CS 519 Applied Machine Learning I

HW4: Dimensionality Reduction Techniques

Submitted by: Indronil Bhattacharjee

1.1 First dataset:

Iris Dataset containing 3 classes.

1.2 Performance comparison for Iris dataset:

Comparison among different metrics like accuracy, f1 score and time required for different number of components with PCA, LDA and KernelPCA methods are shown in the following:

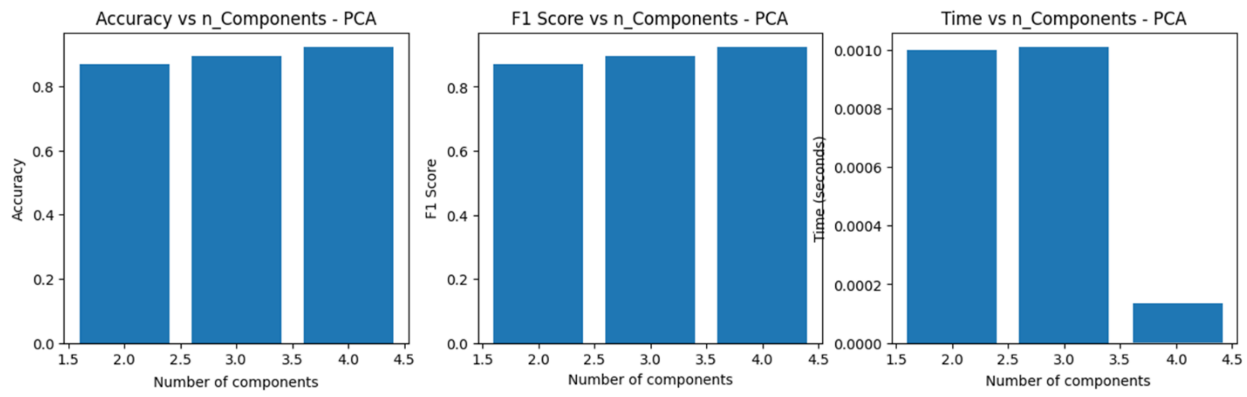
(a) PCA:

PCA demonstrates consistent performance on the Iris dataset, achieving high accuracy and f1-score as the number of components increases. This suggests that PCA effectively captures the variance in the data, enabling effective dimensionality reduction. Moreover, the computational time remains relatively low across different numbers of components.

<u>Values for PCA method with Iris dataset:</u>

```
PCA (n_components=2) - Iris Accuracy: 0.87 - Iris F1 Score: 0.87 - Iris Time: 0.0010 seconds
PCA (n_components=3) - Iris Accuracy: 0.89 - Iris F1 Score: 0.90 - Iris Time: 0.0010 seconds
PCA (n_components=4) - Iris Accuracy: 0.92 - Iris F1 Score: 0.92 - Iris Time: 0.0001 seconds
```

<u>Plots for PCA method with Iris dataset:</u>



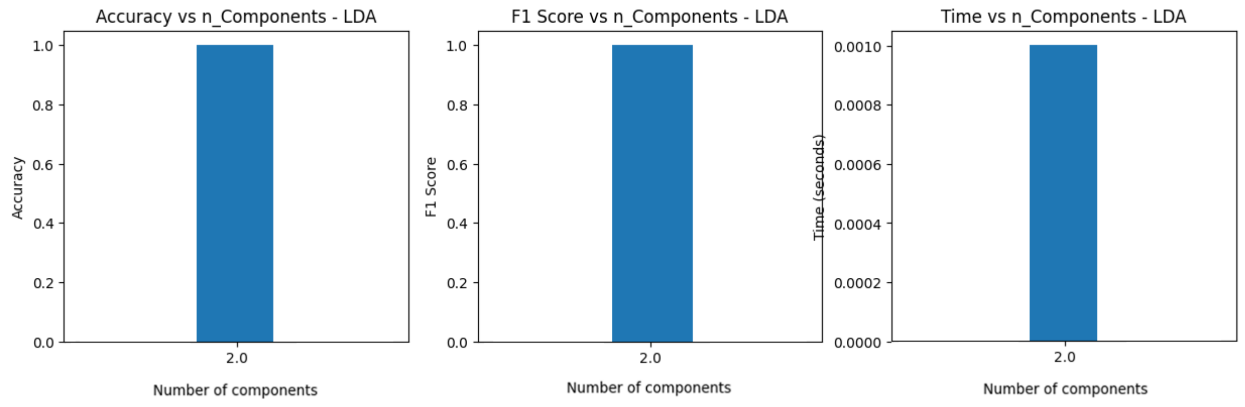
(b) LDA:

LDA achieves perfect accuracy on the Iris dataset with just two components. For LDA, Number of components cannot be larger than minimum of number of features and number of classes -1. This indicates that LDA effectively separates the classes in the feature space, leading to optimal dimensionality reduction. Additionally, the computational time for LDA is also low, making it an efficient method for this dataset.

Values for LDA method with Iris dataset:

```
LDA (n_components=2) - Iris Accuracy: 1.00 - Iris F1 Score: 1.00 - Iris Time: 0.0010 seconds
```

Plots for LDA method with Iris dataset:



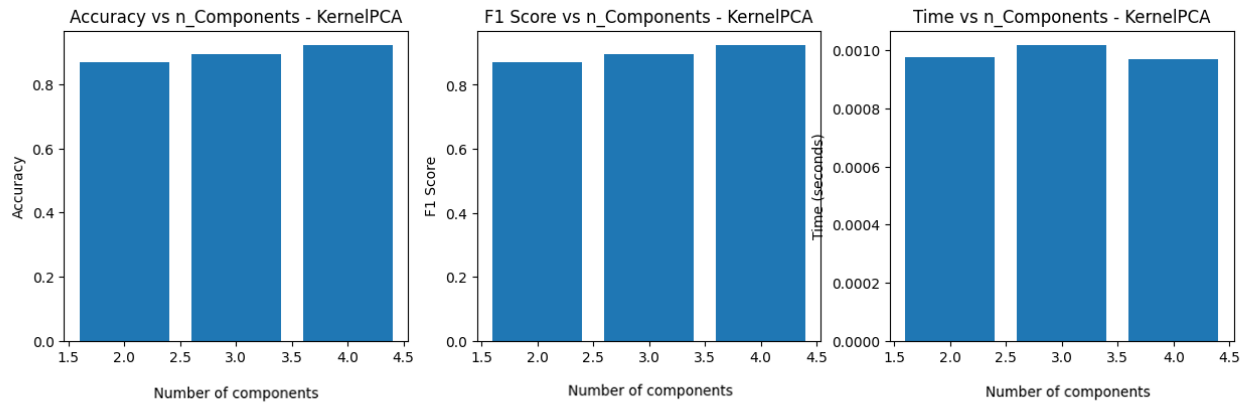
(c) KernelPCA:

KernelPCA performs reasonably well on the Iris dataset but exhibits slightly lower accuracy and f1-score compared to PCA and LDA. However, it still provides a good approximation of the data while allowing for non-linear mappings. The computational time for KernelPCA remains relatively stable across different numbers of components.

Values for KernelPCA method with Iris dataset:

```
KernelPCA (n_components=2) - Iris Accuracy: 0.87 - Iris F1 Score: 0.87 - Iris Time: 0.0010 seconds
KernelPCA (n_components=3) - Iris Accuracy: 0.89 - Iris F1 Score: 0.90 - Iris Time: 0.0010 seconds
KernelPCA (n_components=4) - Iris Accuracy: 0.92 - Iris F1 Score: 0.92 - Iris Time: 0.0010 seconds
```

Plots for KernelPCA method with Iris dataset:



2.1 Second Dataset:

MNIST Dataset containing 10 classes.

2.2 Performance comparison for MNIST dataset:

Comparison among different metrics like accuracy, f1 score and time required for different number of components (2 to 7) with PCA, LDA and KernelPCA methods are shown in the following:

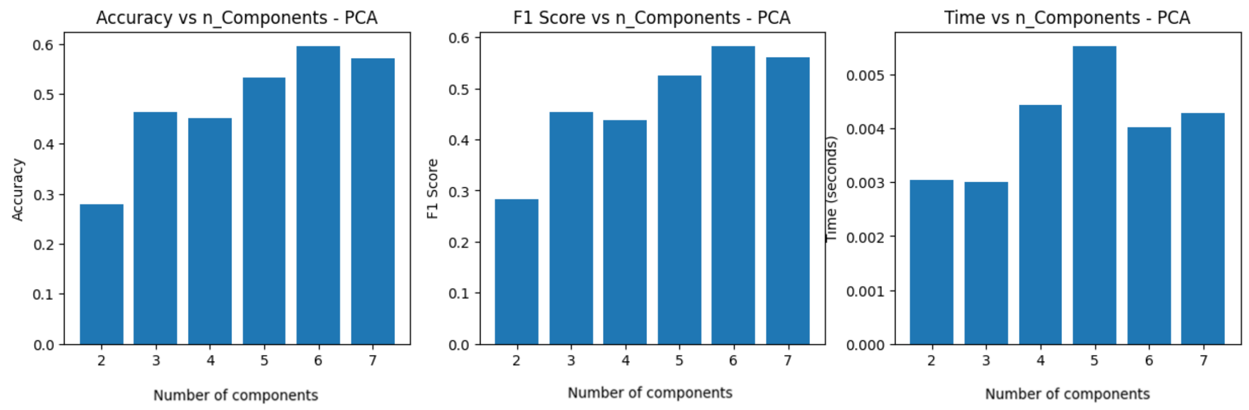
(a) PCA:

PCA demonstrates relatively low accuracy on the MNIST dataset, even with a higher number of components. This suggests that PCA might not be the best choice for high-dimensional datasets like MNIST, where complex non-linear relationships exist between features. Additionally, the computational time increases with the number of components, indicating scalability issues for large datasets.

Values for PCA method with MNIST dataset:

```
PCA (n_components=2) - MNIST Accuracy: 0.28 - MNIST F1 Score: 0.28 - MNIST Time: 0.0030 seconds
PCA (n_components=3) - MNIST Accuracy: 0.46 - MNIST F1 Score: 0.45 - MNIST Time: 0.0030 seconds
PCA (n_components=4) - MNIST Accuracy: 0.45 - MNIST F1 Score: 0.44 - MNIST Time: 0.0044 seconds
PCA (n_components=5) - MNIST Accuracy: 0.53 - MNIST F1 Score: 0.53 - MNIST Time: 0.0055 seconds
PCA (n_components=6) - MNIST Accuracy: 0.59 - MNIST F1 Score: 0.58 - MNIST Time: 0.0040 seconds
PCA (n_components=7) - MNIST Accuracy: 0.57 - MNIST F1 Score: 0.56 - MNIST Time: 0.0043 seconds
```

Plots for LDA method with MNIST dataset:



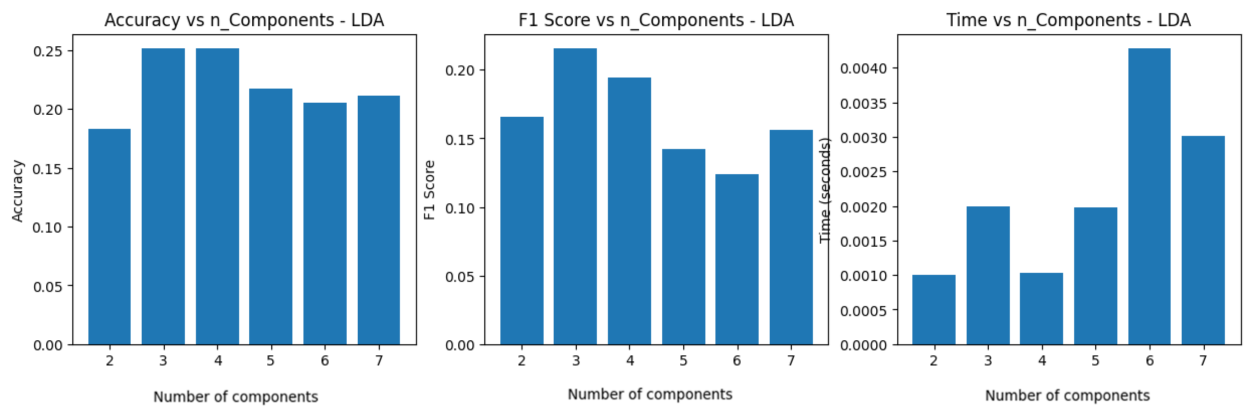
(b) LDA:

LDA also shows poor performance on the MNIST dataset, with consistently low accuracy and f1-score across different numbers of components. This could be attributed to the fact that LDA is more suited for binary or multiclass classification problems with a small number of classes, whereas MNIST is a complex multiclass classification problem with ten classes. As a result, LDA may not effectively capture the discriminative information present in the high-dimensional feature space of the MNIST dataset.

Values for LDA method with MNIST dataset:

```
LDA (n_components=2) - MNIST Accuracy: 0.18 - MNIST F1 Score: 0.17 - MNIST Time: 0.0010 seconds
LDA (n_components=3) - MNIST Accuracy: 0.25 - MNIST F1 Score: 0.22 - MNIST Time: 0.0020 seconds
LDA (n_components=4) - MNIST Accuracy: 0.25 - MNIST F1 Score: 0.19 - MNIST Time: 0.0010 seconds
LDA (n_components=5) - MNIST Accuracy: 0.22 - MNIST F1 Score: 0.14 - MNIST Time: 0.0020 seconds
LDA (n_components=6) - MNIST Accuracy: 0.21 - MNIST F1 Score: 0.12 - MNIST Time: 0.0043 seconds
LDA (n_components=7) - MNIST Accuracy: 0.21 - MNIST F1 Score: 0.16 - MNIST Time: 0.0030 seconds
```

Plots for LDA method with MNIST dataset:



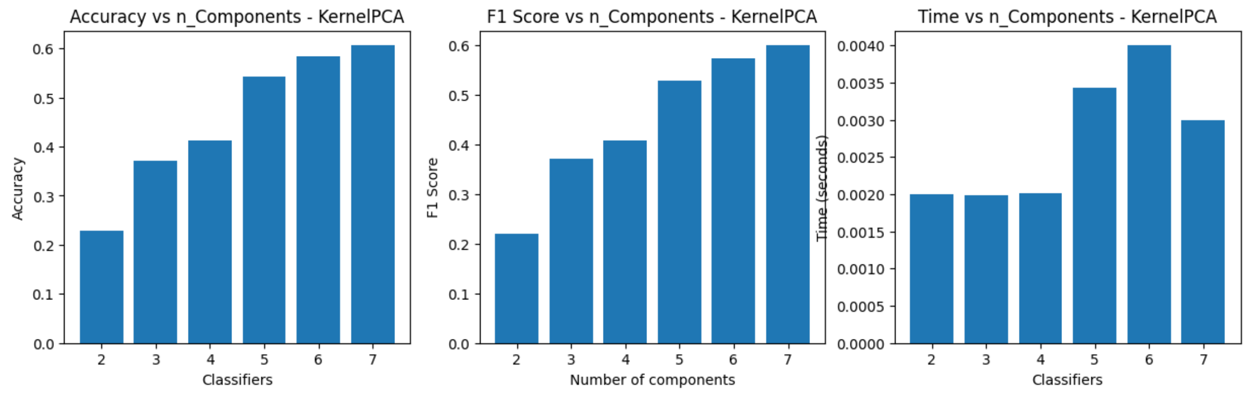
(c) KernelPCA:

KernelPCA emerges as the best-performing method on the MNIST dataset among the three techniques. It achieves higher accuracy and f1-score compared to PCA and LDA, particularly with a larger number of components. This indicates that KernelPCA effectively captures the underlying non-linear relationships in the MNIST dataset, resulting in improved dimensionality reduction performance. However, the computational time for KernelPCA also increases with the number of components, suggesting potential scalability challenges for large-scale applications.

Values for KernelPCA method with MNIST dataset:

```
KernelPCA (n_components=2) - MNIST Accuracy: 0.23 - MNIST F1 Score: 0.22 - MNIST Time: 0.0020 seconds
KernelPCA (n_components=3) - MNIST Accuracy: 0.37 - MNIST F1 Score: 0.37 - MNIST Time: 0.0020 seconds
KernelPCA (n_components=4) - MNIST Accuracy: 0.41 - MNIST F1 Score: 0.41 - MNIST Time: 0.0020 seconds
KernelPCA (n_components=5) - MNIST Accuracy: 0.54 - MNIST F1 Score: 0.53 - MNIST Time: 0.0034 seconds
KernelPCA (n_components=6) - MNIST Accuracy: 0.58 - MNIST F1 Score: 0.57 - MNIST Time: 0.0040 seconds
KernelPCA (n_components=7) - MNIST Accuracy: 0.61 - MNIST F1 Score: 0.60 - MNIST Time: 0.0030 seconds
```

Plots for KernelPCA method with MNIST dataset:



3 Conclusion

Overall, the choice of dimensionality reduction technique depends on the specific characteristics of the dataset and the requirements of the problem at hand. While PCA and LDA are suitable for datasets with low-dimensional linear relationships, KernelPCA stands out for its ability to handle non-linear relationships in high-dimensional datasets like MNIST.

However, practitioners must consider the trade-offs between accuracy and computational efficiency when selecting a dimensionality reduction technique for a particular dataset. Additionally, further experimentation and parameter tuning may be necessary to optimize the performance of each method for specific applications.

Reference:

[1] Fetch dataset from openml by name or dataset id. Scikit-learn user manual. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_openml.html