Mini project 2

Data Set Information: Seven different types of dry beans were used in this project, taking into account the features such as form, shape, type, and structure by the market situation. Use best machine learning algorithm to classify the most well-known 7 types of beans in Turkey; Barbunya, Bombay, Cali, Dermason, Horoz, Seker and Sira, depending only on dimension and shape features of bean varieties with no external discriminatory features

Features Information

1. Area (A): The area of a bean zone and the number of pixels within its boundaries.
2. Perimeter (P): Bean circumference is defined as the length of its border.
3. Major axis length (L): The distance between the ends of the longest line that can be drawn from a bean.
4. Minor axis length (l): The longest line that can be drawn from the bean while standing perpendicular to the main axis.
5. Aspect ratio (K): Defines the relationship between L and l.
6. Eccentricity (Ec): Eccentricity of the ellipse having the same moments as the region.
7. Convex area (C): Number of pixels in the smallest convex polygon that can contain the area of a bean seed.
8. Equivalent diameter (Ed): The diameter of a circle having the same area as a bean seed area.
9. Extent (Ex): The ratio of the pixels in the bounding box to the bean area.
10. Solidity (S): Also known as convexity. The ratio of the pixels in the convex shell to those found in beans.
11. Roundness (R): Calculated with the following formula: (4piA)/(P^2)
12. Compactness (CO): Measures the roundness of an object: Ed/L
13. ShapeFactor1 (SF1)
14. ShapeFactor2 (SF2)
15. ShapeFactor3 (SF3)
16. ShapeFactor4 (SF4)
17. Class (Seker, Barbunya, Bombay, Cali, Dermosan, Horoz and Sira)

import libraries

In [ ]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

Data Loading

In [5]:
```
# read the data

df = pd.read_csv('data.csv')
```

In [6]:
```
# checking whether data is loaded or not (it shows overview of the data)
df.head()
```

Out[6]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFactor
0	28395	610.291	208.178117	173.888747	1.197191	0.549812	28715	190.141097	0.763923	0.988856	0.958027	0.913358	0.007332	0.00314
1	28734	638.018	200.524796	182.734419	1.097356	0.411785	29172	191.272751	0.783968	0.984986	0.887034	0.953861	0.006979	0.00356
2	29380	624.110	212.826130	175.931143	1.209713	0.562727	29690	193.410904	0.778113	0.989559	0.947849	0.908774	0.007244	0.00304
3	30008	645.884	210.557999	182.516516	1.153638	0.498616	30724	195.467062	0.782681	0.976696	0.903936	0.928329	0.007017	0.00321
4	30140	620.134	201.847882	190.279279	1.060798	0.333680	30417	195.896503	0.773098	0.990893	0.984877	0.970516	0.006697	0.00366

In [15]:
```
# checking shape
print(df.shape)
```

```
(13611, 17)
```

In [16]:
```
df.columns
```

Out[16]:
```
Index(['Area', 'Perimeter', 'MajorAxisLength', 'MinorAxisLength',
       'AspectRation', 'Eccentricity', 'ConvexArea', 'EquivDiameter', 'Extent',
       'Solidity', 'roundness', 'Compactness', 'ShapeFactor1', 'ShapeFactor2',
       'ShapeFactor3', 'ShapeFactor4', 'Class'],
      dtype='object')
```

In [17]:
```
# checking size
print(df.size)
```

```
231387
```

In [18]:
```
# describe the data
df.describe()
```

Out[18]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactnes
count	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.0000
mean	53048.284549	855.283459	320.141867	202.270714	1.583242	0.750895	53768.200206	253.064220	0.749733	0.987143	0.873282	0.7998
std	29324.095717	214.289696	85.694186	44.970091	0.246678	0.092002	29774.915817	59.177120	0.049086	0.004660	0.059520	0.0617
min	20420.000000	524.736000	183.601165	122.512653	1.024868	0.218951	20684.000000	161.243764	0.555315	0.919246	0.489618	0.6405
25%	36328.000000	703.523500	253.303633	175.848170	1.432307	0.715928	36714.500000	215.068003	0.718634	0.985670	0.832096	0.7624
50%	44652.000000	794.941000	296.883367	192.431733	1.551124	0.764441	45178.000000	238.438026	0.759859	0.988283	0.883157	0.8012
75%	61332.000000	977.213000	376.495012	217.031741	1.707109	0.810466	62294.000000	279.446467	0.786851	0.990013	0.916869	0.8342
max	254616.000000	1985.370000	738.860154	460.198497	2.430306	0.911423	263261.000000	569.374358	0.866195	0.994677	0.990685	0.9873

In [19]:
```
# cheking information about data (gives count of data types)
df.info()
```

```
<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  13611 non-null  float64
 4   AspectRation     13611 non-null  float64
 5   Eccentricity     13611 non-null  float64
 6   ConvexArea       13611 non-null  int64
 7   EquivDiameter    13611 non-null  float64
 8   Extent           13611 non-null  float64
 9   Solidity         13611 non-null  float64
 10  roundness        13611 non-null  float64
 11  Compactness      13611 non-null  float64
 12  ShapeFactor1     13611 non-null  float64
 13  ShapeFactor2     13611 non-null  float64
 14  ShapeFactor3     13611 non-null  float64
 15  ShapeFactor4     13611 non-null  float64
 16  Class            13611 non-null  object
dtypes: float64(14), int64(2), object(1)
memory usage: 1.8+ MB
```

In [20]:
```
# unique = it shows different data types present in the data
df.dtypes.unique()
```

Out[20]:
```
array([dtype('int64'), dtype('float64'), dtype('O')], dtype=object)
```

In [21]:
```
# data cleaning step
# drop missing values
print(df.isnull().sum())
df = df.dropna(axis=0)
```

```
Area               0
Perimeter          0
MajorAxisLength    0
MinorAxisLength    0
AspectRation       0
Eccentricity       0
ConvexArea         0
EquivDiameter      0
Extent             0
Solidity           0
roundness          0
Compactness        0
ShapeFactor1       0
ShapeFactor2       0
ShapeFactor3       0
ShapeFactor4       0
Class              0
dtype: int64
```

In [23]:
```
df.duplicated().sum()
```

Out[23]:
```
68
```

In [24]:
```
# check for unique values in the target variable
# target varible is class and others are features
print(df['Class'].unique())
```

```
['SEKER' 'BARBUNYA' 'BOMBAY' 'CALI' 'HOROZ' 'SIRA' 'DERMASON']
```

In [9]:
```
print(df['Class'].value_counts())
```

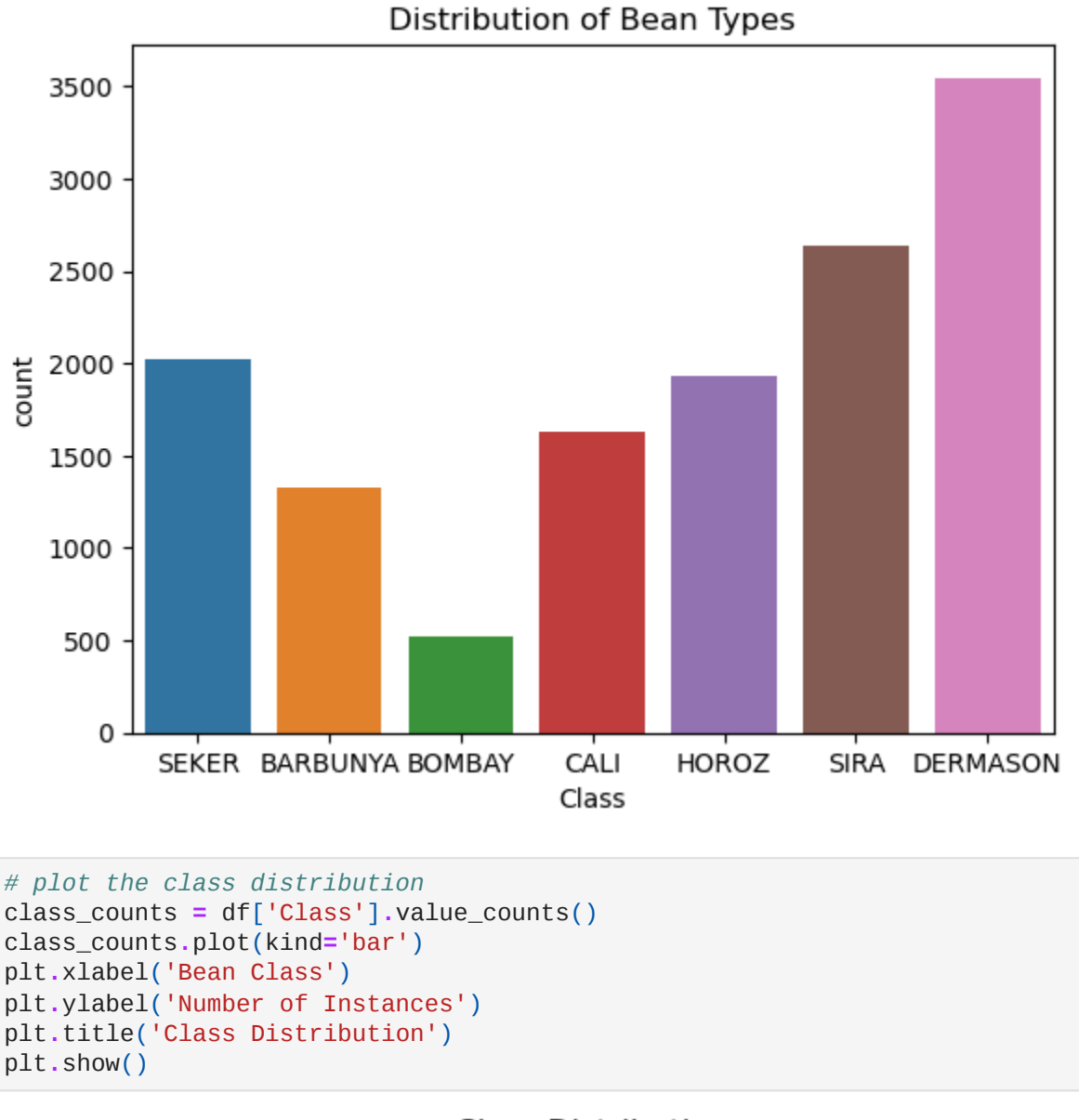
```
DERMASON    3546
SIRA        2636
SEKER       2027
HOROZ       1928
CALI        1630
BARBUNYA    1322
BOMBAY       522
Name: Class, dtype: int64
```

In [6]:
```
df.head()
```

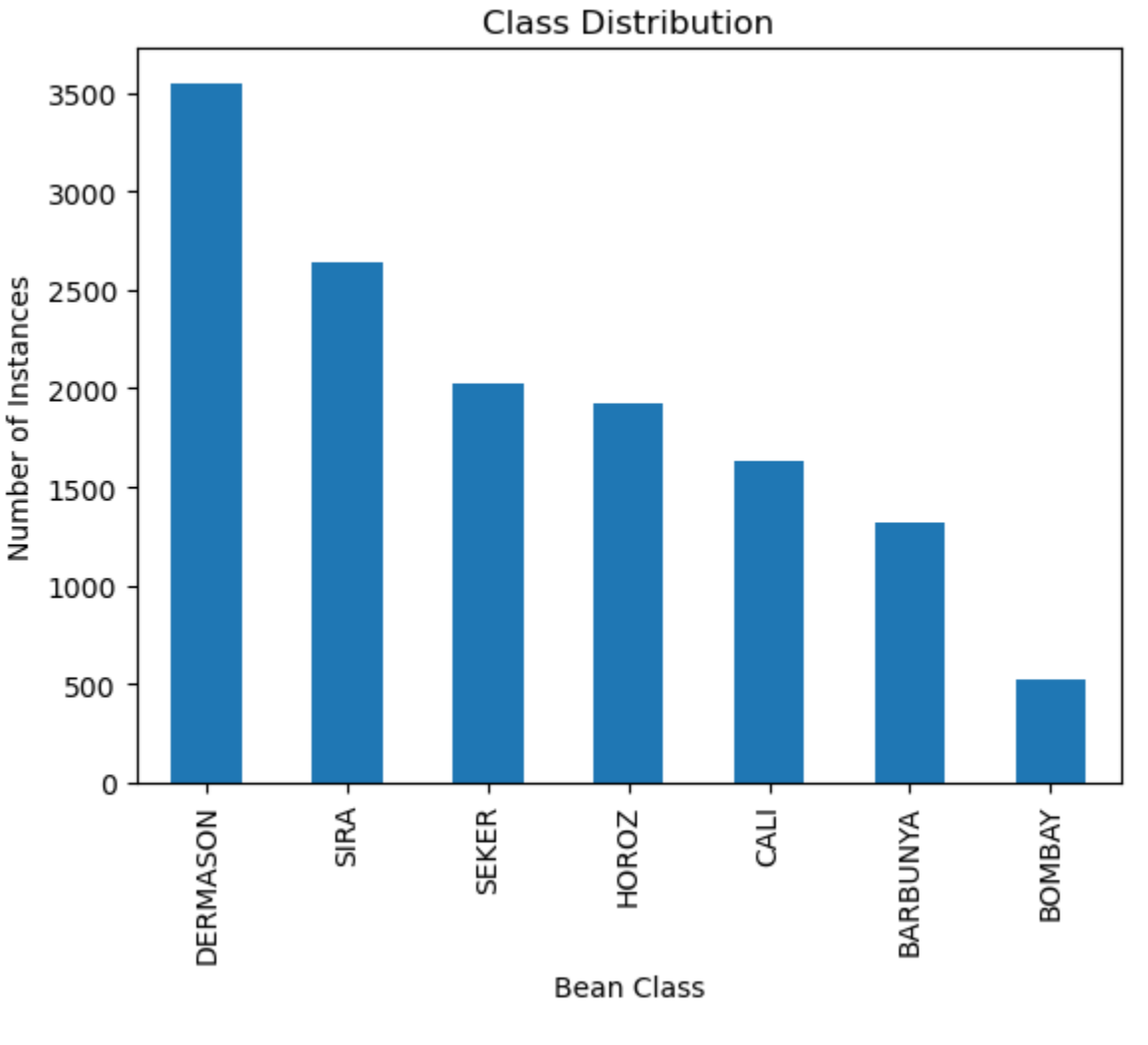
Out[6]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFactor
0	28395	610.291	208.178117	173.888747	1.197191	0.549812	28715	190.141097	0.763923	0.988856	0.958027	0.913358	0.007332	0.00314
1	28734	638.018	200.524796	182.734419	1.097356	0.411785	29172	191.272751	0.783968	0.984986	0.887034	0.953861	0.006979	0.00356
2	29380	624.110	212.826130	175.931143	1.209713	0.562727	29690	193.410904	0.778113	0.989559	0.947849	0.908774	0.007244	0.00304
3	30008	645.884	210.557999	182.516516	1.153638	0.498616	30724	195.467062	0.782681	0.976696	0.903936	0.928329	0.007017	0.00321
4	30140	620.134	201.847882	190.279279	1.060798	0.333680	30417	195.896503	0.773098	0.990893	0.984877	0.970516	0.006697	0.00366

In [25]:
```
# visualize the distribution of the target variable
sns.countplot(x='Class', data=df)
plt.title('Distribution of Bean Types')
plt.show()
```



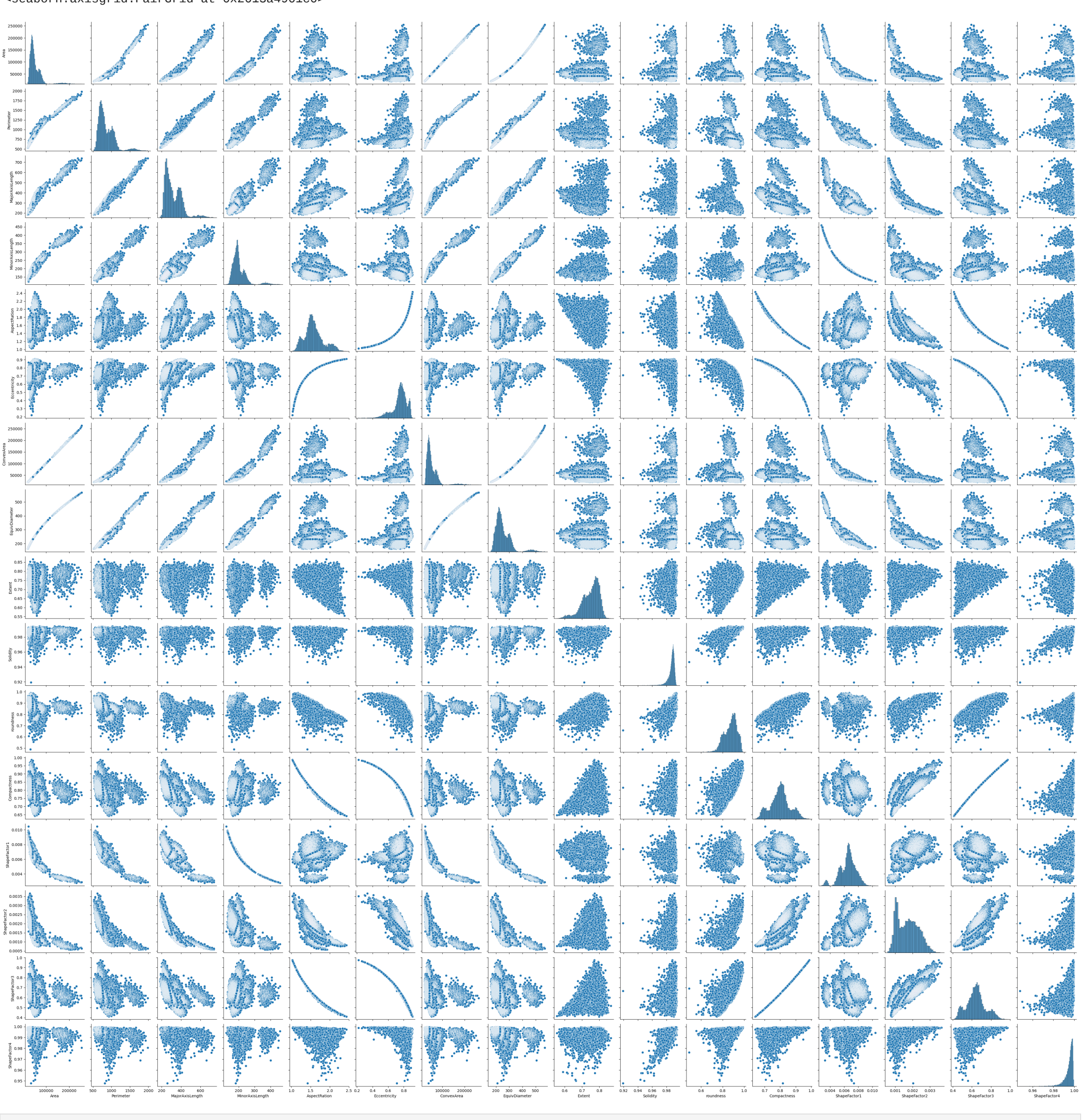
In [26]:
```
# plot the class distribution
class_counts = df['Class'].value_counts()
class_counts.plot(kind='bar')
plt.xlabel('Bean Class')
plt.ylabel('Number of Instances')
plt.title('Class Distribution')
plt.show()
```



pair plot

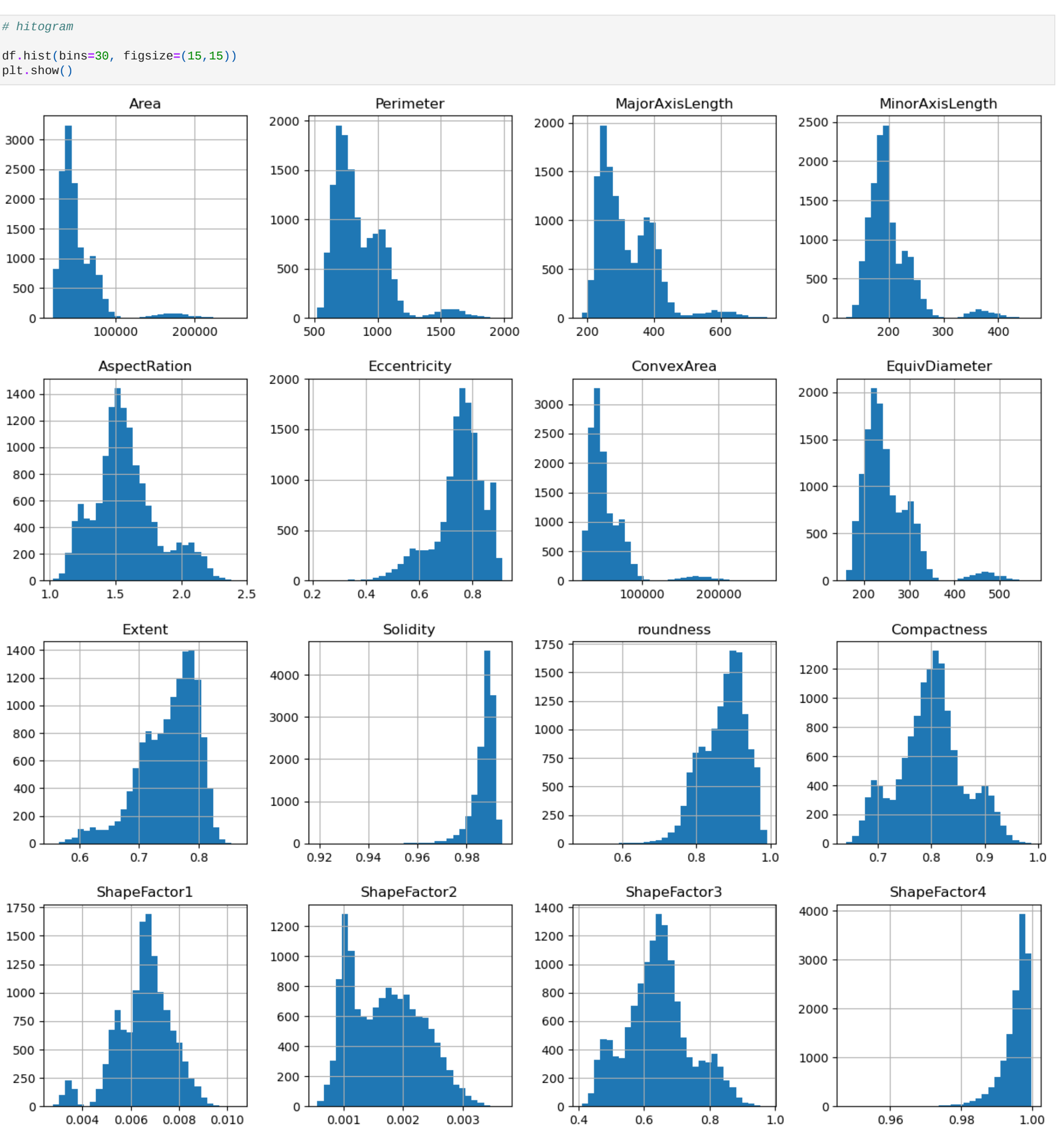
In [6]:
```
sns.pairplot(df)
```

Out[6]:
```
<seaborn.axisgrid.PairGrid at 0x2613a4901c0>
```



In [10]:
```
# hitogram

df.hist(bins=30, figsize=(15,15))
plt.show()
```



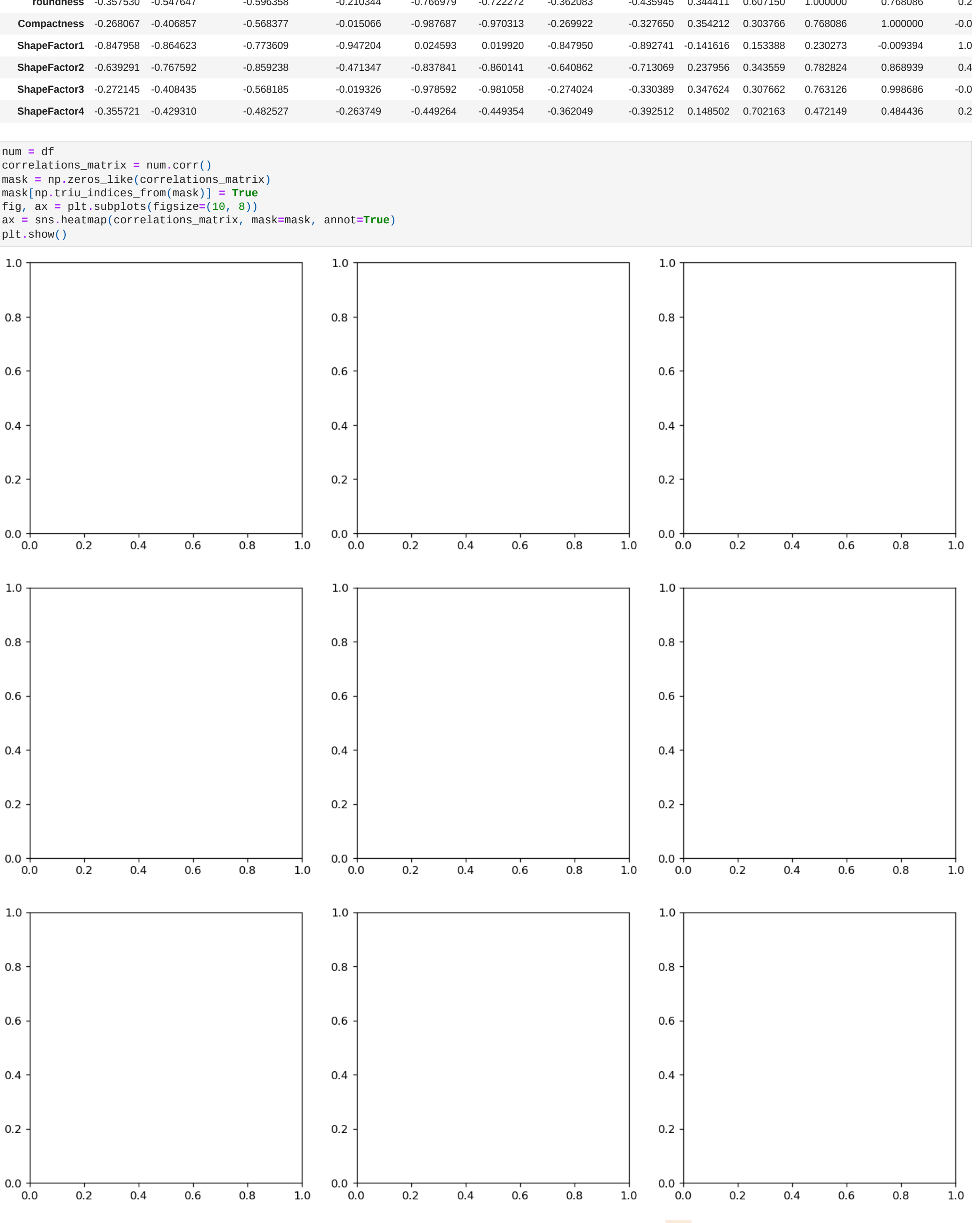
Heat map

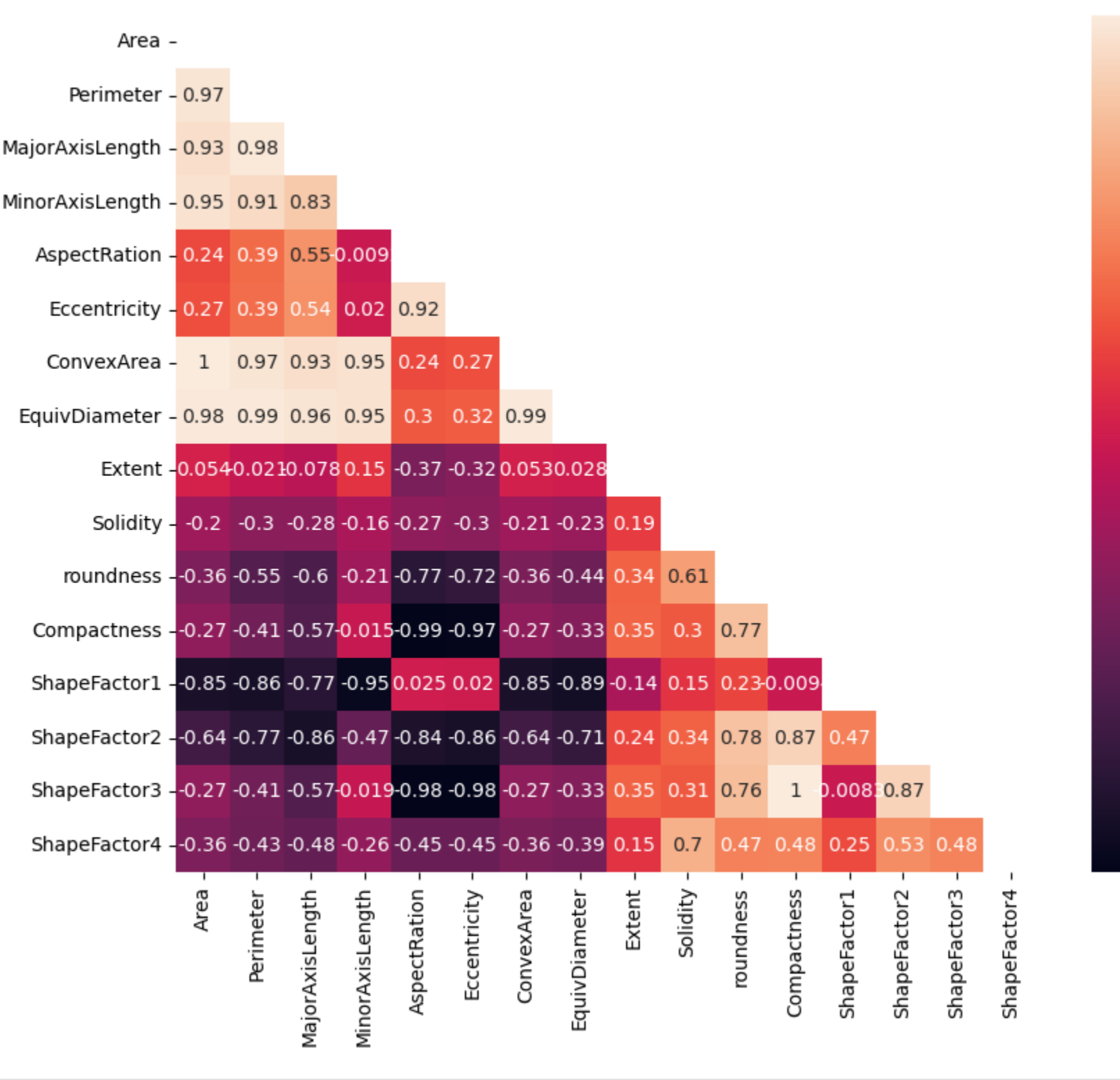
In [7]:
```
df.corr()
```

Out[7]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFa
Area	1.000000	0.966722	0.931834	0.951602	0.241735	0.267481	0.999939	0.984968	0.054345	-0.196585	-0.357530	-0.268067	-0.8
Perimeter	0.966722	1.000000	0.977338	0.913179	0.385276	0.391066	0.967689	0.991380	-0.021160	-0.303970	-0.547647	-0.406857	-0.8
MajorAxisLength	0.931834	0.977338	1.000000	0.826052	0.550335	0.541972	0.932607	0.961733	-0.078062	-0.284302	-0.596358	-0.568377	-0.7
MinorAxisLength	0.951602	0.913179	0.826052	1.000000	-0.009161	0.019574	0.951339	0.948539	0.145957	-0.155831	-0.210344	-0.015066	-0.9
AspectRation	0.241735	0.385276	0.550335	-0.009161	1.000000	0.924293	0.243301	0.303647	-0.370184	-0.267754	-0.766979	-0.987687	0.0
Eccentricity	0.267481	0.391066	0.541972	0.019574	0.924293	1.000000	0.269255	0.318667	-0.319362	-0.297592	-0.722272	-0.970313	0.0
ConvexArea	0.999939	0.967689	0.932607	0.951339	0.243301	0.269255	1.000000	0.985226	0.052564	-0.206191	-0.362083	-0.269922	-0.8
EquivDiameter	0.984968	0.991380	0.961733	0.948539	0.303647	0.318667	0.985226	1.000000	0.028383	-0.231648	-0.435945	-0.327650	-0.8
Extent	0.054345	-0.021160	-0.078062	0.145957	-0.370184	-0.319362	0.052564	0.028383	1.000000	0.191389	0.344411	0.354212	-0.1
Solidity	-0.196585	-0.303970	-0.284302	-0.155831	-0.267754	-0.297592	-0.206191	-0.231648	0.191389	1.000000	0.607150	0.303766	0.1
roundness	-0.357530	-0.547647	-0.596358	-0.210344	-0.766979	-0.722272	-0.362083	-0.435945	0.344411	0.607150	1.000000	0.768086	0.2
Compactness	-0.268067	-0.406857	-0.568377	-0.015066	-0.987687	-0.970313	-0.269922	-0.327650	0.354212	0.303766	0.768086	1.000000	-0.0
ShapeFactor1	-0.847958	-0.864623	-0.773609	-0.947204	0.024593	0.019920	-0.847950	-0.892741	-0.141616	0.153388	0.230273	-0.009394	1.0
ShapeFactor2	-0.639291	-0.767592	-0.859238	-0.471347	-0.837841	-0.860141	-0.640862	-0.713069	0.237956	0.343559	0.782824	0.868939	0.4
ShapeFactor3	-0.272145	-0.408435	-0.568185	-0.019326	-0.978592	-0.981058	-0.274024	-0.330389	0.347624	0.307662	0.763126	0.998686	-0.0
ShapeFactor4	-0.355721	-0.429310	-0.482527	-0.263749	-0.449264	-0.449354	-0.362049	-0.392512	0.148502	0.702163	0.472149	0.484436	0.2

In [10]:
```
num = df
correlations_matrix = num.corr()
mask = np.zeros_like(correlations_matrix)
mask[np.triu_indices_from(mask)] = True
fig, ax = plt.subplots(figsize=(10, 8))
ax = sns.heatmap(correlations_matrix, mask=mask, annot=True)
plt.show()
```





In [ ]:
```
# Label Encoding
```

In [11]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
```

In [14]:
```
labelencoder = LabelEncoder()
df['Class']= labelencoder.fit_transform(df['Class'])

df.head()
```

Out[14]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFactor
0	28395	610.291	208.178117	173.888747	1.197191	0.549812	28715	190.141097	0.763923	0.988856	0.958027	0.913358	0.007332	0.00314
1	28734	638.018	200.524796	182.734419	1.097356	0.411785	29172	191.272751	0.783968	0.984986	0.887034	0.953861	0.006979	0.00356
2	29380	624.110	212.826130	175.931143	1.209713	0.562727	29690	193.410904	0.778113	0.989559	0.947849	0.908774	0.007244	0.00304
3	30008	645.884	210.557999	182.516516	1.153638	0.498616	30724	195.467062	0.782681	0.976696	0.903936	0.928329	0.007017	0.00321
4	30140	620.134	201.847882	190.279279	1.060798	0.333680	30417	195.896503	0.773098	0.990893	0.984877	0.970516	0.006697	0.00366

In [15]:
```
df.tail()
```

Out[15]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFa
13606	42097	759.696	288.721612	185.944705	1.552728	0.765002	42508	231.515799	0.714574	0.990331	0.916603	0.801865	0.006858	0.0
13607	42101	757.499	281.576392	190.713136	1.476439	0.735702	42494	231.526798	0.799943	0.990752	0.922015	0.822252	0.006688	0.0
13608	42139	759.321	281.539928	191.187979	1.472582	0.734065	42569	231.631261	0.729932	0.989899	0.918424	0.822730	0.006681	0.0
13609	42147	763.779	283.382636	190.275731	1.489326	0.741055	42667	231.653247	0.705389	0.987813	0.907906	0.817457	0.006724	0.0
13610	42159	772.237	295.142741	182.204716	1.619841	0.786693	42600	231.686223	0.788962	0.989648	0.888380	0.784997	0.007001	0.0

In [12]:
```
X = df.drop(columns='Class')
y = df['Class']
```

In [17]:
```
X
```

Out[17]:

	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFa
0	28395	610.291	208.178117	173.888747	1.197191	0.549812	28715	190.141097	0.763923	0.988856	0.958027	0.913358	0.007332	0.0
1	28734	638.018	200.524796	182.734419	1.097356	0.411785	29172	191.272751	0.783968	0.984986	0.887034	0.953861	0.006979	0.0
2	29380	624.110	212.826130	175.931143	1.209713	0.562727	29690	193.410904	0.778113	0.989559	0.947849	0.908774	0.007244	0.0
3	30008	645.884	210.557999	182.516516	1.153638	0.498616	30724	195.467062	0.782681	0.976696	0.903936	0.928329	0.007017	0.0
4	30140	620.134	201.847882	190.279279	1.060798	0.333680	30417	195.896503	0.773098	0.990893	0.984877	0.970516	0.006697	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
13606	42097	759.696	288.721612	185.944705	1.552728	0.765002	42508	231.515799	0.714574	0.990331	0.916603	0.801865	0.006858	0.0
13607	42101	757.499	281.576392	190.713136	1.476439	0.735702	42494	231.526798	0.799943	0.990752	0.922015	0.822252	0.006688	0.0
13608	42139	759.321	281.539928	191.187979	1.472582	0.734065	42569	231.631261	0.729932	0.989899	0.918424	0.822730	0.006681	0.0
13609	42147	763.779	283.382636	190.275731	1.489326	0.741055	42667	231.653247	0.705389	0.987813	0.907906	0.817457	0.006724	0.0
13610	42159	772.237	295.142741	182.204716	1.619841	0.786693	42600	231.686223	0.788962	0.989648	0.888380	0.784997	0.007001	0.0

13611 rows × 16 columns

In [18]:
```
y
```

Out[18]:
```
0           SEKER
1           SEKER
2           SEKER
3           SEKER
4           SEKER
           ...
13606    DERMASON
13607    DERMASON
13608    DERMASON
13609    DERMASON
13610    DERMASON
Name: Class, Length: 13611, dtype: object
```

In [19]:
```
X_train, X_test, y_train, y_test=train_test_split(X,y, test_size=0.2, random_state=42)
```

In [32]:
```
X.shape
```

Out[32]:
```
(13611, 16)
```

In [24]:
```
y.shape
```

Out[24]:
```
(13611,)
```

In [27]:
```
print('X_train shape:', X_train.shape)
print('X_test shape:', X_test.shape)
print('y_train shape:', y_train.shape)
print('y_test shape:', y_test.shape)
```

```
X_train shape: (10888, 16)
X_test shape: (2723, 16)
y_train shape: (10888,)
y_test shape: (2723,)
```

In [28]:
```
clf = DecisionTreeClassifier()

clf.fit(X_train, y_train)
```

Out[28]:
```
DecisionTreeClassifier()
```

In [39]:
```
y_pred=clf.predict(X_test)
y_pred
```

Out[39]:
```
array(['SEKER', 'BARBUNYA', 'SEKER', ..., 'DERMASON', 'CALI', 'DERMASON'],
      dtype=object)
```

In [36]:
```
acc = metrics.accuracy_score(y_test,y_pred)
print('Accuracy :',acc)
```

```
Accuracy : 0.8898273962541314
```

In [ ]:

In [ ]:
```
# name = Bhingarde Sakshi Sunil
```

In [ ]: