Principal Component Analysis on Rice Seeds Using Machine Learning

Anshuman, **Student ID** - 40263349
GitHub Link - https://github.com/anshudabar/INSE6220

***Abstract*—Rice stands out as a primary food staple for a significant portion of the global population, particularly in regions like Asia and Africa. Its widespread consumption is attributed to its genetic diversity, which manifests in various traits among rice varieties. This study utilizes Principal Component Analysis (PCA) on a dataset containing rice seeds to distinguish between different rice varieties and assess seed quality. The classification task involves employing algorithms such as Light Gradient Boosting Machine (Lightgbm), Naïve Bayes (NB), and K-Nearest Neighbors Classifier (KNN) on both the original dataset and a transformed dataset (after PCA application) to discern rice varieties based on their features. Performance evaluation of these algorithms utilizes rice images and features extracted from them, with statistical metrics like confusion matrix, F1 score, and decision boundary aiding in assessing classification effectiveness. By leveraging these metrics, the study calculates the training and testing efficacy of the algorithms.**

I. Introduction

Rice farming holds a central position in the agricultural sector of numerous countries, with its market value being determined by several factors. Texture, shape, color, and fracture rate are among the key parameters influencing its pricing. The primary goal of classification is to ensure the authenticity of rice seeds, distinguishing them from other varieties. Machine learning algorithms facilitate the rapid and reliable analysis of vast amounts of data, offering a crucial tool in rice production to enhance final product quality and adhere to food safety standards in an automated, cost-effective, efficient, and non-destructive manner. With advancements in information technology, the analysis and identification of rice seeds are now carried out through automated computer-aided vision systems. This study focuses on five distinct rice varieties: Ipsala, Jasmine, Basmati, Karacadag, and Arborio. The dataset comprises 1334 data points with 11 attributes representing morphological features crucial for classifying rice seeds and discerning between different varieties.

The dataset utilized for training the algorithms was compiled by extracting relevant features. Subsequently, these features underwent classification utilizing algorithms such as Light Gradient Boosting Machine (Lightgbm), Naïve Bayes (NB), and K-Nearest Neighbors Classifier (KNN) from the realm of machine learning. To enhance efficiency, Principal Component Analysis (PCA) was employed to identify the most salient features before classification. Following data pre-processing and transformation through PCA, the efficacy of the transformation was assessed using three classification algorithms (Light Gradient Boosting Machine, Naive Bayes, and K-Nearest Neighbors).

The structure of the report is as follows: Section II delves into the intricacies of principal component analysis; Section III introduces the classification algorithms; Section IV presents and analysis of Data-set; Section V is baded on PCA results, and Section VI discusses the classification outcomes., and Section VII provides a summary of this study.

II. Principal Component Analysis

Principal Component Analysis (PCA) serves as an exploratory method for simplifying intricate data sets in a multivariate context. This technique involves an orthogonal linear transformation of feature vectors into uncorrelated vectors. Consequently, the highest variance in the data projects onto the first coordinate (referred to as the first principal component), with subsequent coordinates capturing decreasing variances. By discerning the most crucial features and eliminating uncorrelated ones, PCA aids in streamlining the dataset. Moreover, reducing the number of features through PCA substantially decreases the time required for model training, thus mitigating the risk of overfitting in the dataset.

PCA algorithm

1) Data Standardization - To ensure an equitable analysis of each variable, it's essential to standardize the data before conducting PCA. This approach prevents biased results from skewing the analysis outcome. Standardizing the variables to a uniform scale involves subtracting the mean and dividing by the standard deviation for each variable using the following formula. This procedure guarantees that every feature possesses a mean of 0 and a variance of 1

$$Z = \frac{n-\mu}{\sigma} \quad (1)$$

$$Where\ Mean(\mu) = \frac{Sum\ of\ the\ terms}{total\ number\ of\ terms}$$

$$Where\ SD(\sigma) = \sqrt{\frac{€(x - Mean)^2}{n(total\ number\ of\ terms}}$$

The centered data matrix (Y) can be represented as

$$\mathbf{Y = HX} \quad (2)$$

2) Computing Covariance Matrix

To discern the highly correlated variables, the covariance matrix can be computed using the provided formula

$$cov(x, y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{x})(Y_i - \bar{y}) \quad (3)$$

This calculation results in an NxN symmetrical matrix (S) that encompasses the covariances among all conceivable datasets, thereby elucidating the correlations among multiple features within a multidimensional dataset. The covariance matrix is represented as

$$S = \frac{1}{n-1} YY^{\intercal} \quad (4)$$

3) Eigen Decomposition

An eigenvector (V) is a non-zero vector that undergoes, at most, a scalar multiplication when subjected to a linear transformation. The associated eigenvalue (λ) represents the scalar by which the eigenvector is scaled. For a square covariance matrix A, a vector V, and a scalar λ, the following condition holds. Given that v is a non-zero vector, rearranging yields the following equation (6).

$$AV = \lambda V \quad (5)$$

$$det(A - \lambda I) = 0 \quad (6)$$

$$S = A\Lambda A^{\intercal}, \quad (7)$$

4) Principal Components

The computation yields the transformed matrix Z, which is of size n×n. In Z, the rows signify the observations, while the columns denote the principal components (PCs). The number of PCs corresponds to the dimension of the original data matrix. The equation defining Z can be expressed as follows:

$$Z = YA \quad (8)$$

III. Machine Learning Based Classification Algorithms

`

Classification models hold significant importance across diverse fields. They are employed in class determination, aiming to ascertain the class to which the data pertains. Operating by generating predictions, classification models leverage the shared attributes within the data to categorize it effectively.

A. Random Forest Classifier

The Random Forest Classifier is a robust ensemble learning method known for its versatility and effectiveness in various applications. It operates by constructing a multitude of decision trees during training and outputs the mode of the classes for classification tasks or the mean prediction for regression tasks. Designed for high-performance computing, the Random Forest Classifier efficiently handles large datasets, benefiting from its parallel processing capabilities. One of its key features is its ability to mitigate overfitting by averaging multiple decision trees, each trained on a different subset of the data.

B. K-Nearest Neighbor(K-NN)

K-Nearest Neighbors (K-NN) represents a non-parametric lazy learning algorithm that encompasses storing all instances corresponding to training data points in an n-dimensional space. Upon receiving unknown discrete data, it scrutinizes the closest k instances saved (nearest neighbors) and returns the most prevalent class as the prediction. For real-valued data, it computes the mean of the K-nearest neighbors. Within the K-NN algorithm, each point is virtually plotted in a multi-dimensional space, where each axis corresponds to a distinct variable. When there's test data available, it undergoes processing sequentially with all existing data. The test data typically shares numerous neighbors closely resembling its characteristics. Consequently, the k nearest pieces of data to the test data are selected. Subsequently, the class with the most instances from the selected data is identified, and the tested data is classified accordingly. In our investigation, a k value of 1 was opted for.

C. Naive Bayes

In machine learning, naïve Bayes operates as a "probabilistic classifier," grounded in the application of Bayes theorem while assuming strong (naïve) independence between the features. Training NB efficiently takes place within a supervised learning framework, contingent on the probability model's structure. NB emerges as a highly accurate classifier applicable in scenarios lacking a correlation between a specific feature and other features within the system. In NB, the learning module leverages this model to forecast the classification of a new sample by constructing an estimated model of existing features. Reconceptualizing the aforementioned equation using Bayesian probability terminology yields:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \quad (9)$$

$$Posterior = \frac{prior\ x\ likelihood}{evidence} \quad (10)$$

IV. Data-set Description

D. Data Classification

The dataset utilized in this analysis is the Rice Seed Data obtained from Muratkolu, comprising a total of 1334 observations. Our data set includes five varieties of rice - Jasmine, Basmati, Arboria, Ipsala, and Karacadag.

	AREA	PERIMETER	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	EQDIASQ	SOLIDITY	CONVEX_AREA	EXTENT	ASPECT_RATIO	ROUNDNESS	CLASS
0	7805	437.915	209.8215	48.0221	0.9735	99.6877	0.9775	7985	0.3547	4.3693	0.5114	Basmati
1	7503	340.757	138.3361	69.8417	0.8632	97.7400	0.9660	7767	0.6637	1.9807	0.8120	Arborio
2	5124	314.617	141.9803	46.5784	0.9447	80.7718	0.9721	5271	0.4760	3.0482	0.6505	Jasmine
3	7990	437.085	201.4386	51.2245	0.9671	100.8622	0.9659	8272	0.6274	3.9325	0.5256	Basmati
4	7433	342.893	140.3350	68.3927	0.8732	97.2830	0.9831	7561	0.6006	2.0519	0.7944	Arborio
5	11648	445.527	178.4659	84.9327	0.8795	121.7813	0.9599	12135	0.5660	2.1013	0.7374	Ipsala
6	7621	450.325	219.0981	45.2301	0.9785	98.5056	0.9718	7842	0.3398	4.8441	0.4722	Basmati
7	8582	367.338	146.6128	75.5406	0.8570	104.5320	0.9740	8811	0.6423	1.9408	0.7992	Arborio
8	5450	320.362	139.9963	50.6910	0.9321	83.3016	0.9626	5662	0.5502	2.7618	0.6673	Jasmine
9	6781	307.023	116.2443	74.8093	0.7654	92.9184	0.9819	6906	0.7208	1.5539	0.9040	Karacadag
10	9656	451.287	206.7047	60.2333	0.9566	110.8801	0.9748	9906	0.5692	3.4317	0.5958	Jasmine
11	8460	362.862	149.4804	72.9154	0.8730	103.7864	0.9822	8613	0.7673	2.0501	0.8074	Arborio
12	7090	401.778	190.9745	47.8482	0.9681	95.0119	0.9789	7243	0.4039	3.9913	0.5519	Basmati
13	6208	300.124	113.9325	70.6069	0.7848	88.9060	0.9658	6428	0.7630	1.6136	0.8661	Karacadag
14	7116	411.103	197.0442	46.7342	0.9715	95.1860	0.9679	7352	0.4445	4.2163	0.5291	Basmati
15	6276	288.886	108.0128	74.3469	0.7254	89.3916	0.9866	6361	0.7821	1.4528	0.9450	Karacadag
16	5832	282.750	108.9852	68.6358	0.7768	86.1715	0.9846	5923	0.7172	1.5879	0.9167	Karacadag
17	6628	311.431	119.7466	72.0507	0.7987	91.8642	0.9687	6842	0.7201	1.6620	0.8588	Karacadag
18	5251	327.214	151.2670	45.0754	0.9546	81.7666	0.9771	5374	0.4392	3.3559	0.6163	Jasmine
19	12769	447.830	184.4764	88.6296	0.8770	127.5068	0.9804	13024	0.6733	2.0814	0.8001	Ipsala
20	5392	326.326	142.1714	48.9332	0.9389	82.8572	0.9620	5605	0.4897	2.9054	0.6363	Jasmine

Figure 1. Data Set

The Above figure illustrates the initial 20 rows of the rice seed data. From this dataset, two types of variables can be identified:

- Dependent Variable
- Independent Variables

The dependent variable is represented by the CLASS attribute, while all other column heads are independent variables (Example - AREA, PERIMETER, etc.)

E. Box Plot

A box plot visually represents data based on its minimum value, first quartile, median, third quartile, and maximum value.

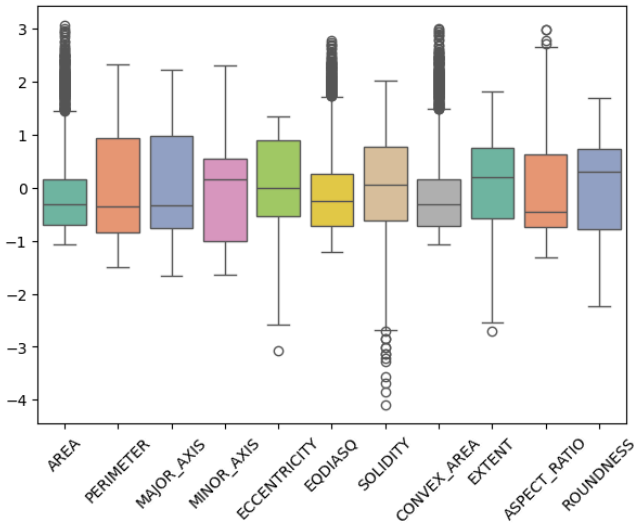


Figure 2. Box Plot

F. Covariance Matrix

The covariance matrix is a matrix used to depict the covariance among corresponding components of an arbitrary vector. It is a square matrix where diagonal elements signify variance, while off-diagonal elements represent covariance. Covariance indicates the degree of direct association between variables.

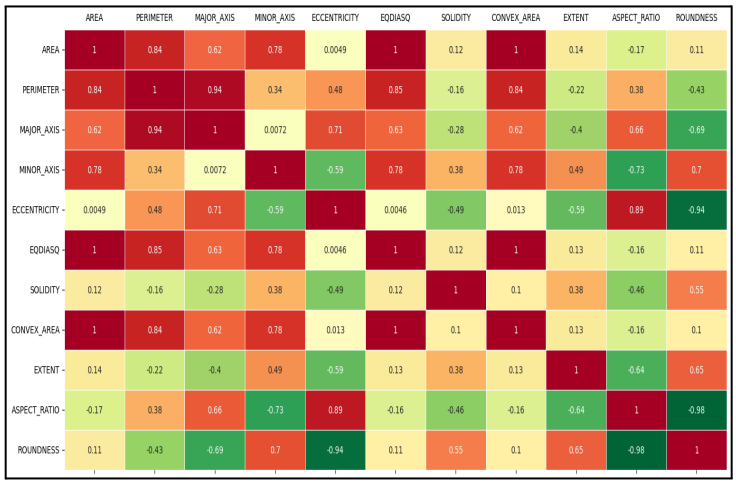
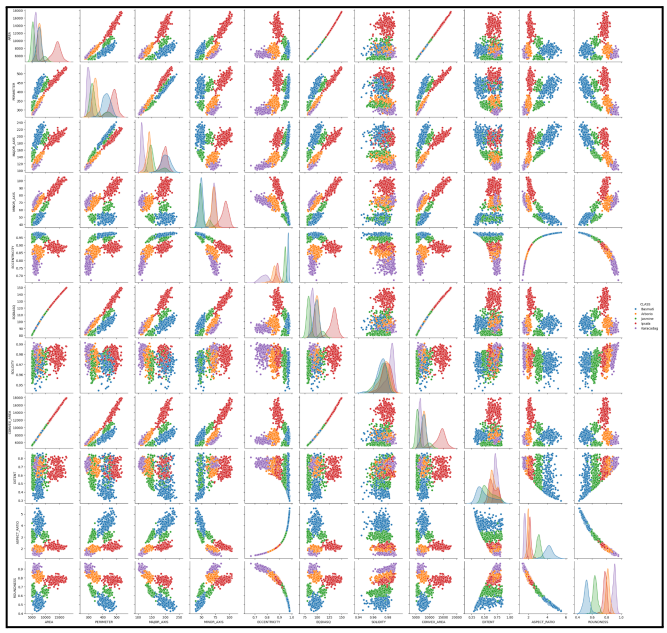


Figure 3. Covariance Matrix

From the above figure, it is clear that AREA is highly correlated with other variables whereas ROUNDNESS shows very little correlation.

G. Pair Plot

A Pair plot involves plotting one variable against another variable from the same dataset. The histogram displayed on the diagonal enables visualization of the distribution of a single variable, while the pair plots presented in the upper and lower triangles illustrate the relationship between two variables. For instance, the pair plot depicted in the leftmost plot of the second row showcases the relationship between Perimeter and Area.



V. PCA Results

Results of Principle component analysis can be analysed under the following sections.

A. Dimensionality Reduction

PCA aims to diminish the data's feature set. This dataset initially comprises 11 attributes/features. However, after implementing PCA, the dataset's attributes have been condensed to 4, as illustrated in the Pareto diagram (Fig 5). The first two components are prioritized for dimensionality reduction, as they account for the majority of the variance.

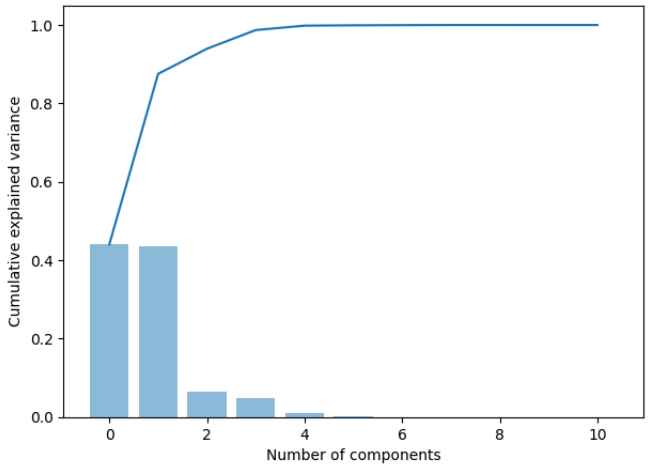


Figure 5. Pareto chart

These observations suggest that the feature set's dimensionality can be reduced to two (r = 2). By employing the eigenvector matrix A, the original n × p dataset undergoes reduction. Each column of the eigenvector matrix A corresponds to a principal component (PC), with each PC encapsulating a certain amount of data defining the dimension (r). The resulting eigenvector matrix (A) for the Rice seed dataset is presented below:

```
0.0852 0.4480 0.0282 −0.0324 −0.1178 0.4134 0.1382 −0.2935 0.0083 −0.0241 −0.7068
0.3054 0.3352 −0.0413 0.0412 0.2330 −0.3397 −0.2500 0.0794 0.7212 0.1651 −0.0708
0.3936 0.2160 −0.1229 0.1044 0.2463 −0.4416 0.1242 −0.2744 −0.3685 −0.5380 0.0276
−0.2041 0.4026 0.1258 −0.1367 −0.0056 0.0995 −0.0974 0.7104 −0.0374 −0.4861 0.0030
0.4210 −0.0818 −0.0827 0.1976 −0.8201 −0.1305 0.1524 0.2090 0.0926 −0.0711 −0.0066
0.0879 0.4481 0.0221 −0.0282 −0.0433 −0.2314 −0.0448 0.2036 −0.5192 0.6477 0.0472
−0.2476 0.1116 −0.9556 −0.0340 −0.0658 0.0294 −0.0795 0.0081 0.0100 −0.0075 0.0139
0.0900 0.4468 0.0482 −0.0321 −0.1124 0.3994 0.1347 −0.2952 0.1376 −0.0279 0.7012
−0.3057 0.1297 0.0589 0.9400 0.0490 0.0131 −0.0099 0.0065 0.0051 0.0000 −0.0004
0.4126 −0.1546 −0.1905 0.1438 0.4165 0.3387 0.5365 0.3921 0.0146 0.1379 0.0038
−0.4299 0.1338 0.0634 −0.1464 −0.0625 −0.4037 0.7456 −0.0388 0.2124 0.0444 −0.0207
```

To calculate the principal components, we can use the following formula

Z = XA

The column values of the above matrix are the principal components :-

Z1 *= 0.08X1 + 0.30X2 + 0.39X3 - 0.20X4 + 0.42X5 + 0.08X6 - 0.24X7 + 0.90X8 - 0.30X9 + 0.41X10 - 0.42X11*

Z2 *= 0.44X1 + 0.33X2 + 0.21X3 + 0.40X4 - 0.08X5 + 0.44X6 + 0.11X7 + 0.44X8 +0.12X9 - 0.15X10 + 0.13X11*

From these values, as depicted in the variance/Pareto plot shown in figure 5, we deduce that the first two components collectively contribute to over 91% of the variance in the rice seed dataset. Consequently, the minimum dimension required to represent our data is d = 2, allowing us to disregard the remaining components for our classification phase, as elaborated in the subsequent section.

Considering these values, as demonstrated in the Pareto plot (Figure 5), we ascertain that the initial two components collectively explain over 90% of the variance within the rice seed dataset. Hence, the minimum dimension required to depict our data is d = 2, allowing us to disregard the subsequent components for our classification phase, elucidated in the subsequent section. Figure 6 illustrates the scatter plot of PC2 Coefficients versus PC1 Coefficients, providing a visual depiction of each feature's contribution to the first two PCs.

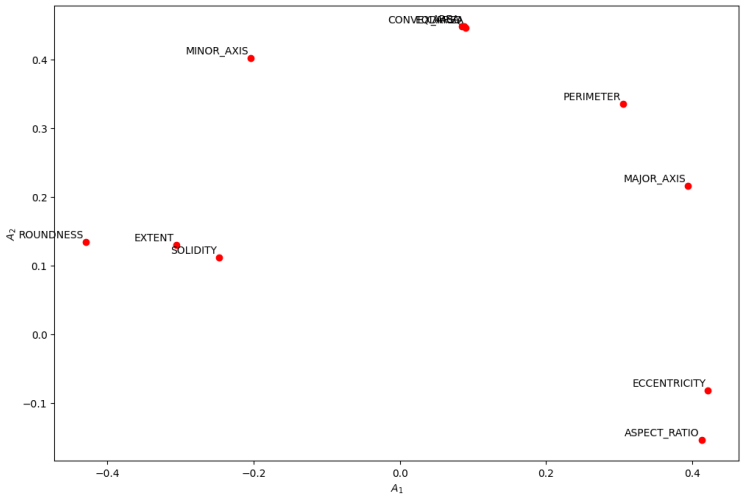


Figure 6. PC coefficient plot

- AREA, CONVEX_AREA, EQDIASQ, PERIMETER, MAJOR_AXIS, ECCENTRICITY and ASPECT_RATIO have positive coefficient for PC1 but MINOR_AXIS, ROUNDNESS, EXTENT AND SOLIDITY have negative coefficient
- Similarly, ECCENTRICITY and ASPECT_RATIO have negative coefficient for PC2 else all have positive coefficient.
- ECCENTRICTIY and ASPECT_RATIO contribute more towards PC1.
- AREA, CONVEX_AREA and EQDIASQ contribute more towards PC2.

B. Scree Plot

A scree plot represents the eigenvalues of factors or principal components in an analysis through a line graph. It serves as a tool for identifying statistically significant factors or components. This plot showcases eigenvalues in a descending curve, with the largest values positioned at the beginning and decreasing towards the end, as depicted in the figure 7. In this illustration, Components 1 and 2 exhibit higher variance, while the graph shows a rapid decrease for Components 4, 5, 6, 7, 8, 9, and 10. Specifically, the first principal component (PC) accounts for 46% of the variance , the second PC accounts for 45% of the variance. These

observations suggest that the feature set's dimensionality can be effectively reduced to two (r = 2), as the cumulative contribution of L1 + L2 amounts to 91%.

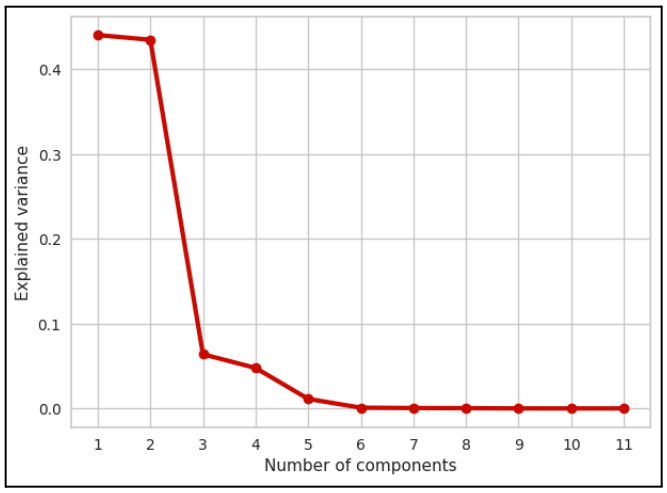


Figure 7. Scree Plot

C. Biplot

The biplot in Figure 8 visualizes principal component coefficients and scores simultaneously. Axes represent principal components and observed variables as vectors. The first principal component displays 4 negative coefficients for MINOR_AXIS, ROUNDNESS, SOLIDITY and EXTENT, and 7 positive coefficients for other features, with corresponding vectors in left and right halves of the plot. The second principal component shows 2 negatuve coefficients for ECCENTRICITY and ASPECT_RATIO, and rest positive coefficients resulting in vectors in top and bottom halves.

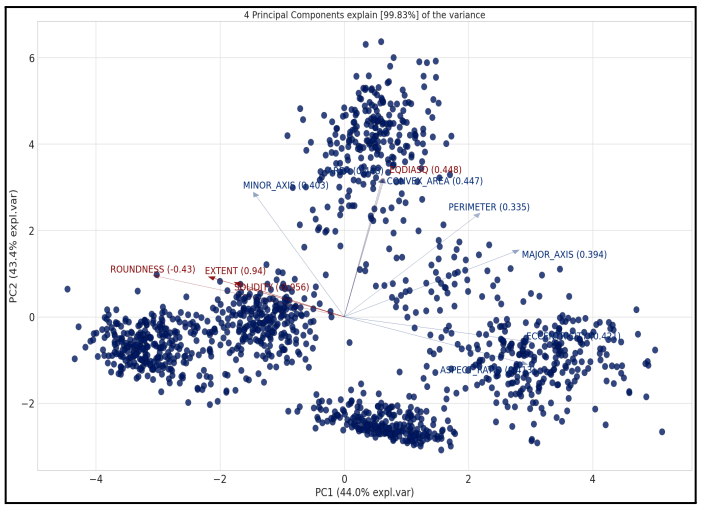


Figure 8. Biplot

VI. Classification Results

This section delves into the performance evaluation of three classification algorithms applied to the Rice seed dataset. To assess the impact of PCA, both the original and PCA-transformed datasets, featuring three PCA components, are subjected to these algorithms. The classification tasks are executed using Python's PyCaret library. The original dataset undergoes a split into training and testing sets, allocated at proportions of 70% and 30%, respectively.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
et	Extra Trees Classifier	0.9762	0.9992	0.9762	0.9780	0.9762	0.9702	0.9707	0.2170
gbc	Gradient Boosting Classifier	0.9750	0.0000	0.9750	0.9771	0.9750	0.9687	0.9693	2.7720
rf	Random Forest Classifier	0.9726	0.9985	0.9726	0.9747	0.9726	0.9658	0.9663	0.7000
qda	Quadratic Discriminant Analysis	0.9726	0.0000	0.9726	0.9746	0.9726	0.9657	0.9663	0.0920
lightgbm	Light Gradient Boosting Machine	0.9714	0.9970	0.9714	0.9733	0.9714	0.9643	0.9648	5.1620
xgboost	Extreme Gradient Boosting	0.9679	0.9969	0.9679	0.9703	0.9678	0.9598	0.9605	0.1670
lda	Linear Discriminant Analysis	0.9619	0.0000	0.9619	0.9652	0.9619	0.9523	0.9532	0.0720
dt	Decision Tree Classifier	0.9595	0.9748	0.9595	0.9618	0.9595	0.9494	0.9500	0.0960
lr	Logistic Regression	0.9583	0.0000	0.9583	0.9613	0.9582	0.9479	0.9487	0.5690
nb	Naive Bayes	0.9488	0.9972	0.9488	0.9523	0.9489	0.9360	0.9368	0.0830
ridge	Ridge Classifier	0.9155	0.0000	0.9155	0.9205	0.9150	0.8943	0.8958	0.0810
knn	K Neighbors Classifier	0.8821	0.9672	0.8821	0.8889	0.8818	0.8526	0.8544	0.1240
ada	Ada Boost Classifier	0.6202	0.0000	0.6202	0.5283	0.5067	0.5221	0.5741	0.4220
svm	SVM - Linear Kernel	0.3286	0.0000	0.3286	0.1789	0.2093	0.1662	0.2151	0.1520
dummy	Dummy Classifier	0.2024	0.5000	0.2024	0.0410	0.0681	0.0000	0.0000	0.0400

Fig 9. Classification model comparison before PCA

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
rf	Random Forest Classifier	0.9750	0.9975	0.9750	0.9767	0.9750	0.9687	0.9691	0.4160
xgboost	Extreme Gradient Boosting	0.9702	0.9974	0.9702	0.9720	0.9702	0.9628	0.9632	0.1450
et	Extra Trees Classifier	0.9679	0.9986	0.9679	0.9697	0.9677	0.9598	0.9603	0.2230
knn	K Neighbors Classifier	0.9655	0.9934	0.9655	0.9669	0.9654	0.9568	0.9572	0.0820
qda	Quadratic Discriminant Analysis	0.9655	0.0000	0.9655	0.9684	0.9655	0.9568	0.9576	0.0560
lightgbm	Light Gradient Boosting Machine	0.9643	0.9976	0.9643	0.9658	0.9642	0.9553	0.9557	2.6450
gbc	Gradient Boosting Classifier	0.9631	0.0000	0.9631	0.9644	0.9631	0.9539	0.9542	1.0350
dt	Decision Tree Classifier	0.9595	0.9748	0.9595	0.9620	0.9595	0.9494	0.9500	0.0620
lr	Logistic Regression	0.9583	0.0000	0.9583	0.9605	0.9583	0.9479	0.9484	0.0700
lda	Linear Discriminant Analysis	0.9571	0.0000	0.9571	0.9588	0.9572	0.9464	0.9468	0.0570
nb	Naive Bayes	0.9476	0.9969	0.9476	0.9510	0.9479	0.9345	0.9352	0.0650
svm	SVM - Linear Kernel	0.8976	0.0000	0.8976	0.9010	0.8914	0.8719	0.8761	0.0710
ridge	Ridge Classifier	0.7750	0.0000	0.7750	0.8317	0.7576	0.7195	0.7380	0.0580
ada	Ada Boost Classifier	0.7071	0.0000	0.7071	0.7117	0.6400	0.6319	0.6706	0.1720
dummy	Dummy Classifier	0.2024	0.5000	0.2024	0.0410	0.0681	0.0000	0.0000	0.0570

Fig 10. Classification model comparison after PCA

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.9881	1.0000	0.9881	0.9888	0.9881	0.9851	0.9853
1	0.9524	0.9909	0.9524	0.9544	0.9524	0.9405	0.9410
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9643	0.9837	0.9643	0.9676	0.9643	0.9554	0.9562
4	0.9762	0.9920	0.9762	0.9775	0.9761	0.9702	0.9706
5	0.9524	0.9993	0.9524	0.9558	0.9524	0.9405	0.9413
6	0.9762	0.9915	0.9762	0.9790	0.9762	0.9702	0.9709
7	0.9643	0.9844	0.9643	0.9650	0.9643	0.9553	0.9555
8	0.9405	0.9807	0.9405	0.9435	0.9407	0.9255	0.9262
9	0.9643	0.9780	0.9643	0.9650	0.9643	0.9553	0.9555
Mean	0.9679	0.9900	0.9679	0.9697	0.9679	0.9598	0.9602
Std	0.0169	0.0077	0.0169	0.0161	0.0168	0.0211	0.0209

Fig 11. Random forest metrics post hyperparameter tuning

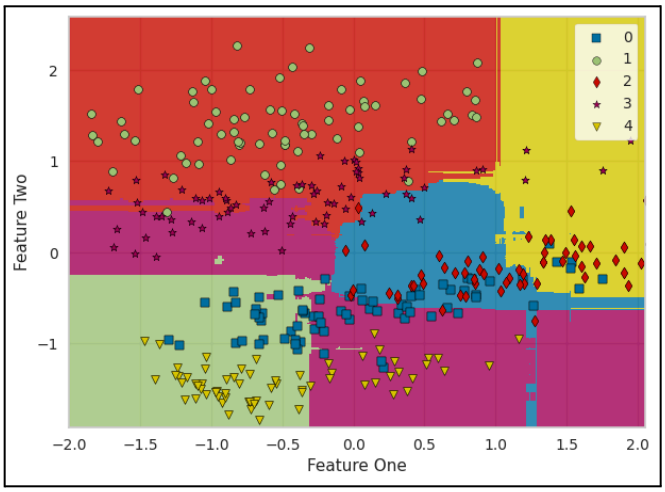
With PyCaret, a performance comparison table was generated to evaluate all available classification algorithms on the target dataset, identifying the best model based on accuracy. As depicted in Figure 9, prior to applying PCA, the top three classification models with the highest accuracies are the Extra Trees Classifier (ET), Gradient Boosting Classifier (GBC), and Random Forest Classifier (RF). As shown in Figure 10, the comparison among classification

models after applying PCA reveals that the top three classifiers are Random Forest Classifier (RF), Extreme Gradient Boosting (XGBOOST), and Extra Trees Classifier (ET). Consequently, these three algorithms are selected for further analysis. Both the original and transformed datasets undergo training, tuning, and evaluation using these three algorithms. While both experiments (classification algorithms applied on the original dataset and transformed dataset) are available in the Google Colab notebook, this report focuses solely on the results obtained after applying PCA (transformed dataset). Hyperparameter tuning plays a crucial role in enhancing model performance. In PyCaret, hyperparameter tuning involves three steps: creating a model, tuning it, and evaluating its performance. Initially, a classification model is generated for each algorithm, followed by using the tune_model() function to fine-tune the model with optimal hyperparameters. Subsequently, effective hyperparameters within a predefined search space for the model are adjusted and assessed using stratified K-fold cross-validation. In this study, 10-fold stratified K-fold validation is applied to the three algorithms.

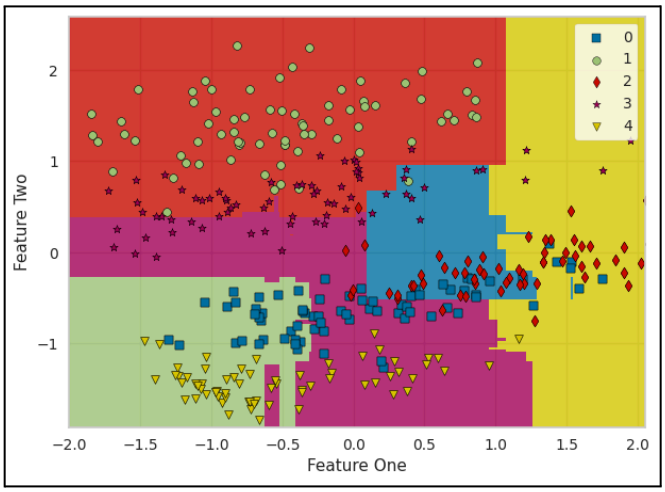
Random Forest Classifier is tuned with various parameters to optimize accuracy and prevent overfitting. One critical parameter is 'n_estimators,' which determines the number of decision trees in the forest ensemble. Increasing the number of estimators can lead to a more robust model but might also increase computational complexity. Additionally, the 'max_depth' parameter controls the maximum depth of each decision tree. A deeper tree can capture more complex patterns in the data but may also result in overfitting, so it's essential to strike a balance. Another parameter to consider is 'min_samples_split,' which determines the minimum number of samples required to split an internal node. Setting a higher value for this parameter can help prevent overfitting by ensuring that splits are only made when enough data is available. Similarly, 'min_samples_leaf' specifies the minimum number of samples required to be at a leaf node. Increasing this parameter can also help regularize the model and prevent overfitting by controlling the size of the leaves. Lastly, 'max_features' controls the number of features to consider when looking for the best split. Limiting the number of features can help reduce overfitting, especially when dealing with high-dimensional data. By carefully tuning these parameters, one can optimize the Random Forest Classifier for both accuracy and generalization performance

The Extreme Gradient Boosting (XGBoost) classifier offers parameters like 'n_estimators' for controlling boosting rounds and 'learning_rate' for step size, crucial for balancing model complexity and convergence speed. Parameters like 'max_depth' control tree complexity to prevent overfitting. Regularization parameters like 'lambda' and 'alpha' penalize large coefficients. Adjusting 'subsample' and 'colsample_bytree' introduces randomness to prevent overfitting. By tuning these, one can optimize XGBoost for accuracy and generalization

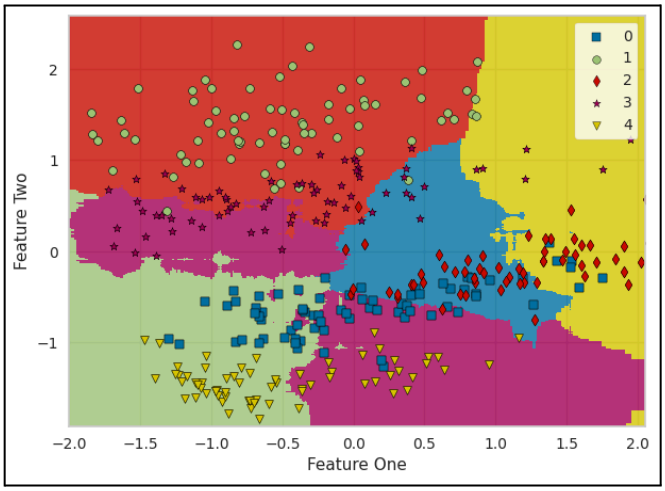
The Extra Trees classifier offers parameters to enhance accuracy and mitigate overfitting. 'n_estimators' sets the number of trees, impacting complexity. 'max_depth' regulates tree depth, affecting overfitting. 'min_samples_split' and 'min_samples_leaf' aid generalization and regularization. 'max_features' limits features for splitting, preventing overfitting in high-dimensional data. Tuning these parameters optimizes Extra Trees for accuracy and generalization



(a) Random Forest Classifier



(b) Extreme Gradient Boosting



(c) Extra Trees Classifier

Figure 12. Decision Boundary of top-3 algorithms

This study evaluates the model's performance using various

metrics such as the Decision boundary, confusion matrix, F1 score, and ROC AUC curve. Figure 12 demonstrates the decision boundaries formed by the model on the transformed dataset. When training a classifier on the rice seed dataset using a specific classification algorithm, a set of hyper-planes, known as Decision Boundaries, segregates data points into distinct classes, signifying transitions between classes. The model can predict values for any combination of inputs within our feature space. In the figures, the x-axis represents the first PC, while the y-axis represents the second PC. All 5 different classes of rice are depicted with unique symbols and colours. The figures showcase the decision boundaries (the lines between regions) of three algorithms. The Extreme Gradient Boosting model's decision boundary appears linear, while Random Forest presents a slightly nonlinear classifier. Similarly Extra Trees Classifier decision surface is complex and non-smooth. Among the three algorithms, Extreme Gradient Boosting performs well, displaying a clear linear separation between all the classes. However, some points from each class may appear in other regions, as it is challenging for a linear model to precisely define the boundary line separating the classes.

A confusion matrix displays the correlation between the actual and predicted classes, providing insights into a model's ability to discern overlapping class properties and identify easily confused classes. Precision and recall values are derived from these matrices. Each row represents instances in a predicted class, while each column denotes instances in an actual class. Figure 13 illustrates the confusion matrix tables for the three algorithms applied to the transformed dataset. Confusion matrices for the original dataset are available in the Google Colab notebook. From Fgire 13, we can say that Random Forest Classifier misclassified the lowers number of instances ans compared to the other two models

Another metric used for performance evaluation is the F1-score. This score combines the precision (positive predicted value) and recall (sensitivity) of a classifier into a single metric by calculating their harmonic mean. The F1-score assists in identifying the superior classifier. Its function can be defined as follows:

$$\textbf{F1 score} = 2 \times \left(\frac{precision \times recall}{precision + recall}\right)$$

From Figure 9 and Figure 10, it's evident that the F1-scores of Random forest classifier and Extreme Gradient Boosting exhibit substantial improvement following PCA application. This observation suggests that dimension reduction diminishes inter-feature dependencies. Furthermore, the F1-scores experienced further enhancement post-tuning with optimal hyperparameters. These findings collectively underscore the advantages of employing PCA and hyperparameter tuning.

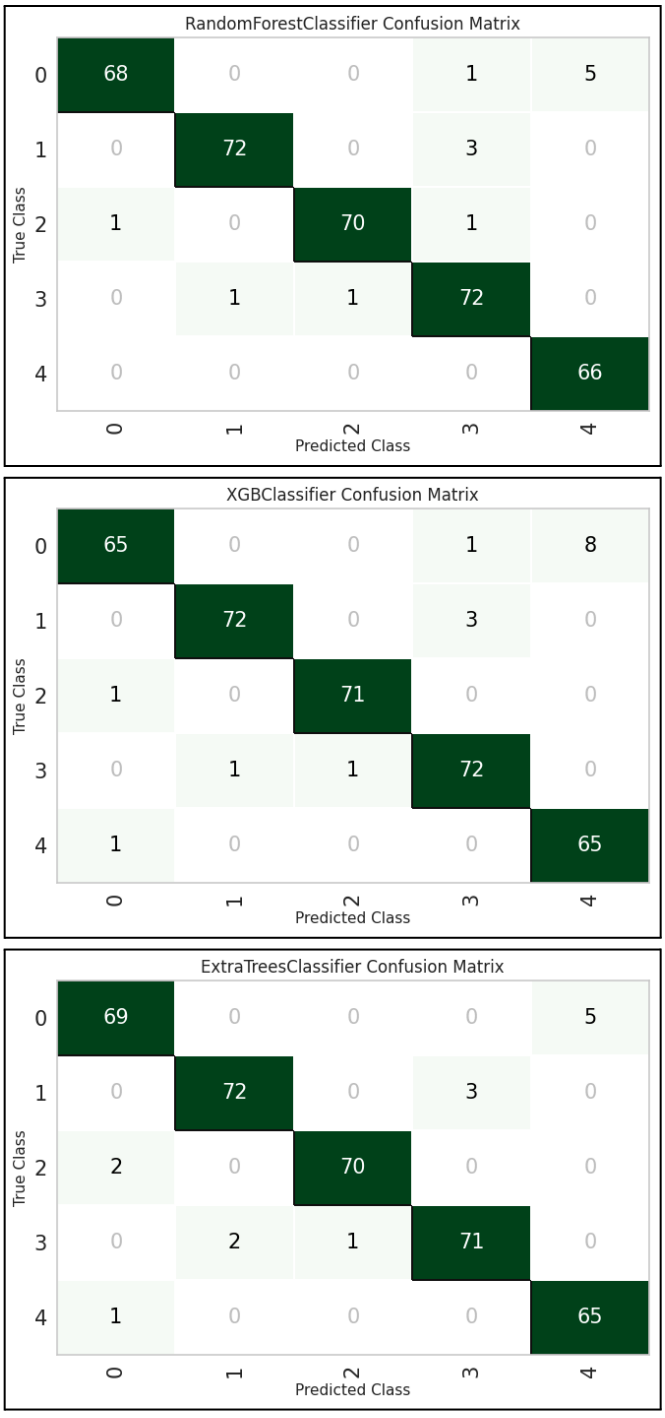


Figure 13. Confusion matrix of top-3 Algorithms

In the final step of the analysis, the receiver operating characteristic (ROC) curve for the LR algorithm is depicted in Figure 14. An ROC curve illustrates a classification model's performance across all classification thresholds. It plots two parameters: the true positive rate (TPR) and the false positive rate (FPR).

$$\text{TRP} = \left(\frac{TP}{TP + FN}\right) \qquad \text{FRP} = \left(\frac{FP}{FP + TN}\right)$$

The AUC (area under the curve) quantifies the total area under the entire ROC curve. Figure 14 illustrates the false positive rate (x-axis) against the true positive rate (y-axis) across various candidate threshold macro and micro average curves.

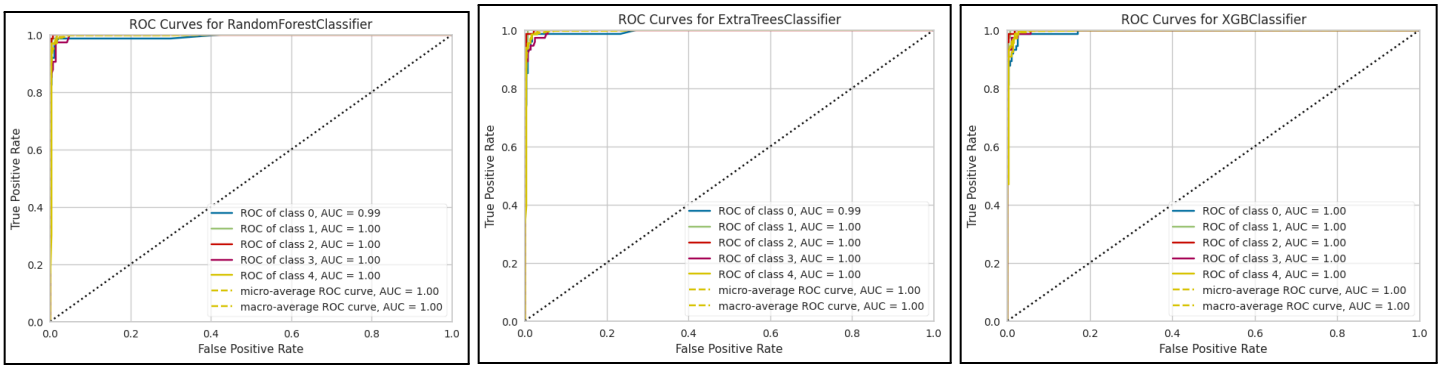


Figure 14 ROC analysis

VII. Conclusion

In conclusion, PCA and three widely-used classification algorithms were employed on the Rice seed dataset, containing morphological feature information for rice variety classification. PCA applied to the original dataset revealed 91% variance captured by the first two principal components reducing the feature set from 11 to 2. Comprehensive experiments were conducted on the first two principal components, generating various plots to validate results from different angles. The impact of PCA on different classifiers was analyzed. Subsequently, the followinfg classification algorithms—LightGBM, K-NN, Naïve Bayes, QDA, Random Forest, Extreme gradient boosting and Extra Trees classifier—were applied to the original and transformed datasets with the the components. Each algorithm was fine-tuned with optimal hyperparameters, and performance evaluation was conducted by comparing confusion matrices, ROC curves, decision boundaries, and F1 scores. The hyperparameter tuning significantly enhanced the performance metrics of each algorithm. While Extra Trees Classifier (ET), Gradient Boosting Classifier (GBC) and Random Forest Classfier (RF) performed best on the original dataset, Random Forest Classfier (RF), Extreme gradient boosting(XGB), and Extra Trees Classifier (ET) exhibited superior performance after PCA application.

REFERENCES

[1]https://www.muratkoklu.com/datasets

[2]https://www.sciencedirect.com/science/article/abs/pii/S0168169921003021

[3]https://www.researchgate.net/publication/373111586_Determining_the_Rice_Seeds_Quality_Using_Convolutional_Neural_Network

[4]https://medium.com/geekculture/ca-pca-implementation-in-python-151fe466a4b4

[5]https://www.geeksforgeeks.org/multiclass-classification-using-scikit-learn

[6]https://www.analyticsvidhya.com/blog/2021/09/pca-and-its-underlying-mathematical-principles

[7]https://github.com/pycaret/pycaret/tree/master/tutorials