

PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and
Technology (Deemed to be University)

Submitted in partial fulfillment of requirements for the award
of Bachelor of Technology Degree in Information Technology

By-

NANDANA. S. NAIR

Reg. No : 39120074



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

NOVEMBER 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of **NANDANA. S. NAIR (Reg. No: 39120074)** who carried out the project entitled “**Using various dimension, reduction analysis in PCA**” under my supervision from October 2021 to November 2021.

Internal Guide

Dr. P. Jeyanthi

DECLARATION

I, **NANDANA. S. NAIR** hereby declare that the project report entitled **Using various dimension, reduction analysis in PCA** done by me under the guidance of **Dr. P. Jeyanthi** is submitted in partial fulfillment of requirements for the award of Bachelor of Technology Degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. R. Subhashini M.E., Ph.D., Head of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. P. Jeyanthi M.E., Ph.D.** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



ABSTRACT

In the modern age of technology, increasing amounts of data are produced and collected. In machine learning, however, too much data can be a bad thing. At a certain point, more features or dimensions can decrease a model's accuracy since there is more data that needs to be generalized – this is known as the **curse of dimensionality**. **Dimensionality reduction** is a way to reduce the complexity of a model and avoid overfitting.

Principal component analysis (PCA) is a dimensionality reduction technique for such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance. Finding such new variables, the principal components, reduces to solving an eigenvalue/eigenvector problem, and the new variables are defined by the dataset at hand, not a priori, hence making PCA an adaptive data analysis technique. It is adaptive in another sense too, since variants of the technique have been developed that are tailored to various different data types and structures.

In this project, will begin by introducing the basic ideas of PCA, discussing what it can and cannot do. Then we will use the **Wine** dataset to apply **Principal Component Analysis (PCA)** algorithm on it and compress the dataset onto a lower-dimensional feature subspace with the goal of maintaining most of the relevant information.

TABLE OF CONTENTS		
CHAPTER	TITLE	PAGE NO
	ABSTRACT	06
1	INTRODUCTION	08
2	SYSTEM ANALYSIS	09
	2.1 PROPOSED SYSTEM	09
3	SYSTEM SPECIFICATION	10
	3.1 HARDWARE SPECIFICATION	10
	3.2 SOFTWARE SPECIFICATION	10
4	MODEL IMPLEMENTATION	12
	4.1 SYSTEM ARCHITECTURE	12
	4.2 MODULES	13
5	SYSTEM DESIGN	19
	5.1 FLOWCHART DIAGRAM	19
6	CONCLUSION	20
	REFERENCES	21
	APPENDIX A: SNAP SHOTS	22
	APPENDIX B: CODING	25

Fig No	Figure Name	Page No.
1	Fig - 1	09
2	Fig - 2	12
3	Fig - 3	15
4	Fig - 4	19

CHAPTER - 1

INTRODUCTION

Large datasets are increasingly widespread in many disciplines. In order to interpret such datasets, methods are required to drastically reduce their dimensionality in an interpretable way, such that most of the information in the data is preserved. Many techniques have been developed for this purpose, but Principal Component Analysis (PCA) is one of the oldest and most widely used. Its idea is simple – reduce the dimensionality of a dataset, while preserving as much ‘variability’ (i.e. statistical information) as possible. Although it is used, and has sometimes been reinvented, in many different disciplines it is a statistical technique and hence much of its development has been by statisticians.

Principal Component Analysis (PCA) is an unsupervised linear transformation technique that is widely used across different fields, most prominently for feature extraction and dimensionality reduction. Other popular applications of PCA include exploratory data analyses and denoising of signals in stock market trading, and the analysis of genome data and gene expression levels in the field of bioinformatics.

PCA helps us to identify patterns in data based on the correlation between features. Basically, PCA aims to find the directions of maximum variance in high-dimensional data and projects it onto a new subspace with equal or fewer dimensions than the original one. Although for inferential purposes a multivariate normal (Gaussian) distribution of the dataset is usually assumed, PCA as a descriptive tool needs no distributional assumptions and, as such, is very much an adaptive exploratory method which can be used on numerical data of various types. Indeed, many adaptations of the basic methodology for different data types and structures have been developed

The earliest literature on PCA dates from Pearson and Hotelling, but it was not until electronic computers became widely available decades later that it was computationally feasible to use it on datasets that were not trivially small. Since then its use has burgeoned and a large number of variants have been developed in many different disciplines.

CHAPTER - 2

SYSTEM ANALYSIS

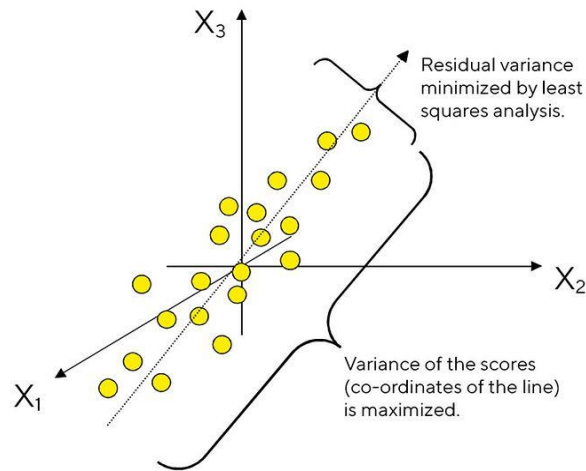
2.1 PROPOSED SYSTEM

PCA is a **Machine Learning dimensionality reduction algorithm**. Machine learning is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. The true purpose of PCA is mainly to decrease the complexity of the model. It is to simplify the model while maintaining the relevance and performance of the model.

Statistically, PCA finds lines, planes and hyper-planes in the K-dimensional space that approximate the data as well as possible in the least squares sense. A line or plane that is the least squares approximation of a set of data points makes the variance of the coordinates on the line or plane as large as possible.

In this project, I will perform PCA step by step using Python's inbuilt ML library **scikit-learn**. I will be using the **Wine dataset** from The UCI Machine Learning Repository. This dataset consists of 178 wine samples with 13 features describing their different chemical properties.

(Fig – 1)



CHAPTER - 3

SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

Processor	:	Processor Intel CORE i3 and above
Internet Connection	:	Existing telephone lines, Data card, Fiber net
RAM	:	4 GB

3.2 SOFTWARE SPECIFICATION

Operating System	:	Windows, Mac, Linux
Language	:	Python Programming – Python 3.9.6
GUI	:	Jupyter Notebook (Anaconda3)

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. It supports multiple programming paradigms, including **structured** (particularly, **procedural**), **object-oriented** and **functional programming**. Python was conceived in the late

1980s by **Guido van Rossum** at Centrum Wiskunde & Informatica (CWI) in the **Netherlands** as a successor to ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system.

Python is used for-

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Anaconda Navigator is a **desktop graphical user interface (GUI) included in Anaconda® distribution** that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- Spyder
- PyCharm
- VSCode
- Glueviz
- Orange 3 App
- RStudio
- Anaconda Prompt (Windows only)

- Anaconda PowerShell (Windows only)

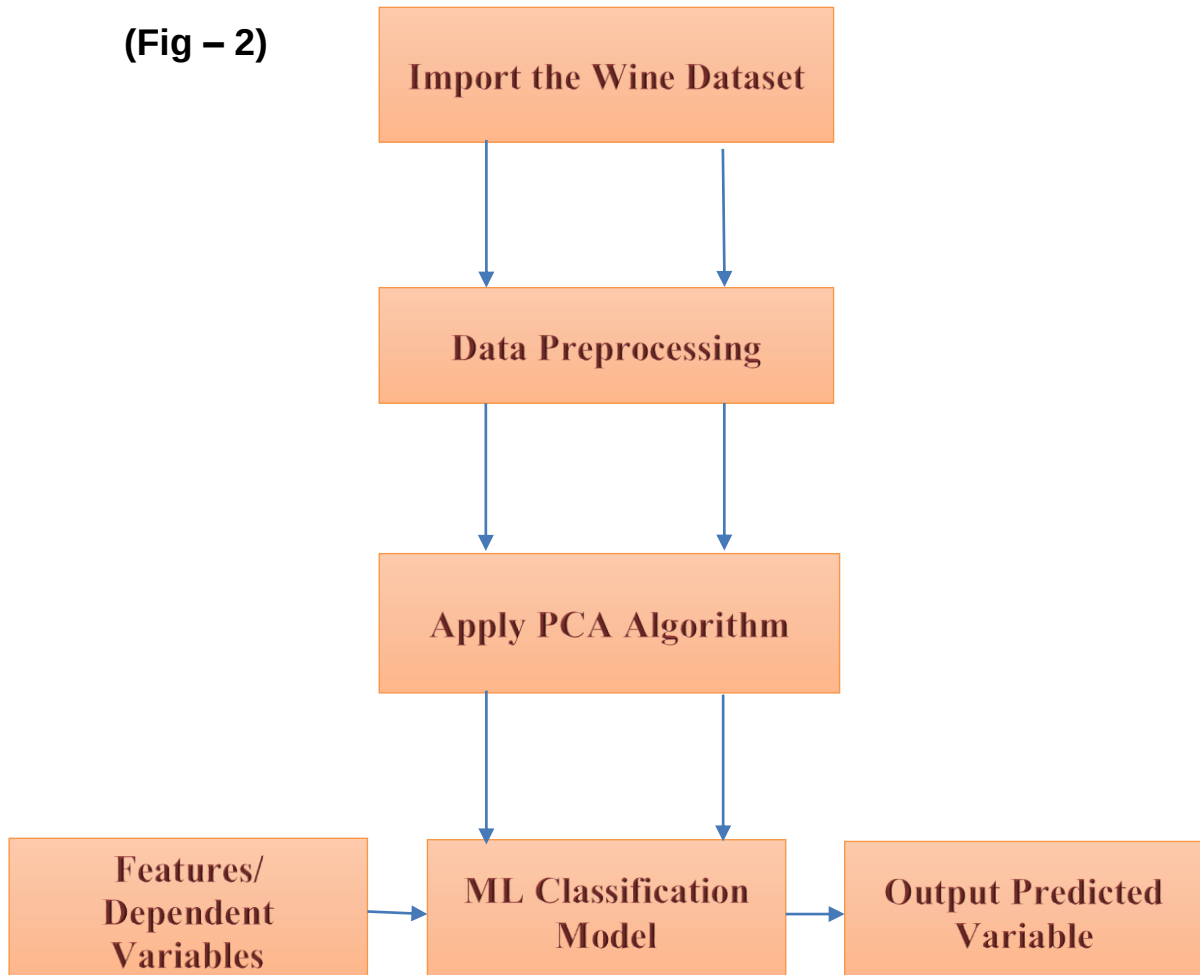
The **Jupyter Notebook** is one of the applications available in the **Anaconda navigator**. It is a server-client application that allows editing and running notebook documents via a web browser. Jupyter Notebook can be executed on a local desktop requiring no internet access or can be installed on a remote server and accessed through the internet.

CHAPTER - 4

MODEL IMPLEMENTATION

4.1 SYSTEM ARCHITECTURE

(Fig – 2)



The **Fig - 1** shows the architectural diagram of this project. It defines the **flow of data** for dimensionality reduction of the **Wine** dataset. The first step is importing the dataset, data has to be loaded into the Jupyter Notebook Python environment for analysis. The second step is, preprocessing of the collected raw data into understandable format. **Standardization and Normalization** are the techniques which are used to transform the data of various formats into a common format to perform PCA. Third step is to apply PCA algorithm on this processed data for getting dataset with reduced dimensions. Then we can apply any Classification algorithm and develop a ML model for predictions.

4.2 MODULES

There are four Modules -

- Importing the Wine Dataset
- Data Preprocessing
- Reducing Dimensions using Principal Component Analysis
- Visualization of dimensionally reduced dataset

4.2.1 IMPORTING THE WINE DATASET

Data is available in different file formats like .txt, .csv, .xlsx, .spss etc. The data needed has to be loaded into the Python environment for analysis. Once data has been extracted from the file it should be stored in a data frame.

Also, packages and libraries necessary for applying PCA algorithm need to be imported like, **Pandas**, **Numpy** & **Scikit-learn**. For visualization import Python's **Matplotlib** library.

4.2.2 DATA PREPROCESSING

Data preprocessing is the data mining technique that involves transforming raw data into understandable or a common format. The raw data is highly susceptible to noise, missing values, and inconsistency.

Real world data is often incomplete, inconsistent, and is likely to contain errors. The missing values problem has to be solved by simple statistical techniques. Data preprocessing is the best method for resolving such issues. In order to improve the quality of the data consequently, the mining results of raw data is preprocessed so the efficiency process improves.

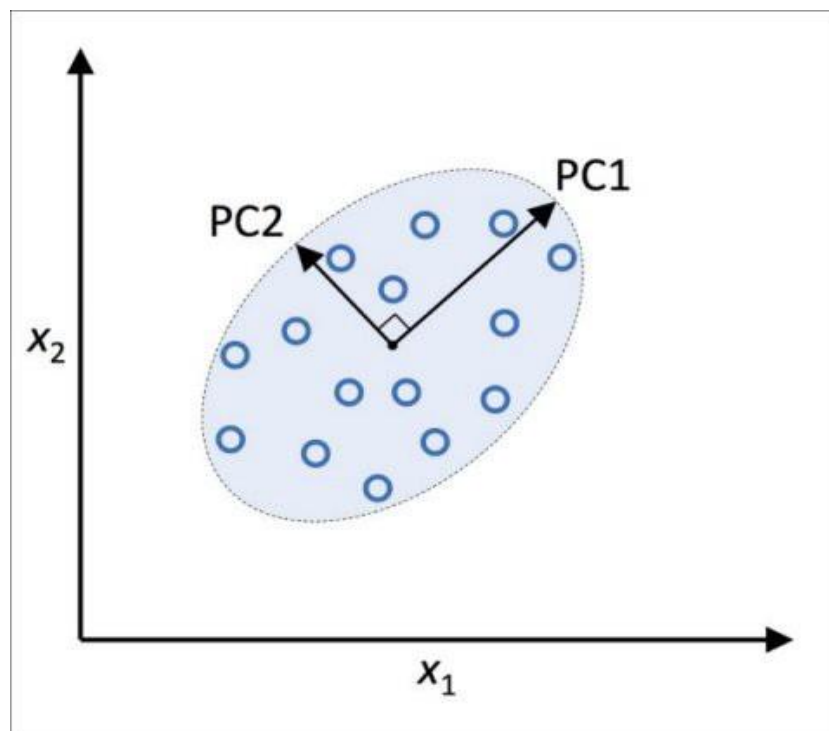
Standardization is one of the main techniques used for converting raw data in different formats into a common format. It enhances the comparison process, allows for collaborative research and large scale analytics.

4.2.3 REDUCING DIMENSIONS USING PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) is a **linear dimensionality reduction** technique that can be utilized for extracting information from a high-dimensional space by projecting it into a lower-dimensional subspace. It tries to preserve the essential parts that have more variation of the data and remove the non-essential parts with fewer variation. Dimensions are nothing but features that represent the data. According

to *Wikipedia*, PCA is a **statistical** procedure that uses an orthogonal transformation to convert possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

The orthogonal axes (**principal components**) of the new subspace can be interpreted as the directions of maximum variance given the constraint that the new feature axes are orthogonal to each other, as illustrated in the following figure:



(Fig – 3)

In the preceding figure, x_1 and x_2 are the original feature axes, and **PC1** and **PC2** are the principal components.

If we use PCA for dimensionality reduction, we construct a $d \times k$ -**dimensional transformation matrix** W that allows us to map a sample vector x onto a new k -dimensional feature subspace that has fewer dimensions than the original d -dimensional feature space. As a result of

transforming the original d -dimensional data onto this new k -dimensional subspace (typically $k \ll d$), the first principal component will have the largest possible variance. Even if the input features are correlated, the resulting **principal components** will be mutually orthogonal (**uncorrelated**).

Steps in PCA Algorithm are:

1. Standardize the d -dimensional dataset.
2. Construct the covariance matrix.
3. Decompose the covariance matrix into its eigenvectors and eigenvalues.
4. Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors.
5. Select k eigenvectors which correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace ($k \leq d$).
6. Construct a projection matrix W from the “top” k eigenvectors.
7. Transform the d -dimensional input dataset X using the projection matrix W to obtain the new k -dimensional feature subspace.

After completing the mandatory preprocessing, we go to the second step-constructing the covariance matrix. The symmetric $d \times d$ -dimensional covariance matrix, where d is number of dimensions in the dataset, stores the pairwise covariances between the different features. For example, the covariance between two features x_j and x_k on the population level can be calculated via the following equation:

$$\sigma_{jk} = \frac{1}{n} \sum_{i=1}^n (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$$

Here, μ_j and μ_k are the sample means of features j and k , respectively.

A positive covariance between two features indicates that the features increase or decrease together, whereas a negative covariance indicates that the features vary in opposite directions. The eigenvectors of the covariance matrix represent the principal components, whereas the corresponding eigenvalues will define their magnitude. In the case of the *Wine* dataset, we would obtain 13 eigenvectors and eigenvalues from the 13×13 -dimensional covariance matrix.

Next let's obtain the eigenpairs of the covariance matrix. An eigenvector v satisfies the following condition:

$$\Sigma v = \lambda v$$

Here, λ is a scalar: the eigenvalue.

Since we want to reduce the dimensionality of our dataset by compressing it onto a new feature subspace, we only select the subset of the eigenvectors (principal components) that contains most of the information (variance). But before we collect those k most informative eigenvectors, first plot the **Variance Explained Ratios** of the eigenvalues.

After we have successfully decomposed the covariance matrix into eigenpairs, let's now proceed with the last three steps of PCA. We will sort the eigenpairs by descending order of the eigenvalues, construct a

projection matrix from the selected eigenvectors, and use the projection matrix to transform the data onto the lower-dimensional subspace. Using the projection matrix, we can now transform a sample X (represented as a 124 x 13-dimensional row vector for our training dataset) onto the PCA subspace (the principal components one and two) obtaining X' , now a two-dimensional sample vector consisting of two new features:

$$X' = XW$$

Since the manual computation of eigenvectors and eigenvalues, and all these processes is somewhat tedious and an elaborate task, we will use the *PCA class* of **scikit-learn** library for all these tasks.

4.2.4 VISUALIZATION OF DIMENSIONALLY REDUCED DATASET

Lastly, visualize the transformed *Wine* training set, now stored as an 124 x 2-dimensional matrix, in a two-dimensional scatterplot. We can easily do this importing scikit-learn Matplotlib library and using its various functions.

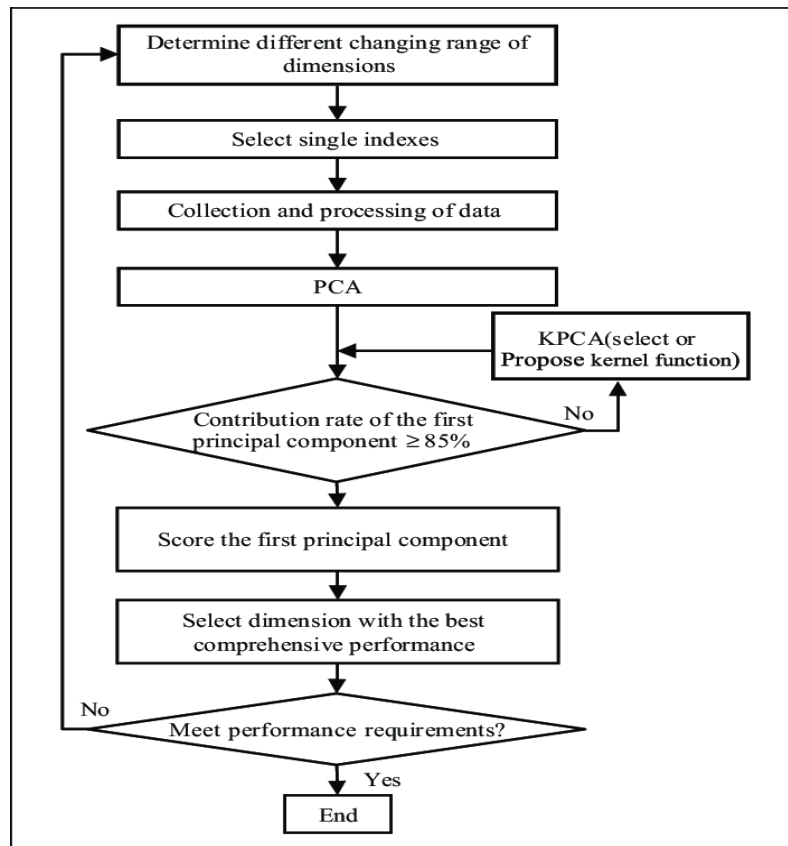
CHAPTER - 5

SYSTEM DESIGN

5.1 FLOWCHART DIAGRAM

A [flowchart](#) is a graphical representation of an algorithm. Programmers often use it as a program-planning tool to solve a problem. It makes use of symbols which are connected among them to indicate the flow of information and processing. The process of drawing a flowchart for an algorithm is known as “flowcharting”.

(Fig – 4)



CHAPTER - 6

CONCLUSION

Although PCA in its standard form is a widely used and adaptive descriptive data analysis tool, it also has many adaptations of its own that make it useful to a wide variety of situations and data types in numerous disciplines. Adaptations of PCA have been proposed, among others, for binary data, ordinal data, compositional data, discrete data, symbolic data or data with special structure, such as time series or datasets with common covariance matrices.

PCA or PCA-related approaches have also played an important direct

role in other statistical methods, such as Linear Regression (with principal component regression) and even simultaneous clustering of both individuals and variables. Methods such as correspondence analysis, canonical correlation analysis or linear discriminant analysis may be only loosely connected to PCA, but, insofar as they are based on factorial decompositions of certain matrices, they share a common approach with PCA.

The literature on PCA is vast and spans many disciplines. Space constraints mean that it has been explored very superficially here. New adaptations and methodological results, as well as applications, are still appearing.

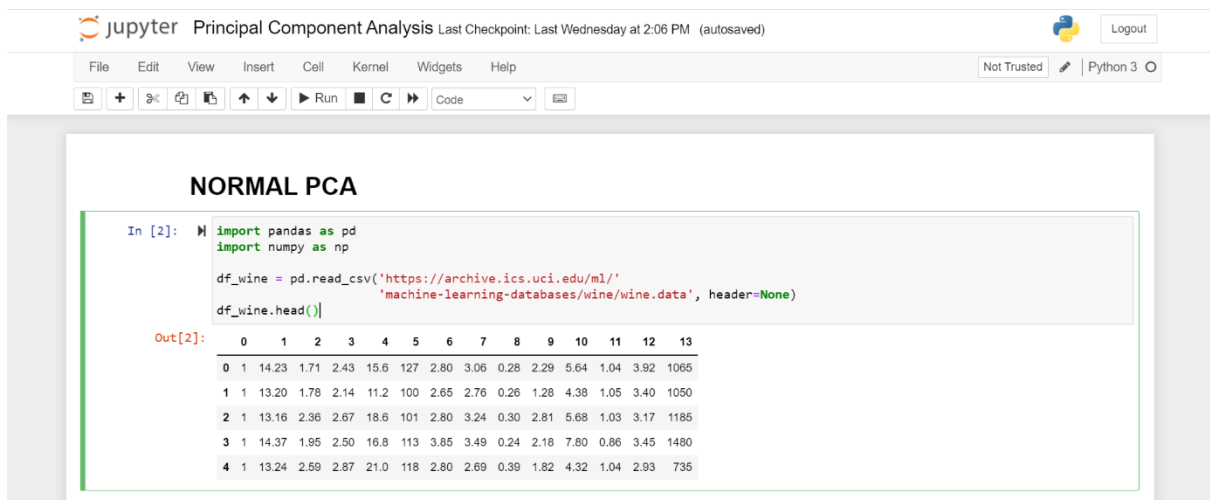
REFERENCES

1. Pearson K. 1901. On lines and planes of closest fit to systems of points in space. *Phil. Mag.* 2, 559–572. (10.1080/14786440109462720) [[CrossRef](#)] [[Google Scholar](#)]
2. Hotelling H. 1933. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* 24, 417–441, 498–520 (10.1037/h0071325) [[CrossRef](#)] [[Google Scholar](#)]
3. Jackson JE. 1991. *A user's guide to principal components*. New York, NY: Wiley. [[Google Scholar](#)]
4. Jolliffe IT. 2002. *Principal component analysis*, 2nd edn New York, NY: Springer-Verlag. [[Google Scholar](#)]

5. Diamantaras KI, Kung SY. 1996. *Principal component neural networks: theory and applications*. New York, NY: Wiley. [[Google Scholar](#)]
6. Flury B. 1988. *Common principal components and related models*. New York, NY: Wiley. [[Google Scholar](#)]
7. Horn R, Johnson C. 1985. *Matrix analysis*. Cambridge, UK: Cambridge University Press. [[Google Scholar](#)]
8. Lorraine Li, Python Machine Learning series, Next Tech Page.
9. Datacamp Community, Tutorials, Principal Component Analysis(PCA) in Python.

APPENDIX A

SNAP SHOTS



The screenshot displays a Jupyter Notebook titled "Principal Component Analysis". The interface includes a top bar with the Jupyter logo, the title, and a "Logout" button. Below this is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A status bar at the bottom of the menu indicates "Not Trusted" and "Python 3".

The notebook content is divided into two main sections. The first section, titled "NORMAL PCA", contains a code cell with the following Python code:

```
In [2]: import pandas as pd
import numpy as np

df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/'
'machine-learning-databases/wine/wine.data', header=None)
df_wine.head()
```

The output of this code cell is displayed below, showing the first five rows of the dataset. The output is a table with 13 columns and 5 rows of data.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735

jupyter Principal Component Analysis Last Checkpoint: Last Wednesday at 2:06 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Processing Dataset and Standardizing

```
In [3]: from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Split into training and testing sets
X, Y = df_wine.iloc[:, 1:].values, df_wine.iloc[:, 0].values
X_train, X_test, Y_train, Y_test = train_test_split(
    X, Y, test_size=0.3,
    stratify=Y, random_state=0
)

# Standardize the features
sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)
```

REDUCING DIMENSIONS USING PCA

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

```
In [11]: # transform the entire 124 x 13-dimensional training set into two principal components using PCA module

from sklearn.decomposition import PCA
pca_wine = PCA(n_components=2)

# fit and transform data
X_train_pca = pca_wine.fit_transform(X_train_std)
X_test_pca = pca_wine.transform(X_test_std)

print(X_train_pca.shape) # As we can see, we have reduced feature into 2 main features
print(X_test_pca.shape)

# create a DataFrame which contains the Principal Components
principal_wine_df = pd.DataFrame(data = X_train_pca , columns = ['PC-1', 'PC-2'])
principal_wine_df.tail()

(124, 2)
(54, 2)
```

Out[11]:

	PC-1	PC-2
119	0.435211	-2.403558
120	2.590451	1.638529
121	4.353084	0.665360
122	-1.843154	-1.506884
123	-0.408610	-1.297206

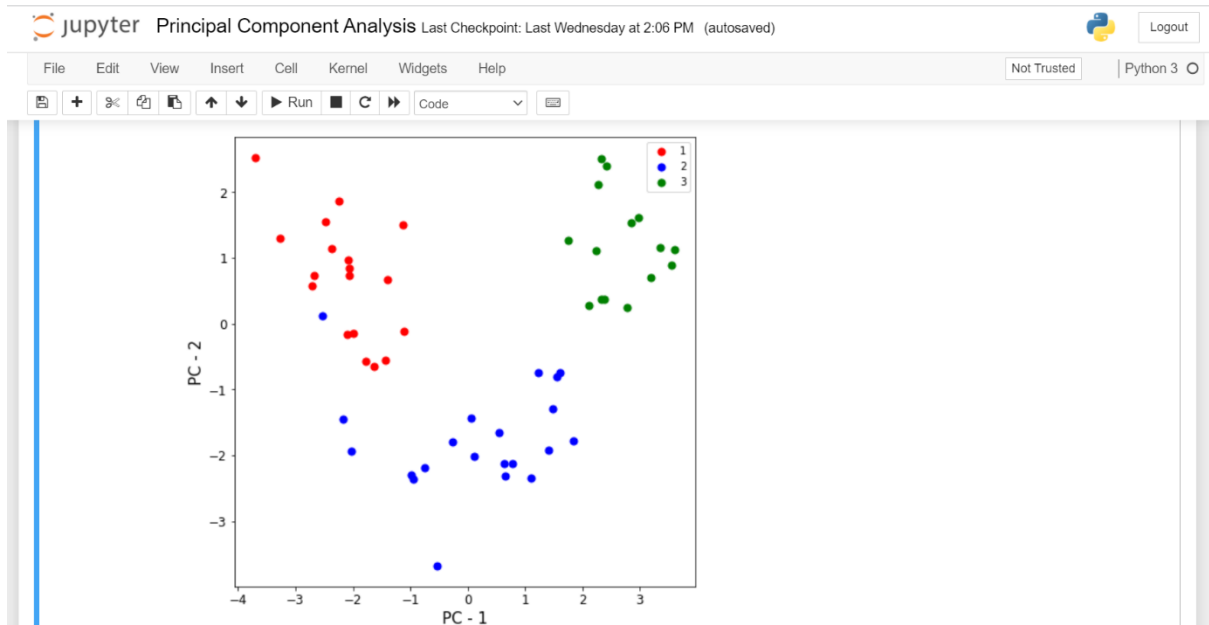
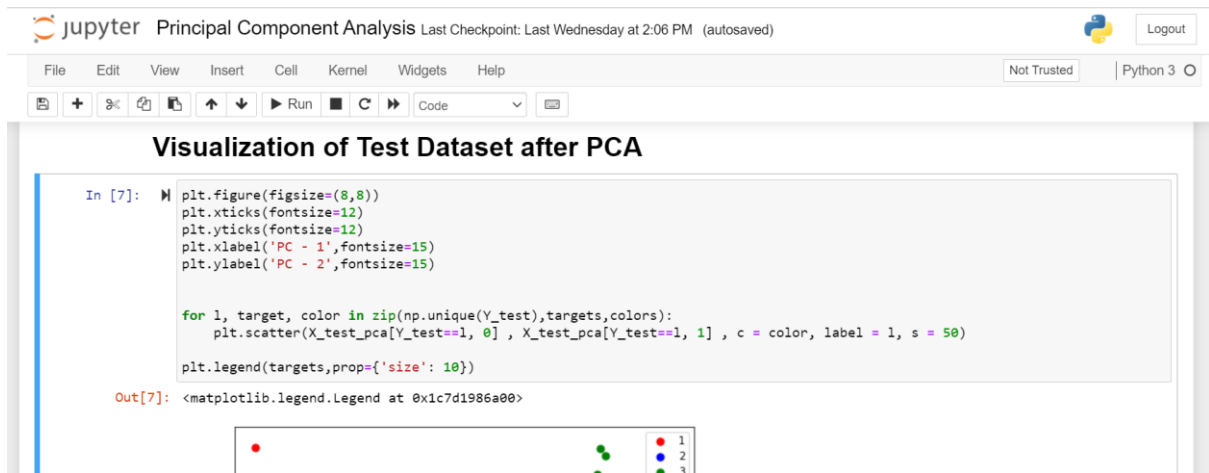
jupyter Principal Component Analysis Last Checkpoint: Last Wednesday at 2:06 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Variance explained ratios

```
In [5]: print('Explained variation per principal component : {}'.format(pca_wine.explained_variance_ratio_))

Explained variation per principal component : [0.36951469 0.18434927]
```

APPENDIX B

CODING

Importing Packages & Dataset

```
import pandas as pd
import numpy as np
df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/'
                      'machine-learning-databases/wine/wine.data', header=None)
df_wine.head()
```

Data Preprocessing

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
# Split into training and testing sets
X, Y = df_wine.iloc[:, 1:].values, df_wine.iloc[:, 0].values
X_train, X_test, Y_train, Y_test = train_test_split(
    X, Y, test_size=0.3,
    stratify=Y, random_state=0
)
# Standardize the features
sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)
```

Reducing Dimensions of Dataset

```

# transform the entire 124 x 13-dimensional training set into two principal
components using PCA module
from sklearn.decomposition import PCA
pca_wine = PCA(n_components=2)

# fit and transform data
X_train_pca = pca_wine.fit_transform(X_train_std)
X_test_pca = pca_wine.transform(X_test_std)
print(X_train_pca.shape) # As we can see, we have reduced feature into 2 main
features
print(X_test_pca.shape)

# create a DataFrame which contains the Principal Components
principal_wine_df = pd.DataFrame(data = X_train_pca , columns = ['PC-1', 'PC-
2'])
principal_wine_df.tail()

```

Variance Explained Ratios

```

print('Explained      variation      per      principal      component:
{}'.format(pca_wine.explained_variance_ratio_))

```

Visualization of Train Dataset after PCA

```

import matplotlib.pyplot as plt
%matplotlib inline
plt.figure()
plt.figure(figsize=(9,9))
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.xlabel('PC - 1',fontsize=20)
plt.ylabel('PC - 2',fontsize=20)
plt.title("Principal Component Analysis of Wine Dataset",fontsize=25)
targets = ['1','2','3']
colors = ['r','b','g']
for l, target, color in zip(np.unique(Y_train),targets,colors):
plt.scatter(X_train_pca[Y_train==l, 0] , X_train_pca[Y_train==l, 1] , c = color,
label = l, s = 50)

plt.legend(targets,prop = {'size': 15})

```

Visualization of Test Dataset after PCA

```

plt.figure(figsize=(8,8))
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.xlabel('PC - 1',fontsize=15)
plt.ylabel('PC - 2',fontsize=15)

for l, target, color in zip(np.unique(Y_test),targets,colors):
    plt.scatter(X_test_pca[Y_test==l, 0] , X_test_pca[Y_test==l, 1] , c = color,
label = l, s = 50)

plt.legend(targets,prop={'size': 10})

```