GE 461 Project
Dimensionality Reduction and Visualization

Gonca Yılmaz 21702727

Spring 2021

1 Introduction

In this project, the goal is to develop a system that recognizes hand-written digits using Python [11]. For reading the dataset, Pandas library is utilized [8]. In the first part of this assignment, a Gaussian classifier is used to train the model. Different dimensionality reduction techniques, Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), are applied before the training phase to observe how they affect the classifier [7]. For the Gaussian classifier, Quadratic Discriminant Analysis (QDA) and computing accuracy scores, the scikit-learn library, for plotting and showing eigenvectors and classification errors, Matplotlib library are used [1],[6],[9]. In the last part of the project, the 400-dimensional data is visualized on 2-dimensional space using the following data visualization techniques: t-distributed Stochastic Neighbor Embedding (t-SNE) and Sammon Mapping [4],[12].

2 Dimensionality Reduction with PCA

In this part, the aim is to project the 400-dimensional data onto lower subspaces by using PCA and observe its impact on the Gaussian classifier. To split the dataset into training and test, compute eigenvalues and apply PCA, scikit-learn, and NumPy libraries are utilized [5],[9].

2.1 Eigenvalues

In this part, the eigenvalues are computed from the training data (50% of the whole data, 2,500 pattern) and plotted below. Looking at the Figure 1, it is plausible to say that 50 components can be enough to capture the digits' pattern. The eigenvalues are approaching 0, starting from near 50.

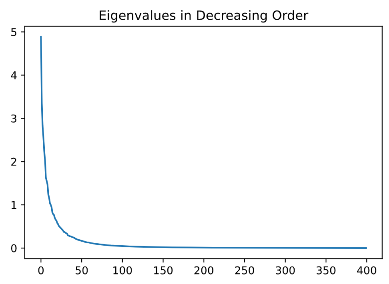

Figure 1: Eigenvalues in Descending Order

2.2 The Mean and the Eigenvectors

In this part, the sample mean of the whole training data is shown and the eigenvectors associated to first 50 components are shown.

As seen in the Figure 2, the sample mean looks more like a combination of digits 3, 8, and 9. Because these three classes are so similar, they might be dominating other classes in the sample mean.

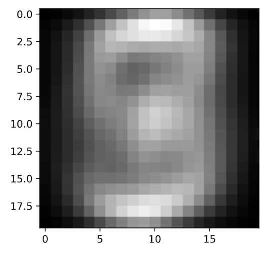

Figure 2: The Sample Mean

In the Figure 3, the chosen 50 components' eigenvectors are shown. As the associated eigenvalue decreases, the eigenvectors contain less information, seem less meaningful and more resemble noise. The first eigenvectors, which have larger eigenvalues, represent some important characteristics of the dataset [7].

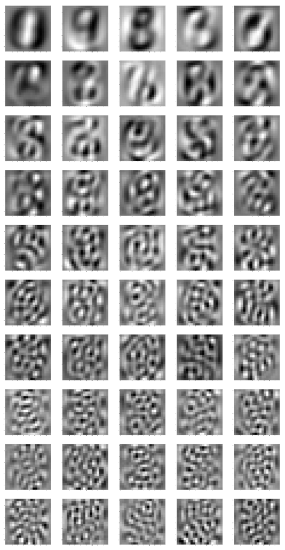

Figure 3: The Eigenvectors of First 50 Components

2.3 Projecting Data onto Different Number of Lower Spaces

In this sub-section, the number of components is changed, and its impact on the classification performance is analyzed. The classification error of training and test sets is plotted to observe the relation between the number of components and the classification. For classification, Quadratic Discriminant Analysis classifier from the scikit-learn library is used [9].

As shown in Figure 4, as the number of components increases, the classification error approaches 0 on training data. This increase is expected because as the number of components increases, the model captures the training data's noise. The noise captured leads to predicting the training set more accurately.

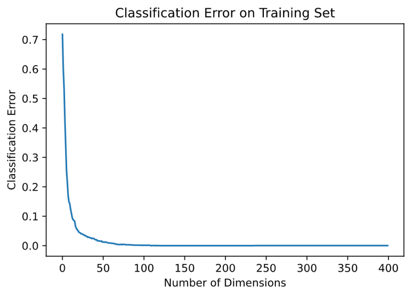

Figure 4: Classification Error on Training Set

However, for the test set, the prediction error increases as the number of dimensions increase (see Figure 5). Because as the number of components increments, the model memorizes the training set. Hence, it performs poorly when new samples are encountered.

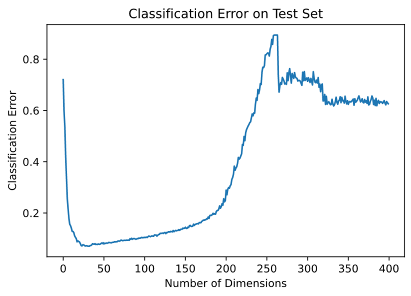

Figure 5: Classification Error on Test Set

3 Dimensionality Reduction with LDA

In this part, LDA is utilized to conduct dimensionality reduction. The Linear Discriminant Analysis model implemented by scikit-learn is employed for applying LDA to the digits set [9].

3.1 New Set of Bases

In this part, LDA is applied to the training dataset, and eigenvectors are plotted.

In Figure 6, we can see all 9 components. These components is used to construct the new axes. Each component explain the variance and represent some pattern in the data [3]. However, some of the eigenvectors explain the variance more than the others. In Listing 1, percentage of variance explained by each selected component can be seen.

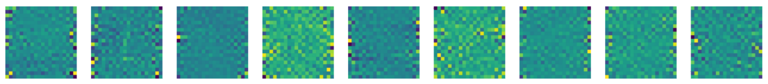

Figure 6: The Set of Bases

```
[73] lda.explained_variance_ratio_
array([0.26755747, 0.1958599 , 0.1554493 , 0.10666771, 0.09259101,
       0.06803125, 0.0581136 , 0.03290155, 0.02282822])
```

Listing 1: Explained Variance Ratio of the Components

3.2 Training Gaussian Classifier with Reduced Dimensionality

In this part, the dataset is split randomly, selecting half of each class's patterns for training and the remaining patterns for testing. Quadratic Discriminant Analysis library is used as Gaussian classifier from scikit-learn [9].

As shown in Figure 7, the classification error decreases in respect to the number of dimensionalities. The more dimensions there are, the better the model understands the patterns in data. The same decrease occurs in Figure 8, which means that the data does not overfit as the dimensions increase.

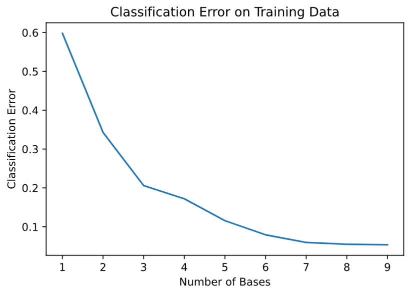

Figure 7: Classification Error on Training Set

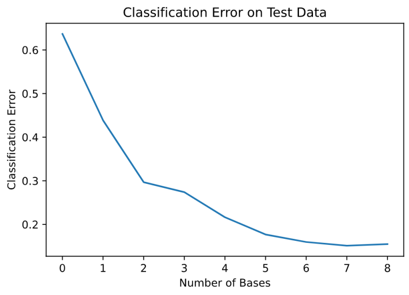

Figure 8: Classification Error on Test Set

4 Visualization of High Dimensional Space

In this section, the 400-dimensional data is visualized on 2 dimensional space. To do so, the following methods are used: Sammon Mapping and t-SNE. Sammon Mapping uses the idea of representing mutual distances between points. However, it does not focus on a better displacement [4]. On the other hand, t-SNE computes the similarity measure between pairs of samples in high dimensional and low dimensional space and tries to optimize these similarities with a cost function. It usually depicts a good separation between different clusters [12]. For plotting these visualization methods, Seaborn library is utilized [13].

4.1 Visualization with Sammon Mapping

In this part, the data is visualized with the aforementioned technique, Sammon Mapping. For the implementation, Tom Pollard's implementation ported to Python from MATLAB is used [10]. As for the set up, the parameters are shown in Listing 2. The default function uses PCA as the initialization method. As for the maximum number of iterations, when the number was increased to 1000, there was not a significant improvement in the error. Hence, the maximum iteration count was not changed.

```
display = 2, #0 to 2. 0 least verbose, 2 max verbose.
inputdist = 'raw', # {'raw','distance'}
maxhalves = 20, #maximum number of step halvings
maxiter = 500, #maximum number of iterations
tolfun = 1e-9, #relative tolerance on objective function
init = 'default' #{'pca', 'cmdscale', random', 'default'}, default
    is 'pca' if input is 'raw'
```

Listing 2: Parameters used in Sammon Mapping

In Figure 9, 400-dimensional data is mapped onto 2-dimensional space. As presented, while some points are more homogeneous, some are more mixed. And there is no clear separation between different classes. As mentioned above, Sammon Mapping does not focus on having the best displacement. However, it is good at keeping the structure and the pairwise distances.

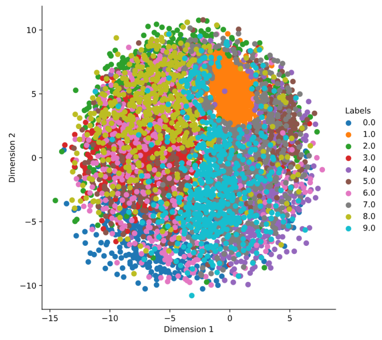

Figure 9: Data Visualized in Two Dimensional Space with Sammon Mapping

4.2 Visualization with t-SNE

In this part, the data is visualized with using t-SNE. For the set up, the parameters shown in Listing 3 are applied. Initialization parameter was changed to apply PCA before t-SNE. For more information, see the official description of the function [2].

```
display = 2, #0 to 2. 0 least verbose, 2 max verbose.
n_components = 2
perplexity = 30.0
early_exaggeration = 12.0
learning_rate = 200.0
n_iter = 1000
n_iter_without_progress = 300
min_grad_norm = 1e-7
metrics = 'euclidean'
init = 'pca' #initialization method
verbose = 0 #verbosity level
method= 'barnes_hut' #the gradient calculation algorithm uses
    Barnes-Hut approximation running in O(NlogN) time
angle = 0.5
#trade-off between speed and accuracy for Barnes-Hut T-SNE
square_distances = 'legacy'
```

Listing 3: Parameters used in Sammon Mapping

In Figure 10, 400-dimensional data is mapped onto 2-dimensional space. As shown, the data points are more homogeneous compared to Sammon Mapping's visualization and there are clear separations between different classes.

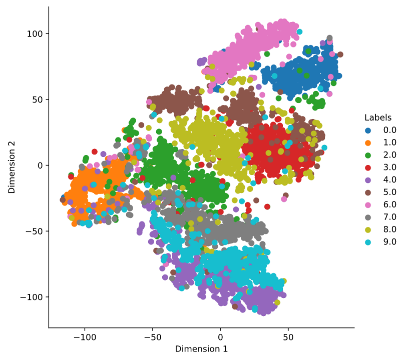

Figure 10: Data Visualized in Two Dimensional Space with t-SNE

References

[1] 1.2. linear and quadratic discriminant analysis.

[2] sklearn.manifold.tsne.

[3] Linear discriminant analysis, Aug 2014.

[4] Ufficio 10z. Sammon mapping, Jun 2019.

[5] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[6] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[7] Aman Kapri. Pca vs lda vs t-sne-let's understand the difference between them!, May 2020.

[8] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[10] Tompollard. tompollard/sammon, Jun 2019.

[11] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[12] Andre Violante. An introduction to t-sne with python example, Aug 2018.

[13] Michael Waskom, Olga Botvinnik, Drew O'Kane, Paul Hobson, Saulius Lukauskas, David C Gemperline, Tom Augspurger, Yaroslav Halchenko, John B. Cole, Jordi Warmenhoven, Julian de Ruiter, Cameron Pye, Stephan Hoyer, Jake Vanderplas, Santi Villalba, Gero Kunter, Eric Quintero, Pete Bachant, Marcel Martin, Kyle Meyer, Alistair Miles, Yoav Ram, Tal Yarkoni, Mike Lee Williams, Constantine Evans, Clark Fitzgerald, Brian, Chris Fonnesbeck, Antony Lee, and Adel Qalieh. mwaskom/seaborn: v0.8.1 (september 2017), September 2017.