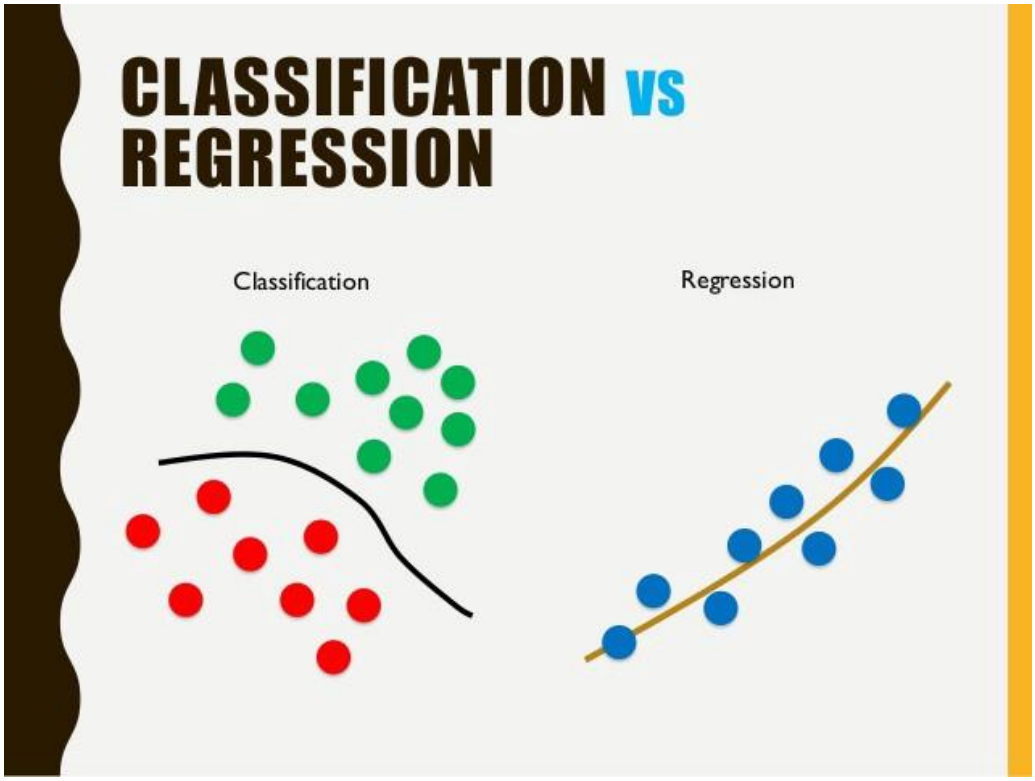
NED UNIVERISTY

OF ENGINEERING & TECHNOLOGY

Machine Learning Report



INSTRUCTOR:	Ms. Hameeza Ahmed

Group Members

SYED UMAIR HASAN	MUHAMMAD FAHAD ALAM
CS-18070	CS-18075
3rd Year, CIS	3rd Year, CIS

Rice Type Classification

A Description of Dataset

Problem of Interest

The data in the dataset was extracted from two kinds of rice, 'Gonen', 'Jasmine'. Our task is to train a model which will classify the type of rice, whether it is 'Gonen' or 'Jasmine'.

Data Source

The data was taken from the Kaggle website.

A Detailed explanation of the data attributes

Data issues and description

The dataset contains 18185 rows and 12 columns. All attributes are numeric variables and they are listed below:

- id
- Area
- MajorAxisLength
- MinorAxisLength
- Eccentricity
- ConvexArea
- EquivDiameter
- Extent
- Perimeter
- Roundness
- AspectRation
- Class

Fortunately, this dataset contains no missing values, there are no categorical features and the dataset is also well balanced. However, it contains some features which have outliers.

Data Preprocessing / Wrangling

Data Cleaning

As the dataset contains outliers in some of the features, therefore those outliers were removed by first calculating the interquartile range and then finding the upper and lower range. Any value which is out of that range is considered as an outlier and then removed.

Data Reduction

The 'Id' column was removed from the dataset as it was irrelevant. The 'Class' column was also dropped from the dataset as it was a target variable.

Data Editing

Many of the columns were not normalized in the dataset therefore , all the input columns were passed through a standard scaler to normalize the data.

Basic statistics of attributes

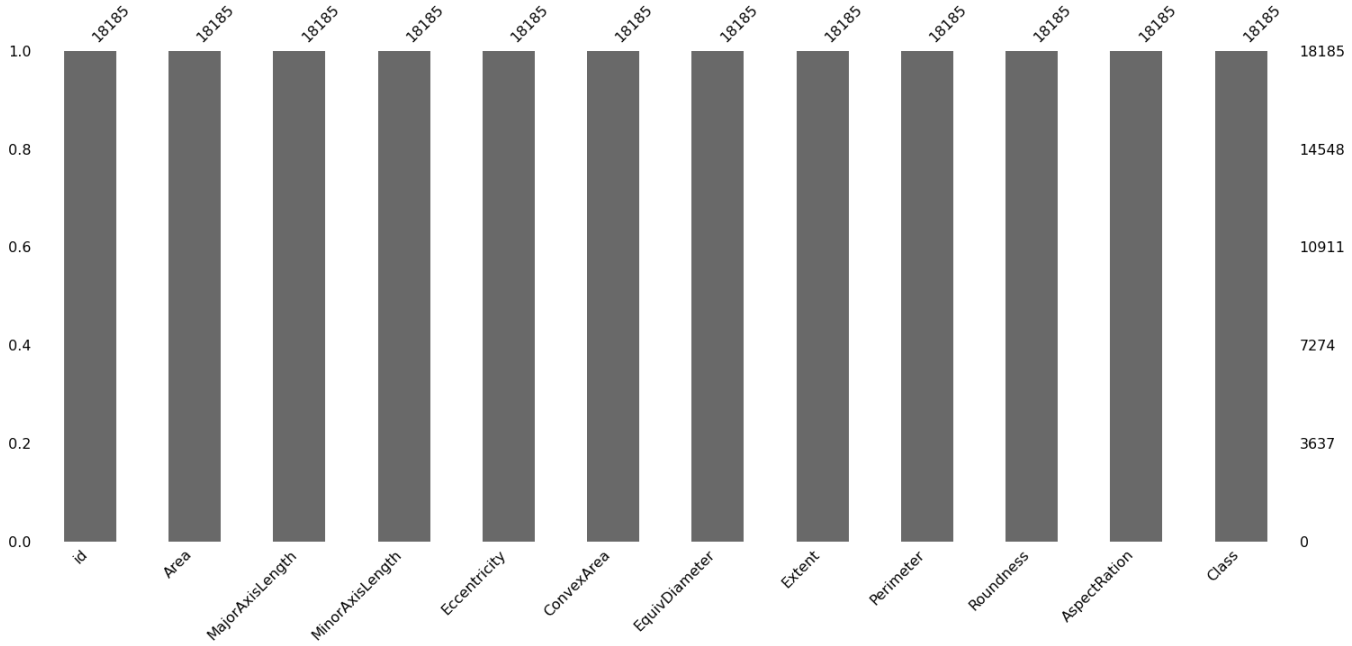
Description of Numerical Attributes

	Area	MajorAxisLength	MinorAxisLength	Eccentricity	ConvexArea	EquivDiameter	Extent	Perimeter	Roundness	AspectRation
count	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000	17597.000000
mean	7111.463772	152.978496	59.961723	0.917226	7301.974086	94.689501	0.615732	353.964315	0.706197	2.615482
std	1412.424549	9.926544	10.004501	0.027213	1445.997988	9.407461	0.104490	25.733769	0.065566	0.424586
min	3698.000000	124.110890	36.177914	0.819338	3883.000000	68.618072	0.383239	280.060000	0.474142	1.744254
25%	6018.000000	146.610657	51.456124	0.892386	6180.000000	87.534882	0.536760	335.976000	0.650238	2.215959
50%	6709.000000	154.289429	55.917314	0.925461	6892.000000	92.423829	0.600259	354.276000	0.697445	2.639618
75%	8464.000000	160.280519	70.225361	0.941634	8687.000000	103.810883	0.694617	373.589000	0.768167	2.970556
max	10210.000000	180.332508	82.550762	0.966774	10659.000000	114.016559	0.886137	429.919000	0.854461	3.911845

Data visualization

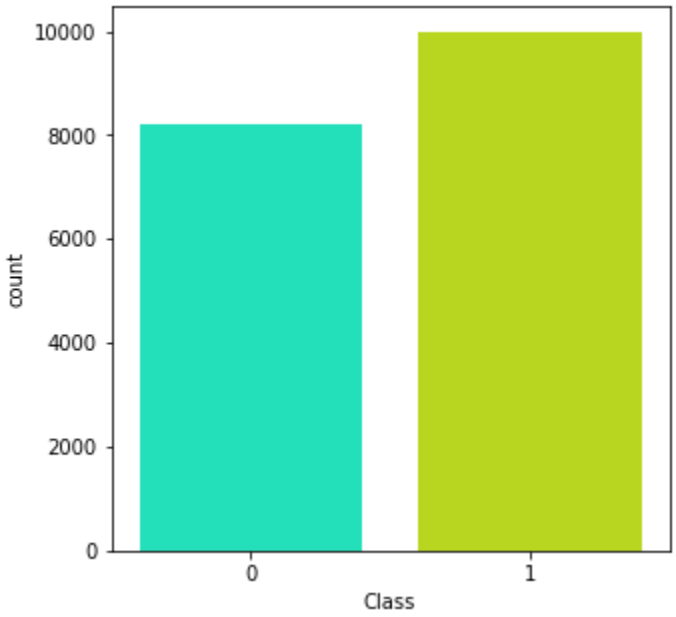
The distribution of the attributes

The bar graph shows that there are no missing values in dataset.

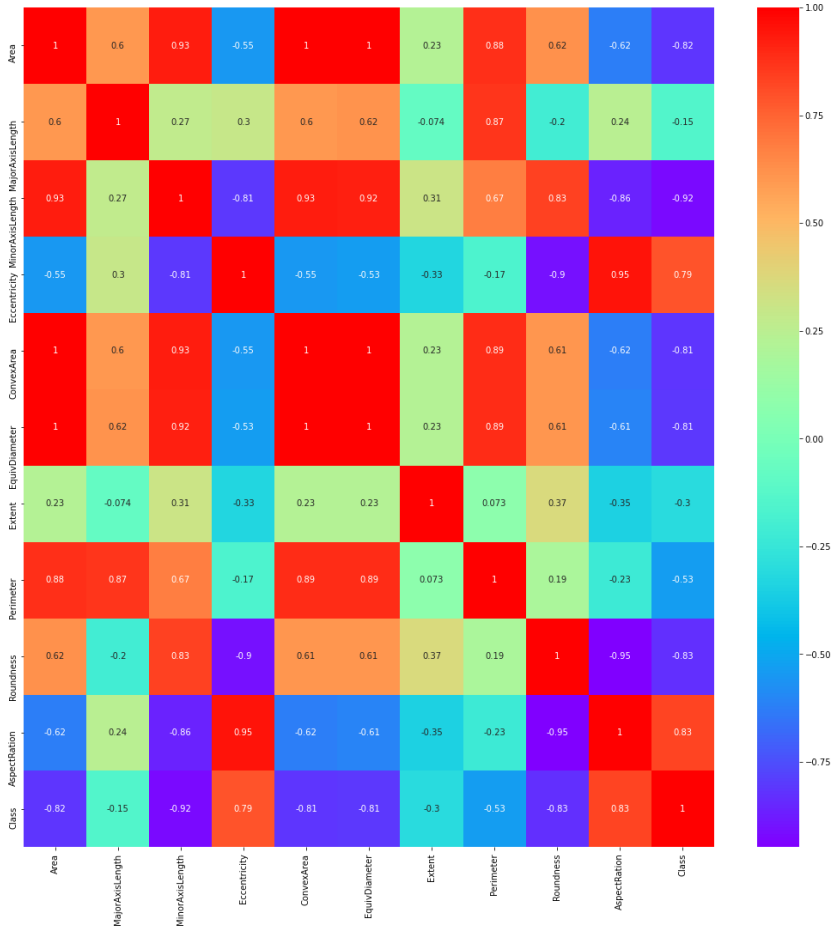


Checking whether dataset is balanced or not

The figure below shows that the dataset is balanced.

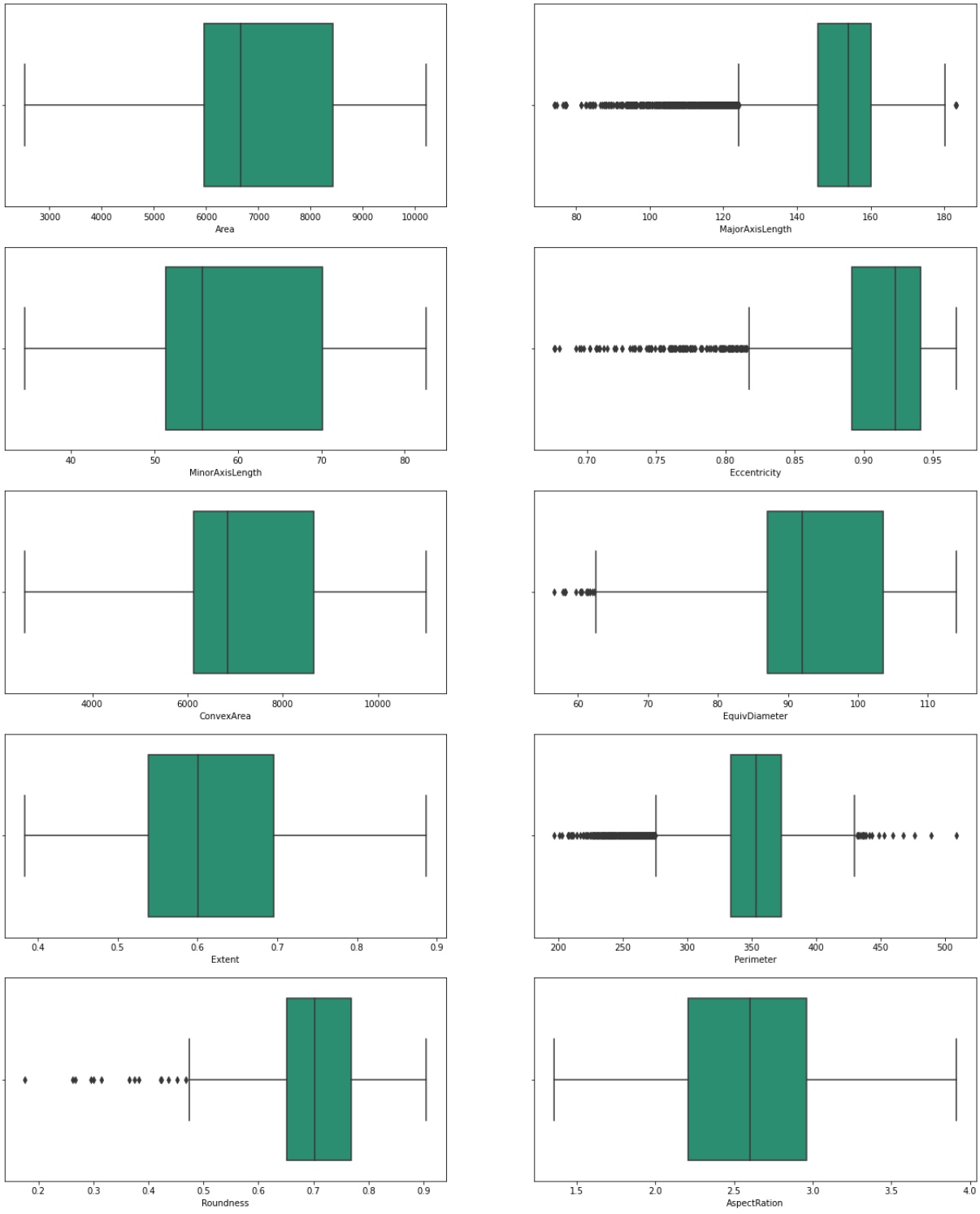


Correlation Matrix



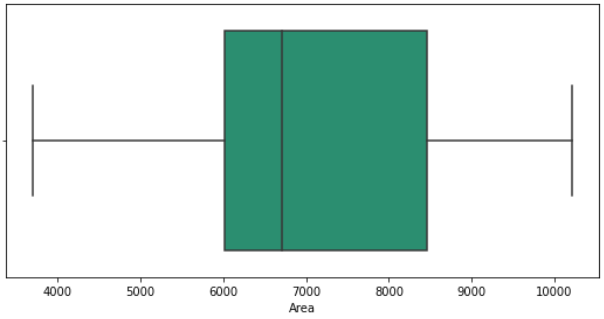
Box Plot before Outliers detection

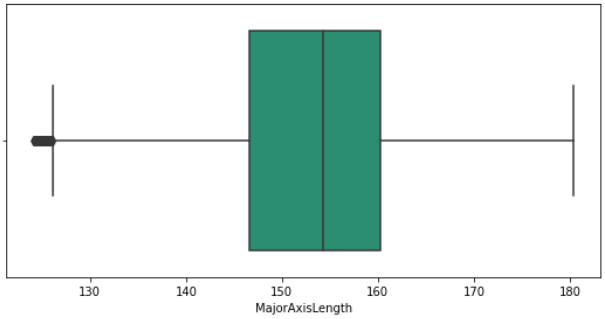
The figure below confirms that some features contain outliers.

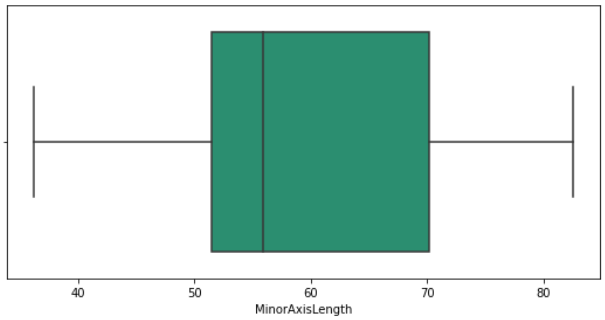


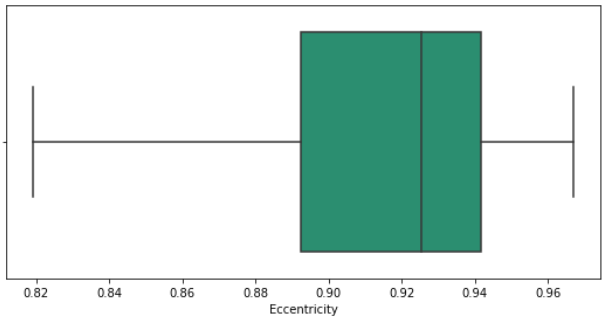
Box Plot after removing Outliers

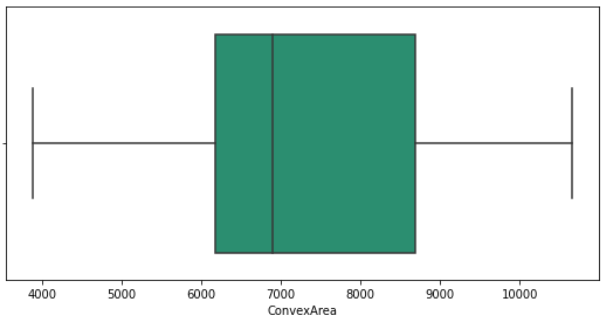
Now the outliers have been removed.

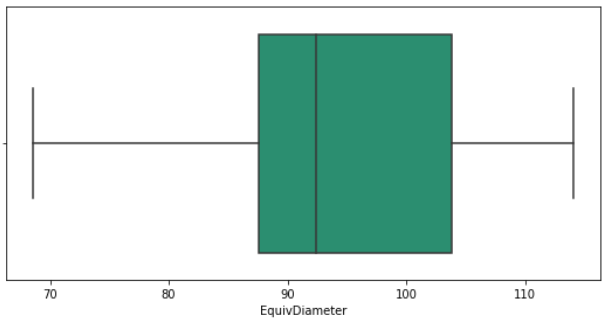


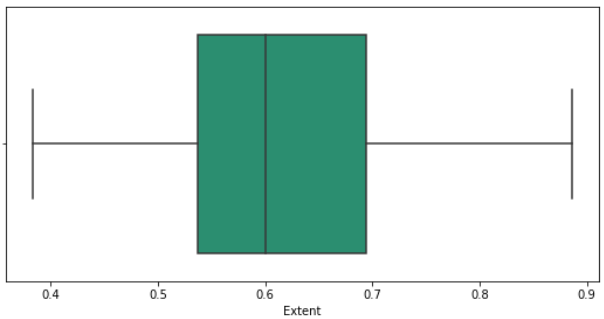


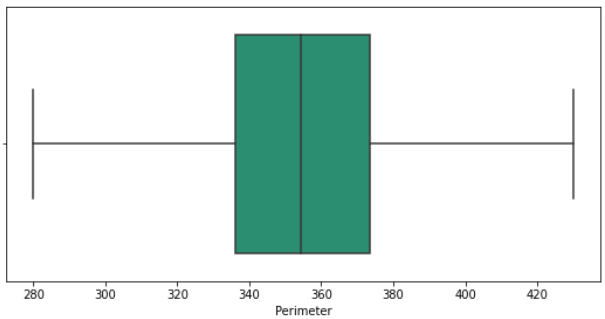


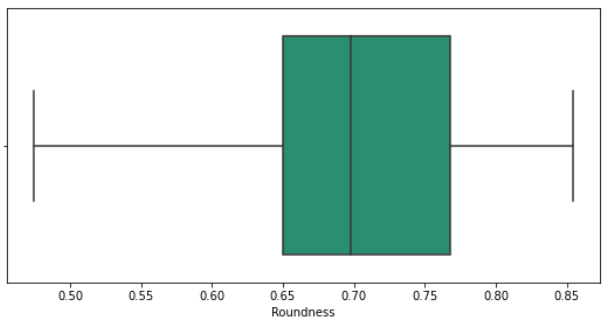


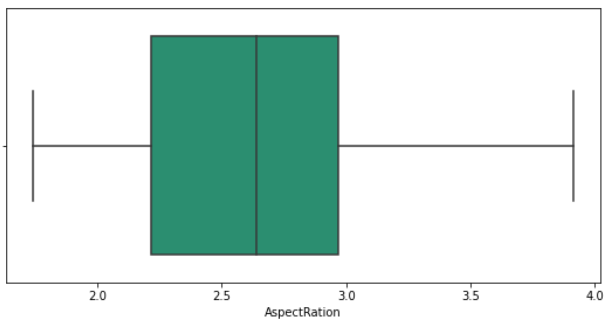




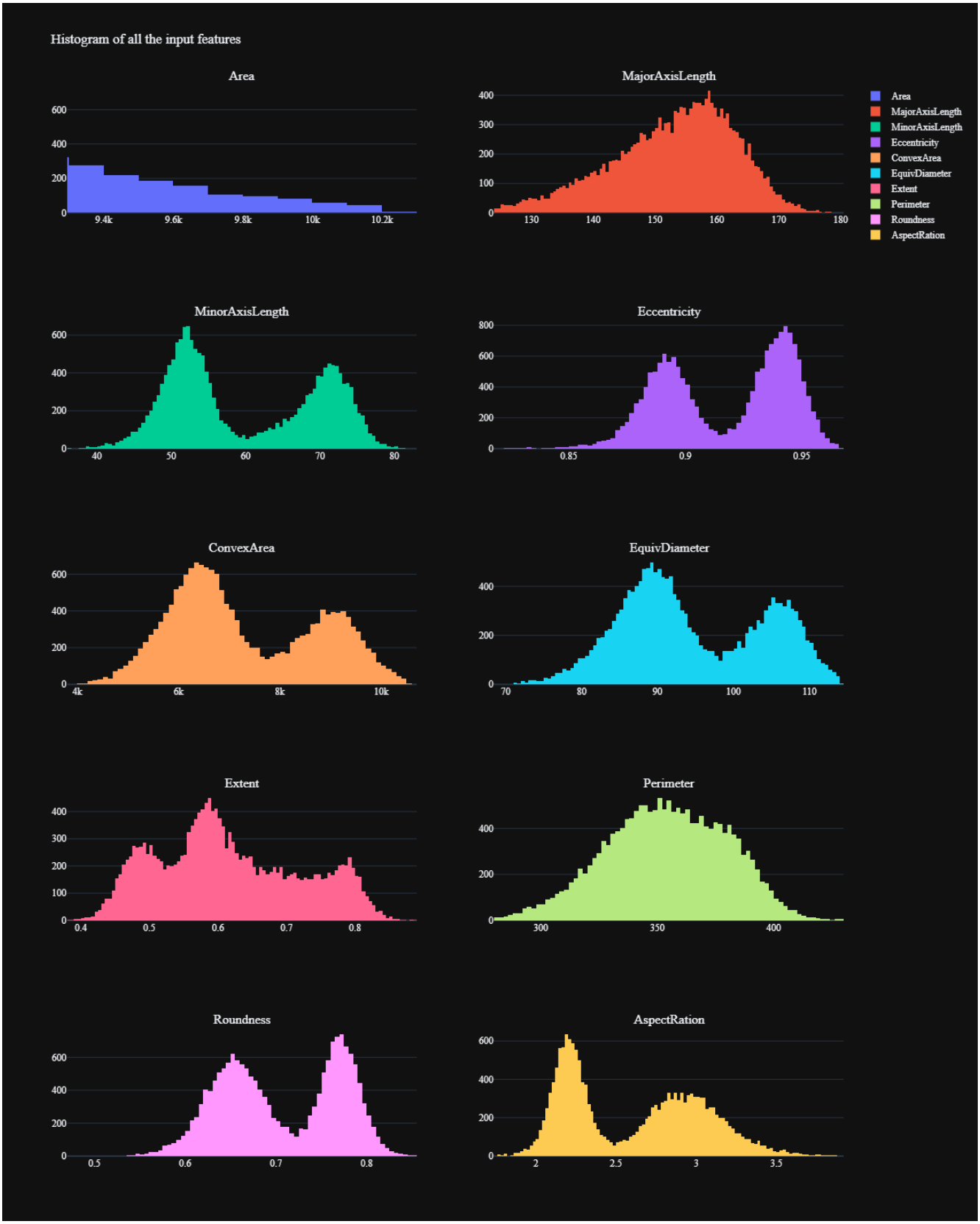








Histogram of all Features



Classification:

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. We have to select the input and output of the models.

Input:

As we have seen the correlation of the attributes of the dataset. Therefore, following attributes were taken as input:

- Area
- MajorAxisLength
- MinorAxisLength
- Eccentricity
- ConvexArea
- EquivDiameter
- Extent
- Perimeter
- Roundness
- AspectRation

We would use Degree Trunc, Distance,Section and Height as the input to the classification model.

Output:

'Class' is the target variable in our dataset.

- Jasmine - 1
- Gonen - 0

Model

We have split our data 30% and 70% for testing and training respectively. We applied Classification with Input Provided above and predicted Output.

We have used the following classification models:

1. Logistic Regression
   The logistic model is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick.

2. Naive Bayes Classifier
   A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem.

3. KNeighborsClassifier
   KNeighborsClassifier is based on the k nearest neighbors of a sample, which has to be classified. The number 'k' is an integer value specified by the user. This is the most frequently used classifier of both algorithms.

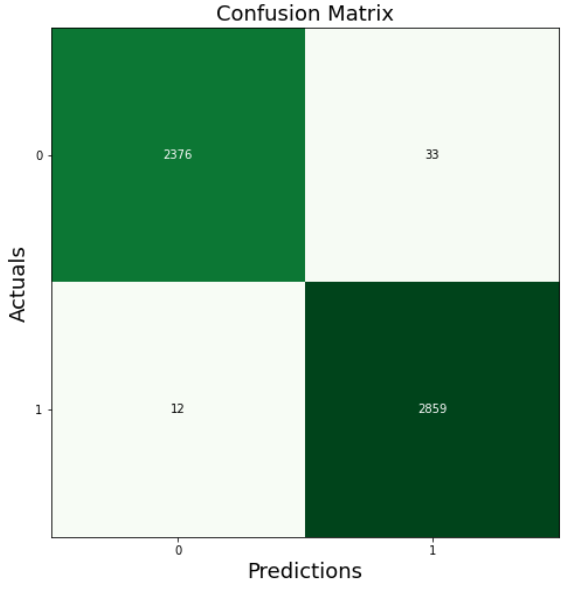
Evaluation of Models

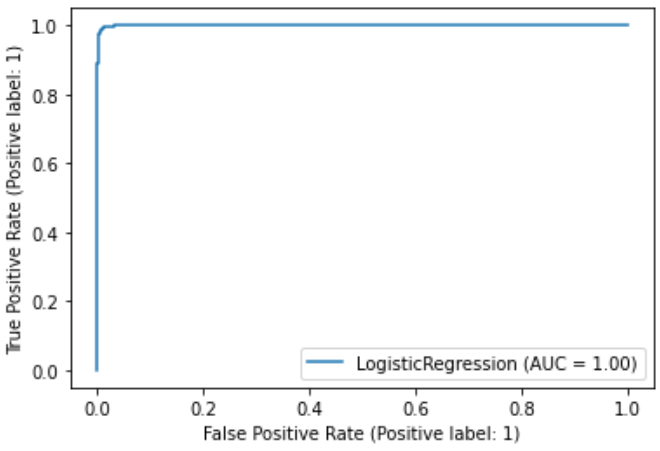
The following table shows the list of performance metrics of these models against the training data is as follows

	Logistic Regression	Naive Bayes Classifier	K-Neighbors Classifier
ACCURACY	99.14	98.90	98.86
RECALL RATE	Class 0 - 0.99 Class 1 - 1.00	Class 0 - 0.98 Class 1 - 1.00	Class 0 - 0.98 Class 1 - 0.99
PRECISION	Class 0 - 0.99 Class 1 - 0.99	Class 0 - 1.00 Class 1 - 0.98	Class 0 - 0.99 Class 1 - 0.99
FPR	Class 0 - 0.004 Class 1 - 0.013	Class 0 - 0.002 Class 1 - 0.021	Class 0 - 0.007 Class 1 - 0.015
AUC	1.00	1.00	0.99

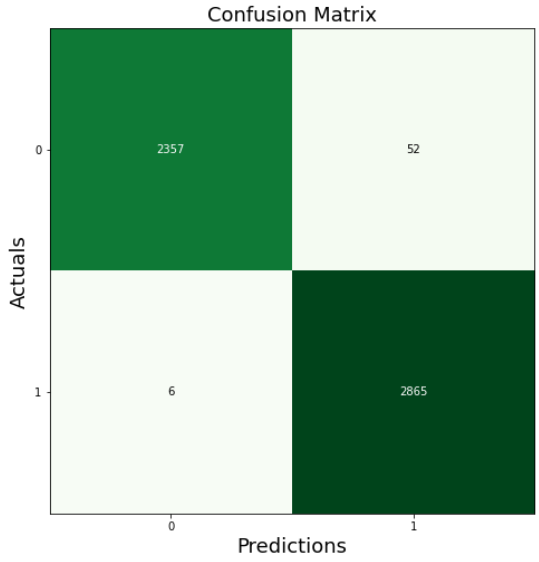
It is quite evident from the table that all models are correctly classifying the type of rice as accuracy is quite high along with high value of AUC. Logistic Regression is the most accurate.

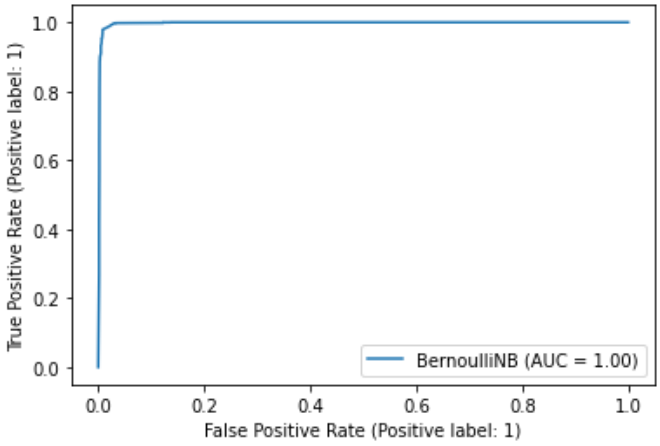
Logistic Regression



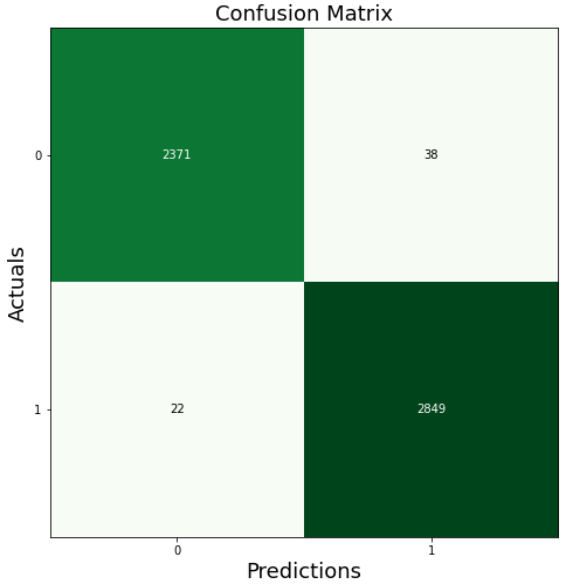


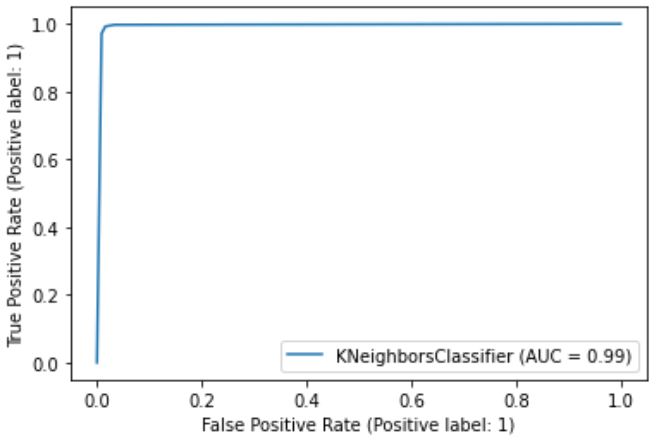
Naive Bayes Classifier





KNeighborsClassifier





Leave One Out Cross Validation

The accuracy score of LOOCV on all 3 models is 1.00. Which means all three models are generalized and working well on unseen data.

Conclusion

Conclusively, we get to know that the dataset was quite clean, and all of the features were relevant except for the 'id'. The models used were classifying the rice type accurately with high precision and low false positive rate.

Asteroid Regression

A Description of Dataset

Problem of Interest

The data in the dataset is about the different features of Asteroid and the properties. Our task is to train a model which will predict the distance of the asteroid from the earth.

Data Source

The data was taken from the Kaggle website.

A Detailed explanation of the data attributes

Data issues and description

The dataset contains 958524 rows and 45 columns.

We have 45 features in total

- 35 are numeric in nature.
- 10 are categorical in nature.

There are some columns having missing values. We would be dealing with them in the data cleaning phase.

Features with their short descriptions are listed below:

- SPK-ID: Object primary SPK-ID
- Object ID: Object internal database ID
- Object full name: Object full name/designation
- pdes: Object primary designation
- name: Object IAU name
- NEO: Near-Earth Object (NEO) flag
- PHA: Potentially Hazardous Asteroid (PHA) flag
- H: Absolute magnitude parameter
- Diameter: object diameter (from equivalent sphere) km Unit
- Albedo: Geometric albedo
- Diameter_sigma: 1-sigma uncertainty in object diameter km Unit
- Orbit_id: Orbit solution ID
- Epoch: Epoch of osculation in modified Julian day form
- Equinox: Equinox of reference frame
- e: Eccentricity
- a: Semi-major axis au Unit
- q: perihelion distance au Unit
- i: inclination; angle with respect to x-y ecliptic plane
- tp: Time of perihelion passage TDB Unit
- moid_ld: Earth Minimum Orbit Intersection Distance au Unit

Unfortunately, this dataset contains missing values as well as categorical features. Some outliers are also present in the data. We would be dealing with these issues in subsequent sections.

Name and prefix columns have many missing values. Id, full_name, pdes, name columns have every value unique. Equinox columns have same value in every row.

In our data only 0.22% asteroids are Potentially Hazardous Asteroid. Other 99.78% asteroids are not potentially hazardous. According to our data only 2.39% asteroids are Near-Earth Object (NEO). Other 97.61% asteroids are not near-earth. In the data we are provided asteroids are divided into 13 classes with 89.3% asteroids in MBA class which is most common class in our data.

Descriptive Statistics

Some Samples of Data

```
asteroid_df.head()
```

	id	spkid	full_name	pdes	name	prefix	neo	pha	H	diameter	albedo	diameter_sigma	orbit_id	epoch	epoch_mjd	epoch_cal	equinox	
0	a0000001	2000001	1 Ceres	1	Ceres	NaN	N	N	3.40	939.400	0.0900	0.200	JPL 47	2458600.5	58600	20190427.0	J2000	0.0760
1	a0000002	2000002	2 Pallas	2	Pallas	NaN	N	N	4.20	545.000	0.1010	18.000	JPL 37	2459000.5	59000	20200531.0	J2000	0.2299
2	a0000003	2000003	3 Juno	3	Juno	NaN	N	N	5.33	246.596	0.2140	10.594	JPL 112	2459000.5	59000	20200531.0	J2000	0.2569
3	a0000004	2000004	4 Vesta	4	Vesta	NaN	N	N	3.00	525.400	0.4228	0.200	JPL 35	2458600.5	58600	20190427.0	J2000	0.0887
4	a0000005	2000005	5 Astraea	5	Astraea	NaN	N	N	6.90	106.699	0.2740	3.140	JPL 114	2459000.5	59000	20200531.0	J2000	0.1909

Information about the features

```
[8] asteroid_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 958524 entries, 0 to 958523
Data columns (total 45 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   id              958524 non-null  object
 1   spkid           958524 non-null  int64
 2   full_name       958524 non-null  object
 3   pdes            958524 non-null  object
 4   name            22064 non-null   object
 5   prefix          18 non-null      object
 6   neo             958520 non-null  object
 7   pha             938603 non-null  object
 8   H               952261 non-null  float64
 9   diameter        136209 non-null  float64
 10  albedo          135103 non-null  float64
 11  diameter_sigma  136081 non-null  float64
 12  orbit_id        958524 non-null  object
 13  epoch           958524 non-null  float64
 14  epoch_mjd       958524 non-null  int64
 15  epoch_cal       958524 non-null  float64
 16  equinox         958524 non-null  object
 17  e               958524 non-null  float64
 18  a               958524 non-null  float64
```

```
 19  q              958524 non-null  float64
 20  i              958524 non-null  float64
 21  om             958524 non-null  float64
 22  w              958524 non-null  float64
 23  ma             958523 non-null  float64
 24  ad             958520 non-null  float64
 25  n              958524 non-null  float64
 26  tp             958524 non-null  float64
 27  tp_cal         958524 non-null  float64
 28  per            958520 non-null  float64
 29  per_y          958523 non-null  float64
 30  moid           938603 non-null  float64
 31  moid_ld        958397 non-null  float64
 32  sigma_e        938602 non-null  float64
 33  sigma_a        938602 non-null  float64
 34  sigma_q        938602 non-null  float64
 35  sigma_i        938602 non-null  float64
 36  sigma_om       938602 non-null  float64
 37  sigma_w        938602 non-null  float64
 38  sigma_ma       938602 non-null  float64
 39  sigma_ad       938598 non-null  float64
 40  sigma_n        938602 non-null  float64
 41  sigma_tp       938602 non-null  float64
 42  sigma_per      938598 non-null  float64
 43  class          958524 non-null  object
 44  rms            958522 non-null  float64
dtypes: float64(33), int64(2), object(10)
memory usage: 329.1+ MB
```

Central Tendencies, Quartile and standard Deviation

```
[9]  asteroid_df.describe()
```

	spkid	H	diameter	albedo	diameter_sigma	epoch	epoch_mjd	epoch_cal	e	a	
count	9.585240e+05	952261.000000	136209.000000	135103.000000	136081.000000	9.585240e+05	958524.000000	9.585240e+05	958524.000000	958524.000000	958524.
mean	3.810114e+06	16.906411	5.506429	0.130627	0.479184	2.458869e+06	58868.781950	2.019693e+07	0.156116	2.902143	2.
std	6.831541e+06	1.790405	9.425164	0.110323	0.782895	7.016716e+02	701.671573	1.930354e+04	0.092643	39.719503	2.
min	2.000001e+06	-1.100000	0.002500	0.001000	0.000500	2.425052e+06	25051.000000	1.927062e+07	0.000000	-14702.447872	0.
25%	2.239632e+06	16.100000	2.780000	0.053000	0.180000	2.459000e+06	59000.000000	2.020053e+07	0.092193	2.387835	1.
50%	2.479262e+06	16.900000	3.972000	0.079000	0.332000	2.459000e+06	59000.000000	2.020053e+07	0.145002	2.646969	2.
75%	3.752518e+06	17.714000	5.765000	0.190000	0.620000	2.459000e+06	59000.000000	2.020053e+07	0.200650	3.001932	2.
max	5.401723e+07	33.200000	939.400000	1.000000	140.000000	2.459000e+06	59000.000000	2.020053e+07	1.855356	33488.895955	80.

Describe Object Type features

```
[10] asteroid_df.describe(include=[object])
```

	id	full_name	pdes	name	prefix	neo	pha	orbit_id	equinox	class
count	958524	958524	958524	22064	18	958520	938603	958524	958524	958524
unique	958524	958524	958524	22064	1	2	2	4690	1	13
top	bK16UA6L	(2015 VV183)	2014 WV397	Oschin	A	N	N	1	J2000	MBA
freq	1	1	1	1	18	935625	936537	50142	958524	855954

Data Cleaning

Remove Unwanted Observations

We are removing id, pdes, prefix, name and equinox as they are unwanted observations, and they would not be any help in prediction

Fix Structural Errors

Converting Categorical Data which is in Object Format to Category. The category data type in pandas is a hybrid data type. It looks and behaves like a string in many instances but internally is represented by an array of integers. This allows the data to be sorted in a custom order and to store the data more efficiently.

Merging all orbit_id with less than 10 occurrences in single orbit_id named 'other'.

Filter out Missing Data

Removing all rows having no value for diameter as we are going to do predict diameter using regression. Filling diameter in anyway would not be any fruitful for us. We removed all rows having no values of 'diameter' and 'H' features.

Fill in Missing Data

Some values are still missing in our dataset. We can either remove the rows with missing values or fill in some appropriate value. In our case filling missing values would be appropriate.

Filling means in place of missing values in columns of feature 'albedo' and 'diameter_sigma'.

Data Transformation

Remove Duplicates

There are no duplicates in our data. So, we can skip this step.

Separating Categorical and Numeric Features

Separate categorical features 'full_name', 'neo', 'pha', 'orbit_id', 'class' from numeric features. As there are different ways to deal with numeric and categorical data. We would deal with them separately.

Hot Encoding

One hot encoding is a process of converting categorical data variables so they can be provided to machine learning algorithms to improve predictions. One hot encoding is a crucial part of feature engineering for machine learning.

We hot encode all categorical data and converted them in numerical format which is required for machine learning.

Normalization

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

We normalized all our numeric data features.

Conclusion

Now our data have no missing values. Numeric Features are normalized, and categorical data is hot encoded. Data is ready to be used in a machine learning model.

Dimensionality Reduction

Principle Component Analysis (PCA)

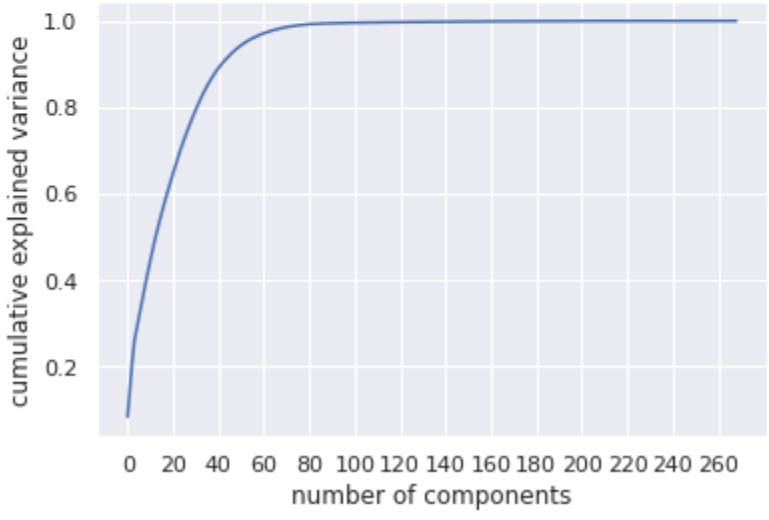
To Choose number of principle components we plot an explained variance graph

```
X = asteroid_df.drop(['diameter', "full_name"],axis=1).values

pca = PCA().fit(X)

plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('number of components')
plt.xticks(np.arange(0, 271, 20))
plt.ylabel('cumulative explained variance')
```

```
Text(0, 0.5, 'cumulative explained variance')
```



By Explained Variance Graph we can see that 40 components are defining around 90% of Variance while 60 components are defining around 97% of Variance.

We originally had 271 features. Reducing dimensions from 271 to 60 with 97% Variance is a good idea. As we lost only 3% Variance and cut features to train to 60 only.

We have reduced Features from 271 to 60 using PCA. 60 components are explaining 97% of Variance of the 271 features.

Regression:

In machine learning, regression refers to a predictive modeling problem where a continuous valued feature is predicted for given example of input data. We must select the input and output of the models.

Input:

After reducing the dataset, we are left with 60 features. We would be using all these 50 features as our input to the models.

Output:

Distance is the target variable.

Model

We have split our data 25% and 75% for testing and training respectively. We applied Regression with Input Provided above and predicted Output.

We have used the following regression models:

1. Simple Linear Regression
2. Polynomial Linear Regression
3. KNN Regressor
4. Decision Tree Regressor
5. SVM Regressor
6. Random Forest Regressor
7. MLP Regressor

Evaluation of Models

The following table shows the list of performance metrics of these models against the training data is as follows

	Model Name	Training R2 Score	Cross Validation R2 Score	Testing R2 Score	Training RMSE	Testing RMSE	Training MRE	Testing MRE
0	Linear Regression	0.215716	-1.648430e+00	1.755172e-01	0.008727	0.010053	inf	0.589134
1	Polynomial Linear Regression	0.855674	-1.091062e+06	-1.874937e+15	0.003744	479422.116498	inf	268973.384073
2	k-Nearest Neighbors (kNN) Regressor	0.731228	-6.748857e-01	3.584493e-01	0.005109	0.008868	inf	0.376861
3	Decision Tree Regressor	0.999366	-1.632503e+00	6.592317e-01	0.000248	0.006463	inf	0.155556
4	SVM Regressor	-84.195897	-4.081097e+01	-6.657276e+01	0.090953	0.091014	inf	23.517535
5	Random Forest Regressor	0.966062	-1.594701e-04	7.448580e-01	0.001815	0.005593	inf	0.121860
6	MLP Regressor	0.659187	-2.853582e+01	5.412986e-01	0.005753	0.007499	inf	0.840298

Conclusion

Random Forest performed the best on this dataset as the test R2 score is 0.745. Decision tree has R2 score of 0.66 so it performed great as well. Other models did not perform very well on this data.