Name: Jigar Siddhpura **SAPID:** 60004200155

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

ML - Experiment 7 - PCA

```
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)
```

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]

Eigenvectors:
[[-0.31981892  0.94747869]]


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

Conclusion : 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.