Data Visualization Fun: An Exercise with PCA, t-SNE, and Violin Plots

Eric Becerril-Blas   
Department of Computer Science  
University of Nevada, Las Vegasbecere1@unlv.nevada.edu  
  
Mingon Kang  
Department of Computer Science  
University of Nevada, Las Vegasmingon.kang@unlv.edu

*Abstract*— Working with a dataset can be a daunting and difficult task for many planning to use it in machine learning applications. The process of understanding the madness behind the relationships of the various features that are associated with them not only make the data analysis process difficult but can also affect a machine learning model’s performance. In this paper we go through the visualization process of two very famous datasets: the MNIST dataset and the Boston Housing dataset.

Keywords— principle component analysis, t-distributed stochastic neighbor embedding, violin plot, machine learning, data visualization, computer science education, mnist

# Introduction

In this paper, we discuss how we play with the following data visualization tools to gain a better understanding of datasets: PCA plot, t-SNE plot, and Violin plots. The data visualization tools used are the scikit-learn PCA and TSNE libraries as well as the matplotlib violin plot feature. The programming language used for these visualizations was python.

# Visualization with Principle Component Analysis

## What is Principle Component Analysis

First, let’s gain an understanding of what principal component analysis is. Let’s say I want to analyze a dataset with a lot of variable. Something like the United States GDP. There would be a lot of variables to consider. This can present many problems such as understanding the relationships between each of these variables. Are there so many variables that I may be overfitting my model? All very important concerns. That is where principle component analysis comes into the picture, we use it to reduce the number of variables [1].

## Visualizing MNIST in 2D with PCA

Funding Agency: Eric Becerril-Blas’ Tuition Payment Funded This Paper

In order to show the dimension reducing power that PCA offers we decided to show it by utilizing the MNIST dataset. Besides the label, each one of these images has 28x28 dimensions, or variables as they are also referred to as. So, we utilize the scikit-learn PCA python library to reduce dimensionality to two. The lines of code are seen in Fig. 1.

A black sign with white text

Description automatically generated

1. Lines of code used to reduce dimensionality with PCA to 2 dimensions.

These lines of code take in a data-frame named images that has all of the pixel values needed to make each image. We then set n\_components = 2 and do pca.fit() on the images which reduces its dimensionality to 2. We then take the liberty of visualizing the 2-dimensional data in a graph using the matplotlib library so that it is easier to understand the clusters of each number. The data points are graphed as numbers and are color coordinated to see which class of number belongs with which. The function which did the graphing in my script is called plot\_2d\_data() and it takes in 3 parameters: the MNIST data reduced in 2 dimensions, the title of the graph, and the filename which the image will be saved as. The 2D PCA is below

A screenshot of a cell phone

Description automatically generated

1. 2D PCA of MNIST dataset.

## Visualizing MNIST in 3D with PCA

The exact same process was done for visualizing the MNIST dataset in 3 dimensions. The lines of code used for reducing the dimensionality of the dataset to 3 dimensions are in Fig. 3.

A close up of a sign

Description automatically generated

1. Lines of code used to reduce dimensionality with PCA to 3 dimensions.

Again, this takes in the data frame of pixel data and fits it to a PCA of 3 dimensions. A function named plot\_3d\_data() is then called which takes in the following parameters: MNIST data reduced in 3 dimensions, graph title, and the name which the image will be saved as. The datapoints are graphed as numbers and color coordinated to show which clusters of numbers they belong to shown in Fig. 4.

A close up of a piece of paper

Description automatically generated

1. 3d plot of the MNIST data using PCA

# t-Distributed Stochastic Neighbor Embedding

## What is t-Distributed Stochastic Neighbor Embedding

t-Distributed Stochastic Neighbor Embedding is an unsupervised, non-linear technique primarily used for data exploration and visualizing high dimensional data. This process is manifold learning algorithm in the scikit-learn python library that in essence constructs a probability distribution in a lower dimensional data space, making both the distributions as close as possible [3].

## Visualizing MNIST in 2D with t-SNE

In order to reduce the dimensionality of the MNIST dataset I used the TSNE function which is part of the scikit-learn manifold library. I made n\_components = 2 and then I fitted and transformed the data to reduced dimensionality utilizing the t-SNE library.



1. Lines of code used to reduce dimensionality with PCA to 2 dimensions usint t-SNE.

I then used the plot\_2d\_data() function in my script to get the visualization of MNIST which can be seen in Figure 6.

A screenshot of a cell phone

Description automatically generated

1. 2D t-SNE visualization of the MNIST data

## Visualzing MNIST in 3D

In order to reduce the dimensionality of the MNIST dataset I used the TSNE function which is part of the scikit-learn manifold library. I made n\_components = 3 and then I fitted and transformed the data to reduced dimensionality utilizing the t-SNE library.



1. Lines of code used to reduce dimensionality with PCA to 2 dimensions.

I then used the plot\_3d\_data() function in my script to get the visualization of MNIST which can be seen in Figure 8.

A close up of text on a white background

Description automatically generated

1. Lines of code used to reduce dimensionality with PCA to 2 dimensions.

# Violin Plots of Boston Housing Data

## How to make Violin Plots

Violin plots are a method of plotting numeric data and can be considered a combination of the boxplot with a kernel density plot [2].

## Boston Housing Data

The dataset used to demonstrate a violin plot is the Boston housing data taken from the UCI machine learning repository. Attribute information of this data is explained below in Table I.

1. Boston Housing Data

|  |  |
| --- | --- |
| CRIM | Per capita crime rate by town |
| ZN | Proportion of residential land zoned for lots over 25,000 sq.ft. |
| INDUS | Proportion of non-retail business acres per town |
| CHAS | Charles River dummy variable (= 1 if tract bounds river; 0 otherwise) |
| NOX | Nitric oxides concentration (parts per 10 million) |
| RM | Average number of rooms per dwelling |
| AGE | Proportion of owner-occupied units built prior to 1940 |
| DIS | Weighted distances to five Boston employment centres |
| RAD | Index of accessibility to radial highways |
| TAX | Full-value property-tax rate per $10,000 |
| PTRATIO | Pupil-teacher ratio by town |
| B | 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town |
| LSTAT | % lower status of the population |
| MEDV | Median value of owner-occupied homes in $1000's |

1. Column information of the boston housing data

In order to visualize this data we used the matplotlib violonplot() function. The code is seen in Fig. 10.

A screenshot of a cell phone

Description automatically generated

1. Code snippet which created the violin plot.

The output of the code snippet in Figure 10 is seen in Figure 11. With each row of data from Table I having its own violin plot.

A screenshot of a social media post

Description automatically generated

1. Violin plots of the boston housing data

# Conclusion

All in all, in this paper we have discussed the three main tasks were accomplished as well as the output of each of the required tasks.

##### Acknowledgment

We thank Matt Brems, Eryk Lewinson, and Vivek Palaniappan for their useful Medium articles. As well as Eric Becerril-Blas for funding his own tuition to learn this topic. Without his generous contribution, this paper would not exist.

##### References

1. Matt Brems, “A One-Stop Shop For Principal Component Analysis”. Medium Article
2. Eryk Lewinson, “Violin Plots Explained”. Medium Article
3. Vivek Palaniappan, “t-SNE: The Bits That No One Learns”. Medium Article