Principal Component Analysis on Raisins Segregation Using Machine Learning

Anusha Dsouza - 40240817
GitHub Link: https://github.com/anushadsouza19/INSE-6220

Abstract*— A statistical technique called principal component analysis, or PCA, enables you to more quickly and simply visualise and analyse a smaller number of "summary indices" that represent the information content of massive data tables. It is utilised in many different applications, including data compression, data analysis, and many more. PCA analysis aids in the reduction or elimination of comparable data that does not even slightly aid in decision-making in the line of comparison. It is a popular unsupervised learning method for reducing the dimensionality of huge datasets. It also enables effective 2D and 3D visualisation of high-dimensional data by identifying the ideal linear combinations of variables. In this work, we use PCA to divide raisins into different categories - Kecimen and Besni. Three machine learning models - Logistic Regression, Ridge Classifier and Linear Descriminant Analysis whose performance has been optimised via hyper-parameter tuning are the subject of our evaluation. We evaluate the models using a variety of performance metrics, including accuracy, precision, recall, and F1-score, and then use confusion matrices and ROC curves to display the findings. The most crucial characteristics in the dataset are also identified using SHAP values. Our results show that the Logistic Regression works better than the other models and can successfully divide the raisins into the different categories, attaining an F1-score of 0.93. This work emphasises the significance of model interpretability in the context of machine learning as well as the value of PCA as a tool for data visualisation.

Keywords: Principal Component Analysis, SHAP, Extra Tree Classifier, Random Forest Classifier and Light Gradient Boosting Machine, Logistic Regression, Ridge Classifier and Linear Descriminant Analysis.

I. INTRODUCTION

Raisins are a high-carbohydrate consumables that also encompass dietary fibre, antioxidants, potassium, and iron. Raisin is also low in sodium and abundant in thiamin, magnesium, phosphorus, and propionic acid. Raisin is most commonly found in morning cereals, bread items, confections and snacks, dairy products, meal preparations, and wine-making(Karimi et al,2011).

In this review of the project two major type of Raisins - Kecimen and Besni are considered which are cultivated in Turkey. Turkey being the top most in the list of world countries in the production of Raisins.

Raisins are obtained through a process of shrinking, this is achieved by their natural process of sun drying or other means such as Alkali treatment, Mechanical drying and Solar powered drying. In olden days the sun dried process is quite common, as it was time consuming it was later replaced with these advancements. Alkali treatment of grapes can reduce the drying time and make it more effective and reducing the cost of manufacturing too.(Karathanos et al., 1995)

There exist many methods for segregating the food products depending upon the quality of it. The primitive one is by human inspection which is inconsistent and tedious process. Even manual error is quite common, the productivity of the process is subjected to speculations. These drawbacks made to come with a new technique to evaluate the quality of the raisins. Machine Vision system will answer the drawbacks which human faced and will extract utmost data from images which in-turn plays a important part in evaluating the quality of the products.(ÇINAR et al., 2020)

Yu et al. (2011) classified sultana into four classes based on color, shape, and degree of wrinkle using the support vector machine (SVM). They achieved nearly 95% classification accuracy utilizing color and texture data on sultana photos, which is the greatest classification rate achieved by the SVM method.

To minimize dimensionality in this report, the Raisin Dataset is first subjected to Principal Component Analysis. Second, the original dataset and PCA transformed dataset are subjected to three prominent classification algorithms: Logistic Regression (LR), Ridge Regression Classifier Model (Ridge) and LDA (Linear Discriminant Analysis). The objective here is to determine whether the sultana is a Kecimen or a Besni. Because the observed results in this report are from a transformed dataset, the results of the classification algorithm are said to be obtained after applying PCA. The original dataset's classification results may be found in the Google Colab notebook.

.

II. Principle Component Analysis

Real-world datasets are frequently high in dimensionality. As a result, making storage, processing, and visualisation is difficult. Principal Component Analysis (PCA) can assist to relieve this problem by reducing the original dataset into a smaller, more manageable dataset that preserves the majority of the essential information (Principal component analysis, 2023). This important property of PCA makes it an efficient feature reduction strategy, with the ultimate goal of lowering the dimensionality of the dataset while maintaining as much variability or statistical information as feasible. PCA may give a simpler representation of the dataset without sacrificing critical characteristics by identifying the principle components that explain the most variance in the data. This can help with data exploration, visualisation, and analysis, as well as predictive model creation.

PCA Algorithm:

The following 6 methods can be used to apply PCA on a data matrix X with size n x p:

Standardization: The fundamental stage in PCA is to standardize the starting variable. They all contribute equally to the analysis at this point. When compared to features with lower variance in a given column, features with higher variance are more significant.

First, compute the mean vector x of each data set column. The mean vector is a p-dimensional vector with the formula

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad (1)$$

Subtract the mean of each column from each item in the data matrix to standardize the data. The final centred data matrix (Y) is written as follows:

$$Y = HX \quad (2)$$

where H is the centering matrix.

Ii) Covariance matrix Computation : The correlation between each pair of components in a given random vector is shown by a covariance matrix, which is a square matrix.

The centred data matrix's p x p covariance matrix S may be constructed as follows:

$$S = \frac{1}{n-1} Y^T Y \quad (3)$$

Iii) Eigen decomposition: The Eigen decomposition may be used to determine the eigenvalues and eigenvectors of S. Each principal component's (PC) orientation is shown by the Eigenvectors. The variation that is recorded by each PC is represented by the Eigenvalues. It is also defined as the coefficients of the eigenvectors. Consequently, the following equation can be used to calculate Eigen decomposition:

$$S = A\Lambda A^T \quad (4)$$

where A is the diagonal matrix of eigenvalues and is the p x p orthogonal matrix of eigenvectors.

The size n x p converted matrix Z is computed as one of the principal components. The columns of Z stand in for the PCs, whereas the rows of Z stand for the observations. The number of PCs is the same as the original data matrix's dimension. The formula for the value of Z is

$$Z = YA \quad (5)$$

III. MACHINE LEARNING-BASED CLASSIFICATION ALGORITHMS

Logistic regression

Logistic regression is similar to linear regression where it uses a binomial response variable.

The most significant benefit over Mantel-Haenszel or is the ability to employ continuous explanatory variables and manage more than two explanatory variables concurrently. Although it may appear trivial, this last characteristic is critical when investigating the impact of various explanatory variables on the response variable. (Sperandei, 2014)

When we examine numerous explanatory factors independently, we neglect covariance across variables and are prone to confounding effects. A logistic regression model will predict the likelihood of an outcome based on individual attributes. Because chance is a ratio, the logarithm of the chance given by will be modelled.

$$log(\pi/1-\pi) = \beta 0 + \beta 1x1 + \beta 2x2 + \ldots \beta mxm \quad (6)$$

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a supervised training technique used in machine learning for classification applications. It is a strategy for determining the optimal linear combination of characteristics for separating classes in a dataset. LDA works by projecting the data into a lower-dimensional space that maximizes class separation. This is accomplished by identifying a collection of linear discriminants that maximize the ratio of between-class variance to within-class variation. In other words, it determines the feature space directions that best segregate the distinct classes of data.

LDA presupposes that the data is Gaussian and that the covariance matrices of the various classes are identical. It also presupposes that the data is linearly separable, which means that a linear decision boundary can correctly categories the various classes. (GeeksforGeeks , 2023)

Ridge classification

Ridge classification is a machine learning approach for analyzing linear discriminant models. It is a type of regularization in which model coefficients are penalized to prevent overfitting. Overfitting is a typical problem in machine learning in which a model becomes too complicated and catches noise in the data rather than the underlying signal. This might result in poor generalization performance when dealing with fresh data. Ridge classification tackles this issue by including a penalty term into the cost function that discourages complexity. Ridge classification works by introducing a cost function penalty term that discourages complexity. The penalty term is usually the sum of the squared coefficients of the model's features. This causes the coefficients to stay modest, preventing overfitting. The penalty period can be changed to adjust the amount of regularization. A higher penalty leads to more regularization and lower coefficient values. This can be useful when there is a scarcity of training data.

IV. DATA SET DESCRIPTION

The dataset used in this study came from the UCI Machine Learning Repository. The data is divided into two categories of raisins: Kecimen and Besni. To categorize the raisins, many characteristics are analyzed and assessed. This study investigates a total of seven morphological characteristics for each grain of raisin in order to perform

image processing based on the forms of the raisins collected. Area, Perimeter, Major_Axis_Length, Minor_Axis_Length, Eccentricity, Convex_Area, and Extent are the characteristics. It has 900 entries for each of these characteristics. The Area is the amount of pixels inside the raisin's bounds. Perimeter is the distance between the raisin's edges and the pixels surrounding it to determine the environment. Major_Axis Length specifies the length of the major axis, which is the longest line that may be drawn on the raisin. The Minor_Axis_Length returns the length of the tiny axis, which is the shortest line on the raisin. Eccentricity offers a measure of the ellipse's eccentricity, which has the same moments as raisins. The Convex_Area returns the number of pixels in the region produced by the raisin's smallest convex shell. Extent is the ratio of the raisin's area to the total pixels in the enclosing box. And there are two classes namely, Kecimen and Besni raisins.

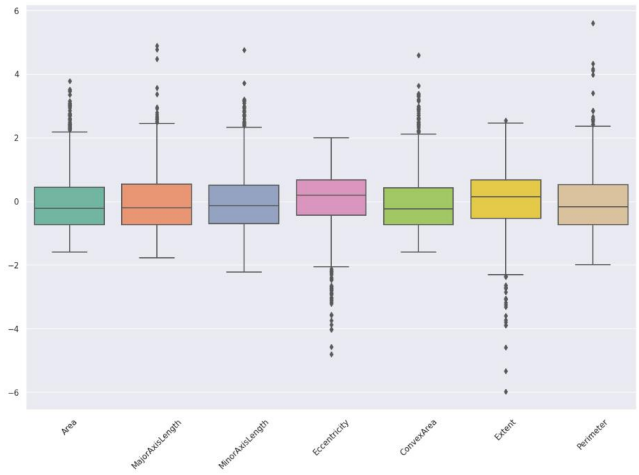


Fig 1: Box Plot

The box and whisker plots, as well as the five-number summary, are used to assess the distributions, central values, and variability of the attributes. Figure 1 depicts the acquired box and whisker plot for this project. This graph indicates that there are outliers in all aspects of raisins. With the exception of 'Extent,' all features have outliers on just one side, either the right or left, but 'Extent' has outliers on both sides.

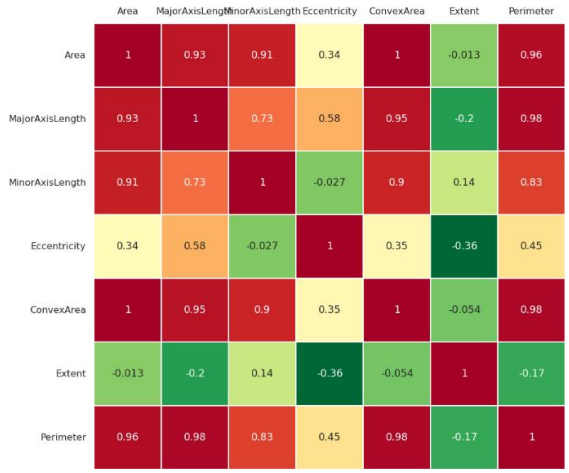


Fig 2: Correlation matrix

A Correlation matrix is shown in Figure 2. It displays the correlation coefficients for the various variables, and the correlation between all possible pairings of values is displayed in a table. It is a powerful tool for identifying and visualising patterns that may be extracted from enormous datasets. To comprehend the correlation, it is necessary to realise that if the result is -1, it shows a perfect negative linear correlation between two variables. If the value is 0, it indicates that there is no linear correlation between the two variables, and if the value is 1, it indicates that there is a perfect positive linear correlation between the two variables. Area, Major_Axis_Length, Minor_Axis_Length, Covex_Area, and Perimeter are the characteristics having the highest positive linear correlation in Figure 2. These five factors are very closely connected. The last two characteristics, Eccentricity and Extent, have a negative linear association with the others.

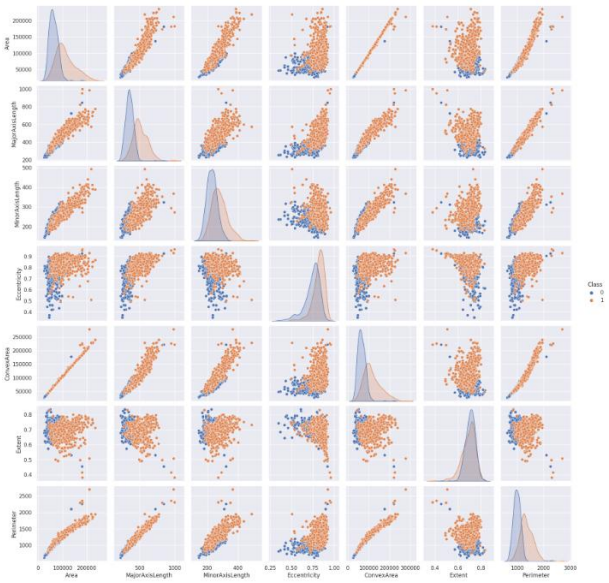


Fig 3: Pair Plot

Figure 3 depicts the Pair plot for this project, which interprets that highly correlated characteristics display a rising pattern of line while the least correlated data show a distinct pattern than others.

V. PCA RESULTS

Depending on the specific requirements and features of the data being analysed, PCA can be applied in a variety of ways. Here the Raisin Dataset is subjected to Principal Component Analysis. The PCA implementation in this project is done in Google Colab Notebook using the PCA library. This is a versatile technique for users. Some of the common ways to implement PCA are using standard python libraries such as NumPy, scikit-learn, TensorFlow, PyTorch, Keras or Statistics and Machine Learning Toolbox. The results achieved by all the techniques are almost identical; the only difference is that when utilising the PCA library, the result may be obtained easily.

The initial dataset of size n x p is reduced using the Eigenvector matrix A. Following the PCA steps, the set of 7 features can be reduced to r number of features. These r features are always fewer in number than the initial number of characteristics. Here, in this example r is less than 7. Each column in the eigenvector matrix A is a PC, and each PC captures a certain amount of data that determines the dimension (r). The raisin dataset's obtained eigenvector matrix (A) is as follows:

Eigenvector Matrix A =

$$\begin{pmatrix} 0.448 & -0.116 & 0.005 & -0.111 & -0.611 & -0.099 & -0.624 \\ 0.443 & 0.136 & -0.100 & 0.495 & 0.087 & -0.685 & 0.227 \\ 0.389 & -0.374 & 0.236 & -0.655 & 0.384 & -0.239 & 0.129 \\ 0.202 & 0.610 & -0.628 & -0.426 & 0.075 & 0.053 & 0.020 \\ 0.450 & -0.087 & 0.036 & 0.055 & -0.392 & 0.471 & 0.639 \\ -0.056 & -0.667 & -0.731 & 0.109 & 0.056 & 0.023 & -0.001 \\ 0.450 & 0.034 & 0.044 & 0.339 & 0.555 & 0.487 & -0.363 \end{pmatrix}$$

And the Eigenvalue (Lambda) λ =

$$\begin{pmatrix} 4.837 \\ 1.454 \\ 6.291 \\ 5.688 \\ 2.183 \\ 6.437 \\ 1.011 \end{pmatrix}$$

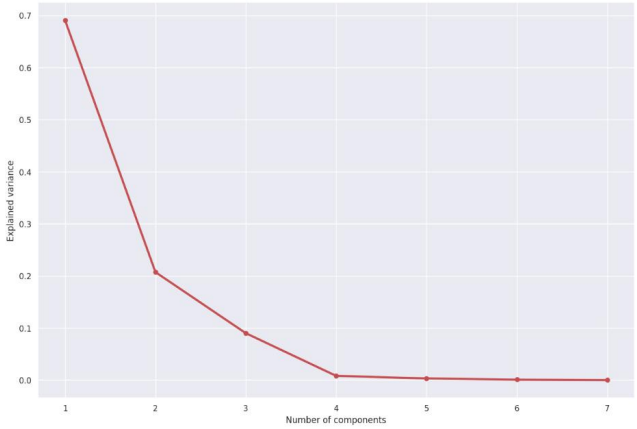


Fig 4: Scree Plot

Figure 4 depicts the Scree plot for this dataset, which demonstrates the variance that each main component can capture. It determines the number of elements that may be retained by locating the point at which the curve flattens. As a result, the number of factors retrieved before the curve flattens signifies the meaningful number of factors for your factor analysis. In this picture, the elbow is reported to be placed on the second main component, implying that two components should be created by the analysis. To put it another way, r = 2 indicates that the dimension may be decreased to two.

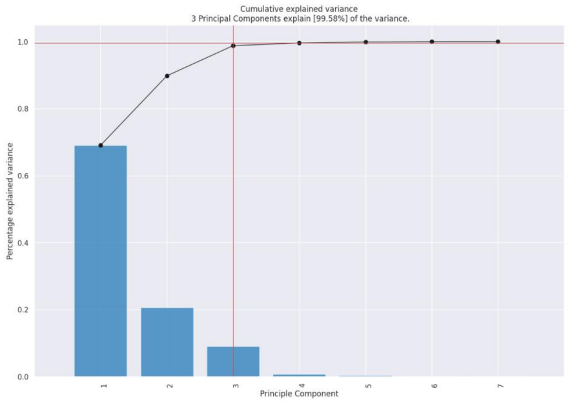


Fig 5: Pareto Plot

Figure 5 represents the Pareto plot for this dataset. This graphic also depicts the variance as a function of the major components. According to the figure, the first two PCs provide 89.7% of the variation in the original dataset. The variance of the first main component is 69%, or l_1=69%. And the variance of the second principal component is 20.7%, or l_2=20.7%.

The first principle component Z_1 is given by:

$Z_1 = 0.448X_1 + 0.443X_2 + 0.389X_3 + 0.202X_4 + 0.450X_5 + -0.056X_6 + 0.450X_7$

From Z_1 it can be observed that all the features are contributing the most except X_6. However, because X_6 (Extent) value is very small, we can ignore them and only consider:

$Z_1 = 0.448X_1 + 0.443X_2 + 0.389X_3 + 0.202X_4 + 0.450X_5 + 0.450X_7$.

The second principal component Z_2 is given by:

$Z_2 = -0.116X_1 + 0.136X_2 + -0.374X_3 + 0.610X_4 + -0.087X_5 + -0.667X_6 + 0.034X_7$

From Z_2, we will also ignore the extremely tiny numbers and simply consider,

$Z_2 = -0.116X_1 + 0.136X_2 + -0.374X_3 + 0.610X_4 + -0.667X_6$.

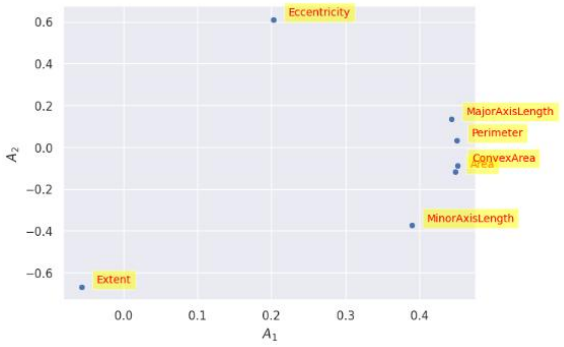


Fig 6: PC coefficient matrix

In Figure 6, the PC coefficient plot is shown. The contribution of each feature to the first two main components is predicted using this figure. According to figure 6, all other attributes have the greatest impact on the first PC, with the exception of eccentricity and extent, which both have the most impact on the second PC. Extent and Eccentricity are both plotted far away from the collection of other attributes.

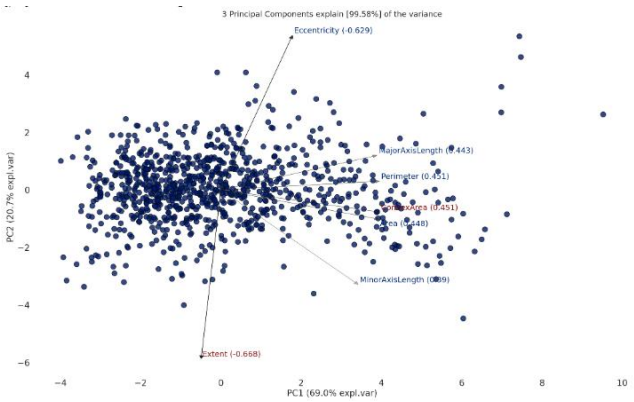


Fig 7: Biplot

The figure 7 depcts the Biplot where PC1 and PC2 are displayed in different ways.The rows of the eigenvector matrix are shown as a vector, and PC1 and PC2 serve as the biplot's axes. Each observation from the dataset is represented by a dot. It is clear from the biplot that eccentricity and extent exhibit distinct plot behaviour from other elements. It is said that every other feature forms a little angle with the first PC, which corresponds to the X-axis. They are alleged to be making extremely pronounced angles with PC2 on the Y-axis at the same time. This illustration supports the interpretation of Figure 6 that all features—aside from eccentricity and extent—contribute significantly to the first PC. In contrast, the eccentricity and extend have a modest angle with PC2 in the biplot since they contribute the most to PC2. Additionally, it could be inferred that all vectors pointing in the same direction have a positive correlation with one another. Major_Axis_Length, Minor_Axis_Length, Convex_Area, and Perimeter are all positively associated with one another in this biplot area.

VI. Classification of results

The outcomes of using three classification algorithm on the project's raisins dataset may be explained as follows. To test the efficiency of PCA on a dataset containing principal components, the classification algorithms are used to both the original dataset of raisins and the dataset of raisins on which PCA has been applied. Python's PyCaret package may be used for this categorization, and training and testing on the dataset have already been done. You can split the original dataset into 30% for testing and 70% for training.

The PyCaret module may be used to generate a performance comparison table between all available classification algorithms on the target dataset in order to identify the model with the highest accuracy.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
et	Extra Trees Classifier	0.8693	0.9320	0.8505	0.8849	0.8659	0.7385	0.7413	0.3240
rf	Random Forest Classifier	0.8622	0.9281	0.8433	0.8789	0.8589	0.7244	0.7278	0.6510
lightgbm	Light Gradient Boosting Machine	0.8622	0.9258	0.8433	0.8768	0.8585	0.7244	0.7268	0.1930
ridge	Ridge Classifier	0.8587	0.0000	0.8505	0.8657	0.8565	0.7174	0.7200	0.0370
lr	Logistic Regression	0.8551	0.9165	0.8257	0.8808	0.8495	0.7102	0.7155	0.2930
lda	Linear Discriminant Analysis	0.8551	0.9209	0.8575	0.8533	0.8542	0.7103	0.7122	0.0460
xgboost	Extreme Gradient Boosting	0.8464	0.9259	0.8328	0.8600	0.8443	0.6928	0.6960	0.1290
qda	Quadratic Discriminant Analysis	0.8463	0.9165	0.7653	0.9122	0.8301	0.6922	0.7035	0.0810
gbc	Gradient Boosting Classifier	0.8463	0.9190	0.8222	0.8671	0.8420	0.6927	0.6967	0.4290
ada	Ada Boost Classifier	0.8429	0.9091	0.8149	0.8657	0.8376	0.6857	0.6897	0.4040
nb	Naive Bayes	0.8234	0.9035	0.7224	0.9049	0.8011	0.6463	0.6619	0.0400
knn	K Neighbors Classifier	0.8180	0.8774	0.7793	0.8439	0.8086	0.6357	0.6397	0.0450
dt	Decision Tree Classifier	0.8005	0.8009	0.8076	0.8012	0.8003	0.6012	0.6075	0.0410
dummy	Dummy Classifier	0.5035	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0420
svm	SVM - Linear Kernel	0.4982	0.0000	0.9000	0.4474	0.5976	0.0000	0.0000	0.0340

Fig 8: Comparison among classification models before applying PCA

The best models with the highest accuracies were established, and they are ET (Extra Tree Classifier), RF (Random Forest Classifier) and Lightbgm(Light Gradient Boosting Machine) in this picture where classification models have been used earlier.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8659	0.9193	0.8505	0.8802	0.8630	0.7318	0.7355	0.2550
ridge	Ridge Classifier	0.8624	0.0000	0.8113	0.9031	0.8537	0.7245	0.7296	0.1110
lda	Linear Discriminant Analysis	0.8624	0.9186	0.8113	0.9031	0.8537	0.7245	0.7296	0.0770
knn	K Neighbors Classifier	0.8622	0.9127	0.8328	0.8846	0.8571	0.7243	0.7267	0.2530
lightgbm	Light Gradient Boosting Machine	0.8606	0.9212	0.8328	0.8811	0.8555	0.7210	0.7233	0.2300
rf	Random Forest Classifier	0.8605	0.9222	0.8292	0.8855	0.8541	0.7209	0.7256	0.9030
et	Extra Trees Classifier	0.8588	0.9218	0.8363	0.8777	0.8544	0.7177	0.7215	0.3570
xgboost	Extreme Gradient Boosting	0.8517	0.9168	0.8291	0.8682	0.8469	0.7032	0.7057	0.2230
gbc	Gradient Boosting Classifier	0.8445	0.9241	0.8148	0.8660	0.8380	0.6889	0.6924	0.2640
qda	Quadratic Discriminant Analysis	0.8411	0.9030	0.7724	0.8971	0.8282	0.6821	0.6910	0.1990
nb	Naive Bayes	0.8410	0.9189	0.7618	0.9087	0.8250	0.6819	0.6950	0.2520
ada	Ada Boost Classifier	0.8393	0.8967	0.8291	0.8468	0.8363	0.6786	0.6810	0.3050
svm	SVM - Linear Kernel	0.8323	0.0000	0.7865	0.8726	0.8227	0.6644	0.6739	0.1180
dt	Decision Tree Classifier	0.8040	0.8038	0.8147	0.7958	0.8037	0.6078	0.6103	0.1300
dummy	Dummy Classifier	0.5035	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1270

Fig 9: Comparison among classification models after applying PCA

The best models with the highest accuracies were established, and they are LR (Logistic Regression), Ridge (Ridge Classifier), and LDA (Linear Discriminant Analysis) and they are shown in this picture where classification

models are seen after the PCA has been performed. We will use these three categorization methods in this report to assess the outcomes of the overall project at this time. As a result, these algorithms will be used for training and assessment. Only the outcomes of the transformed dataset where the PCA was used will be presented in this report. However, the results for both classifications (from the original dataset as well as a transformed dataset) are available in the Google Colab Notebook.

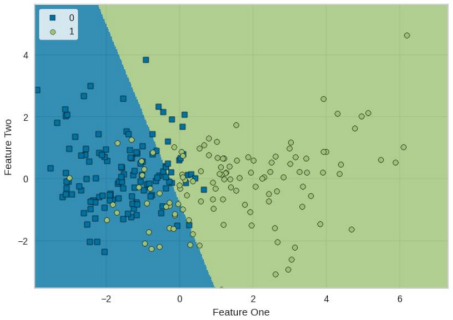
There are three phases in PyCaret for doing hyperparameter tuning, which is used to improve a model's performance. The first stage is building a model, which produces a classification model for each method. The second phase involves fine-tuning the model, which is done using the tune_model() function and ideal hyperparameters. They produce a stratified K-fold Cross-validation score after automatically tuning the model using efficient hyperparameters on a pre-defined search area. For each of the three techniques, PyCaret does 10-fold stratified K-fold validation by default.

```
tuned_lr_pca = tune_model(lr_pca)
```

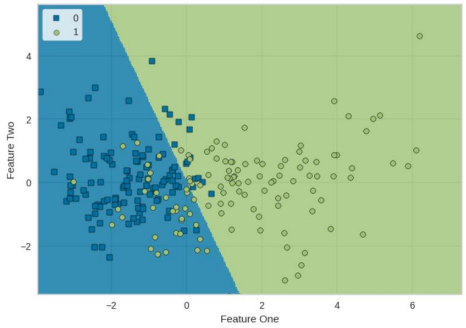
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.8246	0.9113	0.7857	0.8462	0.8148	0.6486	0.6502
1	0.8421	0.8744	0.8571	0.8276	0.8421	0.6843	0.6847
2	0.8772	0.9224	0.8571	0.8889	0.8727	0.7542	0.7546
3	0.7895	0.9027	0.8214	0.7667	0.7931	0.5793	0.5808
4	0.8947	0.9569	0.9286	0.8667	0.8966	0.7897	0.7916
5	0.8596	0.9532	0.8276	0.8889	0.8571	0.7196	0.7213
6	0.8929	0.9260	0.7857	1.0000	0.8800	0.7857	0.8044
7	0.9107	0.9158	0.9286	0.8966	0.9123	0.8214	0.8220
8	0.8393	0.8967	0.7500	0.9130	0.8235	0.6786	0.6897
9	0.9286	0.9413	0.8929	0.9615	0.9259	0.8571	0.8593
Mean	0.8659	0.9201	0.8435	0.8856	0.8618	0.7318	0.7359
Std	0.0406	0.0244	0.0578	0.0626	0.0414	0.0812	0.0820

Fig 10: LR metrics score after hyperparameter tuning

Fig. 10 illustrates that the metrics of the tuned LR model are superior to those of the basic model.



Logistic regression



Ridge Classifier

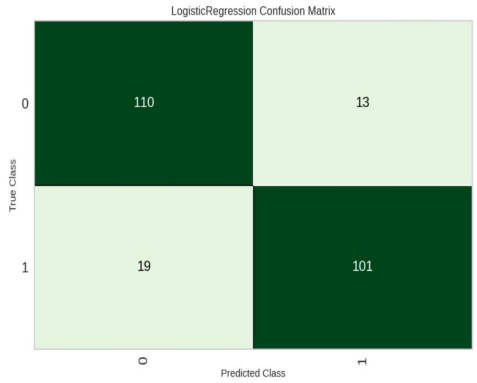
Linear Discriminant Analysis

Fig 11: Decision Boundaries of the three algorithms applied on transformed dataset.

The decision boundaries that the model on all three techniques for the altered dataset created are shown in Fig. 11. A decision boundary divides the samples of the two groups. The first and second PC are represented by the x- and y-axIs, respectively. The class 0 observations are represented by the square dots, whereas the class 1 observations are represented by the round dots. Here, the discrepancies between the algorithms' created decision boundaries may be seen. The chart indicates that LR has the better decision boundary when compared to LDA and Ridge Classifier. The LR decision border may more precisely distinguish between the data examples belonging to the two classes.

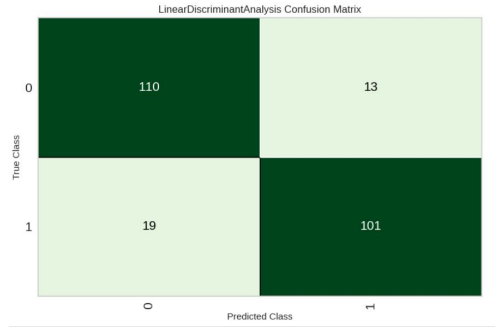
The dataset for raisins is classified in two categories. As a result, the target class will have two outputs: Kecimen and Besni. The performance of a classification model is assessed using a N x N matrix called a confusion matrix, where N is the total number of target classes. The matrix contrasts the actual goal values with what the machine learning model anticipated. The confusion matrix will show how well the classification model performs and identify the different kinds of mistakes it makes. A number of performance indicators, such as accuracy, precision, recall (sensitivity), specificity, and F1 score, may be computed using the confusion matrix. The model's capacity to categorise cases accurately, determine real positive and negative rates, and discriminate across classes is determined by these measures. A 2 x 2 matrix with four values—TP (True Positive), FP (False Positive), FN (False Negative), and TN (True Negative)—is used for binary classification.

The target variable in this confusion matrix will have two possible values: positive or negative. The rows will display the anticipated values, while the columns will display the actual values. If a value's anticipated and actual values are both positive and the same, the value is said to be TP. TN occurs when the expected and actual values are both negative. When a model predicts a positive value but the actual value is negative, this is referred to as a Type 1 error, or FP. In a similar way, FN is sometimes referred to as a Type 2 error and happens when the model predicts a positive result even while the actual value is positive.



Logistic regression

Ridge Classifier



Linear Discriminant Analysis

Fig 12: Confusion Matrices of the three classification algorithms applied on transformed dataset.

From the Confusion Matrix, we can see that 13 instances from class 0 (Kecimen) were incorrectly classed as class 1 (Besni), while 19 instances from class 1 (Besni) were incorrectly classified as class 0 (Kecimen) for Logistic regression. However, while Ridge misclassified 13 instances of class 0 (Kecimen) and 19 instances of class 1 (Besni), LDA misclassified 13 instances of class 0 (Kecimen) and 19 instances of class 1. The parameters known as Precision and Recall may be used to assess how well the categorization performed. Precision reveals the proportion of accurately anticipated situations that really resulted in a good outcome. Precision = TP / (TP + FP) is a formula that may be used to calculate the model's dependability. Recall illustrates how many of the actual positive cases we were able to accurately forecast using our model. Recall is equal to TP divided by (TP + FN). But in actual performance, it frequently occurs that a model's recall decreases as precision increases and vice versa. As a result, another metric known as the F1 score may be used to assess how well the categorization performed. F1-score provides a unified understanding of these two measures because it is the harmonic mean of Precision and Recall. When Precision and Recall are equal, it is at its maximum. It may be computed using the formula below:

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

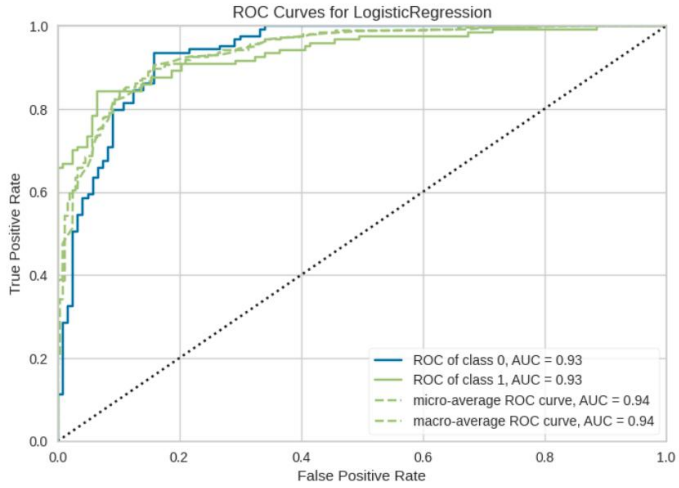


Fig 13: ROC curve LR

The Receiver Operating Characteristic (ROC) curve for the LR algorithm is seen in Figure 13. The ROC curve, which depicts the True Positive Rate and the False Positive Rate on the graph, is another tool used to show how well a classification model performs. True Positive Rate and False Positive Rate are two more key factors in the construction of the confusion matrix. In binary classification, the thresholds are various probability cutoffs that divide the two classes. It employs probability to inform us of a model's capacity to distinguish between classes (Machine Learning, 2023). As a result, it is possible to say that the confusion matrix and the ROC curves are comparable techniques but distinct visual representations of the same data. The output of the LR confusion matrix is shown in the image above. The True Positive Rate is on the Y-axis, while the False Positive Rate is on the X-axis. Different threshold values between 0.0 and 1.0 are plotted as curves with varying slopes. It can be deduced that the

LR algorithm is the best at predicting both types of raisins, and that it can forecast with 93% accuracy, based on the ROC curve and the AUC values. Along with ROC curves, Micro-average ROC curves and Macro-average ROC curves are also present here. It may be inferred from this that the three algorithms are capable of correctly categorising the raisins as Kecimen or Besni.

VII. Explainable AI with Shapley Values

The Shapley value technique to feature attribution in machine learning models is implemented by the Python package Shap. Lundberg and Lee created the "SHapley Additive exPlanations" acronym in 2017. Shap enables the computation of Shap values, which are a means to give each feature in a model's prediction for a certain input a numerical value. Shap values may be used to quantify the contribution of each feature to a prediction and to explain why a certain prediction was produced by a model. Several machine learning models, such as decision trees, random forests, and neural networks, can be used with the Shap library. Additionally, it can be applied to problems involving classification and regression. Summary plots, force plots, and dependency plots are just a few of the visualisation techniques the library offers to aid in the understanding of the Shap values. Users are able to observe the relative significance of each feature, how each feature's contribution varies over many instances, and how the values of several characteristics are connected through these visualisations. To determine how much each feature contributes to the final prediction provided by a model, machine learning uses a sort of feature attribution approach called "shap values." Shap values are computed by building a reference dataset that serves as a standard against which to measure the contribution of each feature to the model's prediction.

The Shap value graphic illustrates how each feature contributed to the model's final prediction.

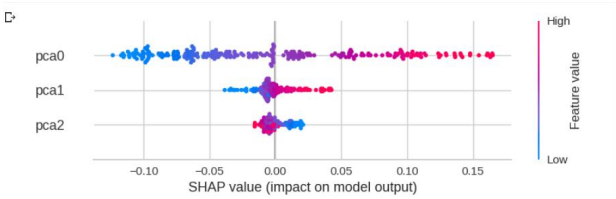
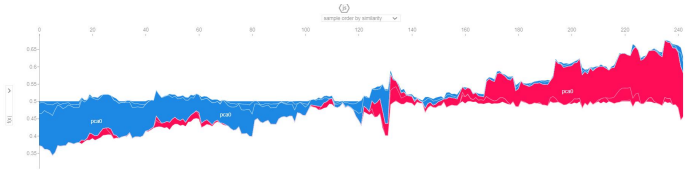


Fig 14: Summary Plot

In this figure 14, the feature's value or contribution is represented by a horizontal axis, while the feature's name is shown on a vertical axis. Each point on the plot corresponds to a single occurrence or observation, and its colour denotes the feature value. Low values are represented by blue points, whereas high values are shown by red points. How much the characteristic contributes to the forecast for that particular instance is shown by the point's location along the horizontal axis.

Fig 15: Force Plot

The figure 15 represents the Force Plot for a single observation. This characteristics individually help to increase the model output from the starting value as shown in this graphic. The base value is the value that would be anticipated in the absence of any characteristics for the current output. In other words, it is the test set's mean prediction. The base value in this case is 0.5. The bold 0.47 in the plot represents the model's score for this observation.



;

Fig 16: Combined Force Plot

The combined force map of all PCs is shown in Fig. 16. This map consists of all separate force charts that have been rotated by 90 degrees and stacked horizontally. The individual force plot's x-axis corresponds to the y-axis in this plot. The converted test set has 242 data points, hence the x-axis has 242 observations.

The combined force graphic depicts how each PC has an impact on the present projection. While values in the red colour have a negative impact on the prediction, those in the blue colour are thought to have a positive impact.

VIII. CONCLUSION

In conclusion, PCA has been used to analyse the original dataset of raisins. Three Machine Learning methods are then applied to the original one and the altered data set to determine which ones are the best fits for this *dataset. The* primary goal is to categorise the raisins into Kecimen and Besni according to their characteristics. The three models ET (Extra Tree Classifier), RF (Random Forest Classifier) and Lightbgm (Light Gradient Boosting Machine) perform the best on the original dataset. LR (Logistic Regression) has been shown to be the first-best model for the modified dataset. Ridge Classifier and LDA (Linear Discriminant Analysis) are the other two top models which was equally best fit for the modified dataset. The feature set may be reduced from seven to two since the first two major components account for 89.7% of the variation. Numerous tests are run on the first two PCs, and various graphs are created to validate the outcomes from various angles. Confusion matrices, ROC curves, and F1-scores have been used to conduct performance evaluation of the models once they have been developed and tuning using the optimal hyper-parameters. To sum up, it has been demonstrated that these algorithms are effective in classifying the different varieties of raisins.

References

1) *Principal component analysis: A review and recent developments*. (n.d.). Retrieved April 29, 2023, from https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202

2) Tripadvisor, J.-C. C. S. E. O. S. at. (n.d.). JC Chouinard. Retrieved April 29, 2023, from https://www.jcchouinard.com/confusion-matrix-in-scikit-learn/

3) *Machine learning - AUC - roc curve*. Python Machine Learning - AUC - ROC Curve. (n.d.). Retrieved April 29, 2023, from https://www.w3schools.com/python/python_ml_auc_roc.asp

4) Karathanos, V. T., Karanikolas, T., Kostaropoulos, A. E., & Saravacos, G. D. (1995). Non enzymatic browning in air-drying of washed raisins. In *Developments in Food Science* (Vol. 37, pp. 1057-1064). Elsevier.

5) ÇINAR İlkay, KOKLU, M., & TAŞDEMİR, Ş. (2020). Kuru üzüm Tanelerinin Makine Görüşü ve Yapay Zeka yöntemleri Kullanılarak Sınıflandırılması. *Gazi Journal ofEngineeringSciences*, *6*(3), 200–209. https://doi.org/10.30855/gmbd.2020.03.03

6) Yu, X., Liu, K., Wu, D., & He, Y. (2012). Raisin quality classification using least squares support vector machine (LSSVM) based on combined color and texture features. *Food and Bioprocess Technology*, *5*, 1552-1563.

7) Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia medica*, *24*(1), 12-18.

8) GeeksforGeeks. (2023, March 13). *ML: Linear discriminant analysis*. GeeksforGeeks. Retrieved April 29, 2023, from https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/