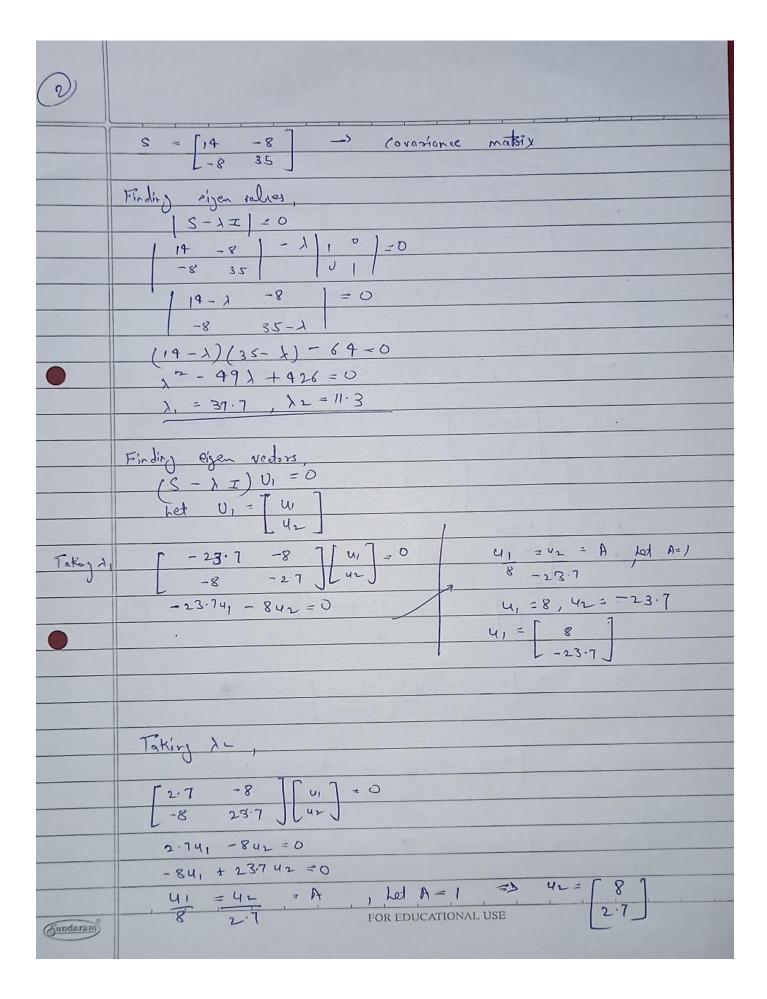
Name: Jigar Siddhpura SAPID: 60004200155

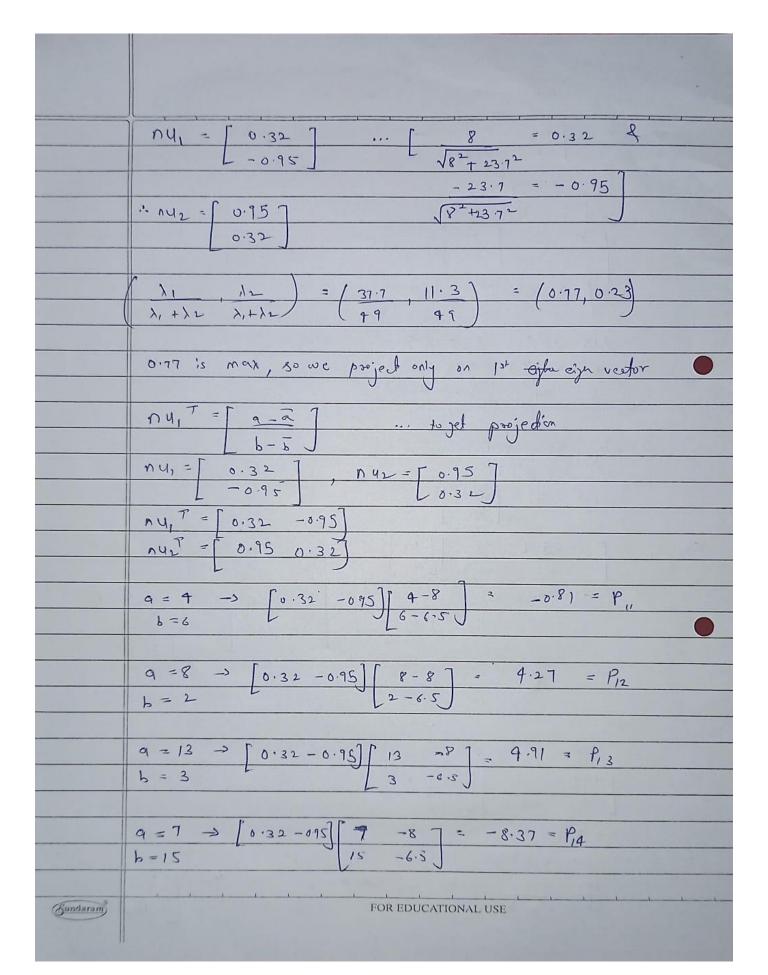
DIV: C/C2 Branch: Computer Engineering

ML - Experiment 7 - PCA

	60004210155
(1)	Jigor Siddhpura
	ML - Exp 7 - PCA C22
	A* , T : 1 - 1 200
	Ain: To implement PCA
	Theory: PlA is a popular unsupervised learning algorithm for
	reducing the dimensionality of data. It increases
	interprebability, yet at the same time, it miniscs information
	interprobability, yet at the same time, it miniscs information loss. It helps to find must significant dataset features in
	the dataset & makes the data casy for plotting in 20 & 30.
	It helps in finding a sequence of linear combinations of
	variables. Principal components are straight line that copture
	most of the agricus of data. They have a direction of
	most of the variance of data. They have a direction &
	magnitule. Principal components are orthogonal projections
	of data anto lower dimensional space.
	- 1 1 h A 1 h a market
	The term dimensionality describes the quantity of features (variables
	used in sevensch. It can be difficult to visualize &
	interpret the relationaries between variables when dealing
	with high - dimensional data such as datasets with
	numbrous variables, while reducing the date variables in most
	could date. The original variables are converted into a new
	got of variables called principal components are used in the
	Study. The dataset's neduced dimensionality depends on how many
	original components use used in the strucy. The objective
	of RA is to select fewer PC that acc. for most
	important vostation.
	THINGS STATE OF THE STATE OF TH
	FOR EDUCATIONAL USE
Sundaram	

	A statistical measure known as 100 relation expresses the direction
	& Strength of linear connection 6th 2 variables. The
	Covariance matrix, a square matrix that displays the
	pairoise correlations both all pair of voriables in the
	dotaset is calculated in the setting of PCA using
	correlation. Corasianues matrix diagonal resoclation elen.
	Stand for each variables vorionce, while the ff diagnal
	vosables.
	Academ .
	Sum: a b S = [vor(x) (ov(x,y))]
	Sum: a b $S = \begin{bmatrix} Ver(x) & (ev(x,y) \end{bmatrix}$ 4 6 $\begin{bmatrix} Cev(x,y) & Ver(y) \end{bmatrix}$
	8 2
	13 3 Formula: cov(a,a) = 1 \(\(\frac{1}{2} \) \(\frac{1}{2} - \arappa \)
	7 15 N-7
	a = 8 $b = 6.5$
	$rov(0,b) = \frac{1}{3} \left[(4-8)(1-6.5) + (8-8)(1-6.5) + (13-8)(3-6.5) \right]$ $+ (7-8)(15-6.5) \right]$
	7 (7-8)(13-6.3)
	= -8
	(or(a,a) - 1 [(+-8)+ (8-8) + (13-8) + (17-8) 2]
	= 14
	$ror(6,1) = \frac{1}{3} \left[(6-6.5)^2 + (2-6.5)^2 + (3-6.5)^2 + (15-6.5)^2 \right]$
	= 35
(Sundaram)	FOR EDUCATIONAL USE





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
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')



import pandas as pd from sklearn.decomposition import PCA

eigenvalues = pca.explained variance

eigenvectors = pca.components

```
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()
```

```
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
     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.