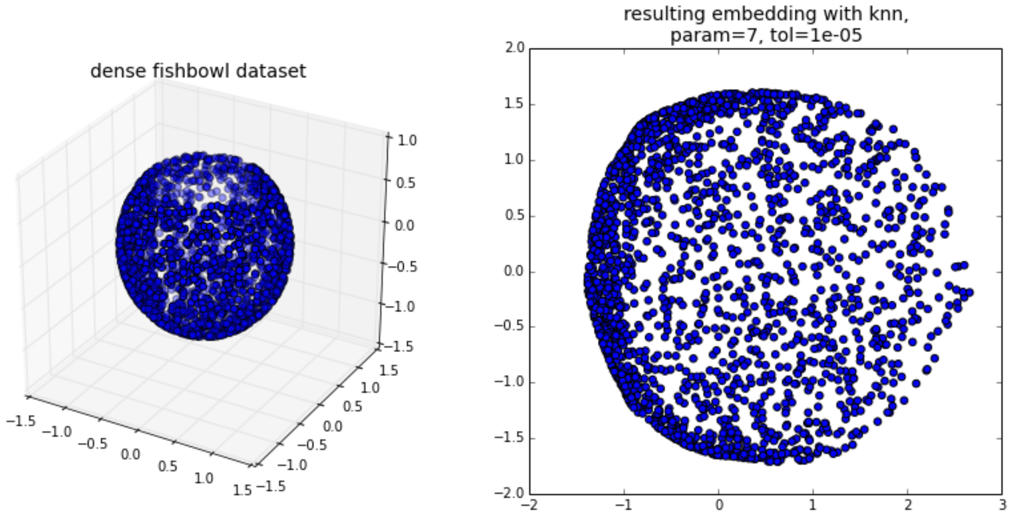


Figure 2.15: Visualization of *dense fishbowl* data set and its resulting embedding

what would be a good coloring? there was a forum discussion on that in ISIS!

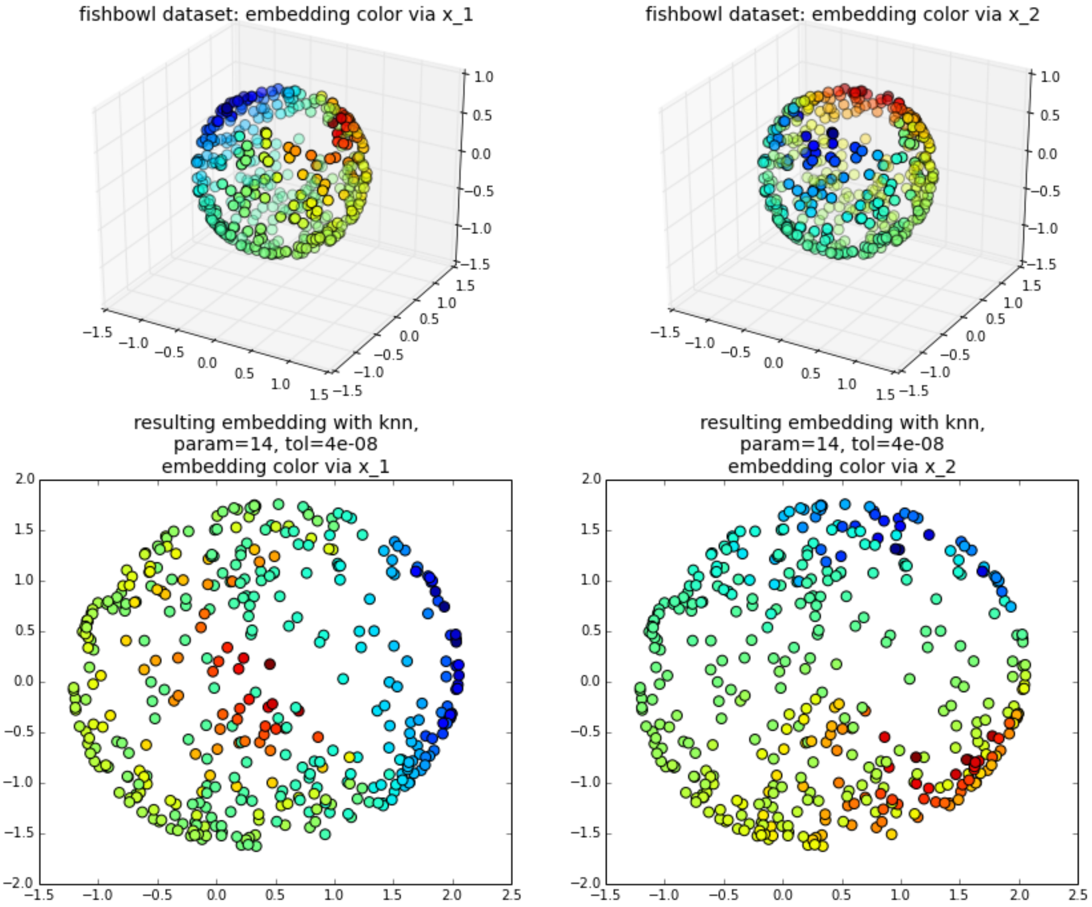
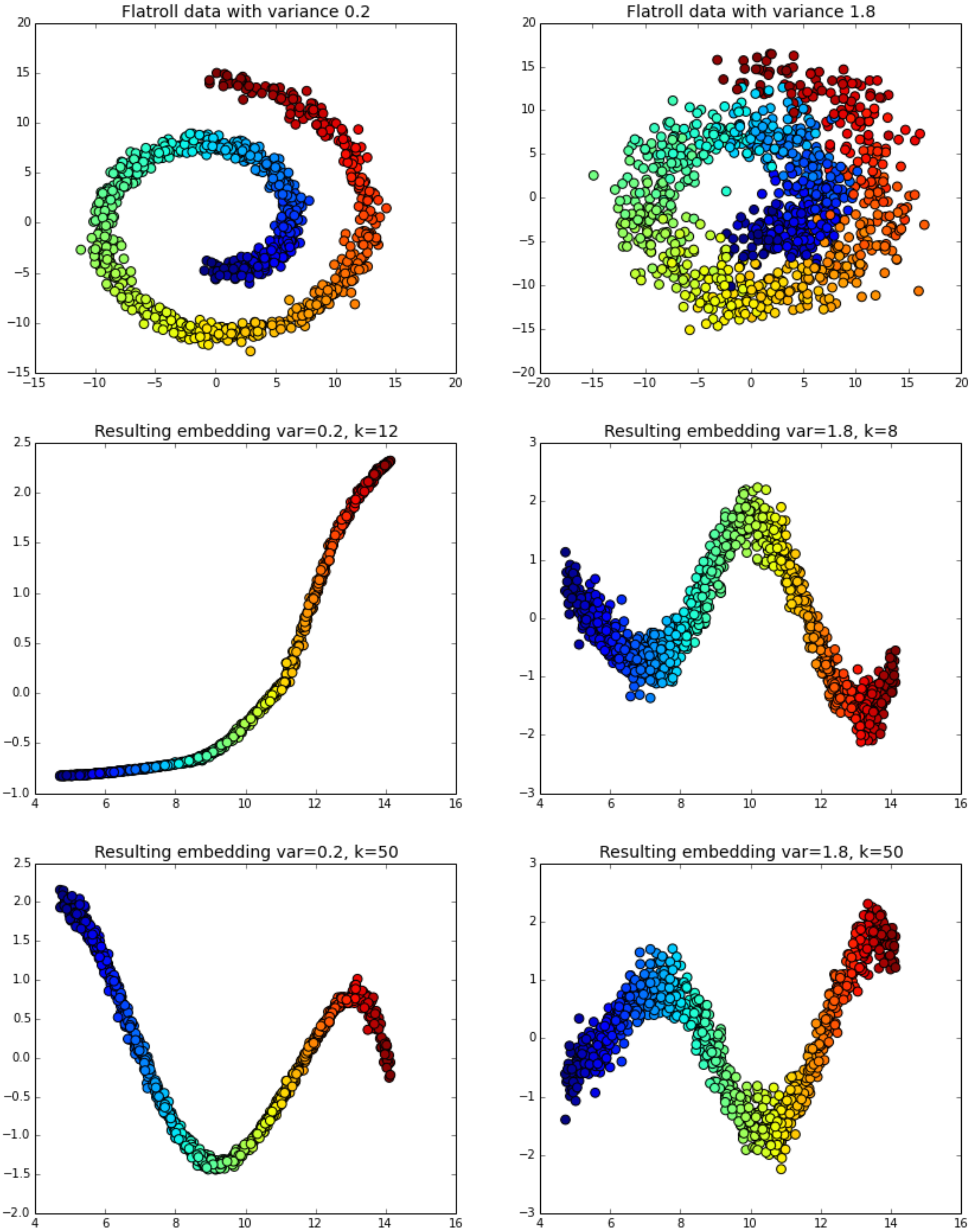


Figure 2.16: Visualization of *fishbowl* data set and its resulting embedding

does this make sense? discussion is lacking!



Flatroll data with variance 0.2

Flatroll data with variance 1.8

Resulting embedding var=0.2, k=12

Resulting embedding var=1.8, k=8

Resulting embedding var=0.2, k=50

Resulting embedding var=1.8, k=50

Figure 2.17: Visualization of two noisy *flatroll* data set and their resulting embedding

again, what do the axes means???

no neighbourhood graph!

Bibliography

Harmeling, S., Dornhege, G., Tax, D., Meinecke, F. & Müller, K.-R. (2006), 'From outliers to prototypes: Ordering data', *Neurocomputing* **69**(13-15), 1608–1618.

Saul, L. K. & Roweis, S. T. (2000), An introduction to locally linear embedding, Technical report.