**Class: BE** **Subject: Machine Learning** **Sem-VII**

## Experiment No: 10

**Course Outcome: CO6** **Blooms Level:L3**

**Aim: To implement PCA**

**Abstract:** This program demonstrates the application of Principal Component Analysis (PCA) for dimensionality reduction and data visualization using the Iris dataset. The implementation is carried out in two ways: (i) using the scikit-learn library for a simplified and practical approach, and (ii) manually with NumPy to illustrate the underlying mathematical concepts such as covariance computation, eigen decomposition, and variance explanation. The results show how PCA transforms high-dimensional data into lower dimensions while retaining maximum variance, making it easier to visualize patterns and clusters in the dataset.

**Sample Input and Output:**

**Theory:**

1. **Standardize the data**
   - Compute $X_{std} = \frac{X-\mu}{\sigma}$.
   - *Why:* PCA is scale-sensitive; standardization gives each feature equal weight. *(Code:* `StandardScaler` *or manual centering/scaling.)*
2. **Compute covariance matrix**
   - $\Sigma = \frac{1}{n-1} X_{std}^{\top} X_{std}$.
   - *Why:* measures how features vary together.
3. **Eigen decomposition (or SVD)**
   - Solve $\Sigma v = \lambda v$ to get eigenvalues $\lambda$ and eigenvectors $v$.
   - *Why:* eigenvectors = principal directions; eigenvalues = variance along those directions.
4. **Sort eigenpairs**
   - Order eigenvectors by descending eigenvalue.
   - *Why:* ensures the first components capture the most variance.

5. **Choose number of components $k$**
   - Use explained-variance ratio: $\text{ratio}_i = \frac{\lambda_i}{\sum_j \lambda_j}$.
   - Pick $k$ so cumulative ratio ≥ threshold (e.g., 90%).
6. **Project data to lower dimension**
   - $Y = X_{std} V_k$ where $V_k$ contains the top-k eigenvectors.
   - *Why:* obtains compact representation that preserves maximum variance.
7. **(Optional) Reconstruct / measure loss**
   - $X_{\text{approx}} = Y V_k^\top + \mu$.
   - *Why:* check how much information is lost by dimensionality reduction.
8. **Interpretation**
   - Use scatter plots of principal components, inspect loadings (elements of eigenvectors) to see how original features contribute to each PC, and report explained variance.

**Program:**

[1] ✓ 1s
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
```

[2] ✓ 0s
```
# Step 1: Load dataset
iris = load_iris()
X = iris.data
y = iris.target
target_names = iris.target_names
```

[3] ✓ 0s
```
# Step 2: Standardize data
X_std = StandardScaler().fit_transform(X)
```

[4] ✓ 0s
```
# Step 3: Compute covariance matrix
cov_matrix = np.cov(X_std.T)
```

[5] ✓ 0s
```
# Step 4: Compute eigenvalues and eigenvectors
eigenvalues, eigenvectors = np.linalg.eig(cov_matrix)
```

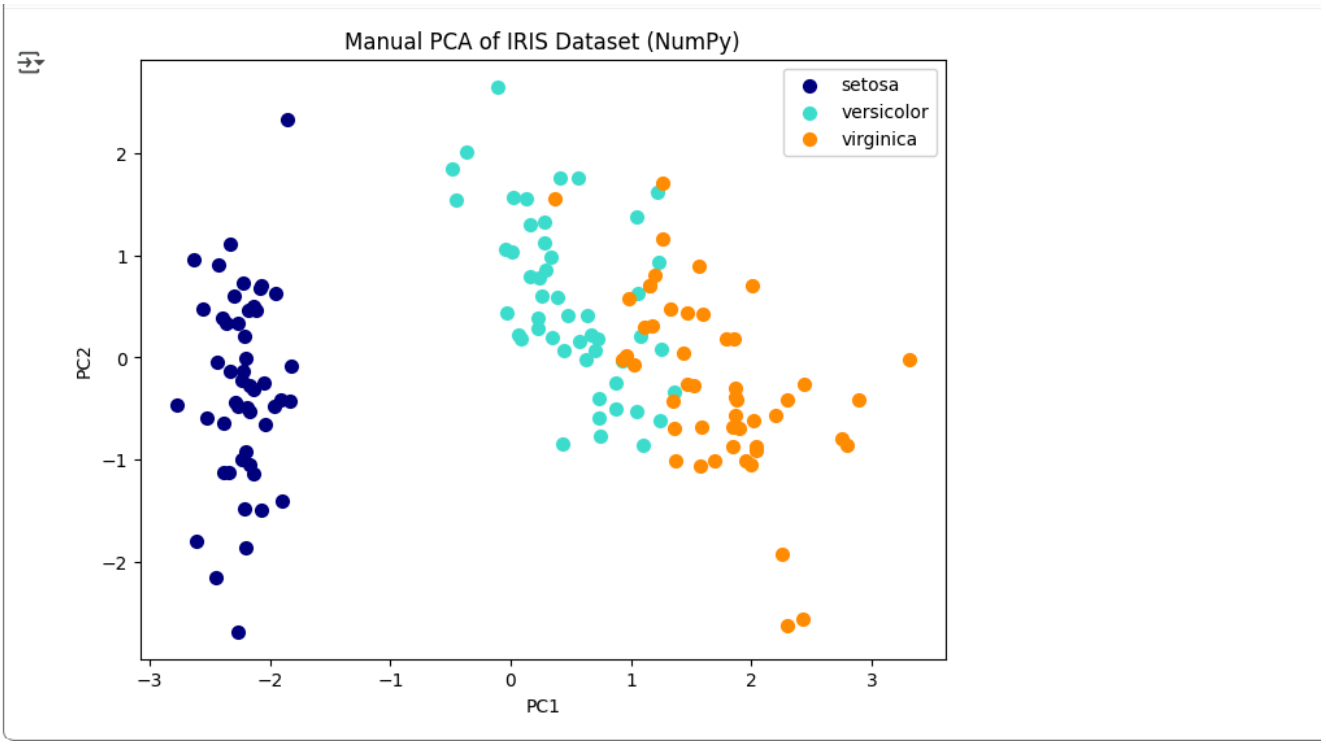
[6] ✓ 0s
```
# Step 5: Sort eigenvalues and eigenvectors
sorted_idx = np.argsort(eigenvalues)[::-1]
eigenvalues = eigenvalues[sorted_idx]
eigenvectors = eigenvectors[:, sorted_idx]
```

[8] ✓ 0s
```
# Step 7: Transform data
X_reduced = np.dot(X_std, eigenvectors_subset)
```

[9] ✓ 1s
```
# Step 8: Plot results
plt.figure(figsize=(8, 6))
colors = ['navy', 'turquoise', 'darkorange']
for color, i, target_name in zip(colors, [0, 1, 2], target_names):
    plt.scatter(X_reduced[y == i, 0], X_reduced[y == i, 1],
                color=color, lw=2, label=target_name)

plt.xlabel("PC1")
plt.ylabel("PC2")
plt.title("Manual PCA of IRIS Dataset (NumPy)")
plt.legend()
plt.show()
```

**Class: BE** **Subject: Machine Learning** **Sem-VII**




```
[10]
✓ 0s
# Step 9: Print explained variance
explained_variance = eigenvalues / np.sum(eigenvalues)
print("Explained variance ratio (manual):", explained_variance[:k])

Explained variance ratio (manual): [0.72962445 0.22850762]
```

**Output:**

Explained variance ratio (manual): [0.72962445 0.22850762]

**Conclusion:** The implementation of Principal Component Analysis (PCA) on the Iris dataset successfully demonstrated how high-dimensional data can be transformed into a lower-dimensional space while retaining most of the variance. Both the scikit-learn and manual NumPy approaches produced consistent results, validating the mathematical foundation of PCA through eigen decomposition. The visualizations clearly showed class separability in the reduced 2D space, highlighting PCA's effectiveness for dimensionality reduction, feature extraction, and data visualization in machine learning tasks.

**Class: BE** **Subject: Machine Learning** **Sem-VII**

**Exercise 1:**

You are provided with the Wine Dataset (UCI Machine Learning Repository / available in sklearn.datasets.load_wine).
The dataset contains 178 wine samples from three different cultivars (classes), described by 13 chemical features such as alcohol, flavanoids, color intensity, magnesium, etc.

Tasks:

1. Load the Wine dataset and display its shape and feature names.
2. Standardize the dataset so all features have zero mean and unit variance.
3. Apply PCA:
   - (a) Compute the explained variance ratio for all components.
   - (b) Determine how many components are required to explain at least 95% of the variance.
4. Reduce the data to 2 principal components and plot a scatter plot with different colors for the three wine classes.
5. Interpret the results:
   - Which components explain the most variance?
   - Does the 2D PCA plot show separation among the wine classes?

**Students shall draw flowchart of exercise question in the writeup and submit.**

**Exercise 2:**

You are provided with the **Breast Cancer Wisconsin Dataset** (available in sklearn.datasets.load_breast_cancer).
The dataset contains **569 samples** classified into **malignant** and **benign** tumors, described by **30 numerical features** (e.g., mean radius, texture, smoothness, concavity).

**Tasks:**

1. Load the Breast Cancer dataset and check its dimensions (number of samples × features).
2. Standardize the data before applying PCA.
3. Apply **PCA** to:
   - (a) Find the variance explained by each component.
   - (b) Plot the **cumulative explained variance** curve and determine the number of components needed to explain at least **90% of the variance**.
4. Reduce the dataset to **2 principal components** and create a scatter plot, coloring the points by their class (malignant vs. benign).
5. Analyze and answer:

- How well do the first two components separate the two classes?
- What is the minimum number of components you need to retain most of the information in the dataset?

**Students shall draw flowchart of exercise question in the writeup and submit.**