ISL Team Project

Data description:

Images of 13,611 grains of 7 different registered dry beans were taken with a high-resolution camera. We have used info function to get the details of the dataset.

```
print(data.info()) # displays information about the dataset
#dataset has samples = 13611
#features = 17
#type of features = 14 - float, 2 - int, 1 - object
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13611 entries, 0 to 13610
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Area             13611 non-null  int64  
 1   Perimeter        13611 non-null  float64
 2   MajorAxisLength  13611 non-null  float64
 3   MinorAxisLength  13609 non-null  float64
 4   AspectRation     13608 non-null  float64
 5   Eccentricity     13611 non-null  float64
 6   ConvexArea       13609 non-null  float64
 7   EquivDiameter    13610 non-null  float64
 8   Extent           13608 non-null  float64
 9   Solidity         13608 non-null  float64
 10  roundness        13611 non-null  float64
 11  Compactness      13611 non-null  float64
```

Data pre-processing:

The Dataset provided has few null values. We have used isnull function to get those null values.

```
missing_values = data[data.isnull().any(axis=1)]
print(missing_values)
```

```
      Area  Perimeter  MajorAxisLength  MinorAxisLength  AspectRation  \
12   31107    640.594       214.648549       184.969253           NaN   
19   31335    635.011       216.790092       184.163440      1.177161   
31   31823    662.532       222.872689       181.894696      1.225284   
47   32218    653.595       222.756071       184.404684      1.207974   
60   32514    649.012       221.445490       187.134423      1.183350   
70   32713    660.043       215.416321       193.486462      1.113341   
81   32885    659.728       227.067404              NaN      1.230320   
91   33019    655.703       224.450211       187.502627      1.197051   
121  33431    666.841       219.907408       193.680468           NaN   
131  33547    653.512       213.754071       199.925701      1.069168   
142  33731    671.696       231.172013       186.126456      1.242016   
158  33961    690.353       238.192499       181.654297      1.311241   
190  34291    671.507       230.658067              NaN      1.216397   
225  34601    685.556       226.519539       194.718757      1.163316   
356  35450    680.993       235.796159       191.585720           NaN   
444  36029    688.635       238.660505       192.575114      1.239311   
```

Using the dataset with null values to perform the clustering was providing not finite results in the distance matrix. Which led the linkage function run into error when forming clusters based on the distances.

The null values were eliminated by replacing the null values to the average value of the entire column by using the median function. This way we didn't lose any of the data and were able to stabilize the model.

```
#for j in datasub.index:
 # for k in datasub.columns:
  #  if '' in datasub.at[j,k]:
   #   datasub.at[j,k] = np.median([float(x) for x

median = data.median()
data.fillna(median, inplace=True)

#print(data.head())
print(data.iloc[12])
```

We also see duplicate rows and we remove them to improve models performances as follows

detecting and handling the duplicates

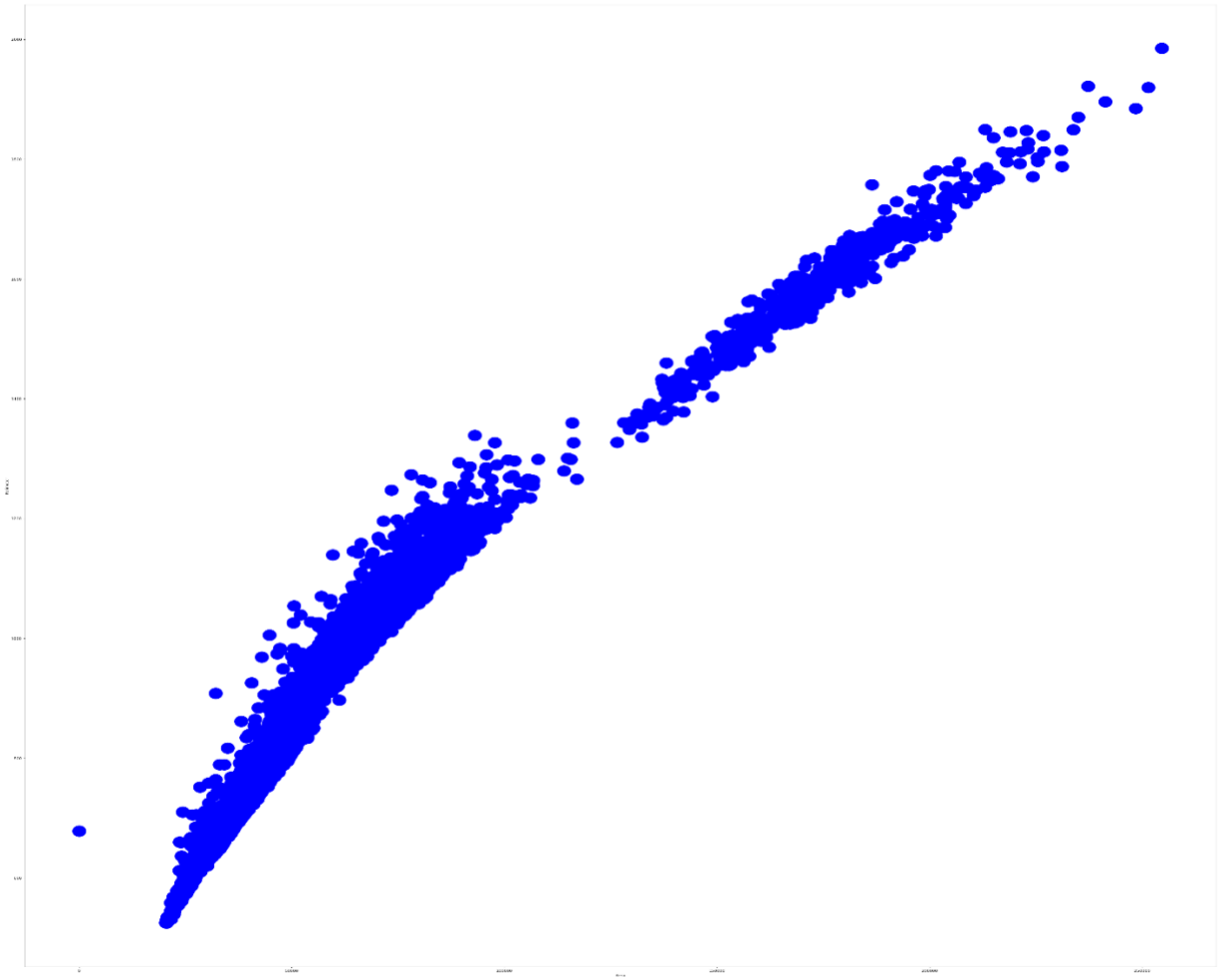
Identify duplicate rows

duplicate_rows = df[df.duplicated()]

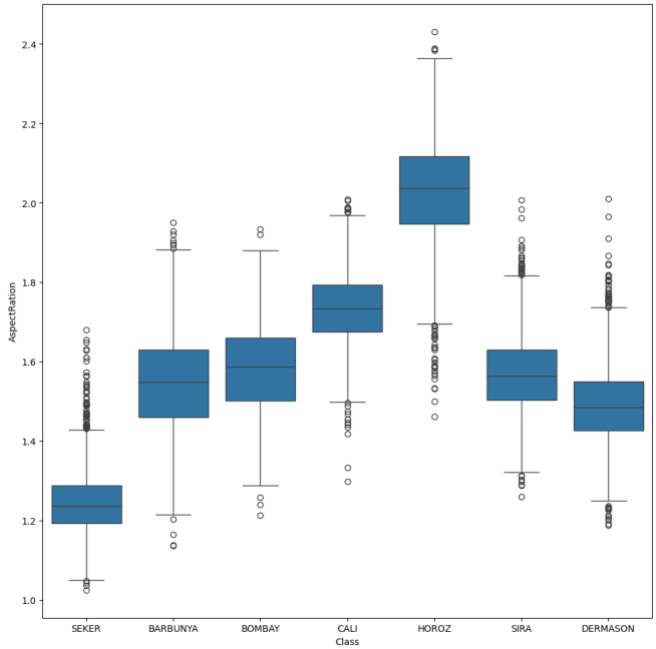
df= df.drop_duplicates()

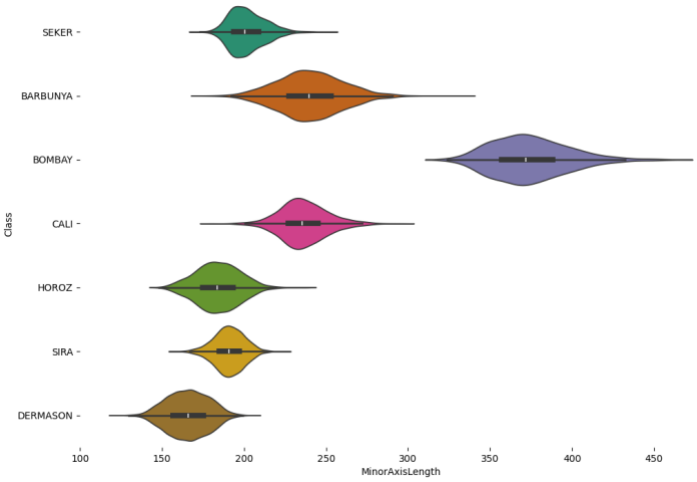
Data Exploration:

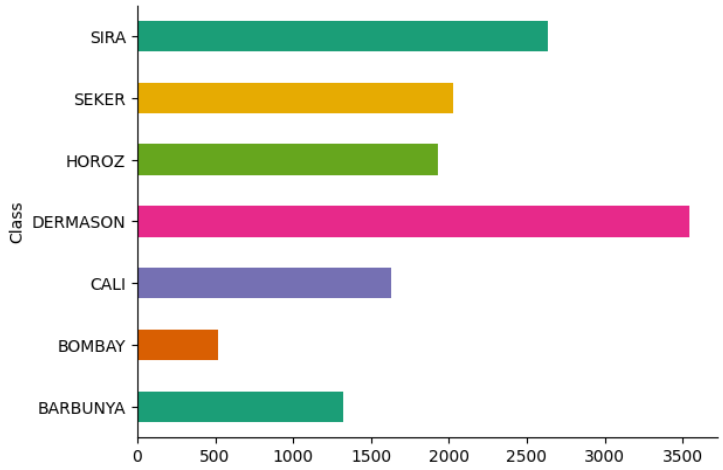
We have plotted graphs for 2 major parameters like area and perimeter in the dataset. Which determine key properties of the dry bean.



The above graph is the relation of area(x) and perimeter (y), we can conclude that both area and perimeter almost have a linear relationship with most of the points have an increasing slope axis. The higher the area the higher is the perimeter of the dry bean.







Visualizing Data Distributions: we have performed exploratory data analysis by creating visualizations such as box plots and violin plots to understand the distribution of features in the dataset. These visualizations provided insights into the spread and central tendencies of our data.
Relationship Exploration: Using seaborn and matplotlib libraries, we have generated visualizations to explore relationships between features and the target. variable. This allowed us to identify potential patterns and correlations within the data.

Performance Evaluation:

For the KNN Model:

Based on the provide results for the KNN model, here's the interpretation

Accuracy:

The accuracy of the KNN model is approximately 85.92%. This indicates that about 85.92% of the predictions made by the model were correct.

Precision Score:

The precision score of the model is approximately 85.82%. Precision measures the proportion of true positive predictions among all positive predictions made by the model. In other words, it indicates how many of the predicted positive instances were actually positive.

Recall:

The recall of the model is also approximately 85.92%. Recall measures the proportion of true positive predictions among all actual positive instances in the dataset. It indicates how many of the actual positive instances were correctly identified by the model.

F1 Score:

The F1 score of your the is approximately 85.85%. The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics. It is a single metric that captures both precision and recall.

For the Decision Trees Model:

In the decision tree classifier, we see that 89.2% of the samples in the test set were correctly classified.

When doing cross validation and finding pruning path we see that 0.91 score in best estimator, indicating that, on average, the model achieved 91% accuracy across the cross-validation folds when using the best value of ccp_alpha**.** And the best tree has 102 leaves.

For the KMeans Model:

We see that Silhouette Score: 0.5367442763949458, which indicates that on average, the objects in your dataset are relatively well matched to their own clusters and are reasonably well separated from neighboring clusters. This suggests that the clustering algorithm has produced clusters that are internally cohesive and sufficiently separated from each other.

For the Hierarchical Clustering Model:

Once the distance matrix is deduced. We store it in Data_dist and use the metric as Euclidean. This provides the distances between pair of points of the dataset.

The total entries in this matrix is n(n-1)/2 where ‘n’ being the number of samples.

Then we use the linkage methods like Complete, Ward, Average to build the dendrogram of the model. From the respective dendrograms we found the accuracy of each method.

Accuracy Complete method= 0.6252888798187158.

Accuracy Ward method= 0.5312564884542805

Accuracy Average method= 6530365189248228

Interpretation:

For the KNN Model:

Influence of Nearest Neighbors (k):

With k=5

k=5, the algorithm considers the five nearest neighbors to make predictions. This might lead to overfitting, where the model is too sensitive to small fluctuations in the data, resulting in lower accuracy.

On the other hand, with k=7

k=7, the algorithm considers a slightly larger neighborhood, smoothing out some of the noise in the data. This can lead to more stable predictions and better generalization to unseen data, resulting in higher accuracy.

Reduced Noise Sensitivity:

By increasing k

k from 5 to 7, the algorithm becomes less sensitive to outliers and noise in the dataset. This can help improve the model's ability to generalize to new data points, leading to better performance.

Better Representation of Data Distribution:

With a larger k

k, the algorithm considers a larger number of data points when making predictions. This can result in a better representation of the underlying data distribution, allowing the model to make more accurate predictions.

For the Decision Trees Model:

Decision Trees Model is used to take decisions as in Interpret the features, and whether to scale the features or not.

Using the Boosting Model, I got the top 3 most signifcant features as MinorAxisLength, ConvexArea, Perimeter

For the KMeans Model:

The data distribution, number of clusters, initialization, feature scaling, outlier presence, cluster shape, and data density are some of the variables that affect a K-means model's performance. The K-means algorithm's performance can be enhanced by being aware of these variables and preparing the data appropriately.

For the Hierarchical Clustering Model:

From all the 3 methods performed we can say the model gives better accuracy with linkage method as Average. Which forms clusters based on the Average distance between each pair of points in two clusters.

Whereas ward method usually forms the more balanced clusters but got the lowest accuracy due the structure of the dataset. The Complete method follows the accuracy race as it forms clusters based on the closeness of points, this can be a decent score.

Conclusion:

In this project we have performed four different models(KNN, Decision trees, K-means, Hierarchical clustering) on the dry bean dataset to evaluate the model accuracy in predicting the values. The data cleaning and pre-processing were performed individually for each model according to the choice of predictors and variables. After importing the Dataset, Visualizations of the data were performed to understand the relation between the variables. Each model is executed and respective accuracies were calculated using the NumPy and panda libraries and the results were interpreted. We identify any data and map it to it's respective class in KNN then we got the features selection from Decision Trees and then using that paired with K-Means we visualized the data and got to the hierarchical clustering with 2 PCA components we got in boosting Tree model. In overall, the choice depends on the model performance with the dataset at various conditions.