Experiment 04

Aim:

Implement and compare **PCA**, **SVD**, and **LDA** for dimensionality reduction and visualization.

Theory:

1) Principal Component Analysis (PCA)

- **What it does:** Finds new axes (principal components) along which the **variance** of the data is maximized (unsupervised).
- **Key steps (intuition):**
  1. (Usually) standardize features so all are comparable in scale.
  2. Compute the **covariance matrix** to capture feature relationships.
  3. Compute **eigenvectors** (directions) and **eigenvalues** (variance captured).
  4. Sort by eigenvalues and project onto the top *k* components.
- **Why use it:** Visualization, noise reduction, compression; improves downstream modeling by removing redundancy.

2) Singular Value Decomposition (SVD)

- **What it does:** Factorizes any matrix *A* into $U\Sigma V^{T}$: *U, V,* are orthogonal; *Σ* holds **singular values** (strength of latent directions).
- **Why it matters:** Efficient low-rank approximation, denoising, compression; PCA can be derived from SVD of the (centered) data matrix.

3) Linear Discriminant Analysis (LDA)

- **What it does: Supervised** projection that maximizes **class separability**, not variance.
- **Key idea:** Build **between-class** and **within-class** scatter matrices; solve a generalized eigenvalue problem to get directions that separate classes best, then project data.

PCA vs SVD vs LDA — Quick Contrast

- **PCA & SVD:** Unsupervised; capture variance and structure; great for compression & visualization.

- **LDA:** Supervised; maximizes class separation; best for classification-oriented dimensionality reduction.

Software & Dataset

- **Language/Libs:** Python, NumPy, scikit-learn, Matplotlib.
- **Dataset:** Iris (classic 3-class flower dataset) from scikit-learn.

Step-by-Step Procedure

A) Setup

1. Install (if needed): pip install numpy scikit-learn matplotlib
2. Import and load data:
   - from sklearn.datasets import load_iris
   - data = load_iris(); X = data.data; y = data.target

B) Preprocessing

3. **Standardize** features for PCA/SVD:
   - from sklearn.preprocessing import StandardScaler
   - X_scaled = StandardScaler().fit_transform(X)
     Note: Your lab sheet later applies LDA on **original** features (not standardized). Follow that for consistency.

C) PCA (2 components)

4. Fit & transform:
   - from sklearn.decomposition import PCA
   - X_pca = PCA(n_components=2).fit_transform(X_scaled)
5. Visualize: scatter plot colored by class; label axes **PC1/PC2**; add grid & colorbar.
6. Record: pca.explained_variance_ratio_ to report variance captured by PC1 and PC2.

D) SVD (2 components)

7. Fit & transform on **scaled** data:
   - from sklearn.decomposition import TruncatedSVD

  - X_svd = TruncatedSVD(n_components=2).fit_transform(X_scaled)

8. Visualize with labels **SVD1/SVD2**; grid & colorbar.
9. Record: svd.singular_values_ to note component strengths.

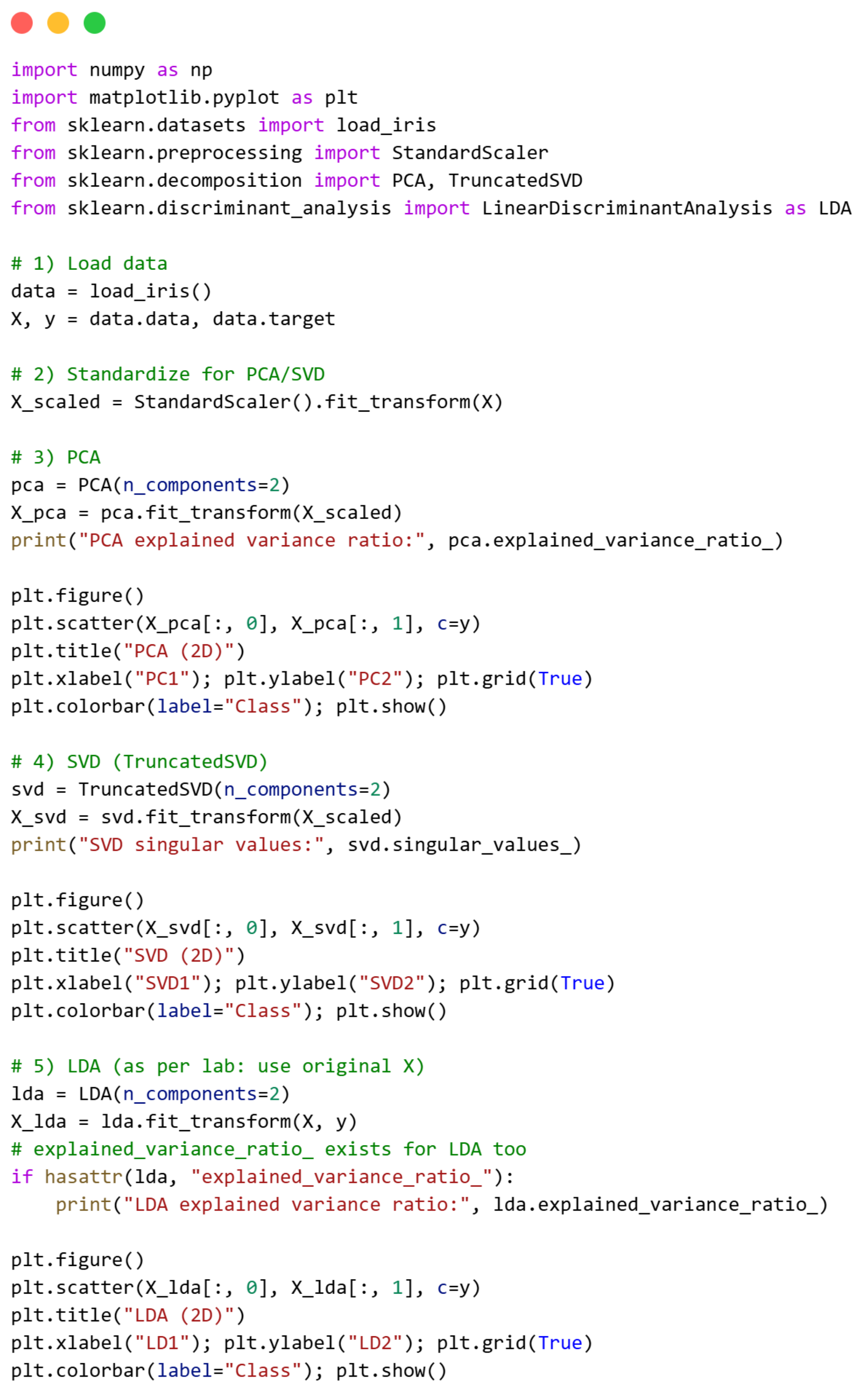
E) LDA (up to C−1 components)

10. Fit & transform on **original** X with labels y:
    - from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
    - X_lda = LDA(n_components=2).fit_transform(X, y)
11. Visualize with labels **LD1/LD2**; grid & colorbar.
12. Record: lda.explained_variance_ratio_ (in sklearn this reflects class separation captured).

F) Observations (What to write)

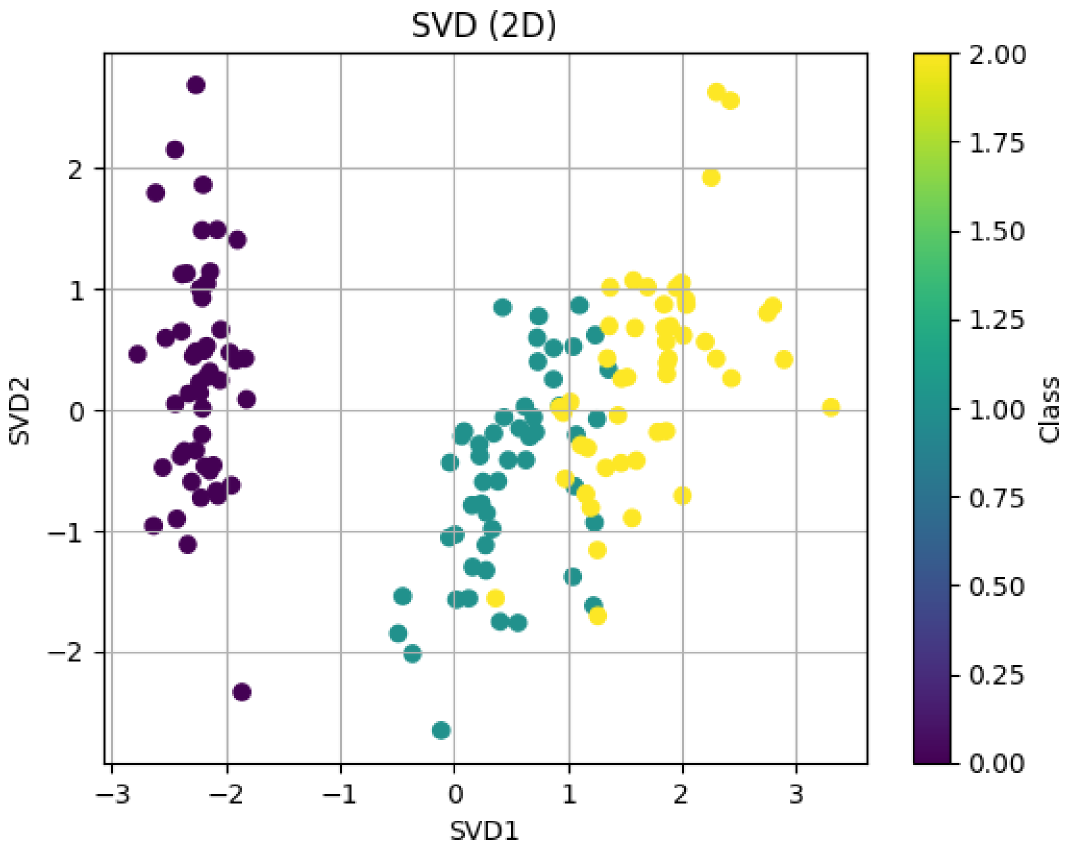
- **PCA plot:** Are classes roughly separated along PC1/PC2? Mention cluster overlap (if any).
- **SVD plot:** Similar to PCA for visualization; note how it reconstructs top directions via singular values.
- **LDA plot:** Expect clearer class separation (since it uses labels).
- **Numbers to report:** PCA variance ratios, SVD singular values, LDA variance ratios.
- **One-paragraph comparison:** Tie your observations to the goals (variance vs separability). Use your sheet's conclusion as reference.

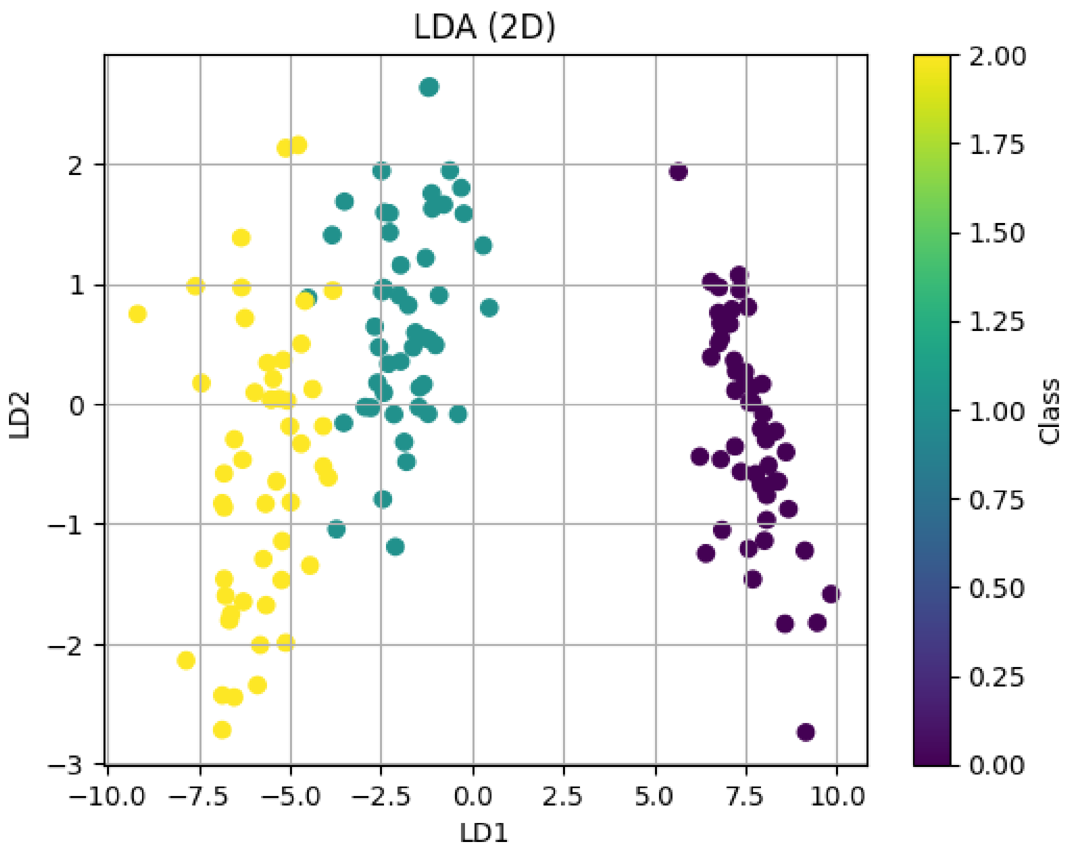
Code:



```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, TruncatedSVD
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

# 1) Load data
data = load_iris()
X, y = data.data, data.target

# 2) Standardize for PCA/SVD
X_scaled = StandardScaler().fit_transform(X)

# 3) PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
print("PCA explained variance ratio:", pca.explained_variance_ratio_)

plt.figure()
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y)
plt.title("PCA (2D)")
plt.xlabel("PC1"); plt.ylabel("PC2"); plt.grid(True)
plt.colorbar(label="Class"); plt.show()

# 4) SVD (TruncatedSVD)
svd = TruncatedSVD(n_components=2)
X_svd = svd.fit_transform(X_scaled)
print("SVD singular values:", svd.singular_values_)

plt.figure()
plt.scatter(X_svd[:, 0], X_svd[:, 1], c=y)
plt.title("SVD (2D)")
plt.xlabel("SVD1"); plt.ylabel("SVD2"); plt.grid(True)
plt.colorbar(label="Class"); plt.show()

# 5) LDA (as per lab: use original X)
lda = LDA(n_components=2)
X_lda = lda.fit_transform(X, y)
# explained_variance_ratio_ exists for LDA too
if hasattr(lda, "explained_variance_ratio_"):
    print("LDA explained variance ratio:", lda.explained_variance_ratio_)

plt.figure()
plt.scatter(X_lda[:, 0], X_lda[:, 1], c=y)
plt.title("LDA (2D)")
plt.xlabel("LD1"); plt.ylabel("LD2"); plt.grid(True)
plt.colorbar(label="Class"); plt.show()
```

Output:


SVD (2D)
SVD1
SVD2
Class
−3
−2
−1
0
1
2
3
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


LDA (2D)
LD1
LD2
Class
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
−3
−2
−1
0
1
2
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

Conclusion:

In this experiment, we applied and compared PCA, SVD, and LDA for dimensionality reduction on the Iris dataset. PCA and SVD, being unsupervised techniques, effectively reduced the dataset dimensions while preserving maximum variance, making them suitable for data compression and exploratory analysis; however, some class overlap was observed as they do not utilize class labels. In contrast, LDA, a supervised method, projected the data to maximize class separability, resulting in clearer distinctions between classes in the reduced space. Overall, the results highlight that PCA/SVD are ideal when the objective is to retain variance, whereas LDA is better suited for classification tasks where class discrimination is critical.