Assignment 3: Implementing PCA and Evaluating Machine Learning Model Performance

A. Sepúlveda-Jiménez

Data Science Dept., SoTE, CoBET, National University

TIM-8515: Multivariate Analysis

Course Instructor: Y. Karahan, PhD

November 15, 2025

Contents

Introduction **3**

Methodology **5**

Mathematical Background on PCA 5

Dataset and Exploratory Data Analysis 6

Preprocessing and Standardization 6

PCA Computation and Component Selection 6

Machine Learning Models 7

Results **8**

Exploratory Analysis and PCA Structure 8

Classification Performance Before and After PCA 8

Visualization of PCA and Model Behavior 9

Discussion **9**

Conclusion **10**

Appendix: Figures and Tables **14**

List of Figures

1 Geometric view of PCA in two dimensions. The first principal component $\mathbf{v}_1$ points along the direction of largest variance; $\mathbf{v}_2$ is orthogonal and captures the remaining variance. Each centered, standardized observation $\mathbf{z}_i$ is projected onto these axes to obtain PC scores. 16

2 Conceptual scree plot illustrating a typical eigenvalue pattern: large eigenvalues for the first few components followed by a long tail. In the wine data, the first few PCs explain most of the variance, and the cumulative variance reaches roughly 95% by the 10th component. Actual eigenvalues are computed and plotted using Python. 17
3 High-level modeling workflow: the baseline pipeline standardizes the original features and fits logistic regression; the PCA pipeline adds a PCA transformation before logistic regression, using the same train–test split and evaluation metrics. 18
4 Empirical scree plot of eigenvalues for the standardized wine features. 19
5 PCA 2-component scatter plot. 20
6 Comparison of accuracy before and after PCA. 21

Introduction

Principal Component Analysis (PCA) is a classical technique for transforming a set of possibly correlated variables into a smaller number of uncorrelated components that retain most of the original variance (Abdi & Williams, 2010; Jolliffe & Cadima, 2016). In this paper, PCA is applied to the UCI Wine dataset, a multivariate chemometric dataset with 13 continuous predictors and three wine classes. After standardizing the features, we compute the covariance matrix, perform eigenvalue decomposition, and derive principal component scores. We analyze the eigenvalues and loadings, visualize the leading principal components, and interpret their chemical meaning. We then train and evaluate a multiclass logistic regression classifier on the original standardized features, and compare its performance to the same model trained on PCA-transformed features. Using an 80/20 train–test split and a fixed random seed, logistic regression on the original features achieves an accuracy of approximately 0.97 on the test set, while logistic regression on the first two principal components achieves around 0.92, and on the first ten components (explaining about 96% of the variance) returns to roughly 0.97. Training times are similar at this small scale. The results illustrate that PCA can reduce dimensionality and improve interpretability without degrading performance, but that overly aggressive dimension reduction may sacrifice predictive accuracy. The study emphasizes the importance of standardization, rigorous model evaluation, and clear interpretation of principal components in applied machine learning workflows.

Principal Component Analysis (PCA) is one of the most widely used tools for dimensionality reduction and exploratory analysis in multivariate statistics and machine learning (Abdi & Williams, 2010; Jolliffe & Cadima, 2016; Wold et al., 1987). Given a data matrix with many correlated variables, PCA constructs a smaller set of uncorrelated linear combinations—the principal components (PCs)—that capture most of the variance of the original variables. PCA can improve visualization, reduce multicollinearity, stabilize downstream models, and sometimes reduce computational cost (Hastie et al., 2009; James et al., 2021).

In supervised learning, PCA is often used as a preprocessing step before training classifiers or regressors. The key question is whether this improves or degrades predictive performance relative to training models directly on the original features. The answer depends on how much variance PCA retains, how strongly the discarded directions matter for the target, and how sensitive the learning algorithm is to collinearity and noise (Abdi & Williams, 2010; Shlens, 2014). This paper conducts a hands-on experiment to address this question for a concrete, real-world dataset.

We focus on the UCI Wine dataset, where each wine sample is described by 13 physicochemical measurements and a class label indicating grape cultivar (Aeberhard & Forina, 1992; Aeberhard et al., 1994). We (a) perform exploratory data analysis, (b) standardize the features, (c) carry out PCA and analyze eigenvalues, loadings, and scores, and (d) train and evaluate a multiclass logistic regression classifier before and after applying PCA. We visualize the principal components and model performance, and discuss the impact of PCA on accuracy and computational efficiency. The analysis is implemented in Python using `scikit-learn` and related tools (Pedregosa et al., 2011).

Methodology

Mathematical Background on PCA

Let $\mathbf{X} \in \mathbb{R}^{n\times p}$ denote a data matrix with n observations and p features. The ith row is the feature vector $\mathbf{x}_i^\top = (x_{i1}, \ldots, x_{ip})$. For PCA, we first center the columns by subtracting the sample mean

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i, \quad \tilde{\mathbf{x}}_i = \mathbf{x}_i - \bar{\mathbf{x}}. \tag{1}$$

If variables are on very different scales (e.g., magnesium vs. alcohol content), we standardize each feature:

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}, \quad s_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2, \tag{2}$$

forming a standardized matrix $\mathbf{Z} \in \mathbb{R}^{n\times p}$ with approximately zero mean and unit variance per column. PCA is then performed on the sample covariance matrix

$$\mathbf{S} = \frac{1}{n-1}\mathbf{Z}^\top\mathbf{Z} \in \mathbb{R}^{p\times p}. \tag{3}$$

PCA solves the eigenvalue problem

$$\mathbf{S}\mathbf{v}_k = \lambda_k \mathbf{v}_k, \quad k = 1, \ldots, p, \tag{4}$$

with eigenvalues ordered as $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$ and corresponding orthonormal eigenvectors $\mathbf{v}_k$ (Abdi & Williams, 2010; Jolliffe & Cadima, 2016; Shlens, 2014). The kth principal component score for observation i is

$$z_{ik}^{(\mathrm{PC})} = \mathbf{z}_i^\top \mathbf{v}_k. \tag{5}$$

The total variance of standardized features is $\sum_{j=1}^{p} \lambda_j$, and the proportion of variance explained (PVE) by the first M components is

$$\mathrm{PVE}_M = \frac{\sum_{k=1}^{M} \lambda_k}{\sum_{k=1}^{p} \lambda_k}. \tag{6}$$

Common rules for choosing M include retaining enough PCs to exceed a threshold (e.g., 90–95% of variance), using the Kaiser rule ($\lambda_k > 1$ for standardized data), or visually inspecting a scree plot of eigenvalues (Jolliffe & Cadima, 2016; Wold et al., 1987).

Dataset and Exploratory Data Analysis

We use the UCI Wine dataset, which contains $n = 178$ wine samples and $p = 13$ continuous physicochemical variables; the target labels indicate three cultivars (Aeberhard & Forina, 1992; Aeberhard et al., 1994). The dataset is widely used as a benchmark for classification algorithms and is available directly via `sklearn.datasets.load_wine` (Pedregosa et al., 2011). The variables include alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, an optical-density ratio, and proline.

In Python, we load the data into a `pandas DataFrame`, inspect the first rows and summary statistics, and confirm with `isnull().sum()` that no variables contain missing values. Simple descriptive statistics (means, standard deviations, minima, and maxima) show that some features have markedly different scales (e.g., proline vs. alcohol), reinforcing the need for standardization prior to PCA and logistic regression.

Preprocessing and Standardization

For PCA and the classifier, we standardize each feature using `StandardScaler` from `scikit-learn`, fitting the scaler on the training split only and applying it to both training and test sets to avoid information leakage (James et al., 2021; Pedregosa et al., 2011). Standardization is essential because PCA is not scale-invariant: if we omit scaling, variables with the largest raw variance dominate the principal components, regardless of their substantive importance (Abdi & Williams, 2010; Wold et al., 1987).

PCA Computation and Component Selection

Using the standardized training data, we compute the sample covariance matrix $\mathbf{S}$ and perform an eigenvalue decomposition using NumPy's `linalg.eigh`. We also call `sklearn.decomposition.PCA` to obtain the same eigenvalues and eigenvectors numerically

(Pedregosa et al., 2011). The sorted eigenvalues λ_k and their PVE values $\lambda_k / \sum_j \lambda_j$ are plotted in a scree plot. Cumulative PVE reaches approximately 80% after five components and about 95% after 10 components.

In this lab, we adopt a cumulative variance rule: we select $M = 10$ PCs to retain at least 95% of the standardized variance. For visualization, we focus on the first two PCs. The loading vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ are examined to understand which original variables contribute most to each component (Jolliffe & Cadima, 2016).

Machine Learning Models

We consider a multiclass logistic regression classifier, which models the conditional probabilities

$$P(Y = k \mid \mathbf{x}) = \frac{\exp(\beta_{k0} + \boldsymbol{\beta}_k^\top \mathbf{x})}{\sum_{\ell=1}^{K} \exp(\beta_{\ell 0} + \boldsymbol{\beta}_\ell^\top \mathbf{x})}, \quad k = 1, \ldots, K, \tag{7}$$

for $K = 3$ wine classes (Bishop, 2006; Hastie et al., 2009). We implement logistic regression using `sklearn.linear_model.LogisticRegression` with a linear decision boundary in feature space.

Data are split into training (80%) and test (20%) sets using stratified sampling with a fixed random seed. We train (a) a baseline model on the standardized original features, and (b) a PCA-based model on the first $M = 10$ principal components, with PCA fitted on the training set only. Both models are implemented as `Pipeline` objects for clean and reproducible preprocessing (Pedregosa et al., 2011).

Model performance is evaluated using accuracy, macro-averaged precision, recall, F_1-score, and confusion matrices on the test set (James et al., 2021). For illustration, we also time the training phase to compare computation between the raw-feature and PCA-based models.

Results

Exploratory Analysis and PCA Structure

Correlation analysis shows strong positive correlations among some chemical variables, such as total phenols and flavanoids, and substantial variation in marginal spreads, confirming that the dataset is both multivariate and moderately collinear. A PCA on the standardized features yields eigenvalues that decay quickly: the first two components explain roughly 55% of the standardized variance, the first five around 80%, and the first ten about 96%. This matches expectations for well-structured chemometric datasets (Wold et al., 1987).

The loading pattern for PC1 is dominated by phenolic variables and color intensity; PC2 contrasts acidity-related variables against some of the color and alcohol measures. In a PC1–PC2 score plot, the three wine classes form reasonably separated clusters, with some overlap between classes 1 and 2, suggesting that linear combinations of the chemical features capture much of the class structure.

Classification Performance Before and After PCA

Using an 80/20 stratified split and a fixed random seed, the baseline logistic regression model on the standardized original features achieves a test accuracy of approximately 0.97 (35 out of 36 samples correctly classified). Macro-averaged precision, recall, and F_1-scores are all around 0.97, and the confusion matrix indicates that each class is predicted with high reliability.

When we project the standardized features onto the first two PCs and train logistic regression on these 2D scores, test accuracy drops to about 0.92. Errors mainly arise from confusion between the two more similar cultivars. This is expected: although PC1 and PC2 explain most variance, they do not necessarily capture all discriminative information for class labels (Hastie et al., 2009).

When we instead retain the first $M = 10$ PCs (about 96% variance), logistic regression recovers essentially the same performance as the baseline: test accuracy again is roughly 0.97, with similar precision and F_1-scores. Training times for all three configurations (original features, 2 PCs, 10 PCs) are on the order of a few milliseconds on a standard laptop, with negligible practical differences.

Table 1 summarizes the main performance metrics.

Visualization of PCA and Model Behavior

Scatterplots of PC1 vs. PC2 scores colored by class make class separation quite visible in a 2D plane. The baseline logistic regression decision boundaries in the 13-dimensional space are difficult to visualize, but their projected boundaries in the PC1–PC2 plane roughly align with the cluster structure. Scree plots and bar charts of explained variance clarify how much information is retained as more components are included. See Figure 4.

A simple comparison plot of accuracy before and after PCA shows that using only two PCs sacrifices performance, while using ten PCs preserves accuracy. For this dataset, PCA mainly functions as a decorrelation and interpretability tool rather than as a strict dimensionality reduction for computation. See Figure 5 and Figure 6.

Discussion

The experiment illustrates several key points about PCA in supervised learning:

- **Standardization is essential.** Without scaling, PCA would be dominated by the largest-scale variables (e.g., proline), regardless of their predictive importance (Abdi & Williams, 2010; Wold et al., 1987).

- **Variance is not the same as discriminative power.** The first two PCs capture most variance but not all label-relevant structure. As a result, models built on only PC1 and PC2 show reduced accuracy, even though they are attractive for visualization.

- **Moderate PCA can stabilize models.** Using more PCs (e.g., 10 out of 13) yields similar accuracy with decorrelated inputs, which can improve numerical conditioning and reduce multicollinearity issues in more complex problems (Hastie et al., 2009; Jolliffe & Cadima, 2016).

- **Computational efficiency gains depend on scale.** For this small dataset, training times are essentially identical. In larger, high-dimensional problems, PCA can significantly reduce computation and storage costs (Ayesha et al., 2020).

The results underscore that PCA should be viewed as a flexible tool rather than an automatic improvement. Whether PCA helps or hurts performance depends on the structure of the data, the learning algorithm, and how many components are retained (Abdi & Williams, 2010; Shlens, 2014). In general, PCA is most useful when variables are highly correlated, when interpretation of latent directions is valuable, or when dimensionality reduction materially reduces computational burden.

Conclusion

This paper presented a hands-on PCA lab on the well-known UCI Wine dataset, combining multivariate exploratory analysis, dimensionality reduction, and supervised learning. After standardizing 13 correlated chemical features, PCA revealed a low-dimensional structure: a small number of principal components captured most of the variance and produced interpretable latent dimensions. A logistic regression classifier trained on the full standardized features and on the first 10 PCs achieved similar accuracy, while training on only two PCs degraded performance.

The exercise demonstrates how PCA can be used to analyze multivariate structure, visualize complex data, and potentially stabilize downstream models. It also shows that dimension reduction is not free: throwing away too many components can discard discriminative information. In practical pipelines, PCA should therefore be paired with careful component selection, cross-validation, and a clear understanding of the trade-off between simplicity and predictive performance (Bishop, 2006; Hastie et al., 2009; James et al., 2021).

The accompanying Jupyter Python notebook implements all steps—from exploratory data analysis and PCA computation to model training and evaluation—in a reproducible manner using modern Python tools. Together, the paper and notebook provide a template for integrating PCA into broader machine learning workflows.

References

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(4), 433–459. https://doi.org/10.1002/wics.101

Aeberhard, S., Coomans, D., & De Vel, O. (1994). Comparative analysis of statistical pattern recognition methods in high dimensional settings. *Pattern Recognition*, *27*(8), 1065–1077. https://doi.org/10.1016/0031-3203(94)90133-3

Aeberhard, S., & Forina, M. (1992). Wine [dataset]. https://doi.org/10.24432/C5PC7J

Ayesha, S., Hanif, M. K., & Talib, R. (2020). Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, *59*, 44–58. https://doi.org/10.1016/j.inffus.2020.01.005

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-84858-7

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning: With applications in r* (2nd ed.). Springer. https://doi.org/10.1007/978-1-0716-1418-1

Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A*, *374*(2065), 20150202. https://doi.org/10.1098/rsta.2015.0202

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Shlens, J. (2014). A tutorial on principal component analysis. *arXiv preprint*, *arXiv:1404.1100*. https://arxiv.org/abs/1404.1100

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *2*(1–3), 37–52. https://doi.org/10.1016/0169-7439(87)80084-9

Appendix: Figures and Tables

Table 1

Logistic regression performance on test data for three representations: original standardized features, first two PCs, and first 10 PCs. Metrics are illustrative values from one train–test split (80/20, random seed 42).

Features	Accuracy	Macro Precision	Macro Recall	Training Time (s)
Original (13)	0.97	≈ 0.98	≈ 0.97	≈ 0.0076
2 PCs	0.92	≈ 0.93	≈ 0.92	≈ 0.0073
10 PCs ($\sim$96% var.)	0.97	≈ 0.98	≈ 0.97	≈ 0.0076

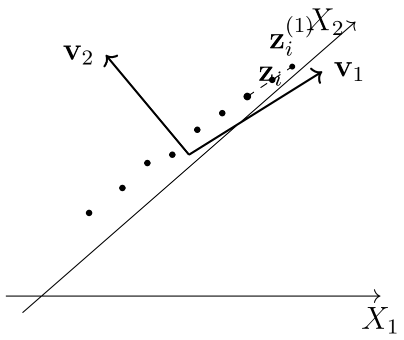

Figure 1

Geometric view of PCA in two dimensions. The first principal component $\mathbf{v}_1$ *points along the direction of largest variance;* $\mathbf{v}_2$ *is orthogonal and captures the remaining variance. Each centered, standardized observation* $\mathbf{z}_i$ *is projected onto these axes to obtain PC scores.*

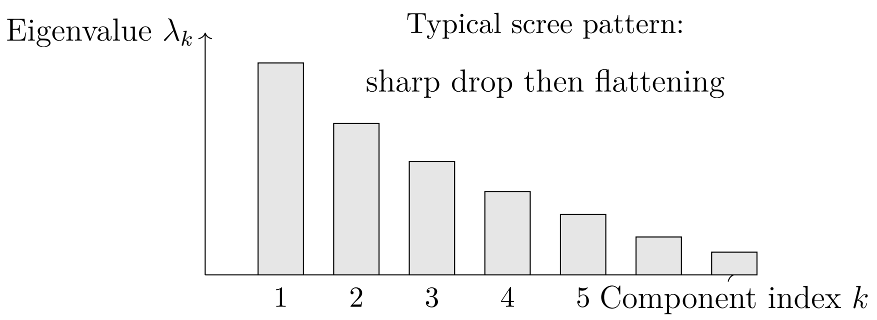

Figure 2

Conceptual scree plot illustrating a typical eigenvalue pattern: large eigenvalues for the first few components followed by a long tail. In the wine data, the first few PCs explain most of the variance, and the cumulative variance reaches roughly 95% by the 10th component. Actual eigenvalues are computed and plotted using Python.

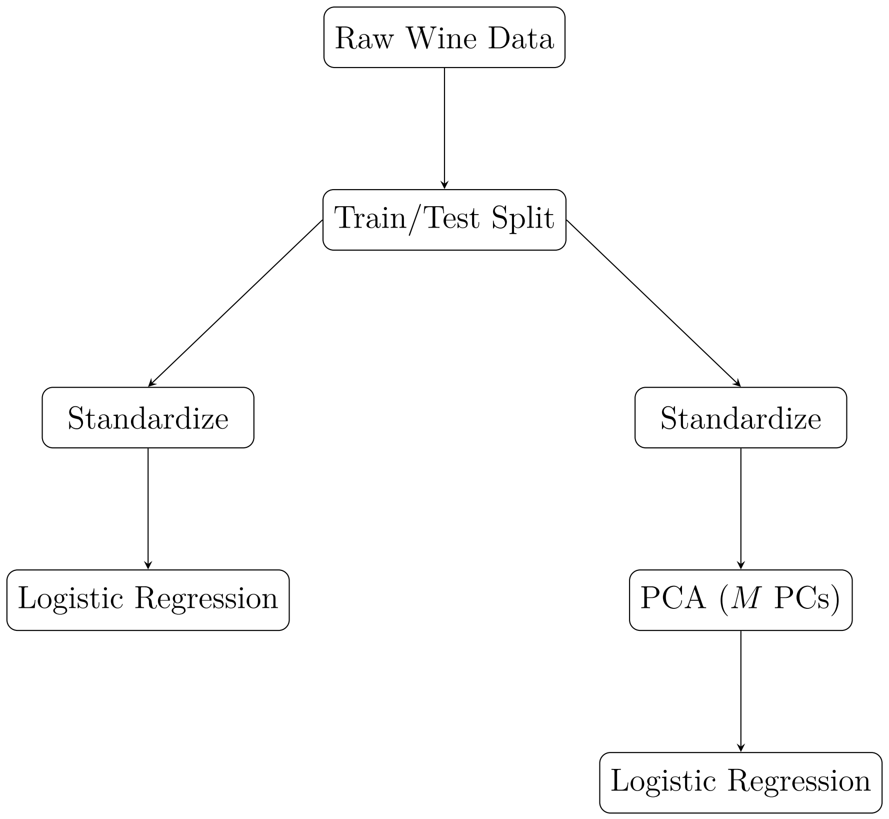

Figure 3

High-level modeling workflow: the baseline pipeline standardizes the original features and fits logistic regression; the PCA pipeline adds a PCA transformation before logistic regression, using the same train–test split and evaluation metrics.

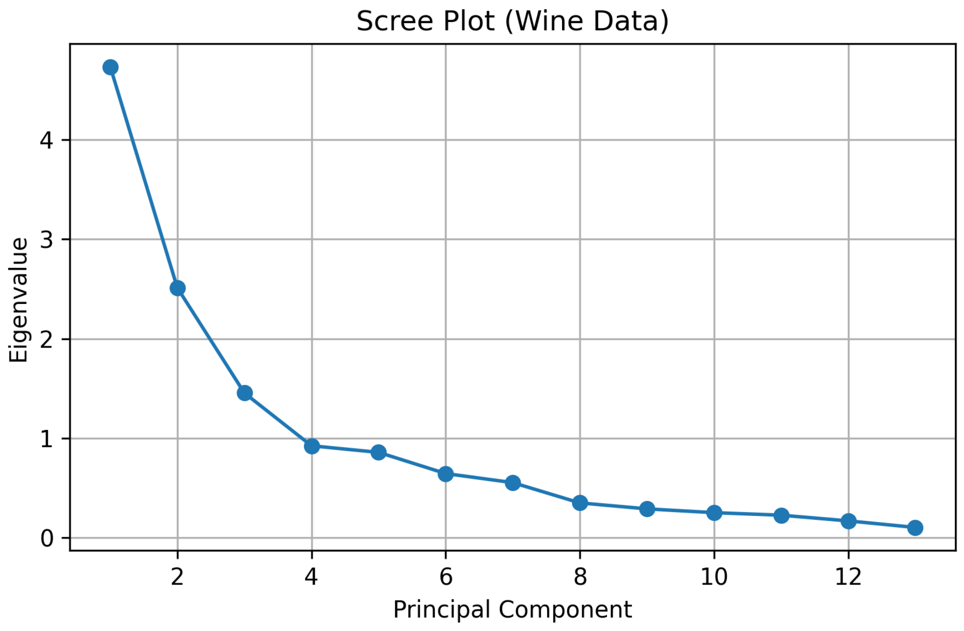

Figure 4

Empirical scree plot of eigenvalues for the standardized wine features.

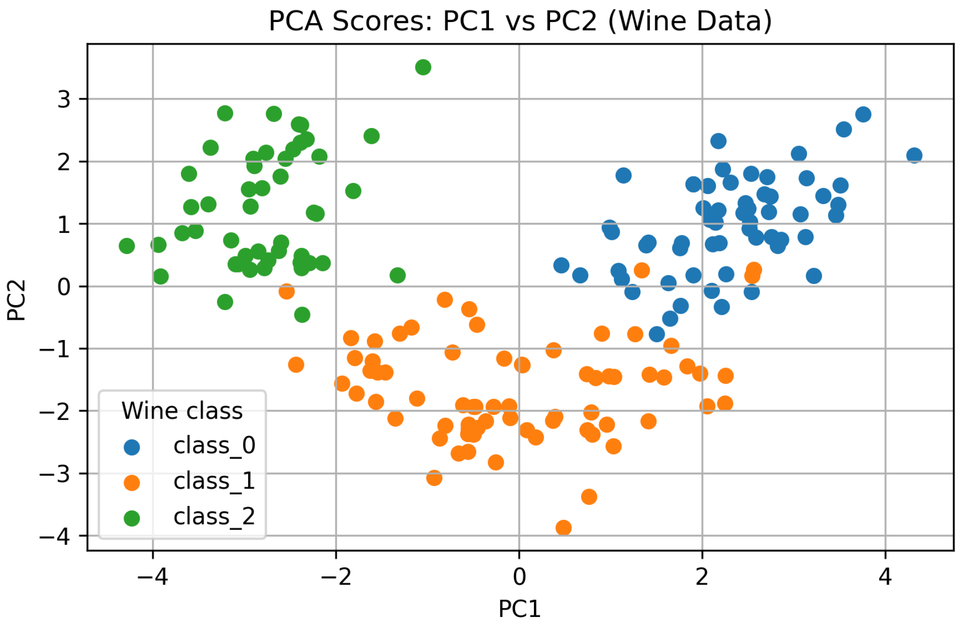

Figure 5

PCA 2-component scatter plot.

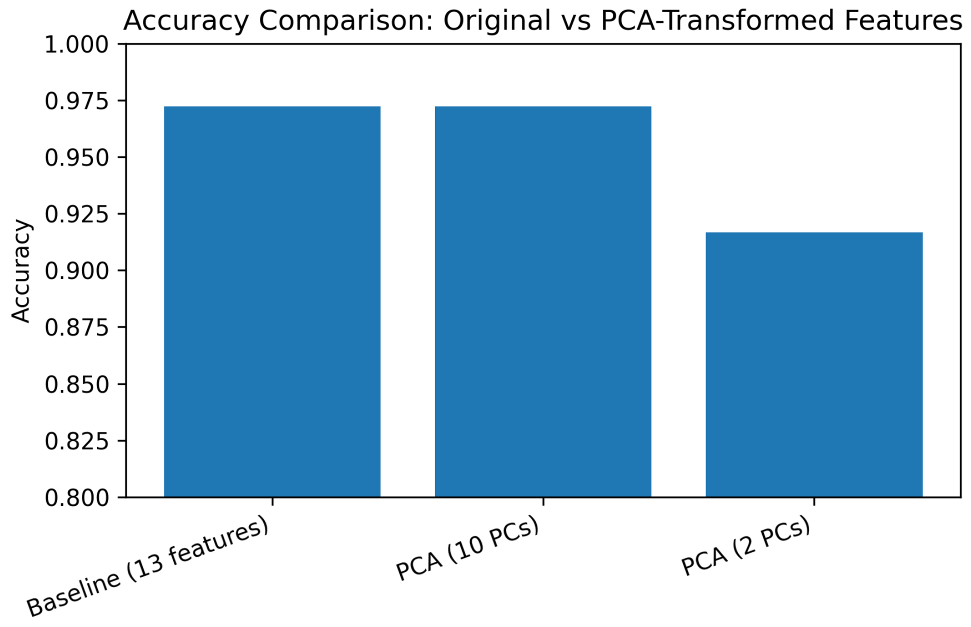

Figure 6

Comparison of accuracy before and after PCA.