Principal Component Analysis and Multi-Class Classification of Dry Bean Types

Mohamed BOUZID
Student ID: 40221466
https://github.com/medbouzid/INSE6220

***Abstract*—Dry beans are the most produced legume crops in the world and different cultures' products are made with them. The wide range of their genetic diversity made seed classification essential for both marketing and production and for a smart agricultural method. In this report, a dataset of 13000 dry beans samples is analyzed with the goal of automatically classifying bean types. First, PCA - a dimensionality reduction technique - is applied to obtain uncorrelated features with the maximum of information. Then, three Machine Learning (ML) algorithms (LightGBM, Random Forest, Gradient Boosting Classifier) have been applied, evaluated and compared to each other using different metrics: F1-Score, Confusion matrix, ROC. The obtained results show a good capability of the models to distinguish between different species with LightGBM being the best.**

***Index Terms*—PCA, Random Forest, Gradient Boosting, LightGBM, F1-score, Dry beans**

I. Introduction

Dry bean is the most produced legume in the world and it has an important role in the agriculture of many countries (USA, Turkey, etc.). It has gained importance as a food given its availability, its cheap price, and its health and dietary benefits. However, climate changes can affect plant growth. As a result, identification of the seed characteristics and breeding of new seeds could improve the plant resistance and tolerance to these changes. It must be noted also that the identification of the best seed is the most common problem for the producer and the market [1]. Any low-quality seed, even if it is in the best condition, will induce a low quantity. Analysis and classification of dry beans types have been an area of concern for crop management for a while. It constitutes an important part of precision farming, an area dedicated to leveraging ML for better management of livestock, crops, water, and soil. ML has been involved in agriculture with the goal of automating human based processes that have been used in agriculture. The focus of this report is to use ML methods to classify and identify bean variety with an emphasis on reducing the dimensionality of the features using PCA. The paper includes a detailed description of the dry bean dataset followed by the application of the PCA method. Finally, three ML algorithms are applied to the original and transformed dataset.

The rest of the report is organized as follows: Section II and III provide a brief review of the PCA and the used ML algorithms respectively, Section IV is a detailed description of the dataset, Section V presents the output of PCA while Section VI presents the classification outputs and finally, Section VII summarizes the results and concludes

II. Principal Component Analysis

In real-world tasks, datasets are usually complex and high-dimensional. Hence, data visualization becomes difficult and data analysis computations become expensive. It might be even impractical to look into the relationships between each variable and this could put us in the danger of overfitting. Technically, we want to reduce the dimension of the feature space by either eliminating the features or extracting other ones from them. The first one is not efficient as we might be dropping important information from variables. The second one does not drop variables but it creates new independent variables where the new feature is a combination of the old ones.

PCA is a technique for dimensionality reduction for feature extraction. It aims to find directions of maximal variance of the data. Those directions must be mutually orthogonal. In other words, PCA is an unsupervised learning technique that transforms high dimension features into a smaller number of uncorrelated variables known as principal components (PC). Every PC is a linear combination of the original variables [4]. These PCs are obtained in a way that the first few PCs are uncorrelated between each other and are capturing the maximum information in the original data by minimizing the distance between the data and its projection onto the PC [5].

In the PCA algorithm, the dataframe is represented as a data matrix X of size $n \times p$, where n is the number of observations and p is the number of process variables. The PCA is applied after that by following these steps [5]:

- **Step 1**: Get the centered data matrix Y by subtracting off-column means. Column means are can be computed as follows :

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

  These columns means can be used to generate a transformation matrix that results in the centered data matrix Y which can be written as:

$$Y = HX \tag{2}$$

- **Step 2**: Calculate the covariance matrix S, of size $p \times p$ of the centered data matrix Y by the following equation:

$$S = \frac{1}{n-1}Y'Y \tag{3}$$

This step makes eigen decomposition (next step) possible and identifies any correlations features of the dataset has.

- **Step 3**: Calculate the eigenvectors and eigenvalues of the covariance matrix S using eigen decomposition. In fact, S can be written as follows:

$$S = A\Lambda A^T \tag{4}$$

where A is a $p \times p$ orthogonal matrix whose columns are the p eigenvectors of S, and Λ is a diagonal matrix whose elements are the eigenvalues of S in descending order. This is the most important step of the PCA since the eigenvectors represent the PCs and the eigenvalues represent the amount of variation present in the PCs respectively. As an example, the first eigenvector a_1 is the new variable with the most variation in data

- **Step 4**: Calculate the transformed data matrix Z of size $n \times p$ as follows:

$$Z = YA \tag{5}$$

The transformed data matrix is a representation of the data in the new coordinate system defined by the PCs.Once the transformed matrix Z is obtained, one might reduce the dimension of Z from $n \times p$ to $n \times r$ by removing the least important PCs. Typically, we want the explained variance of the r PCs to be bigger than 80%.

III. Classification Algorithms

Classification is a supervised learning concept with the goal of categorizing a dataset into classes. In this report, we have tested different algorithms using Pycaret. The best three best performing classification algorithms were chosen as an illustration in this work: a boosting algorithm known as gradient boosting classifier (GBC), a bagging algorithm known as Random Forest (RF), and another boosting algorithm known as Light Gradient Boosting Machine (lightGBM). As we can see the three of them are tree-based ensemble methods.

Ensemble methods is a revolutionary concept that has been introduced in last years and proven to be very powerful in the field of ML. In fact, instead of choosing a single predictor (weak learner), we aggregate them to obtain a better model. Boosting and Bagging are two special types of Ensemble methods.

A. Gradient Boosting Classifier

In the boosting concept, we combine multiple weak learners to have a stronger learner. This is done by allowing each model to pay attention to its predecessor's mistakes. In other words, the output of one model is the input of the following. Gradient boosting is the most popular boosting method. It uses decision trees in the model building like other boosting methods but the processing is somehow different. The difference lies in the underfitteed values of its predecessor. Gradient Boosting Classifier [6] tries to fit the new model to the residual error made by the previous one. At the moment of prediction, the first model just outputs the average but after that GBC builds trees one after another based on the errors made by the previous one until the predefined number of trees is reached or there is no improvement done.

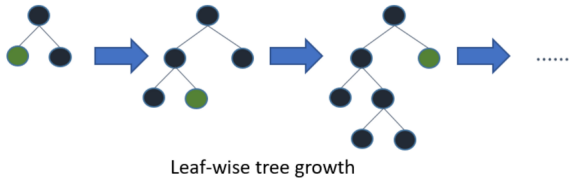


Fig. 1: Leaf-wise tree growth

B. Random Forest

Random Forest, another ensemble method, is a simple yet powerful machine learning algorithm. It relies on the concept of bagging. Bagging is an idea that suggests that combining the results of multiple models (average or majority vote) can result in a better-generalized result. This technique takes bags (subsets) and uses them to have an idea of the distribution. In contrast to boosting where each model boosts its successor, the multiple models of bagging run in parallel, and a combination of them is done at the end to increase the overall result. Random Forest is the most popular bagging method. Its name includes two key terms: Random and Forest. As its name suggests, forest consists of building an ensemble of individual decision trees that are uncorrelated. Each one of them predicts a class and the class label with the majority of votes becomes the model's prediction. The term Random comes from the fact that RF does not take the whole feature space but a random subset of features. This method results in a better model with more diversity.

C. Light GBM

LightGBM [7] is another ensemble boosting method that relies on a tree-based learning algorithm. This algorithm gained huge popularity in thelast years due to its high speed, as the term light implies. This computation speed is very practical as real-world datasets are usually large. The main reason behind this speed is that LightGBM grows the tree leaf-wise in contrast to level-wise that other tree-based algorithms do. The leaf with maximum loss is chosen to be grown and hence large losses are taken care of first. A simple representation of leaf-wise growth is shown in Figure 1. All boosted trees in LightGBM use this leaf-wise growth.

IV. Dataset Description

The dry bean dataset is a multivariate dataset that contains 13k samples of dry beans of different types. It is collected using a computer vision system that extracts the bean shape features from images. There are 7 classes (see Figure 2) in total in the dataset with the most frequent variety being "Dermason" and the least frequent one being "Bombay". There are 16 features in total which are area (A), Perimeter (P), Major Axis Length (L), Minor Axis length (I), Aspect ratio (K), Eccentricity (Ec), Convex Area (C), Equiv Diameter

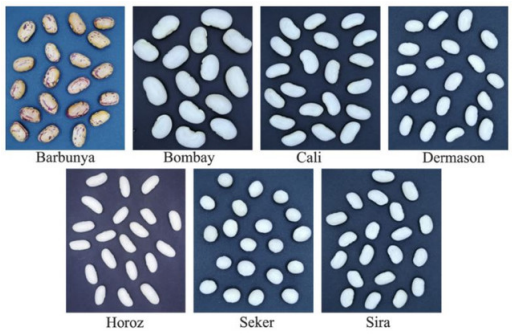


Fig. 2: Dry Beans Types

Fig. 3: Feature Correlation Matrix

($E_d = \sqrt{\frac{4A}{\pi}}$), Extent (Ex), Solidity ($S = \frac{A}{C}$), Roundness ($R = \frac{4\pi A}{P^2}$), Compactness ($CO = \frac{E_d}{L}$), ShapeFactor1, ShapeFactor2, ShapeFactor3, and ShapeFactor4. A detailed description of the dataset and how it was collected can be found in [8].

Due to the high number of features and the equation that relates some of them, we believe that there is a correlation between features and that a study of the correlation matrix could help us identify highly correlated ones. We show the correlation matrix in Figure 3. It is confirmed from the plot that the majority of features have a high correlation with each other. We illustrate by ConvexArea and Area, ShapeFactor3 and Compactness where both pairs correlation is equal to 1. It is also evident by a strong negative correlation (-0.99, -0.98) between other pairs like AspectRatio and Compactness, Major Axis Length and Perimeter, etc. On the other side, Extent (Solidity as well) is a feature that has the least correlation with other features. Those relationships are supported by a visualization of the pair-plot presented in Figure 4 where a strong correlation is distinguished by an increasing or decreasing line. Yet, we do not see any linear trend in Extent and Solidity. Another interesting trend that we see is that the first 8 features negatively correlated (green color) with the last 8 features and vice versa.

Box and whisker plots are also visualized and investigated

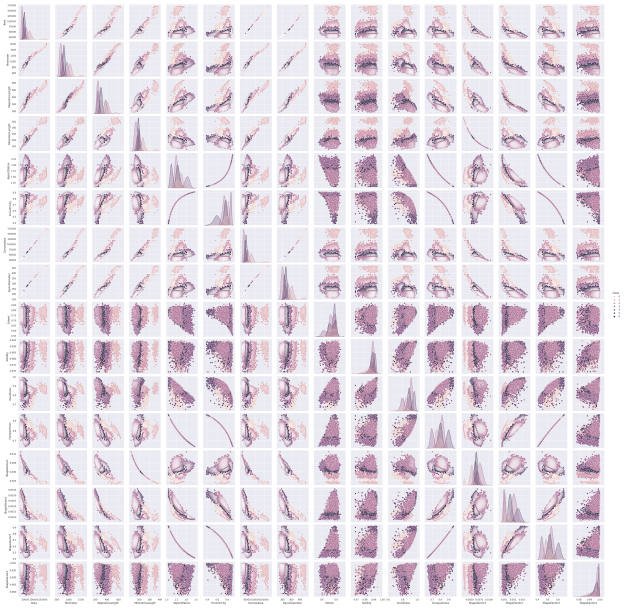


Fig. 4: Feature Pair Plot

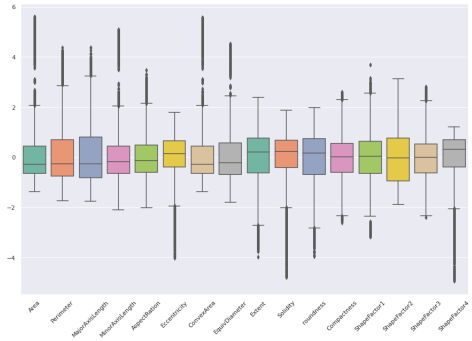


Fig. 5: Box and whisker plot of centered data

in Figure 5 using the five-number summary to measure the centering and the variability of the feature space. It is obvious that Area is right-skewed with a lot of maximum outliers. This is the same case for Perimeter, Aspect Ratio, Major Axis Length and many other features, but with fewer outliers. On the other side, Eccentricity, Extent, Solidity, and some other features are left-skewed. Finally, there are a few normally distributed features like Compactness and ShapeFactor 1. ShapeFactor 2 is the only feature that does not have any outlier and it is approximately normally distributed. We conclude that most of the features are skewed with more or less extreme values.

V. PCA Results

The distribution and the correlation of the data make it perfect for the PCA to intervene and transform the variables into uncorrelated ones. During this project, we followed the steps mentioned in Section II to reduce the dimension of the variables from $p = 16$ to r features with $r < 16$. To do this the Covariance Matrix of the centered matrix

and the eigenvector matrix (A) were calculated. A's columns represent the eigenvectors or the PC. For illustration purposes and because we have too many features, we only show the coordinates of the first 2 PCs of the dry beans dataset.

$$A_{2PC} = \begin{bmatrix} 0.28 & 0.24 \\ 0.31 & 0.18 \\ 0.32 & 0.07 \\ 0.22 & 0.37 \\ 0.24 & -0.32 \\ 0.24 & -0.32 \\ 0.28 & 0.24 \\ 0.29 & 0.22 \\ -0.07 & 0.21 \\ -0.15 & 0.06 \\ -0.26 & 0.17 \\ -0.24 & 0.32 \\ -0.20 & -0.37 \\ -0.31 & 0.12 \\ -0.24 & 0.32 \\ -0.21 & 0.08 \end{bmatrix}$$

Moreover, the corresponding eigenvalues which are the variance that each PC captures are:

$$\lambda = \begin{bmatrix} 9.25 \\ 3.98 \\ 1.15 \\ 0.83 \\ 0.47 \\ 0.16 \\ 0.11 \\ 0.05 \\ 0.008 \\ 0.001 \\ 0.0008 \\ 0.0002 \\ 0.0001 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The eigenvectors and their corresponding eigenvalues account for how much information they capture and will help us choose the reduced dimension r. To do that, we should calculate the percentage of variance accounted by each PC by following this equation [5]:

$$l_j = \frac{\lambda_j}{\sum_{j=1}^{p} \lambda_j} \times 100\% \quad for\ j = 1, ..., p \tag{6}$$

where λ_j corresponds to the j^{th} PC.

An acceptable amount of information that we could consider to reduce the dimension is 80%. After calculating, we find that $l_1 = 57.8\%$ and $l_2 = 24.88\%$. We are able to say that the first two components account for more than 80% of the variance (82.68%). We can plot the scree plot and the Pareto plot, which plot the number of PCs versus the explained and cumulative variance respectively to visualize the amount of variance that

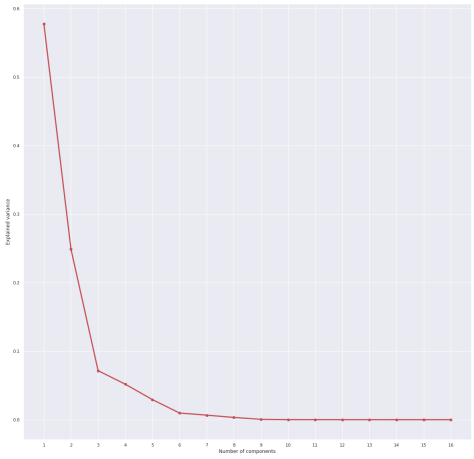


Fig. 6: Scree Plot

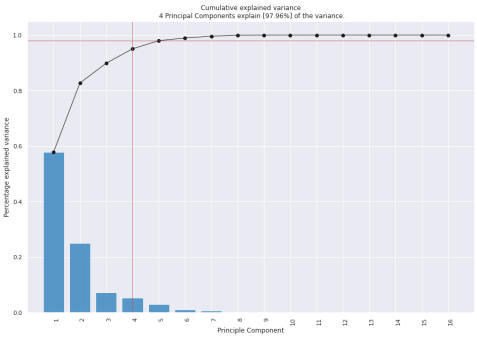


Fig. 7: Pareto Plot

each PC captures and confirm those results. The scree plot, shown in Figure 6 confirms the existence of the elbow at the second PC. In addition, the Pareto plot, shown in Figure 7, confirms that the first two PCs account for more than 80%. It is safe to say that the dimension of the data can be reduced to two.

Let us now analyze those two components and how they are expressed in terms of original variables. From the first column (first PC) of A, we can see that none of the features have a negligible contribution except for X_9 (Extent). We decide to neglect it and the first PC (Z_1) is given by this equation:

$$\begin{aligned} Z_1 = {} & 0.28X_1 + 0.31X_2 + 0.32X_3 + 0.22X_4 \\ & + 0.24X_5 + 0.2X_6 + 0.28X_7 + 0.29X_8 \\ & - 0.15X_{10} - 0.26X_{11} - 0.24X_{12} \\ & - 0.20X_{13} - 0.31X_{14} - 0.24X_{15} - 0.21X_{16} \end{aligned} \tag{7}$$

Looking at this equation, we can see that there is no significant difference between the contributions of different variables. The most contributor factors are X_2 (Perimeter), X_3 (Major Axis Length) and X_{14} (ShapeFactor2) and the least contributor factor is X_{10} (Solidity) but it is not negligible.

Let us now move to the second PC and how it is expressed in function of the original variable. We see from the matrix

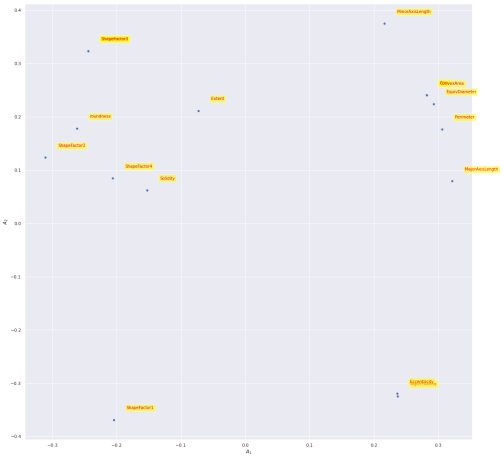


Fig. 8: PC Coefficient Plot

Fig. 9: Biplot

A that X_3 (Major Axis Length) and X_{10} (Solidity) and X_{16} (ShapeFactor4) are negligible contributors. So, the equation of the second PC can be written as:

$$\begin{aligned} Z_2 = {} & 0.24X_1 + 0.18X_2 + 0.37X_4 - 0.32X_5 \\ & - 0.32X_6 + 0.24X_7 + 0.22X_8 + 0.21X_9 \\ & + 0.17X_{11} + 0.32X_{12} - 0.37X_{13} + 0.12X_{14} \\ & + 0.22X_{15} \end{aligned} \tag{8}$$

It is observed that the largest contribution is captured by X_4 (Minor Axis Length) and X_{13} (ShapeFactor1) in absolute value. On the other side, the smallest non-negligible contribution is captured by X_{14} (ShapeFactor2).

In addition, those results are supported by the PC coefficient plot (see Figure 8) which is a visual representation of the contribution of the original variables with respect to the first two PCs that are considered. This representation will help us identify if there are any variables with similar relationship with respect to the PCs. In other words, it is a 2-D representation of the Eq. (7) and Eq.(8) with the axes being Z1 and Z2.

We can see four clusters in this plot. The first group is on the top right which includes variables that have a positive contribution to both PCs. The second group is on the bottom left which shows variables that have a negative contribution to both PCs. The other two groups represent variables that have a negative impact on one PC and a positive impact on the other and vice versa. Another interesting observation we can get from the plot is that some of the variables are on top of each other. We can illustrate by Eccentricity and Aspect Ratio or ConvexArea and Area. This makes sense since those pairs depend on each other in their equation. Moreover, it is visible and confirmed on the plot that Extent is the least contributing factor for Z_1 since it is the only one close to zero on the Z_1-axis while the others have a close contribution in absolute value. On the other side, Solidity, ShapeFactor4, and Major Axis Length are close to zero on the Z_2-Axis so that confirms their negligible contribution to PC 2. Finally, ShapeFactor1, Aspect Ratio and Eccentricity are separated from the others and have negative values on the Z_2-axis.

Another graphical way to analyze the effect of each feature on the Principal components is the biplot (see Figure 9). The term ”bi” means that the biplot allows information on both observations and variables of a data matrix to be displayed graphically. Observations are displayed as scatter points whereas variables are displayed as vectors. The length and the angle of the vector indicate how each variable contributes to the first two PCs. On the other hand, the points represent the score of each observation for the first two PCs.

Let us analyze the biplot. The first PC is represented by the horizontal axis. Starting with the first PC, the representation reaffirms the results of the equation and the coefficient plot that Major Axis Length, ShapeFactor2, and Perimeter are the largest contributors to the first PC. In fact, their angle with respect to the horizontal axis is the narrowest compared to the other variable. Added to that, ShapeFactor2 has a negative coefficient which explains why its vector lies in the left half. Finally, Extent has the widest angle with respect to the horizontal axis, so it is the least contributing factor.

Moving to the second PC, we can confirm that Major Axis Length, ShapeFactor4, and Solidity have the widest angle with respect to the vertical axis. Hence, they have a weak contribution to the second PC. Moreover, Extent has the smallest angles with the vertical axis followed by Minor Axis Length and Shape Factor 1. We can say that they are the largest contributors to the second PC. Added to that, most of the features have a positive impact on the second PC except for ShapeFactor1, Eccentricity, and AspectRatio which confirm what we said before.

Also, if we analyze similarities between vectors, we can see that Minor Axis Length and ShapeFactor1 have the same angle and the same length but in the opposite direction which confirms that they have a strong negative correlation. We can say the same for ShapeFactor3 and Eccentricity. Furthermore, we see a strong correlation in the pairs (Eccentricity, Aspect ratio) and (ConvexArea, Area) as both have the same length, angle

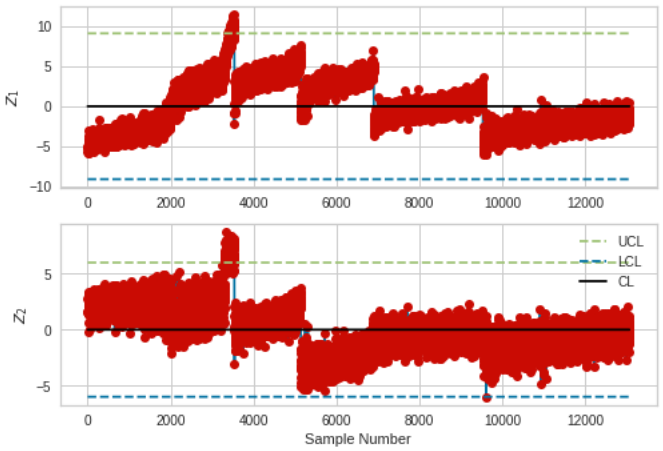


Fig. 10: Control charts for the first two PCs

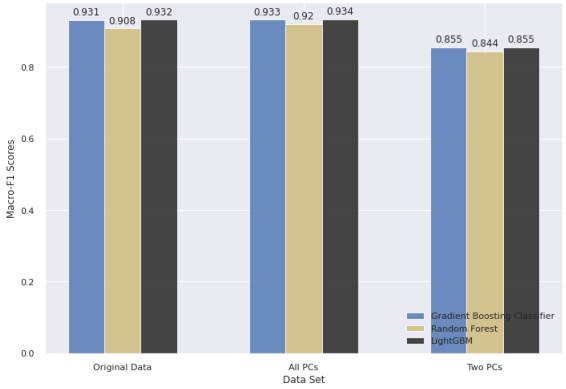


Fig. 11: Macro-averaged F1-scores

and direction. Finally, the features ShapeFactor1, Eccentricity, and Aspect Ratio are isolated from the other variables which indicate that they have a weak correlation with most of the other variables. All of those observations are consistent with what we talked about when analyzing the covariance matrix.

The last part of the biplot is the observations as they indicate the class distribution on the reduced dimension of the biplot. We can see that the first PC has a slightly higher variation (12 + 6 = 18) than the second PC (8 + 6 = 14). The class of each point (observation) is indicated by its color. We can see that Seker (purple) and Dermasson (green) have the smallest scores for the first PC, while Bombay (blue) has the highest score for the first PC and it is separated from the other observations. Bombay also has the highest score for the second PC while Horoz (red) has the lowest score for the second PC. In addition, Dermasson (green) is in the direction of ShapeFactor1 which indicates that ShapeFactor1 is a good characteristic to separate this class from the other. Horoz (red) on the other side is characterized by its Eccentricity. Also, The variables roundness and Compactness are good deciders for the Seker class while Cali (orange), Barbunya (pink) and Bombay (blue) are characterized by their area, Perimeter, minor and major axis length. Last but not least, we can see that the observation of Barbunya and Cali are almost on top of each other and the decision depends on the minor and major Axis length. This indicates a high similarity in characteristics between the two classes which is confirmed by their images in Figure 2. A final interesting observation that we can make is that the cluster of Bombay (blue) is separated from the other clusters.

Finally, control charts were visualized for the two PCs in Figure 10 to measure the centering and variation of the data. It has been shown that the two PCs have some out-of-control points between the 3000 and 4000 samples.

VI. Classification Results

A. Used Metrics

Since we are working on a multi-classification problem, the appropriate metric to use would be the F1-score. In fact, Accuracy would not be the right metric here since the dry beans dataset is a little bit imbalanced. Also, For a better understanding of the results, we investigate the confusion matrix and the Receiver Operating Characteristic (ROC).

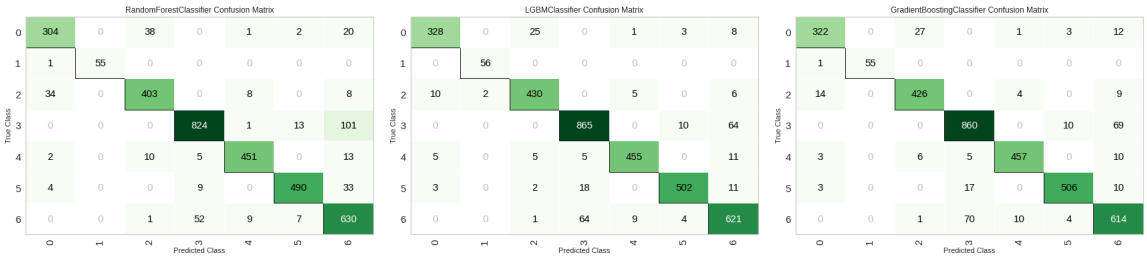
- **F1-Score**: The calculation of F1-Score is commonly used to evaluate the performance of multi-class classification. It depends on precision and recall where the precision measures the number of true positives over the number of true positives plus the number of false positives. Recall measures the number of true positives over the number of true positives plus the number of false negatives. F1-score is written as a combination of precision and recall as follows [9] [10]:

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

  The macro-averaged F1-score ($\sum_{c=1}^{C} \frac{F_1(c)}{C}$ is selected and computed as we have imbalanced classes as mentioned in Section IV.
- **Confusion Matrix**: The confusion matrix is a table representation of size $k \times k$ where k is the number of classes. It outlines the predicted values on the horizontal axis versus the actual values for each class in the vertical axis. It is a very useful metric as it facilitates identifying where the model had the most trouble making correct predictions (which class for example).
- **Receiver Operating Characteristic**: The ROC curve is the curve that plots the false positive rate against the true positive rate for different threshold values [9]. Also, the Area Under the Curve (AUC) is a single value summary of the curve. The best value of the AUC is 1.0 while a value of 0.5 indicates that the classifier is random.

B. Experimental Results

In this part, experimental results are shown and a comparison of the performance of the used algorithms on dry



(a) Random Forest (b) LightGBM (c) Gradient Boosting

Fig. 12: Confusion matrices of the original dataset

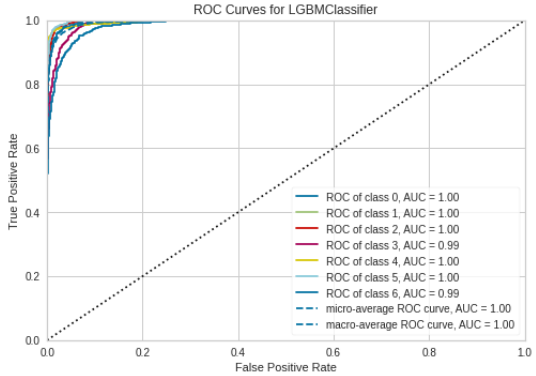


Fig. 13: ROC curves

beans dataset is conducted. In total, 9 experiments have been conducted: the three algorithms are applied on the original data, on the first two PCs dataset and on the transformed dataset with all PCs.

PyCaret [11], a low code machine learning library in python, was used in this project to facilitate the learning process. The dataset was standardized and then split data into a training and a test set where the training set accounts for 0.7 of the data and the test set for 0.3. As Pycaret is a low-code library, it automatically performs a preprocessing of the data (missing values, data imputation, etc.), creates a transformation pipeline, and prepares data for modeling and deployment. Also, Pycaret automatically takes care of all the hyperparameter tuning for each of the three algorithms using a stratified cross-validation technique. For example, for the GBC, the max depth of the trees was increased from 3 to 7 after tuning. For LightGBM, the tuning of the learning rate decreased its value from 0.1 to 0.05 and decreased the number of leaves from 31 to 10.

The macro-averaged F1-score of the three algorithms applied on the three datasets are shown in Figure 11. Starting with the original data, we can see that LightGBM has the highest score (0.932) directly followed by GBC (0.931) and RF(0.908). While LightGBM and GBC had a small increase (of 0.02) after transforming the dataset to all PCs, RF benefited the most from PCA and increased in F1-score to reach 0.92. This result shows the benefits of applying PCA. It eliminates any correlation between variables that may affect the performance of the models. Finally, when testing on two PCs, RF is again the least performer (0.844) while GBC and Light GBM have the same highest score (0.855). Furthermore, all three algorithms obviously decreased in performance in the two PCs dataset which is expected since we are testing on a hugely reduced dimension (from 16 to 2) with almost 18% of explained variance lost.

Added to that, Confusion matrices are generated to identify the classes in which the algorithms had trouble making predictions. The confusion matrices of the three algorithms are shown in Figure 12 and they correspond to the original dataset. The remaining confusion matrices can be found in the notebook. We already know that LightGBM is the best performer on the original dataset followed by GBC and RF respectively. This result is confirmed by calculating the accuracy (sum of the diagonal over the total number of samples) from the confusion matrices. The LightGBM accuracy counts for 92.3%, GBC accuracy correctly classified 91.8% of the values, while RF accuracy is 89.5%. Those results are consistent with what we found previously using F1-score. However, it is always best to use F1-score as the accuracy may falsify results if the data is hugely imbalanced.

If we compare the three confusion matrices we can see similar trends. At first, we can see that class (1), which corresponds to Bombay, is the easiest one to predict. In fact, LightGBM correctly classified all the instances of this classe while RF and GBC only made one error. This may be because we have the lowest amount of instances in this class but due to the fact that Bombay is located far from the other clusters as we have seen in Figure 9. Second, class (3), which corresponds to Dermason, and class (6), which corresponds to Sira, are mostly mistaken for each other by the three algorithms. In fact, 101 instances were incorrectly classified as class (6) by RF while corresponding to class (3) in reality. This may be also be due to those classes having similar shapes and characteristics as seen in Figure 2.

The final metric to be analyzed is the ROC curves. As the LightGBM is the best performer in the original dataset, we plot its ROC curves for the original dataset in Figure 13 while the other algorithms and dataset curves can be found in the notebook. The curves and the AUC values indicate the capability of the algorithm to predict each class. As an

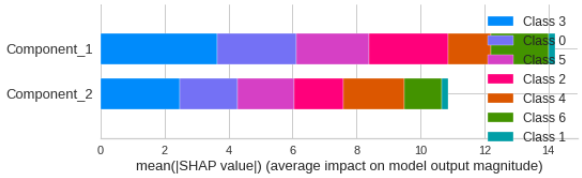


Fig. 14: SHAP symmetry plot

example, LightGBM has an AUC of 1.00 for predicting the class (1) which coincide with the results of the confusion matrix where the model correctly classified all instances of the class (1). Moreover, Class (6) has the smallest AUC (0.98) which is consistent with what we found previously about class (6) being mistaken with class (3). Finally, the macro-averaged AUC is 0.99. Therefore, the LightGBM model successfully distinguishes between the dry beans types.

VII. Explainable AI

In this section, we talk about Explainable AI (XAI) using the SHAP library and how we apply it to our project. In reality, adding explainability and interpretability to a machine learning model makes them trustworthy and understandable by normal people. SHAP library (Shapley Additive explanations) computes the contribution of each feature at the time of prediction

In our experiment, we plotted the SHAP summary plot, a density scatter plot for each feature to identify how much impact each feature has on the model output. We tested this on the validation set of the two PCs dataset while using lightGBM. The plot is shown in Figure 14 where features are sorted by the sum of the SHAP value magnitudes across all samples.

From the plot, we can infer that the first PC has better predictive power than the second feature since it has a larger magnitude. In addition, we can see that the first PC contributes almost equally to the classes except for class (3) and class (1). The first PC contributes the most to the class (3) while it contributes the least to class (1). The second PC has the same trend (contribution to class (3) is the largest and (1) is the lowest) but with less magnitude than the first PC.

The summary plot gives global insights about the impact of our features on the model prediction. Let us now analyze the predictability of the model from a local perspective (single prediction). The plot is known as the force plot and it is shown in Figure 15. For example, we take sample number 40 and let us suppose it is labeled as the class (1). The red feature in the plot increases the value of our prediction while the blue feature reduces the value of our prediction. We can see that the first PC increased the probability of our sample being of class (1) while the second PC decreased the probability of our sample being of class (1).

VIII. Conclusion

The dry bean data is a good dataset to demonstrate the capability of PCA and compare three different ML methods.

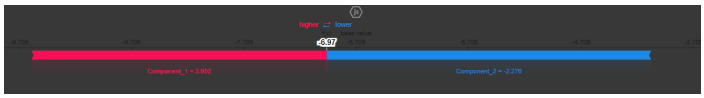


Fig. 15: SHAP single prediction

Our dataset consists of seven types of dry beans characterized by 16 different features.

The first idea consisted of applying the PCA to reduce the dimension of the features. Fortunately, the first two PCs captured more than 80% of the variance in the data and hence they were considered. An analysis of the two PCs was also conducted and supported with PC coefficient plot, biplot and control charts.

Next, Random Forest, LightGBM, and Gradient Boosting Classifier were applied to three datasets: original data, transformed data with all PCs, and data with two PCs. Those algorithms successfully classified different instances of the dataset with a good performance. Also, The results were analyzed and studied using three different metrics: F1-score, Confusion matrix, and ROC curves. LightGBM appeared to be the best performer in the original and transformed dataset. Yet, all three algorithms decreased in performance in the two PCs dataset as expected but overall the three methods yielded great results and successfully distinguished between different classes.

Lastly, the powerful SHAP library was used to interpret and explain the behavior of our models and the impact of each PC at the time of prediction from a local perspective and a global perspective.

Acknowledgement

This project is part of the INSE6220 course. I would like to express my thanks to Professor **Abdessamad Ben Hamza** who instructed this course and simplified concepts. Also, I take the opportunity to thank my supervisor Professor **Nizar Bouguila**.

References

[1] ÖNDER, M., ATEŞ, M. K., KAHRAMAN, A., et al. The problems and suggestions to dry bean farming in Konya region. TABAD, Tarım Bilimleri Araştırma Dergisi, 2012, vol. 5, no 1, p. 143-148.

[2] BENOS, Lefteris, TAGARAKIS, Aristotelis C., DOLIAS, Georgios, et al. Machine learning in agriculture: A comprehensive updated review. Sensors, 2021, vol. 21, no 11, p. 3758.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] LEVER, Jake, KRZYWINSKI, Martin, et ALTMAN, Naomi. Points of significance: Principal component analysis. Nature methods, 2017, vol. 14, no 7, p. 641-643.

[5] A. Ben Hamza, Advanced Statistical Approaches to Quality, unpublished.

[6] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189–1232, 2001.

[7] KE, Guolin, MENG, Qi, FINLEY, Thomas, et al. Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems, 2017, vol. 30.

[8] KOKLU, Murat et OZKAN, Ilker Ali. Multiclass classification of dry beans using computer vision and machine learning techniques. Computers and Electronics in Agriculture, 2020, vol. 174, p. 105507.

[9] MURPHY, Kevin P. Machine learning: a probabilistic perspective. MIT press, 2012.
[10] ZHANG, Dell, WANG, Jun, et ZHAO, Xiaoxue. Estimating the uncertainty of average F1 scores. In : Proceedings of the 2015 International conference on the theory of information retrieval. 2015. p. 317-320.
[11] Moez Ali. PyCaret: An open source, low-code machine learning library in Python, 2020, https://www.pycaret.org