MODERN CLASSIFICATION OF DRY BEANS FOR ENHANCED QUALITY CONTROL AND MARKET SEGMENTATION USING MACHINE LEARNING/ARTIFICIAL INTELLIGENCE

TABLE OF CONTENTS

Contents

ABSTRACT

A dataset of dry beans containing 13,611 instances of the geometric features of 7 classes of beans was analyzed. Both classification and clustering were used to group the beans. PCA was applied for dimensionality reduction. On google colab, the accuracy ranged from 89.52-93.13% except for AdaBoost which gave the lowest accuracy of 49.86% while SVM had the highest accuracy. Cloud-based automated Machine Learning result showed that XGB classifier was the best model with the greatest accuracy. Different clustering models showed results of either 5 or 3 clusters of beans instead of 7.

INTRODUCTION

In the global agricultural sector, ensuring the quality of staple crops such as dry beans, a primary dietary component for millions, is crucial. Traditional bean classification relies heavily on manual sorting, which is inefficient and error-prone, leading to inconsistencies that affect product quality and safety (Koklu, Ozkan 2020). This project addresses these challenges by leveraging Machine Learning (ML) and Artificial Intelligence (AI) to develop an automated bean classification system. By integrating advanced AI algorithms, this initiative aims to significantly enhance the accuracy and speed of bean classification, thereby improving quality control processes in agriculture. This system is expected to benefit farmers, food processors, and consumers by revolutionizing agricultural practices and ensuring higher standards of efficiency, accuracy, and quality in food production (AlZubi, Galyna 2023).

PROJECT PROPOSAL

Brief Description of the Problem

Within the agricultural industry, accurate classification of dry beans is important in maintaining product quality standards, product differentiation, optimizing market segmentation, and maximizing profits. However, conventional classification methods are labor-intensive, error-prone, and not scalable. To overcome these challenges, this project proposes the use of Machine Learning (ML) and Artificial Intelligence (AI) to automate and enhance the classification and clustering of dry beans. This initiative will focus on differentiating seven registered varieties of dry beans, which are often challenging to classify due to their similar physical features.

Objectives:

- **Develop AI Models**: To automate the classification of dry beans based on their physical attributes and intrinsic characteristics using advanced AI models.
- **Implement Clustering Algorithms**: To group similar batches of dry beans, facilitating targeted marketing strategies and enhanced product differentiation.
- **Evaluate Model Performance**: To ensure the accuracy & reliability of the AI-driven models in real-world settings.
- **Comparative Analysis**: Utilize cloud service alongside manual Python implementations for classification to compare efficiency and effectiveness.

Dataset Description

The dataset, sourced from a public Kaggle repository, comprises 13,611 images of dry beans segmented into seven classes, captured using high-resolution imaging and processed for feature extraction, resulting in 17 descriptive features. Each sample is meticulously labeled according to one of the seven bean varieties (Kaggle, 2024). The dataset includes features like area, perimeter, and various shape factors which describe the physical dimensions and aesthetic characteristics of the beans. The definition of each feature of the dataset is shown in table 1:

Table 1: Dataset Features

S/N	Feature	Definition
1	Area	This refers to the area of a bean grain and the total number of pixels within its boundaries (in mm^2).
2	Perimeter	Also known as the circumference of a Bean grain. It is the length of the boundary of a grain (in mm).
3	MajorAxisLength	The measure of the longest line passing through the center of the grain, measured (in mm).
4	MinorAxisLength	The longest line that can be drawn from the center of a bean grain while standing perpendicularly to the main axis (in mm).
5	AspectRatio	It is the ratio of the length of the major axis to the length of the minor axis. It provides an indication of the elongation or roundness of a dry bean grain.
6	Essentricity	It measures how much the shape of a dry bean grain deviates from a perfect circle. It is a dimensionless quantity between 0 and 1, where 0 represents a perfect circle and 1 represents an extremely elongated shape.
7	ConvexArea	Number of pixels or the total area of the smallest convex polygon that encloses a bean grain (mm^2).
8	EquivDiameter	It is the diameter of a circle with the same area as the dry bean grain. It provides a measure of the size of the grain (in mm).
9	Extent	It is the ratio of the area of the dry bean grain to the area of the bounding box that encloses it. It describes how much the shape of the grain fills the bounding box.
10	Solidity	Also known as convexity is the ratio of the pixels in the convex shell/hull to those found in beans. It indicates how solid or compact the grain shape is.
11	Roundness	It measures how closely the shape of the dry bean grain resembles a perfect circle. It is calculated using the formula: $4\pi \times Area/Perimeter^2$
12	Compactness	Measures how compact the dry bean grain is: Ed/L.
13	ShapeFactor1	
14	ShapeFactor2	It describes the elongation or roundness of the dry bean grain.
15	ShapeFactor3	It provides information about the roundness and compactness of the grain shape.
16	ShapeFactor4	It provides information about the solidity and compactness of the grain shape.
17	Class	Different varieties of beans. It may be Seker, Barbunya, Bombay, Cali, Dermosan, Horoz or Sira.

Machine Learning/AI Techniques

The following AI techniques will be employed to address the problem:

- **Multiclass Support Vector Machine (SVM):** Employed for its robustness in handling multi-class scenarios, aiming to find the optimal hyperplane that maximizes the margin between different classes.

- **Multilayer Perceptron (MLP)**: MLP is a type of artificial neural network that consists of multiple layers of nodes, each connected to the next layer. Utilized for its capability in pattern recognition, essential for the nuanced differentiation required in this project.
- **Convolutional Neural Network (CNN)**: CNNs excel in image classification tasks as they can autonomously extract features directly from raw pixel data.
- **Additional Classification Models:** Logistic Regression, Random Forest, AdaBoost, XGBoost, K-Nearest Neighbors, and Gaussian Naive Bayes will be evaluated to ensure a robust comparative analysis.
- **Clustering Techniques:** K-means, DBSCAN, and hierarchical clustering will be applied to discern naturally occurring groups within the bean varieties, enhancing batch-specific strategies.

Tools: Jupyter notebook on Google Colab would be used for both classification and clustering while Microsoft Azure Machine Learning Cloud service will also be used for Classification.

DATA PRE-PROCESSING AND VISUALIZATION

Spelling error of one of the feature titles was corrected. Out of the 13,611 instances, there were no missing data but there were 68 duplicates which were later dropped. The distribution of the classes is shown in figure 1. It can also be observed that there is class imbalance which made classification a bit difficult with some errors.

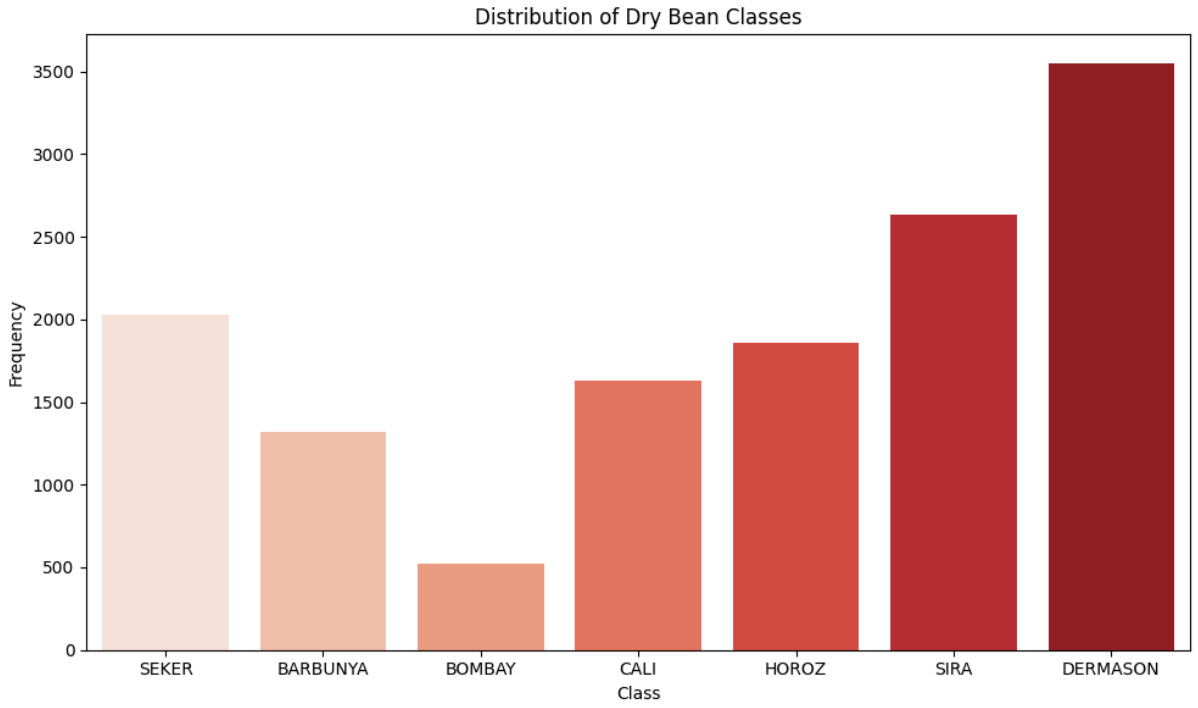


Figure 1: Distribution of Dry Bean Classes

The histogram plots (figure 2) of the numerical features showed that Compactness, ShapeFactor1 and ShapeFactor3 are normally distributed while the rest are either skewed to the left or right.

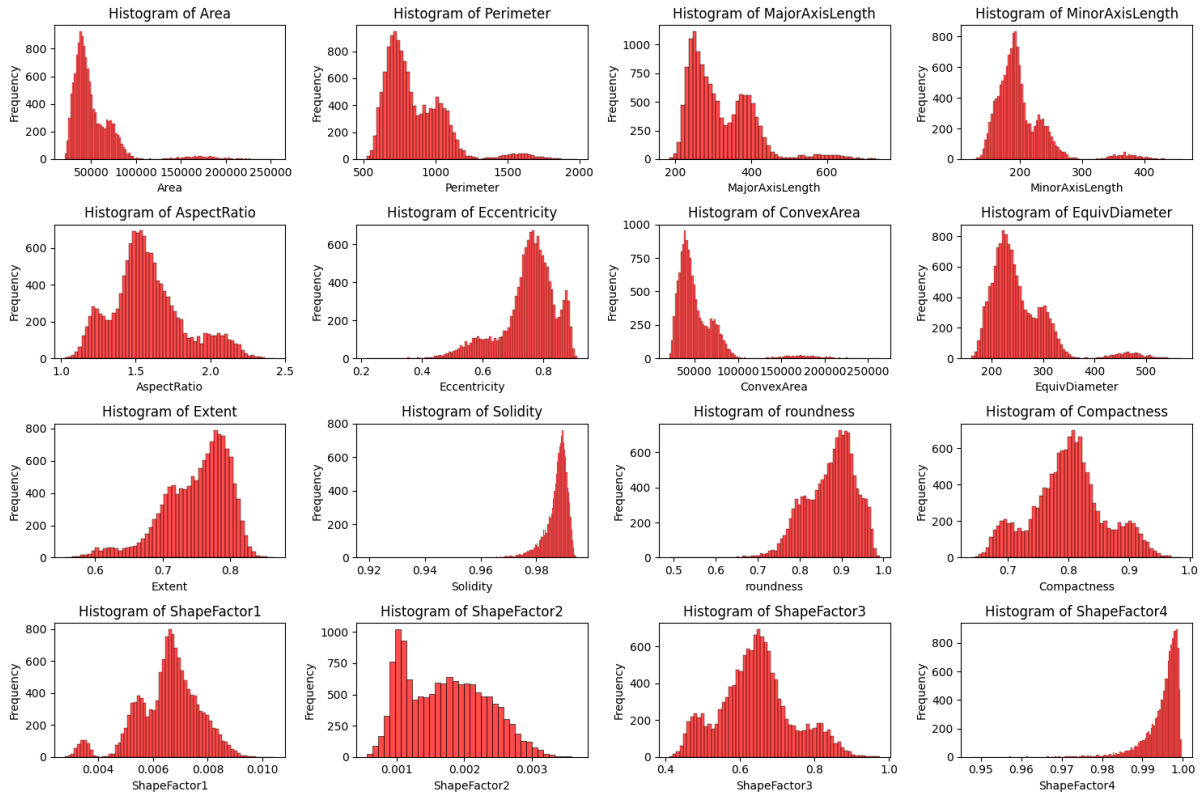


Figure 2: Histogram Plots of the numerical features of Dry Beans

This was further investigated to see whether there is presence of outliers by boxplots. As shown in figure 3, only Shapefactor2 does not have outliers.

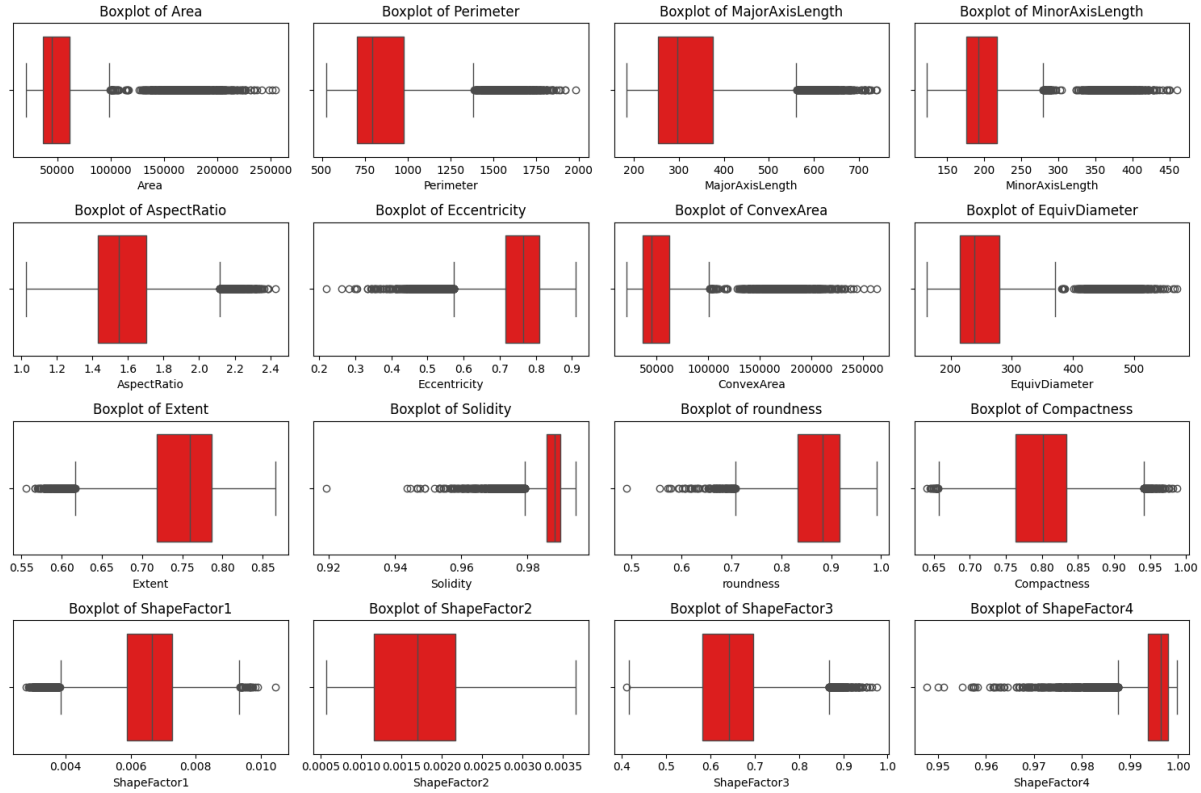


Figure 3: Boxplots of Numerical features of Dry Beans

Because of the presence of a lot of outliers in the data, robust scaler was adopted for scaling and normalising the data. Robust scaler was chosen because it uses the median and interquartile range (IQR) for scaling, rather than the mean and standard deviation, which makes it less sensitive to outliers. It ensured that all features contribute equally to model predictions without being skewed by extreme values (Izonin et al. 2022). 'Class', the only categorical feature was converted to numerical using label encoder. As shown in the correlation matrix in figure 4, Area, Perimeter, MajorAxisLength, MinorAxisLength, ConvexArea and EquivDiameter are highly correlated with each other. Among the features, ShapeFactor1, ShapeFactor2, Roundness, Solidity are highly correlated with the Class while others are either fairly positively correlated or negatively correlated with Class. This can also be seen from the Pairplot in Figure 10.

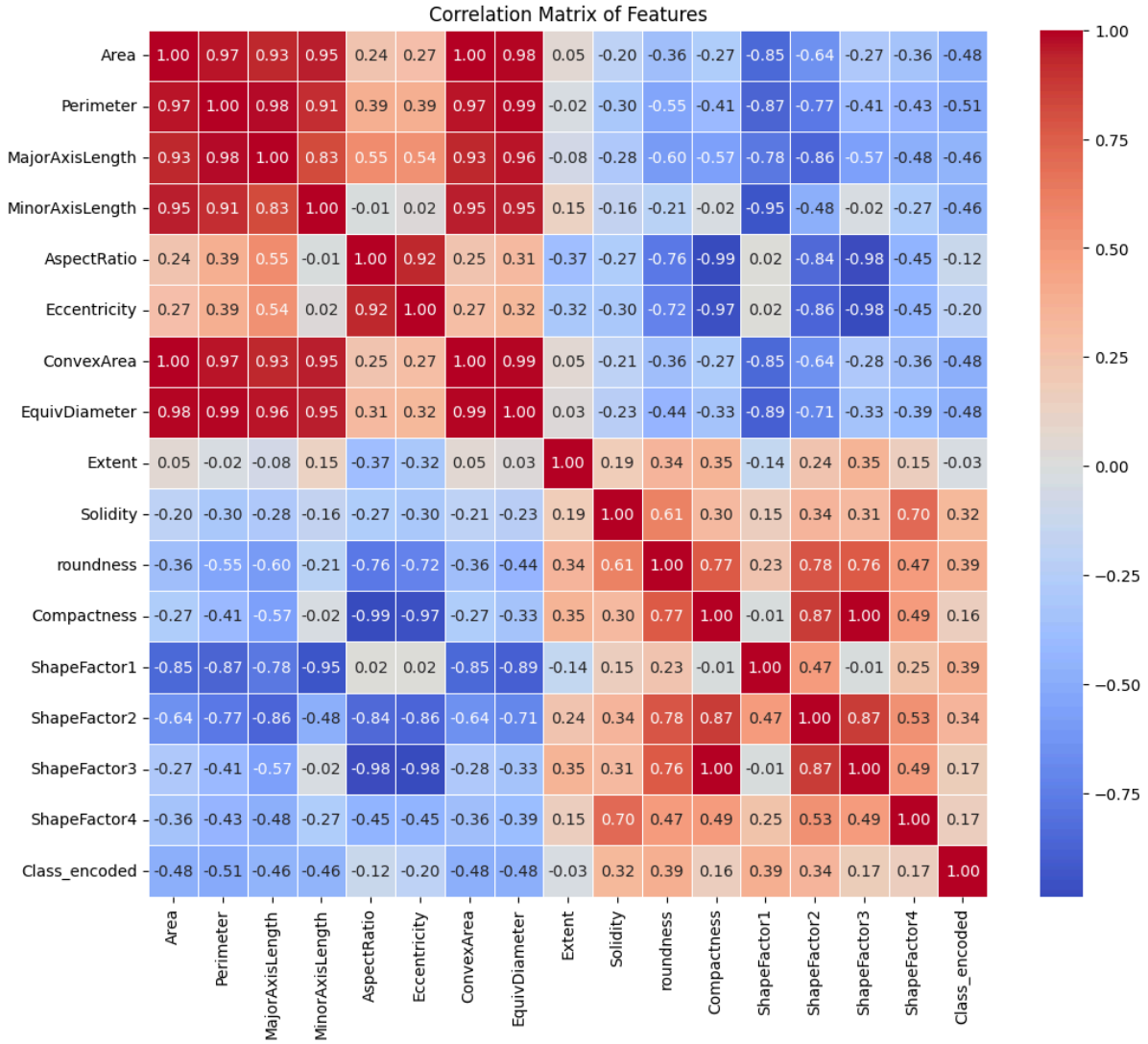


Figure 4: Correlation Matrix of Features

All the previous visualizations showed that Area and ConvexArea are the same hence, ConvexArea feature was dropped to reduce the risk of overfitting. The relationship between Class and some selected features are shown by the Violin Plots in Figure 5-9 to see their differences.

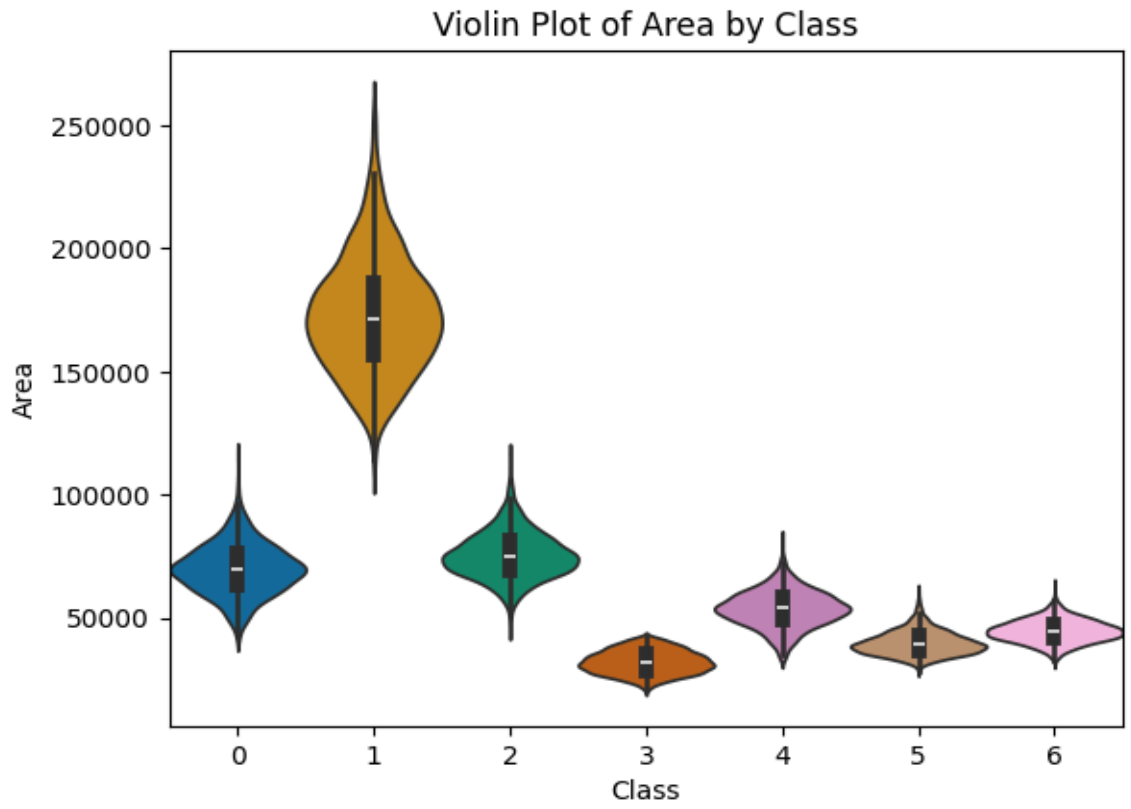


Figure 5: Violin Plot of Area by Class

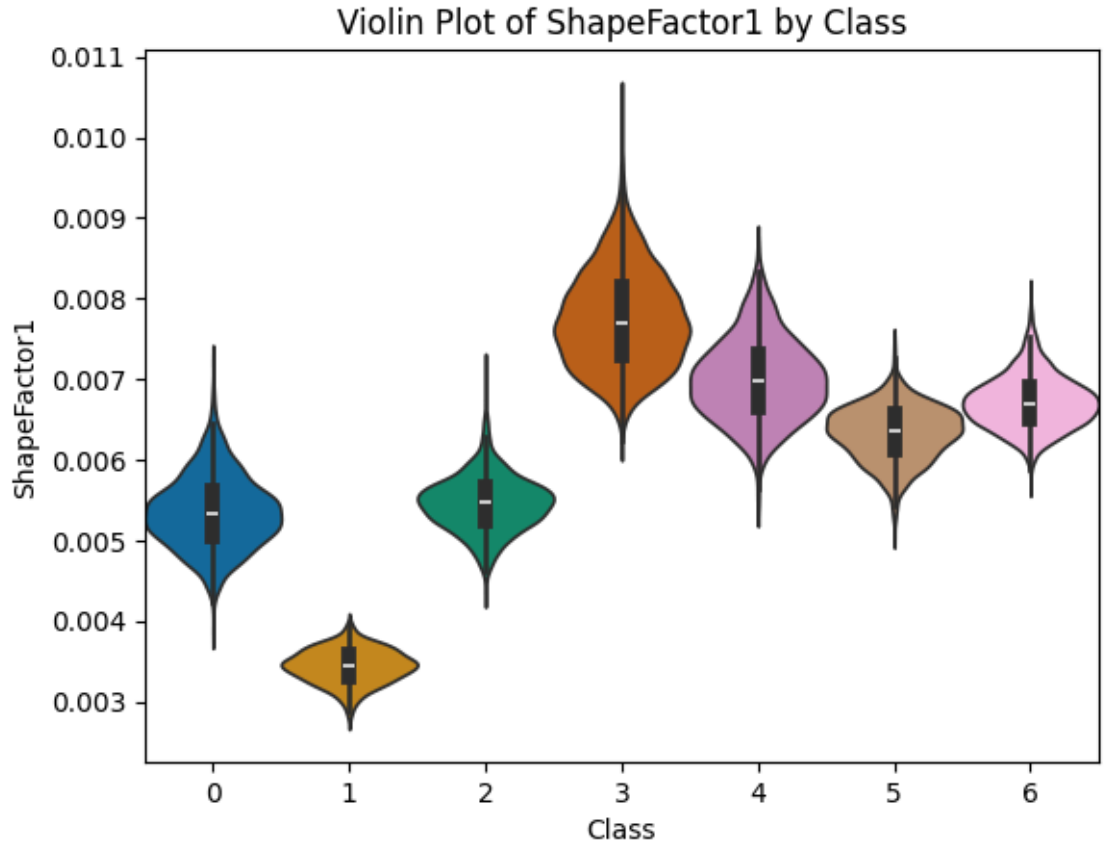


Figure 6: Violin Plot of ShapeFactor1 by Class

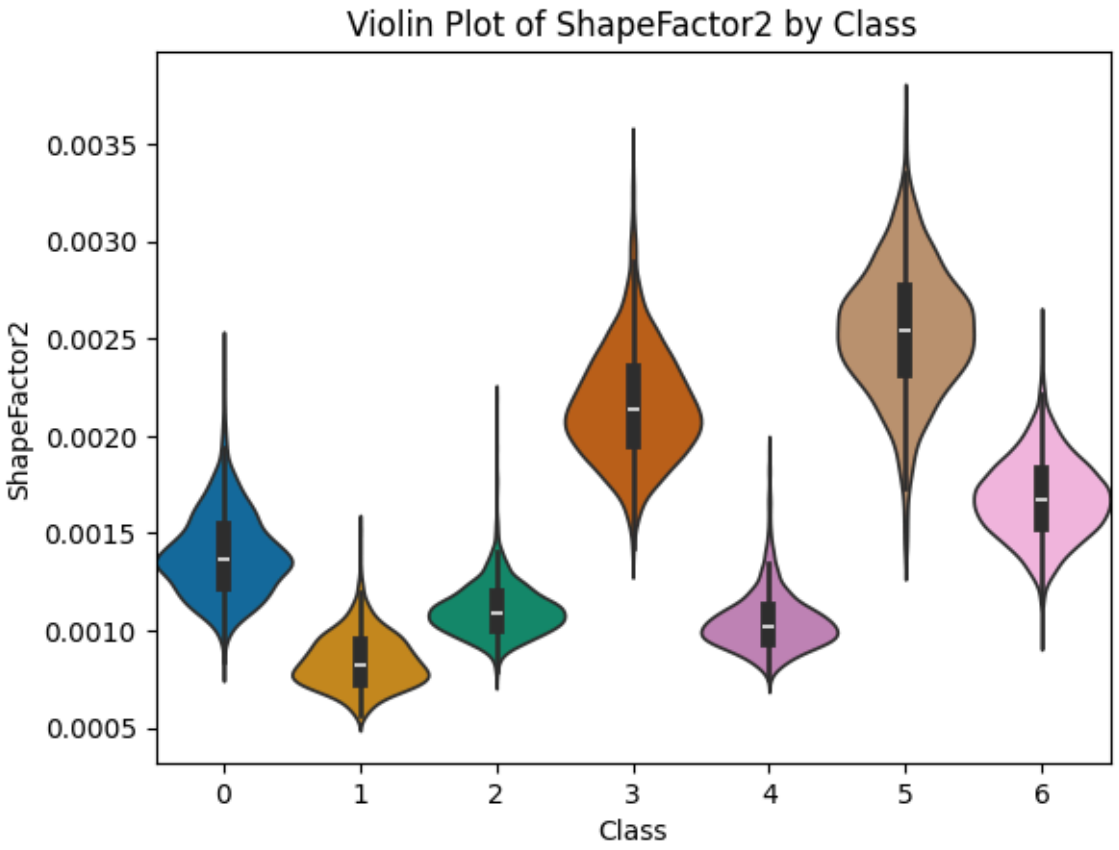


Figure 7: Violin Plot of ShapeFactor2 by Class

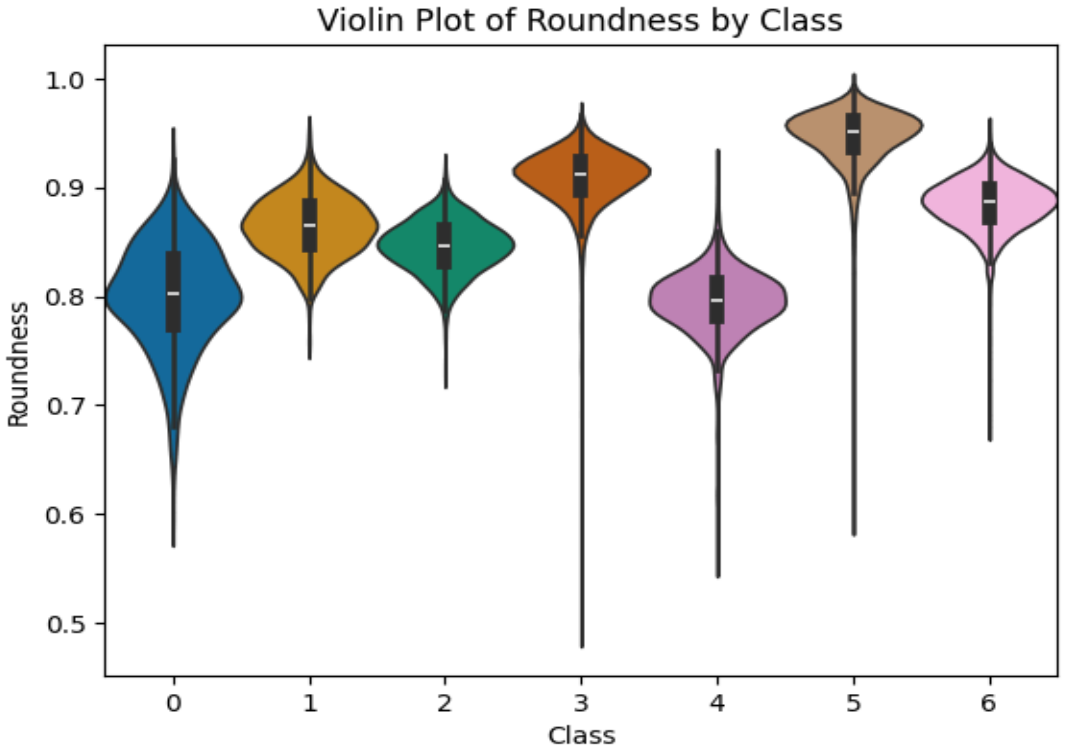


Figure 8: Violin Plot of Roundness by Class

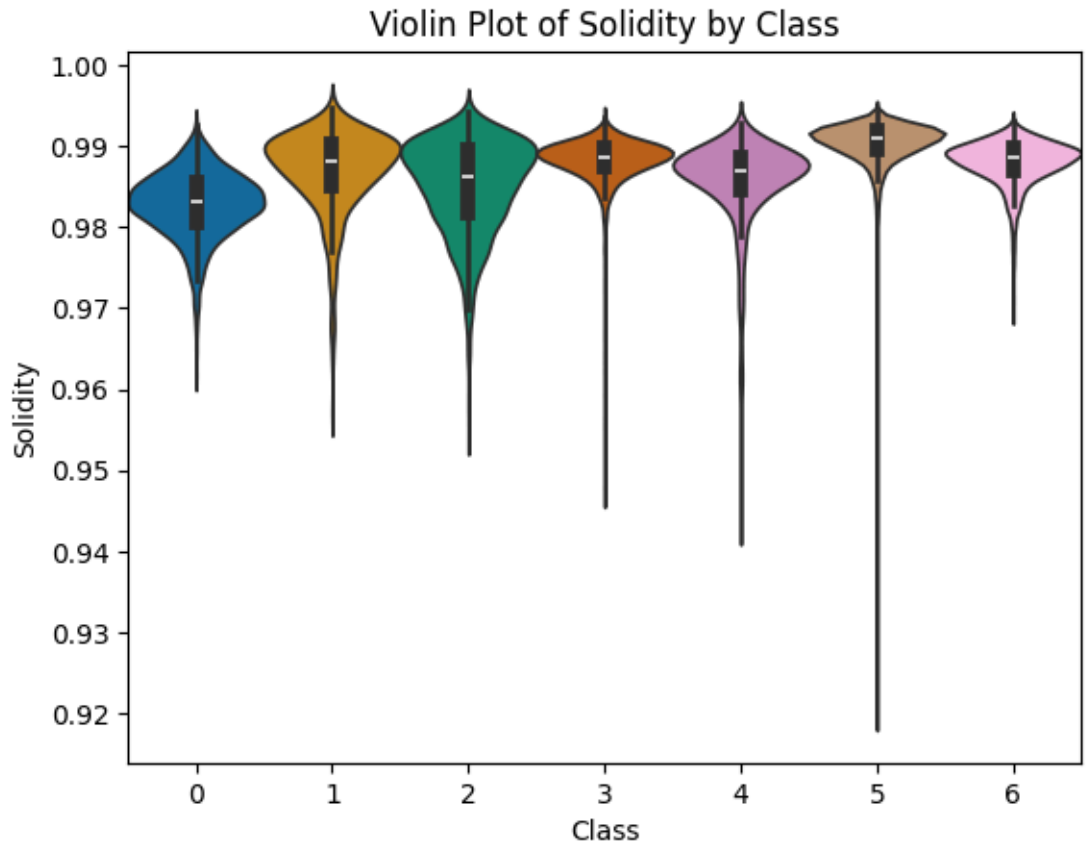


Figure 9: Violin Plot of Solidity by Class

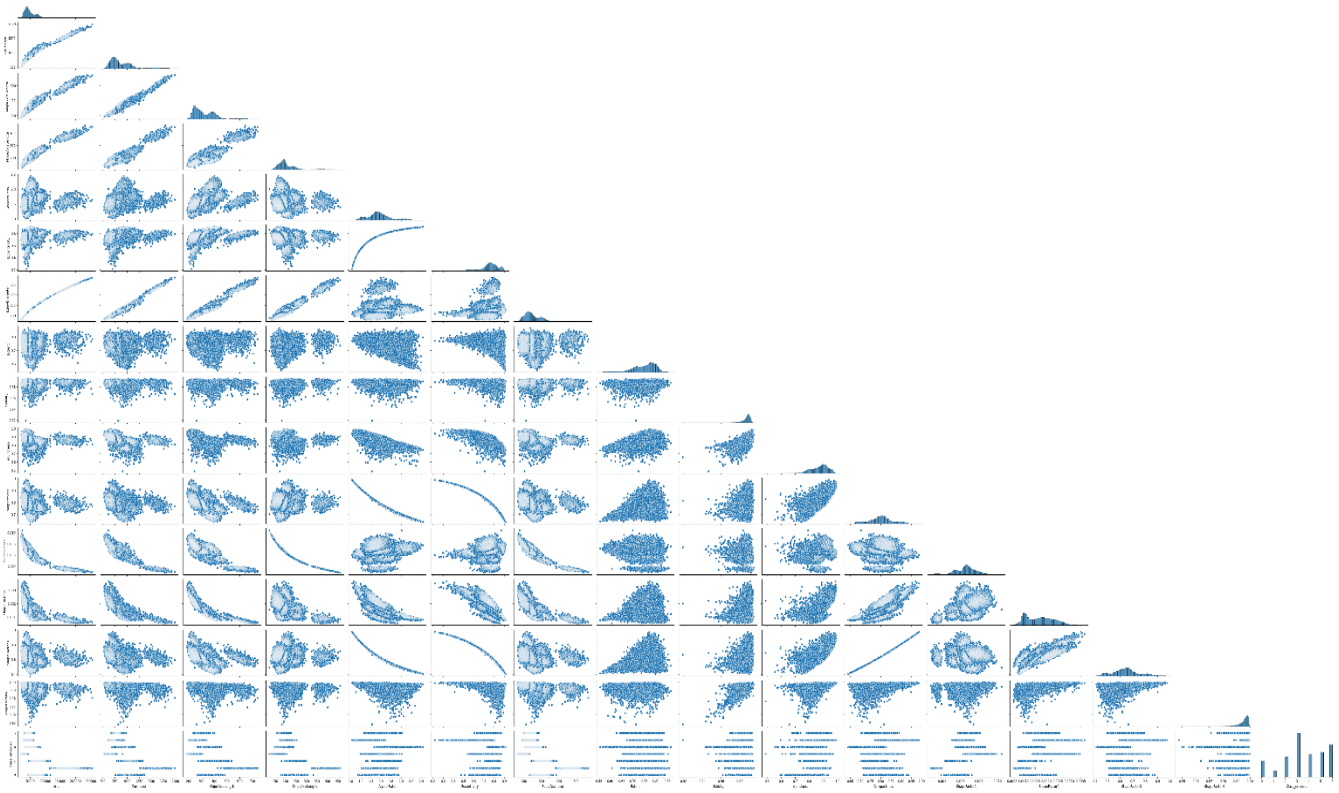


Figure 10: Pairplot of Features

Feature Importance & Dimensionality Reduction

Feature importance was evaluated using a RandomForest Classifier to pinpoint less significant features, as depicted in Figure 11. This strategy reduces computational complexity, curtails the risk of overfitting, and facilitates easier visualization (fig. 10 looked too clumsy). However, a potential downside is the loss of critical information, which may marginally decrease model accuracy. Initially, using the top 7 features for classification yielded an accuracy of 90.00%, compared to 92.17% when all 15 features were utilized. To address the dip in accuracy while still managing feature dimensions, Principal Component Analysis (PCA) was implemented. PCA was configured to preserve at least 99.9% of the original data's variance, ultimately reducing the number of components to 8. This adjustment improved accuracy to 92.52%. Subsequent modeling leveraged PCA for its efficiency and improved performance.

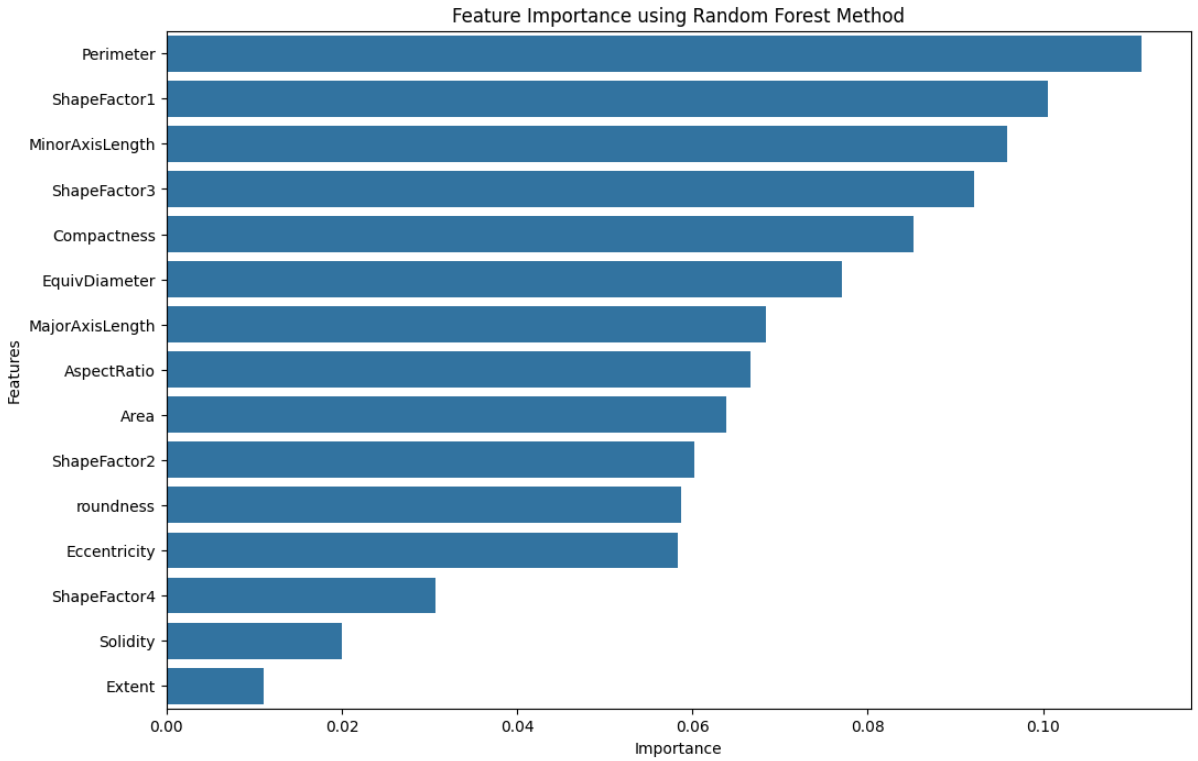


Figure 11: Feature Importance Graph using Random Forest

Implementation & Evaluation:

Classification

The dataset was split into Train and Test sets in the ratio 70:30.

1. **Support Vector machines (SVM)**

SVM are highly versatile, serving both as classifiers and regressors. Renowned for their robustness and flexibility, SVMs operate on the principle of maximizing the margin between classes by constructing an optimal hyperplane. This is particularly effective as it improves the model's generalization capabilities. However, in scenarios where the data is not linearly separable, SVMs employ various kernel functions, such as polynomial or radial basis function (RBF), to transform the data into a higher dimension where a hyperplane can be effectively drawn. A detailed discussion of these mechanisms is provided by Gholami and Fakhari (2017) (Gholami, Fakhari 2017). To identify the best parameters for the SVM, a comprehensive Grid Search was conducted. This method systematically explores a range of potential parameter combinations to determine the most effective settings. The training process was streamlined using a pipeline that integrates scaling, PCA for

dimensionality reduction, and SVM training. This approach ensures that each transformation is applied consistently across both the training and testing phases, thereby preventing data leakage and maintaining the integrity of the model evaluation. A similar method was adopted for the subsequent models. The final model parameters and the accuracy achieved are detailed in figure 12.

```
Fitting 5 folds for each of 48 candidates, totalling 240 fits
Best parameters found: {'svm__C': 100, 'svm__gamma': 0.01, 'svm__kernel': 'rbf'}

Best cross-validation score: 0.93
Accuracy: 0.9313315284272705

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.94      0.94       396
           1       1.00      1.00      1.00       161
           2       0.94      0.95      0.94       473
           3       0.92      0.93      0.92      1065
           4       0.96      0.95      0.96       553
           5       0.96      0.95      0.96       618
           6       0.88      0.88      0.88       797

    accuracy                           0.93      4063
   macro avg       0.94      0.94      0.94      4063
weighted avg       0.93      0.93      0.93      4063
```

Figure 12: Results of Classification using SVM

As shown in fig. 12 and 13, the SVM model demonstrated excellent performance, achieving a cross-validation score of 0.93 and an overall test accuracy of 93.13%. The detailed classification report showed high precision, recall, and F1-scores across all seven bean varieties, with outstanding performance in Class 1 (100% across all metrics) and consistently high scores above 88% in other classes. This robust accuracy underscores the SVM's capability to effectively differentiate between the bean types, ensuring reliable and precise classification suitable for practical applications in agricultural quality control.

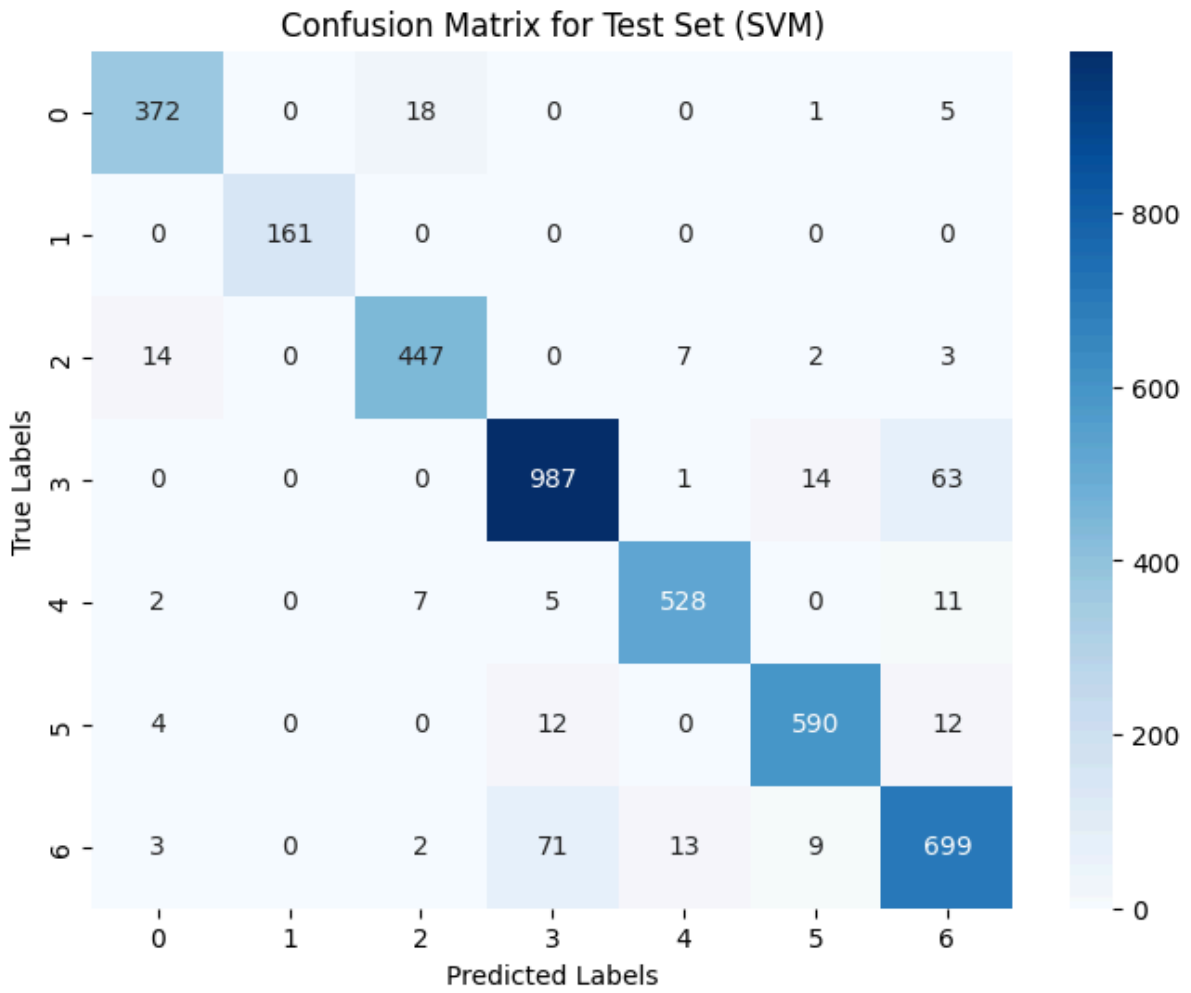


Figure 13: Confusion Matrix (SVM)

2. Multilayer Perceptron (MLP) also known as Artificial Neural Network (ANN)

It is a deep learning technique. The MLP is trained using backpropagation with gradient descent. This method iteratively adjusts the weights of the network to minimize the prediction error, measured by a loss function, over several training epochs. Configured with 4 hidden layers with respective sizes of 25, 18, 10, and 5 neurons. MLP facilitates a deep learning approach, enabling the model to learn complex patterns from the data at multiple abstraction levels. Two architectures of MLP were tried as shown in table 2a. Architecture 2 parameters were because of optimization of other parameters, keeping the hidden layer sizes fixed, done using Grid Search. This is to enhance performance, reduce overfitting, or accelerate convergence.

Table 2a: MLP Architectures

Parameters	Architecture 1	Architecture 2
Activation Function	ReLU (Rectified Linear Activation). Helps prevent the vanishing gradient problem and speeds up training.	Tanh
Optimizer (Solver)	Adam, an adaptive learning rate optimizer, known for its effectiveness in handling large datasets and converging quickly.	Adam
Training Iterations (max_iter)	300. This is to ensure the network has sufficient opportunity to learn and adjust its weights adequately before terminating training.	200
Training Method	Backpropagation with gradient descent.	
learning_rate_init	0.001	0.001

Architecture 1 gave an accuracy of 92.96% (fig. 14) while Architecture 2 gave 92.49% (fig. 15). The confusion matrices of both are shown in figure 16.

```
Accuracy: 0.9296086635491017
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.95      0.93       396
           1       1.00      1.00      1.00       161
           2       0.96      0.93      0.95       473
           3       0.92      0.93      0.93      1065
           4       0.94      0.97      0.95       553
           5       0.94      0.96      0.95       618
           6       0.89      0.86      0.88       797

    accuracy                           0.93      4063
   macro avg       0.94      0.94      0.94      4063
weighted avg       0.93      0.93      0.93      4063
```

Fig. 14: Classification Report (MLP- Architecture 1)

```
Best parameters set:
{'mlp__activation': 'tanh', 'mlp__learning_rate_init': 0.001, 'mlp__max_iter': 200, 'mlp__solver': 'adam'}
Best cross-validation score: 0.93
Test Accuracy: 0.9249323160226434
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.93      0.93       396
           1       1.00      1.00      1.00       161
           2       0.94      0.94      0.94       473
           3       0.93      0.91      0.92      1065
           4       0.95      0.95      0.95       553
           5       0.95      0.96      0.95       618
           6       0.86      0.87      0.86       797

    accuracy                           0.92      4063
   macro avg       0.94      0.94      0.94      4063
weighted avg       0.93      0.92      0.92      4063
```

Fig. 15: Classification Report (MLP- Architecture 2- Optimized)

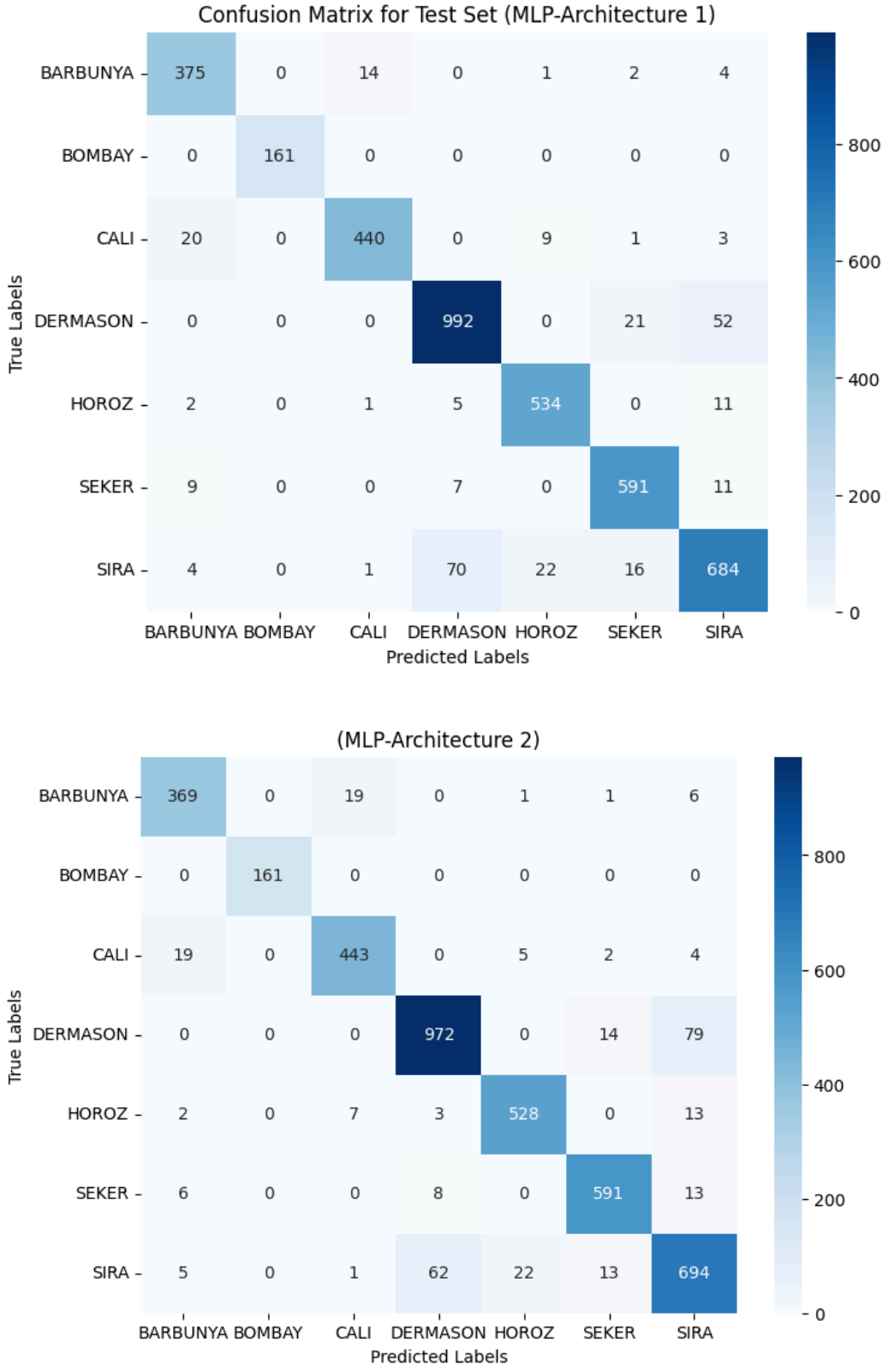


Figure 15: a) Confusion Matrix (MLP-Architecture 1), b) (Architecture 2)

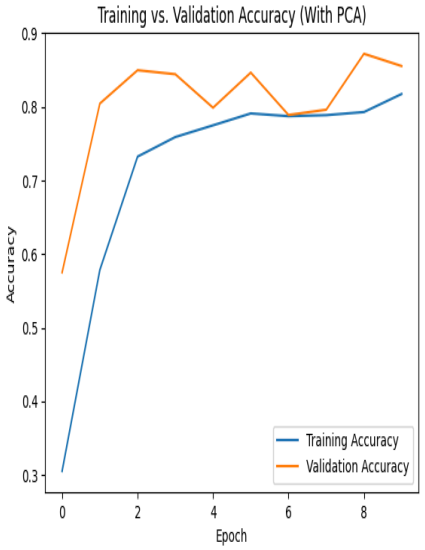
SVM slightly outperformed MLP with an accuracy of 93.13% vs 92.49%. SVM was optimized with a radial basis function kernel, showing robust precision across most bean classes, particularly

struggling with class 6. MLP, configured with a `tanh` activation function and the `adam` solver, demonstrated similar challenges. The choice of `tanh` likely aided in faster convergence due to its effectiveness in handling scaled inputs, while the low learning rate and `adam` solver helped in avoiding quick convergence to suboptimal solutions. Despite MLP's slightly lower accuracy, its adaptability and efficiency in training make it a valuable alternative, especially where model flexibility is crucial. Both models exhibited high overall accuracy, making them suitable for practical applications, though SVM showed a slight edge in performance.

3. Convolutional Neural Network (CNN)

CNNs with 1D convolution layers expect input data to have 3 dimensions specifically for Conv1D: samples, time steps, features. So, the data previously scaled data is reshaped accordingly. Each data sample is reshaped to (length_of_sample, 1), where length_of_sample is the number of features per instance. The 1 signifies a single channel, like how image data might have multiple channels (e.g., RGB). 1D CNN architecture for classification was defined using TensorFlow and Keras. A sequential model was used because it allows for stacking of layers in a sequence where each layer has weights connected only to the next layer. The model was tested with and without PCA. The architecture used and the training process are shown in figure 16 (with PCA) and 17 (Without PCA).

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 8, 32)             128

 conv1d_1 (Conv1D)           (None, 8, 64)             6208

 conv1d_2 (Conv1D)           (None, 8, 128)            24704

 max_pooling1d (MaxPooling1  (None, 4, 128)            0
 D)

 dropout (Dropout)           (None, 4, 128)            0

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 256)               131328

 dense_1 (Dense)             (None, 512)               131584

 dense_2 (Dense)             (None, 7)                 3591

=================================================================
Total params: 297543 (1.14 MB)
Trainable params: 297543 (1.14 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```



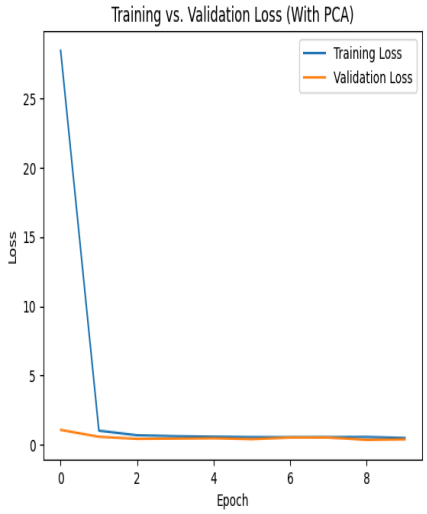


Figure 16. The architecture (left), training process (middle) & of the training vs validation loss of the first 1D CNN with PCA.

Model: "sequential_2"

Layer (type)	Output Shape	Param #
conv1d_4 (Conv1D)	(None, 15, 32)	128
conv1d_5 (Conv1D)	(None, 15, 64)	6208
conv1d_6 (Conv1D)	(None, 15, 128)	24704
max_pooling1d_1 (MaxPooling1D)	(None, 8, 128)	0
dropout_1 (Dropout)	(None, 8, 128)	0
flatten_1 (Flatten)	(None, 1024)	0
dense_3 (Dense)	(None, 256)	262400
dense_4 (Dense)	(None, 512)	131584
dense_5 (Dense)	(None, 7)	3591

Total params: 428615 (1.64 MB)
Trainable params: 428615 (1.64 MB)
Non-trainable params: 0 (0.00 Byte)

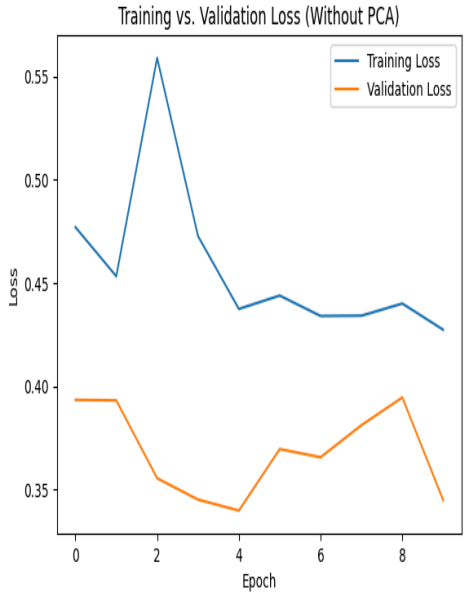
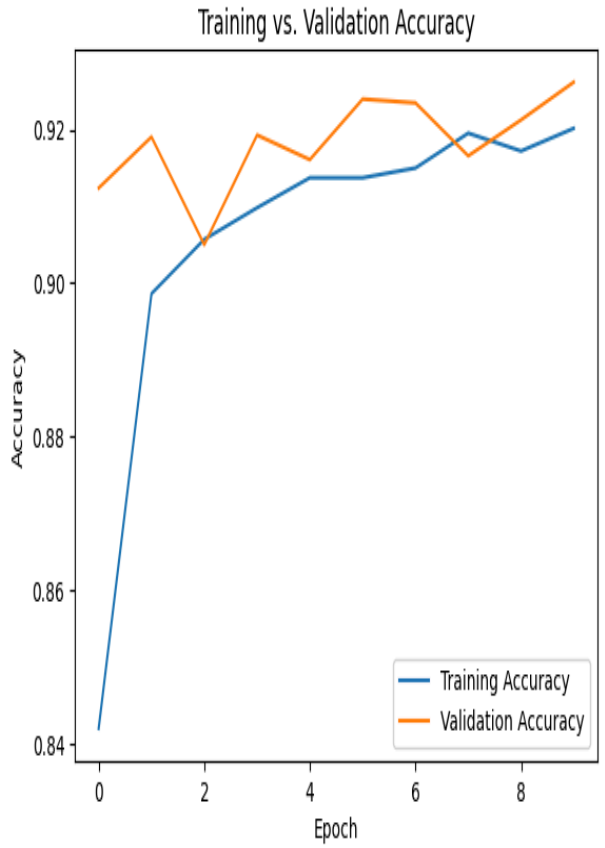


Figure 17: The architecture (left), training process (middle) & of the training vs validation loss of the first 1D CNN without PCA.

At the first attempt with a dropout rate of 0.5, accuracy was 92.30% (With PCA) and 87.92% (Without PCA) meaning that PCA helped in improving the accuracy. We moved on to further tune the parameters with PCA. Increasing the dropout rate at the second attempt to 0.7 gave a slight increase in accuracy of 92.62% (Fig. 18).



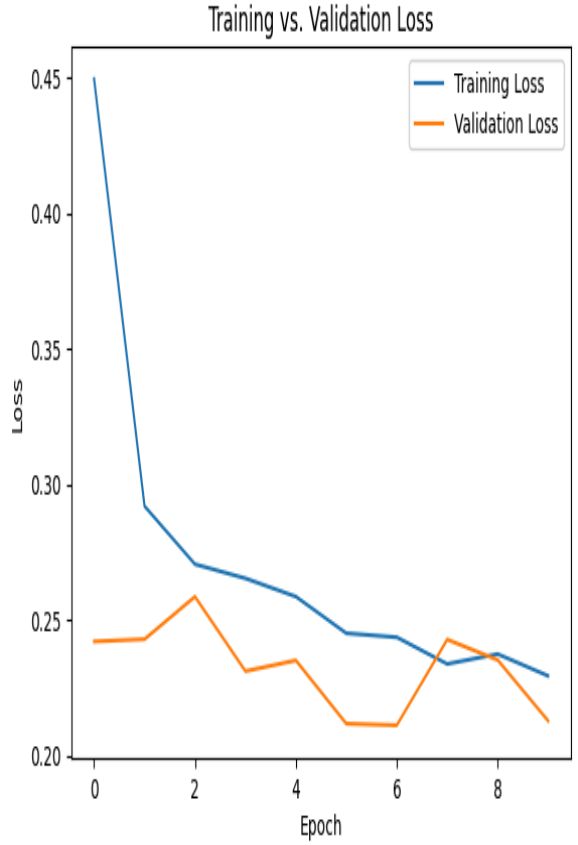


Figure 18: The training process (left) & of the training vs validation loss of the 2nd 1D CNN attempt (right)

The 3rd attempt was the increase in the convolutional layers from 3 to 4 at a dropout of 0.7. Accuracy was 92.74% as shown in fig. 19.

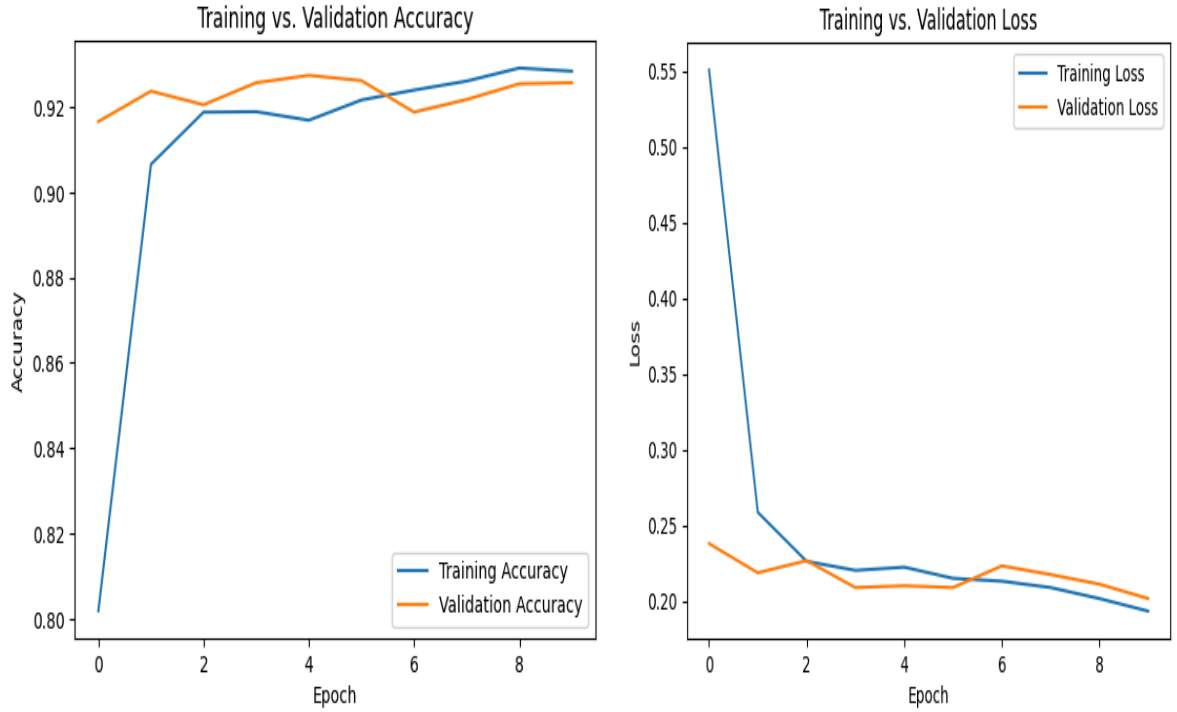


Figure 18: The training process (left) & of the training vs validation loss of the 3rd 1D CNN attempt (right)

Table 2b: Comparison between the Accuracy of SVM, MLP & CNN

Model	Configuration	Accuracy
SVM	C=100, gamma=0.01, kernel='rbf'	93.13%
MLP	tanh activation, learning rate=0.001, max_iter=200, adam, 4 hidden layers of 25, 18, 10,5 neurons, kernel= 'poly'	92.49%
1D CNN	4 convolutional layers, dropout=0.7, 2 Dense layers (256, 512 units), Output Dense layer (7 units).	92.74%

Table 2b shows the comparison in the accuracy of the previous models. The CNN model's architecture, featuring multiple convolutional and dense layers plus a high dropout rate, effectively manages complex patterns and mitigates overfitting. In comparison, SVM employs straightforward decision boundaries, while the MLP utilizes a multi-layer perceptron approach. Each of the models exhibits distinct advantages in accuracy and design complexity.

Clustering

a) KMEANS Clustering

Elbow method was used to determine the optimal number of clusters and the result obtained was 5 (Fig. 19a). Which was the point at which the line curved.

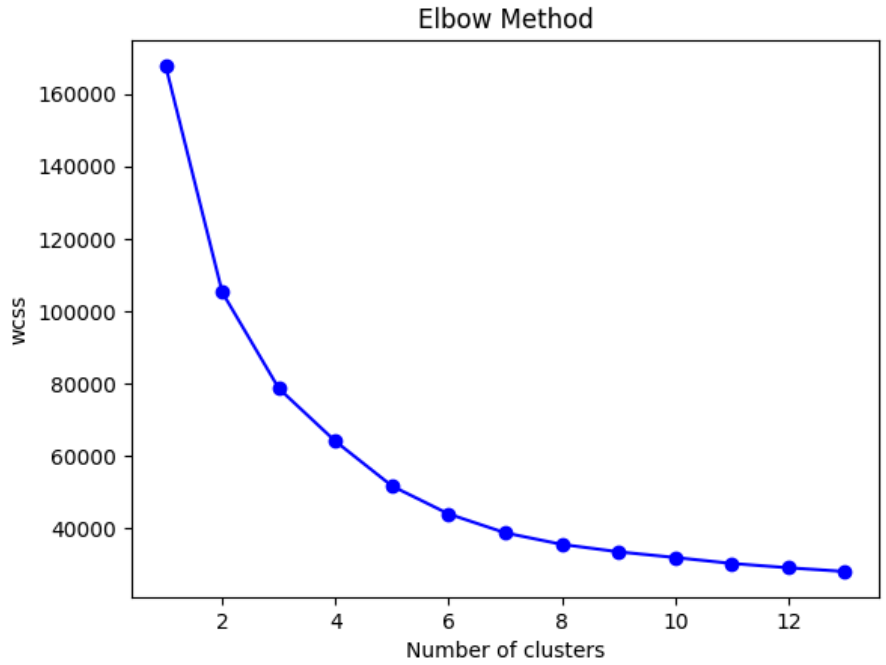


Fig. 19a: KMeans Elbow Graph

The model was set to 5 clusters and the visualization is shown in figure 20. The accuracy of this result is low because the expected number of bean cluster was 7. The evaluation results are shown in table 2c.

Table 2c: KMeans Clustering Evaluation result

Metric	Value	Interpretation
Silhouette Score	0.36	Moderate cluster separation & cohesion
Davies-Bouldin Index	1.02	Clusters are differentiated but not optimal
Calinski-Harabasz Index	7608.15	High cluster density 7 good separation

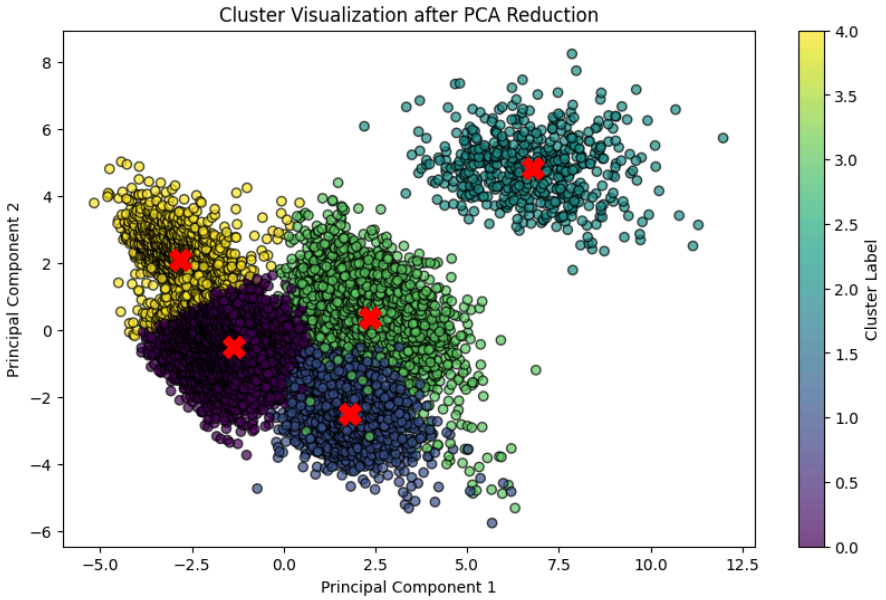


Fig. 20: KMeans Bean Clusters

b) Hierarchical Clustering

Agglomerative hierarchical clustering is a flexible method that starts with each data point as its own cluster and progressively merges them based on similarity, without needing predefined parameters (Bouguettaya et al. 2015). A dendogram was used to determine the optimum number of clusters. It also shows 5 distinct clusters as shown in fig. 21. The evaluation results are shown in table 2d.

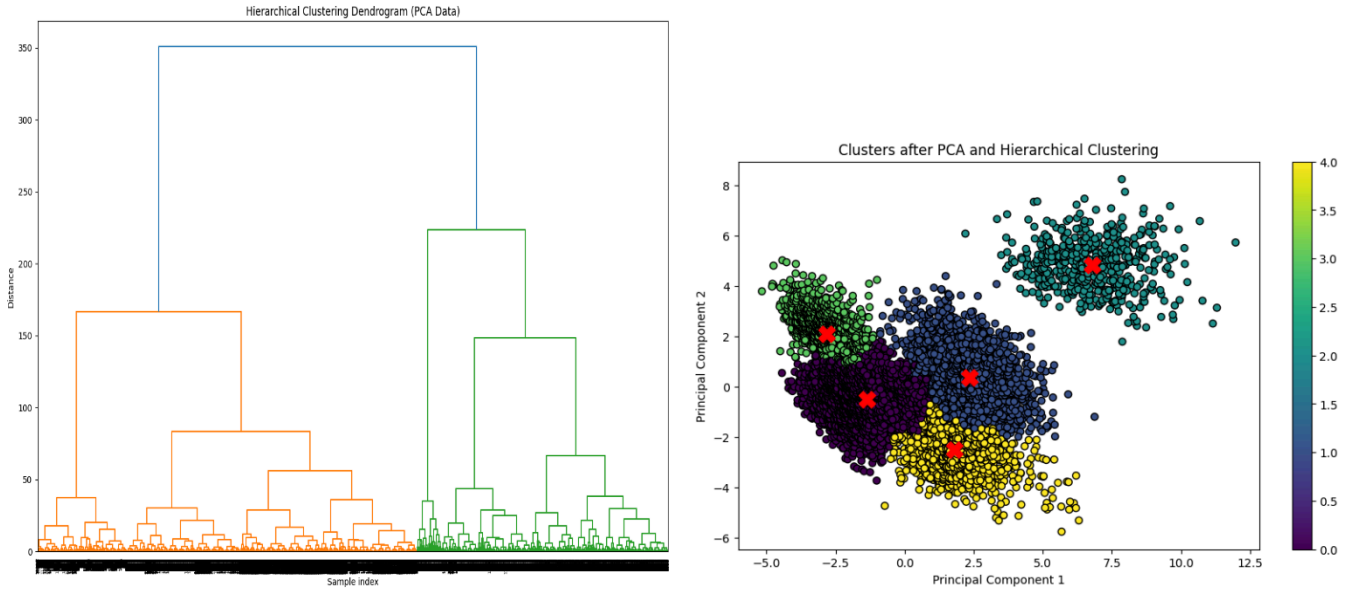


Figure 21: Dendogram (Left) & Dry beans clusters with Hierarchical Clustering (right)

Table 2d: Hierarchical Clustering Evaluation result

Metric	Value	Interpretation
Silhouette Score	0.50	Moderate cluster separation, well-defined
Homogeneity Score	0.63	Shows moderate homogeneity. Most clusters consist predominantly of data points from a single true class, but there are still misgrouped points.

c) DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

It has to do with identifying clusters based on density rather than distance, as in K-Means. After several parameter tuning of eps and min_samples, the maximum cluster results with minimum noise was 3 as shown in fig. 22. The result is of very low accuracy probably because the parameters have close densities.

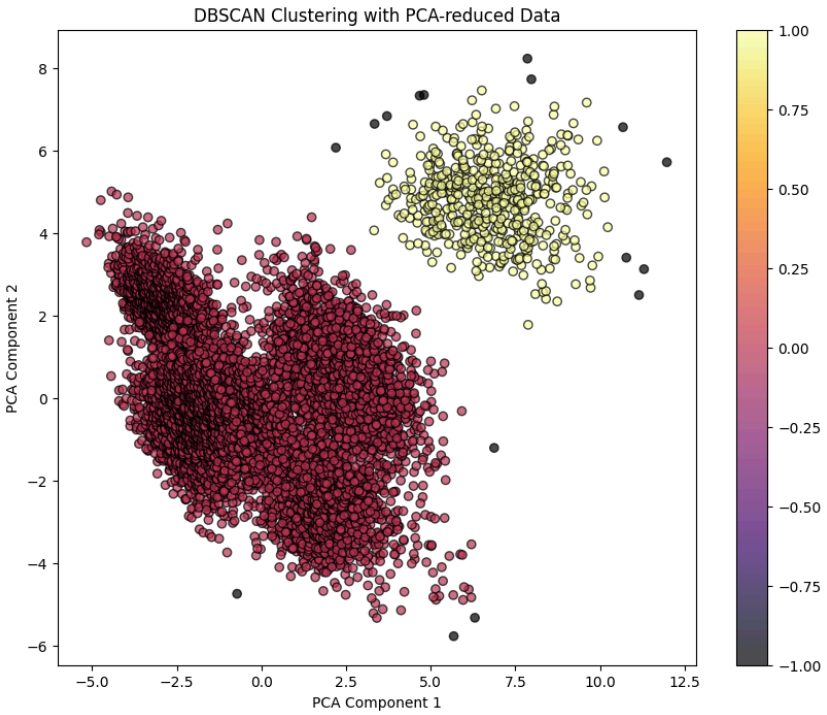


Figure 21: DBSCAN Clustering of dry beans

The clustering methods—KMeans, Hierarchical, and DBSCAN—yielded fewer clusters (5, 5, and 3 respectively) than the seven known bean classes, indicating a potential overlap in features or inadequacy in capturing the distinct variations among classes. This mismatch suggests the need for the exploration of more sophisticated clustering techniques to accurately distinguish all bean varieties. However, an attempt was made to balance the dataset using SMOTE (Synthetic Minority Over-sampling Technique) before clustering, but there was no difference in the optimum no. of clusters suggested by the elbow method (KMeans) as shown in fig. 28.

d) Other Classification Models used

Logistics Regression: Optimum parameters were supplied by GridSearchCV. Test Accuracy is 91.98%. The Confusion Matrix is shown in fig. 22.

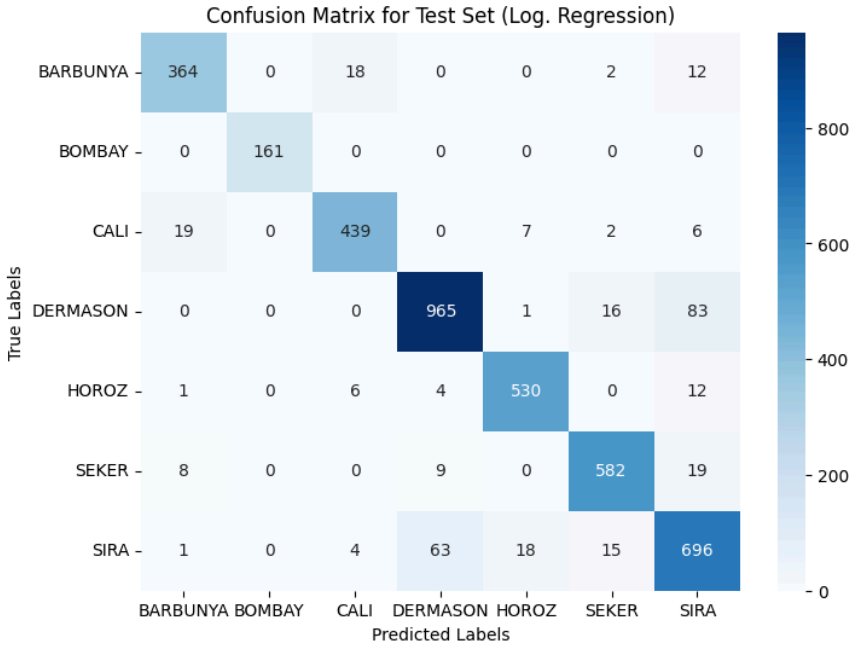


Fig. 22: Logistics Regression- Confusion Matrix for Test Set

XGBoost Classifier

Optimum parameters were supplied by GridSearchCV. Test Accuracy is 92.37%. The Confusion Matrix is shown in fig. 23.

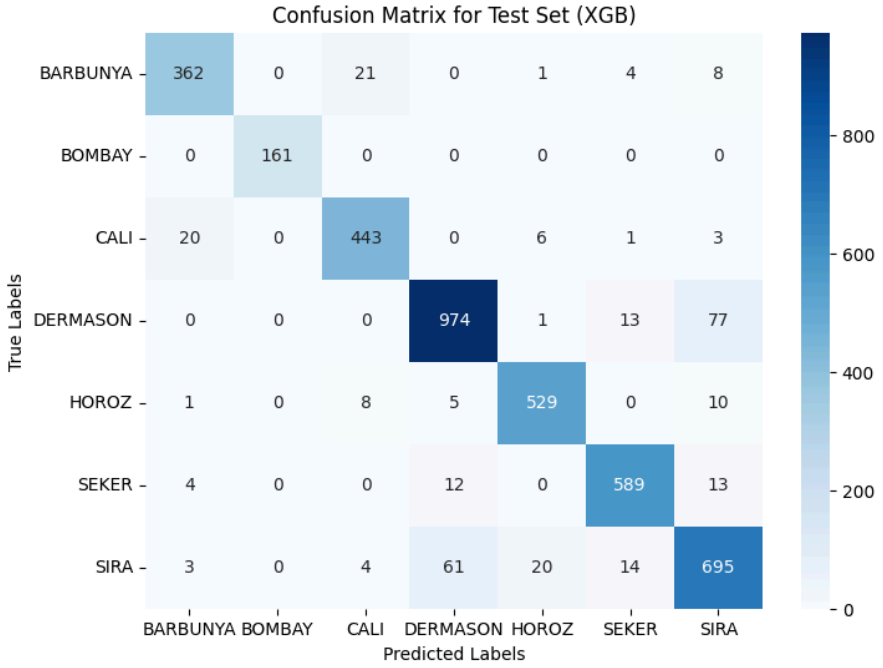


Fig. 23: XGBoost (XGB)- Confusion Matrix for Test Set

KNearest Neighbor Classifier (KNN)

Optimum parameters were supplied by GridSearchCV. Test Accuracy is 92.17%. The Confusion Matrix is shown in fig. 24.

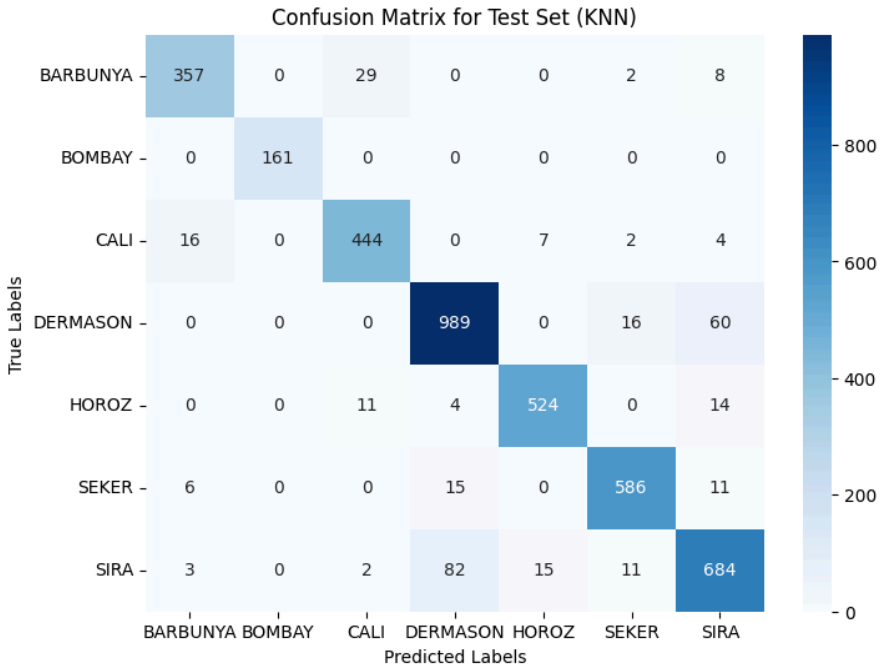


Fig. 24: KNN- Confusion Matrix for Test Set

AdaBoost Classifier (ADB)

Optimum parameters were supplied by GridSearchCV. Test Accuracy is 49.86%. The Confusion Matrix is shown in fig. 25.

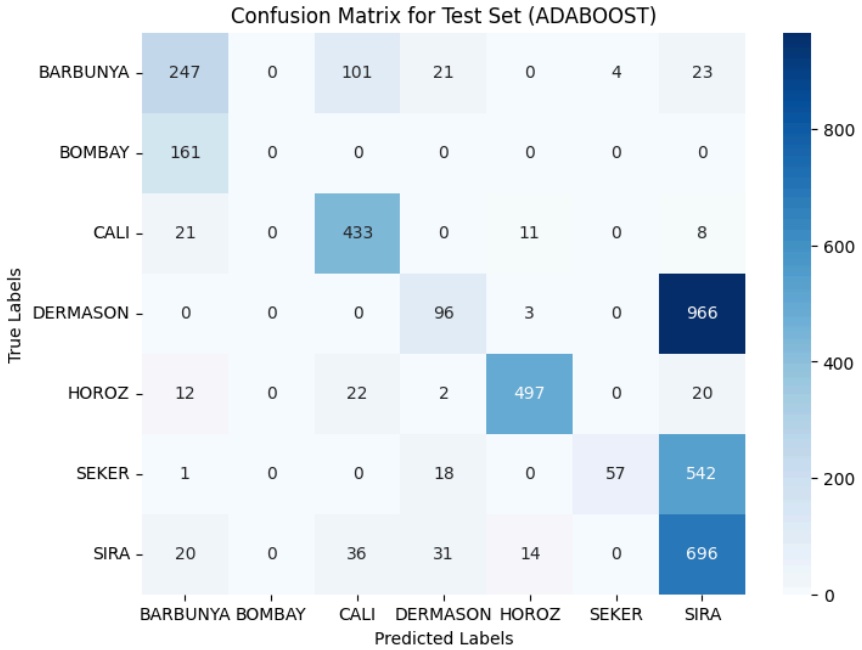


Fig. 25: ADB- Confusion Matrix for Test Set

Gaussian Naive Bayes Classifier (GNB)

Optimum parameters were supplied by GridSearchCV. Test Accuracy is 89.52%. The Confusion Matrix is shown in fig. 26.

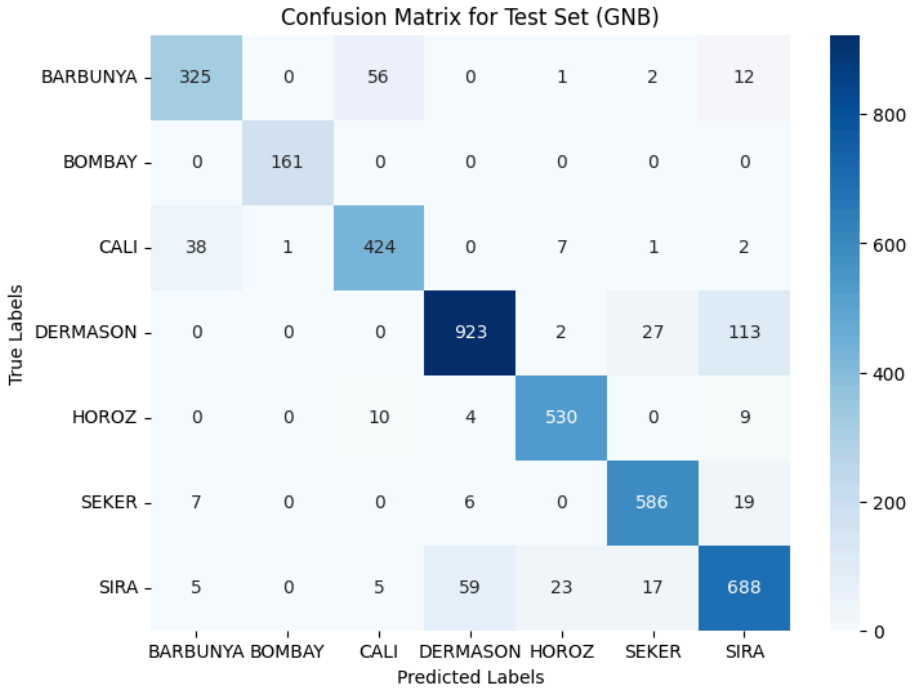


Fig. 26: GNB- Confusion Matrix for Test Set

Table 3: Summary of Classification Models Accuracy- Computation using Google Colab

Model	Accuracy
SVM	93.13%
MLP	92.96%
CNN	92.74%
Logistics Regression	91.98%
Random Forest	92.52%
XGBoost	92.37%
AdaBoost	49.86%
GNB	89.52%
KNN	92.17%

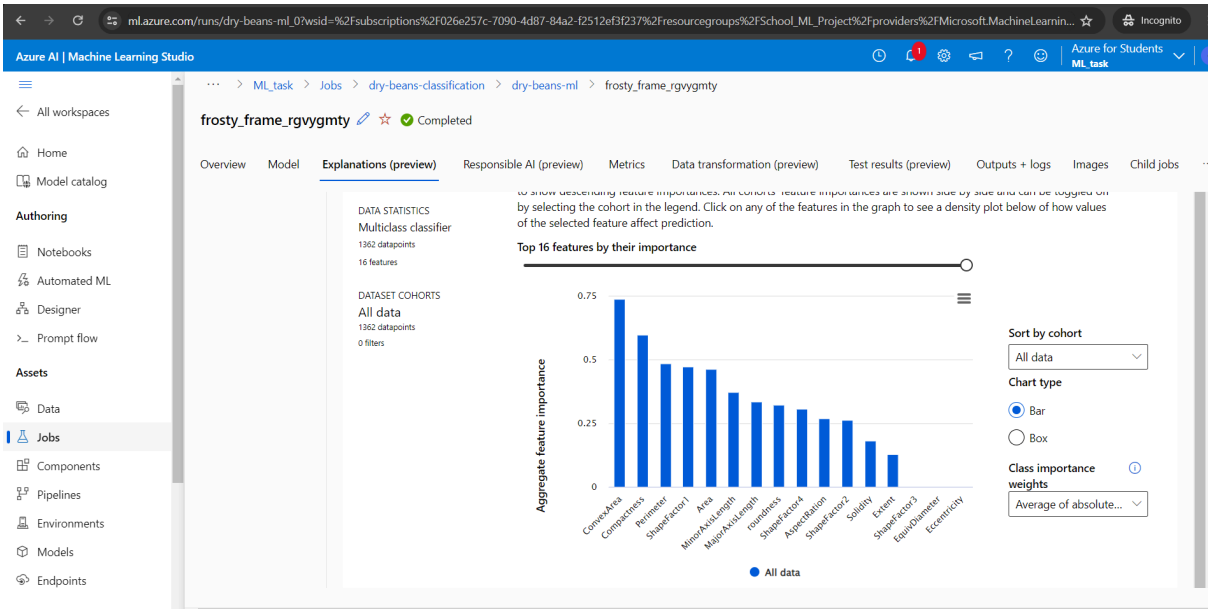
SVM has the highest prediction accuracy while AdaBoost has the least. Except for Adaboost, other models gave relatively close accuracies between the range of 89.52-93.13% which is like the results obtained by Grzegorz (n.d) where accuracy ranges between 88.3-93.6%.

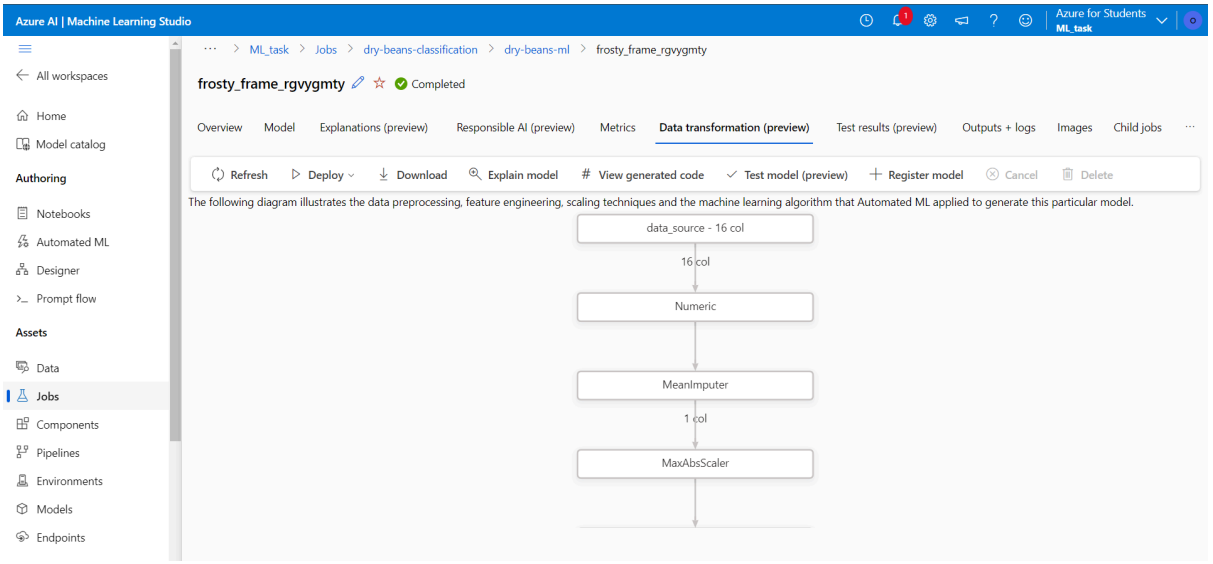
CLOUD-BASED CLASSIFICATION SERVICE

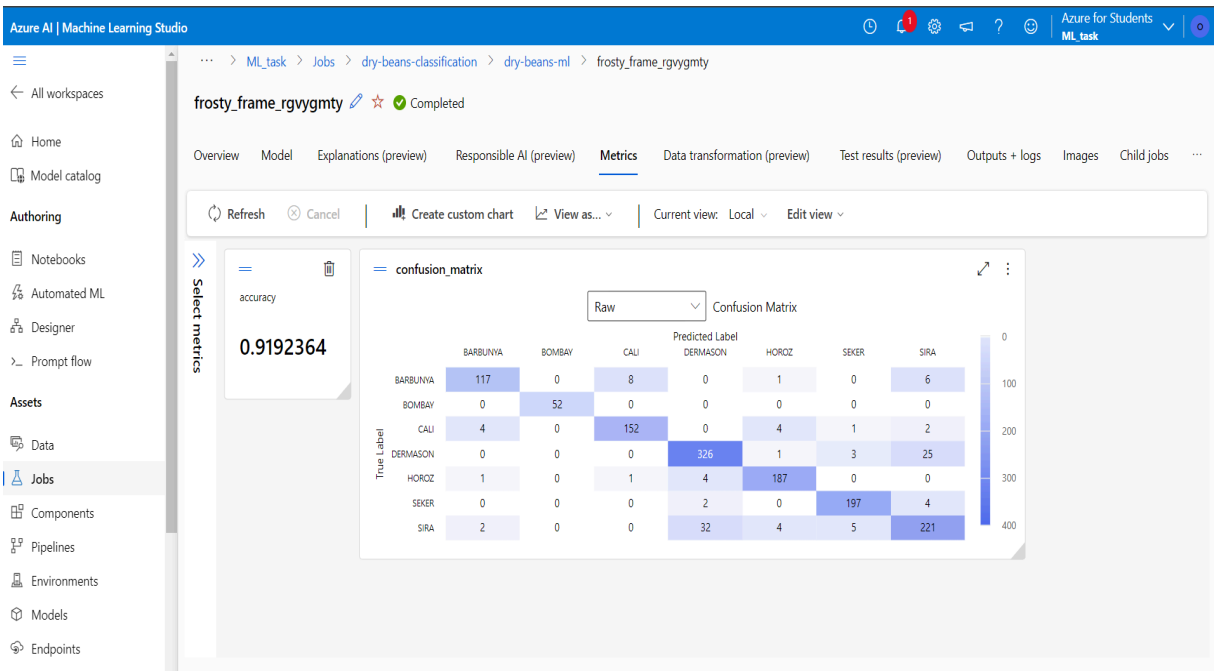
Automated Machine Learning (AML) classification models were built on Microsoft Azure. Some of the models used are: XGB, Random Forest, Log. Regression, KNN and SVM. The Scaler used by the system is MaxAbsScaler (different from Robust Scaler used on google colab) and splitted dataset into training & validation. The highest performing model is XGB with an accuracy of 91.92% which is different from the result of manual computation. The difference may be due to the use of different scaler, data processing and parameter tuning. Figure 27 shows the snapshots of the cloud environment. Before deployment, the model was used to predict a class of beans and it gave an accurate result. Below are the credentials to access the cloud service:

Cloud Solution Access Details:

REST ENDPOINT:
http://cb3f3f8b-0807-4167-83be-e09d899786c5.ukwest.azurecontainer.io/score
SWAGGER URL:
http://cb3f3f8b-0807-4167-83be-e09d899786c5.ukwest.azurecontainer.io/swagger.json
AUTHENTICATION
Primary Key: PEMGzzaIglp0i8bYElQipWhfhYhoBIRA
Secondary Key: fHDJfypisV7xvPbY7sD493tJxVHgnLBA


Azure AI | Machine Learning Studio
ML_task > Jobs > dry-beans-classification > dry-beans-ml > frosty_frame_rgvygmty
frosty_frame_rgvygmty
Completed
Overview
Model
Explanations (preview)
Responsible AI (preview)
Metrics
Data transformation (preview)
Test results (preview)
Outputs + logs
Images
Child jobs
DATA STATISTICS
Multiclass classifier
1362 datapoints
16 features
DATASET COHORTS
All data
1362 datapoints
0 filters
Top 16 features by their importance
Aggregate feature importance
Sort by cohort
All data
Chart type
Bar
Box
Class importance weights
Average of absolute...
All data


Azure AI | Machine Learning Studio
ML_task > Jobs > dry-beans-classification > dry-beans-ml > frosty_frame_rgvygmty
frosty_frame_rgvygmty
Completed
Overview
Model
Explanations (preview)
Responsible AI (preview)
Metrics
Data transformation (preview)
Test results (preview)
Outputs + logs
Images
Child jobs
Refresh
Deploy
Download
Explain model
View generated code
Test model (preview)
Register model
Cancel
Delete
The following diagram illustrates the data preprocessing, feature engineering, scaling techniques and the machine learning algorithm that Automated ML applied to generate this particular model.
data_source - 16 col
16 col
Numeric
MeanImputer
1 col
MaxAbsScaler


Azure AI | Machine Learning Studio
ML_task > Jobs > dry-beans-classification > dry-beans-ml > frosty_frame_rgvygmty
frosty_frame_rgvygmty
Completed
Overview
Model
Explanations (preview)
Responsible AI (preview)
Metrics
Data transformation (preview)
Test results (preview)
Outputs + logs
Images
Child jobs
Refresh
Cancel
Create custom chart
View as...
Current view: Local
Edit view
Select metrics
accuracy
0.9192364
confusion_matrix
Raw
Confusion Matrix
Predicted Label
True Label
BARBUNYA
BOMBAY
CALI
DERMASON
HOROZ
SEKER
SIRA

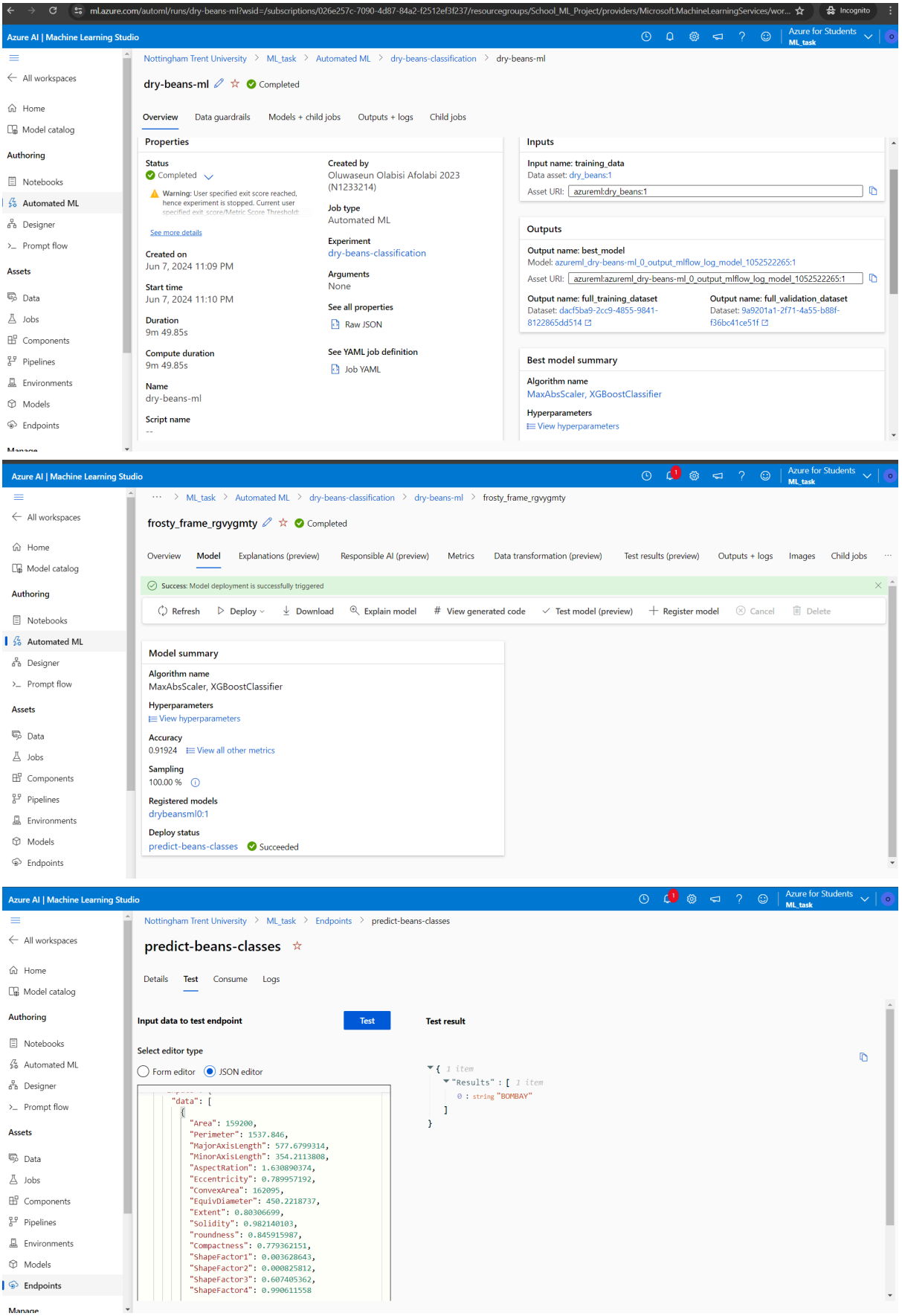


Figure 27: Snapshots of AML Cloud Environment

ETHICAL AND SOCIAL IMPACT OF AI SOLUTIONS IN DRY BEAN CLASSIFICATION AND CLUSTERING

The application of AI in agricultural practices such as dry bean classification and clustering presents a transformative potential for the industry. However, alongside the technological advancements, below are the ethical and social implications these innovations carry:

- **Data Privacy and Security**

Sensitive data related to farming practices, crop yields, and perhaps individual farmer's data might be collected and analyzed. Ensuring the confidentiality and integrity of this data is paramount. Unauthorized access or data breaches can lead to significant financial losses or misuse of farmer's proprietary information (Ryan 2023). Implementing robust security measures and adhering to data protection laws are essential to safeguard these interests (Wang et al. 2021).

- **Bias and Fairness**

AI systems are only as good as the data they are trained on. If the dataset for bean classification is biased or unrepresentative of the actual diversity of bean varieties, the AI model could perpetuate or amplify these biases. For instance, if certain bean types are underrepresented, the model might perform poorly when classifying these beans, leading to economic disadvantages for farmers specializing in those varieties. Continuous monitoring and updating of the dataset to ensure it is comprehensive and representative are necessary steps to mitigate such issues (Ryan 2023).

- **Transparency and Explainability**

AI models, particularly those involving complex algorithms like neural networks, often suffer from a lack of transparency and explainability. Stakeholders such as farmers, regulators, and consumers might distrust AI solutions that do not make their decision-making processes clear. Developing models that are not only accurate but also interpretable is crucial for gaining trust and facilitating wider adoption (Ryan 2019).

- **Impact on Employment**

The automation of bean classification through AI could significantly reduce the need for manual labor, leading to concerns about job displacement. While AI can increase efficiency and reduce the human error associated with manual sorting, it is essential to consider the social impact on individuals whose livelihoods depend on these jobs. Implementing AI solutions should be accompanied by efforts to re-skill and re-deploy affected workers to other areas where their expertise and knowledge of agricultural practices are valuable (Carolan 2006).

- **Sustainability and Environmental Impact**

Efficiently classifying and clustering beans using AI can contribute to more sustainable agricultural practices. By accurately classifying bean types, farmers & processors can optimize their use of resources, reduce waste and error. However, the environmental impact of developing and running AI systems (e.g. Robots & Sensors) which is often energy-intensive should also be considered. Striving for greener AI by optimizing algorithmic efficiency and utilizing sustainable energy sources is a step toward mitigating these impacts (Krishnan et al. 2020).

Ensuring data privacy, addressing bias, enhancing transparency, managing employment impacts, and minimizing environmental impacts are critical. Ethical AI deployment in agriculture requires a balanced approach that considers both technological benefits and potential social ramifications. As this project progresses, continuous evaluation of these aspects will be crucial for responsible and sustainable implementation.

CONCLUSION

As for Beans Classification without the use of cloud, SVM performs best in terms of accuracy while on the cloud, XGBoost performed best. From the confusion matrices, almost all the models classified Bombay Class perfectly except for Adaboost. Adaboost is not recommended for dry bean classification. From the result of the unsupervised learning (clustering), the clustering methods do not produce accurate grouping of the beans according to their classes. Further research would be to analyze and compare the training time of each of the models and to determine the best method of clustering to be used for dry beans data. It would also be good to know if class balancing would improve models' accuracy.

REFERENCES

AlZubi, A.A., Galyna, K., 2023. Artificial intelligence and internet of things for sustainable farming and smart agriculture. *IEEE Access*.

Bouguettaya, A. et al., 2015. Efficient agglomerative hierarchical clustering [online]. *Expert Systems with Applications*, 42(5), pp.2785–2797. Available at: https://www.sciencedirect.com/science/article/pii/S0957417414006150.

Carolan, M.S., 2006. Social change and the adoption and adaptation of knowledge claims: Whose truth do you trust in regard to sustainable agriculture? *Agriculture and human values*, 23, pp.325–339.

Gholami, R., Fakhari, N., 2017. Chapter 27 - Support Vector Machine: Principles, Parameters, and Applications [eBook]. In: Samui, P., Sekhar, S., Balas, V. E., eds. *Handbook of Neural Computation*. Academic Press, 2017, pp. 515–535. Available at: https://www.sciencedirect.com/science/article/pii/B9780128113189000272.

Grzegorz Slwinski (n.d). Dry Beans Classification Using Machine Learning. University of Technology and Economics, ul. Jagiellońska 82f, 03-301 Warsaw, Poland.

Izonin, I. et al., 2022. A two-step data normalization approach for improving classification accuracy in the medical diagnosis domain. *Mathematics*, 10(11), p.1942.

Kaggle. (2024). *Dry Bean Dataset Classification*. Available at:

https://www.kaggle.com/datasets/nimapourmoradi/dry-bean-dataset-classification/data

Koklu, M., Ozkan, I.A., 2020. Multiclass classification of dry beans using computer vision and machine learning techniques [online]. *Computers and Electronics in Agriculture*, 174, p.105507. Available at: https://www.sciencedirect.com/science/article/pii/S0168169919311573.

Krishnan, A., Swarna, S., S, B.H., 2020. Robotics, IoT, and AI in the Automation of Agricultural Industry: A Review. In: *2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)*. pp. 1–6. 10.1109/B-HTC50970.2020.9297856.

Ryan, M., 2019.

Ryan, M., 2023. The social and ethical impacts of artificial intelligence in agriculture: mapping the agricultural AI literature [online]. *AI & SOCIETY*, 38(6), pp.2473–2485. Available at: https://doi.org/10.1007/s00146-021-01377-9.

Wang, T. et al., 2021. From smart farming towards unmanned farms: A new mode of agricultural production. *Agriculture*, 11(2), p.145.

Other study Materials:

https://stackoverflow.com/questions/68253660/pca-on-data-and-training-with-svm-with-k-fold-cv-and-gridsearch?rq=3

https://www.simplilearn.com/data-preprocessing-in-machine-learning-article https://www.simplilearn.com/data-preprocessing-in-machine-learning-article

https://towardsdatascience.com/k-means-clustering-fa4df5990fff

https://medium.com/@polanitzer/a-multi-layer-perceptron-classifier-in-python-predict-digits-from-gray-scale-images-of-hand-drawn-44936176be33

Building our first neural network in keras | by Sanchit Tanwar | Towards Data Science

https://www.analytixlabs.co.in/blog/introduction-support-vector-machine-algorithm/

https://scikit-learn.org/stable/auto_examples/cluster/index.html