DRY BEANS CLASSIFICATION

GROUP-3

ABHIGNA HARSHA SHAMHITH

OBJECTIVES

PROBLEM STATEMENT:

- The problem is to classify dry beans into their respective classes based on their shape and size.
- Accurately classifying dry beans can help farmers in their production decisions, and it can also aid in the development of improved breeding strategies.

PROBLEM DEFINITION:

- The task is to develop a machine learning model that can accurately classify dry beans into their respective classes.
- The goal is to train a model that can accurately predict the class of new dry bean images that are not present in the dataset.
- The model will be evaluated based on its classification accuracy, and the best model will be selected for deployment.

IMAGES TO NUMBERS

TOTAL- 13,611

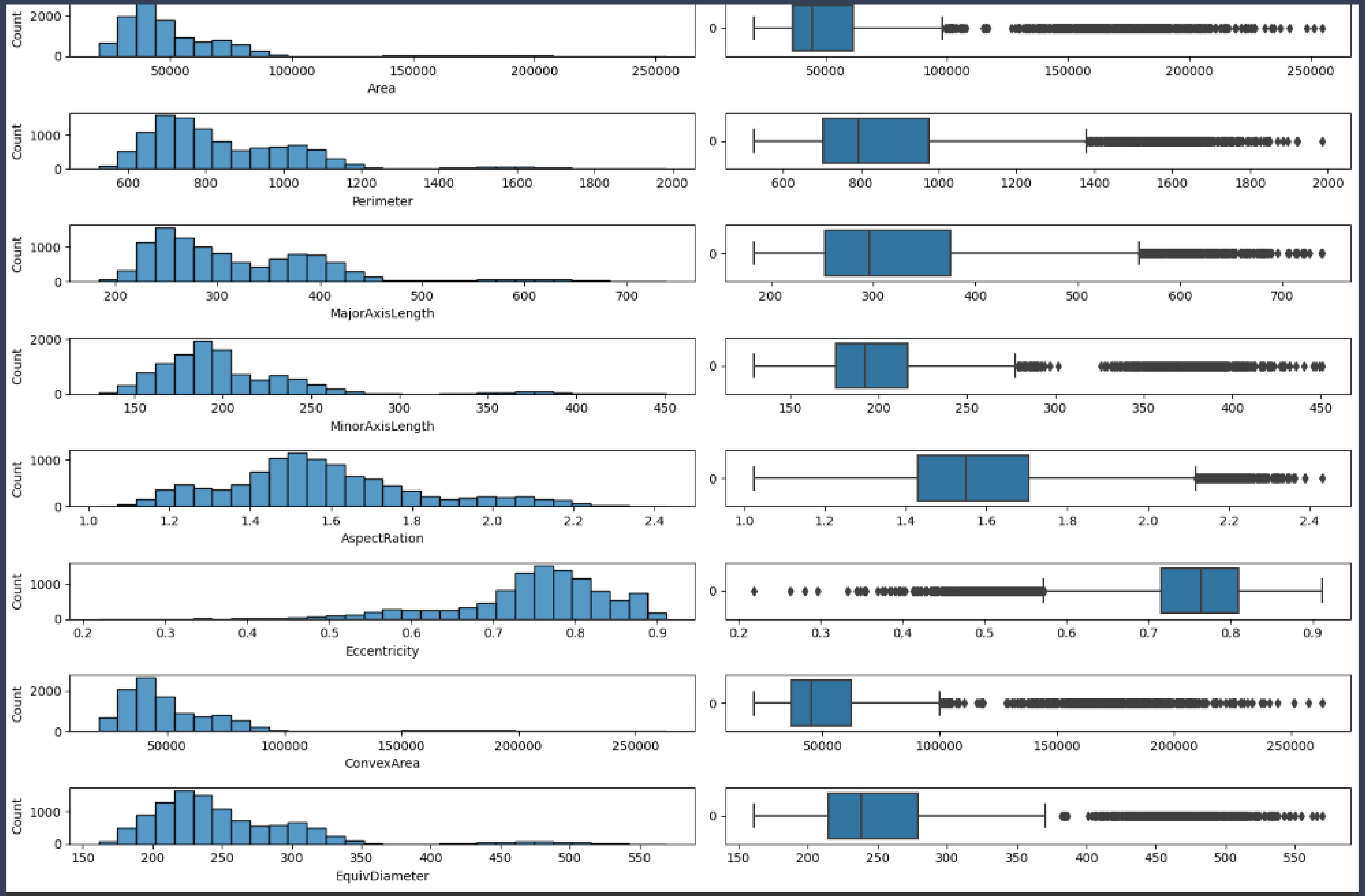
CLASSES-7

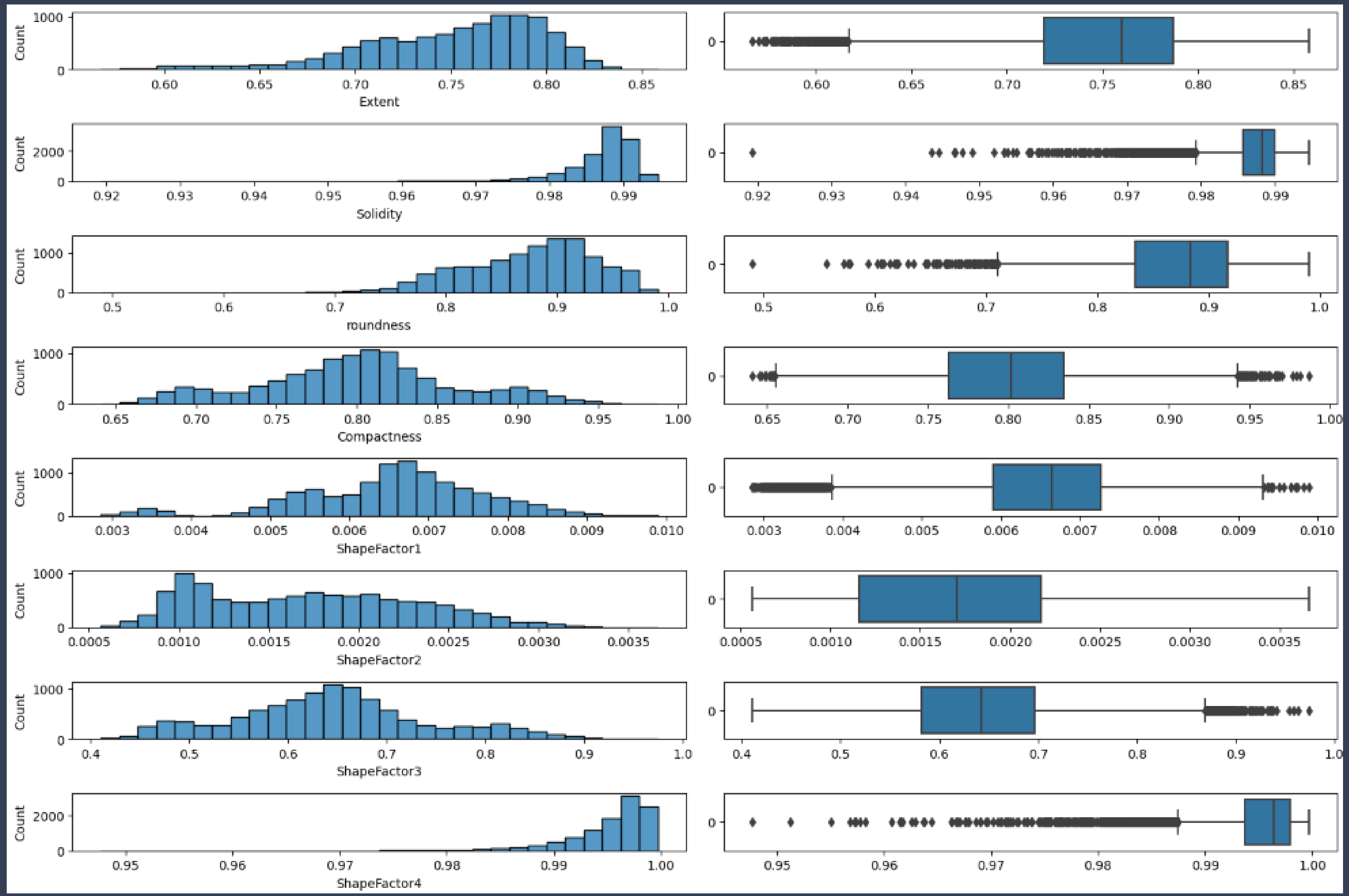
FEATURES-16

12-DIMENSIONS & 4 SHAPE FACTORS

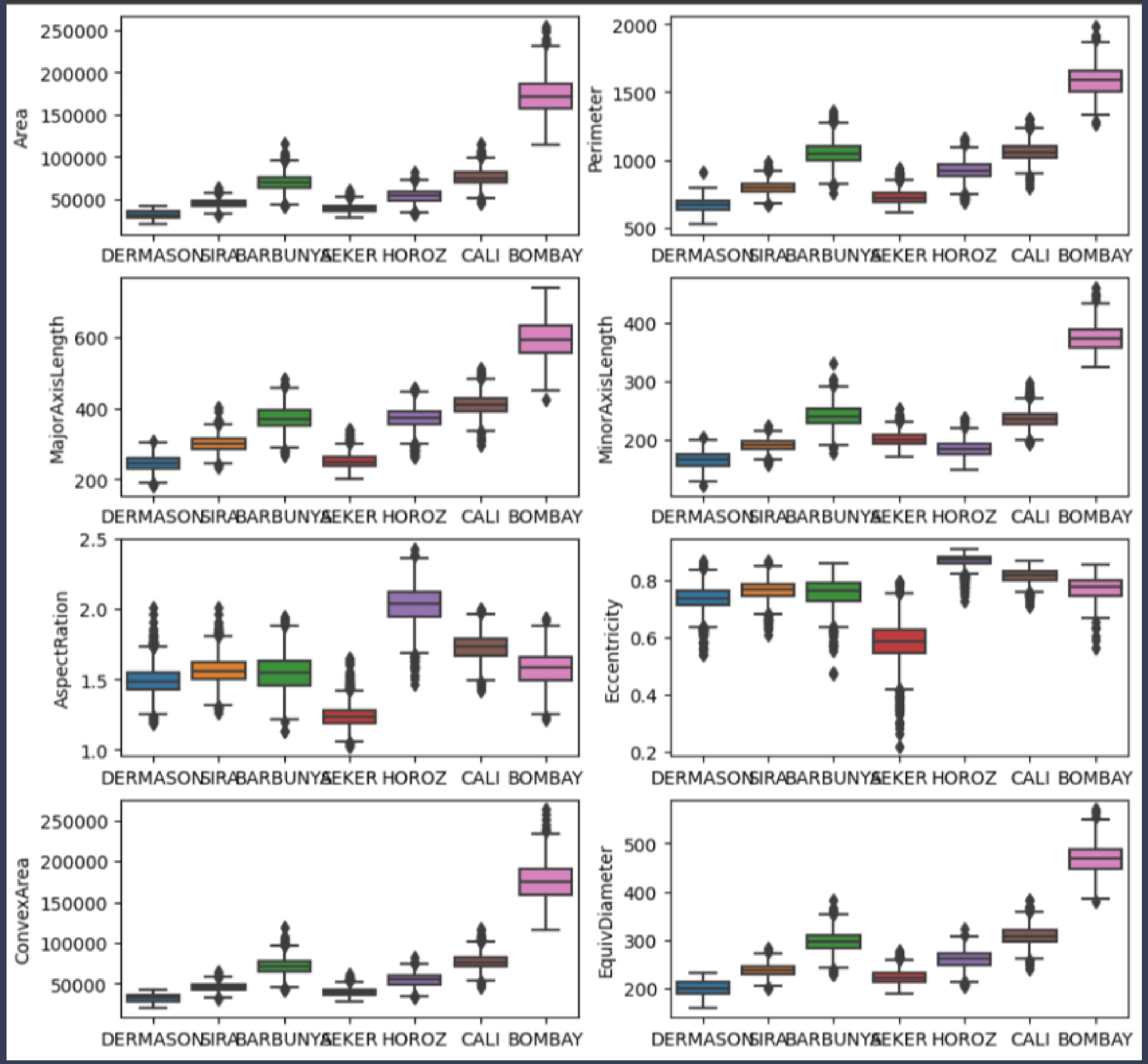
EXPLORATORY DATA ANALYSIS (EDA)

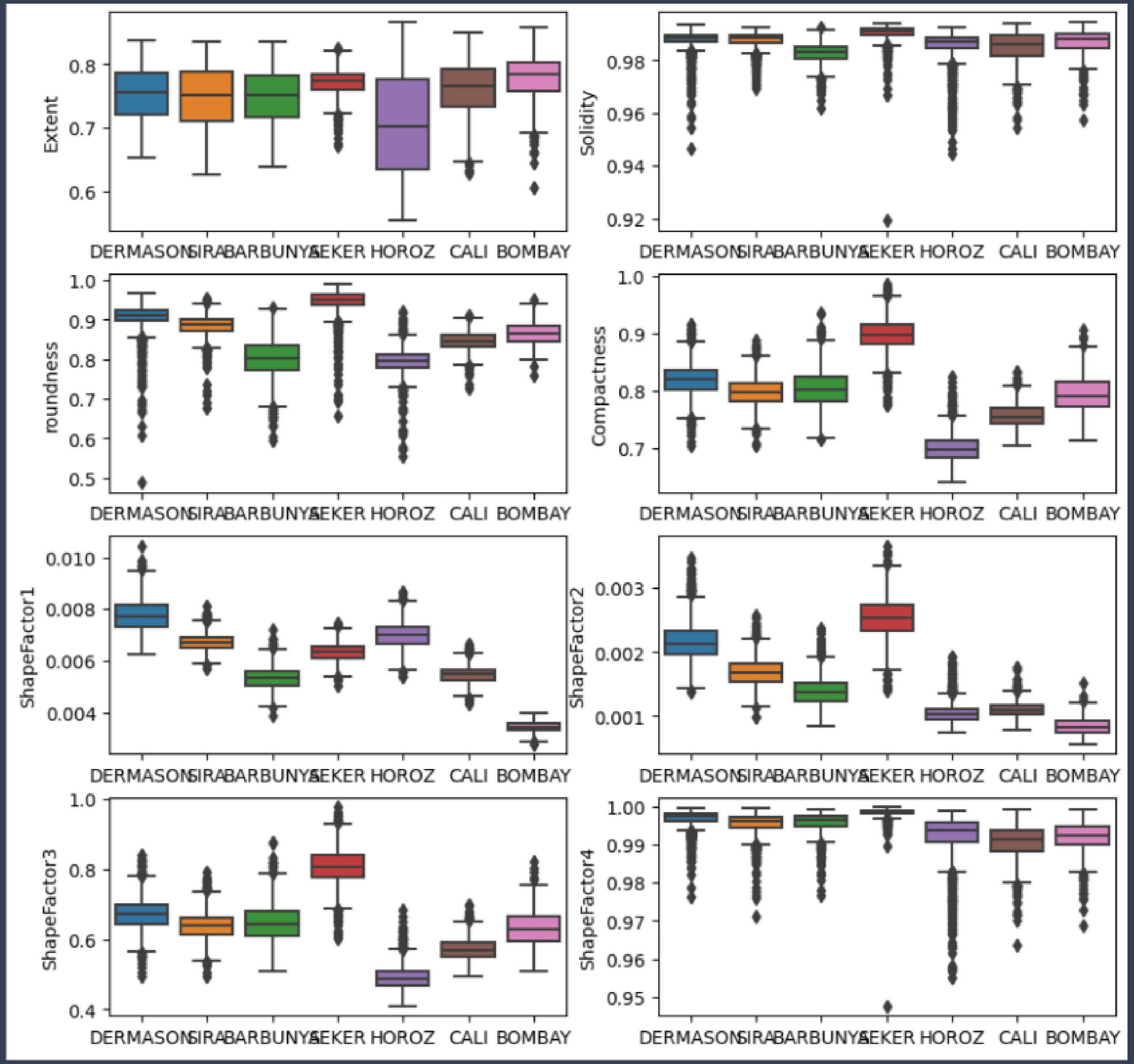
UNIVARIATE ANALYSIS





- Histogram of distributions for each feature





- Box plots of each feature for all response variables

INSIGHTS:

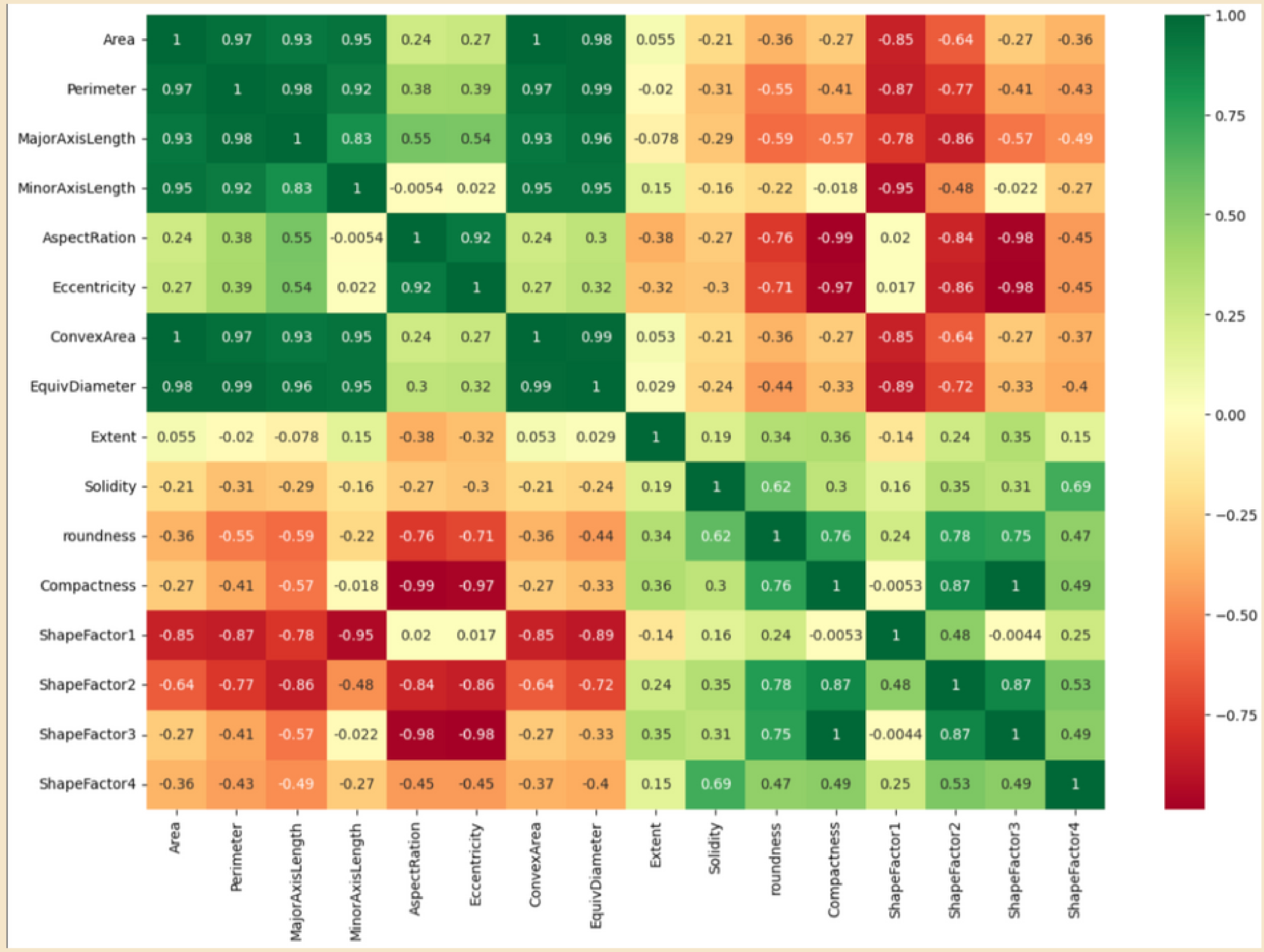
- Most of the features are **left or right skewed** and have a **lot of outliers**(long tail in eccentricity, solidity, roundness, shape factor2, shape factor4)

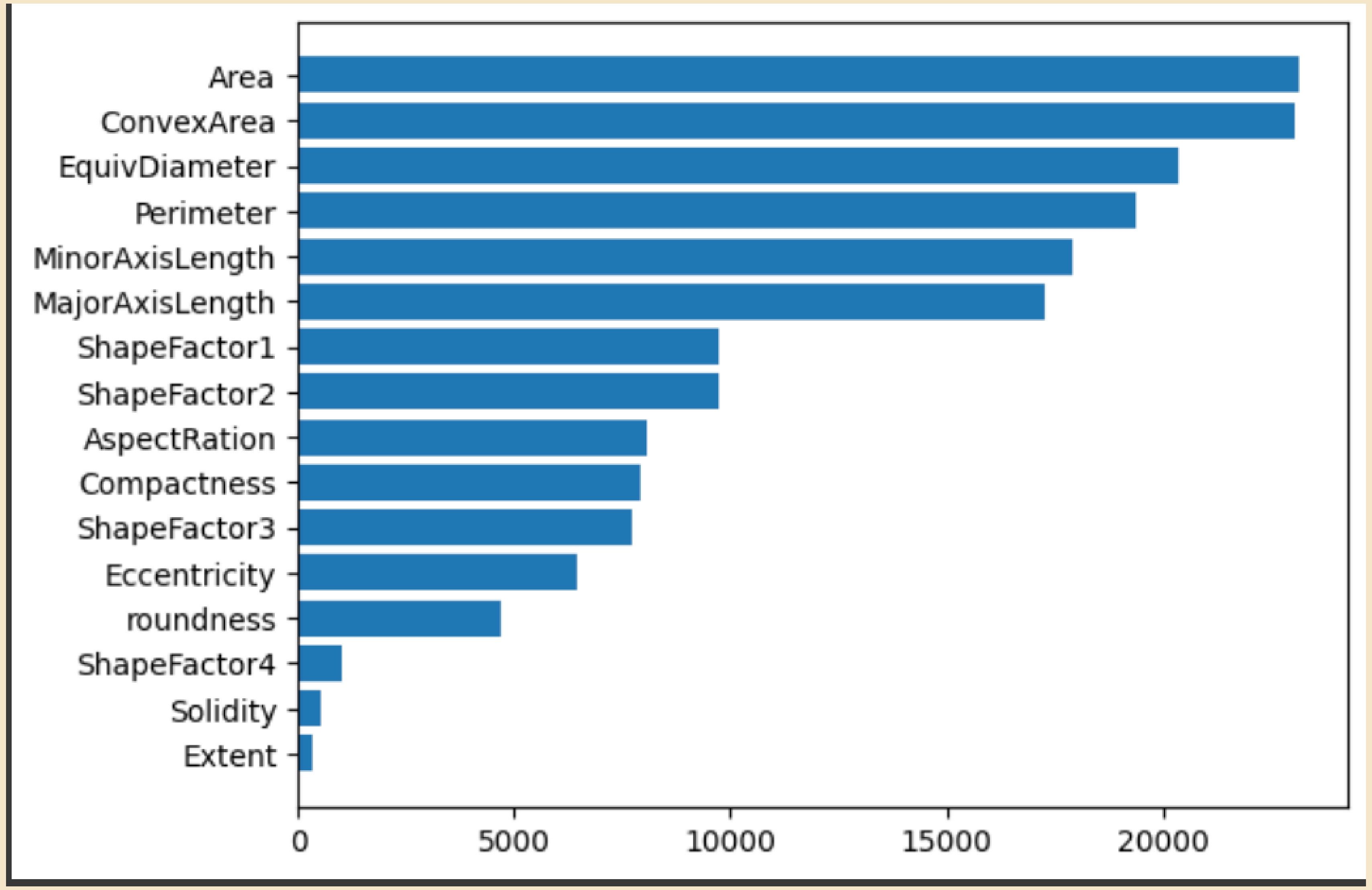
- W.r.t **area related features** (Area, perimeter, convex area, equvidistance, major axis), we can differentiate the **'Bombay' class**

- Both **Barbunya** class and **Cali** class have **similar distributions** and values in many features (area, minor axis length, equivalent diameter, extent, shape factor1), which may lead to mislabeling one as the other.

- Dermason class is similar to Seker class in some features, and Sira class in other features. It may be a difficult class to label accurately!

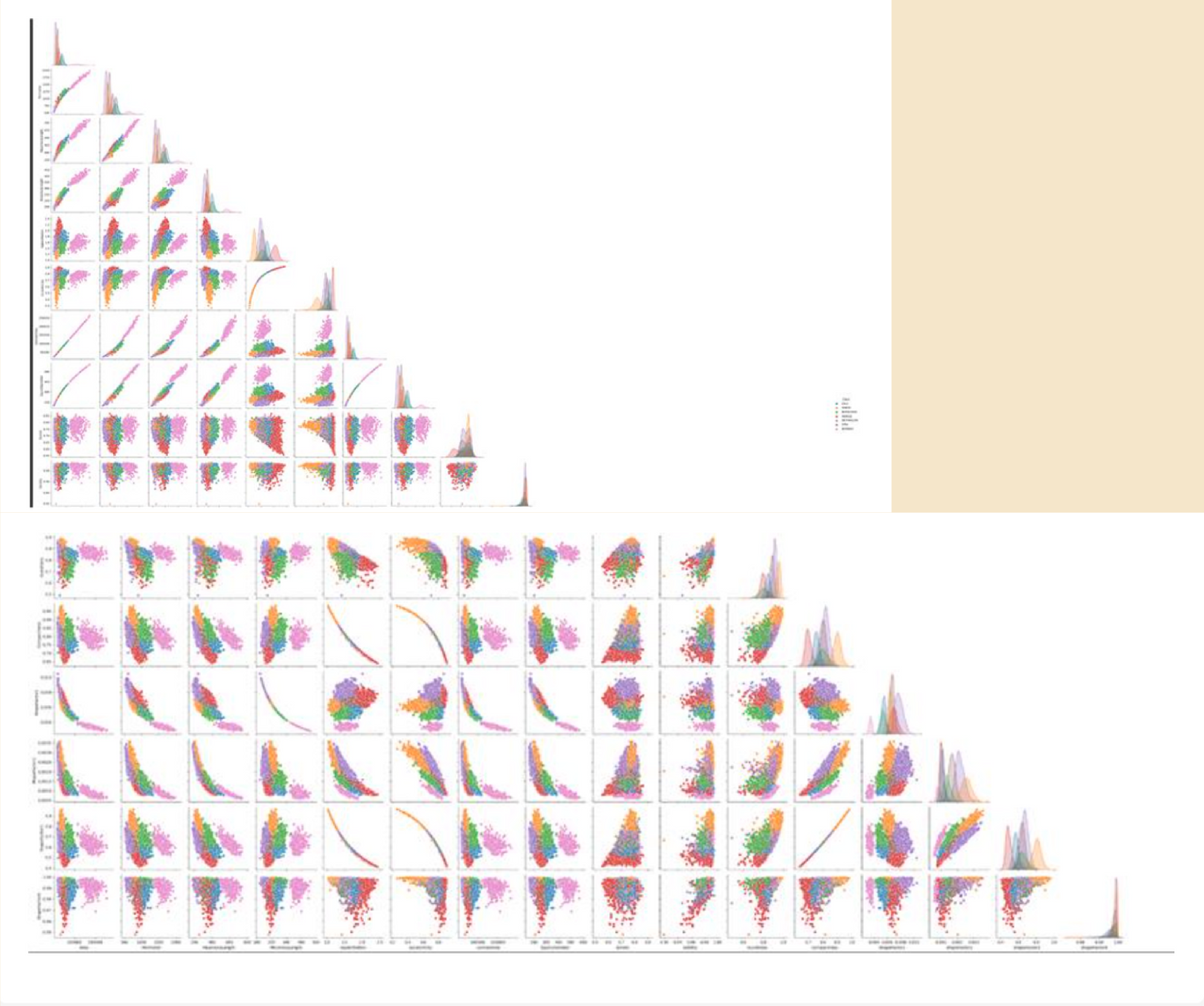
MULTIVARIATE ANALYSIS



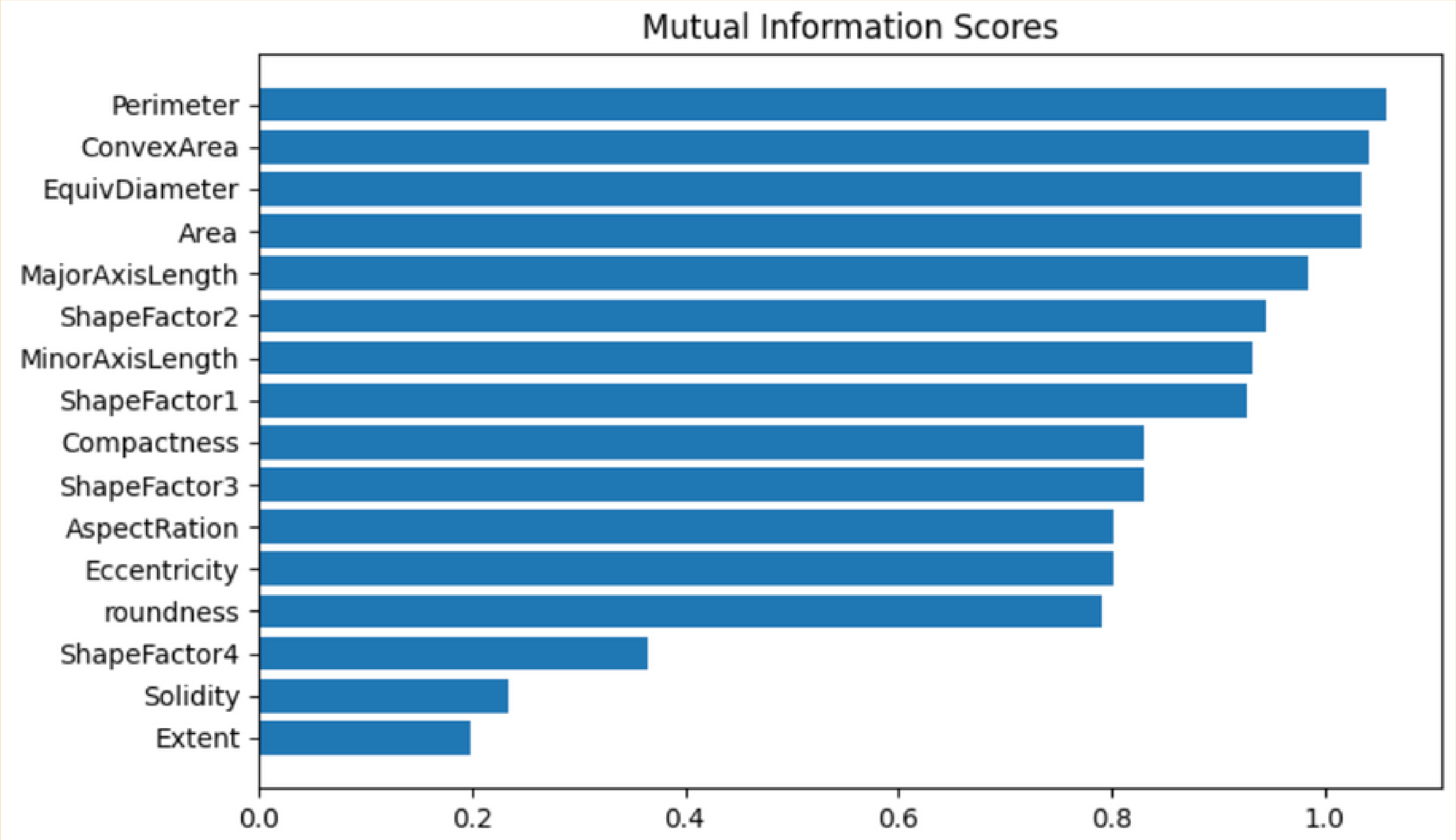


Correlation Matrix

ANNOVA / F-Test



Pairplots



Mutual Information

INSIGHTS:

- For correlation plots following pairs have highest correlation with each other:
  - area & convex area : 1.00
  - compactness & shape factor 3 : 1.00
  - equivalent diameter & perimeter: 0.99
  - equivalent diameter & convex area: 0.99
  - major axis length & perimeter: 0.98
  - area & perimeter: 0.97
  - convex area & perimeter : 0.97
  - major axis length & equivalent diameter : 0.96
  - minor axis length & equivalent diameter : 0.95
  - minor axis length & convex area : 0.95
  - minor axis length & shape factor 1 : -0.95
  - eccentricity & compactness : -0.97
  - eccentricity & shape factor 3 : -0.98
  - aspect ration & shape factor 3 : -0.98
  - aspect ration & compactness : -0.99
- From Mutual Information & F-test, we can see that following variables have least dependency w.r.t to response variable:
  - ShapeFactor4
  - Solidity
  - Extent

Feature Engineering

ADDING FEATURES

Adding features lead to high correlation between the existing and the new features added ,so we tend not to add any

REMOVING OUTLIERS

Eliminated all the outliers for cleaner data

FEATURE SELECTION

L1 norm Penalizing

```
Area -3.164968690693302e-05
Perimeter 0.007156237084468118
MajorAxisLength -0.005505639303736833
MinorAxisLength 0.011872562379017815
AspectRation -1.0639073144590105
Eccentricity 0.0
ConvexArea -2.7738555036235635e-05
EquivDiameter 0.005763542400065923
Solidity 0.0
roundness -5.0072708920517535
Compactness 0.0
ShapeFactor3 0.0
ShapeFactor4 0.0
```

- Same set of features are dropped as observed in multivariate analysis except for compactness and eccentricity

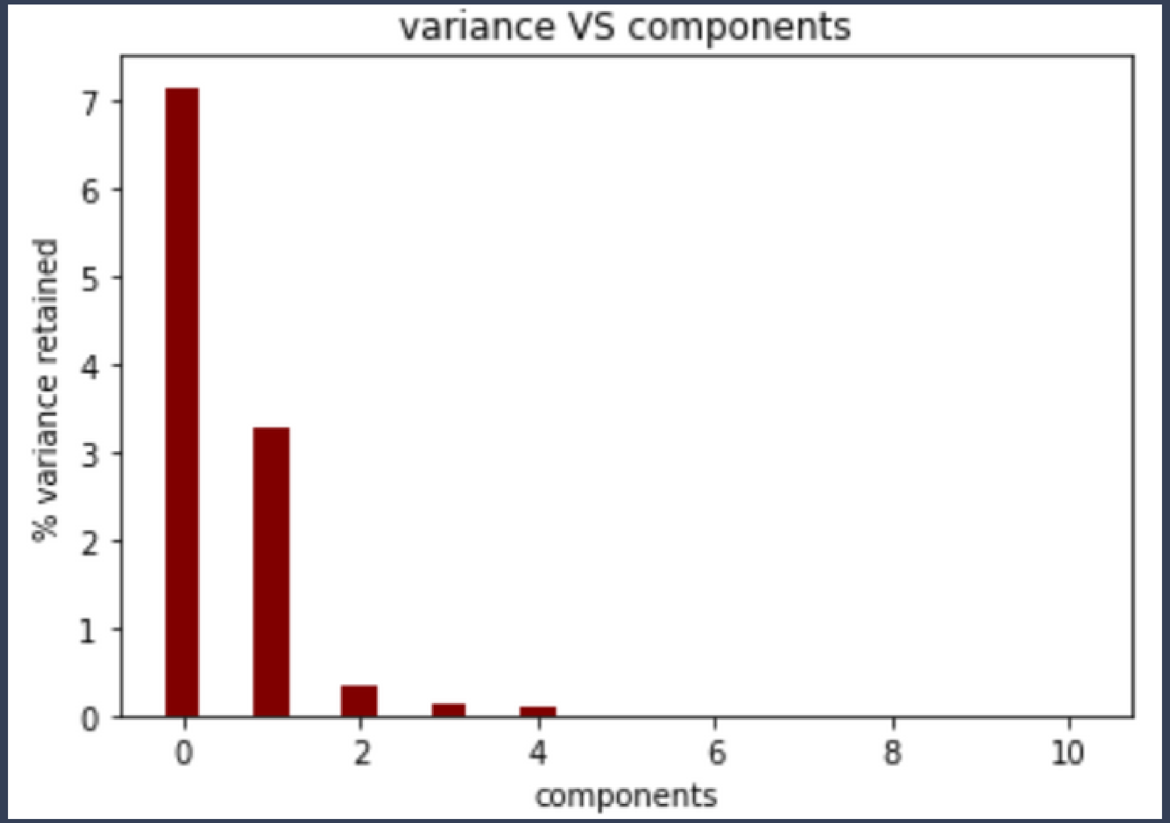
From Multivariate Analysis

Due to high correlation we drop following features:

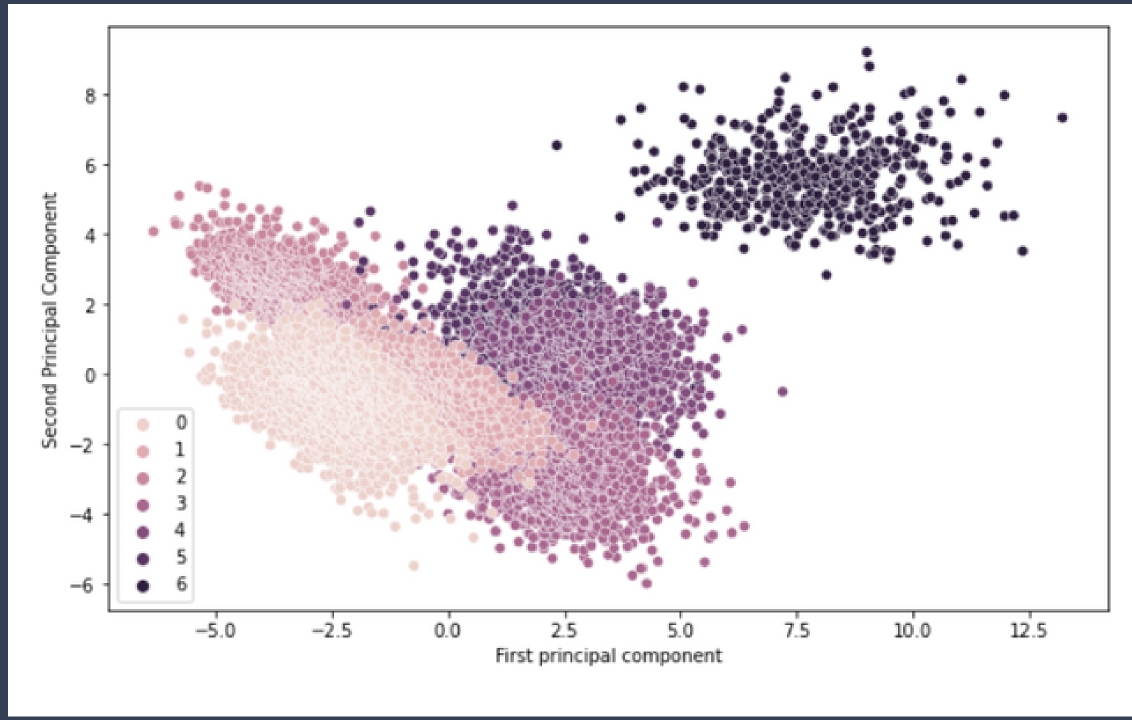
- ShapeFactor4',
- 'Solidity',
- 'Extent','
- ShapeFactor3',
- 'ConvexArea'

DIMENSIONALITY REDUCTION

PCA



- This plot shows the percentage of variance captured VS a number of components; we can see that the first 2-3 components cover most of the variance. The data selected over this plot is from univariate feature selection methods.



The plot of the PC1 vs PC2

Overall Insights before developing model

- We can observe linear relationship among several features.
- Notably, the Bombay class is distinguishable from other classes in some features, suggesting that a model may be able to accurately classify it, despite its small representation in the dataset.
- We can use SVM algorithm for linearly separable data
- To cater for the non-linear data we chose to go with multi-class(Logistic regression) and Neural Networks

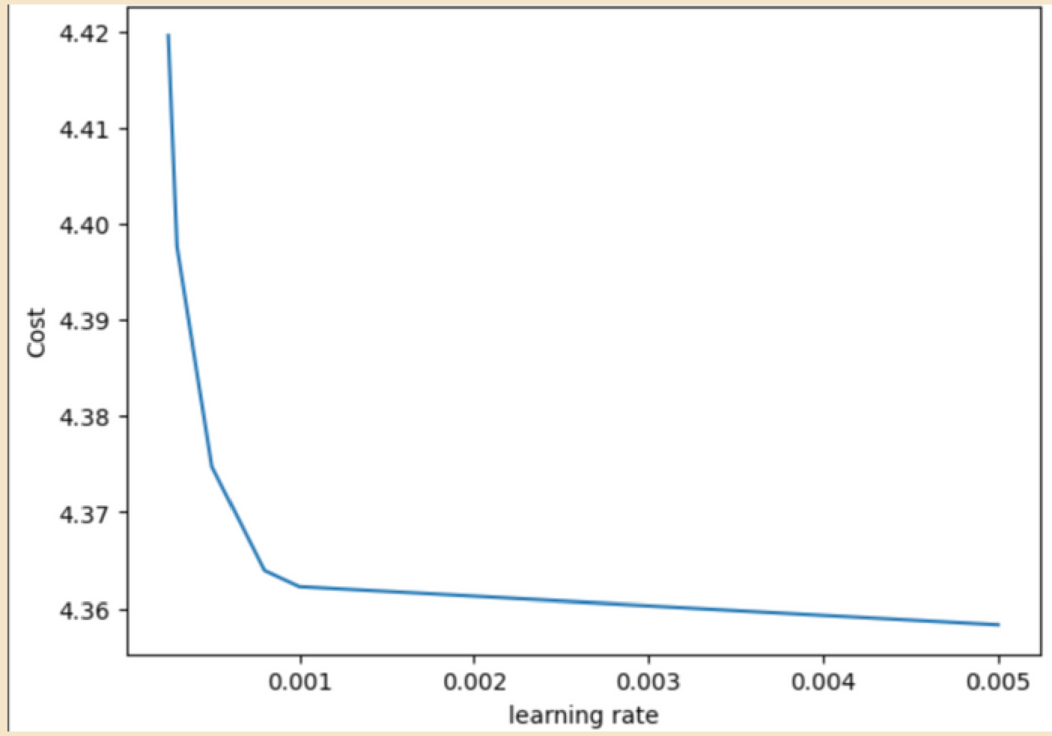
LOGISTIC REGRESSION

Reasons:

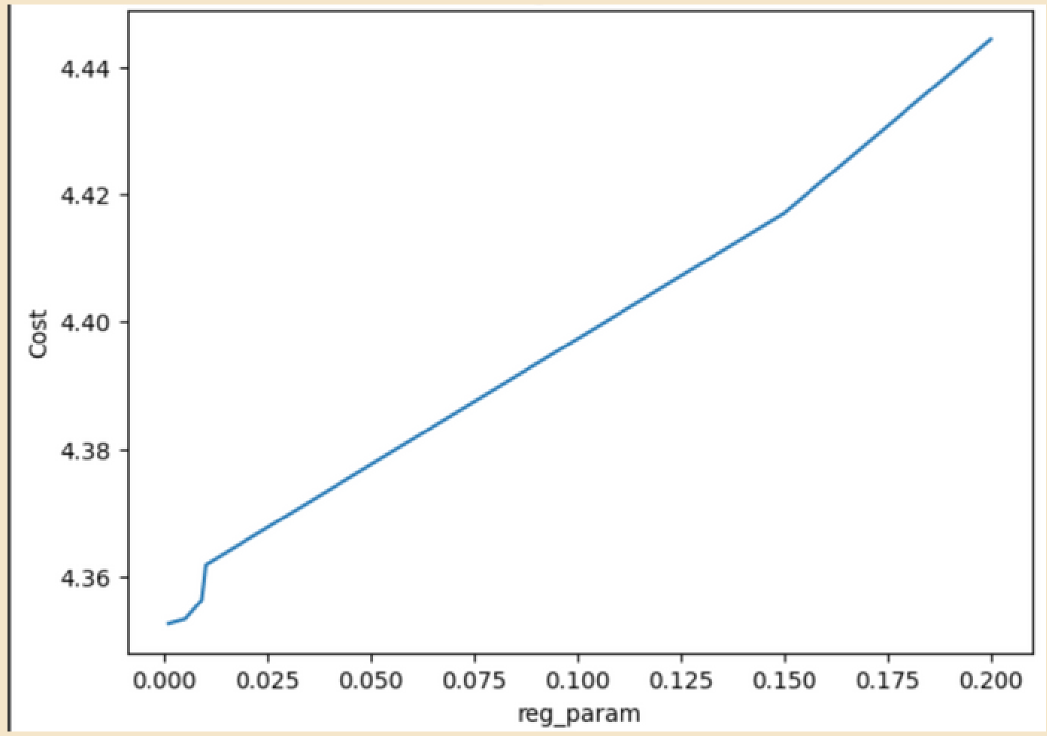
- They are extremely fast and simple, but on the other hand, their performance is usually limited.
- They can be used as a baseline for classification problems

Results:

- Accuracy: 0.8006856023506367
- Train error: 4.255709886642347
- Test error: 3.834306197603518



Learning Rate vs Cost



Reg_Param vs Cost

SVM

Reasons:

- Very powerful and flexible algorithm
- More accurate classification can be obtained

Results:

- 96 corrected out of 100
- Accuracy of SVM ON 100 samples=96%

Cons:

- Cannot implement for all our data
- Very costly,needs high computation power

NEURAL NETWORKS

Reasons:

- They help to group unlabeled data according to similarities among the example inputs

Results:

```
***confusion matrix***
[[ 220    0  121    0    6    6   19]
 [   3  140    0    0    0    0    0]
 [  75    0  391    0   13    0    6]
 [   0    0    0 1005    1   22   57]
 [   2    0   10    9  534    0   14]
 [   3    0    0    7    0  596   20]
 [   6    0    0   98   24   19  657]]
***classification report***
              precision    recall  f1-score   support

           0       0.71      0.59      0.65       372
           1       1.00      0.98      0.99       143
           2       0.75      0.81      0.78       485
           3       0.90      0.93      0.91      1085
           4       0.92      0.94      0.93       569
           5       0.93      0.95      0.94       626
           6       0.85      0.82      0.83       804

    accuracy                           0.87      4084
   macro avg       0.87      0.86      0.86      4084
weighted avg       0.87      0.87      0.87      4084

86.7531831537708
Test Accuracy: 86.7531831537708
train time: 0:00:00.031139
test time:  0:00:06.429996
```

- NN with Adam & Weight Decay

NEURAL NETWORKS

Results:

```
***confusion matrix***
[[ 224    0  115    0    6    6   21]
 [   3  140    0    0    0    0    0]
 [  82    0  384    0   13    0    6]
 [   0    0    0 1001    1   21   62]
 [   2    0   11    9  533    0   14]
 [   3    0    0    7    0  595   21]
 [   5    0    0   92   20   18  669]]
***classification report***
              precision    recall  f1-score   support

           0       0.70      0.60      0.65       372
           1       1.00      0.98      0.99       143
           2       0.75      0.79      0.77       485
           3       0.90      0.92      0.91      1085
           4       0.93      0.94      0.93       569
           5       0.93      0.95      0.94       626
           6       0.84      0.83      0.84       804

    accuracy                           0.87      4084
   macro avg       0.87      0.86      0.86      4084
weighted avg       0.87      0.87      0.87      4084

86.82664054848188
Test Accuracy: 86.82664054848188
train time: 0:00:00.080413
test time:  0:00:07.193343
```

- NN with RMSprop and L2 norm

```
***confusion matrix***
[[ 190    0    6    7   74   80   15]
 [ 102    0    0    0    0   41    0]
 [ 235    0    3    2  238    3    4]
 [   0    0    0 1079    0    6    0]
 [   1    0    0   28  539    0    1]
 [   0    0    0   64    1  559    2]
 [   0    0    0  705   56   24   19]]
***classification report***
              precision    recall  f1-score   support

           0       0.36      0.51      0.42       372
           1       0.00      0.00      0.00       143
           2       0.33      0.01      0.01       485
           3       0.57      0.99      0.73      1085
           4       0.59      0.95      0.73       569
           5       0.78      0.89      0.83       626
           6       0.46      0.02      0.04       804

    accuracy                           0.58      4084
   macro avg       0.44      0.48      0.40      4084
weighted avg       0.52      0.58      0.47      4084

58.496571988246814
Test Accuracy: 58.496571988246814
train time: 0:00:00.050796
test time:  0:09:26.794456
```

- NN with Basemodel

Comparison between NN

```
                                 Accuracy using PCA and Univariate FS
                                   ----------
Adam_using_weight_decay            : 86.75%
RMSprop_using_L2                   : 86.83%
Base_model_without_optim_and_minibatches  : 58.5%

                                  Train Time
                                  -----------
Adam_using_weight_decay                 : 0:00:06.429996
RMSprop_using_L2                        : 0:00:07.193343
Base_model_without_optim_and_minibatches  : 0:09:26.794456
```

CONCLUSION

- From the results above, we can conclude that SVM and Neural Networks give the best results
- For Neural Networks(RMSprop & L2norm, Adam & Weight Decay), the time taken is least while also providing great results.
- SVM provides us with the best accuracy i.e, of 96%

Thank you