

# To develop a system for automatic detection of 7 types of dry bean seeds based on data captured using a high-resolution camera.

## 1 Introduction

In this course work, seven different varieties of dry beans were employed, with consideration given to characteristics including form, shape, type, and structure considering the market environment. A computer vision system was developed to distinguish between seven different registered varieties of dry beans with similar features to achieve uniform seed categorization. A high-resolution camera was used to capture photos of 13,611 grains from 7 different registered dry bean varieties for the classification model. The segmentation and feature extraction steps of the computer vision system's bean image processing produced a total of 16 features from the grains: 12 dimensions and 4 shape types. particularly when it comes to identifying difficult differences between different types of beans in summary dry bean image classification problem [1].

The methods and techniques to be deployed in this coursework are majorly machine learning methods. I am adopting random forest and decision tree; all are classifiers based on the problems we are solving which is a classification problem. The RandomForestClassifier, DecisionTreeClassifier are one of ML model that perform well on classification problem and distribution inbalance data can be fixed by them using the cross validation/Random Search CV improve the performance of the models. The decision tree model result will compare with research work/journal work and the random forest model will be the benchmark.

## 2. Methods and techniques

### 2.1 Ensemble Methods

**Random Forest Classifier:** Random Forest is an ensemble learning method that builds multiple decision trees and combines them to make predictions. It is known for its high accuracy, robustness, and ability to handle large datasets. Random Forest can handle both classification and regression problems, and it is also effective in dealing with missing data and outliers. It is less prone to overfitting than decision trees, but it can be slower to train and predict due to the large no. of trees.[2]

$$RFfi_i = \frac{\sum_{j \in \text{all trees}} \text{normfi}_{ij}}{T}$$

The importance of feature  $i$  as determined by all the trees in the Random Forest model is represented by  $RFfi_{\text{sub}(i)}$ . The normalised feature significance for  $i$  in tree  $j$  is given by  $\text{normfi}_{\text{sub}(ij)}$ , and  $T$  is the total number of trees.

**Decision Tree Classifier:** is a classification algorithm that is part of the scikit-learn library in Python. It is on the concept of a decision tree, which is a tree-like model of decisions and their possible consequences. In the context of the DecisionTreeClassifier, the tree is built by recursively partitioning the input data based on the values of different features. At each node in the tree, a decision is made based on the value of one of the features, and the data is split into two groups based on the decision. This process is repeated recursively on each subset of

data until a stopping criterion is met, such as a maximum depth of the tree or a minimum number of samples required to make a split. One of the main advantages of the DecisionTreeClassifier algorithm is that it can capture complex non-linear relationships between the features and the target variable. [2]

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)}$$

$w_{sub(j)}$  is the weighted number of samples that reach node  $j$ , and  $ni_{sub(j)}$  is the importance of node  $j$ .  $Left(j)$  is the child node from the left split on node  $j$ , and  $C_{sub(j)}$  is the impurity value of node  $j$ .  $child$  node from the right split on node  $j$ ,  $right(j)$

The two classifiers are imported from the sklearn python library.

## 2.2 Evaluation Metrics

**Confusion Matrix:** As the name implies, Confusion Matrix provides us with a matrix as an output that describes the entire performance of the model. Which are represented as True Positives: Situations in which we made a prediction of "YES" and the result was also "YES." True Negatives: Situations in which we anticipated NO but the result was NO. False Positives: The instances in which we projected that the outcome would be YES but it was really NO. False Negatives: Situations in which we expected a NO result but the final result was a YES.

**Accuracy Score:** Taking the average of the numbers along the "main diagonal" will yield the matrix's accuracy. [3]

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalSample}$$

**Precision Score:** The ratio of accurate positive discoveries to those that the classifier expected to be positive is used to calculate it. [3]

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

**F1 Score:** The F1 Score is used to determine how correctness accuracy. [3]

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

**Cross Validation Score:** Evaluate a score by cross-validation. These are evaluation based on the general performance of each model in question.

**Recall Score:** To compute it, divide the total number by of relevant samples (all samples that ought to have been classified as positive) by the number of accurate positive results. [3]

## 3 Findings and Presentations

### 3.1 General Exploratory Analysis

Duplicate row was identified in a dataset. This duplicate dataset could cause skew statistical analysis or machine learning algorithms, as they may artificially inflate the significance of certain values or relationships in the data. The noise was removed. And label encoding to replace 7 classes of bean with integer.

Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRatio	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	s
28395	610.291	208.178117	173.888747	1.197191	0.549812	28715	190.141097	0.763923	0.988856	0.958027	0.913358	
28734	638.018	200.524796	182.734419	1.097356	0.411785	29172	191.272750	0.783968	0.984986	0.887034	0.953861	
29380	624.110	212.826130	175.931143	1.209713	0.562727	29690	193.410904	0.778113	0.989559	0.947849	0.908774	
30008	645.884	210.557999	182.516516	1.153638	0.498616	30724	195.467062	0.782681	0.976696	0.903936	0.928329	
30140	620.134	201.847882	190.279279	1.060798	0.333680	30417	195.896503	0.773098	0.990893	0.984877	0.970516	

Figure 3.0 Dry Bean Dataset

Furthermore, on data exploration fig 3.1 was a clear picture of dataset representation.

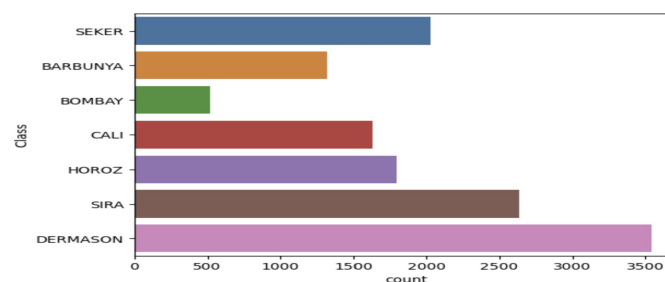


Figure 3.1 Countplot using seaborn

from the countplot, the count and distribution of labels category are inbalance with this prediction would be imperfect. furthermore, an inbalance distribution in label in a dataset can negative affect the performance of the model. The issue was handled by deployment of ensemble methods to ensure accurate predictions on all classes. No missing data identified and visible correlation among the features plotted in jupyter notebook using heatmap in fig. 3.1. Found in the jupyter notebook.

### 3.2 Model Performance Evaluation before hyperparameter tuning

The dataset was randomly splits a pandas DataFrame called 'dry\_bean\_dat1' into three subsets: training, validation, and testing. The size of each subset is defined as 60% for training, 20% for validation, and 20% for testing. The setdiff1d() function is used to exclude the indices already selected for the training and validation set. union1d() function, and the replace parameter is set to False to ensure that each index is unique. Train set for training model, validation set for hyper tuning to improve the model and test set for predicting labels. Having in mind that while a high fitting score on the training set is desirable, it does not necessarily mean that the model will perform well on new, unseen data. To address this issue, it's common to use a separate validation or test set to evaluate the model's performance on new data.

```

Accuracy: {:.2f} 0.9177075679647319 Random Forest
Precision: {:.2f} 0.9177414643957549 Random Forest
Recall: {:.2f} 0.9177075679647319 Random Forest
F1 Score: {:.2f} 0.9173907579895801 Random Forest
Accuracy: {:.2f} 0.8949301983835415 Decision Tree
Precision: {:.2f} 0.895688551347556 Decision Tree
Recall: {:.2f} 0.8949301983835415 Decision Tree
F1 Score: {:.2f} 0.8952120196569714 Decision Tree

```

Figure 3.2 Metric Values

Fig. 3.2 the metrics for evaluation of the two models are accuracy, precision, recall and f1 score for the general performance of the machine learning model and evaluation result on the models is good. but require to improved i.e hyperparameter tuning.

---

Individual classification accuracy: [0.9040590405904059, 0.9836065573770492, 0.9296187683284457, 0.9376731301939059, 0.9444444444444444, 0.9546666666666667, 0.8305084745762712] Random Forest  
 Individual classification accuracy: [0.8708487084870848, 0.9836065573770492, 0.9032258064516129, 0.8905817174515236, 0.9222222222222223, 0.9226666666666666, 0.8493408662900188] Decision Tree

---

Figure 3.3 Accuracy value of each classes of dry bean

The accuracy flows from 'SEKER', 'BARBUNYA', 'BOMBAY', 'CALI', 'HOROS', 'SIRA', 'DERMASON' to the least in that order. Comparing it to the decision tree model that was used in the research work no significant difference seen.

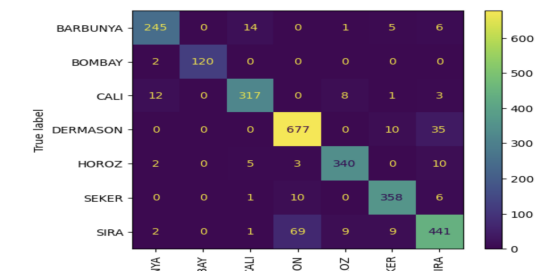


Figure 3.3a Random Forest

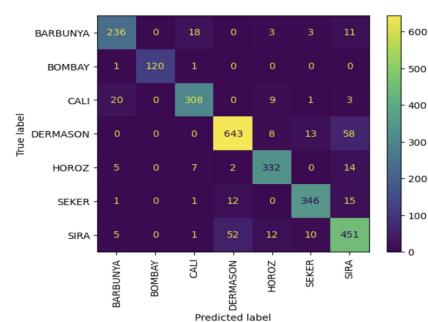


Figure 3.3b Decision Tree

Fig1.3a and Fig1.3b centres on dermason bean class. confusion matrix helps to understand the strengths and weaknesses of the model, and to identify areas for improvement. The first confusion model is random forest and the second is decision tree. The high number of false positives may indicate that the model is too aggressive in predicting positive instances, while a high number of false negatives may indicate that the model is missing important information. By analysing the confusion matrix and the associated performance metrics, we can make informed decisions about how to improve the model's performance on the given task. Dermason and Bombay is had false positive classification but a true positive classification. This suggest a further improvement on the model is possible by hyper tuning the parameter to get the best.

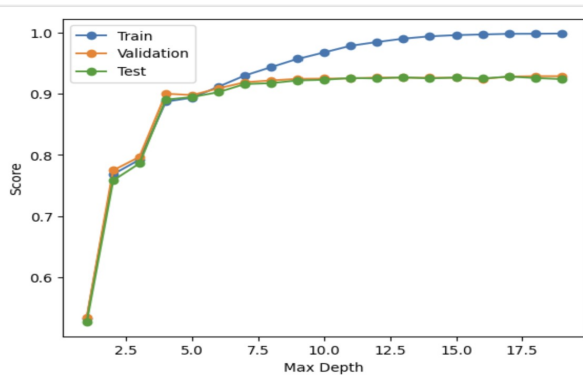


Figure 3.4a Random Forest

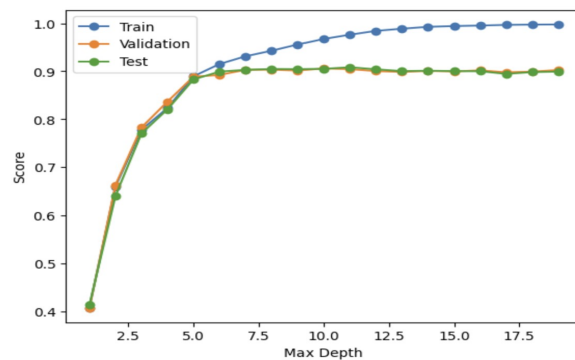


Figure 3.4b Decision Tree

The x-axis represents the maximum depth of the decision tree and random forest, while the y-axis represents the accuracy. The blue line shows the accuracy on the training set, the green line shows the accuracy on the validation set, and the red line shows the accuracy on the test set. This plot indicates the optimal depth that maximizes the accuracy on the validation set, while avoiding overfitting on the training set which could be improved further. The decision tree model is properly fitted to Random Forest as described in fig 3.4 a and b.

### 3.3 Model Performance Evaluation after Hyperparameter tuning.

The purpose of using a validation set is to tune the hyperparameters of the model, such as criterion, max\_depth etc. The validation set allows for the selection of the best hyperparameters that minimize the model's error on the validation set. After tuning the hyperparameters, the final model is evaluated on the testing data to estimate its generalization performance. This is important to assess the model's ability to make accurate predictions on new, unseen data. Random Search CV was used to implement the hyperparameter tuning.

Accuracy: {:.2f} 0.9166054371785451	Accuracy: {:.2f} 0.8923585598824394
Precision: {:.2f} 0.9295144168557276	Precision: {:.2f} 0.9038697439454602
Recall: {:.2f} 0.9226905269851169	Recall: {:.2f} 0.8996889719634213
F1 Score: {:.2f} 0.9259938980056309	F1 Score: {:.2f} 0.9013393803977024

Figure 3.5 Metric Values

It was observed increase in precision, recall, f1 score values showing a sign of improved model but a decrease in accuracy i.e why accuracy should always be a parameter of judgement in machine learning experiments in random forest and decision tree algorithm. This is a clear indication that the model will perform on unseen data. Furthermore, accuracy is not a perfect metric for judgement as observed in fig. 3.5.

Individual classification accuracy: [0.9114391143911439, 0.9590163934426229, 0.8914956011730205, 0.9335180055401662, 0.9277777777777778, 0.9413333333333334, 0.8418079096045198] Random Forest  
 Individual classification accuracy: [0.922509225092251, 0.9590163934426229, 0.8504398826979472, 0.8878116343490304, 0.925, 0.9173333333333333, 0.7947269303201506] Decision Tree

Figure 3.6 Accuracy value of each classes of dry bean

Fig. 3.1.6 indicates an improvement model.

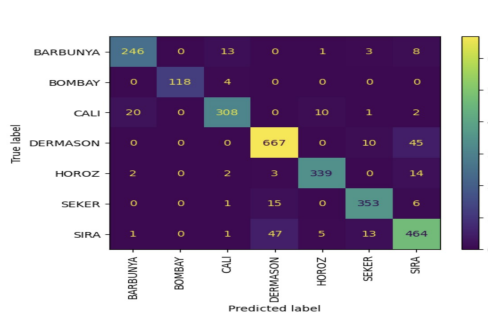


Figure 3.7a Random Forest

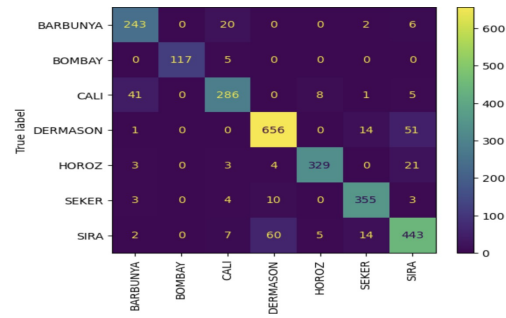


Figure 3.7b Decision Tree

Comparing fig 3.3a, b and fig 3.7a, b. center the comaprison on the value of dermason class type of dry bean. The value increases is a strong indication of model improvement.

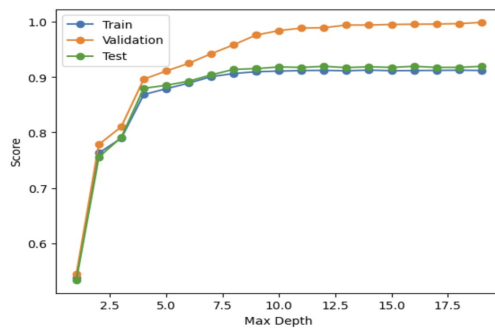


Figure 3.8a Random Forest

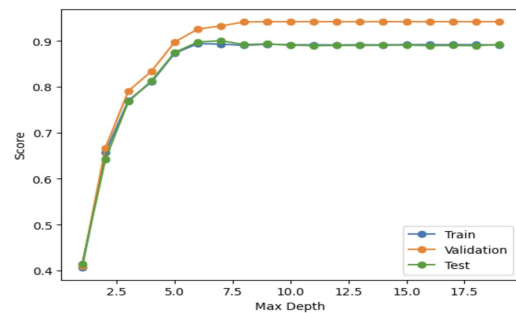


Figure 3.8b Decision Tree

Comparing fig 3.4a, b and fig 3.7a, b. center the comaprison on the value of dermason class type of dry bean. The value increases is a strong indication of model improvement.

#### 4 Extra Task and Evaluation

The automatic detection model was implemented or developed using the machine learning algorithm classifier support vector machine. The csv file was generated having idx and predicted labels. Comparing the models, the random forest and decision tree classify. SVM is generally effective when there is a clear boundary between the classes, decision trees are easy to interpret, and random forest is less prone to overfitting. The choice of algorithm depends on the specifics of the problem at hand, including the size and complexity of the dataset and the nature of the features.

#### Conclusion

In conclusion, the objective of the research work was met by classifying different kinds of dry bean. Dry bean classification is a challenging problem in agricultural research, as accurately identifying and classifying different varieties of dry beans is essential for ensuring crop productivity and food security. The automatic detection system was as well achieve, the automatic detection system on test data was able to predict accurately.

#### Reference

**DATASET:** <https://www.muratkoklu.com/datasets/>

[1] <https://archive.ics.uci.edu/ml/datasets/Dry+Bean+Dataset>

#1 <https://machinelearningmastery.com/difference-test-validation-datasets/> #2 <https://www.fast.ai/2018/09/26/ml-launch/>

#3 [https://www.youtube.com/watch?v=YSFG\\_W8JxBo](https://www.youtube.com/watch?v=YSFG_W8JxBo)

[2] <https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees>

[3] <https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234>

[4] <https://scikit-learn.org/stable/modules/ensemble.html#voting-classifier>