IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication June 30, 2023

Digital Object Identifier TBD

A Comparative Study of Principal Component Analysis on Different Datasets

PRATIGYA PAUDEL[1], SUSHANK GHIMIRE[1]
[1]Institute of Engineering, Thapathali Campus, Bagmati 44600 Nepal (e-mail: pratigyapaudel0@gmail.com)
Corresponding author: Pratigya Paudel (e-mail: pratigyapaudell0@gmail.com).

"This work was completed as a part of a college practical for Data Mining (CT725)."

ABSTRACT In the realm of modern data analysis, Principal Component Analysis (PCA) stands as a foundational and indispensable technique. Its widespread adoption and relevance in various domains testify to its significance in unraveling the complexities of datasets. The goals and principles of PCA are discussed and identified less than the weight it carries. The goal of this paper is to provide insights on the procedure of performing PCA on three datasets, the first of which will be a univariate normal distribution of 20 sample size and the latter two will be wine classification dataset and iris flower classification dataset, standard datasets collected from UCI machine learning repository. The paper then compares the results from performing PCA manually to the existing way of PCA with the use of scikit-learn libraries. The paper aims to discuss the fundamentals of PCA and the results of performing PCA on two different datasets.

INDEX TERMS Principal Component Analysis, Supervised Machine Learning, Wine Classification Dataset

I. INTRODUCTION

PRINCIPAL COMPONENT ANALYSIS (PCA) is a widespread technique for data analysis via the means of dimensionality reduction and data exploration. PCA helps to identify and segregate the important features or patterns from a high-dimensional dataset. PCA aims to transform the original dataset into a number of uncorrelated variables called principal components. PCA offers a straightforward approach to effectively reduce the dimensionality of intricate datasets, unveiling underlying simplified structures that may be concealed within them. This process requires minimal effort and provides a clear roadmap for extracting valuable insights. The goal of this paper is to provide valuable insights on PCA, and implement it with different datasets. We will begin with the theories and formulae involved with PCA, discussing relevant topics as we go along. We will continue by performing PCA on a randomly generated dataset sample using normal distribution and the standard wine classification dataset. By visualizing the results from the analysis, the dimensionality reduction can be interpreted and the dispersion of the dataset for the given targets can be analyzed.

II. METHODOLOGY

A. THEORY

Principal Component Analysis deals with ways to reduce the dimensionality of the data by scaling the dimensions down to the target number of dimensions using the best possible representation. The challenges and limitations that occcur while working with high-dimensional data is referred to as the "curse of dimensionality". There are some major issues that come forth with higher dimension data. Firstly, the increased dimension of the data brings about sparsity of the data points. The spread-out data is hard to analyze and to extract meaningful information from. Similarly, as the dimensions are increased, the distance between the data points tend to be similar, making it hard to distinguish similar and dissimilar data points. It usually calls for higher amount of data to compensate for the gap created from the newer dimensions. High dimension data is also computationally more demanding and requires more resources to be trained. Also, due to the huge gap between the data points, the chances of overfitting increases. PCA solves the problem of curse of dimensionality on feasible datasets using a step-by-step process to extract the necessary information, omitting the vague ones. By selecting the number of components for PCA, a vector with the dimensions of the number of components and the size of the data is created with the values to best represent the given data in orthogonal axes. The random numbers dataset is generated using the "random" library in numpy which in turn uses Mersenne Twister Algorithm to do so. With the standard PCA, the standardization of the dataset usually includes the subtraction of mean from the whole of the dataset with or without the division by standard deviation. The standard datasets used go through the same process, also

referrred to as fit transform. This process is computationally complex and thus requires a lot of time and space. Thus, for the standardization of the manually generated dataset, Randomized PCA is used where the dataset is randomly projected to another vector using elements from a standard Normal Distribution.

B. MATHEMATICAL FORMULAE

1) Generating a random projection matrix

The projection matrix used for standardization is made by selecting elements from a Standard Normal Distribution. The normal distribution follows the formula:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{1}$$

2) Transformation of datasets

Transformation of the synthetic dataset is done by multiplying the Randomly Generated Matrix of size 20x2 by the 2x2 matrix generated by selecting elements from a standard normal distribution.

$$TransformedData = \begin{bmatrix} a_1 & b_2 & \dots & a_{20} \\ a_1 & b_2 & \dots & b_{20} \end{bmatrix} @ \begin{bmatrix} c_1 & d_1 \\ c_2 & d_2 \end{bmatrix} \tag{2}$$

The standard dataset is passed through the fit-transform, which words as below:

$$TransformedData = \left(\frac{x-\mu}{\sigma}\right) \tag{3}$$

3) Calculation of Variance along dominant axis

$$\text{Variance} = \frac{\sum(x - \text{mean})^2}{N} \tag{4}$$

4) Covariance Matrix

$$S_X = \frac{1}{m-1}(X^T \cdot X) \tag{5}$$

5) Eigenvectors and eigenvalues

For a given square matrix B, the equation for eigenvectors and eigenvalues is:

$$B \cdot \mathbf{v} = \lambda \mathbf{v} \tag{6}$$

In Equation 6, $\mathbf{v}$ represents an eigenvector and λ is the corresponding eigenvalue.

To find the eigenvalues, we rearrange the equation as:

$$(B - \lambda_i I) \cdot \mathbf{v}_i = \mathbf{0} \tag{7}$$

6) Proportion of Variance

The proportion of variance for each of the eigenvalue can be calculated as:

$$\text{Proportion of Variance} = \frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i} \tag{8}$$

C. SYSTEM BLOCK DIAGRAM

The system workflow is as shown below:

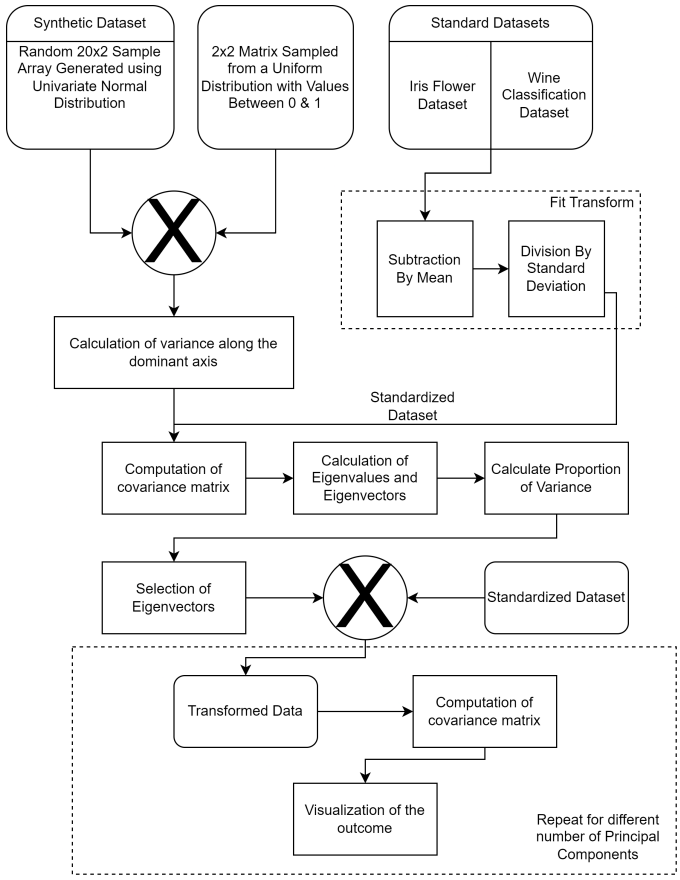

FIGURE 1. PCA Workflow

D. INSTRUMENTATION TOOLS

The entirety of the process is done using Python. Google Colab, short for Google Colaboratory, is an online platform provided by Google for running and sharing Jupyter notebook environments and it was used for all of the coding. Google colab provides a number of built-in functions for data analysis. A number of python libraries have been used to perform PCA. Firstly, the library numpy is used to generate a random sample of 20x2. The function *randn* is used to generate a random sample. The projection matrix for the dataset is prepared using another function *rand* which is used to sample random numbers from a normal distribution. Matrix multiplication is performed using the numpy function *matmul*. *var* and *cov* are used to determine the variance of the data along the dominant axis and the covariance matrix respectively. Within numpy, *linalg* allows calculation of eigenvectors and eigen values using the function *eig*. These are the most significant functions used for the analysis. The standard datasets are loaded through *scikit-learn* using *load-iris* and *load-wine* functions. Another popular library, *Pandas* is used to hold the data values and present the data in tabular form. The proper analysis is then visualized through some graphics libraries like *matplotlib* and *seaborn*. Finally, the obtained PCA results are compared to the PCA performed by *PCA* function within

scikit-learn.

E. WORKING PRINCIPLE

1) Dataset Preparation

Two standard datasets are collected from the scikit-library. The iris and wine classification datasets are imported using the library. The data is then standardized using the fit-transform function to obtain normalized data with zero mean and standard deviation of one. A random dataset is synthesized using numpy libraries and the data is then transformed using a projection matrix formed using data points from a standard normal distribution.

2) Eigenvector Computation

The datasets are then used to compute their covariance. Using the covariance matrix, eigenvalues and eigenvectors are obtained. The proportion of variance is also calculated and the eigen vectors are selected as the principal components for analysis. The dot product between the selected component and the standardized dataset is used to analyze the dataset and transform it.

3) Data Analysis and Visualization

The transformed data is used to compute the covariance matrix which stores the result of PCA. The outcome is then visualized using different python data visualization libraries. The process is repeated with different combination of principal components selected for the data analysis and visualization.

III. RESULTS

A. PCA ON RANDOM DATASET

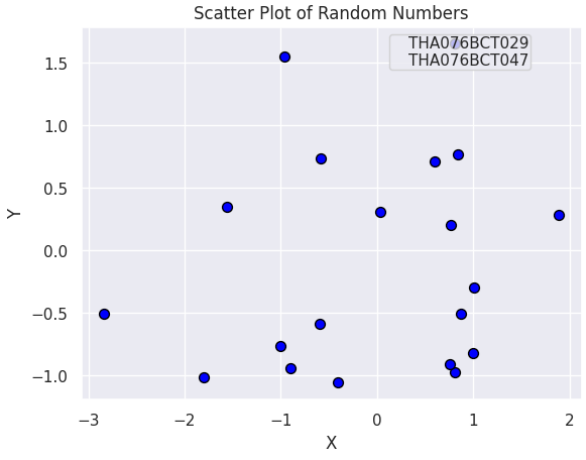

FIGURE 2. Plot of the random numbers

We conducted a PCA analysis on a randomly generated dataset to examine how effectively PCA reduces the dimensions of synthetic data. By creating scatter plots, we assessed the extent of dimensionality reduction and the variance explained by the principal components. We visualized the data in two scenarios: first, when reduced to 1D using each principal component individually, and second, when reduced to 2D using both principal components simultaneously. Figure 2 shows a plot of the numbers initialized randomly.

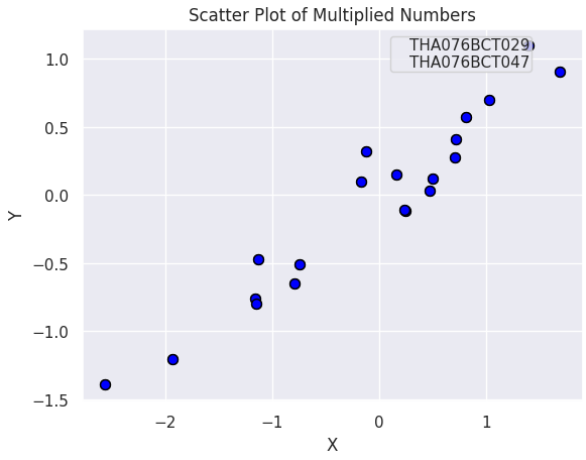

FIGURE 3. Transformation using a matrix sampled from Normal Distribution

Figure 3 shows a plot of the random numbers that have been transformed using a 2x2 matrix initialized by selecting points from a normal distribution. The transformation helps the dataset to converge in a direction.

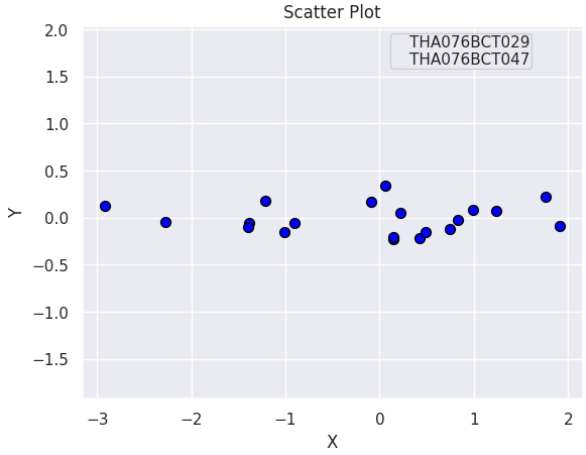

FIGURE 4. Application of PCA with two principle components

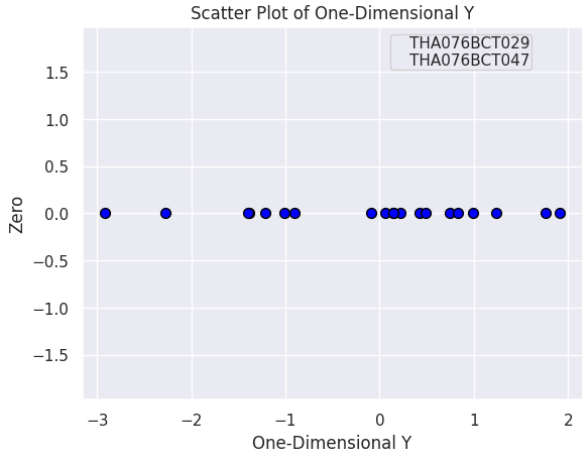

FIGURE 5. Application of PCA with one principle component

The results from the application of 2 principle components

for the analysis can be seen in Figure 4. The entirety of the data lies around the x-axis in the output.

The results from the application of 1 principle component for the analysis can be seen in Figure 5. Since only one principal component has been used, there's a single axis to plot the graph. This figure also demonstrates the idea of dimensionality reduction as enforced by PCA.

B. PCA ON IRIS DATASET

We conducted PCA on the well-known Iris dataset obtained from the scikit-learn library. The dataset was initially visualized using a table that displayed the various attributes associated with each target. The Iris dataset consists of 4 continuous-value attributes for each data entry, along with their corresponding target classes.

TABLE 1. Iris Dataset (Attributes as Rows)

Attribute	Sample 1	Sample 2	Sample 3
Sepal Length	5.1	6.3	7.6
Sepal Width	3.5	2.8	3.0
Petal Length	1.4	5.1	6.6
Petal Width	0.2	1.5	2.1

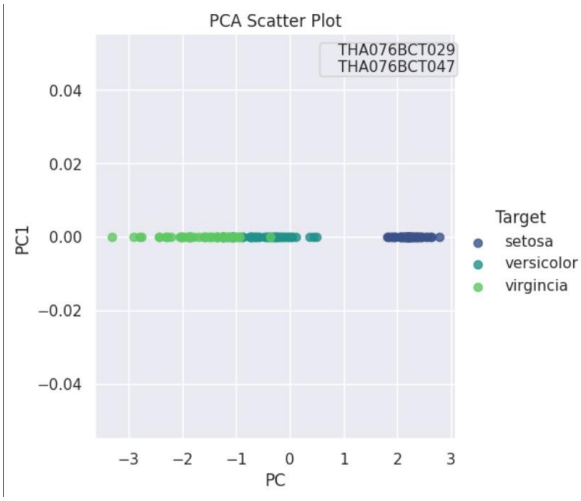

FIGURE 6. Application of PCA with one principal component on Iris Dataset

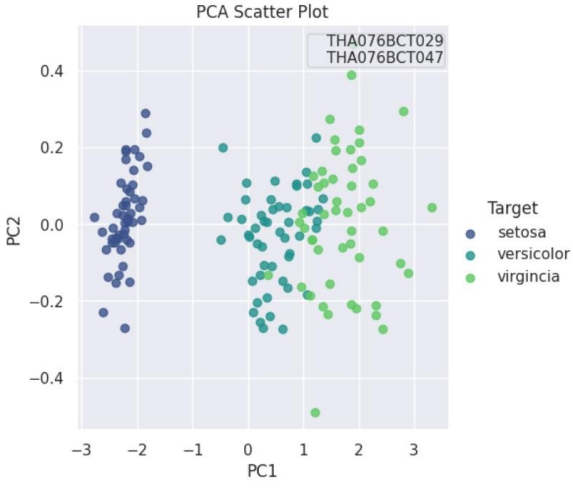

FIGURE 7. Best-case scenario two components analysis on Iris Dataset

FIGURE 8. Worst-case scenario two components analysis on Iris Dataset

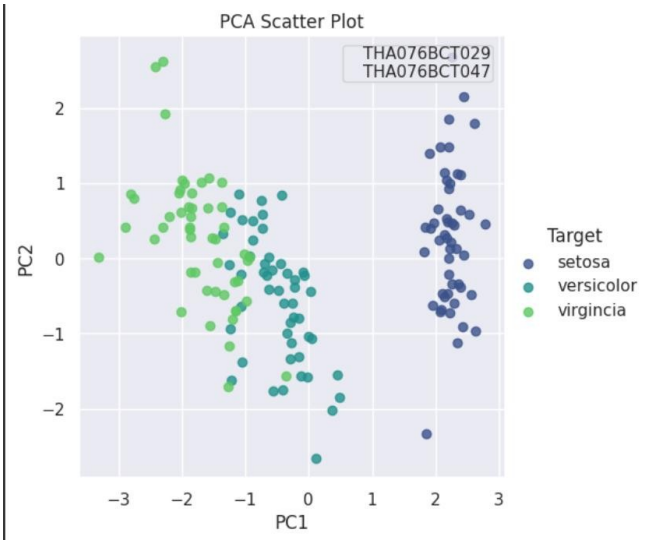

FIGURE 9. Two components analysis using scikit-learn on Iris Dataset

Figure 6 shows the application of PCA using the best principal component on the dataset. As seen from the figure, the target class 'setosa' is easily separable from the data but the other classes are not distinguishable using a single component. Similarly, the best case scenario while using two principal components is already able to separate the data classes much better. Figure 7 shows the classes and also conforms to the dataset pattern as obtained from a single principal component. The worst case for using two components tells a tale much different. The classes are not distinguishable and the features aren't learnt well as shown in Figure 8. Finally the results obtained by using the PCA functionality from scikit-library as is in Figure 9 shows a plot similar to Figure 7 with inverted x-axis. Finally, the data classes are the most separable when using three principal components as evident from Figure 10. It also reciprocates with the findings from using a single and two principal components. The worst case scenario while using three principal components displays a

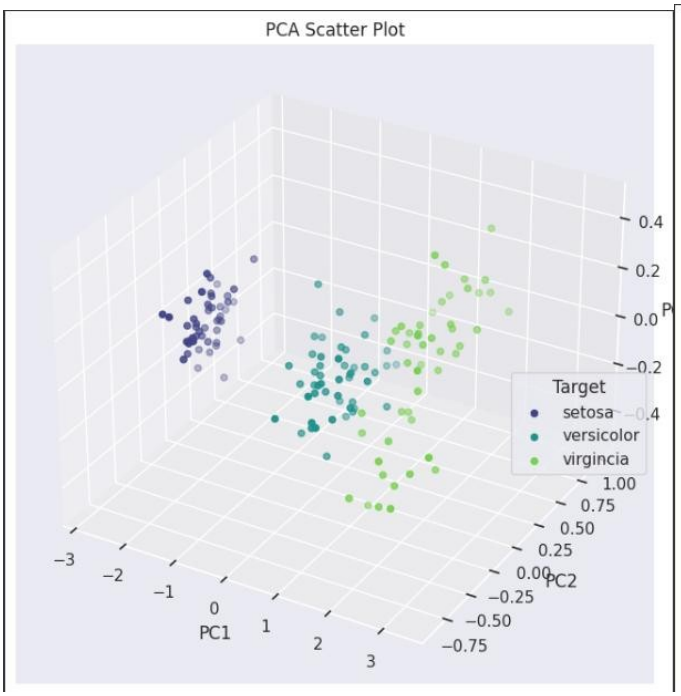

FIGURE 10. Best-case scenario three components analysis on Iris Dataset

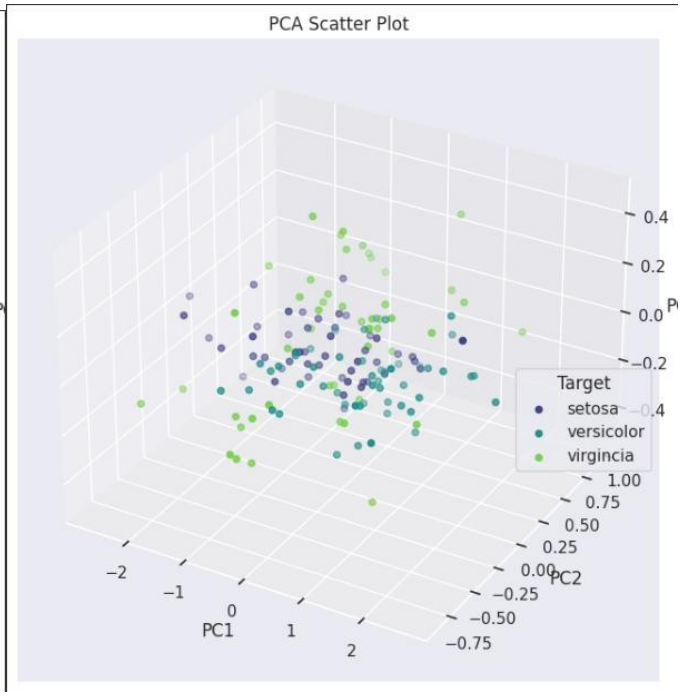

FIGURE 11. Worst-case scenario three components analysis on Iris Dataset

plot, where the classes cannot be identified.

C. PCA ON WINE DATASET

We performed PCA on the standard wine dataset obtained from the scikit-learn library. The dataset was firstly visualized through a table with the different attributes linked with each of the targets. There are 13 continuous-value attributes attached with each of the data and the corresponding target class. The higher number of attributes in the wine classification dataset does introduce more challenges to learn new features and variation in the data than the Iris dataset which had only 4. The performance of PCA on wine classification dataset is plotted for different number of principal components along with their best and worst cases.

TABLE 2. Wine Classification Dataset Sample

Attribute	Sample 1	Sample 2	Sample 3
Alcohol (%)	13.24	12.37	14.06
Malic Acid (g)	1.71	1.17	2.15
Ash (g)	2.64	2.32	2.61
Alkalinity of Ash	15.5	23.0	17.0
Magnesium (mg)	127	88	121
Total Phenol	2.8	2.22	2.51
Flavanoids	3.06	2.45	2.61
Nonflavanoid Phenols	0.28	0.26	0.31
Proanthocyanins	2.29	1.9	1.25
Color Intensity	5.64	4.54	5.05
Hue	1.04	1.06	1.06
OD280/OD315 of Diluted Wines	3.92	3.21	3.58
Proline	1065	938	1050
Target	Class 1	Class 2	Class 3

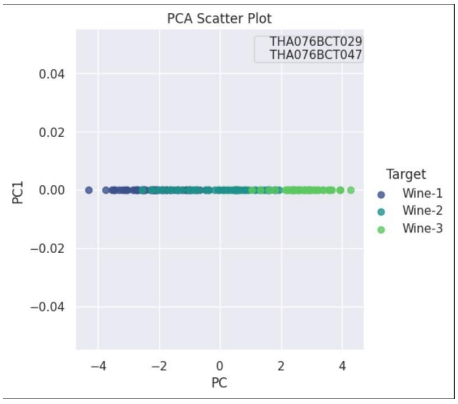

FIGURE 12. Application of PCA with one principle component on Wine Dataset

FIGURE 13. Two components analysis using scikit-learn on Wine Dataset

FIGURE 14. Best-case scenario two components analysis on Wine Dataset

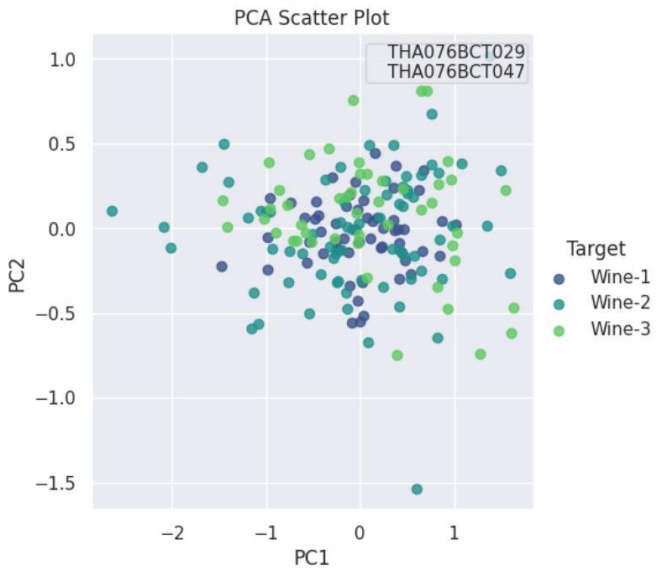

FIGURE 15. Worst-case scenario two components analysis on Wine Dataset

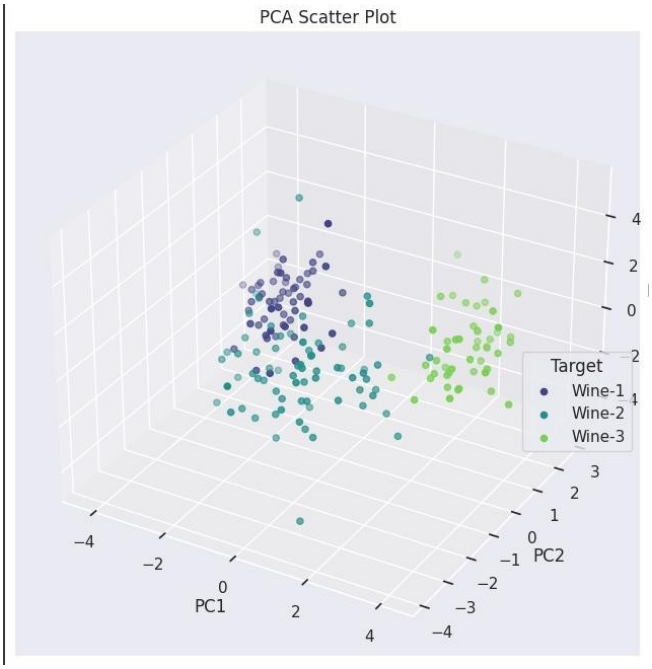

FIGURE 16. Best-case scenario three components analysis on Wine Dataset

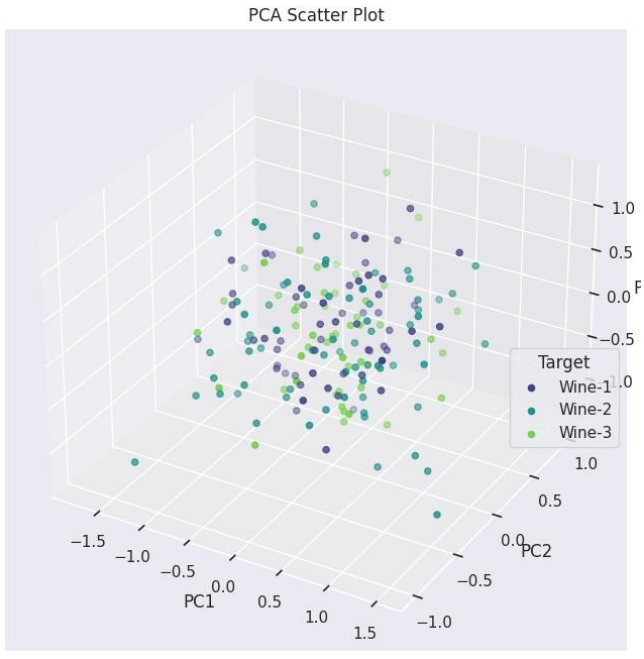

FIGURE 17. Worst-case scenario three components analysis

The results as shown in the Figure 12 shows how the dataset is almost separable into different classes with the application of a single principal component. This result is better than the one obtained with Iris Dataset. The following Figure 13 is the result obtained by performing PCA using the function provided by the scikit-learn library. Although there is some overlap between the wine classes, most of the data points lie in separable spaces. This figure is just like the best case scenario obtained for two principal componenets as in Figure 14 with a vertical flip. The worst case scenario shows how the features have not been learnt and the classes are not separable at all as in Figure 15.

Finally, the data points can also be plotted in a 3d graph using three principal components. The classes can be easily identified and is in conformation with the prior gained information from using one and two principal componenets. On the other hand, the worst case scenario shows data classes overlapped with each other to the point where none of the classes can be identified properly. The features are not learnt well with this combination of principal components, as is presented in

Figure 17.

IV. DISCUSSION AND ANALYSIS

The previous sections demonstrated the successful usage of Principal Component Analysis(PCA) for dimensionality reduction. Using the forementioned principles, PCA was performed on different datasets and the obtained results were plotted in different graphs. The application of PCA on the datasets was able to lower the dimensions of the matrix. The covariance matrix obtained as the subsequent result after the matrix transformation displayed a reduction in off-diagnol elements or low value elements and increase in the diagonal elements. The obtained result was comparable to the goal of PCA.

A number of notable features were learnt from the results of PCA. The general trend seen with the eigenvalues of principal components selected and the amount of distinguishable classes from the results is that they're directly proportional. Higher eigenvalues generally had better results distinguishing the data then the corresponding lower eigenvalues. This, the general idea would be to use the higher eigenvalues. However, there are still some outliers that need acknowledgement that perform better than some other principal components with higher eigenvalues.

While higher number of principal components selected to separate the given data should generally result in better class separability, it cannot be blindly followed as evident from the previous figures. In a lot of cases, taking only two prinpical components have better performance than taking three principal components. Clearly, even though the higher number of principal components is able to capture the features and different variations, they might not be able to cover the class boundaries. The higher number of principal components may also cause the capture of unnecessary features and noise. Thus, the selection of number of prinpical components is required with different cases to obtain the best results.

Thus, the best results from PCA can only be obtained by comparing and testing for the different eigenvalues and the principal components to select. The results also show that valuable features and information can be obtained from the dataset with the application of PCA

V. CONCLUSION

This lab was conducted coherent to the principles of PCA. The application of PCA on a randomized dataset and two standard datasets shows its effectiveness over different types of data. The main purpose of PCA i.e. Dimensionality Reduction was performed successfully and effectively represented using different number of principal componenets. By focusing on the principal components, PCA was able to get rid of noise and other irrelevant information with the correct selection. It has been evident from the analysis that the larger number of principal components helps improve the performance of PCA. However, PCA does come with its own shortcomings. Firstly, the analysis on the random dataset visualized the biasedness of PCA in case of any outliers. When using lesser number of principal componenets, the data becomes less separable. Also, since the principal components have the assumption of linearity, the relationships and the pattern from the data is not always learnt.

VI. REFERENCES

- J. Shlens, "A Tutorial on Principal Component Analysis," in *arXiv preprint arXiv:1404.1100*, 2014.
- B. Komer, J. Bergstra, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn," in *arXiv preprint arXiv:1402.5184*, 2014.

PRATIGYA PAUDEL is a fourth year student, studying computer engineering under IOE, Thapathali Campus. She has been involved in a lot of machine learning projects and has a keen eye for data analysis and AI related stuff. With the enthusiasm for Artificial Intelligence (AI), she is driven by the potential of AI to transform industries and tackle complex challenges. Her academic journey has equipped her with a strong foundation in AI concepts, including machine learning and data analysis. She possesses a relentless curiosity and is always eager to explore the latest advancements in AI. Her goal is to apply her knowledge and make a meaningful contribution in the field.

SUSHANK GHIMIRE is a fourth year student, studying computer engineering under IOE, Thapathali Campus. He possesses a lot of interest, working with data. His educational path has provided him with a solid understanding of AI concepts, encompassing machine learning and data analysis. He possesses an unwavering curiosity and is constantly eager to delve into the latest advancements in AI. His objective is to leverage his knowledge and expertise to create a significant impact in the field.