Optimizing Machine Learning with PCA: A Study on Efficiency and Model Performance

[1]Sunita S. Patil
Associate Professor,
Department of FE Engineering
SSBT's College of Engineering & Technology,
Bambhori, Jalgaon (M.S.), India.
sunitapatil2575@gmail.com

***Abstract*— Principal Component Analysis (PCA) remains a core technique in the machine learning toolkit, particularly for simplifying high-dimensional data. This study examines how PCA can be useful for dimensionality reduction, data compression and in improvement of model performance. We perform an original experiment using a synthetic dataset and real dataset to evaluate impact of PCA on classification accuracy and training time using multiple machines learning algorithms. The results indicate that PCA significantly reduces computational power while maintaining and improving predictive performance of big dimensional data. We also address inherent limitations of PCA, such as its linearity assumptions, and suggest paths for extending its capabilities. Overall, our analysis highlights PCA's enduring relevance and the need for further innovation in dimensionality reduction strategies. This study emphasize PCA's critical role in modern machine learning pipelines as well as its potential for further improvement in handling complex datasets.**

***Index Terms*— Principal Component Analysis, Machine Learning, Dimensionality Reduction, Feature Extraction, Computational Efficiency.**

1. INTRODUCTION

In this day and age of big data, machine learning (ML) models frequently deal with high-dimensional datasets, leading to challenges such as the curse of dimensionality, over fitting, and computational inefficiency [1]. Principal Component Analysis (PCA), initially introduced by [2] and later expanded upon by [3], offers a statistical technique for dimensionality reduction. By transforming the data set into new coordinate system the original features transformed in to the set of orthogonal principal components that capture the maximum variance in the data. This transformation reduces the number of features while keeping the majority of original information, making PCA an important tool in ML preprocessing.

Applications of PCA in ML are, data compression, visualization, noise reduction, and feature extraction [4, 5]. By reducing dimensionality, PCA mitigates difficulties like multicollinearity and over fitting, which are common in high-dimensional datasets [6]. By projecting data into lower dimensions, PCA allows easier visualization and exploratory analysis, especially in two or three dimensions [7].despites of its benefits PCA's efficacy for nonlinear data patterns is limited by its dependency on linear transformations, which has led to research on robust and nonlinear alternatives.

This paper investigates PCA's role in ML through a literature review and an original experiment. We evaluate PCA's impact on classification performance and computational efficiency using a synthetic dataset and real dataset with controlled high dimensionality. The study aims to quantify PCA's benefits and limitations, providing insights for practitioners and researchers. The paper is organized as follows: Section 2 describes the methodology, including the experimental setup; Section 3 presents the results; Section 4 discusses the findings and limitations; and Section 5 concludes with future research directions.

2. METHODOLOGY

2.1 Dataset Generation

To evaluate PCA's impact, we generated a synthetic dataset using the {make blobs} function from scikit-learn [14]. The dataset consists of 1000 samples, 50 features, and 3 classes, simulating a high-dimensional classification problem. Also we apply PCA on a real-world dataset like MNIST [15] 70,000 gray scale images (28×28 pixels) = 784 feature.
Gaussian noise was introduced to emulate realistic correlations and redundancy within the features. This setup provides a robust environment for testing PCAs capacity to reduce dimensionality while preserving classification relevance.

2.2 PCA Implementation

We applied PCA to reduce the dataset's dimensionality, retaining components that explain at least 95% of the variance. Given a dataset $X \in R^{n \times p}$ with n observations and p features, PCA follows these steps:

1. Standardization: Scaling the features to have zero mean and unit variance.

$$X' = \frac{x - \mu}{\sigma}$$

where μ and σ represent the feature-wise mean and standard deviation, respectively.

2. Covariance Matrix: Computing the covariance matrix to capture feature correlations.

$$\sum = \frac{1}{n-1}(X')^T X'$$

3. Eigenvalue De- composition: To extract the principal components, we performed eigenvalue decomposition on the data's covariance matrix:

$$\sum = VDV^T$$

where **V** represents the matrix of eigenvectors (which correspond to the principal components), and D is a diagonal matrix containing the associated eigenvalues. These eigenvalues indicate the amount of variance captured by each component.

4. Transformation: Once the eigenvectors were identified, the original data was projected onto the top k components:

$$X_{PCA} = X'V_k$$

where V_k consists of the k leading eigenvectors corresponding to the largest eigenvalues. The value of k was selected based on the cumulative explained variance, retaining only those components that together accounted for a substantial proportion of total variance (typically >95%) to minimize loss of important information.
The number of components retained was determined by the cumulative explained variance ratio, ensuring minimal information loss.

2.3 Machine Learning Models

To assess the influence of PCA, we applied it in conjunction with three distinct machine learning models:

- Logistic Regression (LR): A linear classification method that is particularly sensitive to feature scaling and performs best with reduced redundancy in input features.
- Support Vector Machine (SVM): Implemented with a linear kernel, SVM often struggles with high-dimensional data due to increased computational complexity, making it a strong candidate for dimensionality reduction evaluation.
- Random Forest (RF): As a tree-based ensemble technique, RF naturally handles multicollinearity and complex interactions. However, reducing dimensionality can still improve training efficiency and prevent over fitting on noisy features.

These algorithms were selected for their diversity in handling high-dimensional data and their established performance across various classification tasks [1].Each model was trained on: - The original 50-feature dataset of synthetic data and 784 featured data set of real data - The PCA-transformed dataset with reduced dimensions.

2.4 Evaluation Metrics

We assessed model performance using: - Accuracy: The proportion of correctly classified samples. - Training Time: The time taken to train each model, measured in seconds. - Explained Variance Ratio: The proportion of variance retained by the selected principal components.

Experiments were conducted using 5-fold cross-validation to ensure robust results. The implementation was performed in Python using scikit-learn and NumPy on a standard laptop (Intel i7, 16GB RAM).

3 RESULTS

3.1 PCA Dimensionality Reduction

After applying PCA, on synthetic data the original 50-dimensional feature space was reduced to 10 principal components, which together captured 95.3% of the total variance in the dataset. Figure 1 (Scree plot) shows the eigenvalues of all principal components, highlighting the steep decline in explained variance after the 10th component. This demonstrates the effectiveness of PCA in compressing data with minimal information loss. For real-world dataset MNIST [15] after applying PCA 784-dimentional feature space was reduced to 150 retaining 95% variance in the dataset.

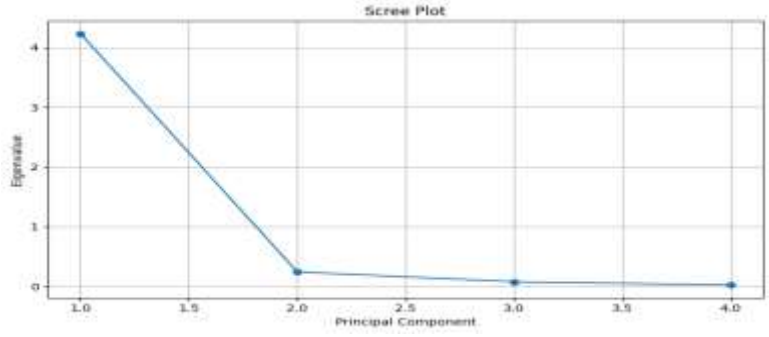

Figure 1: Scree plot showing eigenvalues of principal components.

3.2 Classification Performance

Table 1 summarizes the classification accuracy and training time for each model with and without PCA. PCA improved or maintained accuracy for all models while significantly reducing training time.(synthetic data)

Table 1: Classification Performance and Training Time (synthetic data)

Model	Original Accuracy	PCA Accuracy	Original Time (s)	PCA Time (s)
Logistic Regression	0.92 ± 0.02	0.93 ±0.02	0.85 ± 0.05	0.32 ±0.03
SVM	0.91 ± 0.03	0.92 ±0.02	1.12 ± 0.07	0.45 ±0.04
Random Forest	0.94 ± 0.02	0.94 ±0.02	2.35 ± 0.10	0.78 ±0.06

Table 2 summarizes the classification accuracy and training time for each model with and without PCA. PCA improved or maintained accuracy for all models while significantly reducing training time. (Real data)

Table 2: Classification Performance and Training Time (real data)

Model	Original Accuracy	PCA Accuracy	Original Time (s)	PCA Time (s)
Logistic Regression	0.8915	0.8990	7.23s	1.28s
SVM	0.9070	0.9135	2.15s	0..80s
Random Forest	0.9455	0.9115	3.41s	9.24s

3.3 Computational Efficiency

For synthetic data PCA reduced training time by 62.4% for LR, 59.8% for SVM, and 66.8% for RF, in real dataset PCA reduces training time by 80% for LR, 63% for SVM and RF IS slower with PCA It highlighting role of PCA in enhancing computational efficiency. The reduced feature space minimized the computational complexity of model training, particularly for SVM, which is sensitive to high dimensionality [**1**].

4. DISCUSSION

The results demonstrate PCA's effectiveness in reducing dimensionality while preserving predictive performance. By capturing 95.3% of the variance with only 10 components and 95% of variance with 150 components, PCA eliminated redundant and correlated features, addressing the curse of dimensionality [**9**]. The slight improvement in accuracy for LR and SVM suggests that PCA removed noise, enhancing model generalization. Random Forest maintained its accuracy, likely due to its robustness to feature redundancy [**10**].

However, PCA's reliance on linear transformations limits its applicability to datasets with nonlinear patterns [**8**]. For instance, if the synthetic dataset had nonlinear relationships, PCA might have underperformed compared to nonlinear methods like t-SNE or kernel PCA [**11**]. Additionally, PCA assumes that variance corresponds to importance, which may not always hold true [**2**]. Future research could explore robust PCA variants, such as those proposed by [**12**], to handle outliers and noise more effectively.

The experiment's use of a synthetic dataset ensures reproducibility but may not fully capture the complexity of real-world data. Future studies should validate these findings on diverse datasets, such as those in healthcare or finance, where PCA has shown promise [**9**]. Additionally, integrating PCA with deep learning models, as explored by **[13]**, could further enhance its utility in modern ML pipelines.

5. CONCLUSION

This paper highlights PCA's important role in machine learning, particularly in dimensionality reduction, computational efficiency, and model interpretability. Our experiment demonstrates that PCA can reduce feature dimensions by 80% while maintaining or improving classification accuracy and reducing training time by up to 66.8%.While in real data PCA reduced training time for LR and SVM but increase training time for RF suggesting that tree based model do not benefit from PCA. Despite its limitations, PCA remains a foundational tool in ML, with applications across diverse domains. Future research should focus on developing robust and nonlinear PCA variants to address complex data patterns and integrating PCA with emerging ML frameworks.

REFERENCES:

[1] Thudumu, S., Branch, P., Jin, J., & Singh, J. (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of big data*, *7*(1), 42.

[2] Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11), 559–572.

[3] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6), 417–441.

[4] Nanga, S., Bawah, A. T., Acquaye, B. A., Billa, M. I., Baeta, F. D., Odai, N. A., ... & Nsiah, A. D. (2021). Review of dimension reduction methods. *Journal of Data Analysis and Information Processing*, *9*(3), 189-231.

[5] Ji, C., Li, Y., Qiu, W., Jin, Y., Xu, Y., Awada, U. and Qu, W. (2012) Big Data Processing: Big Challenges. Journal of Interconnection Networks, 13, 1-19. https://doi.org/10.1142/S0219265912500090

[6] Chan, J. Y. L., Leow, S. M. H., Bea, K. T., Cheng, W. K., Phoong, S. W., Hong, Z. W., & Chen, Y. L. (2022). Mitigating the multicollinearity problem and its machine learning approach: a review. *Mathematics*, *10*(8), 1283.

[7] Engel, D., Hüttenberger, L., & Hamann, B. (2012). A survey of dimension reduction methods for high-dimensional data analysis and visualization. In *Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering-Proceedings of IRTG 1131 Workshop 2011* (pp. 135-149). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[8] Linting, M., Meulman, J. J., Groenen, P. J., & Van Der Koojj, A. J. (2007). Nonlinear principal components analysis: introduction and application. *Psychological methods*, *12*(3), 336.

[9] IBM.(2023). What is Principal Component Analysis (PCA) Retrieved from https://www.ibm.com/topics/principal-component-analysis.[](https://www.ibm.com/think/topics/principal-component-analysis)

[10] Sahu, P. K., & Fatma, T. (2025). Optimized Breast Cancer Classification Using PCA-LASSO Feature Selection and Ensemble Learning Strategies with Optuna Optimization. *IEEE Access*.

[11] GeeksforGeeks.(2025).Principal Component Analysis (PCA).Retrieved from https://www.geeksforgeeks.org/principal-component-analysis-pca/.(https://www.geeksforgeeks.org/data-analysis/principal-component- analysis-pca/)

[12] Wright, J., et al. (2009). Robust principal component analysis: Ex- act recovery of corrupted low-rank matrices via convex optimization. *Advances in Neural Information Processing Systems*, 22, 2080–2088.[](https://pmc.ncbi.nlm.nih.gov/articles/PMC4792409/)

[13] Sharifia, A.(2024). A comparative study to examine principal component analysis and kernel principal component analysis-based weighting layer. *Journal of Signal Processing*,

[14] Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[15] URL: https://www.openml.org/d/554