Report for Sheet 01

Lab Course Machine Learning and Data Analysis

Mario Tambos

May 5, 2017

Implementation comments

The code was structured mostly in functions following an imperative paradigm, except for the PCA class, which was required by the assignment's statement.

One function was declared for each assignment in Part 2.

Given the low time performance of the LLE's implementation for the assignment, it was decided to use scikit-learn's LocallyLinearEmbedding class for Assignment 8, in order to optimize the k hyperparameter more quickly.

Beyond matplotlib, numpy and scipy, networkx was used to build and display the node neighborhood graphs, tqdm to show progress bar on loops, and pandas together with seaborn to show the boxplots in Assignment 6.

All tasks were completed. However, the denoising test failed in Assignment 1, and no good value for k could be found in Assignment 8 for the high noise case.

In the denoising case, the cause of the failure is unclear. The result's provided by the exercise's PCA implementation were successfully checked against results from the scikit-learn's PCA class.

In the high-noise case in Assignment 8, the problems seems to be that there is no right k value. For low values of k, all points are embedded in a very small interval. For larger values of k, the points use neighbors from the inner loops of the spiral to calculate the embedding, resulting in something similar to the bottom plot in figure 1.

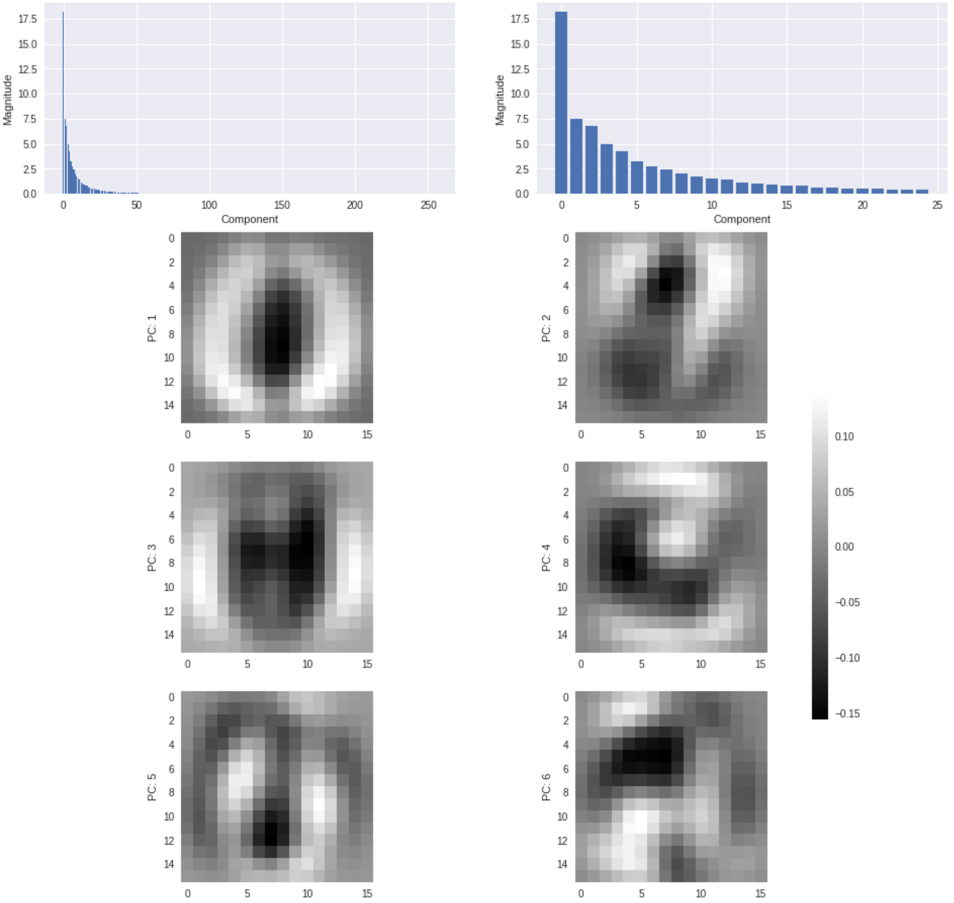


Figure 1: PCA for the usps dataset. The top left plot shows all principal values extracted, while the top right plot shows only the first (largest) 25 values. The bottom 6 images show the first 6 principal directions.

Assignment 5

2)

b)

Figure 1 shows the result of performing PCA on the usps dataset.

3)

The most notable differences when contrasting figure 1 against figures 2a, 3a and 4a are the resulting principal components. The only other noticeable difference is the higher magnitude of the smallest eigenvalues in the high noise scenario.

As expected, the reconstructions in figure 2b are better than in figure 3b. In the outliers scenario (figure 4b), the reconstructions were surprisingly good for the outliers, but the reconstructions of the images without noise were comparable to figure 3b.

Low Gaussian noise

The experiment results for the scenario where Gaussian noise drawn from $\mathcal{N}(0, \sigma^2 = 0.3)$ was added is shown in figure 2.

High Gaussian noise

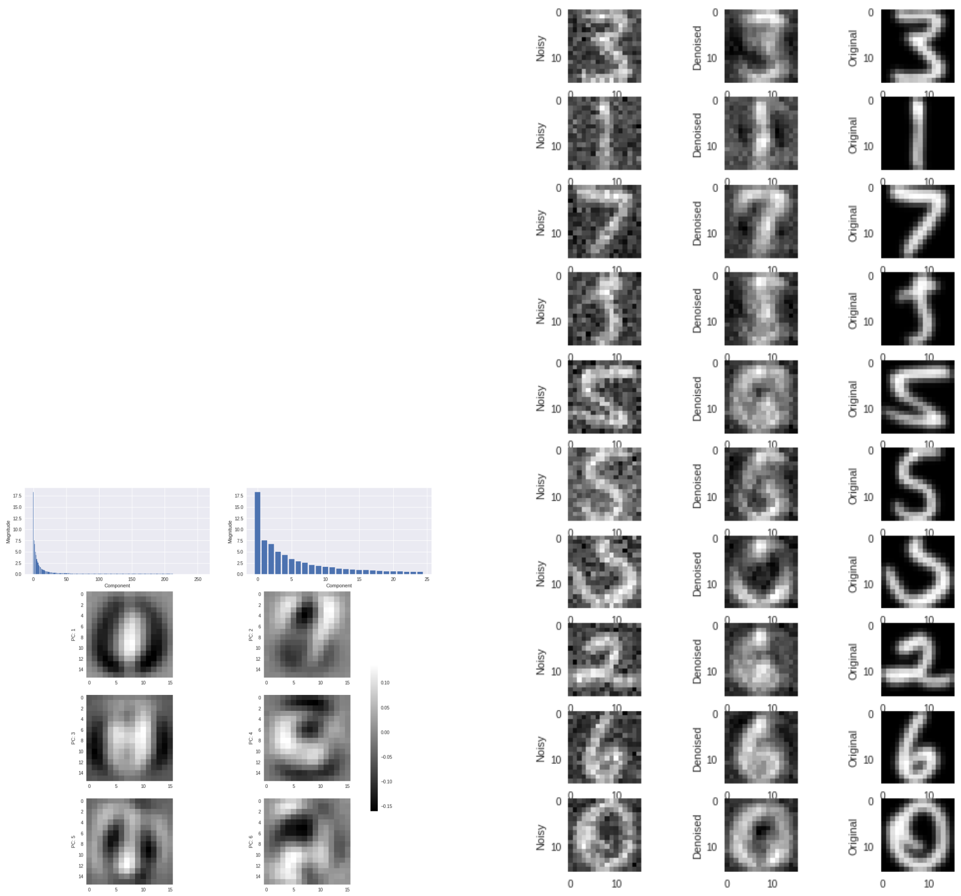
The experiment results for the scenario where Gaussian noise drawn from $\mathcal{N}(0, \sigma^2 = 0.7)$ was added is shown in figure 3.

Outliers

The experiment results for the scenario where Gaussian noise drawn from $\mathcal{N}(0, \sigma^2 = 5)$ was added to the first five images is shown in figure 4.

Assignment 6

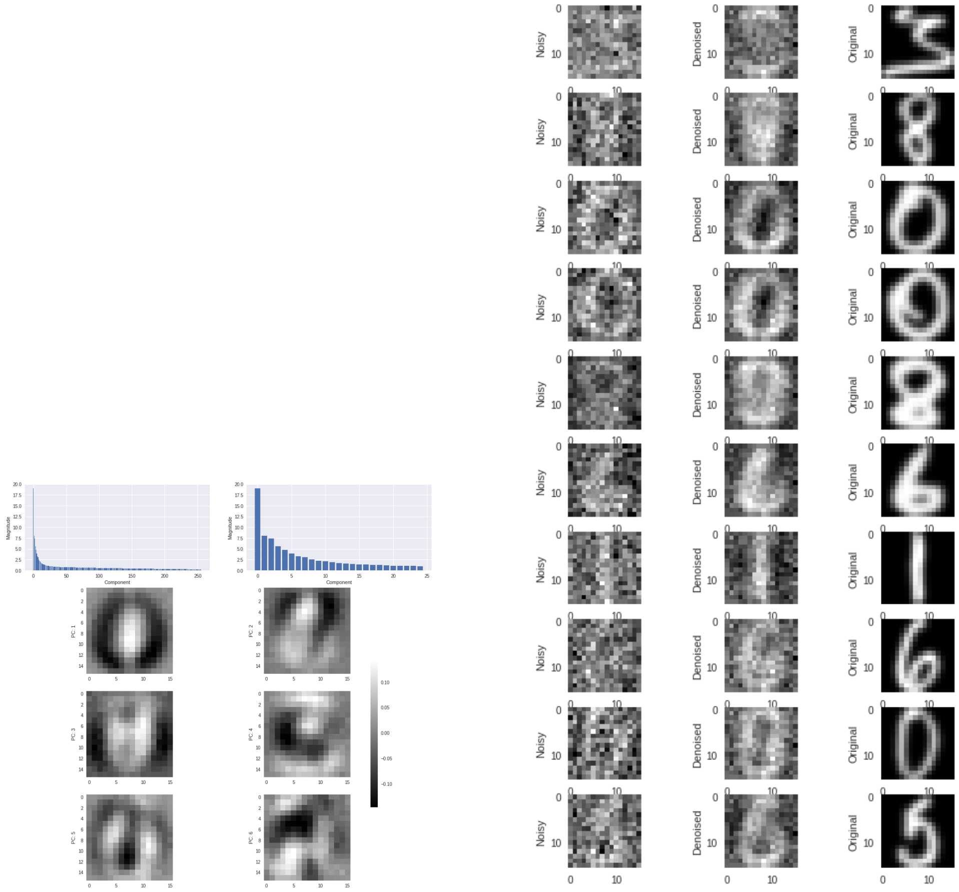
Figure 5 shows the experiment's results in three boxplots. The γ-index method with $k = 3$ performed consistently better than with $k = 5$, but both were outperformed by the distance-to-the-mean method in all cases. Moreover, the confidence intervals of the three methods were tighter the higher the outlier rate.



(a) The top left plot shows all principal components, while the top right plot shows only the first 25 components. The bottom 6 images show the first 6 principal directions.

(b) Random sample of 10 items. The first column shows the noisy versions, the middle column the denoised versions (using PCA) and the last column shows the original images.

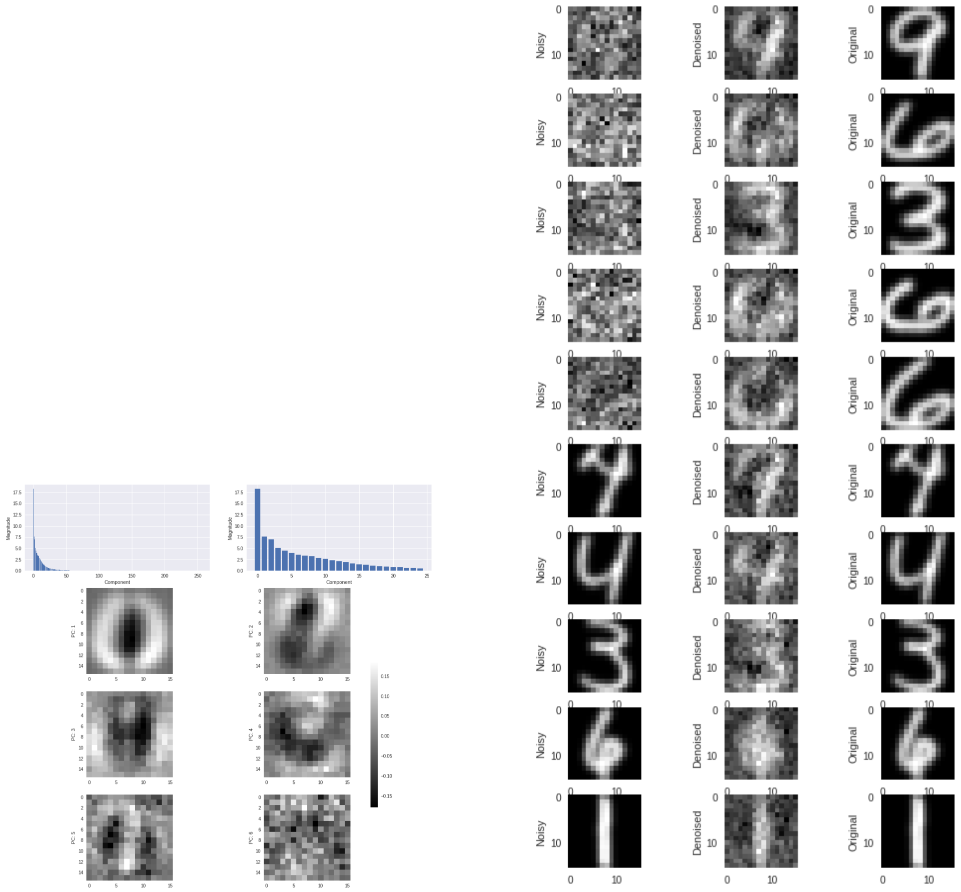
Figure 2: PCA denoising. Low Gaussian noise ($\sigma = 0.3$) scenario.



(a) The top left plot shows all principal components, while the top right plot shows only the first 25 components. The bottom 6 images show the first 6 principal directions.

(b) Random sample of 10 items. The first column shows the noisy versions, the middle column the denoised versions (using PCA) and the last column shows the original images.

Figure 3: PCA denoising. High Gaussian noise ($\sigma = 0.7$) scenario.



(a) The top left plot shows all principal components, while the top right plot shows only the first 25 components. The bottom 6 images show the first 6 principal directions.

(b) Random sample of 10 items. The first column shows the noisy versions, the middle column the denoised versions (using PCA) and the last column shows the original images.

Figure 4: PCA denoising. Outliers ($\mathcal{N}(0, \sigma = 5)$ added to the first 5 images) scenario.

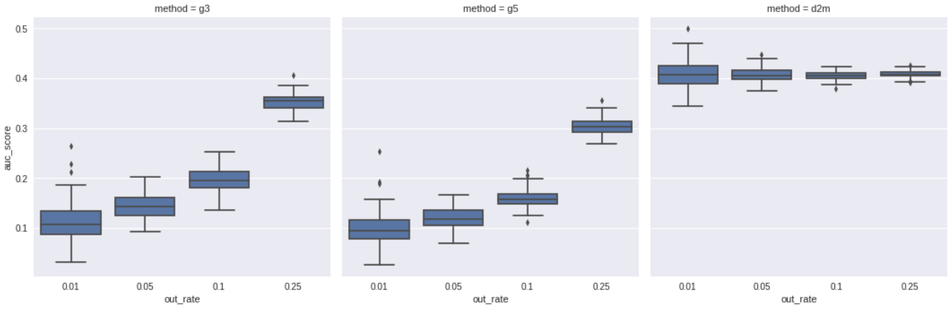


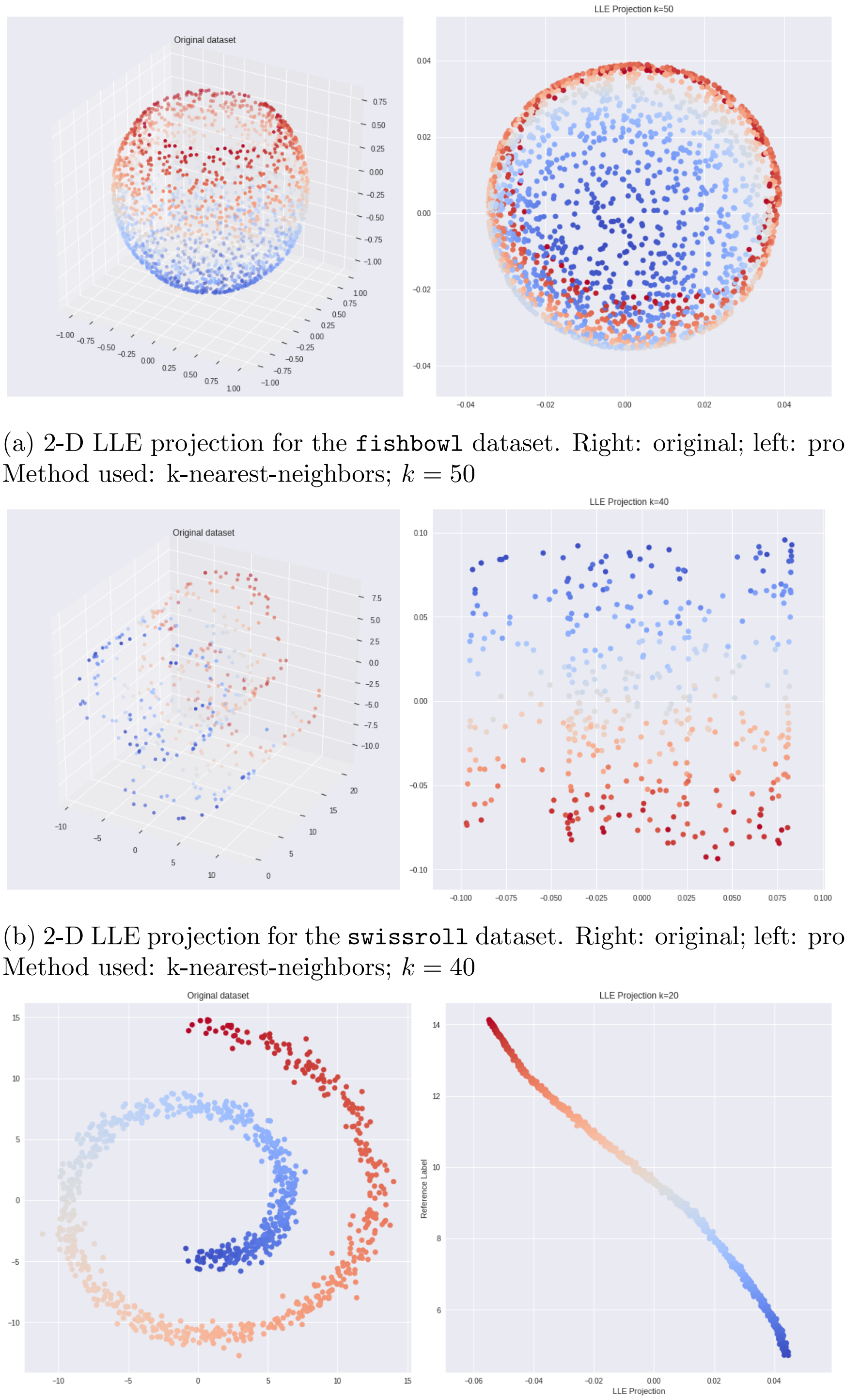
Figure 5: Outlier detection AUC scores on the `banana` dataset. The leftmost plot shows the AUC scores using the γ-index method with $k = 3$. The middle plot shows the AUC scores using the γ-index method with $k = 5$. The rightmost plot shows the AUC scores using the distance-to-the-mean method. The x-axis indicates the outlier rate used (0.01, 0.05, 0.1 and 0.25).

Assignment 7

Figure 6 shows the 1-D and 2-D LLE projections for the `fishbowl`, `swissroll` and `flatroll` datasets.

Assignment 8

Figure 6 shows the 1-D LLE projections for the `flatroll` dataset, in which Gaussian noise drawn from $\mathcal{N}(0, \sigma = 0.2)$ and $\mathcal{N}(0, \sigma = 1.8)$ was added.



(a) 2-D LLE projection for the `fishbowl` dataset. Right: original; left: projection. Method used: k-nearest-neighbors; $k = 50$

(b) 2-D LLE projection for the `swissroll` dataset. Right: original; left: projection. Method used: k-nearest-neighbors; $k = 40$

(c) 1-D LLE projection for the `flatroll` dataset. Right: original; left: projection. Method used: k-nearest-neighbors; $k = 20$

Figure 6: LLE projections.

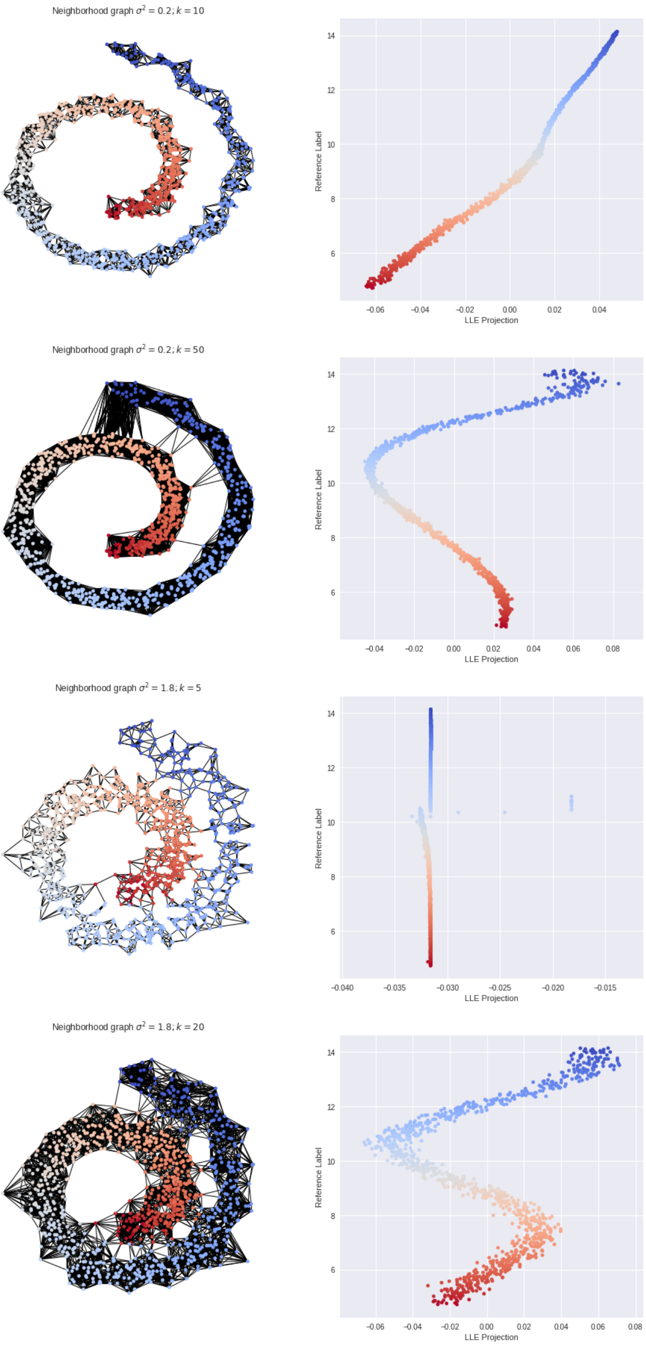


Figure 7: 1-D LLE projections for the `flatroll` dataset + Gaussian noise. The top two plots show the dataset + $\mathcal{N}(0, \sigma^2 = 0.2)$; with the topmost one using parameter $k = 10$ and the following one using $k = 50$. The bottom two plots show the dataset + $\mathcal{N}(0, \sigma^2 = 1.8)$; with the upper one using parameter $k = 5$ and the following one using $k = 20$.