CLR204 Assessment 3 Report

David Lawler

Target for Prediction

The target for prediction is the diameter (km) of an asteroid.

The Dataset

Taking a look at the dataset, we can see that there are 16427 asteroids and 23 features in the dataset. We can see data cleaning has been handled and data types of the variables using the info function.

link: https://github.com/blakelobato/Predicting-AsteroidDiameter-Dash/blob/master/model/Pred_Ast_Diam_2.csv

Head Function

```
df.head(25)
```

	orbit_id	e	a	i	om	w	ma	n	tp	moid	...	data_arc	n_obs_used	rms	diameter	albedo	diameter_sigma	first_year_obs	first_month_obs	last_obs_year	last_obs_month
0	JPL 35	0.242027	2.201791	2.536221	313.311389	18.989048	301.072249	0.301675	2.458796e+06	0.657747	...	46399.0	2611	0.46222	9.300	0.2082	0.800	1892	10	2019	10
1	JPL 25	0.256856	2.338209	22.326589	10.489602	105.115594	87.454449	0.275663	2.458283e+06	0.875501	...	38117.0	1528	0.38116	9.822	0.3140	0.130	1915	4	2019	8
2	JPL 28	0.160543	2.228812	1.747387	121.579382	252.465454	208.942016	0.296206	2.459110e+06	0.871683	...	36040.0	2357	0.44671	8.196	0.3790	0.100	1920	9	2019	5
3	JPL 35	0.167945	2.241299	2.428619	161.636895	172.846491	20.350289	0.293734	2.458531e+06	0.854020	...	33289.0	2574	0.43691	6.534	0.2170	0.068	1928	10	2019	11
4	JPL 34	0.253295	2.467536	6.757106	137.130656	259.158793	127.366908	0.254278	2.458100e+06	0.862972	...	39907.0	2523	0.44695	9.111	0.2560	0.303	1910	2	2019	5
5	JPL 67	0.073742	1.944104	22.508840	175.320955	124.031963	224.445860	0.363601	2.458973e+06	0.833343	...	44070.0	2492	0.37174	8.934	0.7260	0.748	1898	9	2019	5
6	JPL 34	0.103066	2.244712	5.995089	203.399440	264.392525	244.456912	0.293064	2.458995e+06	1.040690	...	40578.0	2484	0.43574	9.326	0.2210	0.239	1908	10	2019	11
7	JPL 29	0.110058	2.230630	5.389819	72.373045	354.339267	127.022318	0.295844	2.458171e+06	0.998781	...	41871.0	2440	0.39707	9.010	0.2870	0.099	1904	12	2019	8
8	JPL 29	0.239092	2.253449	5.680974	336.764958	76.779174	181.329773	0.291362	2.459214e+06	0.736039	...	41608.0	1810	0.40250	8.456	0.2980	0.031	1905	9	2019	8
9	JPL 33	0.353074	2.628043	32.583941	155.112383	76.773043	169.922720	0.231342	2.457866e+06	0.923279	...	41518.0	1834	0.40240	9.230	0.3255	0.900	1906	3	2019	11
10	JPL 36	0.127957	2.220819	1.710039	41.005619	18.012785	213.767825	0.297807	2.459092e+06	0.948648	...	40958.0	2423	0.43430	9.446	0.2170	0.166	1907	9	2019	10
11	JPL 27	0.213370	2.324423	6.902487	302.807248	38.110992	184.455090	0.278120	2.459232e+06	0.824229	...	40785.0	2581	0.41209	8.153	0.2690	0.151	1907	9	2019	5
12	JPL 27	0.194725	2.441702	7.331119	254.679340	175.697462	32.854176	0.258324	2.458473e+06	0.980618	...	40786.0	2534	0.41882	9.725	0.5080	0.092	1907	9	2019	5
13	JPL 34	0.278983	2.545744	12.715483	357.019751	349.524955	156.468717	0.242651	2.457956e+06	0.829815	...	40783.0	2192	0.42847	7.726	0.3450	0.077	1907	9	2019	5
14	JPL 31	0.137261	2.174837	2.458112	213.756930	174.868642	291.496345	0.307301	2.458823e+06	0.880428	...	39864.0	2760	0.43224	7.250	0.3680	0.273	1910	10	2019	11
15	JPL 34	0.107869	2.180337	4.273327	281.903596	90.788273	301.127993	0.306139	2.458793e+06	0.950041	...	39787.0	2719	0.41832	9.003	0.3150	0.283	1910	12	2019	11

Shape Function

```
[6]  df.shape

     (16427, 23)
```

Feature List

```
feature_list = df.columns.values.tolist()
print (feature_list)
```

```
['orbit_id', 'e', 'a', 'i', 'om', 'w', 'ma', 'n', 'tp', 'moid', 'moid_jup', 'class',
'producer', 'data_arc', 'n_obs_used', 'rms', 'diameter', 'albedo', 'diameter_sigma', 'first_year_obs', 'first_month_obs', 'last_obs_year', 'last_obs_month']
```

Info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16427 entries, 0 to 16426
Data columns (total 23 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   orbit_id         16427 non-null  object
 1   e                16427 non-null  float64
 2   a                16427 non-null  float64
 3   i                16427 non-null  float64
 4   om               16427 non-null  float64
 5   w                16427 non-null  float64
 6   ma               16427 non-null  float64
 7   n                16426 non-null  float64
 8   tp               16426 non-null  float64
 9   moid             16426 non-null  float64
 10  moid_jup         16426 non-null  float64
 11  class            16426 non-null  object
 12  producer         16426 non-null  object
 13  data_arc         16426 non-null  float64
 14  n_obs_used       16426 non-null  float64
 15  rms              16426 non-null  float64
 16  diameter         16426 non-null  float64
 17  albedo           16426 non-null  float64
 18  diameter_sigma   16426 non-null  float64
 19  first_year_obs   16426 non-null  float64
 20  first_month_obs  16426 non-null  float64
 21  last_obs_year    16426 non-null  float64
 22  last_obs_month   16426 non-null  float64
dtypes: float64(20), object(3)
memory usage: 2.9+ MB
```

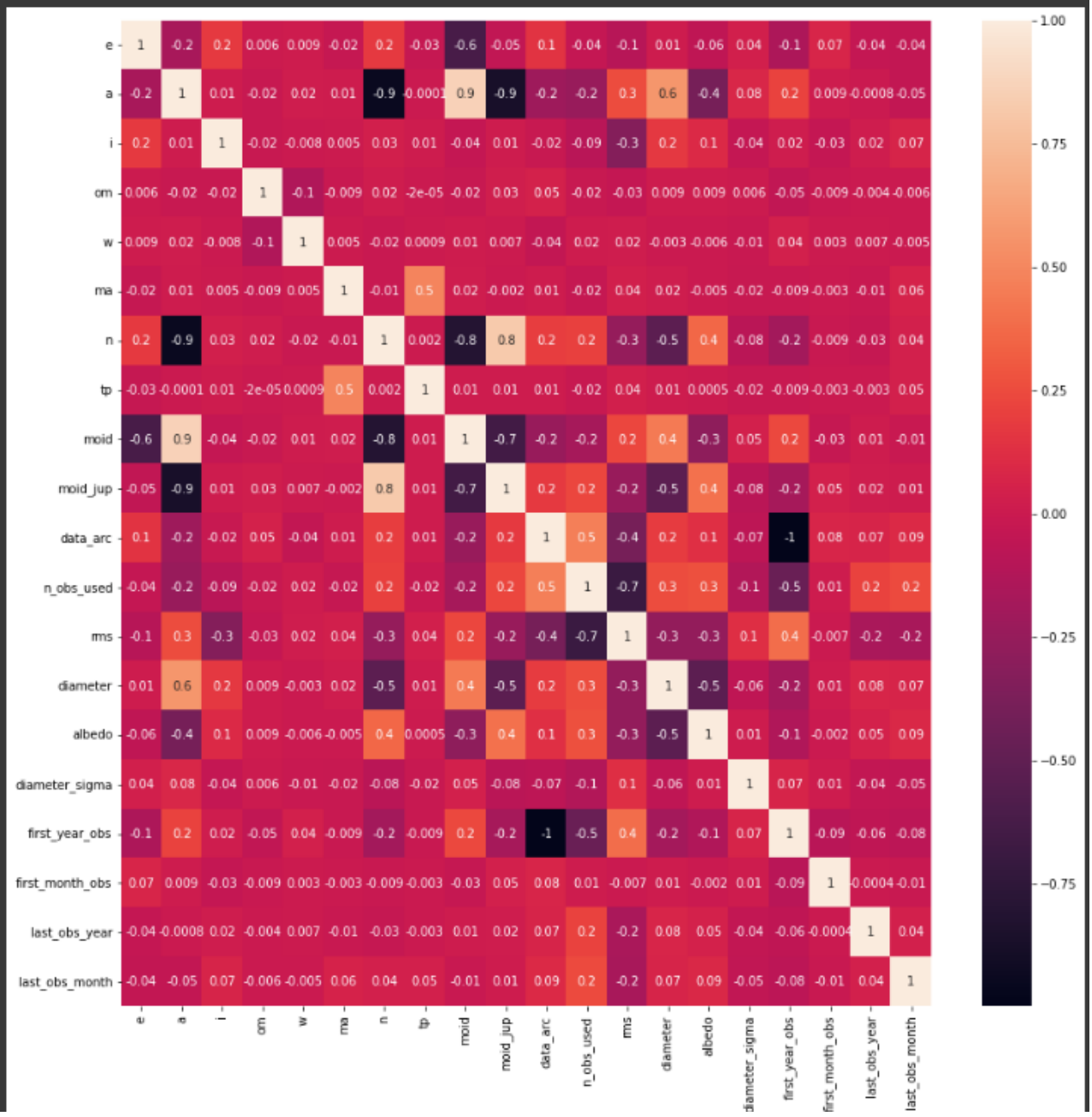
Correlation Map

Highly correlated features are:

- moid & a: 0.9
- moid_jup & n: 0.8
- diameter & a: 0.6

```
# Figure 2. Heat map of Pearson Correlation between data features
import seaborn as sb
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(15,15))         # Sample figsize in inches
dataplot=sb.heatmap(df.corr(),annot=True, fmt='.01g', ax=ax)
```



Since we are looking to predict diameter, we should look for other features th at have positive correlation to diameter.

```
# Since we are looking to predict diameter, we should loo
corr_matrix = df.corr()
corr_matrix['diameter']. sort_values(ascending = False)
```

```
diameter           1.000000
a                  0.566838
moid               0.441137
n_obs_used         0.256112
data_arc           0.186074
i                  0.170802
last_obs_year      0.076029
last_obs_month     0.070366
ma                 0.019964
first_month_obs    0.014316
e                  0.012918
tp                 0.010162
om                 0.008666
w                 -0.002656
diameter_sigma    -0.057958
first_year_obs    -0.184392
rms               -0.260019
albedo            -0.518144
n                 -0.519039
moid_jup          -0.522444
Name: diameter, dtype: float64
```

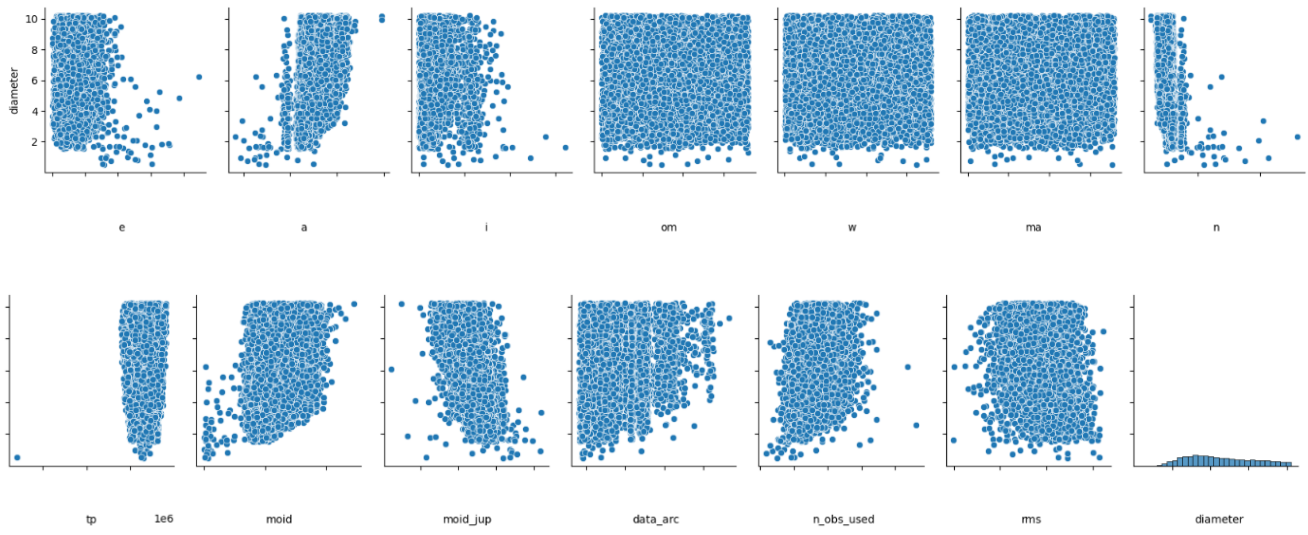
So for our scatterplots I think we should pay particular attention to a, moid, n_obs_used and data_arc in respect to diameter.

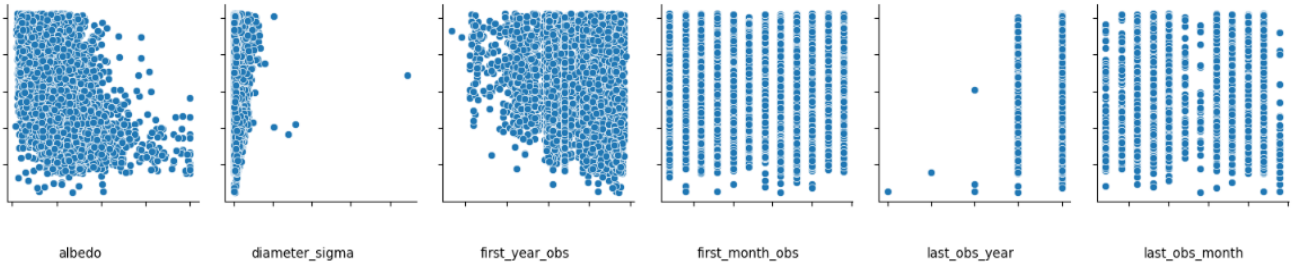
Semi major axis (a) - One half of the major axis of the elliptical orbit; also the mean distance from the Sun.

Minimum orbit intersection distance (moid) - The distance between the closest points of the osculating orbits of two bodies.

Pair Plots for Bivariate Analysis

Every feature pair has been plotted using seaborn pair plot in the EDA. Here is the diameter pairs, where diameter is the y axis.





Model Implementation

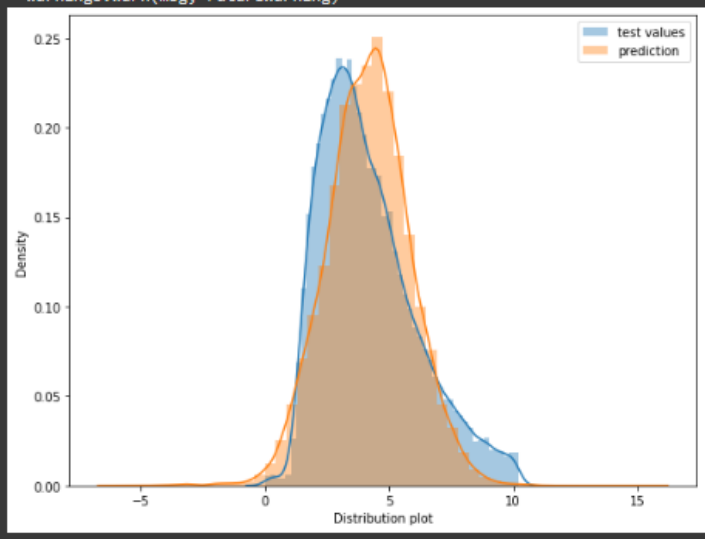
Models	Performance or Evaluation Metrics R2 Score	Evaluation Comment (e.g., good, bad or best)
Logistic Regression	0.676	OK
Decision Tree	0.771	Better
Random Forest	0.883	Best

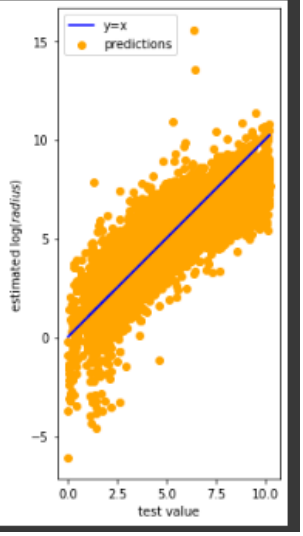
Linear Regression

```
The R^2 score achieved using this regression is: 0.676

[68] plot(Y_pred_lr)

/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be
  warnings.warn(msg, FutureWarning)
```



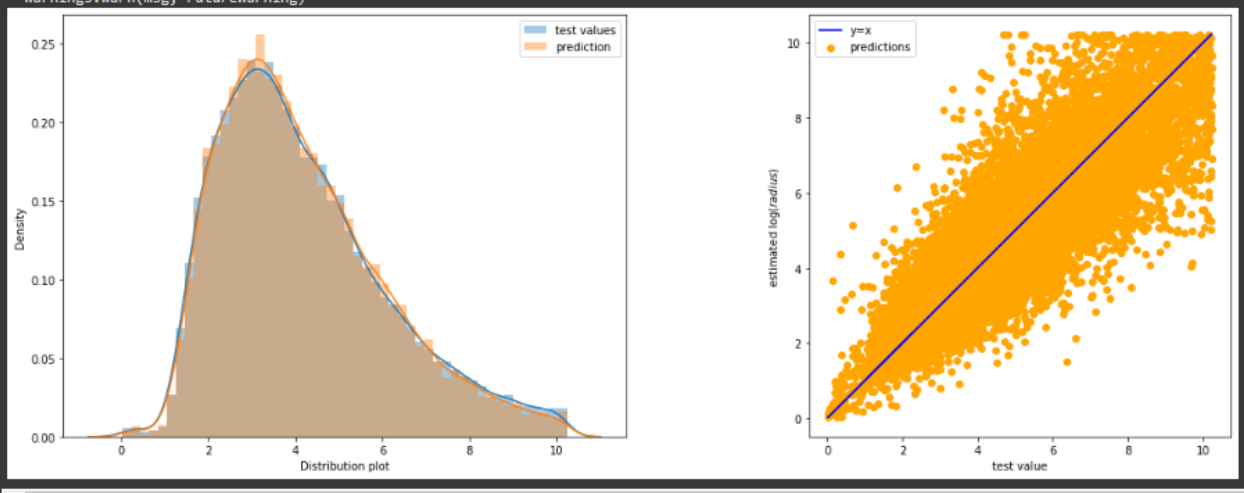


Decision Tree

```
The R^2 score achieved using this regression is: 0.771
```

```
[70] plot(Y_pred_tree)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a f
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a f
  warnings.warn(msg, FutureWarning)
```

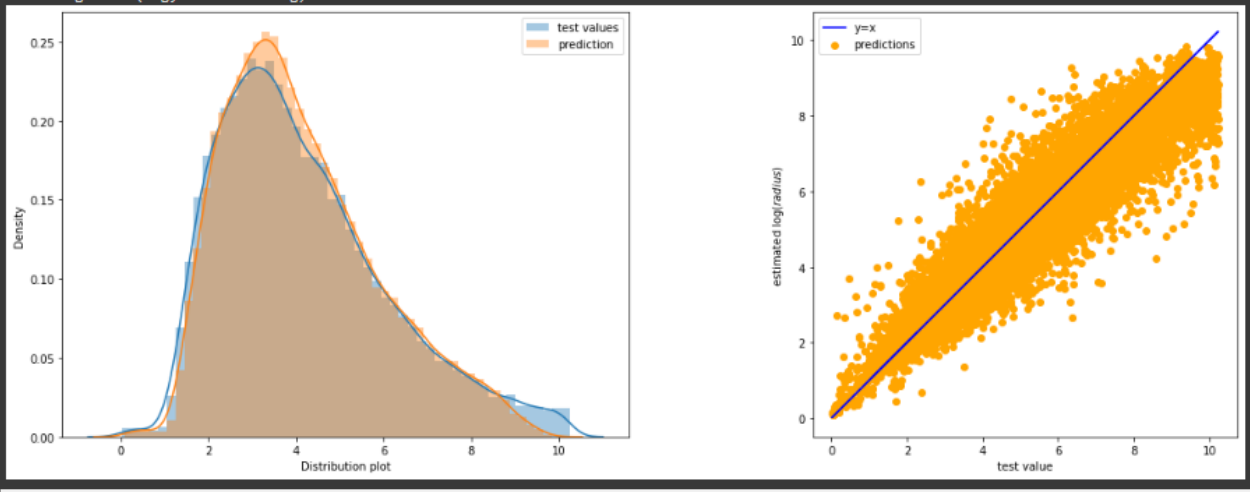


Random Forest

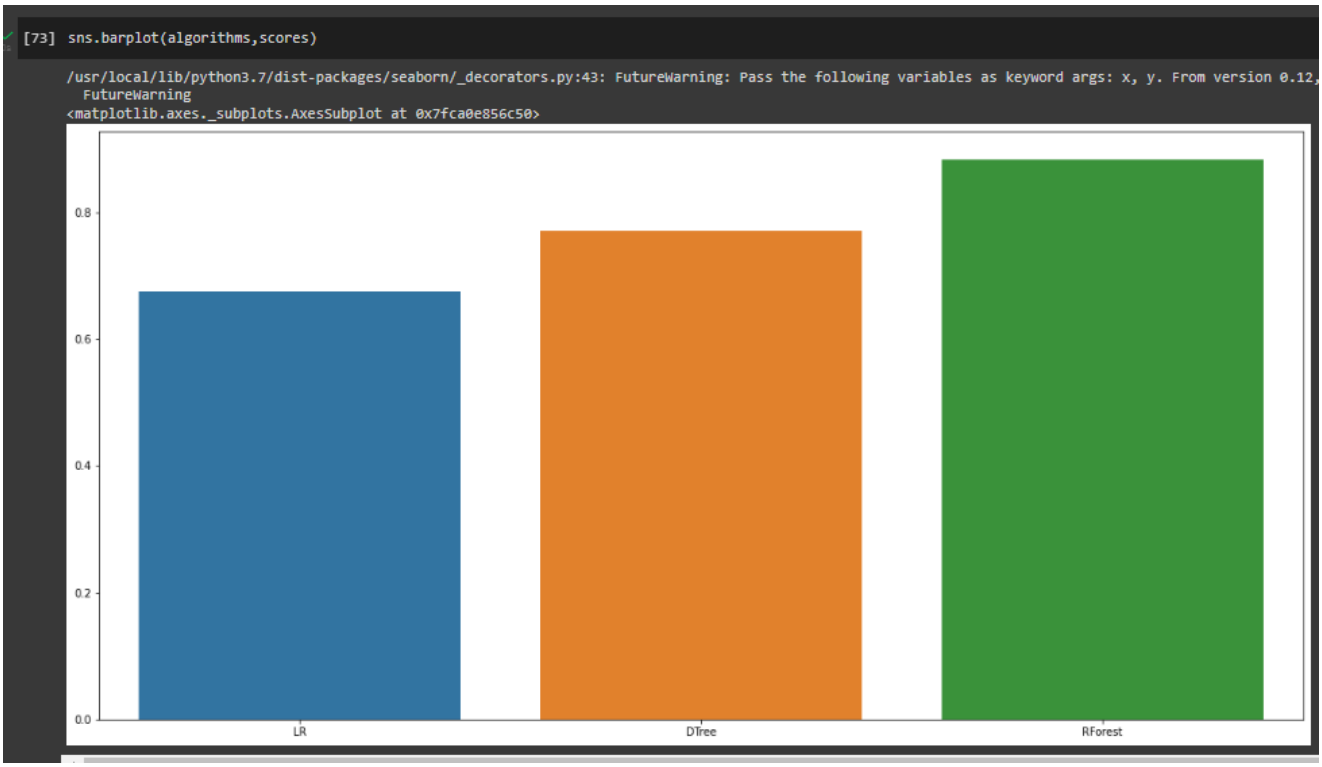
```
The R^2 score achieved using this regression is: 0.883
```

```
[72] plot(Y_pred_forest)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a f
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a f
  warnings.warn(msg, FutureWarning)
```



The Best Model is Random Forest

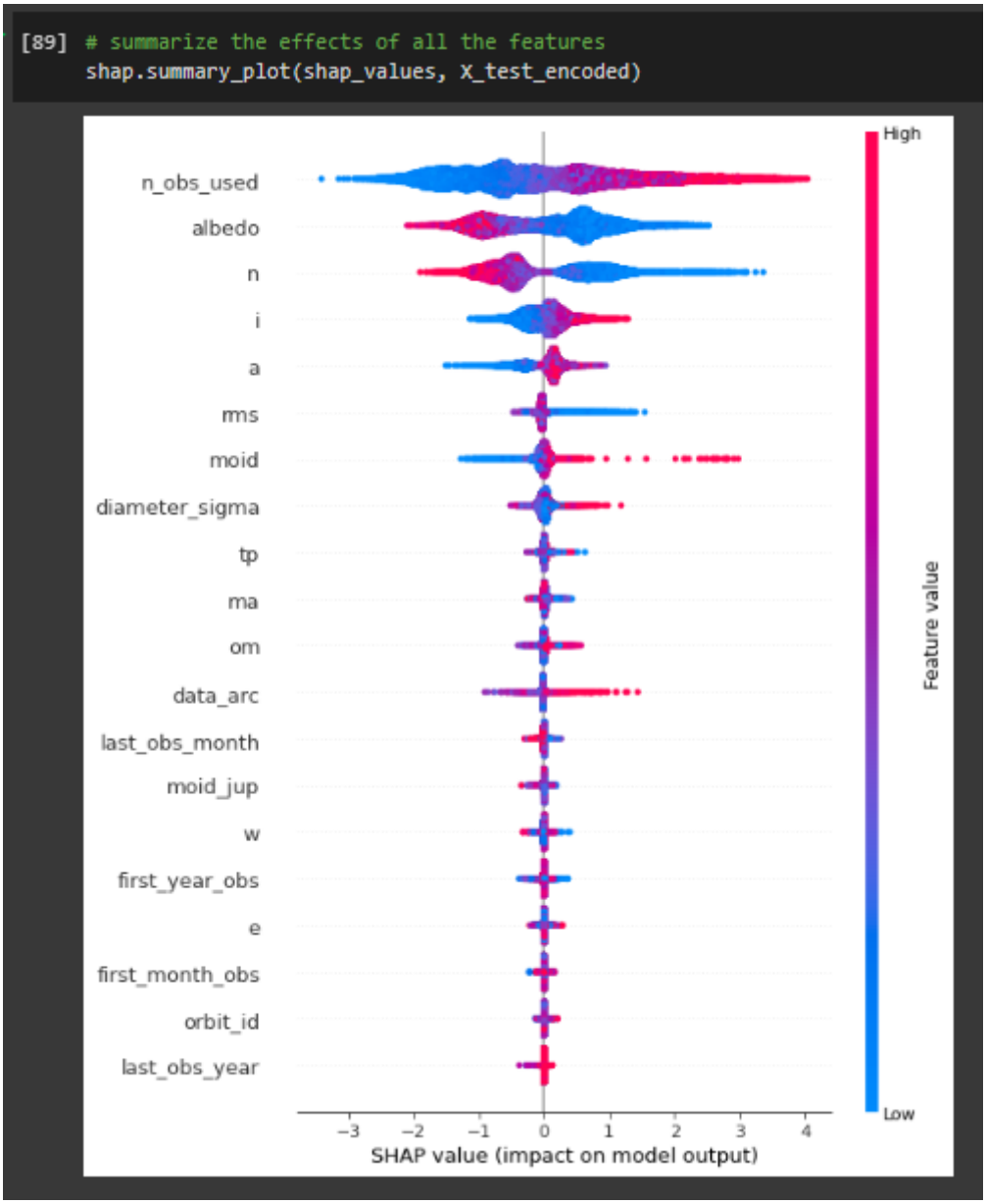


Model Insights

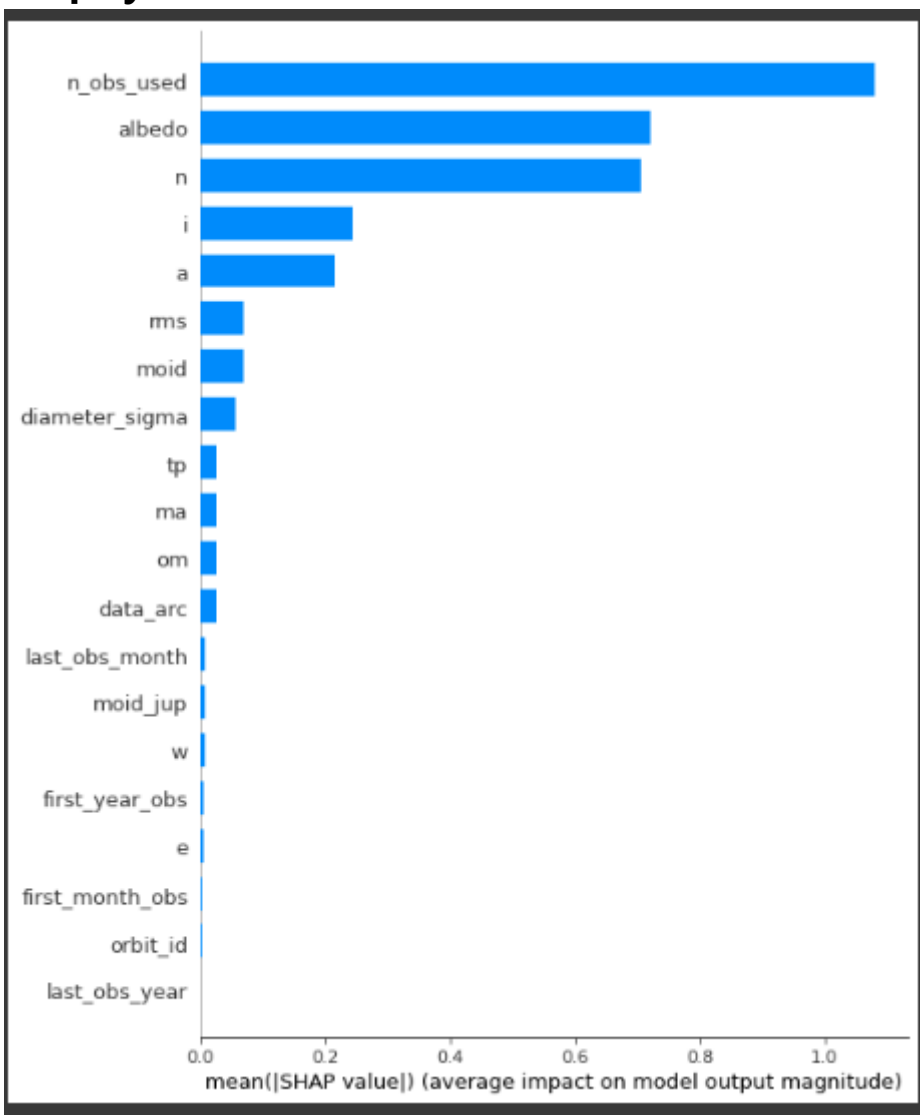
For the shapely plots we used the decision tree model because random forest wouldn't load. By a wide margin, the three most important features to predict diameter are n_obs_used, albedo, and n values.

Shapely Summary Plot

```
[89] # summarize the effects of all the features
     shap.summary_plot(shap_values, X_test_encoded)
```



Shapley Best Predictive Features



Recommendations

A lot of columns where edited and removed from this dataset; I think revisiting the whole dataset may lead to more insights. I also think we could improve our prediction using a boosting algorithm and even neural network classifiers.

Thank you for the trimester Dr Ali.