Principal Component Analysis and Study of Two Varieties of Raisins Using Machine Learning

Shreya Dixit, 40232504

GitHub Link: https://github.com/shreyadixit08/INSE6220

***Abstract*—A popular dimensionality reduction method in machine learning and statistics is Principal Component Analysis, or PCA. By locating the principal components—linear combinations of the original features—it seeks to convert high-dimensional correlated data into a lower-dimensional representation. The highest variation in the data is captured by these uncorrelated variables, enabling a more condensed depiction without sacrificing the important patterns. This study employs PCA on Raisins dataset to discern between two varieties of raisins: Kecimen and Besni. Extreme Gradient Boosting (XGBoost), Logistic Regression (LR), Gradient Boosting Classifier (GBC) and K-Nearest Neighbour (K-NN) classification algorithms are implemented. Hyperparameter tuning enhances each model's performance and XGBoost exhibits superior performance among PyCaret library's machine learning models. The assessment of the optimal model with principal component (PC) components reveals that the Extreme Gradient Boosting (XGBoost), which is a combination of multiple decision trees, is the most effective.**

***Index Terms*—Principal Component Analysis (PCA), Extreme Gradient Boosting (XGBoost), Logistic Regression (LR), Gradient Boosting Classifier (GBC), K-Nearest Neighbour (K-NN), Multi-class Classification.**

I. INTRODUCTION

Raisins are dried grapes, created through a natural drying process that involves sun exposure or artificial drying methods like oven drying. This process concentrates sugars and flavors, resulting in a sweet, shriveled fruit. Two notable varieties of raisins are Kecimen and Besni, both originating from Turkey. These raisins likely derive their names from specific regions or grape varieties. The drying methods for these raisins may include traditional sun exposure, contributing to their unique characteristics. Both Kecimen and Besni raisins may exhibit distinctive qualities based on the grape varieties used, the local climate, and the specific drying methods employed. Raisins, in general, are known for their natural sweetness and are used in various culinary applications, such as baking, cooking, or as a standalone snack.

This report commences with the application of Principal Component Analysis (PCA) to the raisin dataset for the purpose of dimensionality reduction. Subsequently, well-known classification algorithms are employed on both the original and PCA-transformed datasets to distinguish between Kecimen and Besni raisins. The report also incorporates the utilization of Shapley values, for interpreting classification models. Results from the original dataset can be referenced in the Google Colab notebook.

The structure of the remaining report is outlined as follows: Section II details the PCA methodology, Section III provides an overview of the four classification algorithms, Section IV describes the raisins dataset, Section V discusses PCA results, Section VI offers an extensive analysis of classification results, Section VII deliberates on the application of explainable AI Shapley values, and finally, Section VIII concludes the report.

II. PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique widely employed in statistics, machine learning, and data analysis. The initial step involves centering the data by subtracting the mean of each variable, ensuring that the data is centered around the origin. Subsequently, the covariance matrix is computed from the centered data,

revealing the relationships between different variables. The process continues with the calculation of eigenvectors and eigenvalues from the covariance matrix. Eigenvectors are then sorted based on their corresponding eigenvalues, with the highest eigenvalue indicating the principal component with the most significant variance. Depending on the desired level of dimensionality reduction, a specific number of top-ranked eigenvectors are selected, forming a new basis for the data. The original data is transformed by projecting it onto the chosen principal components, resulting in a lower-dimensional representation.

PCA facilitates data visualization in a lower-dimensional space, easing the interpretation of complex patterns. It is effective in reducing collinearity in correlated datasets and serves as a preprocessing step to enhance the efficiency of subsequent machine learning algorithms. PCA's role extends to exploratory data analysis, helping uncover hidden patterns and relationships within the data, contributing to a deeper understanding of the dataset.

A. PCA Algorithm

PCA can be applied to a data matrix X with dimension n × p using the followings steps:

1) Standardization/Normalization:
Before applying PCA, it is common practice to standardize or normalize the data to ensure that all variables are on the same scale. Initially, the mean vector $\overline{x}$ for each column in the dataset is computed. This *p*-dimensional vector is expressed as:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Standardization involves subtracting the mean of each column from the respective entries in the data matrix, resulting in the final centered data matrix *Y* expressed as

$$Y = HX$$

where *H* denotes the centering matrix.

2) Computation of Covariance Matrix
This step aims to assess the relationships among variables, identifying potential redundancies. The covariance matrix *S*, sized, p × p is computed as:

$$S = \frac{1}{N-1} Y'Y$$

3) Eigen Decomposition
Utilizing eigen decomposition, the eigenvalues and eigenvectors of S are determined. Eigenvectors indicate the direction of each principal component (PC), while eigenvalues signify the variance captured by each PC. The eigen decomposition equation is:

$$S = A \wedge A' = \sum_{j=1}^{p} \lambda_j \, a_j a'_j$$

where A is the p × p orthogonal matrix of eigenvectors, and ∧ is the diagonal matrix of eigenvalues.

4) Transformation of data
This step computes the transformed matrix Z with dimensions n × p, where rows represent observations and columns represent principal components (PCs). The number of PCs equals the dimension of the original data matrix. The equation for Z is:

$$Z = YA$$

III. CLASSIFICATION ALGORITHMS BASED ON MACHINE LEARNING

Classification Algorithms refer to a set of supervised learning techniques used to categorize data points into predefined classes or labels. These algorithms learn patterns and relationships from labeled training data and apply this knowledge to classify new, unseen data-points. Here are brief explanations of four classification algorithms.

A. Extreme Gradient Boosting (XGBoost)

XGBoost, or Extreme Gradient Boosting, is a well-known, quick, and precise machine learning method. It falls under the category of gradient boosting techniques, which combine the predictions of weak learners—typically decision trees—in order to create powerful predictive models. XGBoost enhances and bolsters the dependable and scalable nature of traditional gradient boosting through the use of parallel processing and regularisation techniques. In order to minimise a loss function, the algorithm optimises the total of

the predictions plus a regularisation term. The function:

$$\mathrm{L}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

includes a term for the loss on predictions $l(y_i, \hat{y}_i)$ and a regularization term $\Omega(f_k)$ to control the complexity of the individual trees. The model is trained through gradient descent, updating the weights of the weak learners to minimize the overall objective function.

B. Logistic Regression (LR)

Logistic regression is a machine learning statistical method employed for binary classification tasks. Despite its name, it functions as a classification algorithm rather than a regression one. The logistic regression model utilizes the logistic function, also known as the sigmoid function, to express the probability of an instance belonging to a particular class. The logistic function is defined as follows:

$$P(Y = 1) = \frac{1}{1 + e^{-(b_0 + b_1 X)}}$$

In this equation, *X* represents the input feature, b_0 and b_1 denote the coefficients, and *P(Y=1)* represents the probability of the positive class. During the training phase, logistic regression aims to optimize the likelihood of the observed data by estimating these coefficients. Instances with a probability equal to or greater than 0.5 are predicted to belong to the positive class, while those with a probability less than 0.5 are predicted to belong to the negative class. The decision boundary is established at 0.5.

C. Gradient Boosting Classifier (GBC)

The Gradient Boosting Classifier is an ensemble learning technique that builds a powerful predictive model by aggregating the predictions of several weak learners, usually decision trees. It constructs trees one after the other, fixing the mistakes of the earlier trees as it goes. By changing the weights given to instances that are incorrectly classified, the technique minimises a loss function. A weighted total of the forecasts made by each individual tree makes up the final forecast. In terms of mathematics, the Gradient Boosting Classifier's prediction (for binary classification) looks like this:

$$F(x) = \sum_{i=1}^{N} \alpha_i f_i(x)$$

Here, *F(x)* is the final prediction, α_i is the weight associated with the *ith* weak learner, and $f_i(x)$ is the prediction of the *ith* weak learner. The algorithm seeks to find the optimal weights and weak learners that minimize the overall loss.

D. K-Nearest Neighbour (K-NN)

A supervised machine learning approach called K-Nearest Neighbours (K-NN) is utilised for both regression and classification problems. K-NN classifies data points by taking into account the majority class of its k-nearest neighbours in the feature space. The approach is predicated on the idea that values for comparable cases in the input space will be similar. In a multidimensional space, data points are represented as vectors, and the proximity between points is calculated using a distance metric, like the Euclidean distance. The number of neighbours to take into account is determined by the parameter k, which affects how sensitive the algorithm is to local patterns. Although K-NN is simple and efficient for nonlinear, complicated decision boundaries, it can be computationally demanding, and the selection of the distance metric and k plays a crucial role in its performance.

IV. DATASET DESCRIPTION

This project utilizes a dataset comprising two distinct varieties of raisin grains derived from grapes cultivated in Turkey. The dataset incorporates images of these raisins, and image-processing techniques were applied to extract a total of seven characteristic features for each raisin type. The dataset encompasses a total of 900 grains for each of the raisin types, Kecimen and Besni. The distribution of data observations for both types is visually represented in

the pie chart presented in Fig. 1, where an equal number of observations for each raisin type is illustrated.

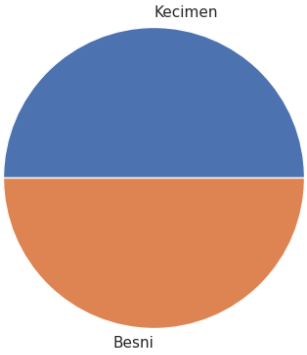

Fig. 1 Pie Chart of raisin varieties

A box plot, also known as a box and whisker plot, is a visual representation that succinctly summarizes the distribution of a dataset. The rectangular "box" in the plot represents the interquartile range (IQR), encompassing the middle 50% of the data. Inside the box, a horizontal line signifies the dataset's median, or 50th percentile. The "whiskers" extending beyond the box depict the range of the data and usually reach a specific multiple of the IQR. Optionally, individual points beyond the whiskers can be plotted to identify potential outliers. Box and whisker plots provide a clear visual overview of central tendency, distribution, and any outliers in a dataset, facilitating easy comparison between datasets. These plots can be oriented either horizontally or vertically, depending on the preference for presentation and clarity of interpretation. This plot can be seen in Fig. 2 given below.

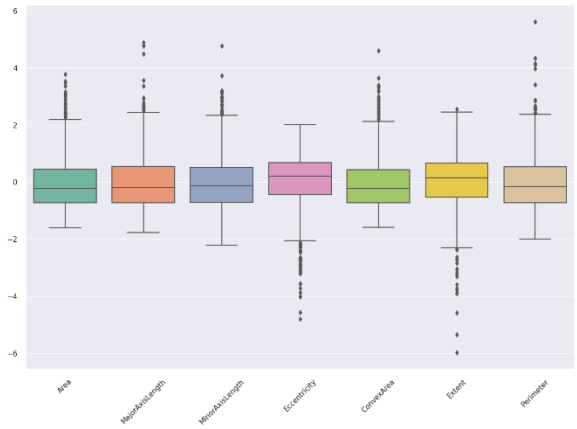

Fig. 2 Box Plot

Swarm plots, also known as strip plots, are a type of data visualization used to display individual data points along a single axis. Unlike traditional scatter plots, swarm plots prevent overlap by spreading points horizontally, providing a clearer representation of the distribution and density of data. Each data point is shown as a marker, offering insights into the individual values within a dataset. This plot can be seen in Fig. 3 given below.

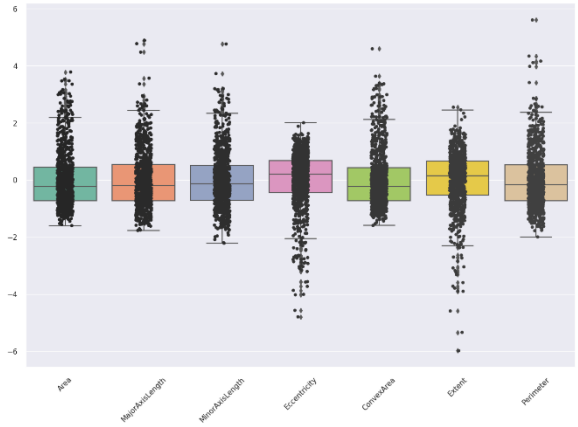

Fig. 3 Swarm or Strip Plot

A mathematical tool that sheds light on the connections between several variables in a dataset is the covariance matrix. It methodically arranges this data into a symmetric matrix by combining variances and covariances. The variances of the separate variables are represented by the diagonal members of the matrix, which show how variable each variable is. The covariances between pairs of variables are indicated by off-diagonal elements, which show how much these variables vary in tandem. A relationship is implied to be inverse when there is a negative covariance, and direct when there is a positive covariance. The area has a positive correlation with the major and minor axis lengths, convex area, and perimeter, and is exhibited in Fig. 4. Likewise, there exists a positive correlation between perimeter and major axis length. On the other hand, area and extent are negatively correlated.

Fig. 4 Covariance Matrix

In data visualization, a pair plot, sometimes referred to as a scatterplot matrix, is a graphical technique used to examine correlations between pairs of variables in a multivariate dataset. The plot is made up of a grid of scatterplots, with a different subplot representing each set of two variables. Histograms or kernel density plots show the univariate distribution of each variable along the diagonal of the grid. This plot is shown in Fig. 5, which is provided below.

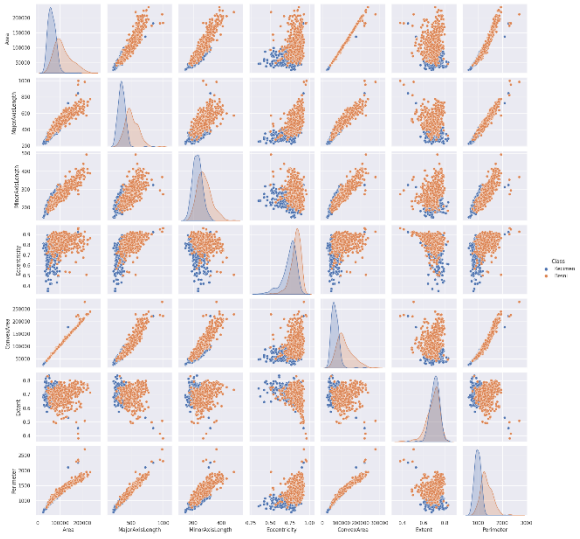

Fig 5 Pair Plot

V. PCA RESULTS

PCA is utilized on the raisin's dataset, and its implementation can be approached in two ways:

1) constructing PCA from scratch using standard Python libraries like NumPy.

2) Employing a well-established PCA library.

Both methods are demonstrated in the Google Colab notebook. Although the results from both approaches are comparable, utilizing the PCA library provides enhanced flexibility to users with the simplicity of a single line of code. The figures and plots presented in this report are generated using the PCA library. Through the application of PCA steps, the original feature set of 7 can be reduced to a subset of r features, where r is less than 7. It is reduced to 3 variables (PCs) in this case. The n × p dataset is reduced using the eigenvector matrix A. Each column of A represents a principal component PC capturing a distinct amount of data that determines the dimension r. The resulting eigenvector matrix A for the raisins dataset is detailed below.

$$A=\begin{bmatrix} 0.448 & -0.116 & 0.005 & -0.111 & -0.611 & -0.099 & -0.624 \\ 0.443 & 0.136 & -0.101 & 0.495 & 0.087 & -0.685 & 0.227 \\ 0.389 & -0.375 & 0.236 & -0.656 & 0.384 & -0.239 & 0.129 \\ 0.203 & 0.611 & -0.628 & -0.426 & 0.075 & 0.053 & 0.020 \\ 0.451 & -0.087 & 0.036 & 0.056 & -0.392 & 0.471 & 0.639 \\ -0.056 & -0.667 & -0.732 & 0.109 & 0.057 & 0.023 & -0.002 \\ 0.451 & 0.034 & 0.044 & 0.339 & 0.555 & 0.487 & -0.364 \end{bmatrix}$$

and the eigenvalue matrix is:

$$\lambda = \begin{bmatrix} 4.837 \\ 1.455 \\ 0.629 \\ 0.057 \\ 0.022 \\ 0.006 \\ 0.001 \end{bmatrix}$$

Eigenvalues quantify the variability in data captured by each principal component. The analysis of the variance recorded by principal components is facilitated by examining the Scree plot and Pareto plot, presented in Fig. 6 and Fig. 7 respectively. The explained variance represents the percentage of variance attributed to the jth pc, calculated using the formula:

$$j = \frac{\lambda_j}{\sum_j^p \lambda_j} \times 100\,, j = 1,2,\dots,p$$

where λ_j is the eigenvalue and amount of variance of the jth principal component. The Scree plot indicates that the first two principal components collectively account for approximately 89% of the total variance in the original data. Additionally, the Pareto plot in Fig. 7 illustrates that the first three principal components contribute to 99.6% of the total variance.

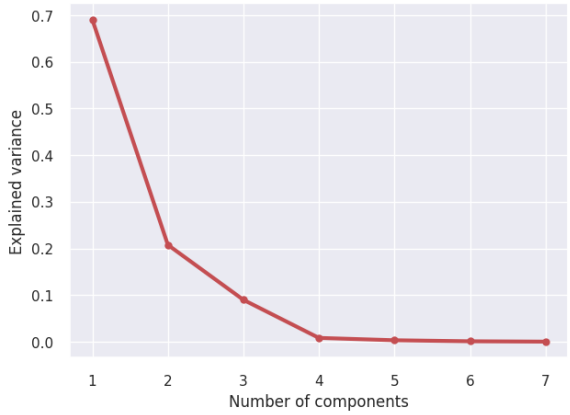

Fig. 6 Scree Plot

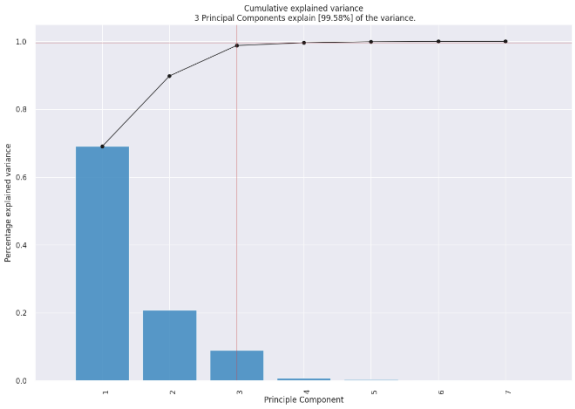

Fig. 7 Pareto Plot

These findings suggest that the dimension of the feature set can be effectively reduced to three (r = 3). The formulation for the first principal component, Z1, is as follows:

$$Z_1 = 0.448X_1 + 0.443X_2 + 0.389X_3 + 0.203X_4 + 0.451X_5 - 0.056X_6 + 0.451X_7$$

From the initial principal component, it is evident that X1, X5 and X7 make the most significant contributions. Nevertheless, it's important to note that none of the features have an insignificant impact on the first principal component. Moving to the second principal component Z2, the formulation is as follows:

$$Z_2 = -0.116X_1 + 0.136X_2 - 0.375X_3 + 0.611X_4 - 0.087X_5 - 0.667X_6 + 0.034X_7$$

Similarly computing the third principal component Z3:

$$Z_3 = -0.005X_1 - 0.101X_2 + 0.236X_3 - 0.628X_4 + 0.036X_5 - 0.732X_6 + 0.044X_7$$

The below Fig. 8 represents a PC coefficient plot which is a graphical representation that displays the coefficients of each variable in the principal components derived from a dimensionality reduction technique, such as Principal Component Analysis (PCA). Each variable is represented by a vector, and the length and direction of the vector convey the strength and direction of the variable's contribution to the corresponding principal component.

Fig. 8 Coefficient Plot

Now, let's look at the biplot given in Fig. 9, which is a graphical representation in multivariate statistics that combines information from both the observations and the variables in a dataset. In a biplot, data points are plotted in a way that preserves the distances and angles between them, while the variables are represented as vectors. The direction and length of the vectors convey information about the relationships and strengths of the variables. The Kecimen raisins are represented by blue dots, whereas Besni raisins are represented by green dots.

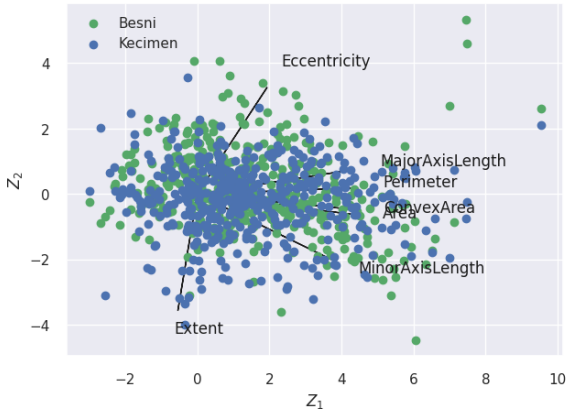

Fig. 9 Biplot

VI. CLASSIFICATION RESULTS

The performance of three popular classification techniques on the Raisins dataset is covered in this section. The original dataset and the dataset that PCA turned into three components are both subjected to the classification algorithms in order to evaluate the effect of PCA on the dataset. PyCaret is a Python module that automates numerous operations related to model creation and deployment, hence streamlining the machine learning workflow. For operations including feature engineering, data preprocessing, model selection, hyperparameter tuning, and model deployment, it offers an intuitive user interface.

Prior to PCA, the three best classification models on the raisins dataset, as shown in Fig. 10, were Ridge Classifier (RIDGE), Linear Discriminant Analysis (LDA), and Logistic Regression (LR).

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.8781	0.0000	0.8781	0.8826	0.8776	0.7560	0.7605	0.0630
lda	Linear Discriminant Analysis	0.8781	0.9364	0.8781	0.8818	0.8777	0.7560	0.7598	0.0230
lr	Logistic Regression	0.8693	0.9369	0.8693	0.8734	0.8688	0.7384	0.7426	0.3660
rf	Random Forest Classifier	0.8675	0.9204	0.8675	0.8734	0.8668	0.7347	0.7407	0.2170
qda	Quadratic Discriminant Analysis	0.8623	0.9293	0.8623	0.8762	0.8605	0.7239	0.7379	0.0320
lightgbm	Light Gradient Boosting Machine	0.8604	0.9131	0.8604	0.8627	0.8600	0.7206	0.7229	0.3150
xgboost	Extreme Gradient Boosting	0.8587	0.9129	0.8587	0.8638	0.8579	0.7170	0.7222	0.0610
gbc	Gradient Boosting Classifier	0.8534	0.9154	0.8534	0.8587	0.8526	0.7064	0.7118	0.1500
et	Extra Trees Classifier	0.8534	0.9227	0.8534	0.8566	0.8529	0.7065	0.7098	0.1280
ada	Ada Boost Classifier	0.8499	0.9081	0.8499	0.8585	0.8485	0.6994	0.7080	0.1330
nb	Naive Bayes	0.8340	0.9135	0.8340	0.8489	0.8312	0.6672	0.6819	0.0240
knn	K Neighbors Classifier	0.8234	0.8807	0.8234	0.8314	0.8219	0.6464	0.6544	0.0380
dt	Decision Tree Classifier	0.8216	0.8212	0.8216	0.8249	0.8210	0.6430	0.6463	0.0250
svm	SVM - Linear Kernel	0.5523	0.0000	0.5523	0.5097	0.4243	0.1097	0.1565	0.0230
dummy	Dummy Classifier	0.5044	0.5000	0.5044	0.2544	0.3382	0.0000	0.0000	0.0200

Fig.10 Classification models comparison before PCA

On the other hand, the comparison following PCA is displayed in Fig. 11, where it is evident that the top three models providing the highest accuracy on the transformed dataset are Gradient Boosting Classifier (GBC), Logistic Regression (LR), and Extreme Gradient Boosting (XGBoost). As a result, these three algorithms are selected to be evaluated in the remaining trials. These methods are used to train, fine-tune, and assess the altered datasets. Thirty percent of the sultana dataset is used for testing, while the remaining seventy percent is used for training. We've looked at how Principal Component Analysis (PCA) affects the aforementioned techniques and evaluated the model performance according to several standards.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
xgboost	Extreme Gradient Boosting	0.8694	0.9152	0.8694	0.8729	0.8690	0.7385	0.7421	0.1560
lr	Logistic Regression	0.8693	0.9277	0.8693	0.8724	0.8690	0.7384	0.7416	0.0560
gbc	Gradient Boosting Classifier	0.8657	0.9310	0.8657	0.8737	0.8647	0.7309	0.7391	0.2080
ridge	Ridge Classifier	0.8640	0.0000	0.8640	0.8744	0.8628	0.7274	0.7379	0.0640
rf	Random Forest Classifier	0.8640	0.9243	0.8640	0.8701	0.8633	0.7277	0.7339	0.4140
lda	Linear Discriminant Analysis	0.8640	0.9283	0.8640	0.8744	0.8628	0.7274	0.7379	0.0640
lightgbm	Light Gradient Boosting Machine	0.8622	0.9196	0.8622	0.8667	0.8616	0.7241	0.7287	0.3100
svm	SVM - Linear Kernel	0.8587	0.0000	0.8587	0.8659	0.8580	0.7173	0.7245	0.0620
et	Extra Trees Classifier	0.8587	0.9250	0.8587	0.8636	0.8580	0.7171	0.7221	0.2800
ada	Ada Boost Classifier	0.8534	0.9051	0.8534	0.8584	0.8524	0.7063	0.7115	0.2000
knn	K Neighbors Classifier	0.8499	0.9103	0.8499	0.8541	0.8492	0.6994	0.7037	0.0760
nb	Naive Bayes	0.8481	0.9215	0.8481	0.8624	0.8461	0.6955	0.7098	0.0560
qda	Quadratic Discriminant Analysis	0.8446	0.9078	0.8446	0.8563	0.8431	0.6885	0.7003	0.0560
dt	Decision Tree Classifier	0.8411	0.8409	0.8411	0.8449	0.8405	0.6820	0.6858	0.0450
dummy	Dummy Classifier	0.5044	0.5000	0.5044	0.2544	0.3382	0.0000	0.0000	0.0440

Fig. 11 Classification models comparison after PCA

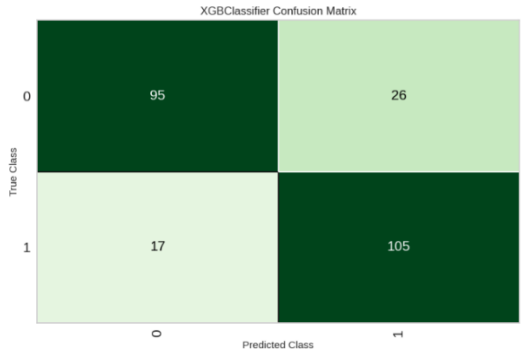

Fig. 12 Confusion Matrix of Best Model - Extreme Gradient Boosting (XGBoost)

In Fig. 12 we can see the best model offering the highest accuracy is Extreme Gradient Boosting (XGBoost) which falls under the category of gradient boosting algorithms, which sequentially combine the predictions of weak learners, typically decision trees, to create a strong predictive model. The algorithm minimizes a loss function by optimizing the sum of predictions and a regularization term. The matrix clearly shows that Besni has been accurately identified as Besni 95 times and mistakenly classified as

Kecimen 26 times. Likewise, Kecimen has been correctly classified 105 times and incorrectly 17 times.

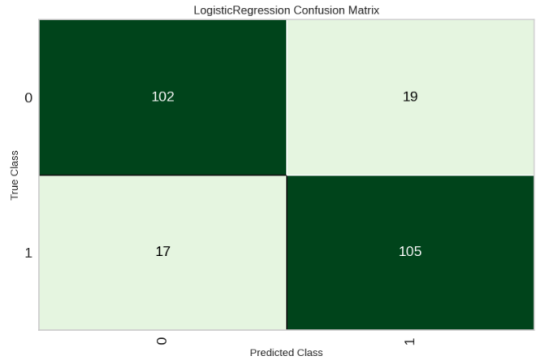

Fig. 13 Confusion Matrix of Logistic Regression (LR)

Fig. 13 shows the second-best model with high accuracy which is Logistic Regression (LR). It can be seen through the matrix that Besni is correctly identified as Besni 99 times and mistakenly classified as Kecimen 22 times. Similarly, Kecimen has been accurately classified 110 times and incorrectly 12 times.

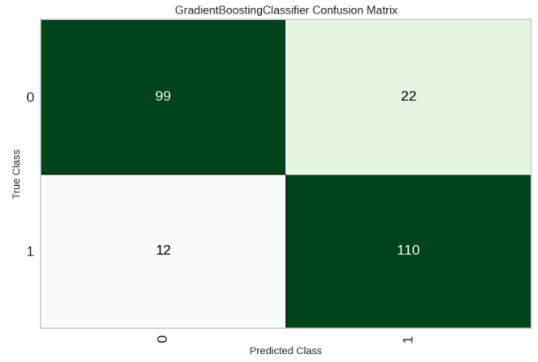

Fig. 14 Confusion Matrix of Gradient Boosting Classifier (GBC)

Gradient Boosting Classifier (GBC), a technique that operates sequentially and trains each new tree to rectify the mistakes of the aggregated ensemble of previous trees, is the third-best model, as shown in Fig. 14. By using gradient descent to minimize a loss function, it modifies the weights of the weak learners. The total of all the trees' projections is used to get the final forecast.

VII. EXPLAINABLE AI WITH SHAPLEY VALUES

It involves using Shapley values, a concept from cooperative game theory, to interpret and explain the predictions of machine learning models. The process involves evaluating the model's output for all possible combinations of features and determining the marginal contribution of each feature to the prediction. Shapley values consider all possible coalitions of features and calculate their average marginal contributions, ensuring a fair distribution of credit. Various algorithms, such as the Shapley Additive Explanations (SHAP) library, provide efficient implementations for computing Shapley values. Once computed, Shapley values offer insights into the importance and impact of individual features, contributing to the overall interpretability of the machine learning model.

It uses optimal Shapley values based on game theory ideas to explain individual forecasts. Taking a cue from game theory, every characteristic of a dataset acts as a coalition member. A player can represent a collection of feature values or a single feature value. Shapley values offer a way to divide the prediction among the feature sets equitably. Tree-based models including decision trees, random forests, and additional tree classifiers are now supported by the Python Shap library.

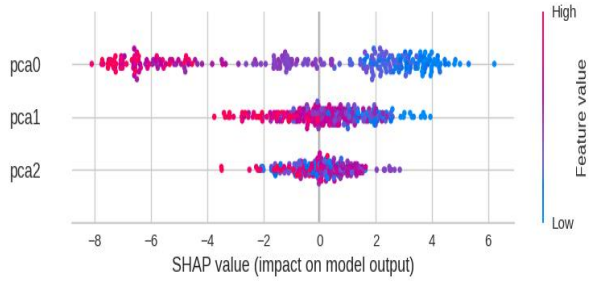

Fig. 15 SHAP Summary Plot

In this analysis, Extreme Gradient Boosting (XGBoost) which is the best model, in conjunction with principal components, is employed to differentiate between two varieties of raisins, Kecimen and Besni. The summary plot given in Fig. 15 illustrates the arrangement of feature values based on the sum of Shap values. The x-axis displays the Shapley values, while the y-axis corresponds to the

Principal Components (PCs). To elaborate, the first PC is denoted as pca_0, the second PC as pca_1, and the third PC as pca_2.

Fig. 16 shows the force plot for the single observation. This plot reveals the contribution of each feature in pushing the model output from the base value, which signifies the predicted value in the absence of any known features, essentially the mean prediction of the test set base value. The bold 0.14 in the plot represents the model's score for this observation, where higher scores lead to a prediction of 1, and lower scores lead to a prediction of 0. The red color indicates that the second PC influences a higher prediction, while the blue color on the first PC indicates a lower prediction.

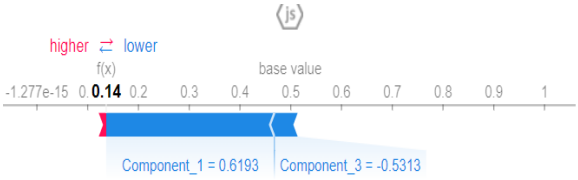

Fig. 16 Force Plot for Single Observation

In Fig. 17, a combined force plot of all Principal Components (PCs) is presented. This plot integrates individual force plots with a 90-degree rotation, stacking them horizontally. The y-axis corresponds to the x-axis of the individual force plots, with 240 data points in the transformed test set. The combined force plot illustrates the influence of each PC on the current prediction. Values in blue are considered to have a positive influence on the prediction, while red values have a negative influence.

Fig. 17 Combined Force Plot

VIII. CONCLUSION

In summary, Principal Component Analysis (PCA) was applied to capture over 95% of the recorded variations using three principal components, and the feature set was reduced to 3 from 7. Among various machine learning algorithms, the Ridge Classifier followed by Logistic Regression (LR) emerged as the top-performing model for the data. However, when testing with three principal components, Extreme Gradient Boosting (XGBoost) demonstrated superior performance on the transformed data. The analysis successfully distinguished between raisin types, showcasing the effectiveness of PCA and various machine learning algorithms in the study.

REFERENCES

[1] CINAR I., KOKLU M. and TASDEMIR S., (2020). Classification of Raisin Grains Using Machine Vision and Artificial Intelligence Methods, Gazi Journal of Engineering Sciences, vol. 6, no. 3, pp. 200-209, December, 2020, DOI: 10.30855/gmbd.2020.03.03

[2] https://chat.openai.com/

[3] https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[4] https://www.jcchouinard.com/pca-with-python/

[5] https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/

[6] https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

[7] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

[8] https://www.geeksforgeeks.org/k-nearest-neighbours/

[9] https://shap.readthedocs.io/en/latest/example_notebooks/overviews/An%20introduction%20to%20explainable%20AI%20with%20Shapley%20values.html

[10] https://math.libretexts.org/Bookshelves/Linear_Algebra/A_First_Course_in_Linear_Algebra_(Kuttler)/07%3A_Spectral_Theory/7.01%3A_Eigenvalues_and_Eigenvectors_of_a_Matrix

[11] https://www.kaggle.com/datasets/muratkokludataset/raisin-dataset