NAME: - Medhansh K PRN: - 122A8035 BATCH: - B2

EXPERIMENT 11

**AIM: -** Principal Component Analysis

# THEROY: -

Here's a breakdown of the steps involved in Principal Component Analysis (PCA):

# Data Preparation:

* 1. Gather your dataset, which should consist of numerical features.
  2. Organize the data into a matrix, where rows represent samples and columns represent features.

# Standardization (Scaling):

* 1. This is a crucial step. Standardize the features to have zero mean and unit variance.
  2. This ensures that features with different scales contribute equally to the PCA.
  3. Common methods include z-score standardization.

# Covariance Matrix Calculation:

* 1. Compute the covariance matrix of the standardized data.
  2. The covariance matrix shows how the features vary together.

# Eigenvalue and Eigenvector Decomposition:

* 1. Calculate the eigenvalues and eigenvectors of the covariance matrix.
  2. Eigenvectors represent the directions of maximum variance.
  3. Eigenvalues represent the magnitude of the variance along those directions.

# Eigenvector Sorting:

* 1. Sort the eigenvectors in descending order based on their corresponding eigenvalues.
  2. This arranges the principal components in order of importance (variance explained).

# Principal Component Selection:

* 1. Choose the top 'k' eigenvectors, where 'k' is the desired number of dimensions for the reduced dataset.
  2. The selection of k is often determined by the desired amount of explained variance.
  3. The explained variance ratio helps to make this decision.

# Data Projection:

* 1. Multiply the standardized data by the selected eigenvectors (the principal components).
  2. This transforms the original data into the new, lower-dimensional space defined by the principal components.

# Interpretation:

* 1. Analyze the transformed data.
  2. Visualize the data in the reduced dimensions.
  3. Interpret the principal components in terms of the original features.
  4. Examine the explained variance ratio to understand how much information is retained.

# Principal Component Analysis Algorithm Step by Step Implementation

import pandas as pd import numpy as np

from sklearn.datasets import load\_breast\_cancer

# instantiating

cancer = load\_breast\_cancer(as\_frame=True) # creating dataframe

df = cancer.frame

print('Original Dataframe shape :',df.shape)

X = df[cancer['feature\_names']] print('Inputs Dataframe shape :', X.shape)

****

# Mean

X\_mean = X.mean()

# Standard deviation X\_std = X.std()

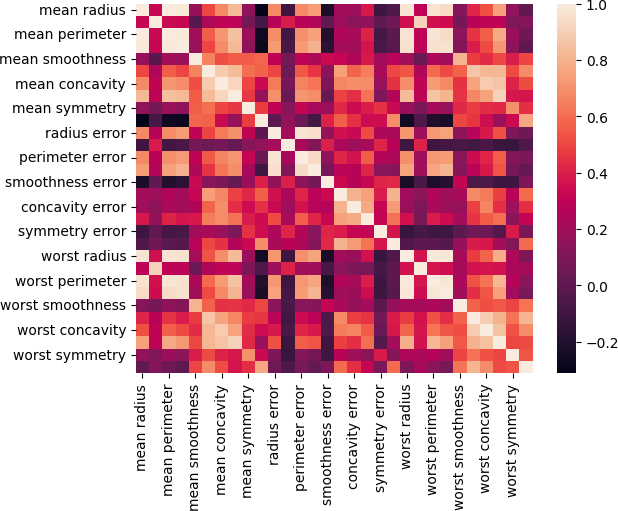
# Standardization

Z = (X - X\_mean) / X\_std

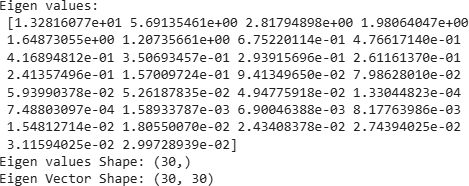
# covariance c = Z.cov()

# Plot the covariance matrix import matplotlib.pyplot as plt import seaborn as sns sns.heatmap(c)

plt.show()



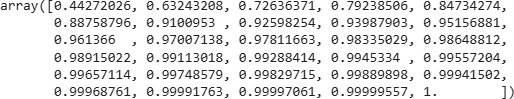
eigenvalues, eigenvectors = np.linalg.eig(c) print('Eigen values:\n', eigenvalues) print('Eigen values Shape:', eigenvalues.shape) print('Eigen Vector Shape:', eigenvectors.shape)

****

# Index the eigenvalues in descending order idx = eigenvalues.argsort()[::-1]

# Sort the eigenvalues in descending order eigenvalues = eigenvalues[idx]

# sort the corresponding eigenvectors accordingly eigenvectors = eigenvectors[:,idx]

****

explained\_var = np.cumsum(eigenvalues) / np.sum(eigenvalues) explained\_var

****

n\_components = np.argmax(explained\_var >= 0.50) + 1 n\_components

# PCA component or unit matrix

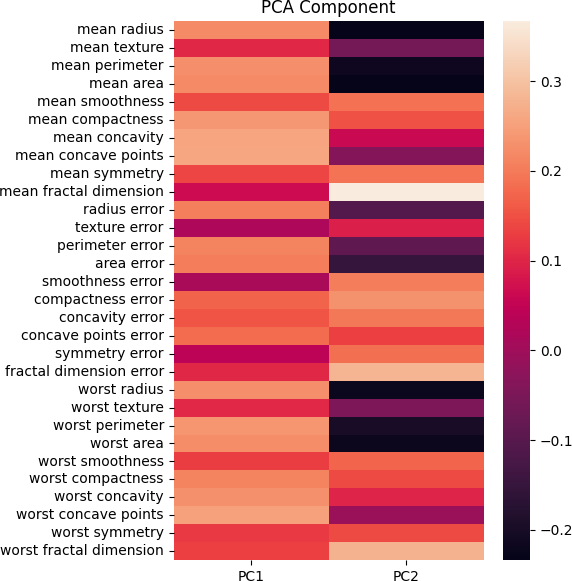
u = eigenvectors[:,:n\_components] pca\_component = pd.DataFrame(u,

index = cancer['feature\_names'], columns = ['PC1','PC2']

)

# plotting heatmap plt.figure(figsize =(5, 7)) sns.heatmap(pca\_component)

plt.title('PCA Component') plt.show()

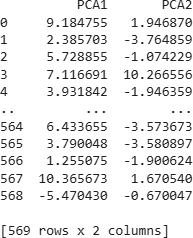
****

# Matrix multiplication or dot Product Z\_pca = Z @ pca\_component

# Rename the columns name

Z\_pca.rename({'PC1': 'PCA1', 'PC2': 'PCA2'}, axis=1, inplace=True) # Print the Pricipal Component values

print(Z\_pca)



# Importing PCA

from sklearn.decomposition import PCA

# Let's say, components = 2 pca = PCA(n\_components=2) pca.fit(Z)

x\_pca = pca.transform(Z)

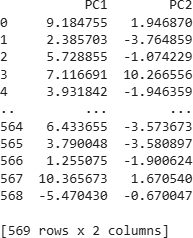
# Create the dataframe

df\_pca1 = pd.DataFrame(x\_pca,

columns=['PC{}'. format(i+1)

for i in range(n\_components)])

print(df\_pca1)



# giving a larger plot

plt.figure(figsize=(8, 6))

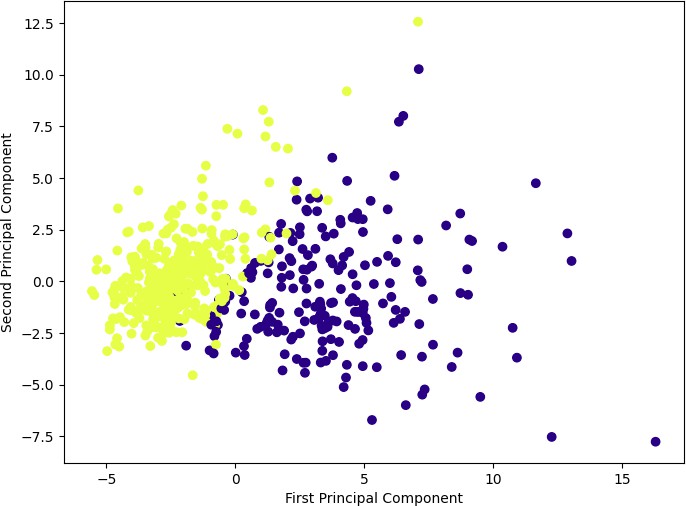
plt.scatter(x\_pca[:, 0], x\_pca[:, 1], c=cancer['target'],

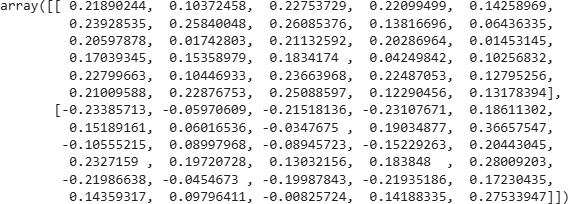
cmap='plasma')

# labeling x and y axes plt.xlabel('First Principal Component')

plt.ylabel('Second Principal Component')

plt.show()



****

# components pca.components\_

**CONCLUSION: -**

In summary, PCA helps in distilling complex data into its most informative elements, making it simpler and more efficient to analyze.

It identifies the directions (called principal components) where the data varies the most.

It projects the data onto these directions, reducing the number of dimensions while retaining as much information as possible.

The new set of uncorrelated variables (principal components) is easier to work with and can be used for tasks like regression, classification, or visualization.