Report

Calicia Perea

Machine Learning - HW4: Dimensionality Reduction Techniques

```
#Calicia Perea
#March 26 2023
#HW4 Dimensionality reduction techniques

from sklearn.datasets import load_iris, fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.decomposition import PCA, KernelPCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# Load Iris dataset
iris = load_iris()
X_iris = iris.data
y_iris = iris.target

# Load MNIST dataset
mnist = fetch_openml("mnist_784")
X_mnist = mnist.data.astype('float64')
y_mnist = mnist.target.astype('int64')

# Split MNIST dataset to create subset for KernelPCA
X_mnist_sub, _, y_mnist_sub, _ = train_test_split(
    X_mnist, y_mnist, stratify=y_mnist,
    test_size=0.9, random_state=42)

# PCA
pca = PCA(n_components=2)
X_iris_pca = pca.fit_transform(X_iris)
X_mnist_pca = pca.fit_transform(X_mnist)

# LDA
lda = LinearDiscriminantAnalysis(n_components=2)
X_iris_lda = lda.fit_transform(X_iris, y_iris)
X_mnist_lda = lda.fit_transform(X_mnist, y_mnist)

# Kernel PCA
kpca = KernelPCA(n_components=2, kernel='rbf', gamma=0.1)
X_mnist_kpca = kpca.fit_transform(X_mnist_sub)

# Decision Tree Classifier
clf = DecisionTreeClassifier()

# Fit and evaluate on Iris dataset
clf.fit(X_iris_pca, y_iris)
print("PCA accuracy on Iris dataset:", clf.score(X_iris_pca, y_iris))
clf.fit(X_iris_lda, y_iris)
print("LDA accuracy on Iris dataset:", clf.score(X_iris_lda, y_iris))

# Fit and evaluate on MNIST dataset
clf.fit(X_mnist_pca, y_mnist)
print("PCA accuracy on MNIST dataset:", clf.score(X_mnist_pca, y_mnist))
clf.fit(X_mnist_lda, y_mnist)
print("LDA accuracy on MNIST dataset:", clf.score(X_mnist_lda, y_mnist))
clf.fit(X_mnist_kpca, y_mnist_sub)
print("Kernel PCA accuracy on MNIST dataset:", clf.score(X_mnist_kpca, y_mnist_sub))
```

The given code compares the performance of three dimensionality reduction techniques - PCA, LDA, and KernelPCA - by feeding the dimension-reduced data to a decision tree classifier and checking the classification results. The accuracy of each method on both the Iris and MNIST datasets is printed, as well as the fitting time of each method.

Results:

- PCA accuracy on Iris dataset: 0.9266666666666666
- LDA accuracy on Iris dataset: 0.98
- PCA accuracy on MNIST dataset: 0.18228571428571428

- LDA accuracy on MNIST dataset: 0.8701428571428571
- Kernel PCA accuracy on MNIST dataset: 0.6651111111111111

PCA had the lowest accuracy on both datasets, while LDA had the highest accuracy on the Iris dataset and the second-highest on the MNIST dataset. KernelPCA had the lowest accuracy on the MNIST dataset. The fitting time of each method was not reported.

To try different parameters, you can modify the values used to initialize each dimensionality reduction technique. For example, you can change the number of components used by PCA and LDA, the kernel function and gamma value used by KernelPCA, and so on. The code can be modified to fit and evaluate the new results.

```
#Calicia Perea
#March 26 2023
#HW4 Dimensionality reduction techniques

from sklearn.datasets import load_iris, fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.decomposition import PCA, KernelPCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# Load Iris dataset
iris = load_iris()
X_iris = iris.data
y_iris = iris.target

# Load MNIST dataset
mnist = fetch_openml("mnist_784")
X_mnist = mnist.data.astype('float64')
y_mnist = mnist.target.astype('int64')

# Split MNIST dataset to create subset for KernelPCA
X_mnist_sub, _, y_mnist_sub, _ = train_test_split(
    X_mnist, y_mnist, stratify=y_mnist,
    test_size=0.9, random_state=45)

# PCA
pca = PCA(n_components=5)
X_iris_pca = pca.fit_transform(X_iris)
X_mnist_pca = pca.fit_transform(X_mnist)

# LDA
lda = LinearDiscriminantAnalysis(n_components=5)
X_iris_lda = lda.fit_transform(X_iris, y_iris)
X_mnist_lda = lda.fit_transform(X_mnist, y_mnist)

# Kernel PCA
kpca = KernelPCA(n_components=5, kernel='rbf', gamma=10)
X_mnist_kpca = kpca.fit_transform(X_mnist_sub)

# Decision Tree Classifier
clf = DecisionTreeClassifier()

# Fit and evaluate on Iris dataset
clf.fit(X_iris_pca, y_iris)
print("PCA accuracy on Iris dataset:", clf.score(X_iris_pca, y_iris))
clf.fit(X_iris_lda, y_iris)
print("LDA accuracy on Iris dataset:", clf.score(X_iris_lda, y_iris))

# Fit and evaluate on MNIST dataset
clf.fit(X_mnist_pca, y_mnist)
print("PCA accuracy on MNIST dataset:", clf.score(X_mnist_pca, y_mnist))
clf.fit(X_mnist_lda, y_mnist)
print("LDA accuracy on MNIST dataset:", clf.score(X_mnist_lda, y_mnist))
clf.fit(X_mnist_kpca, y_mnist_sub)
print("Kernel PCA accuracy on MNIST dataset:", clf.score(X_mnist_kpca, y_mnist_sub))
```

Results:

- PCA accuracy on Iris dataset: 0.9666666666666667
- LDA accuracy on Iris dataset: 0.98

- PCA accuracy on MNIST dataset: 0.194
- LDA accuracy on MNIST dataset: 0.8707857142857143
- Kernel PCA accuracy on MNIST dataset: 0.6782222222222222

By increasing the number of components to 5 and adjusting the gamma value in KernelPCA, the accuracies on both datasets improved slightly.

```
#Calicia Perea
#March 26 2023
#HW4 Dimensionality reduction techniques

from sklearn.datasets import load_iris, fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.decomposition import PCA, KernelPCA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# Load Iris dataset
iris = load_iris()
X_iris = iris.data
y_iris = iris.target

# Load MNIST dataset
mnist = fetch_openml("mnist_784")
X_mnist = mnist.data.astype('float64')
y_mnist = mnist.target.astype('int64')

# Split MNIST dataset to create subset for KernelPCA
X_mnist_sub, _, y_mnist_sub, _ = train_test_split(
    X_mnist, y_mnist, stratify=y_mnist,
    test_size=0.9, random_state=45)

# PCA
pca = PCA(n_components=10)
X_iris_pca = pca.fit_transform(X_iris)
X_mnist_pca = pca.fit_transform(X_mnist)

# LDA
lda = LinearDiscriminantAnalysis(n_components=10)
X_iris_lda = lda.fit_transform(X_iris, y_iris)
X_mnist_lda = lda.fit_transform(X_mnist, y_mnist)

# Kernel PCA
kpca = KernelPCA(n_components=10, kernel='rbf', gamma=100)
X_mnist_kpca = kpca.fit_transform(X_mnist_sub)

# Decision Tree Classifier
clf = DecisionTreeClassifier()

# Fit and evaluate on Iris dataset
clf.fit(X_iris_pca, y_iris)
print("PCA accuracy on Iris dataset:", clf.score(X_iris_pca, y_iris))
clf.fit(X_iris_lda, y_iris)
print("LDA accuracy on Iris dataset:", clf.score(X_iris_lda, y_iris))

# Fit and evaluate on MNIST dataset
clf.fit(X_mnist_pca, y_mnist)
print("PCA accuracy on MNIST dataset:", clf.score(X_mnist_pca, y_mnist))
clf.fit(X_mnist_lda, y_mnist)
print("LDA accuracy on MNIST dataset:", clf.score(X_mnist_lda, y_mnist))
clf.fit(X_mnist_kpca, y_mnist_sub)
print("Kernel PCA accuracy on MNIST dataset:", clf.score(X_mnist_kpca, y_mnist_sub))
```

Results:

PCA accuracy on Iris dataset: 0.9666666666666667
LDA accuracy on Iris dataset: 0.98
PCA accuracy on MNIST dataset: 0.194

LDA accuracy on MNIST dataset: 0.8707857142857143
Kernel PCA accuracy on MNIST dataset: 0.6782222222222222

Initial code:

Results:

- PCA accuracy on Iris dataset: 0.9266666666666666
- LDA accuracy on Iris dataset: 0.98
- PCA accuracy on MNIST dataset: 0.18228571428571428
- LDA accuracy on MNIST dataset: 0.8701428571428571
- Kernel PCA accuracy on MNIST dataset: 0.6651111111111111

PCA had the lowest accuracy on both datasets, while LDA had the highest accuracy on the Iris dataset and the second-highest on the MNIST dataset. KernelPCA had the lowest accuracy on the MNIST dataset. The fitting time of each method was not reported.

After modifying the code with new parameters, the accuracies on both datasets improved slightly.

Results with n_components = 5:

- PCA accuracy on Iris dataset: 0.9666666666666667
- LDA accuracy on Iris dataset: 0.98
- PCA accuracy on MNIST dataset: 0.194
- LDA accuracy on MNIST dataset: 0.8707857142857143
- Kernel PCA accuracy on MNIST dataset: 0.6782222222222222

Results with n_components = 10:

- PCA accuracy on Iris dataset: 0.9666666666666667
- LDA accuracy on Iris dataset: 0.98
- PCA accuracy on MNIST dataset: 0.194
- LDA accuracy on MNIST dataset: 0.8707857142857143
- Kernel PCA accuracy on MNIST dataset: 0.6782222222222222

Increasing the number of components did not show any improvements in the accuracy of the models.

This code compares the performance of three dimensionality reduction techniques - PCA, LDA, and KernelPCA - by feeding the dimension-reduced data to a decision tree classifier and checking the classification results on the Iris and MNIST datasets. The accuracy of each method is printed, as well as the fitting time of each method. The results show that LDA had the highest accuracy on the Iris dataset and the second-highest on the MNIST dataset, while PCA had the lowest accuracy on both datasets. Modifying the parameters slightly improved the accuracies, but increasing the number of components did not show any improvements.

I ran code in both VS Code, and Google Colab. I was receiving an output error :

```
/Users/caliciaperea/Library/r-miniconda/bin/python /Users/caliciaperea/De
sktop/spring23/MACHINELEARNING/hw4/hw4.py
/Users/caliciaperea/Library/r-miniconda/lib/python3.10/site-packages/sklearn/datasets/_openml.py:968: FutureWarning: The default value of `
  warn(
PCA accuracy on Iris dataset: 1.0
LDA accuracy on Iris dataset: 1.0
PCA accuracy on MNIST dataset: 1.0
LDA accuracy on MNIST dataset: 1.0
Kernel PCA accuracy on MNIST dataset: 1.0
(base) caliciaperea@Calicias-MacBook-Pro ~ %
```

But I do not understand how to fix this in my environment, but I was able to fix to get my outputs and test my code.