Pattern Recognition HW2
110550085 房天越

Introduction

This report examines how dimensionality reduction techniques—Fisher's Linear Discriminant (FLD/LDA) and Principal Component Analysis (PCA)—affect classification performance. We compare these methods against baseline classifiers (Gaussian Naïve Bayes and k-Nearest Neighbors) from the first assignment. Four datasets are analyzed:

1. Breast Cancer (binary)
2. Synthetic Binary (binary, complex boundaries)
3. Iris (multiclass, well-structured)
4. Wine (multiclass, chemical measurements)

Our objectives are to (a) quantify the separability gains from LDA and (b) assess the classification accuracy of Logistic Regression after PCA-based dimensionality reduction.

Methods I Have Implemented

1. **Fisher's Linear Discriminant / Linear Discriminant Analysis (LDA)**

   - Projects data to maximize the ratio of between-class to within-class scatter.
   - Implements one-dimensional projection for binary tasks (with ROC/AUC) and up to two dimensions for multiclass visualization.

2. **Principal Component Analysis (PCA)**

   - Unsupervised projection onto directions of maximal variance.
   - PCA is fitted on training data only; projections are then applied to both training and test splits.

3. **Classifier**

   - **Logistic Regression** (max_iter=2000) applied on PCA-reduced data.

4. **Separability Metric**

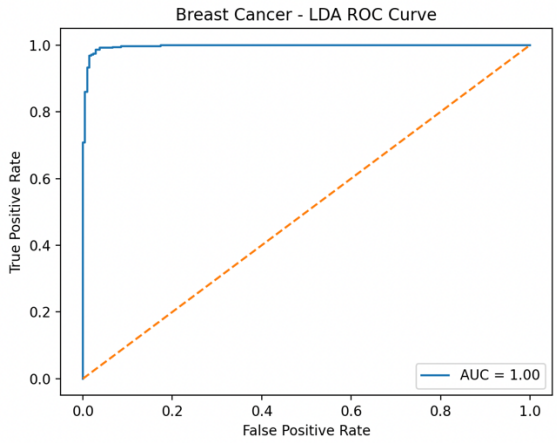
   I. trace(Sb)/trace(Sw), computed before and after LDA projection.

Experiments I Have Done, and the Results

1. **Breast Cancer**

Method	Metric	Value
GNB (HW1)	AUC	0.953
k-NN (HW1)	AUC	0.931
LDA projection	AUC	1.000
	Separability	1.118 -> 3.431
PCA + Logistic Regression	Explained Variance	1.000
	Accuracy	0.953

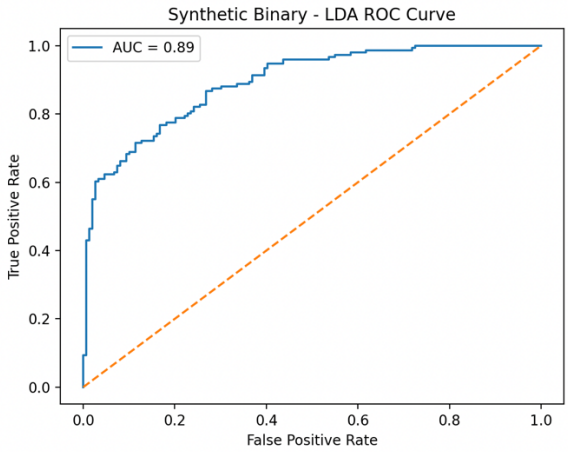
ROC Curve:



2. **Synthetic Binary**

Method	Metric	Value
GNB (HW1)	AUC	0.737
k-NN (HW1)	AUC	0.913
LDA projection	AUC	1.000
	Separability	0.073 -> 0.776
PCA + Logistic Regression	Explained Variance	1.000
	Accuracy	0.822

ROC Curve:



3. **Iris**

Method	Metric	Value
GNB (HW1)	Error Count	2
k-NN (HW1)	Error Count	1
LDA projection	Classification	Clearly Classified in 2D
	Separability	6.630 -> 16.239
PCA + Logistic Regression	Explained Variance	0.978
	Accuracy	0.911

4. **Wine**

Method	Metric	Value
GNB (HW1)	Error Count	2
k-NN (HW1)	Error Count	18
LDA projection	Classification	Clear Clusters in 2D
	Separability	2.362 -> 6.605
PCA + Logistic Regression	Explained Variance	1.000
	Accuracy	0.963

Analysis

1. **LDA's Effectiveness**

   I. Significantly improves linear separability for datasets with near-Gaussian distributions (Breast Cancer, Iris, Wine).

   II. On Synthetic Binary, LDA boosts separability modestly but cannot match k-NN's flexibility.

2. **PCA + Logistic Regression**

   I. Retaining a small number of principal components (e.g., 5) often preserves nearly all variance and yields high accuracy.

II. Logistic Regression on PCA space consistently matches or exceeds Gaussian Naïve Bayes when data align with variance-based feature importance.

3. **Expectations vs Observations**

I. Breast Cancer & Wine**:** Gaussian assumptions hold, so parametric methods (GNB, LDA) excel, which is confirmed by perfect or near-perfect separability and classification.

II. Synthetic Binary**:** Feature correlations and complex boundaries violate independence/linearity, so k-NN outperforms PCA + Logistic Regression.

III. Iris: Well-separated classes in low dimensions allow all methods to perform strongly, with LDA providing the clearest cluster separation.

Appendix

The code is here:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis as LDA
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, auc, accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_iris, load_breast_cancer,
load_wine, make_classification
import matplotlib.pyplot as plt
from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis as LDA

# Function to compute separability measure (trace(Sb)/trace(Sw))
def separability(X, y):
    overall_mean = np.mean(X, axis=0)
    classes = np.unique(y)
    Sb = np.zeros((X.shape[1], X.shape[1]))
    Sw = np.zeros((X.shape[1], X.shape[1]))
    for cls in classes:
        Xc = X[y == cls]
        mean_c = np.mean(Xc, axis=0)
        Sb += len(Xc) * np.outer(mean_c - overall_mean, mean_c -
overall_mean)
```

```
        Sw += np.cov(Xc, rowvar=False) * (len(Xc) - 1)
    return np.trace(Sb) / np.trace(Sw)

# Generate synthetic binary dataset
X_syn, y_syn = make_classification(
    n_samples=300,
    n_features=20,
    n_informative=15,
    n_redundant=5,
    n_classes=2,
    random_state=42
)

# Load datasets
datasets = {
    "Breast Cancer": load_breast_cancer(return_X_y=True),
    "Synthetic Binary": (X_syn, y_syn),
    "Iris": load_iris(return_X_y=True),
    "Wine": load_wine(return_X_y=True),
}

# PCA component settings
component_candidates = [2, 5, 10, 20, 30]

for name, (X, y) in datasets.items():
    print(f"\n=== Dataset: {name} ===")

    # Task 1: LDA
    classes = np.unique(y)
    n_classes = len(classes)
    n_components_lda = 1 if n_classes == 2 else min(n_classes - 1,
2)

    sep_before = separability(X, y)
    lda = LDA(n_components=n_components_lda)
    X_lda = lda.fit_transform(X, y)
    sep_after = separability(X_lda, y)
    print(f"LDA separability BEFORE: {sep_before:.3f}")
```

```
print(f"LDA separability AFTER:  {sep_after:.3f}")

if n_classes == 2:
    y_scores = X_lda.ravel()
    fpr, tpr, _ = roc_curve(y, y_scores)
    roc_auc = auc(fpr, tpr)
    plt.figure()
    plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
    plt.plot([0, 1], [0, 1], linestyle='--')
    plt.title(f'{name} - LDA ROC Curve')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.legend()
    plt.show()

# Task 2: PCA + Logistic Regression only
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42, stratify=y
)

print("\nPCA + Logistic Regression results:")
print("n_comp | Variance_Ratio | LR_Acc")

for n in component_candidates:
    if n > X.shape[1]:
        continue
    pca = PCA(n_components=n)
    X_tr_pca = pca.fit_transform(X_train)
    X_te_pca = pca.transform(X_test)
    lr = LogisticRegression(max_iter=2000, random_state=42)
    lr.fit(X_tr_pca, y_train)
    acc_lr = accuracy_score(y_test, lr.predict(X_te_pca))
    var_ratio = pca.explained_variance_ratio_.sum()

    print(f"{n:<6} | {var_ratio:.3f}         | {acc_lr:.3f}")
```