

logistic Regression Project:Asteroid Classification

Data Acquisition

In [1]: `#Import Libraries`

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]: `#Load dataset`

```
data=pd.read_csv(r"C:\Users\DD\Desktop\Asteroid\nasa.csv")
```

In [3]: `#read the data`

```
data
```

Out[3]:

Dia in max)	Est Dia in Miles(min)	Est Dia in Miles(max)	Est Dia in Feet(min)	...	Asc Node Longitude	Orbital Period	Perihelion Distance	Perihelion Arg	Aphelion Di
2297	0.079051	0.176763	417.388066	...	314.373913	609.599786	0.808259	57.257470	2.00570
7897	0.090762	0.202951	479.225620	...	136.717242	425.869294	0.718200	313.091975	1.49730
4482	0.143849	0.321655	759.521423	...	259.475979	643.580228	0.950791	248.415038	1.96680
10675	0.005469	0.012229	28.876199	...	57.173266	514.082140	0.983902	18.707701	1.52790
2297	0.079051	0.176763	417.388066	...	84.629307	495.597821	0.967687	158.263596	1.48350
...
17028	0.027410	0.061290	144.723824	...	164.183305	457.179984	0.741558	276.395697	1.58120
5700	0.003784	0.008460	19.977449	...	345.225230	407.185767	0.996434	42.111064	1.15380
11509	0.047633	0.106510	251.501181	...	37.026468	690.054279	0.965760	274.692712	2.09070
11818	0.072095	0.161210	380.662441	...	163.802909	662.048343	1.185467	180.346090	1.78770
19655	0.248946	0.556661	1314.437764	...	187.642183	653.679098	0.876110	222.436688	2.07190

Information about data set

In [4]: `data.shape`

Out[4]: (4687, 40)

In [5]: `#top 10 rows`
data.head(10)

Out[5]:

Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	Est Dia in Miles(min)	Est Dia in Miles(max)
0	3703080	3703080	21.6	0.127220	0.284472	127.219878	284.472297	0.079051
1	3723955	3723955	21.3	0.146068	0.326618	146.067964	326.617897	0.090762

In [5]: #top 10 rows
data.head(10)

Out[5]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	Est Dia in Miles(min)	Est Dia in Miles(max)
0	3703080	3703080	21.6	0.127220	0.284472	127.219878	284.472297	0.079051	0.176
1	3723955	3723955	21.3	0.146068	0.326618	146.067964	326.617897	0.090762	0.202
2	2446862	2446862	20.3	0.231502	0.517654	231.502122	517.654482	0.143849	0.321
3	3092506	3092506	27.4	0.008801	0.019681	8.801465	19.680675	0.005469	0.012
4	3514799	3514799	21.6	0.127220	0.284472	127.219878	284.472297	0.079051	0.176
5	3671135	3671135	19.6	0.319562	0.714562	319.561887	714.562102	0.198566	0.444
6	2495323	2495323	19.6	0.319562	0.714562	319.561887	714.562102	0.198566	0.444
7	2153315	2153315	19.2	0.384198	0.859093	384.197891	859.092601	0.238729	0.533
8	2162463	2162463	17.8	0.732074	1.636967	732.073989	1636.967205	0.454890	1.017
9	2306383	2306383	21.5	0.133216	0.297879	133.215567	297.879063	0.082776	0.185

10 rows × 40 columns

In [6]: #bottom 10 rows
data.tail(10)

Out[6]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	Est Dia in Miles(min)	Est Dia in Miles(max)
4677	3759644	3759644	24.500	0.033462	0.074824	33.462237	74.823838	0.020792	0.0
4678	3358223	3358223	22.700	0.076658	0.171412	76.657557	171.411509	0.047633	0.1
4679	3394709	3394709	26.300	0.014607	0.032662	14.606796	32.661790	0.009076	0.0
4680	3608620	3608620	23.600	0.050647	0.113250	50.647146	113.250461	0.031471	0.0
4681	3662283	3662283	20.700	0.192555	0.430566	192.555078	430.566244	0.119648	0.2
4682	3759007	3759007	23.900	0.044112	0.098637	44.111820	98.637028	0.027410	0.0
4683	3759295	3759295	28.200	0.006089	0.013616	6.089126	13.615700	0.003784	0.0
4684	3759714	3759714	22.700	0.076658	0.171412	76.657557	171.411509	0.047633	0.1
4685	3759720	3759720	21.800	0.116026	0.259442	116.025908	259.441818	0.072095	0.1
4686	3772978	3772978	19.109	0.400641	0.895860	400.640618	895.859655	0.248946	0.5

10 rows × 40 columns

In [7]: #describe the data
data.describe(include='all')

Out[7]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)
count	4.687000e+03	4.687000e+03	4687.000000	4687.000000	4687.000000	4687.000000	4687.00
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [7]: `#describe the data
data.describe(include='all')`

Out[7]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est C M(
count	4.687000e+03	4.687000e+03	4687.000000	4687.000000	4687.000000	4687.000000	4687.00
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	3.272298e+06	3.272298e+06	22.267865	0.204604	0.457509	204.604203	457.50
std	5.486011e+05	5.486011e+05	2.890972	0.369573	0.826391	369.573402	826.39
min	2.000433e+06	2.000433e+06	11.160000	0.001011	0.002260	1.010543	2.25
25%	3.097594e+06	3.097594e+06	20.100000	0.033462	0.074824	33.462237	74.82
50%	3.514799e+06	3.514799e+06	21.900000	0.110804	0.247765	110.803882	247.76
75%	3.690060e+06	3.690060e+06	24.500000	0.253837	0.567597	253.837029	567.59
max	3.781897e+06	3.781897e+06	32.100000	15.579552	34.836938	15579.552410	34836.93

11 rows × 40 columns



In [8]: `#it will display information about that object, including its data type, attributes
#This information is provided interactively to help you understand the object better
data.info`

```
In [8]: #it will display information about that object, including its data type, attribut  
#This information is provided interactively to help you understand the object bet  
data.info
```

```
Out[8]: <bound method DataFrame.info of  
tude Est Dia in KM(min) \n  
0 3703080 3703080 21.600 0.127220  
1 3723955 3723955 21.300 0.146068  
2 2446862 2446862 20.300 0.231502  
3 3092506 3092506 27.400 0.008801  
4 3514799 3514799 21.600 0.127220  
... ... ... ... ...  
4682 3759007 3759007 23.900 0.044112  
4683 3759295 3759295 28.200 0.006089
```

	Neo	Reference ID	Name	Absolute Magnitude	\	
0	3703080	3703080	21.600	0.127220		
1	3723955	3723955	21.300	0.146068		
2	2446862	2446862	20.300	0.231502		
3	3092506	3092506	27.400	0.008801		
4	3514799	3514799	21.600	0.127220		
...		
4682	3759007	3759007	23.900	0.044112		
4683	3759295	3759295	28.200	0.006089		
4684	3759714	3759714	22.700	0.076658		
4685	3759720	3759720	21.800	0.116026		
4686	3772978	3772978	19.109	0.400641		
	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	\		
0	0.284472	127.219878	284.472297			
1	0.326618	146.067964	326.617897			
2	0.517654	231.502122	517.654482			
3	0.019681	8.801465	19.680675			
4	0.284472	127.219878	284.472297			
...		
4682	0.098637	44.111820	98.637028			
4683	0.013616	6.089126	13.615700			
4684	0.171412	76.657557	171.411509			
4685	0.259442	116.025908	259.441818			
4686	0.895860	400.640618	895.859655			
	Est Dia in Miles(min)	Est Dia in Miles(max)	Est Dia in Feet(min)	...		
\\						
0	0.079051	0.176763	417.388066	...		
1	0.090762	0.202951	479.225620	...		
2	0.143849	0.321655	759.521423	...		
3	0.005469	0.012229	28.876199	...		
4	0.079051	0.176763	417.388066	...		
...		
4682	0.027410	0.061290	144.723824	...		
4683	0.003784	0.008460	19.977449	...		
4684	0.047633	0.106510	251.501181	...		
4685	0.072095	0.161210	380.662441	...		
4686	0.248946	0.556661	1314.437764	...		
	Asc Node	Longitude	Orbital Period	Perihelion Distance	Perihelion Argument	\
0	314.373913	609.599786		0.808259	57.257470	
1	136.717242	425.869294		0.718200	313.091975	
2	259.475979	643.580228		0.950791	248.415038	
3	57.173266	514.082140		0.983902	18.707701	
4	84.629307	495.597821		0.967687	158.263596	
...	
4682	164.183305	457.179984		0.741558	276.395697	
4683	345.225230	407.185767		0.996434	42.111064	
4684	37.026468	690.054279		0.965760	274.692712	
4685	163.802909	662.048343		1.185467	180.346090	
4686	187.642183	653.679098		0.876110	222.436688	
	Aphelion Dist	Perihelion Time	Mean Anomaly	Mean Motion	Equinox	\
0	2.005764	2458161.642	264.837533	0.590551	J2000	
1	1.497352	2457794.969	173.741112	0.845330	J2000	
2	Hazardous	2458120.468	292.893654	0.559371	J2000	
3	False	2457902.337	68.741007	0.700277	J2000	
4	False	2457814.455	135.142133	0.726395	J2000	
2..	True	
4682	False	2457708.228	304.306024	0.787436	J2000	
4683	False	2458087.617	282.978786	0.884117	J2000	
4684	2.090708	2458300.480	203.501147	0.521698	J2000	
4685	False	2458288.261	203.524965	0.543767	J2000	
4686	False	2458318.587	184.820424	0.550729	J2000	
4684	False					

```

2      Hazard 1.966857    2458120.468    292.893654    0.559371    J2000
3      True   1.527904    2457902.337    68.741007    0.700277    J2000
4      False  1.03543    2457814.455    135.142133    0.726395    J2000
2..    True   ...        ...          ...          ...          ...
4682   False  1.581299    2457708.228    304.306024    0.787436    J2000
4683   True   1.53835    2458087.617    282.978786    0.884117    J2000
4684   2.090708  2458300.480    203.501147    0.521698    J2000
4685   False  1.587733    2458288.261    203.524965    0.543767    J2000
4686   False  1.51980    2458318.587    184.820424    0.550729    J2000
4684   False
4685   False
4686   False

```

[4687 rows x 40 columns]>

In [9]: *#it provides a concise summary of the DataFrame's structure, including the number of non-null values, data types of columns, and memory usage. #It's a useful method for quickly assessing the basic characteristics of your data.*

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4687 entries, 0 to 4686
Data columns (total 40 columns):
 #   Column          Non-Null Count  Dtype  

```

In [9]: *#it provides a concise summary of the DataFrame's structure, including the number of non-null values, data types of columns, and memory usage #It's a useful method for quickly assessing the basic characteristics of your data*

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4687 entries, 0 to 4686
Data columns (total 40 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0   Neo Reference ID    4687 non-null   int64  
 1   Name              4687 non-null   int64  
 2   Absolute Magnitude 4687 non-null   float64 
 3   Est Dia in KM(min) 4687 non-null   float64 
 4   Est Dia in KM(max) 4687 non-null   float64 
 5   Est Dia in M(min)  4687 non-null   float64 
 6   Est Dia in M(max)  4687 non-null   float64 
 7   Est Dia in Miles(min) 4687 non-null   float64 
 8   Est Dia in Miles(max) 4687 non-null   float64 
 9   Est Dia in Feet(min) 4687 non-null   float64 
 10  Est Dia in Feet(max) 4687 non-null   float64 
 11  Close Approach Date 4687 non-null   object  
 12  Epoch Date Close Approach 4687 non-null   float64 
 13  Relative Velocity km per sec 4687 non-null   float64 
 14  Relative Velocity km per hr  4687 non-null   float64 
 15  Miles per hour          4687 non-null   float64 
 16  Miss Dist.(Astronomical) 4687 non-null   float64 
 17  Miss Dist.(lunar)        4687 non-null   float64 
 18  Miss Dist.(kilometers)  4687 non-null   float64 
 19  Miss Dist.(miles)       4687 non-null   float64 
 20  Orbiting Body          4687 non-null   object  
 21  Orbit ID               4677 non-null   float64 
 22  Orbit Determination Date 4609 non-null   object  
 23  Orbit Uncertainty      4442 non-null   float64 
 24  Minimum Orbit Intersection 4371 non-null   float64 
 25  Jupiter Tisserand Invariant 4323 non-null   float64 
 26  Epoch Osculation       4211 non-null   float64 
 27  Eccentricity          4123 non-null   float64 
 28  Semi Major Axis        4061 non-null   float64 
 29  Inclination           4034 non-null   float64 
 30  Asc Node Longitude    4197 non-null   float64 
 31  Orbital Period         3784 non-null   float64 
 32  Perihelion Distance   3822 non-null   float64 
 33  Perihelion Arg         3836 non-null   float64 
 34  Aphelion Dist          4193 non-null   float64 
 35  Perihelion Time        4390 non-null   float64 
 36  Mean Anomaly           4534 non-null   float64 
 37  Mean Motion            4577 non-null   float64 
 38  Equinox                4642 non-null   object  
 39  Hazardous             4687 non-null   bool    
dtypes: bool(1), float64(33), int64(2), object(4)
memory usage: 1.4+ MB
```

In [10]: *#any null value present*
data.isnull().values.any()

Out[10]: True
In [11]: *# check the null values entirely present in the dataset*
data.isnull()

Out[11]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	Est Dia in Miles(min)	Est Dia in Miles(max)	Est Dia in Feet(min)	Est Dia in Feet(max)
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False

Out[10]: True
In [11]: # check the null values entirely present in the dataset
`data.isnull()`

Out[11]:

	Neo Reference ID	Name	Absolute Magnitude	Est Dia in KM(min)	Est Dia in KM(max)	Est Dia in M(min)	Est Dia in M(max)	Est Dia in Miles(min)	Est Dia in Miles(max)	Fe
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...
4682	False	False	False	False	False	False	False	False	False	False
4683	False	False	False	False	False	False	False	False	False	False
4684	False	False	False	False	False	False	False	False	False	False
4685	False	False	False	False	False	False	False	False	False	False
4686	False	False	False	False	False	False	False	False	False	False

4687 rows × 40 columns



In [12]: # number of null values in each column
`data.isnull().sum()`

Out[12]: Neo Reference ID 0
Name 0
Absolute Magnitude 0
Est Dia in KM(min) 0
Est Dia in KM(max) 0
Est Dia in M(min) 0
Est Dia in M(max) 0

```
In [12]: # number of null values in each column
data.isnull().sum()
```

```
Out[12]: Neo Reference ID          0
Name                          0
Absolute Magnitude           0
Est Dia in KM(min)           0
Est Dia in KM(max)           0
Est Dia in M(min)            0
Est Dia in M(max)             0
Est Dia in Miles(min)         0
Est Dia in Miles(max)         0
Est Dia in Feet(min)          0
Est Dia in Feet(max)          0
Close Approach Date          0
Epoch Date Close Approach     0
Relative Velocity km per sec   0
Relative Velocity km per hr    0
Miles per hour                 0
Miss Dist.(Astronomical)      0
Miss Dist.(lunar)              0
Miss Dist.(kilometers)         0
Miss Dist.(miles)              0
Orbiting Body                  0
Orbit ID                      10
Orbit Determination Date       78
Orbit Uncertainty              245
Minimum Orbit Intersection       316
Jupiter Tisserand Invariant     364
Epoch Osculation                476
Eccentricity                   564
Semi Major Axis                 626
Inclination                     653
Asc Node Longitude               490
Orbital Period                   903
Perihelion Distance              865
Perihelion Arg                   851
Aphelion Dist                     494
Perihelion Time                   297
Mean Anomaly                      153
Mean Motion                       110
Equinox                           45
Hazardous                         0
dtype: int64
```

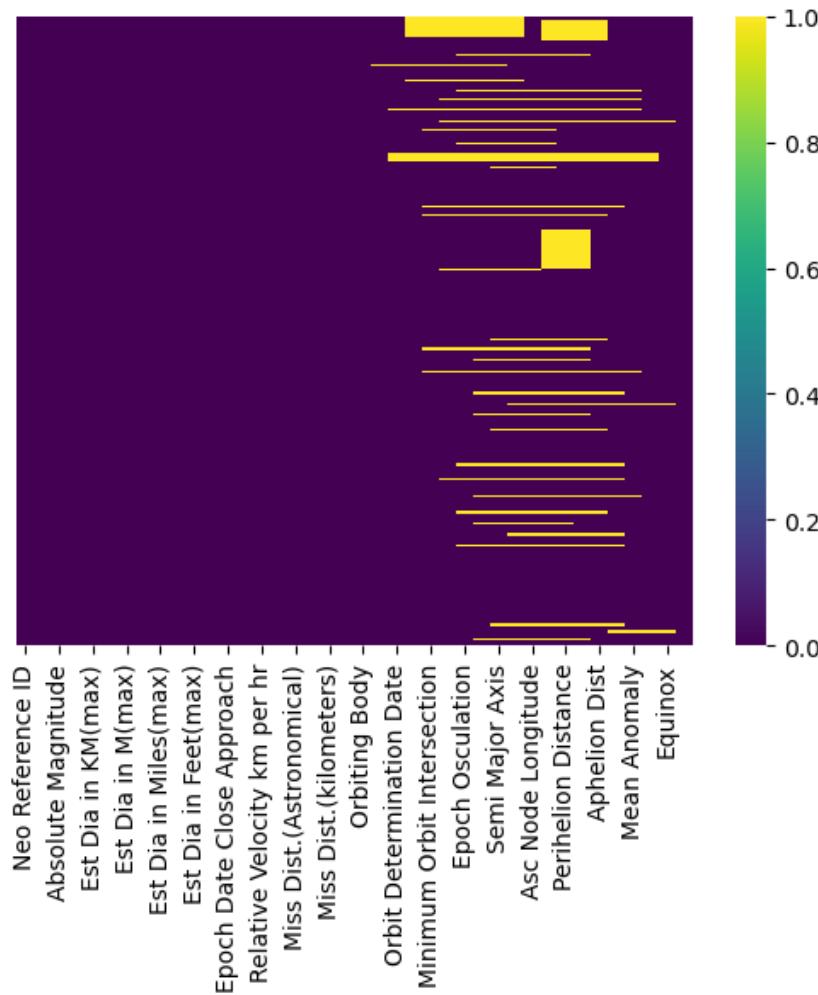
```
In [13]: # to illustrate null values using heatmap
sns.heatmap(data.isnull(), yticklabels=False, cmap="viridis")
```

```
Out[13]: <Axes: >
```



```
In [13]: # to illustrate null values using heatmap
sns.heatmap(data.isnull(), yticklabels=False, cmap="viridis")
```

Out[13]: <Axes: >



Filling missing values

```
In [14]: # there is no need to fill the some data because there is no value for me so drop
data.drop(["Miss Dist.(Astronomical)", "Miss Dist.(lunar)", "Miss Dist.(miles)",
           "Relative Velocity km per sec", "Est Dia in M(max)",
           "Relative Velocity km per hr", "Est Dia in Feet(max)",
           "Est Dia in Feet(min)", "Est Dia in Miles(max)",
           "Est Dia in Miles(min)", "Est Dia in KM(max)", "Est Dia in KM(min)",
           "Neo Reference ID", "Orbit ID", "Name", "Close Approach Date",
           "Equinox", "Epoch Date Close Approach", "Orbiting Body", "Orbit Determination Date"], axis=1)
```

```
In [15]: data.isnull().sum()
```

```
Out[15]: Absolute Magnitude          0
Est Dia in M(min)                 0
Miles per hour                   0
Miss Dist.(kilometers)            0
Orbit Uncertainty                  245
Minimum Orbit Intersection        316
Jupiter Tisserand Invariant      364
Epoch Osculation                  176
```

In [15]: `data.isnull().sum()`

```
Out[15]: Absolute Magnitude          0
Est Dia in M(min)                 0
Miles per hour                   0
Miss Dist.(kilometers)           0
Orbit Uncertainty                245
Minimum Orbit Intersection        316
Jupiter Tisserand Invariant     364
Epoch Osculation                 476
Eccentricity                     564
Semi Major Axis                  626
Inclination                      653
Asc Node Longitude               490
Orbital Period                   903
Perihelion Distance              865
Perihelion Arg                  851
Aphelion Dist                   494
Perihelion Time                  297
Mean Anomaly                      153
Mean Motion                      110
Hazardous                         0
dtype: int64
```

In [16]: `#fill my important data`

```
data['Orbit Uncertainty']=data['Orbit Uncertainty'].fillna(data['Orbit Uncertainty'].mean())
data['Minimum Orbit Intersection']=data['Minimum Orbit Intersection'].fillna(data['Minimum Orbit Intersection'].mean())
data['Jupiter Tisserand Invariant']=data['Jupiter Tisserand Invariant'].fillna(data['Jupiter Tisserand Invariant'].mean())
data['Epoch Osculation']=data['Epoch Osculation'].fillna(data['Epoch Osculation'].mean())
data['Eccentricity']=data["Eccentricity"].fillna(data["Eccentricity"].mean(),inplace=True)
data['Semi Major Axis']=data['Semi Major Axis'].fillna(data['Semi Major Axis'].mean())
data['Inclination']=data['Inclination'].fillna(data['Inclination'].mean())
data['Asc Node Longitude']=data['Asc Node Longitude'].fillna(data['Asc Node Longitude'].mean())
```

In [17]: `data['Orbital Period']=data['Orbital Period'].fillna(data['Orbital Period'].mean())
data['Perihelion Distance']=data['Perihelion Distance'].fillna(data['Perihelion Distance'].mean())
data['Perihelion Arg']=data['Perihelion Arg'].fillna(data['Perihelion Arg'].mean())
data['Aphelion Dist']=data['Aphelion Dist'].fillna(data['Aphelion Dist'].mean())
data['Perihelion Time']=data['Perihelion Time'].fillna(data['Perihelion Time'].mean())
data['Mean Anomaly']=data['Mean Anomaly'].fillna(data['Mean Anomaly'].mean())
data['Mean Motion']=data['Mean Motion'].fillna(data['Mean Motion'].mean())`

In [18]: `data.isnull().sum()`

```
Out[18]: Absolute Magnitude          0
Est Dia in M(min)                 0
Miles per hour                   0
Miss Dist.(kilometers)           0
Orbit Uncertainty                0
Minimum Orbit Intersection        0
Jupiter Tisserand Invariant     0
Epoch Osculation                 0
Eccentricity                     0
Semi Major Axis                  0
Inclination                      0
Asc Node Longitude               0
Orbital Period                   0
Perihelion Distance              0
Perihelion Arg                  0
Aphelion Dist                   0
```

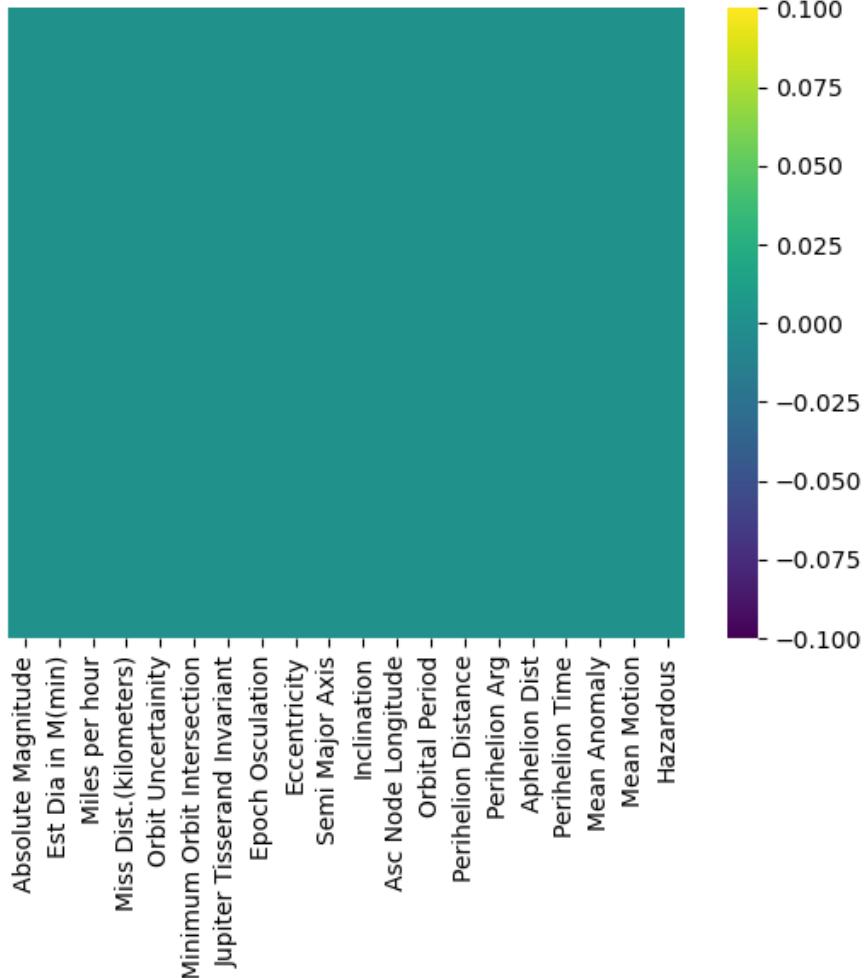
In [19]: `#to illustrate null values using heatmap`

```
sns.heatmap(data.isnull(), yticklabels=False,cmap='viridis')
```

```
Out[19]: Perihelion Time             0
<Axes>
Mean Anomaly                      0
```



```
In [19]: #To illustrate null values using heatmap
perihelionDistance = data.isnull(), yticklabels=False,cmap='viridis')
Aphelion Dist      0
Perihelion Time    0
Mean Anomaly       0
```



Data Visualization

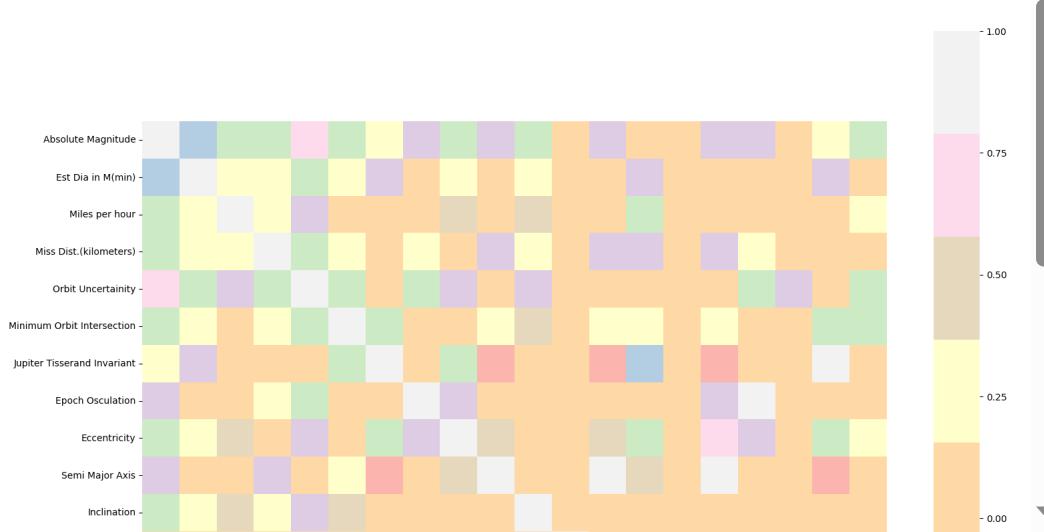
```
In [23]: #Examining a correlation matrix of all the features
corrmat = data.corr()
plt.subplots(figsize=(18,18))
sns.heatmap(corrmat,cmap="Pastel1", square=True)
```

Out[23]: <Axes: >

In [23]: #Examining a correlation matrix of all the features

```
corrrmat = data.corr()
plt.subplots(figsize=(18,18))
sns.heatmap(corrrmat,cmap="Pastel1", square=True)
```

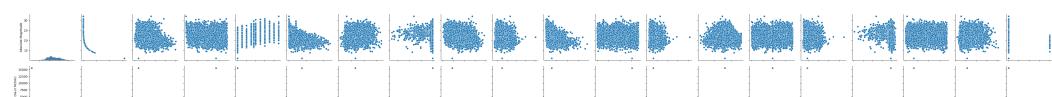
Out[23]: <Axes: >



In [24]: sns.pairplot(data)

<__array_function__ internals>:200: RuntimeWarning: Converting input from bool to <class 'numpy.uint8'> for compatibility.

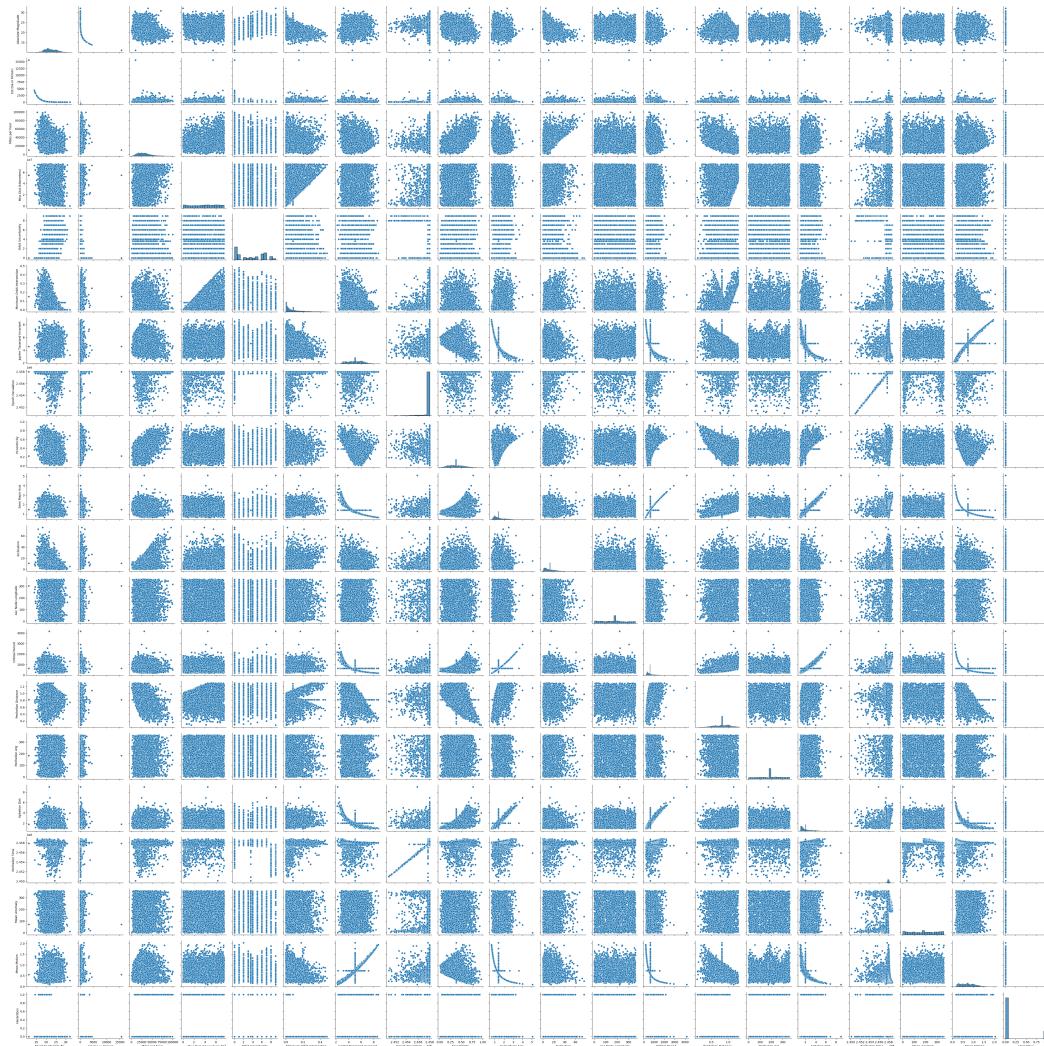
Out[24]: <seaborn.axisgrid.PairGrid at 0x29b51828710>



In [24]: `sns.pairplot(data)`

```
<__array_function__ internals>:200: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
```

Out[24]: <seaborn.axisgrid.PairGrid at 0x29b51828710>

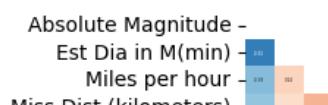


In [25]: `corr=num_data.corr()`

```
msk = np.triu(np.ones_like(corr))
```

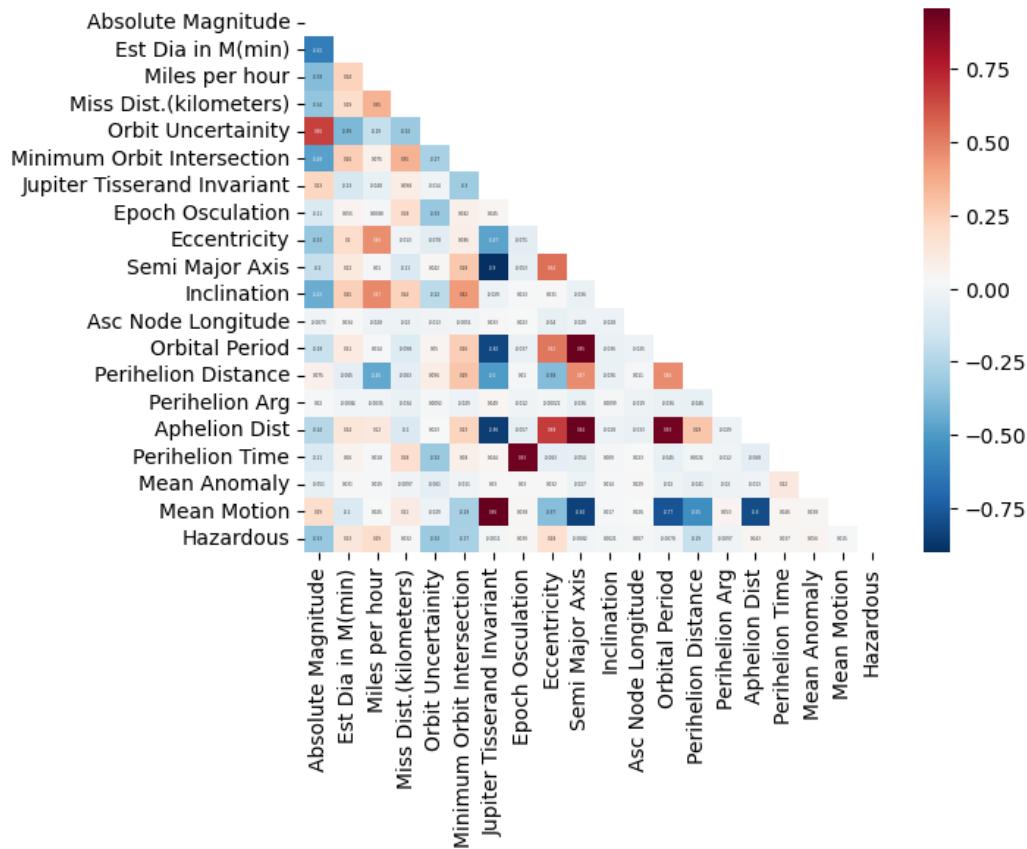
```
sns.heatmap(corr,cmap=plt.cm.RdBu_r,annot=True,annot_kws={'size':2},mask=msk)
```

Out[25]: <Axes: >

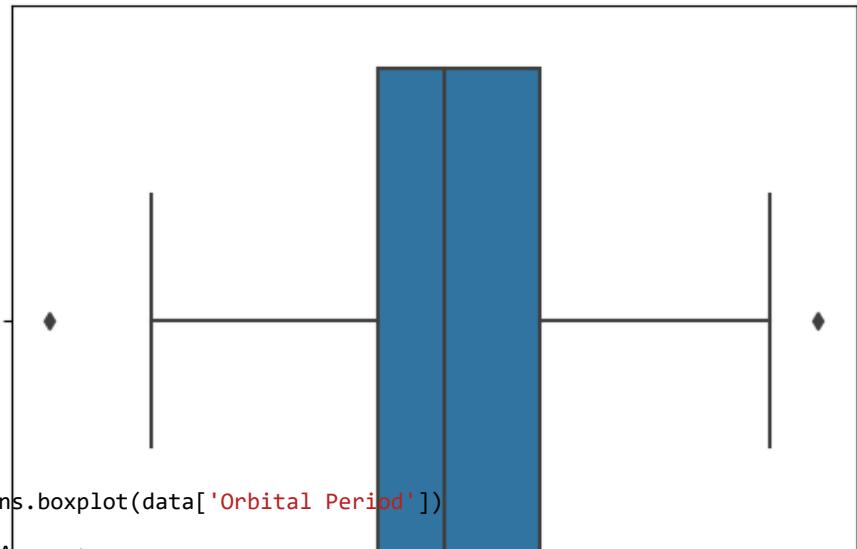


```
In [25]: corr=num_data.corr()
msk = np.triu(np.ones_like(corr))
sns.heatmap(corr,cmap=plt.cm.RdBu_r,annot=True,annot_kws={'size':2},mask=msk)
```

Out[25]: <Axes: >



```
In [26]: for i in num_data.columns:
    sns.boxplot(x=data[i])
    plt.show()
```



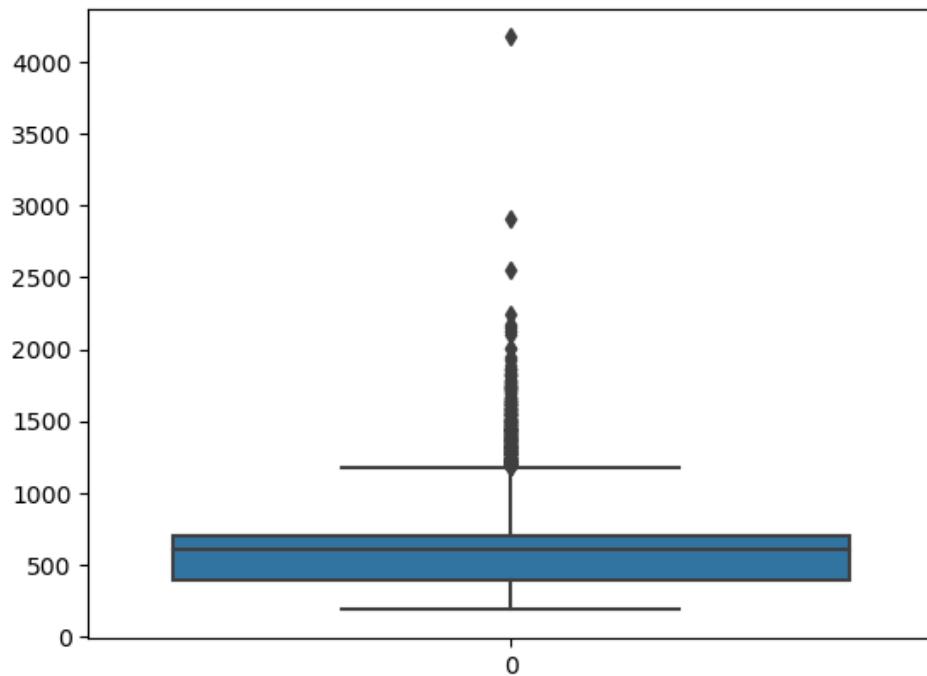
In [27]: sns.boxplot(data['Orbital Period'])

Out[27]: <Axes: >

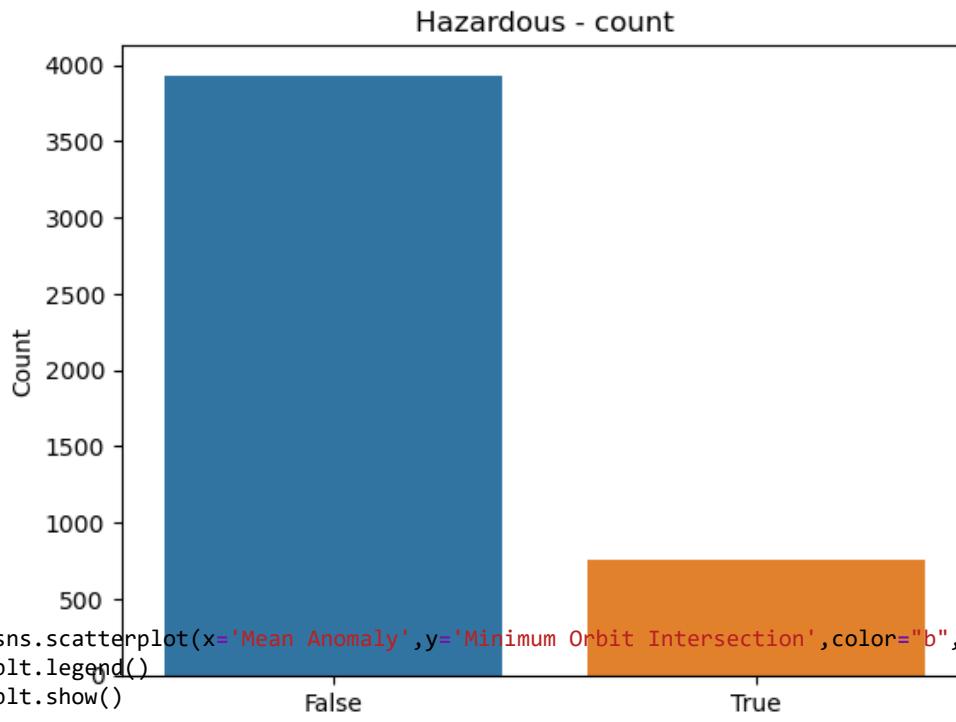


```
In [27]: sns.boxplot(data['Orbital Period'])
```

```
Out[27]: <Axes: >
```



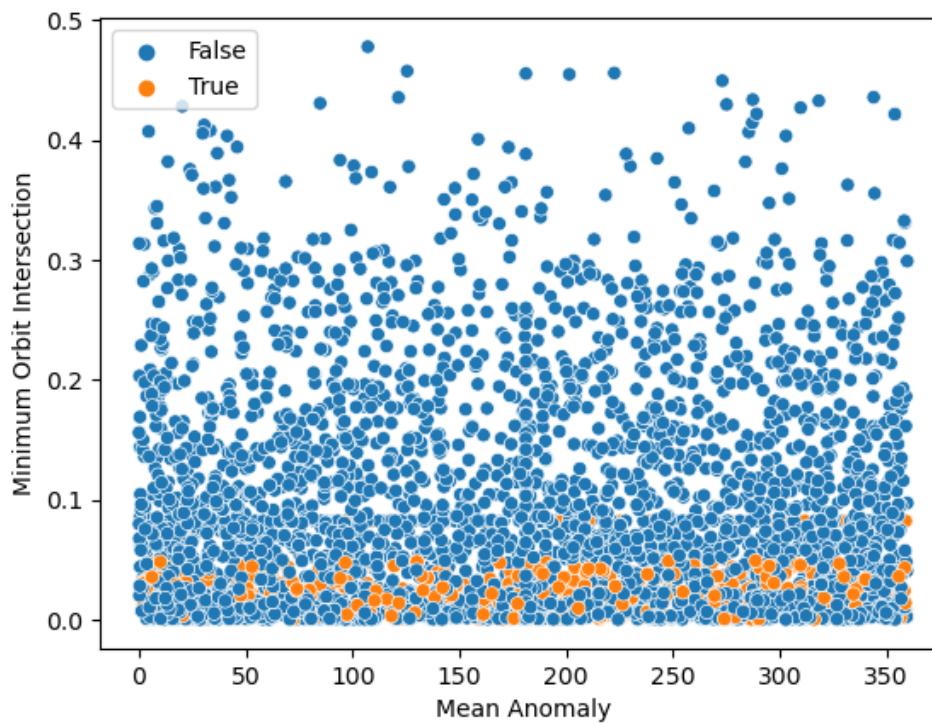
```
In [30]: a=data.groupby("Hazardous")["Hazardous"].count()
sns.barplot(x=a.index,y=a.values)
plt.title("Hazardous - count")
plt.xlabel("Hazardous")
plt.ylabel("Count")
plt.show()
```



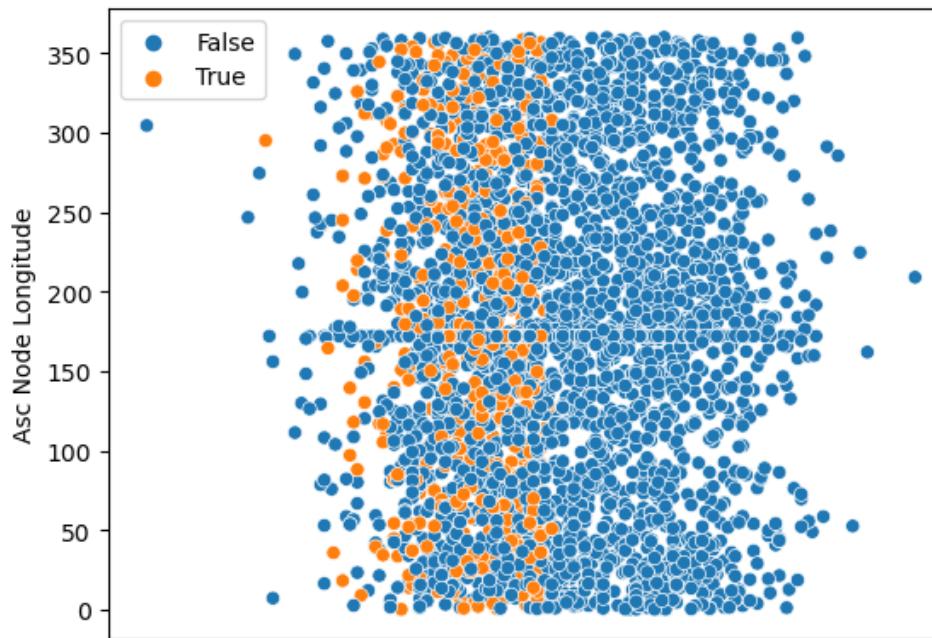
```
In [39]: sns.scatterplot(x='Mean Anomaly',y='Minimum Orbit Intersection',color="b",hue="Ha
```



```
In [39]: sns.scatterplot(x='Mean Anomaly',y='Minimum Orbit Intersection',color="b",hue="Hazardous")
plt.legend()
plt.show()
```



```
In [44]: sns.scatterplot(x='Absolute Magnitude',y='Asc Node Longitude',color="b",hue="Hazardous")
plt.legend()
plt.show()
```



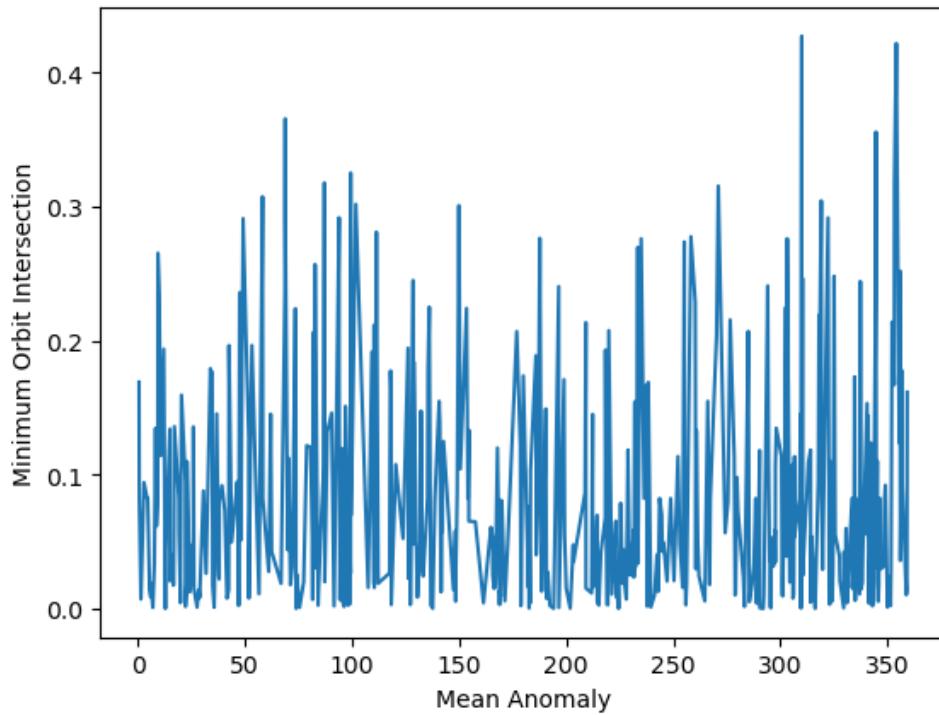
```
In [53]: x=data.sample(500)
sns.lineplot(x='Mean Anomaly',y='Minimum Orbit Intersection',data=x)
```

```
Out[53]: <Axes: xlabel='Mean Anomaly', ylabel='Minimum Orbit Intersection'>
```

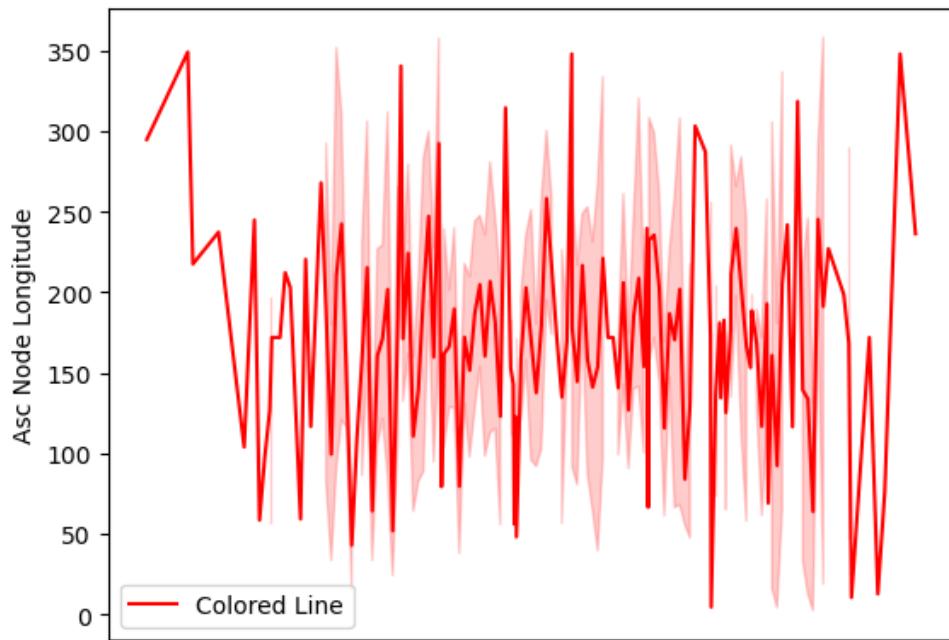


```
In [53]: x = data.sample(500)
sns.lineplot(x='Mean Anomaly', y="Minimum Orbit Intersection", data=x)
```

Out[53]: <Axes: xlabel='Mean Anomaly', ylabel='Minimum Orbit Intersection'>



```
In [55]: sns.lineplot(x='Absolute Magnitude', y="Asc Node Longitude", data=x, color='red', label='Line')
plt.show()
```

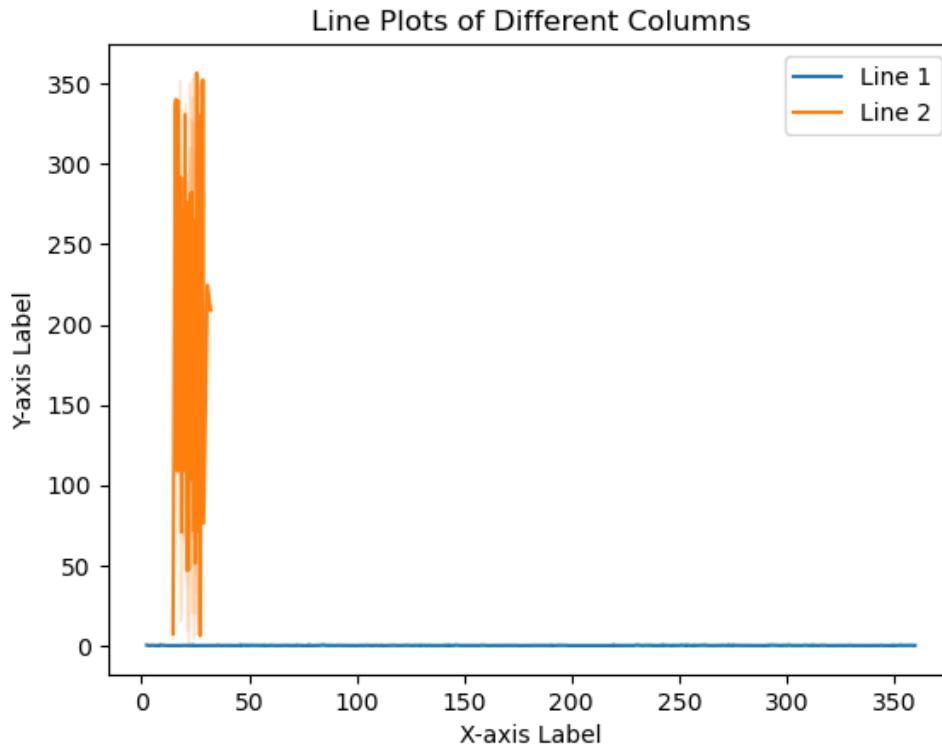


```
In [52]: x = data.sample(500)
sns.lineplot(x='Mean Anomaly', y="Minimum Orbit Intersection", data=x, label='Line')
sns.lineplot(x='Absolute Magnitude', y="Asc Node Longitude", data=x, label='Line')

plt.xlabel('X-axis Label')
plt.ylabel('Y-axis Label')
plt.title('Line Plots of Different Columns')
plt.legend()
plt.show()
```

```
In [52]: x = data.sample(500)
sns.lineplot(x='Mean Anomaly', y="Absolute Magnitude", data=x, label='Line 1')
sns.lineplot(x='Absolute Magnitude', y="Asc Node Longitude", data=x, label='Line 2')

plt.xlabel('X-axis Label')
plt.ylabel('Y-axis Label')
plt.title('Line Plots of Different Columns')
plt.legend()
plt.show()
```



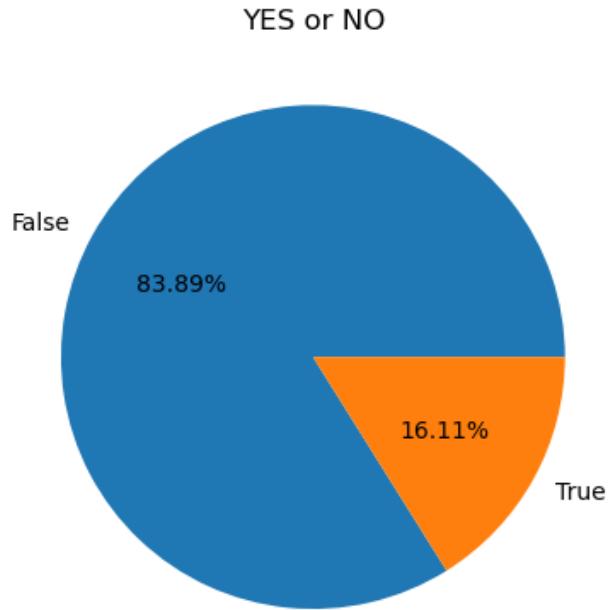
```
In [57]: b=data.groupby("Hazardous")["Hazardous"].count()
plt.pie(b,labels=b.index,autopct=".2f%%")
plt.title("YES or NO")
```

Out[57]: Text(0.5, 1.0, 'YES or NO')

YES or NO

```
In [57]: b=data.groupby("Hazardous")["Hazardous"].count()
plt.pie(b,labels=b.index,autopct=".2f%%")
plt.title("YES or NO")
```

Out[57]: Text(0.5, 1.0, 'YES or NO')



```
In [62]: x = data.sample(1000)
sns.scatterplot(x='Est Dia in M(min)', y='Miss Dist.(kilometers)', data=x)
plt.xlabel('Est Dia in M(min)')
plt.ylabel('Miss Dist.(kilometers)')
plt.title('Est Dia in M(min) vs. Miss Dist.(kilometers)')
plt.show()
```

