EXPT NO: 10 **A python program to implement Dimensionality**

DATE: 04.11.2024 **Reduction -PCA.**

AIM:

To write a python program to implement Dimensionality Reduction - PCA .

PROCEDURE:

ImplementingDimensionality reduction -pca using the Iris dataset involve the following steps:

Step 1: Import Necessary Libraries

First, import the libraries that are essential for data manipulation, visualization, and model building.

```
# Importing necessary libraries

from sklearn import datasets

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA
```

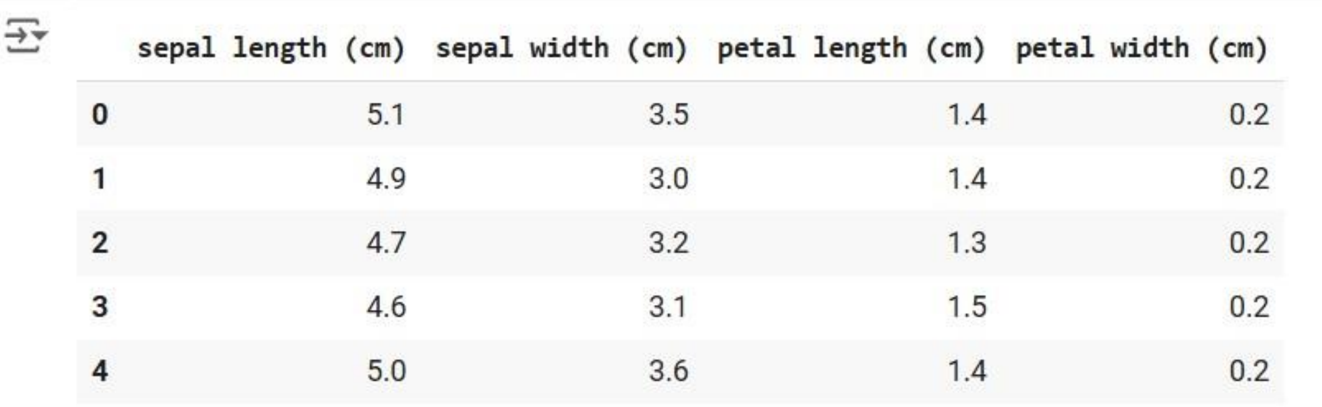
```
import seaborn as sns

import matplotlib.pyplot as plt
```

Step 2: Load the Iris Dataset

The Iris dataset can be loaded and display the first few rows of the dataset

```
# Load the Iris dataset

iris = datasets.load_iris()

df = pd.DataFrame(iris['data'], columns=iris['feature_names'])



# Display the first few rows of the dataset

df.head()
```

OUTPUT :



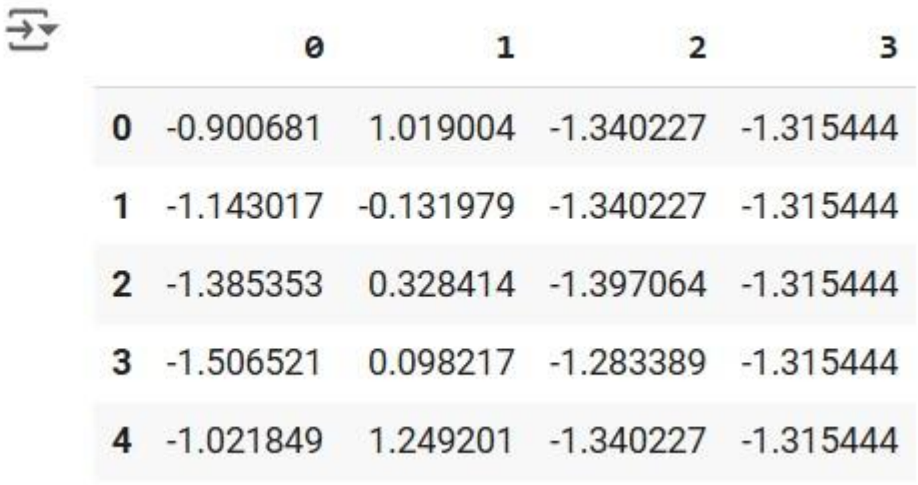
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Step 3 : Standardize the data

```
# Standardize the features using StandardScaler

scalar = StandardScaler()

scaled_data = pd.DataFrame(scalar.fit_transform(df))  # Scaling the data



# Display the scaled data (optional)
```

```
scaled_data.head()
```

OUTPUT :

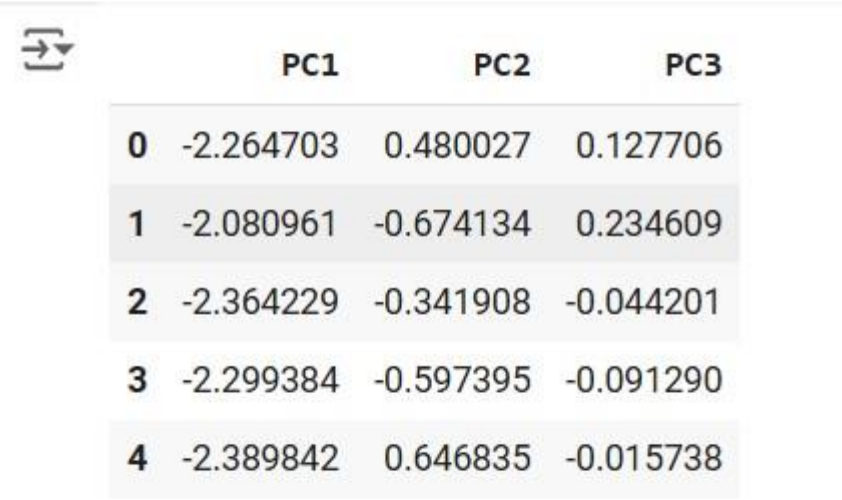


	0	1	2	3
0	-0.900681	1.019004	-1.340227	-1.315444
1	-1.143017	-0.131979	-1.340227	-1.315444
2	-1.385353	0.328414	-1.397064	-1.315444
3	-1.506521	0.098217	-1.283389	-1.315444
4	-1.021849	1.249201	-1.340227	-1.315444

Step 4 : Apply PCA

```
# Apply PCA to reduce the data to 3 components

pca = PCA(n_components=3)

pca.fit(scaled_data) # Fit PCA on scaled data
data_pca = pca.transform(scaled_data) # Transform the data to principal
components



# Convert PCA data to a DataFrame for easier inspection

data_pca = pd.DataFrame(data_pca, columns=['PC1', 'PC2', 'PC3'])

data_pca.head()
```

OUTPUT :

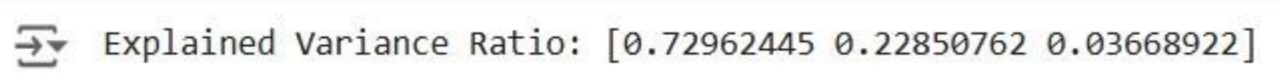


	PC1	PC2	PC3
0	-2.264703	0.480027	0.127706
1	-2.080961	-0.674134	0.234609
2	-2.364229	-0.341908	-0.044201
3	-2.299384	-0.597395	-0.091290
4	-2.389842	0.646835	-0.015738

Step 5 : Explained Variance Ratio

```
# Calculate the explained variance ratio for each principal component

explained_variance = pca.explained_variance_ratio_

print(f"Explained Variance Ratio: {explained_variance}")



# This output shows how much variance each principal component explains.
```

OUTPUT :

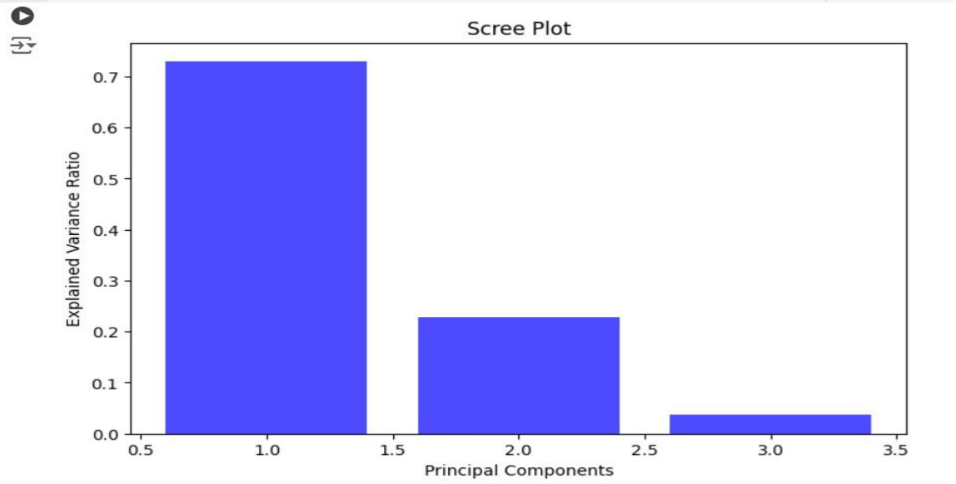


```
Explained Variance Ratio: [0.72962445 0.22850762 0.03668922]
```

Step 6 :Visualize the reduced data.

```
# Plotting the explained variance ratio as a scree plot
plt.figure(figsize=(8, 5))


plt.bar(range(1, len(explained_variance) + 1), explained_variance,
alpha=0.7, color='blue')


plt.ylabel('Explained Variance Ratio')


plt.xlabel('Principal Components')


plt.title('Scree Plot')
plt.show()
```

OUTPUT :



RESULT :

Thus the Dimensionality Reduction has been implemented using PCA in python program Successfully.