Name: Jigar Siddhpura **SAPID:** 60004200155

DIV: C/C2 **Branch:** Computer Engineering

ML - Experiment 7 - PCA

1

60004210155
Jigar Siddhpura
C22

ML - Exp 7 - PCA

Aim: To implement PCA

Theory: PCA is a popular unsupervised learning algorithm for reducing the dimensionality of data. It increases interpretability, yet at the same time, it minimizes information loss. It helps to find most significant ~~dataset~~ features in the dataset & makes the data easy for plotting in 2D & 3D. It helps in finding a sequence of linear combinations of variables. Principal components are straight line that capture most of the variance of data. They have a direction & magnitude. Principal components are orthogonal projections of data onto lower dimensional space.

The term 'dimensionality' describes the quantity of features / variables used in research. It can be difficult to visualize & interpret the relationships between variables when dealing with high-dimensional data, such as datasets with numerous variables, while reducing the ~~data~~ variables in most crucial data. The original variables are converted into a new set of variables called principal components are used in the study. The dataset's reduced dimensionality depends on how many principal components are used in the study. The objective of PCA is to select fewer PC that acc for most important variation.

A statistical measure known as correlation expresses the direction & strength of linear connection btw 2 variables. The covariance matrix, a square matrix that displays the pairwise correlations btw all pair of variables in the dataset, is calculated in the setting of PCA using correlation. Covariances' matrix diagonal ~~correlation~~ elem. stand for each variable's variance, while the off diagonal ~~variance~~ elem. indicate the covariance btw diff pair of variables.

Sum:

a	b
4	6
8	2
13	3
7	15

$$S = \begin{bmatrix} var(x) & cov(x,y) \\ cov(x,y) & var(y) \end{bmatrix}$$

Formula: $cov(a,a) = \frac{1}{N-1} \sum (a_i - \bar{a})(a_i - \bar{a})$

$\bar{a} = 8 \qquad \bar{b} = 6.5$

$$cov(a,b) = \frac{1}{3}\left[(4-8)(6-6.5) + (8-8)(2-6.5) + (13-8)(3-6.5) + (7-8)(15-6.5)\right]$$

$$= -8$$

$$cov(a,a) = \frac{1}{3}\left[(4-8)^2 + (8-8)^2 + (13-8)^2 + (7-8)^2\right]$$

$$= 14$$

$$cov(b,b) = \frac{1}{3}\left[(6-6.5)^2 + (2-6.5)^2 + (3-6.5)^2 + (15-6.5)^2\right]$$

$$= 35$$

(2)

$S = \begin{bmatrix} 14 & -8 \\ -8 & 35 \end{bmatrix} \longrightarrow$ Covariance matrix

Finding eigen values,

$|S - \lambda I| = 0$

$\left| \begin{matrix} 14 & -8 \\ -8 & 35 \end{matrix} \right| - \lambda \left| \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \right| = 0$

$\left| \begin{matrix} 14-\lambda & -8 \\ -8 & 35-\lambda \end{matrix} \right| = 0$

$(14-\lambda)(35-\lambda) - 64 = 0$

$\lambda^2 - 49\lambda + 426 = 0$

$\lambda_1 = 37.7 \quad , \quad \lambda_2 = 11.3$

Finding eigen vectors,

$(S - \lambda I) U_1 = 0$

Let $U_1 = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$

Taking λ_1

$\begin{bmatrix} -23.7 & -8 \\ -8 & -2.7 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$

$-23.7u_1 - 8u_2 = 0$

$\frac{u_1}{8} = \frac{u_2}{-23.7} = A$, let $A = 1$

$u_1 = 8 , u_2 = -23.7$

$u_1 = \begin{bmatrix} 8 \\ -23.7 \end{bmatrix}$

Taking λ_2,

$\begin{bmatrix} 2.7 & -8 \\ -8 & 23.7 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$

$2.7u_1 - 8u_2 = 0$

$-8u_1 + 23.7u_2 = 0$

$\frac{u_1}{8} = \frac{u_2}{2.7} = A$, Let $A = 1 \Rightarrow u_2 = \begin{bmatrix} 8 \\ 2.7 \end{bmatrix}$

$nu_1 = \begin{bmatrix} 0.32 \\ -0.95 \end{bmatrix}$ ... $\left[\frac{8}{\sqrt{8^2 + 23.7^2}} = 0.32 \right.$ &

$\left. \frac{-23.7}{\sqrt{8^2 + 23.7^2}} = -0.95 \right]$

$\therefore nu_2 = \begin{bmatrix} 0.95 \\ 0.32 \end{bmatrix}$

$$\left(\frac{\lambda_1}{\lambda_1 + \lambda_2}, \frac{\lambda_2}{\lambda_1 + \lambda_2} \right) = \left(\frac{37.7}{49}, \frac{11.3}{49} \right) = (0.77, 0.23)$$

0.77 is max, so we project only on 1^{st} ~~eigen~~ eign vector

$nu_1^T = \begin{bmatrix} a - \bar{a} \\ b - \bar{b} \end{bmatrix}$ ... to get projection

$nu_1 = \begin{bmatrix} 0.32 \\ -0.95 \end{bmatrix}$, $nu_2 = \begin{bmatrix} 0.95 \\ 0.32 \end{bmatrix}$

$nu_1^T = \begin{bmatrix} 0.32 & -0.95 \end{bmatrix}$

$nu_2^T = \begin{bmatrix} 0.95 & 0.32 \end{bmatrix}$

$a = 4$, $b = 6$ $\rightarrow \begin{bmatrix} 0.32 & -0.95 \end{bmatrix} \begin{bmatrix} 4-8 \\ 6-6.5 \end{bmatrix} = -0.81 = P_{11}$

$a = 8$, $b = 2$ $\rightarrow \begin{bmatrix} 0.32 & -0.95 \end{bmatrix} \begin{bmatrix} 8-8 \\ 2-6.5 \end{bmatrix} = 4.27 = P_{12}$

$a = 13$, $b = 3$ $\rightarrow \begin{bmatrix} 0.32 & -0.95 \end{bmatrix} \begin{bmatrix} 13 & -8 \\ 3 & -6.5 \end{bmatrix} = 4.91 = P_{13}$

$a = 7$, $b = 15$ $\rightarrow \begin{bmatrix} 0.32 & -0.95 \end{bmatrix} \begin{bmatrix} 7 & -8 \\ 15 & -6.5 \end{bmatrix} = -8.37 = P_{14}$

```
import pandas as pd
import numpy as np
from numpy.linalg import eig
```

DATASET 1 - CODE

```
data = np.array([[4, 6], [8, 2], [13, 3], [7, 15]])

def PCA(df):
  centered_data = df - df.mean()

  cov_matrix = np.cov(centered_data, rowvar=False)

  eigenvalues, eigenvectors = np.linalg.eig(cov_matrix)

  sorted_indices = np.argsort(eigenvalues)[::-1]
  eigenvalues = eigenvalues[sorted_indices]
  eigenvectors = eigenvectors[:, sorted_indices]

  new_values = np.dot(centered_data, eigenvectors)[:,0]

  print("Centered Data:")
  print(centered_data)
  print("\nCovariance Matrix:")
  print(cov_matrix)
  print("\nEigenvalues:")
  print(eigenvalues)
  print("\nEigenvectors:")
  print(eigenvectors)
  print("\nNew Values:")
  print(new_values)

df2 = pd.DataFrame(data)
PCA(df2)
```

OUTPUT

```
Centered Data:
     0    1
0 -4.0 -0.5
1  0.0 -4.5
2  5.0 -3.5
3 -1.0  8.5

Covariance Matrix:
[[14. -8.]
 [-8. 35.]]

Eigenvalues:
[37.70037878 11.29962122]

Eigenvectors:
[[ 0.31981892 -0.94747869]
 [-0.94747869 -0.31981892]]

New Values:
[-0.80553633  4.26365409  4.91526999 -8.37338775]
```

DATASET 2 - CODE

df = pd.read_csv('/content/gdrive/MyDrive/ML/salary_data.csv')

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

```
import pandas as pd
from sklearn.decomposition import PCA

def apply_pca(data, n_components):
    pca = PCA(n_components=n_components)
    principalComponents = pca.fit_transform(data)
    principalDf = pd.DataFrame(data=principalComponents, columns=[f'New Values' for i in range(n_components)])

    # Calculate additional PCA components
    centered_data = data - data.mean()
    cov_matrix = pca.get_covariance()
    eigenvalues = pca.explained_variance_
    eigenvectors = pca.components_
```

```
    print("Centered Data:")
    print(centered_data)
    print("\nCovariance Matrix:")
    print(cov_matrix)
    print("\nEigenvalues:")
    print(eigenvalues)
    print("\nEigenvectors:")
    print(eigenvectors)
    print("\n")

    return principalDf

n_components = 1
result = apply_pca(df2, n_components)
print(result)
```

<u>OUTPUT</u>

```
Centered Data:
     0    1
0 -4.0 -0.5
1  0.0 -4.5
2  5.0 -3.5
3 -1.0  8.5

Covariance Matrix:
[[14. -8.]
 [-8. 35.]]

Eigenvalues:
[37.70037878]

Eigenvectors:
[[-0.31981892  0.94747869]]


   New Values
0    0.805536
1   -4.263654
2   -4.915270
3    8.373388
```

<u>Conclusion :</u> PCA, a powerful dimensionality reduction technique, simplifies large datasets by transforming variables into a smaller set while retaining essential information. It can be used for for summarizing complex datasets, uncovering relationships between variables, and simplifying data analysis processes effectively.