IC SOLUTIONS

INTERNSHIP PROJECT REPORT

ON

"CLASSIFICATION OF DRY BEANS"

Submitted by

S.NO	NAME	USN	EMAIL
1.	PARISMITA DEVI	1GA18CS103	deviparismita98@gmailcom
2.	RASHMI KESHARI	1JS18CS128	kesharirashmi18@gmail.com

UNDER GUIDANCE OF

ABHISHEK C

Acknowledgement

We cannot begin to express our thanks to the **IC Solutions** company who provided us such a great internship opportunity in PYTHON ML online in this pandemic situation at really affordable cost with a hands-on live Project. We are very grateful for having such a chance for being the part of this opportunity.

Then we would like to extend our sincere thanks to our instructor **Abhishek C**, for giving such a good knowledge about Python essentials, Artificial Intelligence, Machine Learning etc. from basics during the training and giving his vital support throughout the internship which really helped us in successfully doing our Projects.

We really had a great learning experience with this Internship.

Abstract

The Project involves to analyse the data of the given dataset and to find the accuracy scores of it using various classification algorithms which uses the concept of Python, Data Analysis, Data Visualization and Machine Learning

The dataset given is “Dry_Bean_Dataset - Altered” which contains the Pictures of 13,611 grains of seven different types of dry beans, taken with a high-resolution camera. The data include 17 columns (17 attributes) in which 12 are dimensions (like Area, Perimeter, MajorAxisLength, MinorAxisLength etc.), 4 are shape forms (ShapeFactor1, ShapeFactor2, ShapeFactor3, ShapeFactor4) and last is a Class attribute and 13,611 number of rows containing data of those attribute where some of the data are missing.

The process of the Project starts from Handling Missing values by dropping the particular row or filling it then plotting the graphs of different attributes against each other for visualizing the data and then dividing the dataset into two parts first the numerical part and second the class values, then comes using the StandardScalar of preprocessing for standardizing and transforming the data. The train/test is a method to measure the accuracy of a model, in this method we split the data into two sets, so the next step of the process is splitting the dataset into training set and testing set where training phase to create the model and testing phase to find the accuracy score using classification algorithm to classify the dry beans to its among its seven class.

About the Company

ICS is a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses. At ICS, we believe that service and quality is the key to success.

We provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that you may have. Experience the service like none other!

Some of our services include:

Development - We develop responsive, functional and super-fast websites. We keep User Experience in mind while creating websites. A website should load quickly and should be accessible even on a small view-port and slow internet connection.

Mobile Application - We offer a wide range of professional android, iOS & Hybrid app development services for our global clients, from a start up to a large enterprise.

Design - We offer professional Graphic design, Brochure design & Logo design. We are experts in crafting visual content to convey the right message to the customers.

Consultancy - We are here to provide you with expert advice on your design and development requirement.

Videos - We create a polished professional video that impresses your audience.

Index

Introduction

This Python ML internship is provided by the IC Solutions company, which is conducted on an online platform GOOGLE MEET for all semesters and branches of students. It is one-month internship with one-week training for 2 hours daily along with it.

The content covered during the training from basics are:

- 'Intro to AI' which gave a basic knowledge from its Birth to what is AI (Artificial Intelligence), why we use AI, success of AI etc.
- 'Python Essentials' includes the overview of jupyter notebook, data types in Python, Operators, if elif else statements, for and while loop, list Comprehension, Functions etc.
- 'Data Analysis Libraries' NumPy and Pandas, how to install it and how to code with it.
- 'Data Visualization Libraries' Matplotlib and Seaborn, how to install it and how to visualize data in the form of graphs from it.
- 'Intro to ML' which gave the basic knowledge about is what is Machine Learning followed by all ML algorithms, Linear Regression, Logistic Regression, Support Vector Machines, Decision Trees, Random Forest, Bais – Variance, K means, PCA (Principal Component Analysis).

Training ended with the Project topic which is a Dry_Bean_Dataset having the details (dimensions, shapes, class) of the images of the grains of the seven registered dry bean were taken with a high resolution camera for which we have to apply the classification algorithm to classify each grains to its respective class of dry bean, which has to be done to successfully complete this internship.

Problem Statement and Objective

Problem Statement:

Given a dataset "Dry_Bean_Dataset - Altered" which consists the details (like dimensions, shapes) of the grains from seven different types of dry beans and to classify the grains to its specific class of dry bean we will use classification algorithm and so we have to calculate the accuracy score of all the classification algorithm and find which is algorithm is having the highest accuracy score to for classification.

Objective:

- Import the libraries NumPy, pandas, matplotlib etc. required.
- Then import the dataset (Dry_Bean_Dataset – Altered) stored in the form of matrix.
- Then data cleaning of the missing values by using iterative imputer.
- Plot the graphs by analysing the data of the attributes against each other.
- Dividing the data in X and Y, numerical values in X and Y contains the class values.
- Then using StandardScalar function from preprocessing for standardizing and transforming the data in such a way that the mean of the transformed data is 0 and Variance is 1.
- Then train/test method for creating the model(training phase) and then finding the accuracy score of the model (testing phase) of all classification algorithms.

Requirement Specification

Hardware Requirements

- 4 GB RAM
- Pentium IV or higher processor
- 1 GB Hard free drive space

Software Requirements

- Anaconda IDE for Jupyter Notebook
- Python Compiler
- NumPy, Pandas, ScikitLearn, Seaborn and Matplotlib

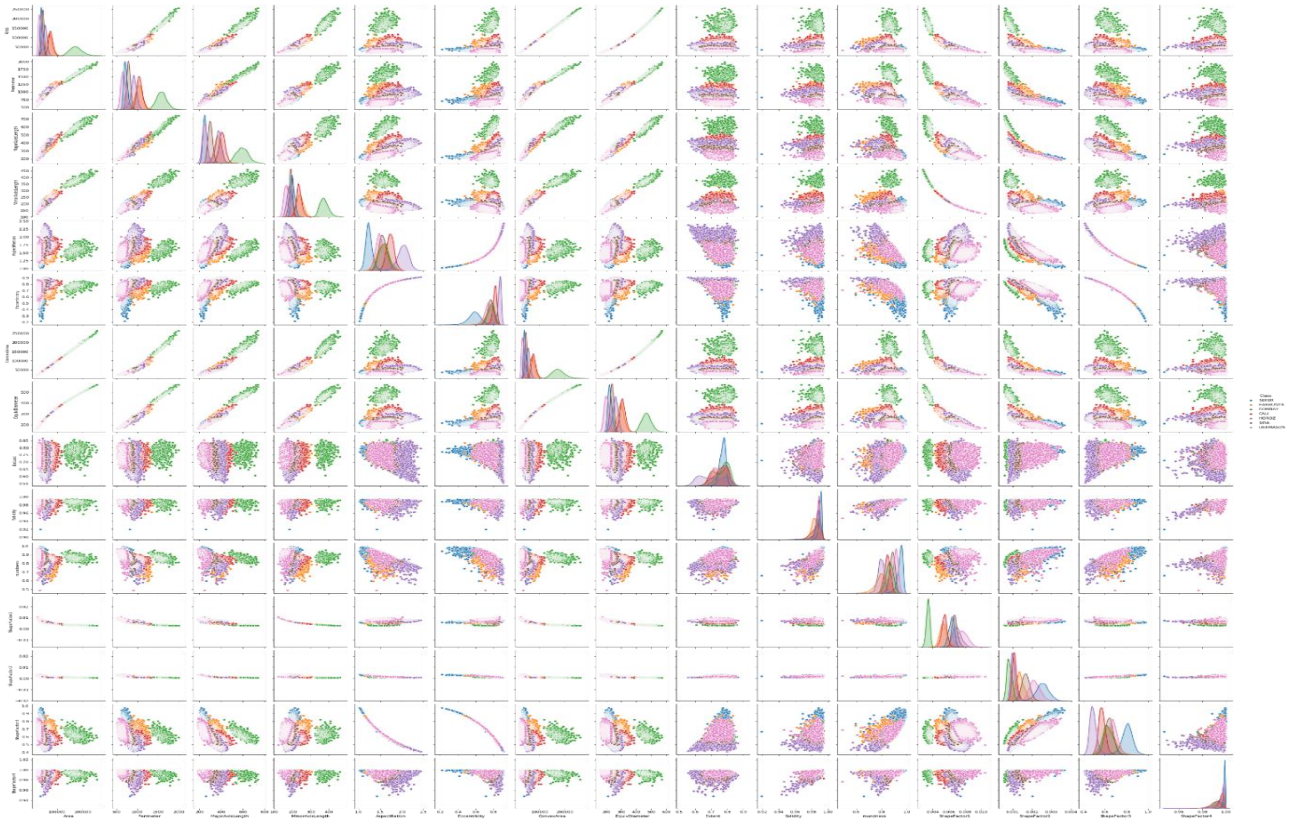
Exploratory data Analysis

Data Cleaning:

```
from sklearn.experimental import enable_iterative_imputer

from sklearn.impute import IterativeImputer

imp = IterativeImputer(random_state=0)

imp.fit(df[['Area','Perimeter','MajorAxisLength','MinorAxisLength','AspectRation','Eccentricity','ConvexArea','EquivDiameter','Extent','Solidity','roundness','ShapeFactor1','ShapeFactor2','ShapeFactor3','ShapeFactor4']])

IterativeImputer(random_state=0)

df2=df[['Area','Perimeter','MajorAxisLength','MinorAxisLength','AspectRation','Eccentricity','ConvexArea','EquivDiameter','Extent','Solidity','roundness','ShapeFactor1','ShapeFactor2','ShapeFactor3','ShapeFactor4']]

df2 = imp.transform(df2)


df2=pd.DataFrame(df2,columns=['Area','Perimeter','MajorAxisLength','MinorAxisLength','AspectRation','Eccentricity','ConvexArea','EquivDiameter','Extent','Solidity','roundness','ShapeFactor1','ShapeFactor2','ShapeFactor3','ShapeFactor4'])


df2['Class'] = list(df["Class"])
```

Graphs for the dataset after data cleaning:

Plot 1: -

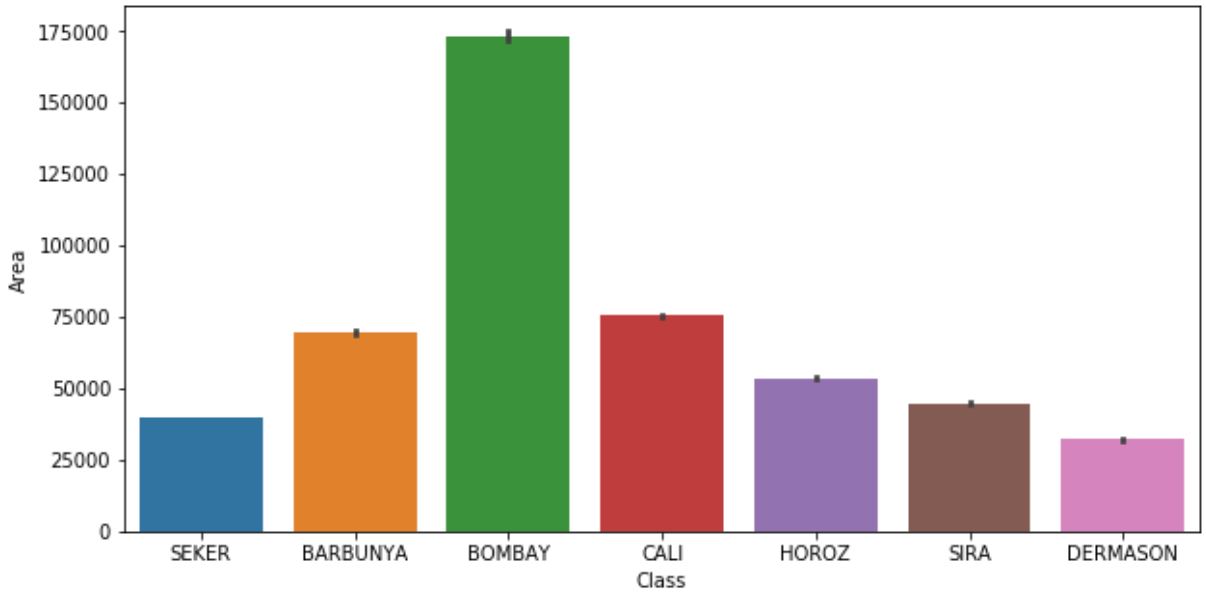


Code- sns.pairplot(df2, hue='Class')

It uses the library named Seaborn of Python. Here it particularly plots the graph called pairplot. It plots the graph of all the features given in dataset against each other.

- The Bombay bean (green cluster) have the most distinct variation as compared to other dry bean classes.
- The Dermason bean (pink cluster) is seen to be nearly distinct in case of features like Extent, ShapeFactor1, ShapeFactor2, ShapeFactor3 and ShapeFactor4.
- ShapeFactor1 and ShapeFactor2 remains nearly constant against given variation of all other given features.

Plot 2: -



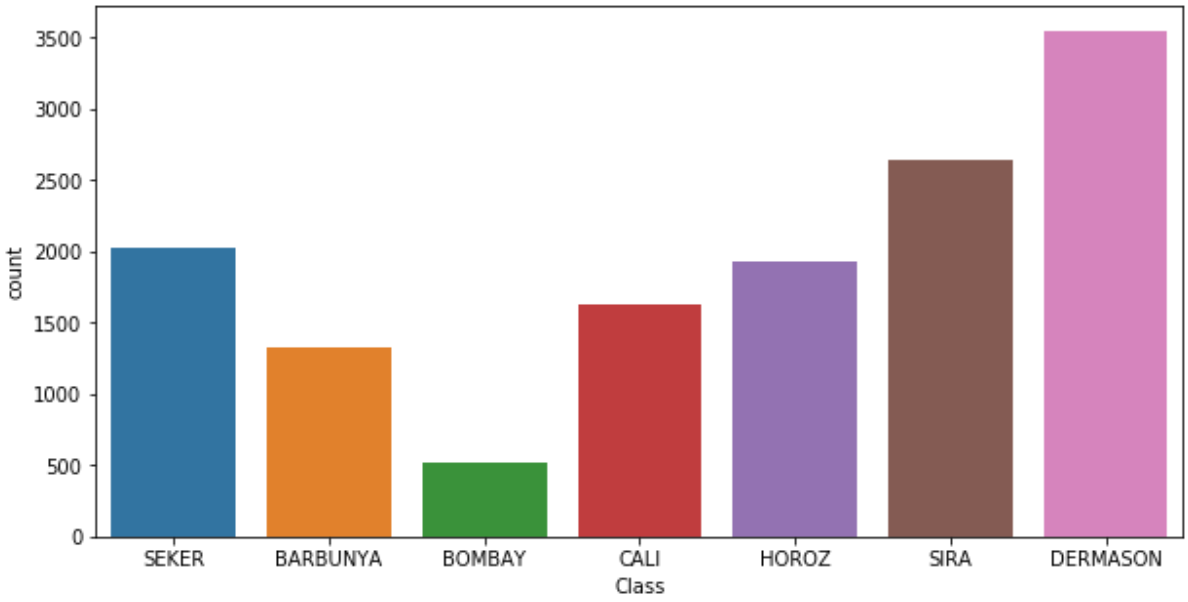
Code-plt.figure(figsize=(10,5))

sns.barplot(x='Class',y='Area',data=df2)

This uses the barplot function in Seaborn library in Python. It plots a bar graph of the maximum area of all the given dry bean categories.

- It can be noticed that BOMBAY dry bean have the highest area recorded than all other categories. The area of it have a very large difference than other categories.
- The area flow from larger to smaller goes like CALI, BARBUNYA with almost similar area range around 75000 followed by HOROZ, SIRA, SEKER and DERMASON which is of the smallest area.

Plot 3: -



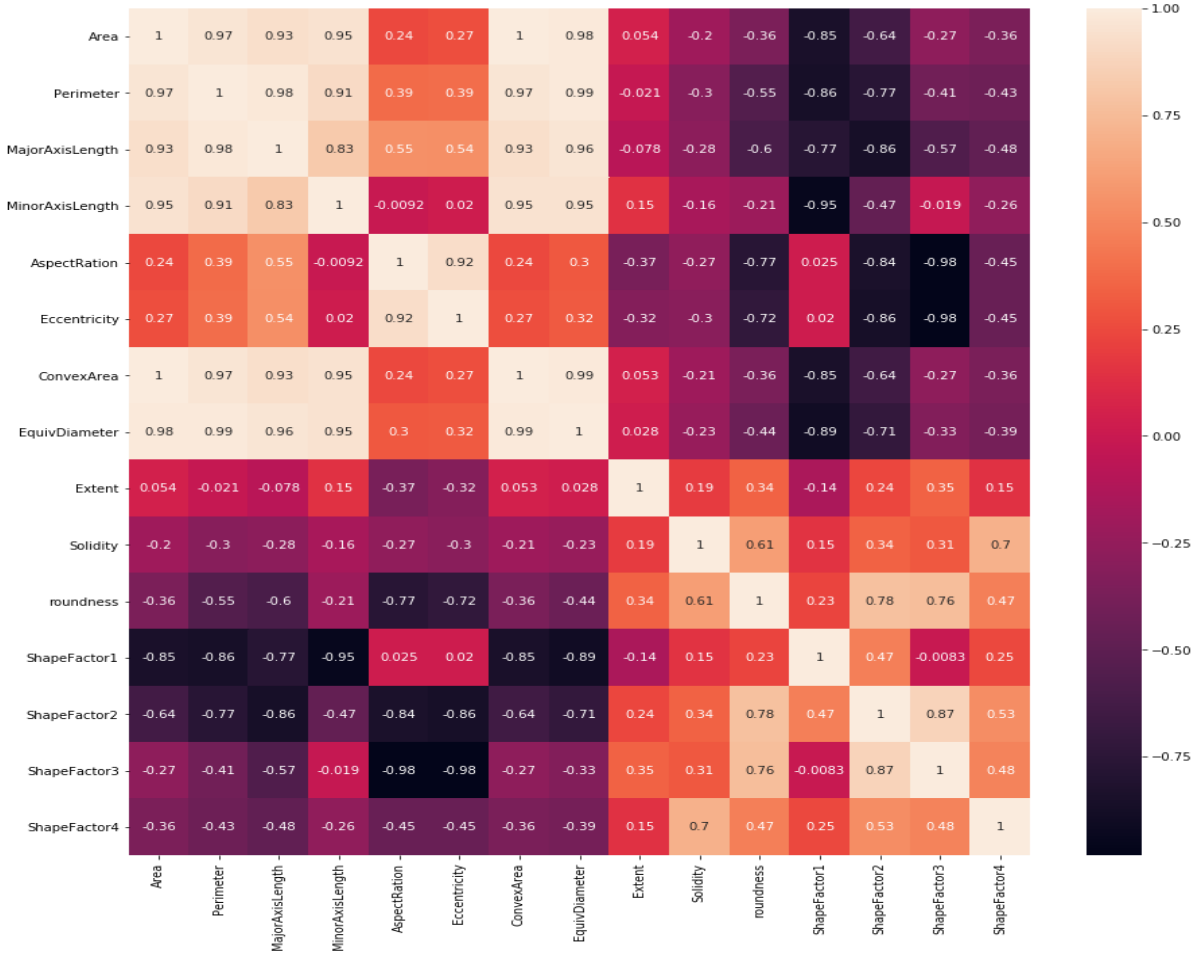
Code- plt.figure(figsize=(10,5))

sns.countplot(x='Class',data=df2)

It uses the countplot function from Seaborn library of Python. It gives a bar graph plot of the count of total data in a particular class.

- We can see that data was collected in large number from DERMASON dry bean class, so this class of dry bean is more prominently available.
- BOMBAY dry beans are the least in count in the given dataset, so this category of dry bean may be very rare to collect.
- From the previous plot it can be noticed that BOMBAY and DERMASON have the largest and smallest area respectively among all categories. All the observations may conclude that there is an inverse relation of area of dry bean to its availability.

Plot 4: -



Code- plt.figure(figsize=(15,15))

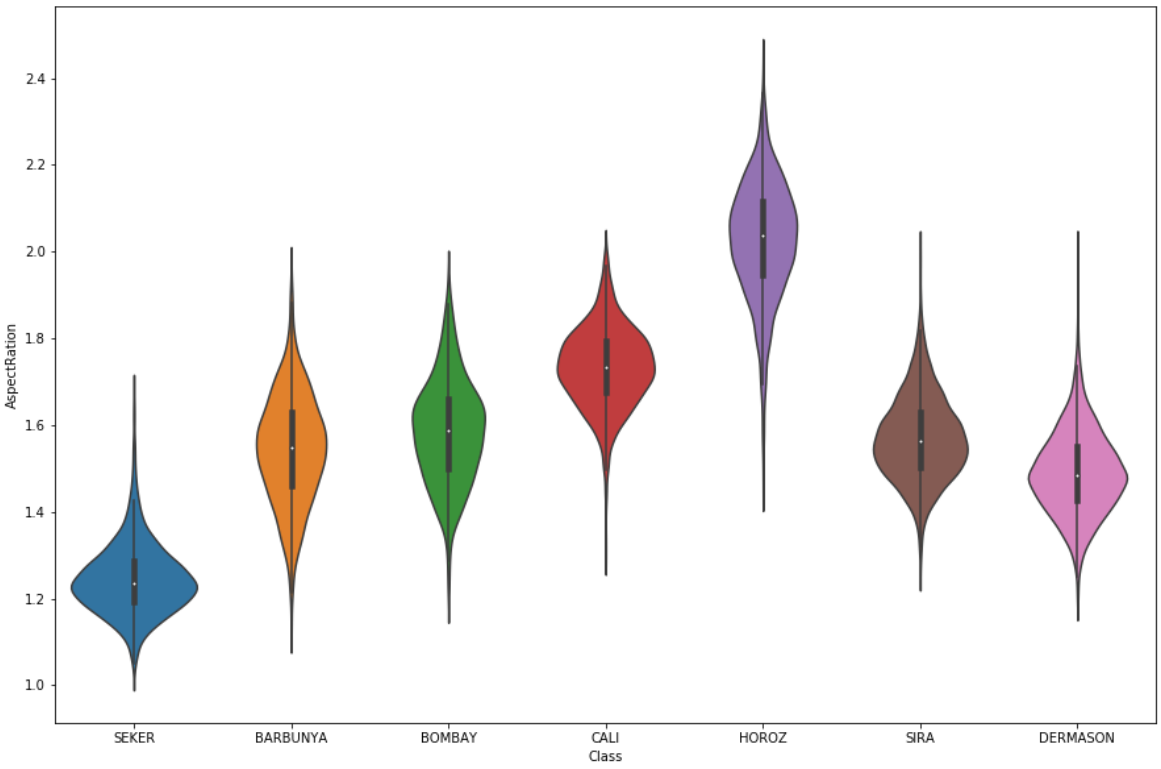
sns.heatmap(df2.corr(), annot=True)

A heatmap is a graphical representation of data that uses a system of color-coding to represent different values. The variation in color could maybe be in hue or intensity which clearly depicts the relationship between one parameter to another.

- It can be observed that eccentricity and aspect ratio are highly correlated to ShapeFactor3 with a score of -0.98. Increase of one factor leads to the decrement of the other.

- It is also seen that Perimeter and ConvexArea are positively correlated to EquivDiameter with a score of 0.99. increase of one factor leads to the increment of the other.
- Eccentricity and roundness are correlated to each other with a factor of -0.72.

Plot 5:-



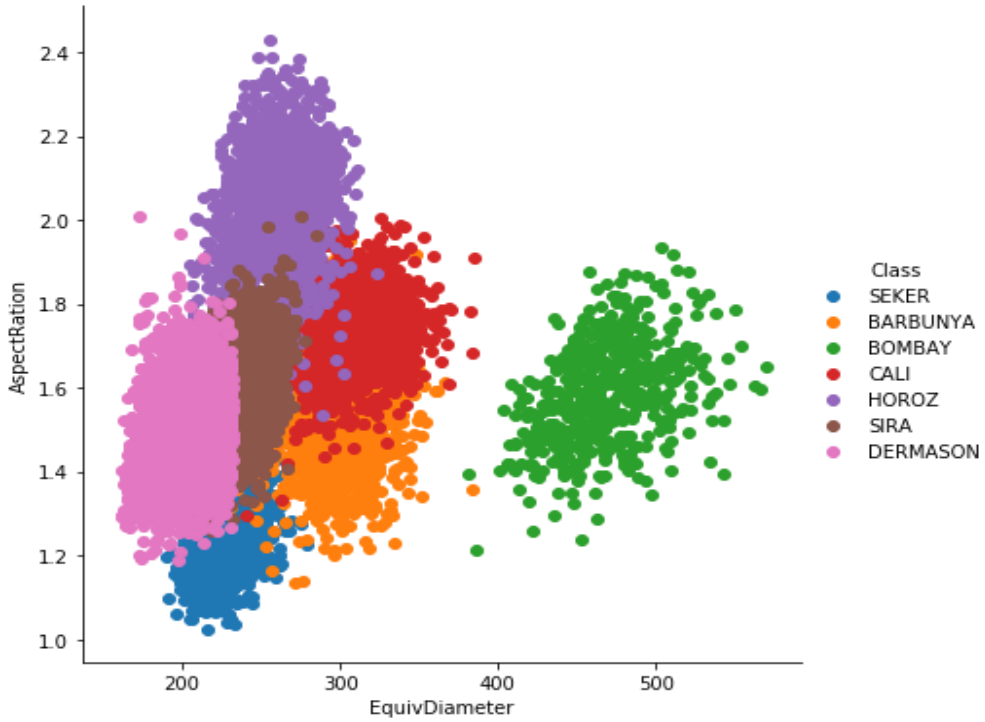
Code- plt.figure(figsize=(15,10))

sns.violinplot(x='Class', y='AspectRation', data=df2)

A violin plot is a method of plotting numeric data. These are similar to box plots, except that they also show the probability density of the data at different values. Here it is plotted for aspect ratio of each dry bean class. The aspect ratio of a geometric shape is the ratio of its sizes in different dimensions.

- The HOROZ dry bean class is seen with the highest AspectRatio around 18-22 among all other bean classes.
- SEKER class is with the lowest aspect ratio range. Most of the beans in this class have a ratio of around 12.
- Though BOMBAY and DERMASON had visible contrast in case of area with being highest and lowest among all, the AspectRatio of both were seen to be nearly similar.

Plot 6:-



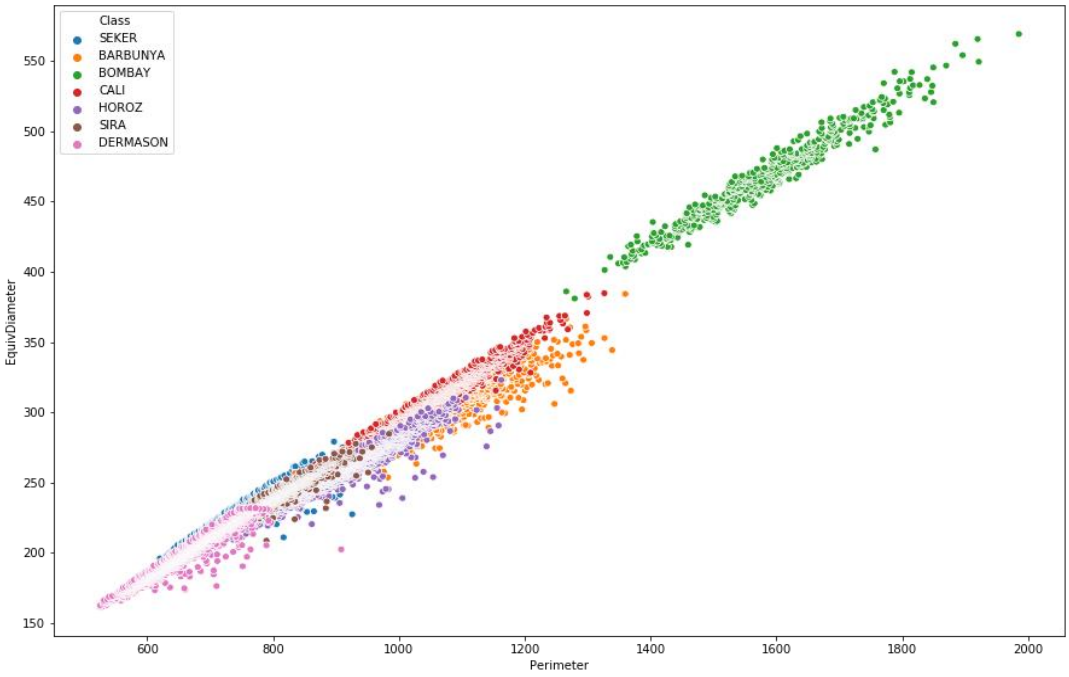
Code- plt.figure(figsize=(25,20))

sns.FacetGrid(df2,hue="Class",height=6).map(plt.scatter,'EquivDiameter','AspectRation').add_legend()

This class maps a dataset onto multiple axes arrayed in a grid of rows and columns that correspond to levels of variables in the dataset. This class helps in visualizing distribution of one variable as well as the relationship between multiple variables separately within subsets of your dataset using multiple panels.

- It is visible that most of the beans are of EuivDiameter between 200-350 except the bean BOMBAY which is seen to be in the range of 400-500.
- The BOMBAY bean from all previous graphs plotted above can be seen to have the most distinct feature among all other classes of beans. This can be easy to identify a BOMBAY bean.
- HOROZ bean is with the highest AspectRatio of all can be concluded again from this graph.

Plot 7:-



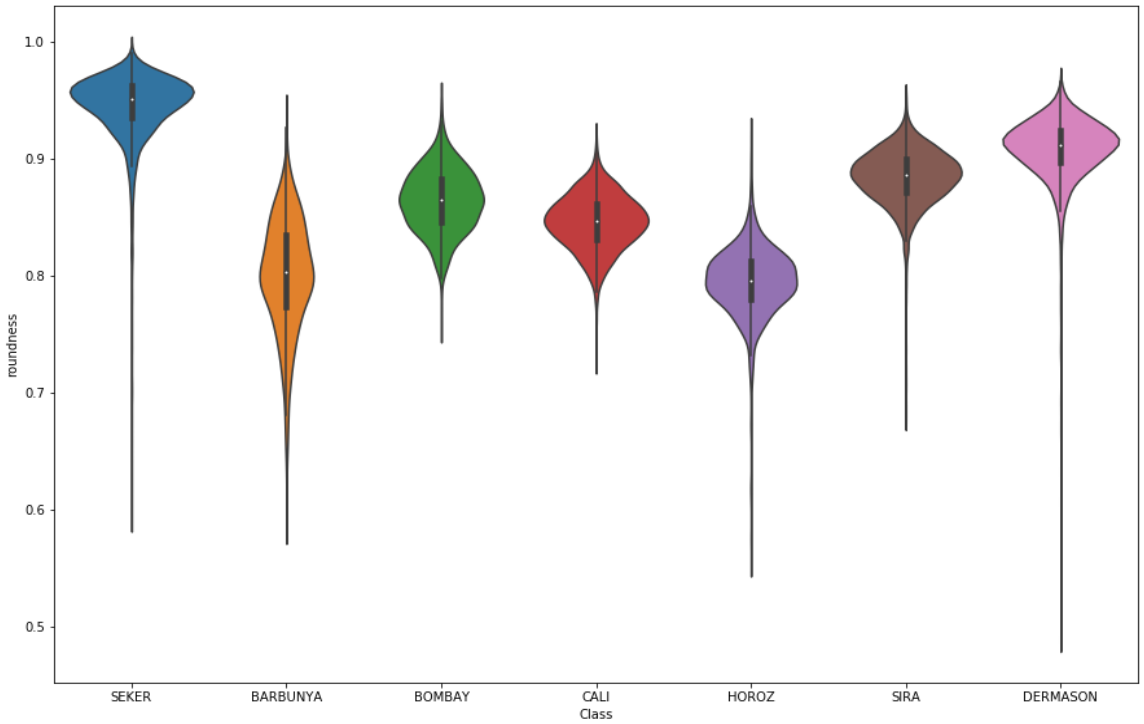
Code- plt.figure(figsize=(15,10))

sns.scatterplot(x='Perimeter', y='EquivDiameter', hue='Class', data=df2)

A scatter plot is a type of plot or mathematical diagram using Cartesian coordinates to display values for typically two variables for a set of data. It is normally **used** to observe and visually display the relationship between variables. Here the graph is plotted between EquivDiameter and Perimeter.

- The plot is observed to be a quite linear one with increasing factor. As the Perimeter increases EquivDiameter also increases.
- The graph initiates from DERMASON class which have the lowest perimeter and equivDiameter, goes on forming a linear increasing cluster of SEKER, SIRA, HOROZ, BARBUNYA and CALI.
- A break in continuity can be noticed while following to the BOMBAY bean class. Thus it can be concluded to have visibly distinct feature.

Plot 8:-



Code- plt.figure(figsize=(15,10))

sns.violinplot(x='Class', y='roundness', data=df2)

This violin plot is plotted against roundness character of all the classes of dry beans to inspect the intensity of roundness factor in all the bean class.

- SEKER beans have the highest roundness factor with most of the beans with factor approximate to around 0.95.
- BARBUNYA beans have a long range of roundness factor from 0.7 to 0.9. The roundness varies largely within its own class.
- BOMBAY and CALI have the roundness factor around 0.85 and HOROZ with a factor of 0.8(approx.).
- SIRA and DERMASON have roundness factor around 0.9.

Preparing the ML Model

Dataset named ‘Dry_Bean_Dataset – Altered’ is provided with 16 different attributes which defines the 7 classes of dry beans with various specifications. We have to analyze all the attributes to specify the class of the bean, which can be done using Classification.

Train-Test Split:

```
X = df2.drop('Class', axis=1).values.astype(np.float)

y = df['Class'].values

from sklearn import preprocessing

X = preprocessing.StandardScaler().fit(X).transform(X.astype(float))

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=100)
```

K-Neighbors-Classification

Training Phase

```
from sklearn.neighbors import KNeighborsClassifier

k = 7

neigh = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)
```

Testing Phase

```
pred = neigh.predict(X_test)

from sklearn.metrics import accuracy_score, classification_report

print(classification_report(y_test, pred))
```

```
              precision    recall  f1-score   support

    BARBUNYA       0.94      0.89      0.92       261
      BOMBAY       1.00      1.00      1.00       117
        CALI       0.91      0.95      0.93       317
    DERMASON       0.90      0.91      0.91       671
       HOROZ       0.98      0.95      0.96       408
       SEKER       0.98      0.95      0.96       413
        SIRA       0.85      0.88      0.86       536

    accuracy                           0.92      2723
   macro avg       0.94      0.93      0.93      2723
weighted avg       0.92      0.92      0.92      2723
```

```
print(np.round(accuracy_score(y_test,pred),decimals=4))
```

Accuracy Score of K-Neighbors Classification is 0.9221.

Support Vector Classification

Training Phase

```
from sklearn.svm import SVC

model = SVC()

model.fit(X_train,y_train)
```

Testing Phase

```
pred2 = model.predict(X_test)

print(classification_report(y_test, pred2))
```

```
              precision    recall  f1-score   support

    BARBUNYA       0.93      0.92      0.93       261
      BOMBAY       1.00      1.00      1.00       117
        CALI       0.94      0.95      0.95       317
    DERMASON       0.91      0.93      0.92       671
       HOROZ       0.98      0.96      0.97       408
       SEKER       0.97      0.95      0.96       413
        SIRA       0.88      0.90      0.89       536

    accuracy                           0.93      2723
   macro avg       0.95      0.94      0.94      2723
weighted avg       0.93      0.93      0.93      2723
```

```
print(np.round(accuracy_score(y_test,pred2),decimals=4))
```

Accuracy Score of Support Vector Classification is 0.9339.

Stochastic Gradient Descent Classification

Training Phase

```
from sklearn.linear_model import SGDClassifier

clf = SGDClassifier(loss="hinge", penalty="l2", max_iter=1000)

clf.fit(X_train, y_train)

SGDClassifier()
```

Testing Phase

```
pred3 = clf.predict(X_test)

print(classification_report(y_test, pred3))
```

```
              precision    recall  f1-score   support

    BARBUNYA       0.93      0.89      0.91       261
      BOMBAY       1.00      1.00      1.00       117
        CALI       0.93      0.95      0.94       317
    DERMASON       0.91      0.90      0.91       671
       HOROZ       0.96      0.97      0.96       408
       SEKER       0.96      0.95      0.96       413
        SIRA       0.86      0.88      0.87       536

    accuracy                           0.92      2723
   macro avg       0.94      0.93      0.93      2723
weighted avg       0.92      0.92      0.92      2723
```

```
print(np.round(accuracy_score(y_test,pred3),decimals=4))
```

Accuracy Score of Stochastic Gradient Descent Classification is 0.9232.

Decision Tree Classification

Training Phase

```
from sklearn import tree

dtree = tree.DecisionTreeClassifier()

dtree = dtree.fit(X_train, y_train)
```

Testing Phase

```
pred4 = dtree.predict(X_test)

print(classification_report(y_test, pred4))
```

	precision	recall	f1-score	support
BARBUNYA	0.85	0.87	0.86	261
BOMBAY	1.00	1.00	1.00	117
CALI	0.89	0.90	0.90	317
DERMASON	0.88	0.88	0.88	671
HOROZ	0.95	0.93	0.94	408
SEKER	0.93	0.91	0.92	413
SIRA	0.82	0.84	0.83	536
accuracy			0.89	2723
macro avg	0.90	0.90	0.90	2723
weighted avg	0.89	0.89	0.89	2723

```
print(np.round(accuracy_score(y_test,pred4),decimals=4))
```

Accuracy Score of Decision Tree Classification is 0.8902.

Nearest Centroid Classification

Training Phase

```
from sklearn.neighbors import NearestCentroid

ncd = NearestCentroid()

ncd.fit(X_train, y_train)
```

NearestCentroid()

Testing Phase

```
pred5 = ncd.predict(X_test)

print(classification_report(y_test, pred5))
```

```
              precision    recall  f1-score   support

    BARBUNYA       0.94      0.84      0.88       261
      BOMBAY       1.00      1.00      1.00       117
        CALI       0.89      0.93      0.91       317
    DERMASON       0.93      0.85      0.89       671
       HOROZ       0.95      0.92      0.94       408
       SEKER       0.94      0.94      0.94       413
        SIRA       0.78      0.90      0.84       536

    accuracy                           0.90      2723
   macro avg       0.92      0.91      0.91      2723
weighted avg       0.91      0.90      0.90      2723
```

```
print(np.round(accuracy_score(y_test,pred5),decimals=4))
```

Accuracy Score of Nearest Centroid Classification is 0.9008.

Machine Learning Model Chart

Serial Number	ML Algorithm Used	Accuracy_Score (approx. 4 decimal places)
01	Support Vector Classification	0.9339
02	K-Neighbors-Classification	0.9221
03	Stochastic Gradient Descent Classification	0.9232
04	Nearest Centroid Classification	0.9008
05	Decision Tree Classification	0.8902

Hurdles

The hurdle I faced was in cleaning the dataset in an efficient way to get more accurate values for the missing data. I used the *IterativeImputer* from python *sklearn.impute* library which is a multivariate imputer that estimates each feature from all the others. It is a strategy for imputing missing values by modelling each feature with missing values as a function of other features in a round-robin fashion.

Conclusion

Support Vector Classification algorithm has got the highest accuracy score of 0.9339, so it is the best algorithm to classify the data to their specific class.

Decision Tree Classification algorithm is having the lowest accuracy score of 0.8906, so it is the least fitted algorithm among all five algorithms we trained and tested for the dataset.

Using the exploratory data analysis, we can get to know the relationships between various parameters (example, plot between the class versus total no of data of that class tells us that which class of bean is highly available and which is least available).

Bibliography

- *https://seaborn.pydata.org/examples/index.html*
- *https://scikit-learn.org/stable/supervised_learning.html#supervised-learning*
- *https://www.geeksforgeeks.org/python-programming-language/*