In [78]:

```
#Importing PANDAS Library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings


#Reading Data Set from Excel File
data_set = pd.read_excel (r'C:\Users\MAHAM\Desktop\TTDS Project\junaid1.xlsx')
print ("Successfully Imported Data\n", data_set)
```

```
Successfully Imported Data
          Id  AREA  PERIMETER  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ROUNDNESS  \
0         1  7805    437.915    209.8215     48.0221        0.9735     0.5114
1         2  7503    340.757    138.3361     69.8417        0.8632     0.8120
2         3  5124    314.617    141.9803     46.5784        0.9447     0.6505
3         4  7990    437.085    201.4386     51.2245        0.9671     0.5256
4         5  7433    342.893    140.3350     68.3927        0.8732     0.7944
...     ...   ...        ...         ...         ...           ...        ...
9494   9996  6670    309.007    117.3843     73.6883        0.7784     0.8778
9495   9997  5951    292.260    118.2284     64.4155        0.8385     0.8755
9496   9998  6964    418.046    201.5034     44.2449        0.9756     0.5007
9497   9999  5479    331.863    153.6018     46.0075        0.9541     0.6252
9498  10000  5409    330.328    152.1949     46.3460        0.9525     0.6229

      SOLIDITY  CONVEX_AREA  EXTENT  ASPECT_RATIO  COMPACTNESS  SHAPEFACTOR_1  \
0       0.9775         7985  0.3547        4.3693       0.4751         0.0269
1       0.9660         7767  0.6637        1.9807       0.7065         0.0184
2       0.9721         5271  0.4760        3.0482       0.5689         0.0277
3       0.9659         8272  0.6274        3.9325       0.5007         0.0252
4       0.9831         7561  0.6006        2.0519       0.6932         0.0189
...        ...          ...     ...           ...          ...            ...
9494    0.9751         6840  0.7126        1.5930       0.7851         0.0176
9495    0.9844         6045  0.6975        1.8354       0.7363         0.0199
9496    0.9766         7131  0.7039        4.5543       0.4673         0.0289
9497    0.9761         5613  0.4427        3.3386       0.5438         0.0280
9498    0.9680         5588  0.4878        3.2839       0.5453         0.0281

      SHAPEFACTOR_2  SHAPEFACTOR_3  SHAPEFACTOR_4         CLASS
0            0.0062         0.2257         0.9863       Basmati
1            0.0093         0.4992         0.9888       Arborio
2            0.0091         0.3236         0.9865       Jasmine
3            0.0064         0.2507         0.9859       Basmati
4            0.0092         0.4806         0.9860       Arborio
...             ...            ...            ...           ...
9494         0.0110         0.6163         0.9818  Super Colonel
9495         0.0108         0.5421         0.9949  Super Colonel
9496         0.0064         0.2184         0.9945       Basmati
9497         0.0084         0.2957         0.9872       Jasmine
9498         0.0086         0.2973         0.9764       Jasmine

[9499 rows x 17 columns]
```

In [13]:

```
data_set.count()
```

Out[13]:

```
Id                 9499
AREA               9499
PERIMETER          9499
MAJOR_AXIS         9499
MINOR_AXIS         9499
```

```
ECCENTRICITY     9499
ROUNDNESS        9499
SOLIDITY         9499
CONVEX_AREA      9499
EXTENT           9499
ASPECT_RATIO     9499
COMPACTNESS      9499
SHAPEFACTOR_1    9499
SHAPEFACTOR_2    9499
SHAPEFACTOR_3    9499
SHAPEFACTOR_4    9499
CLASS            9499
dtype: int64
```

In [ ]:

In [16]:

```
data_set.head()
```

Out[16]:

	Id	AREA	PERIMETER	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	SOLIDITY	CON
0	1	7805	437.915	209.8215	48.0221	0.9735	0.5114	0.9775	
1	2	7503	340.757	138.3361	69.8417	0.8632	0.8120	0.9660	
2	3	5124	314.617	141.9803	46.5784	0.9447	0.6505	0.9721	
3	4	7990	437.085	201.4386	51.2245	0.9671	0.5256	0.9659	
4	5	7433	342.893	140.3350	68.3927	0.8732	0.7944	0.9831	

In [30]:

```
data_set.shape
```

Out[30]:

```
(9499, 17)
```

In [45]:

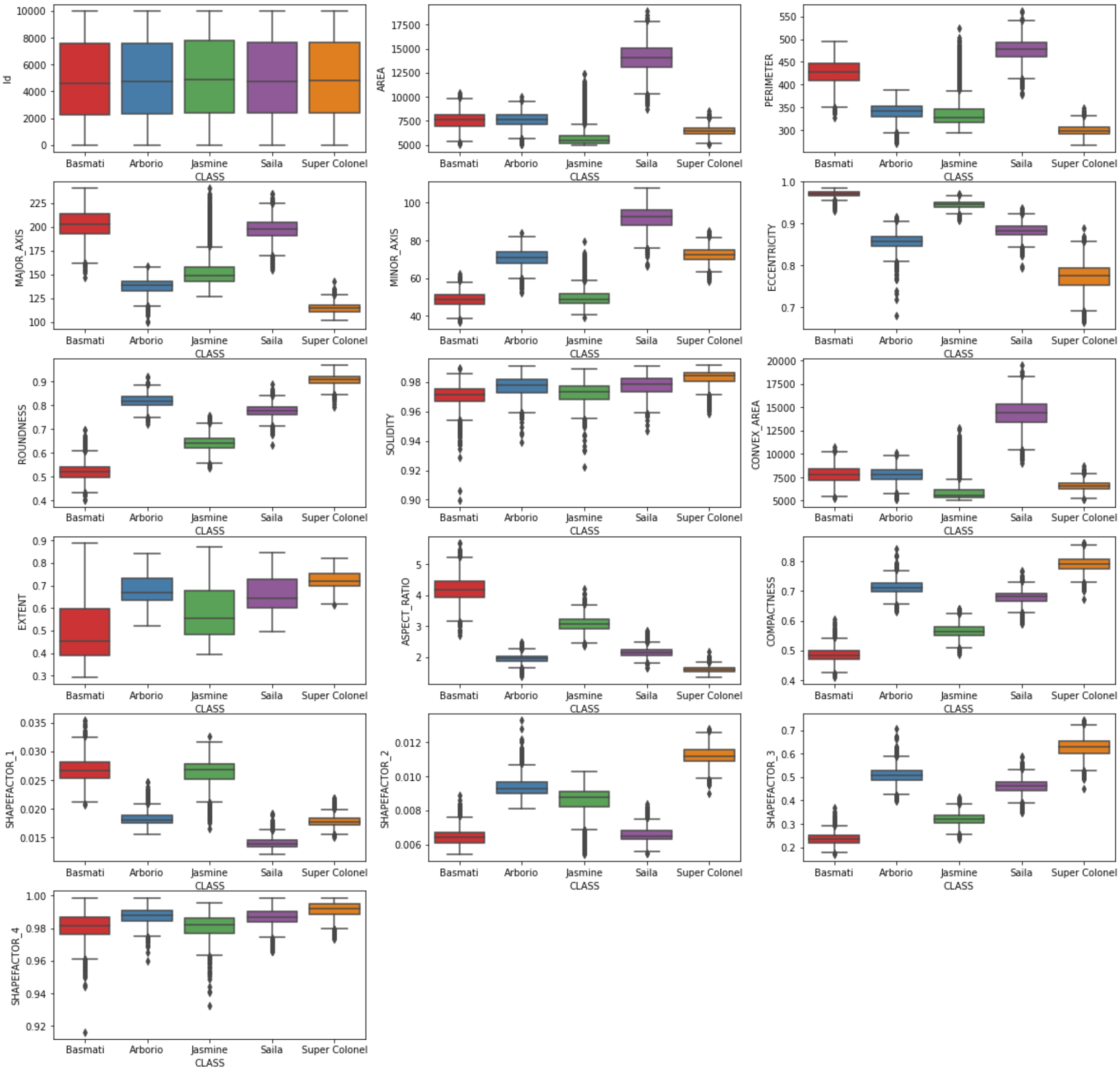
```
data_set['CLASS'].value_counts()
```

Out[45]:

```
Basmati          1936
Jasmine          1919
Saila            1898
Super Colonel    1885
Arborio          1861
Name: CLASS, dtype: int64
```

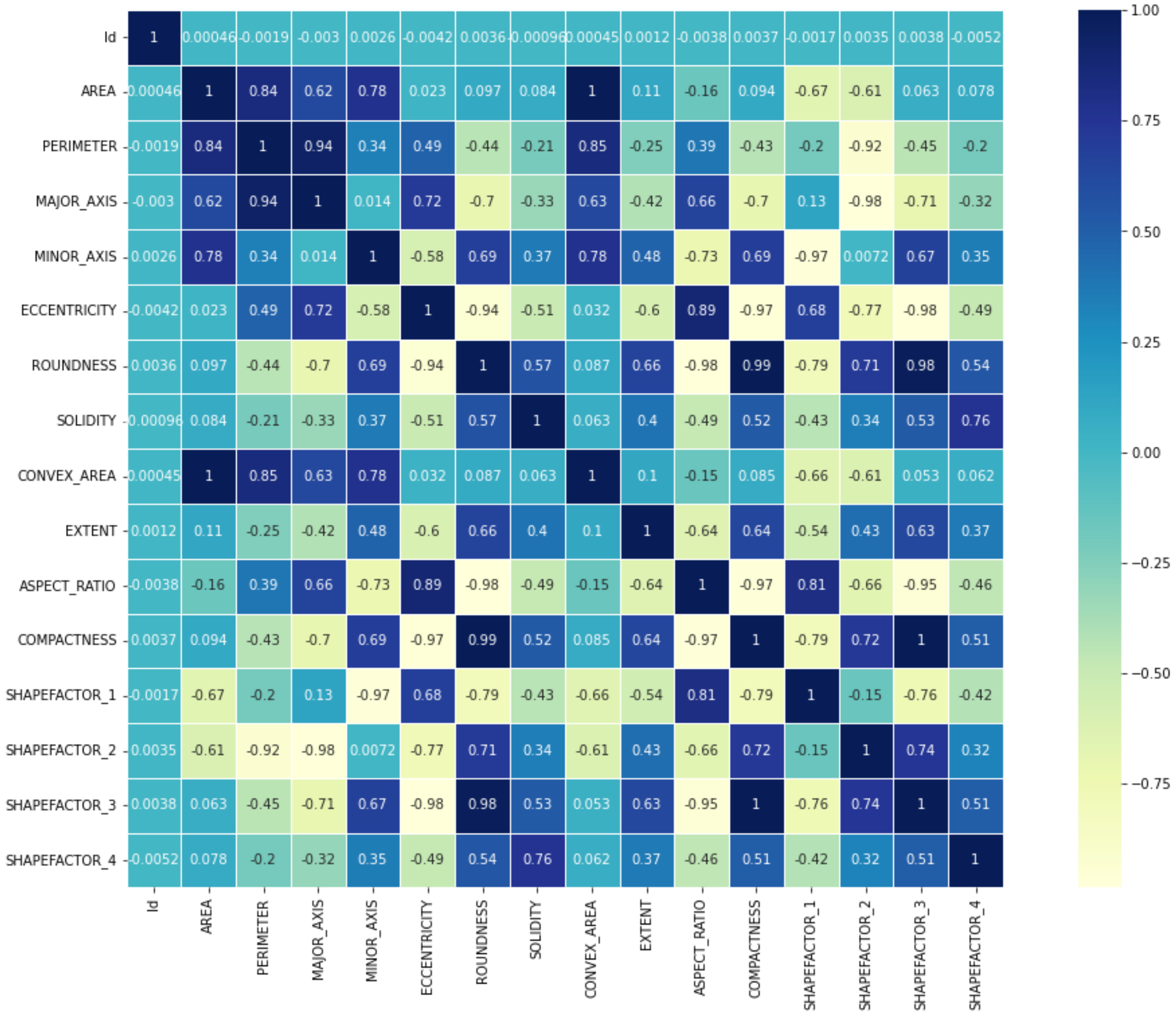
In [110…

```
# X = data_set.drop('CLASS', axis=1).values
#y = data_set['CLASS'].values
#plt.plot(X,Y)
#plt.show()


plt.figure(figsize=(20,20))
for i in range(16):
    plt.subplot(6, 3, i + 1)
    sns.boxplot(x="CLASS", y=data_set.columns[i], data=data_set, palette="Set1")
plt.show()
```



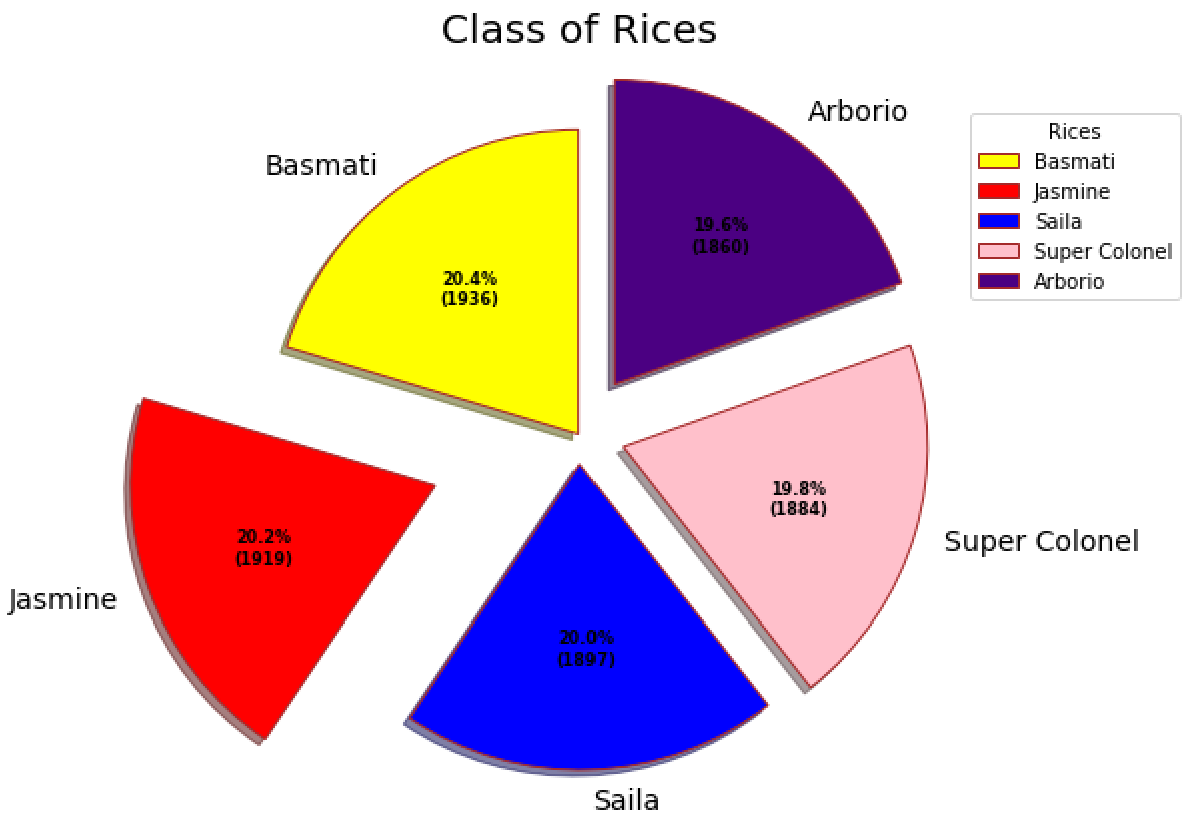
```
In [102…  plt.figure(figsize=(20,12))
          sns.heatmap(data_set.iloc[:,:16].corr(), cmap="YlGnBu",annot=True, fmt=".2g", linewi
```

```
Out[102…  <AxesSubplot:>
```



In [58]:

```
label = data_set["CLASS"].value_counts().index
value = data_set["CLASS"].value_counts().values
explode = (0.0, 0.5, 0.1, 0.15, 0.2)
colors = ( "yellow", "red", "blue","pink", "indigo")
wp = { 'linewidth' : 1, 'edgecolor' : "brown" }

def func(pct, allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return "{:.1f}%\n({:d})".format(pct, absolute)

fig, ax = plt.subplots(figsize =(10, 7))
wedges, texts, autotexts = ax.pie(value,
                                  autopct = lambda pct: func(pct, value),
                                  explode = explode,
                                  labels = label,
                                  shadow = True,
                                  colors = colors,
                                  startangle = 90,
                                  wedgeprops = wp,
                                  textprops = dict(color ="k",fontsize=14))


ax.legend(wedges, label,
          title ="Rices",
          loc ="center left",
          bbox_to_anchor =(1, 0, 0.8, 1.6))
plt.setp(autotexts, size = 8, weight ="bold")
ax.set_title("Class of Rices",fontsize=20)
plt.show()
```



In [60]:

```
#Finding Null Values in Data Set
print(data_set.isna().sum())
```

```
Id                0
AREA              0
PERIMETER         0
MAJOR_AXIS        0
MINOR_AXIS        0
ECCENTRICITY      0
ROUNDNESS         0
SOLIDITY          0
CONVEX_AREA       0
EXTENT            0
ASPECT_RATIO      0
COMPACTNESS       0
SHAPEFACTOR_1     0
SHAPEFACTOR_2     0
SHAPEFACTOR_3     0
SHAPEFACTOR_4     0
CLASS             0
dtype: int64
```

In [62]:

```
#Dropping the Null Values
clear_data_set = data_set.dropna()
print(clear_data_set.isna().sum())
```

```
Id                0
AREA              0
PERIMETER         0
MAJOR_AXIS        0
MINOR_AXIS        0
ECCENTRICITY      0
ROUNDNESS         0
SOLIDITY          0
CONVEX_AREA       0
EXTENT            0
```

```
ASPECT_RATIO     0
COMPACTNESS      0
SHAPEFACTOR_1    0
SHAPEFACTOR_2    0
SHAPEFACTOR_3    0
SHAPEFACTOR_4    0
CLASS            0
dtype: int64
```

In [82]:

```
#Removing Area Attribute
ds1 = data_set.drop("AREA", axis='columns')
ds1.head()
```

Out[82]:

	Id	PERIMETER	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	SOLIDITY	CONVEX_AR
0	1	437.915	209.8215	48.0221	0.9735	0.5114	0.9775	79
1	2	340.757	138.3361	69.8417	0.8632	0.8120	0.9660	77
2	3	314.617	141.9803	46.5784	0.9447	0.6505	0.9721	52
3	4	437.085	201.4386	51.2245	0.9671	0.5256	0.9659	82
4	5	342.893	140.3350	68.3927	0.8732	0.7944	0.9831	75

In [84]:

```
#Removing Perimeter Attribute
ds2 = ds1.drop("PERIMETER", axis='columns')
ds2.head()
```

Out[84]:

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	SOLIDITY	CONVEX_AREA	EXTENT
0	1	209.8215	48.0221	0.9735	0.5114	0.9775	7985	0.3547
1	2	138.3361	69.8417	0.8632	0.8120	0.9660	7767	0.6637
2	3	141.9803	46.5784	0.9447	0.6505	0.9721	5271	0.4760
3	4	201.4386	51.2245	0.9671	0.5256	0.9659	8272	0.6274
4	5	140.3350	68.3927	0.8732	0.7944	0.9831	7561	0.6006

In [85]:

```
#Removing Solidity Attribute
ds3 = ds2.drop("SOLIDITY", axis='columns')
ds3.head()
```

Out[85]:

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	CONVEX_AREA	EXTENT	ASPECT_R
0	1	209.8215	48.0221	0.9735	0.5114	7985	0.3547	4
1	2	138.3361	69.8417	0.8632	0.8120	7767	0.6637	1
2	3	141.9803	46.5784	0.9447	0.6505	5271	0.4760	3
3	4	201.4386	51.2245	0.9671	0.5256	8272	0.6274	3
4	5	140.3350	68.3927	0.8732	0.7944	7561	0.6006	2

In [86]:

```
#Removing Convex Area Attribute
```

```
ds4 = ds3.drop("CONVEX_AREA", axis='columns')
ds4.head()
```

Out[86]:

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	EXTENT	ASPECT_RATIO	COMPACT
0	1	209.8215	48.0221	0.9735	0.5114	0.3547	4.3693	0
1	2	138.3361	69.8417	0.8632	0.8120	0.6637	1.9807	0
2	3	141.9803	46.5784	0.9447	0.6505	0.4760	3.0482	0
3	4	201.4386	51.2245	0.9671	0.5256	0.6274	3.9325	0
4	5	140.3350	68.3927	0.8732	0.7944	0.6006	2.0519	0

In [87]:

```
#Removing Extent Attribute
ds5 = ds4.drop("EXTENT", axis='columns')
ds5.head()
```

Out[87]:

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO	COMPACTNESS	SH
0	1	209.8215	48.0221	0.9735	0.5114	4.3693	0.4751	
1	2	138.3361	69.8417	0.8632	0.8120	1.9807	0.7065	
2	3	141.9803	46.5784	0.9447	0.6505	3.0482	0.5689	
3	4	201.4386	51.2245	0.9671	0.5256	3.9325	0.5007	
4	5	140.3350	68.3927	0.8732	0.7944	2.0519	0.6932	

In [112…

```
#Removing Shape Factor 4 Attribute
ds6 = ds5.drop("SHAPEFACTOR_4", axis='columns')
ds6.head()
```

Out[112…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO	COMPACTNESS	SH
0	1	209.8215	48.0221	0.9735	0.5114	4.3693	0.4751	
1	2	138.3361	69.8417	0.8632	0.8120	1.9807	0.7065	
2	3	141.9803	46.5784	0.9447	0.6505	3.0482	0.5689	
3	4	201.4386	51.2245	0.9671	0.5256	3.9325	0.5007	
4	5	140.3350	68.3927	0.8732	0.7944	2.0519	0.6932	

In [113…

```
fds = ds6
fds.head()
```

Out[113…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO	COMPACTNESS	SH
0	1	209.8215	48.0221	0.9735	0.5114	4.3693	0.4751	
1	2	138.3361	69.8417	0.8632	0.8120	1.9807	0.7065	
2	3	141.9803	46.5784	0.9447	0.6505	3.0482	0.5689	
3	4	201.4386	51.2245	0.9671	0.5256	3.9325	0.5007	

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO	COMPACTNESS	SH
4	5	140.3350	68.3927	0.8732	0.7944	2.0519	0.6932	

In [114…

```
#Max Min Normalization
fds['MAJOR_AXIS'] = (fds['MAJOR_AXIS'] - fds['MAJOR_AXIS'].min()) / (fds['MAJOR_AXIS
fds['MMINOR_AXIS'] = (fds['MINOR_AXIS'] - fds['MINOR_AXIS'].min()) / (fds['MINOR_AXI
fds['ECCENTRICITY'] = (fds['ECCENTRICITY'] - fds['ECCENTRICITY'].min()) / (fds['ECCE
fds['ROUNDNESS'] = (fds['ROUNDNESS'] - fds['ROUNDNESS'].min()) / (fds['ROUNDNESS'].m
fds['ASPECT_RATIO'] = (fds['ASPECT_RATIO'] - fds['ASPECT_RATIO'].min()) / (fds['ASPE
fds['COMPACTNESS'] = (fds['COMPACTNESS'] - fds['COMPACTNESS'].min()) / (fds['COMPACT
fds['SHAPEFACTOR_1'] = (fds['SHAPEFACTOR_1'] - fds['SHAPEFACTOR_1'].min()) / (fds['S
fds['SHAPEFACTOR_2'] = (fds['SHAPEFACTOR_2'] - fds['SHAPEFACTOR_2'].min()) / (fds['S
fds['SHAPEFACTOR_3'] = (fds['SHAPEFACTOR_3'] - fds['SHAPEFACTOR_3'].min()) / (fds['S


fds.describe()
```

Out[114…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO	COM
count	9499.000000	9499.000000	9499.000000	9499.000000	9499.000000	9499.000000	9
mean	4987.340773	0.443569	66.714497	0.695052	0.582034	0.291321	
std	2961.286339	0.258804	16.706252	0.224498	0.246399	0.222934	
min	1.000000	0.000000	36.456900	0.000000	0.000000	0.000000	
25%	2375.500000	0.235299	49.642550	0.569068	0.385481	0.123882	
50%	4750.000000	0.356698	69.108300	0.694730	0.657330	0.189246	
75%	7625.500000	0.695569	75.781400	0.895853	0.764847	0.435725	
max	10000.000000	1.000000	107.731700	1.000000	1.000000	1.000000	

In [115…

```
fds.corr()
```

Out[115…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIC
Id	1.000000	-0.003001	0.002589	-0.004230	0.003638	-0.00380
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	-0.699880	0.66272
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.72775
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	-0.940931	0.88716
ROUNDNESS	0.003638	-0.699880	0.688965	-0.940931	1.000000	-0.98118
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	-0.981183	1.00000
COMPACTNESS	0.003700	-0.696504	0.692341	-0.966256	0.992809	-0.97355
SHAPEFACTOR_1	-0.001653	0.125046	-0.973588	0.683299	-0.785363	0.81458
SHAPEFACTOR_2	0.003497	-0.984638	0.007220	-0.772735	0.714834	-0.66112
SHAPEFACTOR_3	0.003845	-0.705294	0.666889	-0.982797	0.984708	-0.95379
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.72775

In [116…

```
#Correlation is maximum between ShapeFactor1, Deleting ShapeFactor 1
fds1 = fds.drop("SHAPEFACTOR_1", axis='columns')
print(fds1)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ROUNDNESS  ASPECT_RATIO  \
0         1    0.781659     48.0221      0.965700   0.192519      0.695577
1         2    0.275116     69.8417      0.621765   0.725403      0.147658
2         3    0.300939     46.5784      0.875896   0.439107      0.392531
3         4    0.722258     51.2245      0.945744   0.217692      0.595380
4         5    0.289280     68.3927      0.652947   0.694203      0.163990
...     ...         ...         ...           ...        ...           ...
9494   9996    0.126652     73.6883      0.357343   0.842049      0.058724
9495   9997    0.132634     64.4155      0.544746   0.837972      0.114328
9496   9998    0.722718     44.2449      0.972248   0.173551      0.738014
9497   9999    0.383288     46.0075      0.905207   0.394256      0.459146
9498  10000    0.373319     46.3460      0.900218   0.390179      0.446598

      COMPACTNESS  SHAPEFACTOR_2  SHAPEFACTOR_3         CLASS  MMINOR_AXIS
0        0.143839       0.101266       0.100035       Basmati     0.162262
1        0.656693       0.493671       0.576682       Arborio     0.468396
2        0.351729       0.468354       0.270652       Jasmine     0.142007
3        0.200576       0.126582       0.143604       Basmati     0.207192
4        0.627216       0.481013       0.544266       Arborio     0.448066
...           ...            ...            ...           ...          ...
9494     0.830895       0.708861       0.780760  Super Colonel     0.522364
9495     0.722739       0.683544       0.651446  Super Colonel     0.392265
9496     0.126551       0.126582       0.087313       Basmati     0.109267
9497     0.296099       0.379747       0.222029       Jasmine     0.133997
9498     0.299424       0.405063       0.224817       Jasmine     0.138746

[9499 rows x 11 columns]
```

In [117…

```
#Finding Correlation of new Data Set
fds1.corr()
```

Out[117…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATIO
Id	1.000000	-0.003001	0.002589	-0.004230	0.003638	-0.003808
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	-0.699880	0.662720
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.727758
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	-0.940931	0.887169
ROUNDNESS	0.003638	-0.699880	0.688965	-0.940931	1.000000	-0.981183
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	-0.981183	1.000000
COMPACTNESS	0.003700	-0.696504	0.692341	-0.966256	0.992809	-0.973556
SHAPEFACTOR_2	0.003497	-0.984638	0.007220	-0.772735	0.714834	-0.661127
SHAPEFACTOR_3	0.003845	-0.705294	0.666889	-0.982797	0.984708	-0.953792
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.727758

In [118…

```
#Correlation is maximum between ShapeFactor2 and Major  Axis Length, Deleting ShapeF
fds2 = fds1.drop("SHAPEFACTOR_2", axis='columns')
print(fds2)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ROUNDNESS  ASPECT_RATIO  \
0         1    0.781659     48.0221      0.965700   0.192519      0.695577   
1         2    0.275116     69.8417      0.621765   0.725403      0.147658   
2         3    0.300939     46.5784      0.875896   0.439107      0.392531   
3         4    0.722258     51.2245      0.945744   0.217692      0.595380   
4         5    0.289280     68.3927      0.652947   0.694203      0.163990   
...     ...         ...         ...           ...        ...           ...   
9494   9996    0.126652     73.6883      0.357343   0.842049      0.058724   
9495   9997    0.132634     64.4155      0.544746   0.837972      0.114328   
9496   9998    0.722718     44.2449      0.972248   0.173551      0.738014   
9497   9999    0.383288     46.0075      0.905207   0.394256      0.459146   
9498  10000    0.373319     46.3460      0.900218   0.390179      0.446598   

      COMPACTNESS  SHAPEFACTOR_3          CLASS  MMINOR_AXIS  
0        0.143839       0.100035        Basmati     0.162262  
1        0.656693       0.576682        Arborio     0.468396  
2        0.351729       0.270652        Jasmine     0.142007  
3        0.200576       0.143604        Basmati     0.207192  
4        0.627216       0.544266        Arborio     0.448066  
...           ...            ...            ...          ...  
9494     0.830895       0.780760  Super Colonel     0.522364  
9495     0.722739       0.651446  Super Colonel     0.392265  
9496     0.126551       0.087313        Basmati     0.109267  
9497     0.296099       0.222029        Jasmine     0.133997  
9498     0.299424       0.224817        Jasmine     0.138746  

[9499 rows x 10 columns]
```

In [119…

```
fds2.corr()
```

Out[119…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ROUNDNESS	ASPECT_RATI
Id	1.000000	-0.003001	0.002589	-0.004230	0.003638	-0.00380
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	-0.699880	0.66272
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.72775
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	-0.940931	0.88716
ROUNDNESS	0.003638	-0.699880	0.688965	-0.940931	1.000000	-0.98118
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	-0.981183	1.00000
COMPACTNESS	0.003700	-0.696504	0.692341	-0.966256	0.992809	-0.97355
SHAPEFACTOR_3	0.003845	-0.705294	0.666889	-0.982797	0.984708	-0.95379
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	0.688965	-0.72775

In [120…

```
#Correlation is maximum between Roundness with Eccentricity, Aspect_ratio, Compactne
fds3 = fds2.drop("ROUNDNESS", axis='columns')
print(fds3)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  COMPACTNESS  \
0         1    0.781659     48.0221      0.965700      0.695577     0.143839   
1         2    0.275116     69.8417      0.621765      0.147658     0.656693   
2         3    0.300939     46.5784      0.875896      0.392531     0.351729   
3         4    0.722258     51.2245      0.945744      0.595380     0.200576   
4         5    0.289280     68.3927      0.652947      0.163990     0.627216   
...     ...         ...         ...           ...           ...          ...   
9494   9996    0.126652     73.6883      0.357343      0.058724     0.830895   
9495   9997    0.132634     64.4155      0.544746      0.114328     0.722739   
```

```
9496   9998    0.722718    44.2449     0.972248     0.738014     0.126551
9497   9999    0.383288    46.0075     0.905207     0.459146     0.296099
9498  10000    0.373319    46.3460     0.900218     0.446598     0.299424

      SHAPEFACTOR_3          CLASS  MMINOR_AXIS
0          0.100035        Basmati     0.162262
1          0.576682        Arborio     0.468396
2          0.270652        Jasmine     0.142007
3          0.143604        Basmati     0.207192
4          0.544266        Arborio     0.448066
...             ...            ...          ...
9494       0.780760  Super Colonel     0.522364
9495       0.651446  Super Colonel     0.392265
9496       0.087313        Basmati     0.109267
9497       0.222029        Jasmine     0.133997
9498       0.224817        Jasmine     0.138746

[9499 rows x 9 columns]
```

In [121…

```
fds3.corr()
```

Out[121…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ASPECT_RATIO	COMPACTNI
Id	1.000000	-0.003001	0.002589	-0.004230	-0.003808	0.0037
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	0.662720	-0.6965
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	0.6923
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	0.887169	-0.9662
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	1.000000	-0.9735
COMPACTNESS	0.003700	-0.696504	0.692341	-0.966256	-0.973556	1.000(
SHAPEFACTOR_3	0.003845	-0.705294	0.666889	-0.982797	-0.953792	0.997(
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	0.6923

In [122…

```
#Correlation is maximum between Roundness with Eccentricity, Aspect_ratio, Compactne
fds4 = fds3.drop("COMPACTNESS", axis='columns')

print(fds4)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  \
0         1    0.781659     48.0221      0.965700      0.695577   
1         2    0.275116     69.8417      0.621765      0.147658   
2         3    0.300939     46.5784      0.875896      0.392531   
3         4    0.722258     51.2245      0.945744      0.595380   
4         5    0.289280     68.3927      0.652947      0.163990   
...     ...         ...         ...           ...           ...   
9494   9996    0.126652     73.6883      0.357343      0.058724   
9495   9997    0.132634     64.4155      0.544746      0.114328   
9496   9998    0.722718     44.2449      0.972248      0.738014   
9497   9999    0.383288     46.0075      0.905207      0.459146   
9498  10000    0.373319     46.3460      0.900218      0.446598   

      SHAPEFACTOR_3          CLASS  MMINOR_AXIS
0          0.100035        Basmati     0.162262
1          0.576682        Arborio     0.468396
2          0.270652        Jasmine     0.142007
3          0.143604        Basmati     0.207192
```

```
4         0.544266        Arborio     0.448066
...            ...            ...          ...
9494      0.780760  Super Colonel     0.522364
9495      0.651446  Super Colonel     0.392265
9496      0.087313        Basmati     0.109267
9497      0.222029        Jasmine     0.133997
9498      0.224817        Jasmine     0.138746

[9499 rows x 8 columns]
```

In [123…

```
fds4.corr()
```

Out[123…

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ASPECT_RATIO	SHAPEFACT
Id	1.000000	-0.003001	0.002589	-0.004230	-0.003808	0.00
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	0.662720	-0.70
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	0.66
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	0.887169	-0.98
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	1.000000	-0.95
SHAPEFACTOR_3	0.003845	-0.705294	0.666889	-0.982797	-0.953792	1.00
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	0.66

In [142…

```
#Correlation is maximum between Roundness with Eccentricity, Aspect_ratio, Compactne
fds5 = fds4.drop("SHAPEFACTOR_3", axis='columns')

print(fds5)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  \
0         1    0.781659     48.0221      0.965700      0.695577   
1         2    0.275116     69.8417      0.621765      0.147658   
2         3    0.300939     46.5784      0.875896      0.392531   
3         4    0.722258     51.2245      0.945744      0.595380   
4         5    0.289280     68.3927      0.652947      0.163990   
...     ...         ...         ...           ...           ...   
9494   9996    0.126652     73.6883      0.357343      0.058724   
9495   9997    0.132634     64.4155      0.544746      0.114328   
9496   9998    0.722718     44.2449      0.972248      0.738014   
9497   9999    0.383288     46.0075      0.905207      0.459146   
9498  10000    0.373319     46.3460      0.900218      0.446598   

              CLASS  MMINOR_AXIS  
0           Basmati     0.162262  
1           Arborio     0.468396  
2           Jasmine     0.142007  
3           Basmati     0.207192  
4           Arborio     0.448066  
...             ...          ...  
9494  Super Colonel     0.522364  
9495  Super Colonel     0.392265  
9496        Basmati     0.109267  
9497        Jasmine     0.133997  
9498        Jasmine     0.138746  

[9499 rows x 7 columns]
```

In [143…

```
fds5.corr()
```

Out[143...

	Id	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	ASPECT_RATIO	MMINOR_AXI
Id	1.000000	-0.003001	0.002589	-0.004230	-0.003808	0.00258
MAJOR_AXIS	-0.003001	1.000000	0.013642	0.718649	0.662720	0.01364
MINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	1.00000
ECCENTRICITY	-0.004230	0.718649	-0.583472	1.000000	0.887169	-0.58347
ASPECT_RATIO	-0.003808	0.662720	-0.727758	0.887169	1.000000	-0.72775
MMINOR_AXIS	0.002589	0.013642	1.000000	-0.583472	-0.727758	1.00000

In [144...

```
print(fds5)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  \
0         1    0.781659     48.0221      0.965700      0.695577   
1         2    0.275116     69.8417      0.621765      0.147658   
2         3    0.300939     46.5784      0.875896      0.392531   
3         4    0.722258     51.2245      0.945744      0.595380   
4         5    0.289280     68.3927      0.652947      0.163990   
...     ...         ...         ...           ...           ...   
9494   9996    0.126652     73.6883      0.357343      0.058724   
9495   9997    0.132634     64.4155      0.544746      0.114328   
9496   9998    0.722718     44.2449      0.972248      0.738014   
9497   9999    0.383288     46.0075      0.905207      0.459146   
9498  10000    0.373319     46.3460      0.900218      0.446598   

              CLASS  MMINOR_AXIS  
0           Basmati     0.162262  
1           Arborio     0.468396  
2           Jasmine     0.142007  
3           Basmati     0.207192  
4           Arborio     0.448066  
...             ...          ...  
9494  Super Colonel     0.522364  
9495  Super Colonel     0.392265  
9496        Basmati     0.109267  
9497        Jasmine     0.133997  
9498        Jasmine     0.138746  

[9499 rows x 7 columns]
```

In [146...

```
d = {'Basmati':1,'Arborio':2, 'Jasmine':3, 'Super Colonel':4, 'Saila':5}
fds5['CLASS'] = fds5['CLASS'].map(d)

print(fds5)
```

```
         Id  MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  CLASS  \
0         1    0.781659     48.0221      0.965700      0.695577      1   
1         2    0.275116     69.8417      0.621765      0.147658      2   
2         3    0.300939     46.5784      0.875896      0.392531      3   
3         4    0.722258     51.2245      0.945744      0.595380      1   
4         5    0.289280     68.3927      0.652947      0.163990      2   
...     ...         ...         ...           ...           ...    ...   
9494   9996    0.126652     73.6883      0.357343      0.058724      4   
9495   9997    0.132634     64.4155      0.544746      0.114328      4   
9496   9998    0.722718     44.2449      0.972248      0.738014      1   
9497   9999    0.383288     46.0075      0.905207      0.459146      3   
```

```
9498  10000    0.373319     46.3460      0.900218      0.446598      3

      MMINOR_AXIS
0         0.162262
1         0.468396
2         0.142007
3         0.207192
4         0.448066
...            ...
9494      0.522364
9495      0.392265
9496      0.109267
9497      0.133997
9498      0.138746

[9499 rows x 7 columns]
```

In [149…

```python
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split


features = ['MAJOR_AXIS','MINOR_AXIS','ECCENTRICITY', 'ASPECT_RATIO', 'CLASS']
x = fds5[features]
y = fds5['CLASS']

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

In [150…

```python
print(x_train)
```

```
      MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  CLASS
2413    0.692447     81.9237      0.766760      0.245561      5
5680    0.686962     92.5927      0.680387      0.180002      5
1139    0.745234     97.3841      0.672903      0.175437      5
1897    0.300872     47.2501      0.870596      0.382553      3
5715    0.331698     80.6921      0.531338      0.109258      2
...          ...         ...           ...           ...    ...
5734    0.659635     51.7330      0.933895      0.547323      1
5191    0.765768     92.0375      0.724977      0.210671      5
5390    0.353296     43.3480      0.914250      0.483736      3
860     0.754756     64.8661      0.889616      0.421893      3
7270    0.093737     73.0019      0.306205      0.047552      4

[7599 rows x 5 columns]
```

In [151…

```python
print(x_test)
```

```
      MAJOR_AXIS  MINOR_AXIS  ECCENTRICITY  ASPECT_RATIO  CLASS
7788    0.520445     48.6825      0.922357      0.508281      1
6856    0.731536     47.9805      0.959775      0.662614      1
5112    0.217617     73.5489      0.503274      0.099440      2
2944    0.074818     68.0050      0.381977      0.064573      4
6836    0.078962     67.5611      0.399751      0.069000      4
...          ...         ...           ...           ...    ...
8747    0.689657     99.2686      0.622700      0.148163      5
6291    0.729799     75.0289      0.826317      0.312428      5
4617    0.334657     73.6567      0.627066      0.150296      2
7195    0.326089     72.3489      0.635797      0.154723      2
4336    0.864467     65.0194      0.910820      0.474790      3

[1900 rows x 5 columns]
```

In [152…

```
print(y_train)
```

```
2413    5
5680    5
1139    5
1897    3
5715    2
       ..
5734    1
5191    5
5390    3
860     3
7270    4
Name: CLASS, Length: 7599, dtype: int64
```

In [153…

```
print(y_test)
```

```
7788    1
6856    1
5112    2
2944    4
6836    4
       ..
8747    5
6291    5
4617    2
7195    2
4336    3
Name: CLASS, Length: 1900, dtype: int64
```

In [154…

```
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
import sklearn.metrics as metrics

dtree = DecisionTreeClassifier()
dtree = dtree.fit(x_train,y_train)
y_pred = dtree.predict(x_test)
accuracy = dtree.score(x_test,y_test)
print(accuracy)
```

```
0.9994736842105263
```

In [155…

```
matrix = confusion_matrix(y_test,y_pred)
print("Confussion matrix - ",matrix)
accuracy = (matrix[0][0] + matrix[1][1])/(matrix[0][0]+matrix[0][1]+matrix[1][0]+mat
print("Accuracy - ",accuracy)
print("f1 score - ",metrics.f1_score(y_test, y_pred))
```

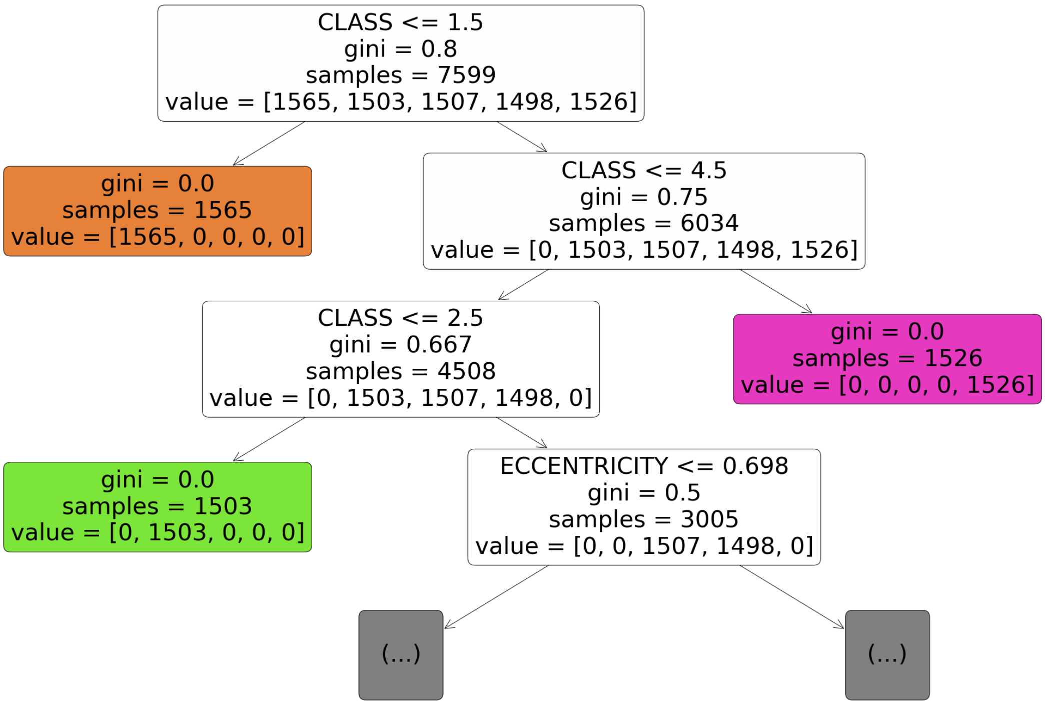
```
Confussion matrix -  [[371   0   0   0   0]
 [  0 358   0   0   0]
 [  0   0 412   0   0]
 [  0   0   1 386   0]
 [  0   0   0   0 372]]
Accuracy -  1.0
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_14380/983472536.py in <module>
      3 accuracy = (matrix[0][0] + matrix[1][1])/(matrix[0][0]+matrix[0][1]+matrix[1
][0]+matrix[1][1])
      4 print("Accuracy - ",accuracy)
----> 5 print("f1 score - ",metrics.f1_score(y_test, y_pred))
```

```
~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **kwarg
s)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in f1_score(y_true,
y_pred, labels, pos_label, average, sample_weight, zero_division)
   1069     modified with ``zero_division``.
   1070     """
-> 1071     return fbeta_score(y_true, y_pred, beta=1, labels=labels,
   1072                        pos_label=pos_label, average=average,
   1073                        sample_weight=sample_weight,

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **kwarg
s)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in fbeta_score(y_tr
ue, y_pred, beta, labels, pos_label, average, sample_weight, zero_division)
   1193     """
   1194 
-> 1195     _, _, f, _ = precision_recall_fscore_support(y_true, y_pred,
   1196                                                  beta=beta,
   1197                                                  labels=labels,

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **kwarg
s)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in precision_recall
_fscore_support(y_true, y_pred, beta, labels, pos_label, average, warn_for, sample_w
eight, zero_division)
   1462     if beta < 0:
   1463         raise ValueError("beta should be >=0 in the F-beta score")
-> 1464     labels = _check_set_wise_labels(y_true, y_pred, average, labels,
   1465                                     pos_label)
   1466 

~\anaconda3\lib\site-packages\sklearn\metrics\_classification.py in _check_set_wise_
labels(y_true, y_pred, average, labels, pos_label)
   1292             if y_type == 'multiclass':
   1293                 average_options.remove('samples')
-> 1294             raise ValueError("Target is %s but average='binary'. Please "
   1295                              "choose another average setting, one of %r."
   1296                              % (y_type, average_options))

ValueError: Target is multiclass but average='binary'. Please choose another average
setting, one of [None, 'micro', 'macro', 'weighted'].
```

In [156…

```
from sklearn.tree import plot_tree,export_text
import matplotlib.pyplot as plt
```

```
plt.figure(figsize=(40,25))
plot_tree(dtree,feature_names=features,max_depth=3,filled=True,rounded=True);
```



In [158…

```
from sklearn.linear_model import LogisticRegression
import sklearn.metrics as metrics
reg = LogisticRegression(random_state = 10)
reg.fit(x_train,y_train)
```

```
C:\Users\MAHAM\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[158…

```
LogisticRegression(random_state=10)
```

In [159…

```
import math
from sklearn.metrics import accuracy_score
y_pred = reg.predict(x_test)
accuracy_score(y_test,y_pred)
```

Out[159…

```
0.9989473684210526
```

In [160…

```
from sklearn.neighbors import KNeighborsClassifier
import sklearn.metrics as metrics
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train,y_train)
```

Out[160…

```
KNeighborsClassifier(n_neighbors=7)
```

In [161…

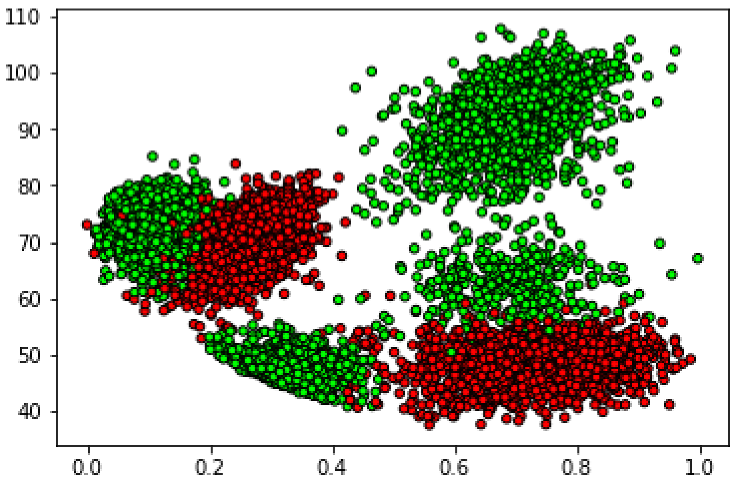
```
print(knn.score(x_test,y_test))
```

```
y_pred = knn.predict(x_test)
```

```
0.9957894736842106
```

In [164…

```
from matplotlib.colors import ListedColormap
cmap = ListedColormap(['#FF0000','#00FF00'])
X = np.array(x_train.values.tolist())
y = np.array(y_train)
plt.figure()
plt.scatter(X[:,0],X[:,1],c=y,cmap=cmap,edgecolors='k',s=20)
plt.show()
```



In [ ]: