Principal Component Analysis and Classification of Raisins

Krishna Madani - 40196344
GitHub Link**:** **https://github.com/krishh1703**

Abstract — Principal Component Analysis (PCA) is an unsupervised learning technique that is applied to large datasets in order to reduce their dimensionality. PCA is very helpful to find the sequence of linear combinations of variables and plotting data in 2D and 3D. It is an efficient and most popular technique that compresses data by retaining all the information. Thus, it can be said that PCA is used to visualize multidimensional data, resize images, analyze the data, search high-dimensional datasets patterns, etc. For this report, I have applied PCA to a dataset of Raisin to classify them into two classes namely Kecimen and Besni. The three Machine Learning Algorithm: K Nearest Neighbors, Logistic Regression, and Ridge Regression Classifier models are applied to this dataset and have been used to transform the dataset to identify the type of Raisins after applying PCA. All these models are tuned with hyper-parameters to produce superior performance metrics and are then assessed using the F1 score, the fusion matrix, and the receiver operating characteristic (ROC) curves. To demonstrate how each model fits the dataset, the decision boundaries for each model are also displayed. For this project, in the PyCaret library, the KNN (K Nearest Neighbor) is the best model as compared to the other models of Machine Learning. Therefore, the algorithms successfully determine the two classes of the Raisin dataset, and a F1-score of nearly 1 has been obtained.

Index Terms — Principal Component Analysis, Binary Classification, K-Nearest Neighbor, Logistic Regression, Ridge Regression Classifier Model.

I. INTRODUCTION

'A raisin is a dried grape. Raisins are produced in many regions of the world and may be eaten raw or used in cooking, baking, and brewing [3]'. In this project, there are two varieties of Raisins- Kecimen and Besni that are grown in Turkey. 'Turkey is one of the countries that ranks top in the world's grape production. [4]'. 'Firstly, a total of 900 pieces raisin grains were obtained, from an equal number of both varieties. These images were subjected to various preprocessing steps and 7 morphological feature extraction operations were performed using image processing techniques. [4]'. The assessment for the quality and classification of food like raisin is possible using the traditional methods but it is time-consuming and ultimately it becomes expensive. 'In addition, human-made procedures from traditional methods can be inconsistent and more inefficient, as well as physical conditions such as fatigue and even people's psychological mood can affect the outcome of the work. [4]'. The development of alternative techniques to rapidly and accurately assess the fundamental characteristics of products like raisins- is motivated in large part by these unpleasant scenarios and issues. Thus, using ML techniques, the classification can be made easily and very accurately to classify them based on their features.

In this report, to reduce the dimensionality, first of all, the Principal Component Analysis is applied to the Raisin Dataset. Secondly, three popular classification algorithms, K-Nearest Neighbor (K-NN), Logistic Regression (LR), and Ridge Regression Classifier Model (Ridge) are applied to the original dataset and PCA transformed dataset. The goal here is to classify whether the raisin is a Kecimen or a Besni. The observed results mentioned in this report are used from a transformed dataset and therefore the results obtained from the classification algorithm are said to be obtained after applying PCA. The categorization outcomes of the original dataset can be found in the Google Colab notebook.

The report structure is organized as follows: Section I gives the Introduction, Section II - The PCA methodology description, Section III - The overview of the three classification algorithms, Section IV provides the Raisin dataset description, Section V discusses about PCA results, Section VI - extensive analysis of the classification results, and in section VII, the conclusion is drawn.

II. PRINCIPAL COMPONENT ANALYSIS

Almost all the datasets available in the real world have very high dimensionality. Thus, it is challenging to store these huge datasets, and processing them is also challenging, expensive and something that is even impossible to visualize. The original dataset that is large is transformed into a smaller dataset by applying PCA and it still contains the majority of information of the original dataset. This is the main characteristic of PCA. Hence, PCA is a feature reduction technique or 'PCA is a technique for reducing the complexity of high-dimensional data while preserving trends and patterns. [5]'. It achieves this by reducing the number of dimensions in the data to act as feature summaries.

- PCA algorithm

PCA can be applied to a data matrix X with dimension n

× p using the followings steps [6]:

- Standardization: To standardize the initial variable is the main step in PCA. By this step, they all contribute equally to the analysis. At first, compute the mean vector $\bar{x}$ of each column of the data set. The mean vector is a p-dimensional vector that can be expressed by

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

  To standardize the data, subtract the mean of each column from each item in the data matrix. The final centered data matrix (Y) can be expressed as follows:

$$Y = HX,$$

  where H represents the centering matrix.

- Covariance matrix computation: The p × p covariance matrix S of the centered data matrix can be computed as follows:

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

- Eigen decomposition: The eigenvalues and eigenvectors of S can be obtained using the Eigen decomposition. The Eigenvectors represent the direction of each principle component (PC). The Eigenvalues represent the variance that is captured by each PC. Thus, Eigen decomposition can be computed using the following equation:

$$S = A\Lambda AT$$

  where A means the p × p orthogonal matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues.

- Principal Components: It computes the transformed matrix Z that is of the size n × p. The rows of Z represent the observations and the columns of Z represent the PCs. The number of PCs is equal to the dimension of the original data matrix. The equation of Z can be given by:

$$Z = Y\ A$$

III. MACHINE LEARNING CLASSIFICATION ALGORITHMS

➢ K-Nearest Neighbor:

The most used and simplest algorithm in Machine Learning is K-NN. 'The K-Nearest Neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point [7]'. In simple words, this algorithm stores all the instances in n-dimensional space that match the training data points. 'When an unknown discrete data is received, it analyzes the closest k number of instances saved (nearest neighbors) and returns the most common class as the prediction and for real-valued data, it returns the mean of K-nearest neighbors. [6]'. It is also called a lazy learning algorithm as it does not use the training data points to make any generalizations. KNN being a supervised learning classifier, depends on the labeled input set of data. It trains the set of data that has labels and then predicts the labels for a point that does not have any labels. Thus, there are many advantages of this algorithm: this K-NN has a quick calculation time, it is versatile as it can be used for both regression and classification, it is highly accurate and there is no need of making any assumptions about the data. But it should also be noted that the accuracy depends on the data and its quality. Also if the data is very large, then the prediction might be slower.

➢ Logistic Regression:

In Machine learning, Logistic Regression (LR) classifier is a supervised learning classifier. The link between a dependent variable and one or more independent variables can be estimated using logistic regression. Therefore, the prediction between a categorical variable versus a continuous variable can be obtained by this algorithm. The categorical variable is binary that has values like true or false, yes or no, 0 or 1, etc. Also, Logistic regression can represent data sufficiently and accurately if the sample size of the dataset is large, otherwise, it is not able to represent the values efficiently across all the response categories. Here, in this project to classify the raisin, we have two classes so the LR classifier is the best-fit algorithm to make the classification into- Kecimen raisin and Besni raisin.

➢ Ridge Regression:

'Ridge regression is a model tuning method that is used to analyze any data that suffers from multi-collinearity. This method performs L2 regularization. [10]'. The equation of a ridge regression model can be given as Y = XB + e, where Y is the dependent variable, X is the independent variable, B is the regression coefficient that is to be estimated, and e represents the errors that are residuals. 'Once we add the lambda function to this equation, the variance that is not evaluated by the general model is considered. After the data is ready and identified to be part of L2 regularization, there are steps that one can undertake. [10]'. The very first step in this regression model is Standardization. To standardize the variables, the step is to subtract the mean and divide it by the standard deviation of each for both the dependent and independent variables. In this step, all the ridge regression calculations are based on standardized variables. Once the regression coefficient is obtained, all these variables are considered as per their original scale. But, the ridge trace is done on a standard scale. The ridge regression assumptions are the same as the linear regression

assumptions – they are linear, constant variance, and independent. The only difference is that the ridge regression does not provide confidence limits, so there is no need of assuming the distribution of errors to be normal. Therefore, ridge regression helps in reducing errors.

IV. DATA SET DESCRIPTION

The dataset used here for this project has been obtained from the UCI Machine Learning Repository. The data can be classified into two types of Raisin- Kecimen and Besni. Multiple features are considered and have been measured to classify the raisins. Here, in this project, there is a total of 7 morphological features studied for each grain of raisin in order to do image processing based on the shapes of the raisins obtained. The features are Area, Perimeter, Major_Axis_Length, Minor_Axis_Length, Eccentricity, Convex_Area, and Extent. The Area is the portion contained by the number of pixels within the boundaries of the raisin grain. Perimeter is the measured environment present between the boundaries of the raisin grain and the pixels around it. Major_Axis_Length represents the longest line that could be drawn on the raisin grain while the small axis is the Minor_Axis_Length representing the shortest line that can be drawn on the raisin grain. Eccentricity is the measure of the eccentricity of the ellipse, it has the same moments as raisins. Convex_Area is the number of pixels of the smallest convex shell region that is formed by the raisin grain. Extent is a ratio of the region formed by the raisin grain to the total pixels in the box that is bounded.

The distributions, central values, and variability of the features are measured using box and whisker plots and the five-number summary. The obtained box and whisker plot for this project is shown in Figure 1.

This figure shows that there are outliers existing in all the features of raisins. Except for 'Extent', all features have outliers only on one side either the right side or left side, whereas 'Extent' has outliers on both sides.

Figure 2 is a Correlation matrix. It shows the correlation coefficients for the different variables, and the correlation between all the possible pairs for the values is represented in a table. It is a useful tool to identify and visualize the patterns that could be summarized from a large dataset. In order to understand the correlation, it is important to know that if the value is -1 then it represents a perfect negative linear correlation between two variables. If the value is 0 then indicated no liner correlation between the variables and if the value is 1, it represents a perfect positive linear correlation between the two variables.

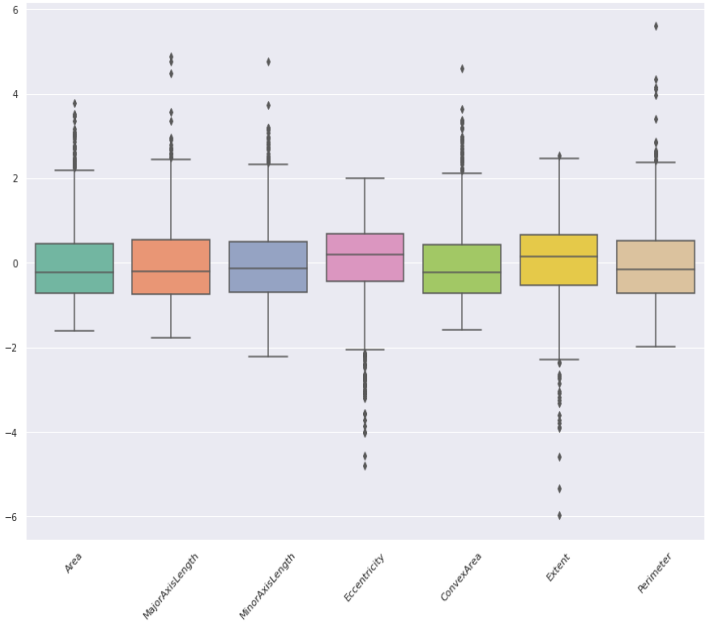


Figure 1: Box and Whisker Plot

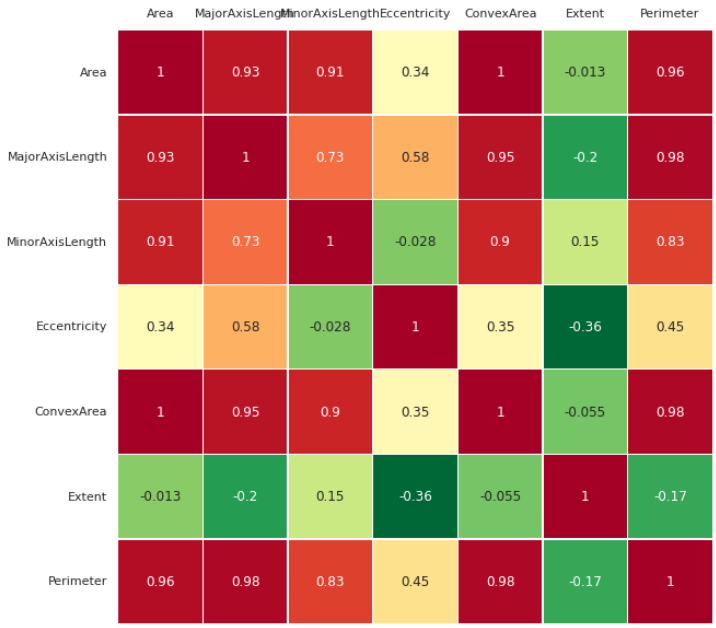


Figure 2: Correlation Matrix

Here, in Figure 2, the features with large positive linear correlation are Area, Major_Axis_Length, Minor_Axis_Length, Covex_Area, and Perimeter. All these five are highly correlated. The remaining two features- Eccentricity and Extent show a negative linear correlation with others.

Figure 3 is the Pair plot for this project that interprets that the highly correlated features show an increasing pattern of line whereas the least correlated features show a different pattern than others.

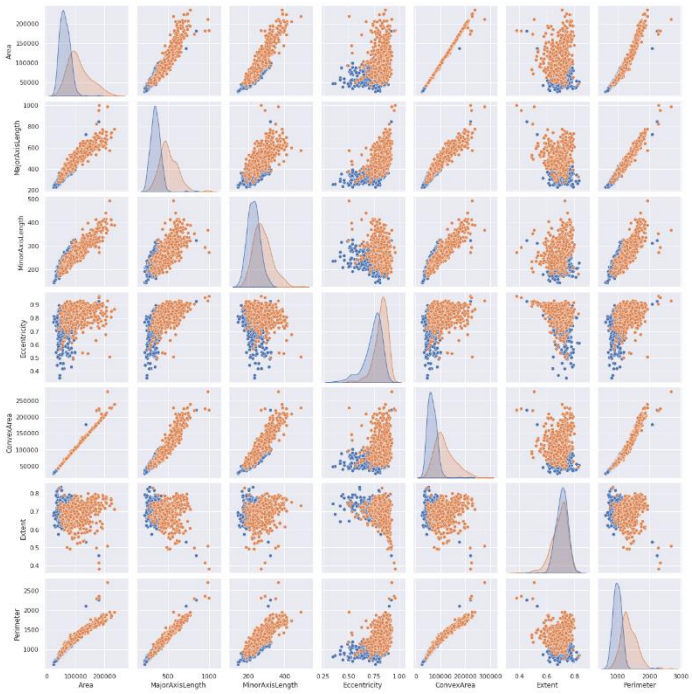


Figure 3: Pair Plot

V. PCA RESULTS

Principal Component Analysis is applied to the Raisin Dataset. Here, in this project, the PCA implementation is done using the PCA library in the Google Colab Notebook. This is a flexible method for the users. There is another method of implementing PCA- to develop PCA using standard Python Libraries like Numpy. The results obtained by both the methods are same, the only difference is that using the PCA library, the result can be obtained easily by just writing a single line of code.

Eigenvector matrix A is used to reduce the original dataset of size n × p. After applying the steps of PCA, the features set which has 7 features can be reduced to r number of features. These r features are always less than the original number of features. Here, r < 7. The column of each eigenvector matrix A is a PC and each PC captures an amount of data that determines the dimension (r). The obtained eigenvector matrix (A) for the raisin dataset is as follows:

Eigenvector Matrix A =

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

And the Eigenvalue (Lambda) λ =

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

Figure 4 represents the Scree plot for this dataset and it shows the variance that can be captured by each principal component. Here, in this figure, the elbow is said to be located on the second principal component, hence it can be said that 2 number of factors should be generated from the analysis. In other words, r = 2 means that the dimension can be reduced to two.

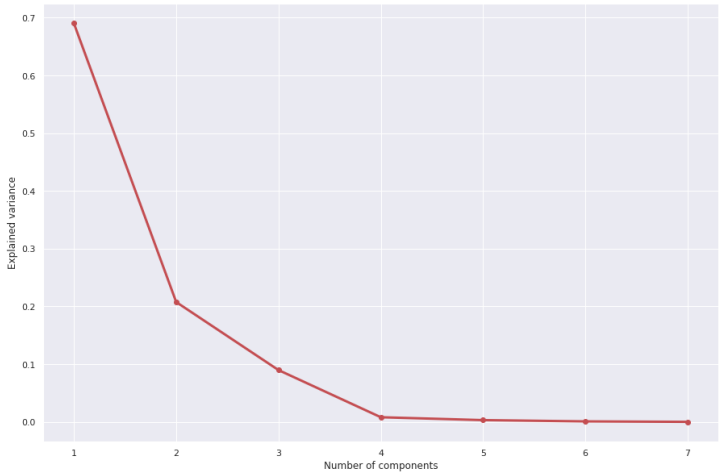


Figure 4: Scree Plot

Figure 5 represents the Pareto plot for this dataset. This plot also shows the variance by the principal components. It can be said from both the figures (Figure 4 and Figure 5) that the first two PCs make a total of 89.7% of contribution to the variance of the original dataset. The first principal component has a variance of 69% or we can say l1=69%. And the second principal component has a variance of 20.7% or we can say l2=20.7%.

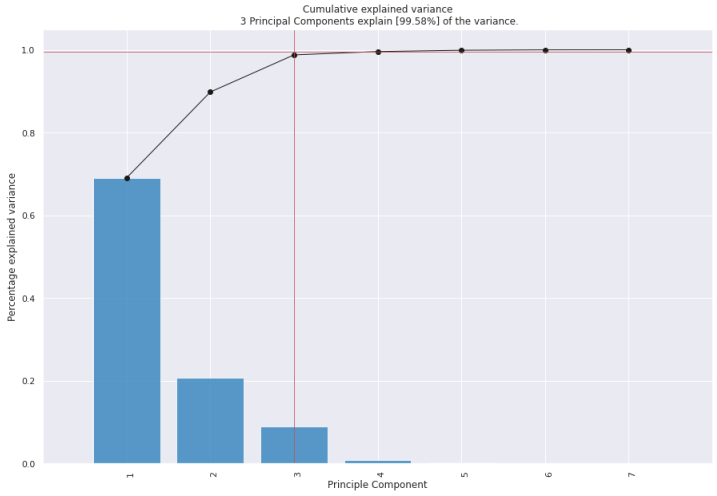


Figure 5: Pareto Plot

The first principal component is Z1 = $0.448X_1 + 0.443X_2 + 0.389X_3 + 0.202X_4 + 0.450X_5 + -0.056X_6 + 0.450X_7$

but we can neglect some values as they are very small and consider only Z1 = $0.448X_1 + 0.443X_2 + 0.389X_3 + 0.202X_4 + 0.450X_5 + 0.450X_7$

The second principal component is Z2 = $-0.116X_1 + 0.136X_2 + -0.374X_3 + 0.610X_4 + -0.087X_5 + -0.667X_6 + 0.034X_7$ here also, we will neglect the very small values and consider only Z2 = $-0.116X_1 + 0.136X_2 + -0.374X_3 + 0.610X_4 + -0.667X_6$

Figure 6 is used to represent the PC coefficient plot. This plot is used to predict the contribution of each feature on the first two principal components. From figure 6, it can be said that all other features except Eccentricity and Extent have the maximum contribution on the first PC while these both have maximum contribution towards the second PC. Both Eccentricity and Extent are plotted far from the cluster of other features.

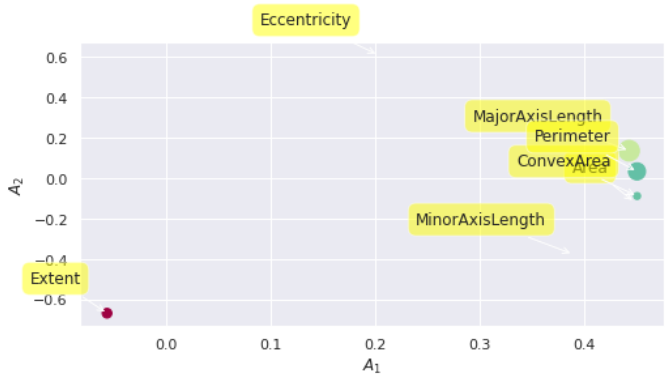


Figure 6: PC Coefficient Plot

A different representation can be obtained from the next plot which is the Biplot shown in Figure 7. It displays the first two PCs in a different way.

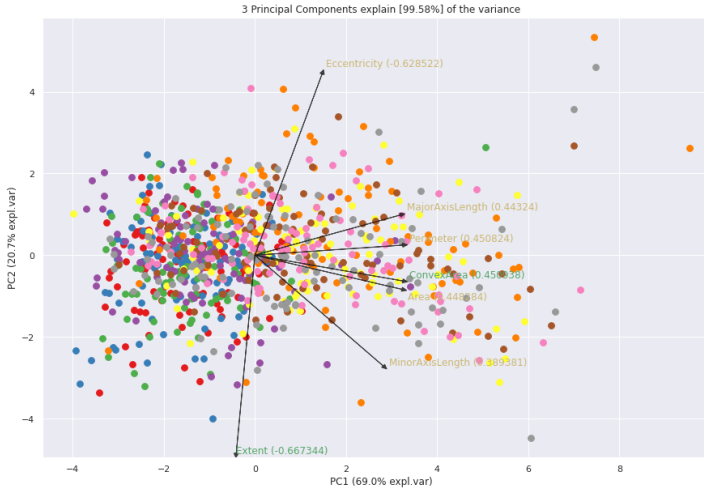


Figure 7: Biplot

PC1 and PC2 are the axes of the biplot and rows of the eigenvector matrix are plotted as a vector. The dots represent each observation from the dataset. From the biplot, it could be observed that Eccentricity and Extent show different plot behavior than others. All other features are said to be having a small angle with the first PC that corresponds to the X-axis. At the same time, they are said to be forming very large angles with PC2 on Y-axis. This representation justifies the statement interpreted from Figure 6 that all features except Eccentricity and Extent contribute in a large proportion to the first PC. On the other hand, the Eccentricity and Extend contribute maximum to PC2 and so they have a small angle with PC2 in the biplot. Also, it could be interpreted that all the vectors that are in the same direction, are correlated positively with each other. Here, in the biplot- Area, Major_Axis_Length, Minor_Axis_Length, Convex_Area and Perimeter are positively correlated with each other.

VI. CLASSIFICATION OF RESULTS

After applying three classification algorithms on the raisin dataset for this project, the results can be discussed as follows. The Classification algorithms are applied on both- the original dataset of raisins as well as on PCA applied dataset of raisins in order to check the effectiveness of PCA on the dataset with the principal components. PyCaret library of Python can be used for this classification and there is training and testing done on the dataset. The original dataset can be divided into 70% for training and 30% for testing purposes.

To find the best model which possesses the highest accuracy, the PyCaret library can be used to create a performance comparison table among all available classification algorithms on the target dataset. Figure 8 and Figure 9 are as follows:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8852	0.9384	0.8634	0.9065	0.8829	0.7704	0.7736	0.322
et	Extra Trees Classifier	0.8763	0.9296	0.8387	0.9114	0.8720	0.7528	0.7572	0.159
rf	Random Forest Classifier	0.8693	0.9251	0.8246	0.9096	0.8635	0.7387	0.7439	0.195
gbc	Gradient Boosting Classifier	0.8641	0.9228	0.8248	0.9018	0.8586	0.7282	0.7346	0.128
lda	Linear Discriminant Analysis	0.8623	0.9286	0.8600	0.8695	0.8630	0.7245	0.7274	0.011
ridge	Ridge Classifier	0.8605	0.0000	0.8461	0.8775	0.8591	0.7210	0.7251	0.010
qda	Quadratic Discriminant Analysis	0.8603	0.9246	0.8036	0.9093	0.8519	0.7209	0.7273	0.012
lightgbm	Light Gradient Boosting Machine	0.8568	0.9227	0.8280	0.8819	0.8533	0.7136	0.7160	0.125
ada	Ada Boost Classifier	0.8534	0.9106	0.8283	0.8774	0.8509	0.7069	0.7097	0.089
nb	Naive Bayes	0.8377	0.9099	0.7618	0.9033	0.8254	0.6757	0.6854	0.011
knn	K Neighbors Classifier	0.8146	0.8654	0.7867	0.8400	0.8108	0.6294	0.6334	0.017
dt	Decision Tree Classifier	0.8075	0.8074	0.8212	0.8022	0.8107	0.6148	0.6166	0.012
svm	SVM - Linear Kernel	0.5105	0.0000	0.9000	0.4572	0.6063	0.0109	0.0240	0.010
dummy	Dummy Classifier	0.5035	0.5000	1.0000	0.5035	0.6698	0.0000	0.0000	0.010

Figure 8: Comparison among classification models before applying PCA

In this figure, where classification models are applied before, the best models with the highest accuracies were founded and those are LR (Logistic Regression), ET (Extra Tree Classifier), and RF (Random Forest Classifier).

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.8604	0.9060	0.8108	0.9032	0.8541	0.7211	0.7255	0.022
lr	Logistic Regression	0.8570	0.9206	0.8424	0.8748	0.8561	0.7139	0.7179	0.015
ridge	Ridge Classifier	0.8569	0.0000	0.7969	0.9102	0.8486	0.7142	0.7212	0.009
lda	Linear Discriminant Analysis	0.8569	0.9213	0.7969	0.9102	0.8486	0.7142	0.7212	0.011
lightgbm	Light Gradient Boosting Machine	0.8553	0.9056	0.8283	0.8792	0.8517	0.7107	0.7137	0.038
nb	Naive Bayes	0.8552	0.9199	0.7933	0.9089	0.8467	0.7107	0.7171	0.012
ada	Ada Boost Classifier	0.8517	0.9077	0.8145	0.8839	0.8465	0.7036	0.7076	0.082
qda	Quadratic Discriminant Analysis	0.8463	0.9112	0.7865	0.8994	0.8370	0.6930	0.7011	0.022
rf	Random Forest Classifier	0.8429	0.9069	0.8004	0.8800	0.8366	0.6859	0.6909	0.195
gbc	Gradient Boosting Classifier	0.8412	0.9094	0.8144	0.8657	0.8376	0.6824	0.6858	0.082
et	Extra Trees Classifier	0.8357	0.9171	0.8001	0.8651	0.8290	0.6716	0.6763	0.161
svm	SVM - Linear Kernel	0.8184	0.0000	0.8387	0.8202	0.8257	0.6367	0.6430	0.010
dt	Decision Tree Classifier	0.8007	0.8004	0.8107	0.8023	0.8039	0.6010	0.6052	0.010
dummy	Dummy Classifier	0.5035	0.5000	1.0000	0.5035	0.6698	0.0000	0.0000	0.007

Figure 9: Comparison among classification models after applying PCA

In this figure, where classification models are observed after the PCA has been applied, the best models with the highest accuracies were founded and those are KNN (K-Nearest Neighbor Classifier), LR (Logistic Regression), and Ridge (Ridge Classifier). Here, in this report, we will carry out these three classification algorithms for evaluating the results of the entire project now. So, the training, tuning, and evaluation will be performed according to these algorithms. This report will only show the results of the transformed dataset (where the PCA is applied). But in the Google Colab Notebook, there are the results for both the classifications (original dataset as well as a transformed dataset).

Hyperparameter tuning is used to obtain improvement in the performance of a model and there are three steps in PyCaret to perform this. The first step is creating a model, the classification model per algorithm is produced. The second step is to tune the model and the tune_model() is used to tune the model with hyperparameters that are ideal. The model is automatically tuned with effective hyperparameters on a search space that is already pre-defined and they obtain a stratified K-fold Cross-validation score. In PyCaret, for all three algorithms, by default, there is an application of 10-fold stratified K-fold validation.

Figure 10 shows the KNN metrics score after the hyperparameter tuning is applied. To tune different models, different parameters are used. For example, the number of K-Nearest members is tuned for KNN algorithm and the penalty to L2 is tuned for the LR algorithm model. Also the difference can be seen in figure 10 which represents that the tuned KNN model is better than the base model of that metrics.

Fig. 11 illustrates the decision boundaries formed by the model on all three algorithms for the transformed dataset. The samples of both classes are separated by a decision boundary. The x-axis and y-axis correspond to the first and the second PC respectively. The square-shaped dots represent the observations for class 0 and the round-shaped dots represent the observations for class 1. The differences among the decision boundaries that are formed by the algorithms can be observed here. It can be said from the figure that KNN has the best decision boundary than LR and Ridge Classifier. The decision boundary of KNN can separate the data instances of both classes more accurately.

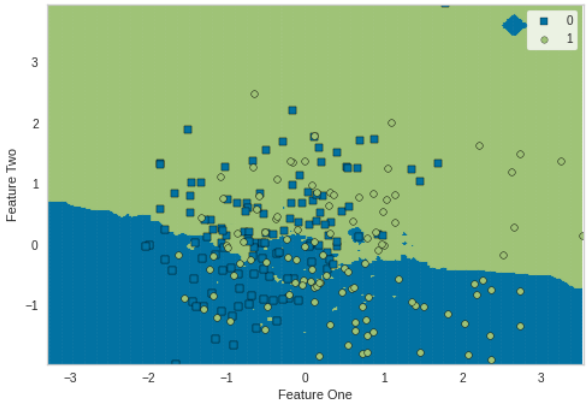
```
tuned_knn_pca = tune_model(knn_pca)
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.8421	0.9002	0.7586	0.9167	0.8302	0.6851	0.6958
1	0.8772	0.8904	0.8621	0.8929	0.8772	0.7545	0.7549
2	0.8772	0.9655	0.8621	0.8929	0.8772	0.7545	0.7549
3	0.8070	0.9317	0.6897	0.9091	0.7843	0.6156	0.6349
4	0.8596	0.8849	0.8276	0.8889	0.8571	0.7196	0.7213
5	0.9298	0.9446	0.8571	1.0000	0.9231	0.8593	0.8679
6	0.8393	0.9082	0.7857	0.8800	0.8302	0.6786	0.6825
7	0.8929	0.9267	0.8571	0.9231	0.8889	0.7857	0.7877
8	0.9286	0.9528	0.8929	0.9615	0.9259	0.8571	0.8593
9	0.8571	0.9184	0.8214	0.8846	0.8519	0.7143	0.7161
Mean	0.8711	0.9223	0.8214	0.9150	0.8646	0.7424	0.7475
SD	0.0369	0.0255	0.0579	0.0365	0.0414	0.0733	0.0707

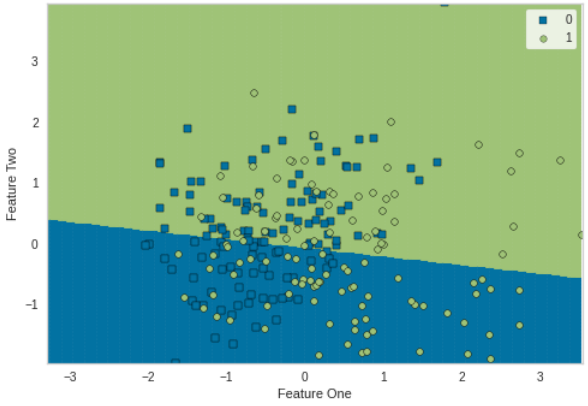
Figure 10: KNN metrics score after hyperparameter tuning

The raisin dataset is a binary classification. Therefore, there will be two outputs of the target class- Kecimen and Besni. 'A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. [11]'. The confusion matrix will demonstrate the performance of the classification model and will detect the types of errors that it makes. For Binary classification, there is a 2 x 2 matrix and there are four values for this matrix namely TP (True Positive), FP (False Positive), FN(False Negative), and TN(True Negative). Here, in this confusion matrix, the target variable will have values- Positive or Negative. The columns will represent the actual values while the rows

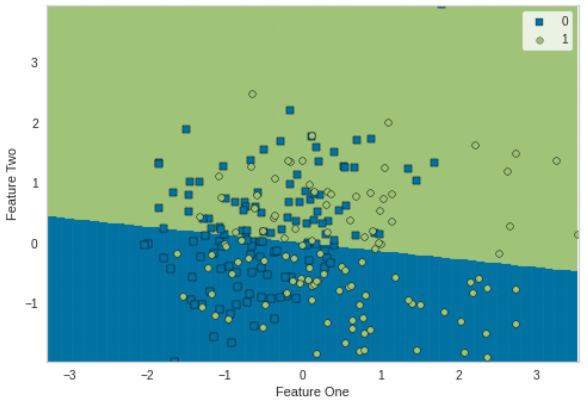
will represent the predicted values. A value is said to be TP (True Positive) if the predicted value and the actual value is the same and is a positive value. A TN (True Negative) is when the predicted value matches the actual value and is a negative figure. The FP (False Positive) is also known as a Type 1 error, and it occurs when the actual value was negative but the model predicts a positive value. And similarly, FN (False Negative) is also known as a Type 2 error, and it occurs when the actual value is positive but the model predicts it as a negative value.



KNN (K-Nearest neighbr)

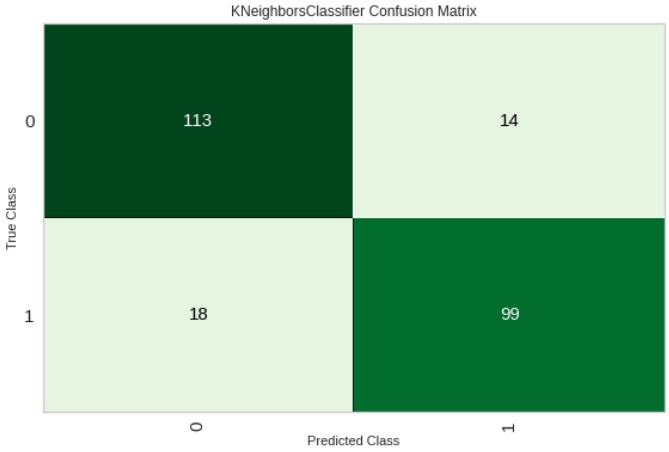


LR (Logistic regression)

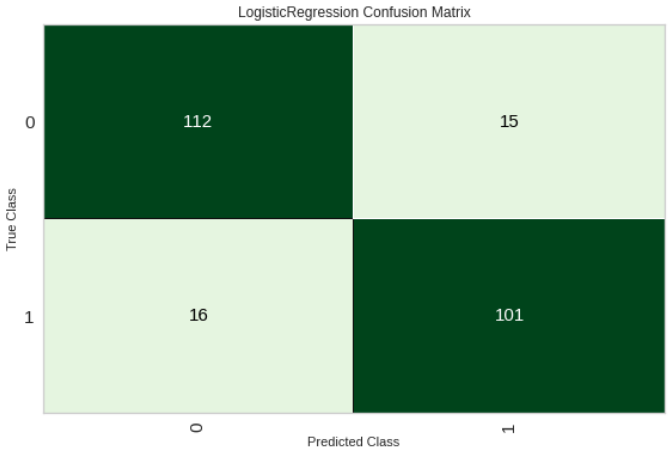


Ridge Classifier

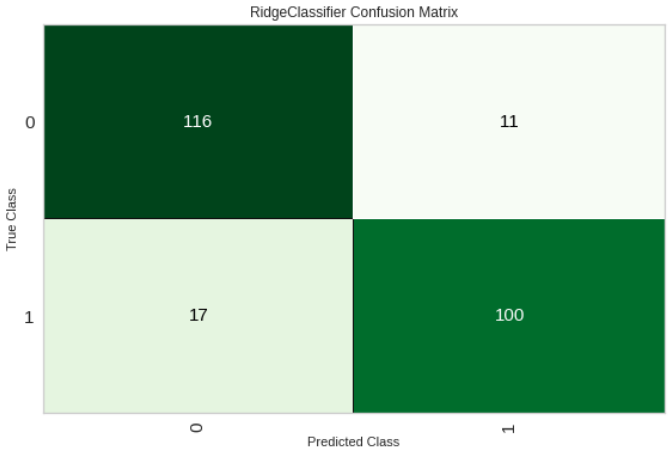
Figure 11: Decision Boundaries of the three algorithms applied on transformed dataset



KNN Confusion Matrix



LR Confusion Matrix



Ridge Classifier Confusion Matrix

Figure 12: Confusion Matrices of the three algorithms applied on transformed dataset

KNN misclassified 14 instances from class 0 (Kecimen) as class 1 (Besni) and 18 instances of class 1 (Besni) are misclassified as class 0 (Kecimen). On the other hand, LR misclassified 15 instances of class 0 (Kecimen) and 16 instances of class 1 (Besni) whereas Ridge misclassified 11 instances of class 0 (Kecimen) and 17 instances of class 1 (Besni). The performance of the classification can be evaluated by the measurements called Precision and

Recall. 'Precision tells us how many of the correctly predicted cases actually turned out to be positive. [11]'. It determines the reliability of the model and it can be calculated as Precision = TP / (TP + FP). 'Recall tells us how many of the actual positive cases we were able to predict correctly with our model. [11]'. It can be calculated as Recall = TP / (TP + FN). But it often happens in practical performance that if the precision is increased then the recall decreases for the model and vice-versa. Thus, there is another measurement called the F1 score that can be used to evaluate the performance of the classification. 'F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall. [11]'. It can be calculated by the following formula:

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

Figure 13 shows the Receiver Operating Characteristic (ROC) curve for the KNN algorithm. The ROC Graph is also used to demonstrate the performance of a classification model, and this curve plots two parameters on the graph which are the True Positive Rate and the False Positive Rate. The main parameters for building the confusion matrix are also True Positive Rate and False Positive Rate. Thus, it can be said that the confusion matrix and the ROC curves are similar methods but different visual representations for the same data. Here, in the below figure, the result of the KNN confusion matrix can be observed. The X-axis has the False Positive Rate and Y-axis has the True Positive Rate. There are different threshold values plotted in the shape of curves, where the values are between 0.0 and 1.0. it can be interpreted that the ROC curve and the AUC values evaluate the KNN algorithm and declare it to be the best at predicting both types of raisin and it can predict with 93% accuracy. There are Micro-average ROC curves and Macro-average ROC curves present here along with ROC curves. Thus, it can be concluded that the three algorithms are capable of successfully classifying the raisins as Kecimen or Besni.

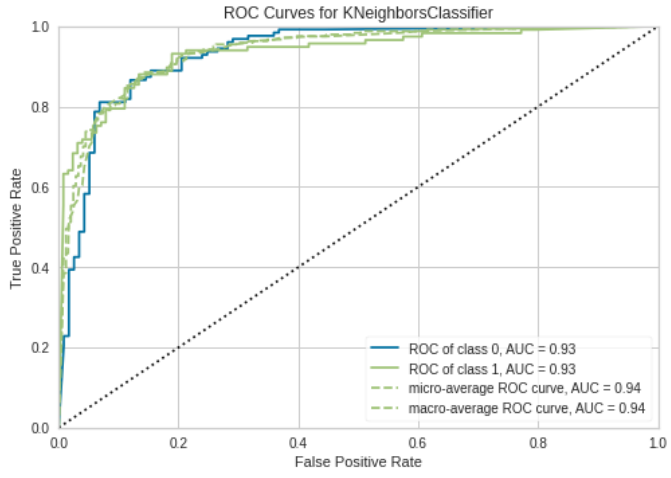


Figure 13: ROC curve KNN

VII. CONCLUSION

The conclusion is- PCA has been applied to the original dataset of raisins. Then on the transformed data set along with the original one, three Machine Learning algorithms are applied which are obtained as the best models for this dataset. The main aim is to classify the raisins into Kecimen and Besni, based on their features. The best models for the original dataset are LR (Logistic Regression), ET (Extra Tree Classifier), and RF (Random Forest Classifier). For the transformed dataset, KNN (K-Nearest Neighbor) has been obtained as the first-best model. The other two best models are LR (Logistic Regression) and Ridge Classifier model. The first two principal components form 89.7% variance and the feature set can be reduced to 2 from 7. On the first two PCs, numerous experiments are carried out, and various graphs are produced to validate the results from different perspectives. The models are created and then they are tuned with the ideal hyper-parameters and performance evaluation has been conducted by using confusion matrices and ROC curves and F1-scores. All in all, to summarize, these algorithms have been proven successful to determine the types of raisins.

REFERENCES

[1] Biswal, A. (2022, November 30). Principal Component Analysis in machine learning: Simplilearn. Simplilearn.com. Retrieved December 15, 2022, from https://www.simplilearn.com/tutorials/machine-learning-tutorial/principal-component-analysis

[2] UCI Machine Learning Repository: Raisin Dataset Dataset. (n.d.). Retrieved December 15, 2022, from https://archive.ics.uci.edu/ml/datasets/Raisin+Dataset

[3] Wikimedia Foundation. (2022, December 12). Raisin. Wikipedia. Retrieved December 15, 2022, from https://en.wikipedia.org/wiki/Raisin

[4] Home » dergipark. (n.d.). Retrieved December 15, 2022, from https://dergipark.org.tr/tr/download/article-file/1227592

[5] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433– 459, 2010.

[6] A. B. Hamza, Advanced Statistical Approaches to Quality. Unpublished

[7] What is the K-nearest neighbors algorithm? IBM. (n.d.). Retrieved December 16, 2022, from https://www.ibm.com/topics/knn

[8] Chatterjee, M. (2022, December 13). A quick introduction to KNN algorithm. Great Learning Blog: Free Resources what Matters to shape your Career! Retrieved December 16, 2022, from https://www.mygreatlearning.com/blog/knn-algorithm-introduction

[9] What is logistic regression? IBM. (n.d.). Retrieved

December 16, 2022, from https://www.ibm.com/topics/logistic-regression

[10] Team, G. L. (2022, November 16). What is ridge regression? Great Learning Blog: Free Resources what Matters to shape your Career! Retrieved December 16, 2022, from https://www.mygreatlearning.com/blog/what-is-ridge-regression/

[11] Bhandari, A. (2022, November 29). Confusion matrix for Machine Learning. Analytics Vidhya. Retrieved December 16, 2022, from https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/