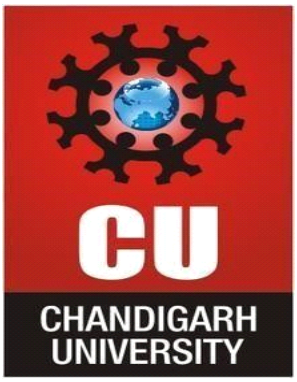
CHANDIGARH UNIVERSITY

UNIVERSITY INSTITUTE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



Submitted By: Rajiv Paul	**Submitted To:** Parveen Badoni
Subject Name	Machine Learning Lab
Subject Code	20CSP-317
Branch	BE-CSE
Semester	5th

LAB INDEX

Sr. No	Program	Date	Evaluation				Sign
			LW (12)	VV (8)	FW (10	Total (30	
1.1							
1.2							
1.3							
2.1							
2.2							
2.3							
2.4							
3.1							
3.2	To Implement Principal Component Analysis.	02/11/22					
3.3							

EXPERIMENT – 9

Student Name:Rajiv Paul **UID: 20BCS1812**
Branch:CSE **Section/Group: 20BCS_WM-702A**
Semester: 5th **Date of Performance: 01/11/2022**
Subject Name: Machine Learning Lab **Subject Code: 20CSP-317**

1. AIM OF THE EXPERIMENT:

To Implement Principal Component Analysis.

2. TASK TO BE DONE:

We will create a method named PCA to implement Principal Component Analysis on a dataset (i.e. iris).

3. PROGRAM CODE & OUTPUT:

i) Importing libraries

- import numpy as np
- import matplotlib.pyplot as plt
- import pandas as pd
- import seaborn as sb

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import pandas as pd
4 import seaborn as sns
```

ii) Creating a method for PCA.

```
def PCA(X , num_components):
    X_meaned = X - np.mean(X , axis = 0)
    cov_mat = np.cov(X_meaned , rowvar = False)
    eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)
    sorted_index = np.argsort(eigen_values)[::-1]
    sorted_eigenvalue = eigen_values[sorted_index]
    sorted_eigenvectors = eigen_vectors[:,sorted_index]
    eigenvector_subset =
    sorted_eigenvectors[:,0:num_components]
    X_reduced = np.dot(eigenvector_subset.transpose() ,
    X_meaned.transpose() ).transpose()
    return X_reduced
```

```python
def PCA(X , num_components):

    #Step-1
    X_meaned = X - np.mean(X , axis = 0)

    #Step-2
    cov_mat = np.cov(X_meaned , rowvar = False)

    #Step-3
    eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)

    #Step-4
    sorted_index = np.argsort(eigen_values)[::-1]
    sorted_eigenvalue = eigen_values[sorted_index]
    sorted_eigenvectors = eigen_vectors[:,sorted_index]

    #Step-5
    eigenvector_subset = sorted_eigenvectors[:,0:num_components]

    #Step-6
    X_reduced = np.dot(eigenvector_subset.transpose() , X_meaned.transpose() ).transpose()

    return X_reduced
```

iii) Importing Dataset (i.e. iris).

- url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
- data = pd.read_csv(url, names=['sepal length','sepal width','petal length','petal width','target'])
- pd.DataFrame(data)

```
1 url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
2 data = pd.read_csv(url, names=['sepal length','sepal width','petal length','petal width','target'])
3 pd.DataFrame(data)
```

	sepal length	sepal width	petal length	petal width	target
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

iv) Separating target column from the data.

- x = data.iloc[:,0:4]
- target = data.iloc[:,4]
- print(target)

v)

```
x = data.iloc[:,0:4]
target = data.iloc[:,4]
print(target)
```

```
0          Iris-setosa
1          Iris-setosa
2          Iris-setosa
3          Iris-setosa
4          Iris-setosa
              ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: target, Length: 150, dtype: object
```

```
mat_reduced = PCA(x , 2)
pd.DataFrame(mat_reduced)
```

	0	1
0	2.684207	0.326607
1	2.715391	-0.169557
2	2.889820	-0.137346
3	2.746437	-0.311124
4	2.728593	0.333925
...	...	...
145	-1.944017	0.187415
146	-1.525664	-0.375021
147	-1.764046	0.078519
148	-1.901629	0.115877
149	-1.389666	-0.282887

150 rows × 2 columns

vi) Creating a Pandas DataFrame of reduced Dataset.

- principal_df = pd.DataFrame(mat_reduced , columns = ['PC1','PC2'])
- print(principal_df)

```
principal_df = pd.DataFrame(mat_reduced , columns = ['PC1','PC2'])
print(principal_df)
```

```
          PC1       PC2
0    2.684207  0.326607
1    2.715391 -0.169557
2    2.889820 -0.137346
3    2.746437 -0.311124
4    2.728593  0.333925
..        ...       ...
145 -1.944017  0.187415
146 -1.525664 -0.375021
147 -1.764046  0.078519
148 -1.901629  0.115877
149 -1.389666 -0.282887

[150 rows x 2 columns]
```

vii) ...omplete

- principal_df = pd.concat([principal_df , pd.DataFrame(target)] , axis = 1)
- print(principal_df)

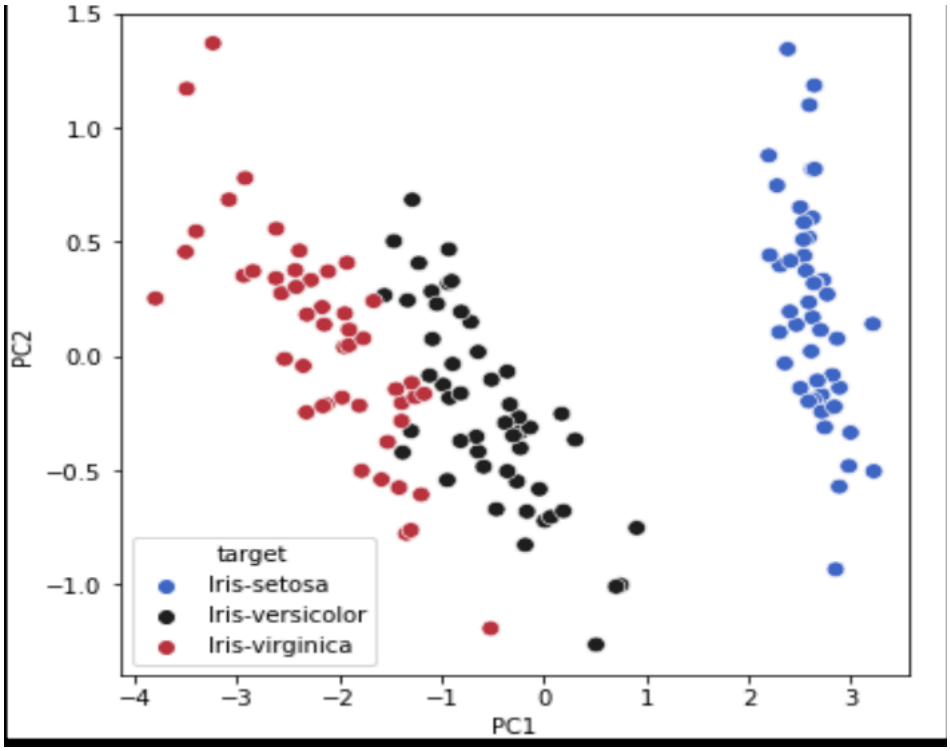
```
principal_df = pd.concat([principal_df , pd.DataFrame(target)] , axis = 1)
print(principal_df)
```

```
          PC1       PC2          target
0    2.684207  0.326607     Iris-setosa
1    2.715391 -0.169557     Iris-setosa
2    2.889820 -0.137346     Iris-setosa
3    2.746437 -0.311124     Iris-setosa
4    2.728593  0.333925     Iris-setosa
..        ...       ...             ...
145 -1.944017  0.187415  Iris-virginica
146 -1.525664 -0.375021  Iris-virginica
147 -1.764046  0.078519  Iris-virginica
148 -1.901629  0.115877  Iris-virginica
149 -1.389666 -0.282887  Iris-virginica

[150 rows x 3 columns]
```

viii) Visualizing predicted values of iris dataset.

- plt.figure(figsize = (6,6))
- plt.title("Kinshuk Chauhan \n 20BCS4917")
- sb.scatterplot(data = principal_df, x = 'PC1', y='PC2', hue='target', s=60, palette='icefire')


1.5
1.0
0.5
0.0
−0.5
−1.0
PC2
−4
−3
−2
−1
0
1
2
3
PC1
target
Iris-setosa
Iris-versicolor
Iris-virginica