

Asteroid Dataset

NASA JPL Asteroid Dataset

Big-Data Architecture and Governance

Group 4:

Jaeline Granada,

Harsh Rajendra Oswal,

Divya Patil



Asteroids are celestial objects which can be found all over space and have a wide range of attributes which can help in predicting their potential danger to earth or other planets. Attributes like the nearness to earth, the size of asteroid in terms of diameter, their orbit belt, hazard classification, absolute magnitude, eccentricity of orbital path, angular inclination, the angular speed and the Earth minimum orbit intersection distance are the attributes we have selected in from the dataset to provide intuitive visualizations that can help astronomers and subject matter experts in their further analysis.

Risks/Issues of project

Type	Risk/ Issue -Description	Mitigation
Issue	Improper Environment setup for Data cleansing and data validation i.e., insufficient packages to run python scripts	Make sure all packages are installed and proper environment is set up to run all scripts error for data cleansing/wrangling process
Risk	Many fields needed to track asteroids details has missing data which might be an analytical risk leading to biased visualization	Deriving new columns by understanding existing database attributes to generate reports without errors.
Risk	Inconsistent entries in data available for asteroids classification and thus analysis might be biased eventually which further leads to incomplete data merging or error in the creating graph database	Categorization of the range for available data records is to be done before loading data into the database.
Risk	Lack of Scientific Knowledge in terms of asteroids dataset among team members could lead to incorrect scientific analysis and interpretation	Prior research/study of astronomy and space for understanding dataset

Describe challenges encountered and how you resolved them?

Difficulty understanding data context	Reference to Astronomy Data websites and Small-Body Database published by NASA
Lack of knowledge of scientific terms	NASA JSPL web page reference metadata used
Some columns containing 98% of null values	Accessed the value and dropped unwanted columns, alter values to be used for analysis
Many columns highly skewed when observing distribution	Classifying data into categories
Pinpointing data analysis focus for visualization	Selecting higher weightage metrics like NEO, PHA
Lack of qualitative data beyond asteroid id's/names and general class	Create association by relations between data attributes

Columns used for dimensions, and columns that are used for measurement.

Dimensions

- **Diameter**
- **H (Absolute Magnitude)**
- **Eccentricity**
- **Inclination**
- **Minimum orbit intersection distance(moid)**
- **Angular speed**

Measures

- **Pha**
- **Neo**

How would you generate new dimensions?

- Using diameter and absolute magnitude we can create new dimensions for asteroids size and magnitude range classification .
- Using eccentricity and inclination, we can create a new dimension for analyzing movements of asteroids and trace their path in space.
- Using angular speed and Minimum orbit intersection distance(moid) we can create dimensions to predict collision threats by objects to the earth.

Data Profiling Instructions

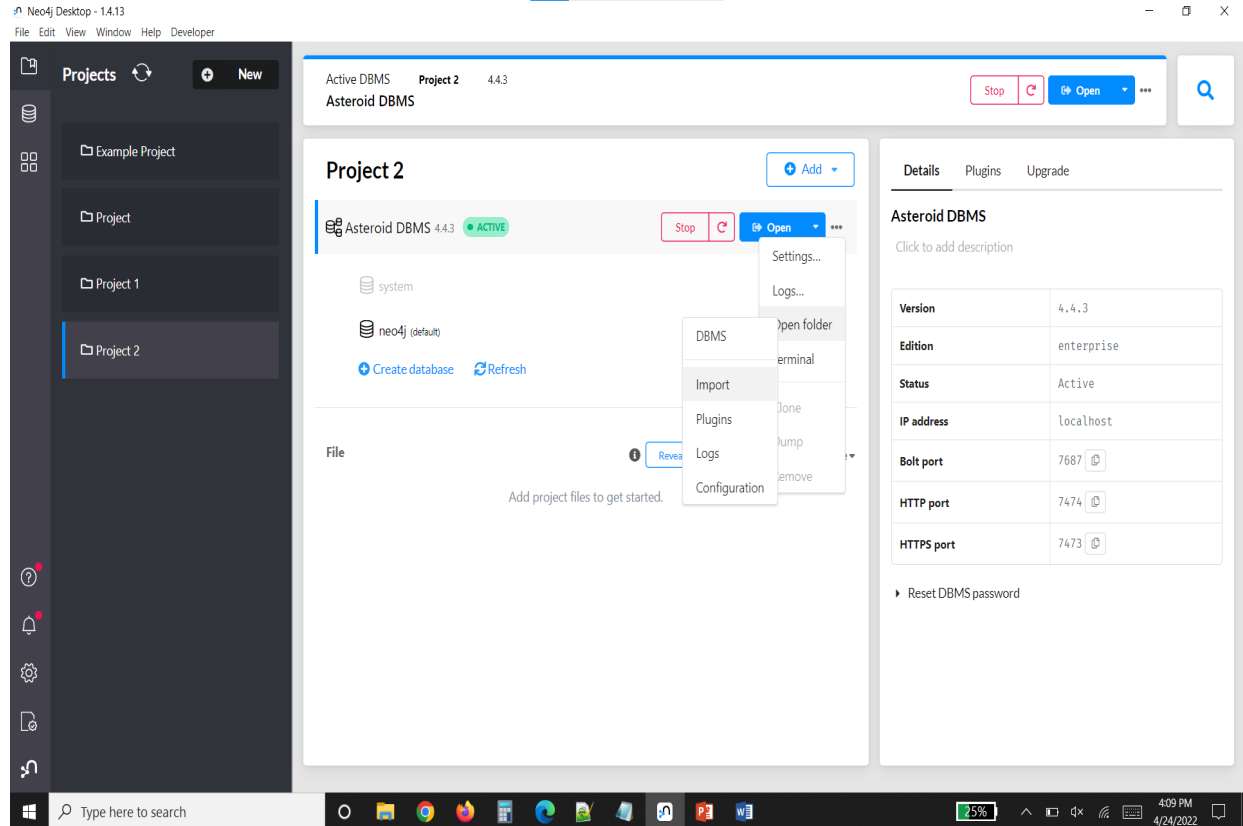
- 1) Download Python (follow instructions on <https://www.python.org/downloads/> based on operating system
- 2) Install pandas library by running `pip install pandas`
- 3) Install pandas profiling to be able to create an html profile report by running the command `pip install pandas-profiling`
- 4) Run profiling.py script and output.html file will be created
- 5) Open output.html on browser to view data profile report

Data Wrangling/Cleansing Instructions

- 1) Make sure pandas library is installed (should have been installed during data profiling)
- 2) Run exploration.py script
- 3) Once script is finished running, “asteroid-filtered-dataset.csv” file will be created
- 4) Use this new filtered dataset for database/visualization

Neo4j Instructions

Graph Database End-User Instructions (Database creation and load):



- Create a new database in Neo4j
- Copy the CSV file into the dbms import folder of the database created
- Start the database and open the database

Constraint Creation:

```
neo4j@bolt://localhost:7687/neo4j - Neo4j Browser
File Edit View Window Help Developer
1 CREATE CONSTRAINT ON (asteroid:Asteroid) ASSERT asteroid.id IS UNIQUE;
2 CREATE CONSTRAINT ON (neo:NEO) ASSERT neo.is_neo IS UNIQUE;
3 CREATE CONSTRAINT ON (h:H) ASSERT h.mag IS UNIQUE;
4 CREATE CONSTRAINT ON (diameter:Diameter) ASSERT diameter.diameter_class IS UNIQUE;
5 CREATE CONSTRAINT ON (eccentricity:Eccentricity) ASSERT eccentricity.e IS UNIQUE;
6 CREATE CONSTRAINT ON (inclination:Inclination) ASSERT inclination.i IS UNIQUE;
7 CREATE CONSTRAINT ON (class:Class) ASSERT class.class_name IS UNIQUE;
8 CREATE CONSTRAINT ON (moid:MOID) ASSERT moid.dist IS UNIQUE;
9 CREATE CONSTRAINT ON (pha:PHA) ASSERT pha.hazard IS UNIQUE;
10 CREATE CONSTRAINT ON (n:N) ASSERT n.speed IS UNIQUE;
11 CREATE CONSTRAINT ON (orbitid:ORBITID) ASSERT orbitid.orbit IS UNIQUE;
```

Node Creation:

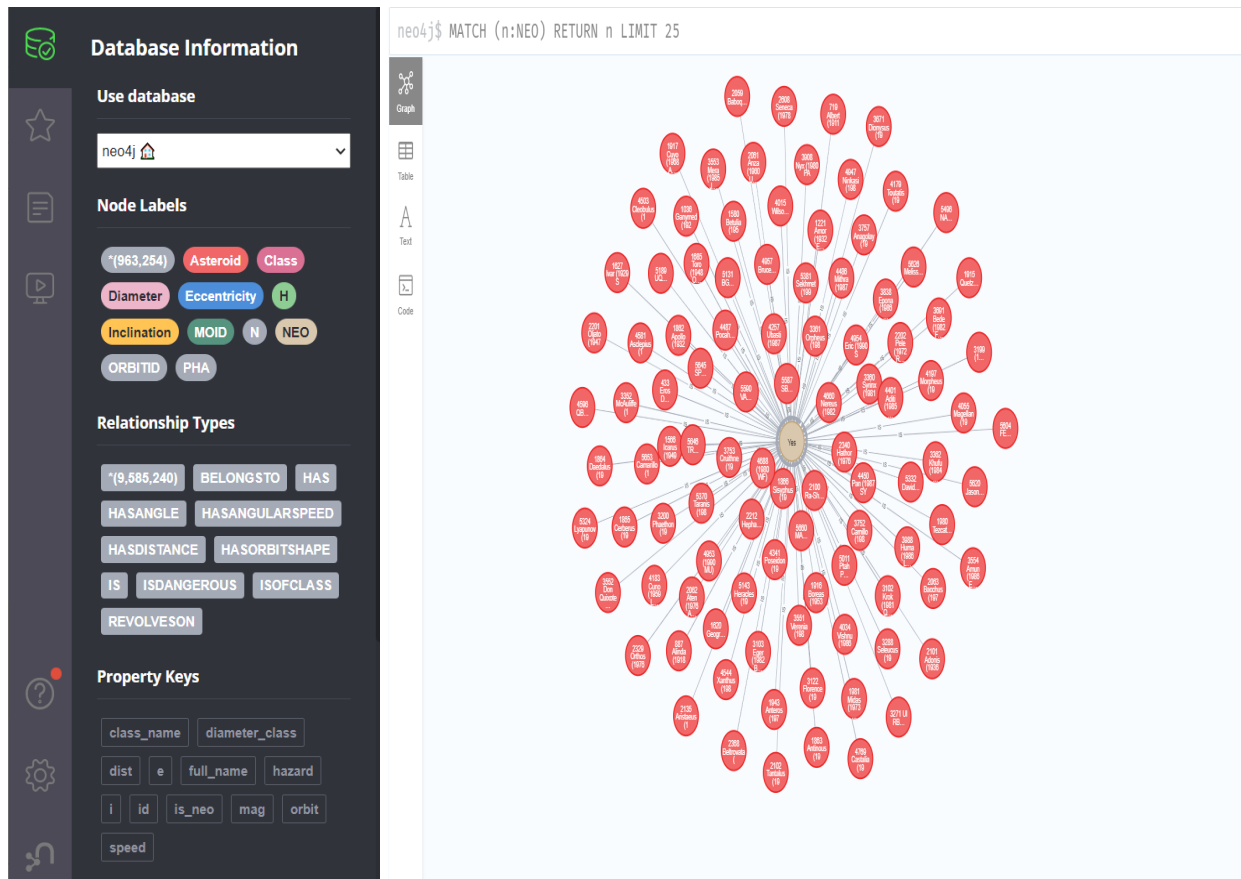
```
15 //Create Nodes
16 //Create Asteroid node
17 :auto USING PERIODIC COMMIT 500
18 LOAD CSV With HEADERS FROM 'file:///Clean_Asteroid_Data.csv' AS row
19 MERGE
20 (asteroid:Asteroid {id:row.asteroidId})
21 ON CREATE SET
22 asteroid.full_name = row.full_name;
```

Relationship Creation:

```
neo4j@bolt://localhost:7687/neo4j - Neo4j Browser
File Edit View Window Help Developer
93 //RELATIONSHIPS
94 //Relationship between Asteroid and NEO
95 :auto USING PERIODIC COMMIT 500
96 LOAD CSV With HEADERS FROM 'file:///Clean_Asteroid_Data.csv' AS row
97 MATCH (asteroid:Asteroid {id:row.asteroidId})
98 MATCH (neo:NEO {is_neo:row.neo})
99 MERGE (asteroid)-[:IS]-(neo);
```

- Copy the cypher code(asteroid_script.txt) to create the constraints, nodes and relationships between them
- Run the code to create the graph database
- The scripts will run one by one and will take a few minutes due to the large dataset size

The graph database should be prepared and can be tested by queries



Analytics Dashboard Documentation

- 1) Download Python if not already downloaded (follow instructions on <https://www.python.org/downloads/> based on operating system)
- 2) Install Anaconda Distribution (follow instructions on <https://docs.anaconda.com/anaconda/install/index.html> based on operating system)
- 3) Once installed, open Anaconda command prompt (anaconda3)
- 4) Install jupyter notebook by running command `pip install notebook`
- 5) Then, install dash framework by running pip command `pip install jupyter-dash`
- 6) After installing dash framework, install other required packages:
 - Pandas: `pip install pandas`
 - Matplotlib: `pip install matplotlib`
 - Plotly: `pip install plotly=5.7.0`
 - NumPy: `pip install numpy`
 - Py2Neo: `pip install py2neo`
- 7) Open jupyter notebook by running command `jupyter notebook` on Anaconda command prompt
- 8) Jupyter Notebook web page should appear
- 9) Navigate to the folder containing dataset and app.ipynb. Ensure both files are in the same folder
- 10) Open app.ipynb on Jupyter Notebook
- 11) Run app.ipynb script
- 12) Wait until code fully loads and IP address appears at the bottom of the page (<http://127.0.0.1:8050/>)
- 13) Navigate to IP address and wait for dashboard to fully load
- 14) Dashboard is displayed!

Technical Metadata :

	A	B	C
1	fields	data type	Description
2	id	(String)	Identifies asteroids uniquely characters
3	full_name	(String)	Identifies name of asteroids
4	neo	(String)	determines asteroids near earth's orbit
5	pha	(String)	Identifies asteroids potential hazards
6	H	(Decimal)	Identifies asteroids absolute magnitude
7	diameter	(Decimal)	determines asteroid's diameter
8	orbit_id	(String)	identifies orbit of asteroids
9	eccentricity	(Decimal)	determines value of eccentricity of asteroid's orbit
0	inclination	(Decimal)	determines the angle of path trace by asteroids
1	n	(Decimal)	identifies angular speed of asteroids
2	moid	(Decimal)	identifies minimum orbit intersection distance
3	class	(String)	identifies class of asteroids
4	diameter_cat	(String)	identifies the classification of asteroid's diameters
5	H_cat	(String)	identifies the classification of asteroid's magnitude
6	eccentricity_cat	(String)	determines the range of eccentricity
7	inclination_cat	(String)	determines the range of inclination of asteroids
8	moid_cat	(String)	determines the range of minimum orbit intersection distance
9	n_cat	(String)	identifies the classification of angular speed

Business Metadata :

From a business point of view, pha(potentially hazardous asteroids) and neo(near-earth object) will help to understand which asteroids are near to the earth and whether they are dangerous to earth or not.

Full_name identifies the asteroid's name and orbit_id represents which asteroid is revolving in which orbit and helps to locate them.

The diameter and absolute magnitude(H) column determine the size and shape of asteroids which helps to categorize them into different categories like small, medium, and large.

Columns like eccentricity and inclination help to trace the path of asteroids moving in space.

n(angular speed) corresponds to the speed of asteroids in space which helps to understand if an asteroid coming from the asteroid belt collides with the earth

Testing and Validation

System Integration Testing and User Acceptance Testing

Test 1: Checking Number of Null Values

Explanation: Number of null values was accessed before and after data wrangling. Null values were not removed but were altered for use in data analysis.

```
print(df.isna().sum())
```

Before handling null values

```
id          0
full_name   0
neo         4
pha        19921
H           6263
diameter    822315
orbit_id    0
e           0
i           0
n           0
moid        19921
class       0
dtype: int64
```



After handling null values

```
id          0
full_name   0
neo         0
pha         0
H           0
diameter    0
orbit_id    0
e           0
i           0
n           0
moid        0
class       0
dtype: int64
```

Test 2: Null Values Replaced

Explanation: These metrics were used in our data analysis and we wanted to account for unknown values instead of just removing them from the dataset.

```
#replace null values with 'unknown'
df['pha'] = df['pha'].fillna('Unknown')

df['neo'] = df['neo'].fillna('Unknown')

df['diameter'] = df['diameter'].fillna('Unknown')

df['H'] = df['H'].fillna('Unknown')

df['moid'] = df['moid'].fillna('Unknown')
```

Before replacing columns containing null values (same id as reference)

	id	full_name	neo	pha	H	diameter	orbit_id	e	i	n	moid	class
741612	bK13CD4A	(2013 CA134)	NaN	N	10.748	NaN	JPL 7	1.855356	8.643584	0.06463	4.25033	HYA

After replacing columns containing null values

	id	full_name	neo pha	H diameter	orbit_id	e	i	n	moid	class		
741612	bK13CD4A	(2013 CA134)	Unknown	N	10.748	Unknown	JPL 7	1.855356	8.643584	0.06463	4.25033	HYA

Test 3: Dropping Duplicates

Explanation: No Duplicates were found in the dataset. The number of rows and columns were unchanged after performing a duplicate drop function.

```
print(df.shape)
df.drop_duplicates()
print("duplicates dropped")
print(df.shape)
```

Before and after dropping duplicates

```
(958524, 45)
duplicates dropped
(958524, 45)
```

Test 4: Dropped Columns Not Being Used for Analysis

Explanation: Columns dropped were sigma values or scientific values that we could not fully understand scientifically for use in our data analysis.

```
df = df.drop('sigma_q', 1)
df = df.drop('sigma_i', 1)
df = df.drop('sigma_om', 1)
df = df.drop('sigma_w', 1)
df = df.drop('sigma_ma', 1)
df = df.drop('sigma_ad', 1)
df = df.drop('sigma_n', 1)
df = df.drop('sigma_tp', 1)
df = df.drop('sigma_per', 1)
df = df.drop('sigma_e', 1)
df = df.drop('sigma_a', 1)
```

Before dropping columns

```
[958524 rows x 45 columns]
```



After dropping columns

```
[958524 rows x 12 columns]
```

Test 5: Remove Whitespaces

Explanation: We removed whitespaces from our columns that had string values.

```
#remove whitespaces in full_name
df['full_name'] = df['full_name'].str.strip()
#remove whitespaces in orbit_id
df['full_name'] = df['orbit_id'].str.strip()
```

Before removing whitespaces from column “full_name”

id	full_name	neo	pha
a0000001	1 Ceres	N	N
a0000002	2 Pallas	N	N
a0000003	3 Juno	N	N



After removing whitespaces

id	full_name	neo	pha
a0000001	1 Ceres	N	N
a0000002	2 Pallas	N	N
a0000003	3 Juno	N	N

Test 6: Renamed columns

Explanation: Renamed specific columns for better understanding pertaining to business terms.

```
df.rename(columns={"id": "asteroidId", "e": "eccentricity", "i": "inclination"}, inplace=True)
```

Before renaming columns in dataset

```
Index(['id', 'full_name', 'neo', 'pha', 'H', 'diameter', 'orbit_id', 'e', 'i',  
      'n', 'moid', 'class'],  
      dtype='object')
```

After renaming columns

```
Index(['asteroidId', 'full_name', 'neo', 'pha', 'H', 'diameter', 'orbit_id',  
      'eccentricity', 'inclination', 'n', 'moid', 'class'],  
      dtype='object')
```

Test 7: Creating new columns based on value ranges

Explanation: We created new column categories for our metrics based on ranges to classify asteroids.

```
# create a function
def d_cat(diameter):
    if type(diameter) == str:
        return "unknown-size"
    if diameter>=0 and diameter<=24.9999:
        return "small_size"
    elif diameter>=25 and diameter<=99.9999:
        return "medium_size"
    elif diameter>=100:
        return "large_size"
# create a new column based on condition
df['diameter_cat'] = df['diameter'].apply(d_cat)
```

Before adding new diameter category column

```
Index(['id', 'full_name', 'neo', 'pha', 'H', 'diameter', 'orbit_id', 'e', 'i',
      'n', 'moid', 'class'],
      dtype='object')
```

After adding the diameter category column to our dataset

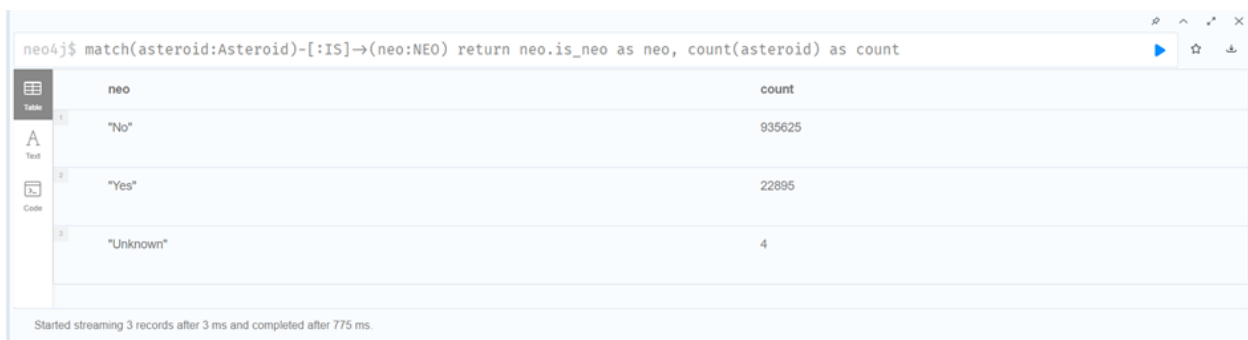
```
Index(['id', 'full_name', 'neo', 'pha', 'H', 'diameter', 'orbit_id', 'e', 'i',
      'n', 'moid', 'class', 'diameter_cat'],
      dtype='object')
```

```
0         large_size
1         large_size
2         large_size
3         large_size
4         large_size
...
958519    unknown-size
958520    unknown-size
958521    unknown-size
958522    unknown-size
958523    unknown-size
Name: diameter_cat, Length: 958524, dtype: object
```

Neo4j Testing screenshots using Cypher Queries

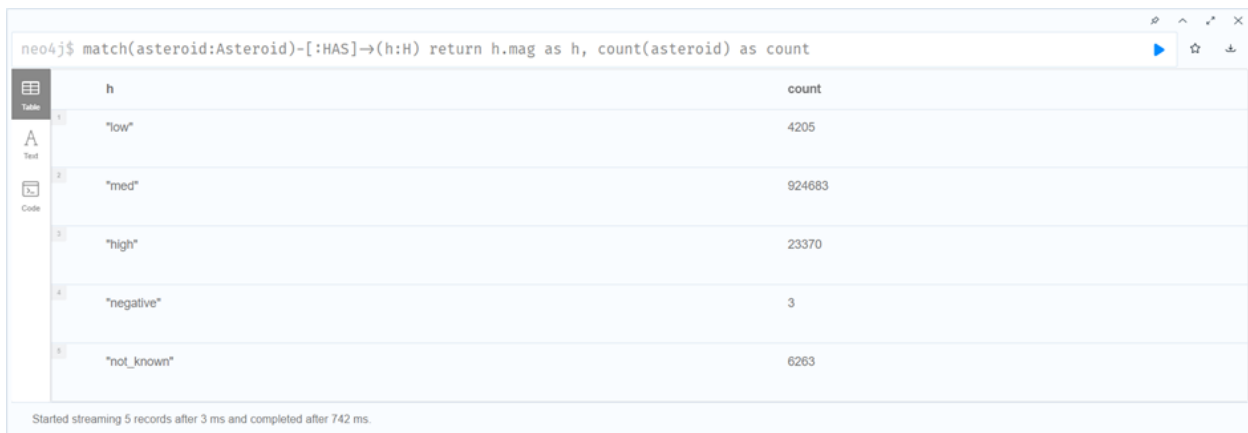
Steps for testing and verification of data using Neo4j:

- Copy the cypher code(neo4j_testing_script.txt) containing the test queries
- Run the code to view graphs and tables verifying data load to Neo4j
- The scripts will run one by one and will take a few minutes displaying the below results



The screenshot shows the Neo4j Cypher query interface. The query entered is: `neo4j$ match(asteroid:Asteroid)-[:IS]-(neo:NEO) return neo.is_neo as neo, count(asteroid) as count`. The results are displayed in a table with two columns: 'neo' and 'count'. The table contains three rows of data. A status bar at the bottom indicates: 'Started streaming 3 records after 3 ms and completed after 775 ms.'

	neo	count
1	"No"	935625
2	"Yes"	22895
3	"Unknown"	4



The screenshot shows the Neo4j Cypher query interface. The query entered is: `neo4j$ match(asteroid:Asteroid)-[:HAS]-(h:H) return h.mag as h, count(asteroid) as count`. The results are displayed in a table with two columns: 'h' and 'count'. The table contains five rows of data. A status bar at the bottom indicates: 'Started streaming 5 records after 3 ms and completed after 742 ms.'

	h	count
1	"low"	4205
2	"med"	924683
3	"high"	23370
4	"negative"	3
5	"not_known"	6263

neo4j\$ match(asteroid:Asteroid)-[:ISOFLCLASS]→(diameter:Diameter) return diameter.diameter_class as diameter, count(asteroid)...

	diameter	count
1	"large_size"	240
2	"medium_size"	1669
3	"small_size"	134300
4	"unknown-size"	822315

Started streaming 4 records after 3 ms and completed after 762 ms.

neo4j\$ match(asteroid:Asteroid)-[:HASORBITSHAPE]→(eccentricity:Eccentricity) return eccentricity.e as eccentricity, count(as...

	eccentricity	count
1	"elliptical"	910840
2	"circular"	47680
3	"hyperbola"	4

Started streaming 3 records after 3 ms and completed after 672 ms.

neo4j\$ match(asteroid:Asteroid)-[:HASANGLE]→(inclination:Inclination) return inclination.i as inclination, count(asteroid) a...

	inclination	count
1	"acute"	958408
2	"obtuse"	116

Started streaming 2 records after 2 ms and completed after 659 ms.

neo4j\$ match(asteroid:Asteroid)-[:BELONGSTO]→(class:Class) return class.class_name as class, count(asteroid) as count

	class	count
1	"MBA"	855954
2	"OMB"	28355
3	"MCA"	18685
4	"AMO"	8457
5	"IMB"	20360
6	"TJN"	8221
7	"CFN"	506

Started streaming 13 records after 2 ms and completed after 732 ms.

neo4j\$ match(asteroid:Asteroid)-[:HASDISTANCE]→(moid:MOID) return moid.dist as moid_cat, count(asteroid) as count

	moid_cat	count
1	"small"	926816
2	"medium"	8512
3	"large"	3275
4	"unknown"	19921

Started streaming 4 records after 3 ms and completed after 667 ms.

neo4j\$ match(asteroid:Asteroid)-[:ISDANGEROUS]→(pha:PHA) return pha.hazard as pha, count(asteroid) as count

	pha	count
1	"No"	936537
2	"Yes"	2066
3	"Unknown"	19921

Started streaming 3 records after 1 ms and completed after 467 ms.

neo4j\$ match(asteroid:Asteroid)-[:HASANGULARSPEED]→(n:N) return n.speed as n, count(asteroid) as count

	n	count
1	"extremely"	593270
2	"slow"	352832
3	"fast"	12422

Started streaming 3 records after 2 ms and completed after 565 ms.

neo4j\$ match(asteroid:Asteroid)-[:REVOLVESON]→(orbitid:ORBITID) return orbitid.orbit as orbit_id, count(asteroid) as count

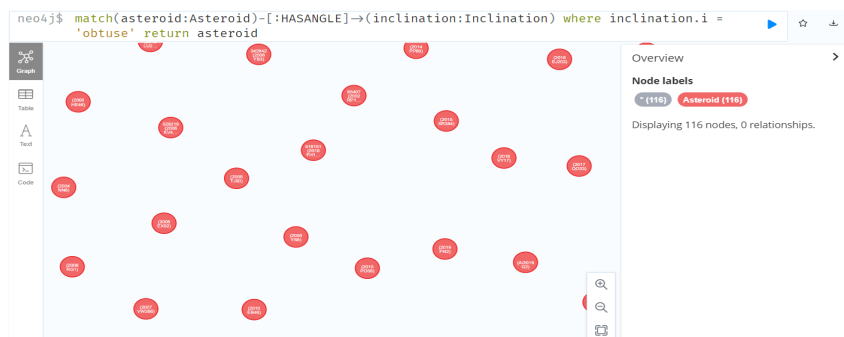
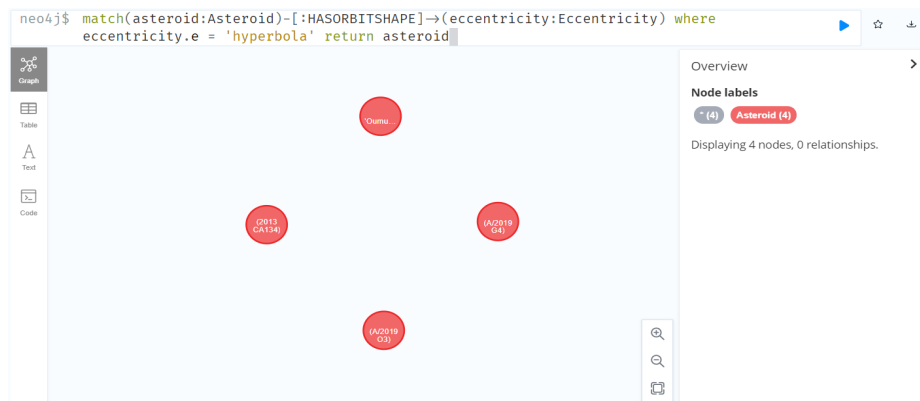
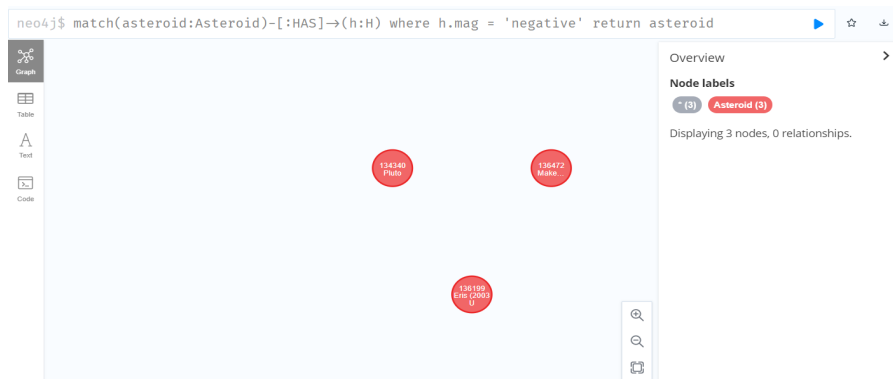
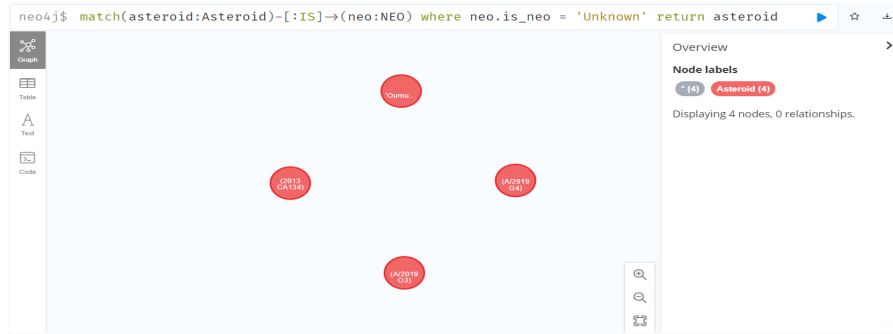
	orbit_id	count
1	"JPL 47"	45
2	"JPL 37"	569
3	"JPL 112"	8
4	"JPL 35"	1083
5	"JPL 114"	9
6	"JPL 89"	9
7	"110"	0

Started streaming 4690 records after 2 ms and completed after 8 ms, displaying first 1000 rows.

neo4j\$ match(asteroid:Asteroid)-[:ISOFLCLASS]→(diameter:Diameter) return diameter.diameter_class as diameter, count(asteroid) as count

	diameter	count
1	"large_size"	240
2	"medium_size"	1669
3	"small_size"	134300
4	"unknown-size"	822315

Started streaming 4 records after 1 ms and completed after 446 ms.



Who would use this Dashboard and how would they benefit from it?

End-Users : Astronomers examining potentially hazardous asteroids based on metrics and categories

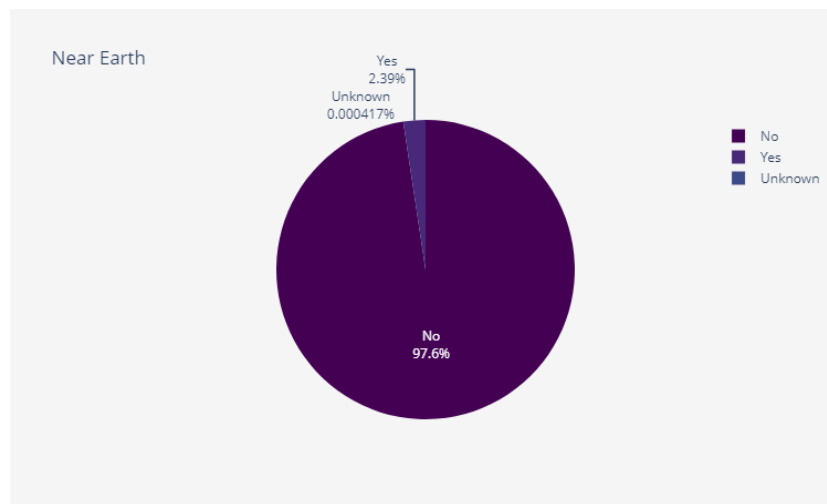
What value would be generated using this dashboard?

- The data such as distance of intersection to other objects, the angular speed, nearness to earth and classification of level of magnitude, diameter and potential hazard are essential parameters for Astronomers, these attributes and their patterns can be easily identified through our dashboard at a single glance instead of requiring numerous calculations for every parameter used.
- Asteroids posing danger to the earth can be focused on towards further studies.
- The classification of Asteroids can help Astronomers and Astrophysicists make better decisions while planning vehicular orbital launches ensuring safety of crew and costly resources used.
- These visuals will assist them in identifying and classifying Asteroids for further analysis.

Dashboard Interpretation/Findings

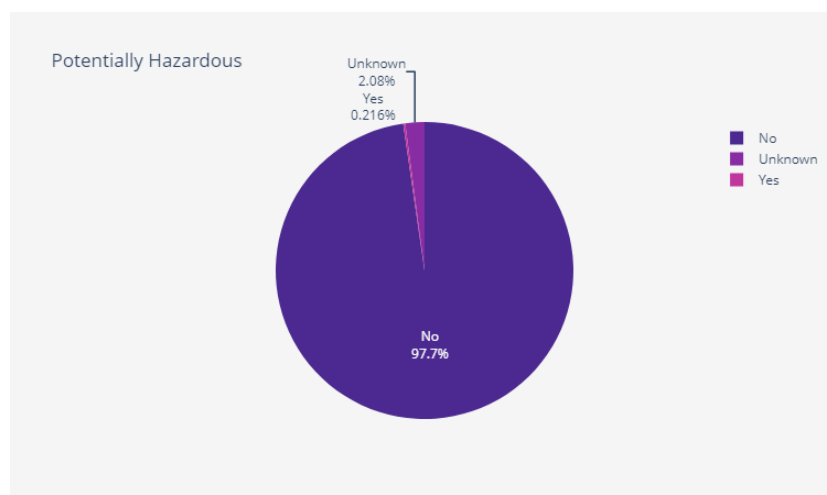
First Diagram:

This pie chart demonstrates the percentage of asteroids that are and are not located near earth. This is the first step in classifying asteroids as potentially hazardous. 97% of asteroids have a value of “No”, showing that the majority of asteroids are not located near earth. Less than 0.00004% of asteroids are not known to be located near earth or not.



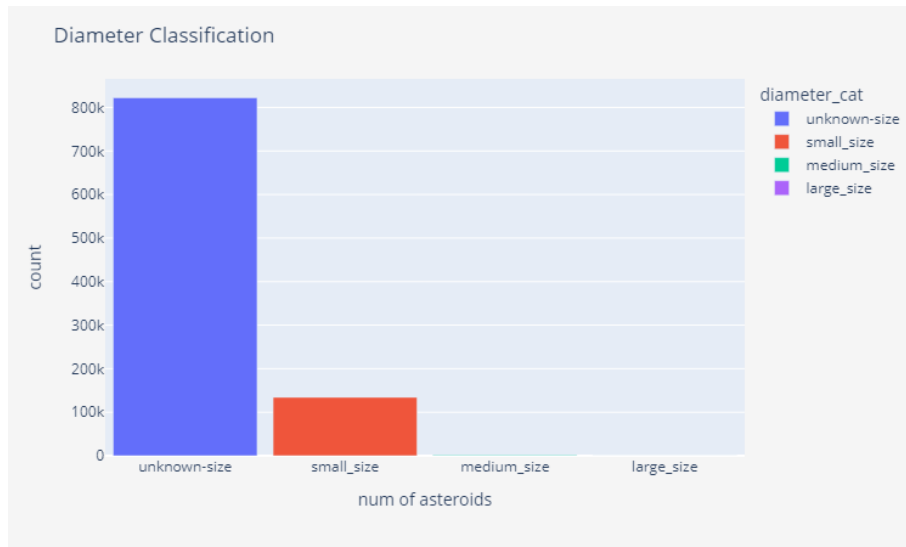
Second Diagram:

This pie chart highlights the percentage of asteroids that are and are not considered to be potentially hazardous. Although this is a boolean value, there are other metrics to consider in the following diagrams for future predictions of asteroid threats. The chart shows that a majority of 97.7% of asteroids are considered not to be potentially hazardous. The unknown value must also be observed as 0.21% of asteroids have yet to be determined to be potentially hazardous or not. This will assist in filtering asteroids that require further analysis and metrics.



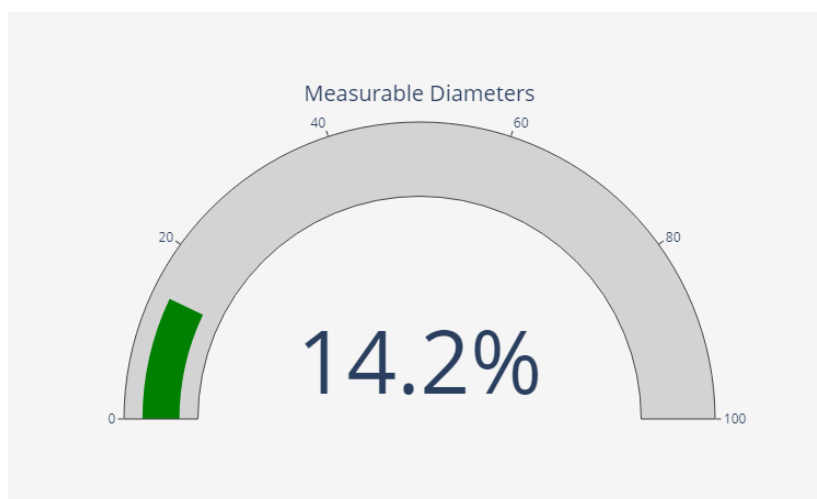
Third Diagram:

This bar graph depicts the distribution of asteroid categories based on diameters. The first bar which includes over 800 thousand asteroids shows the unknown category. For asteroid diameters that are known, the majority of them are of small size. This can be related to either the inability to measure large diameters or asteroids being of small size overall.



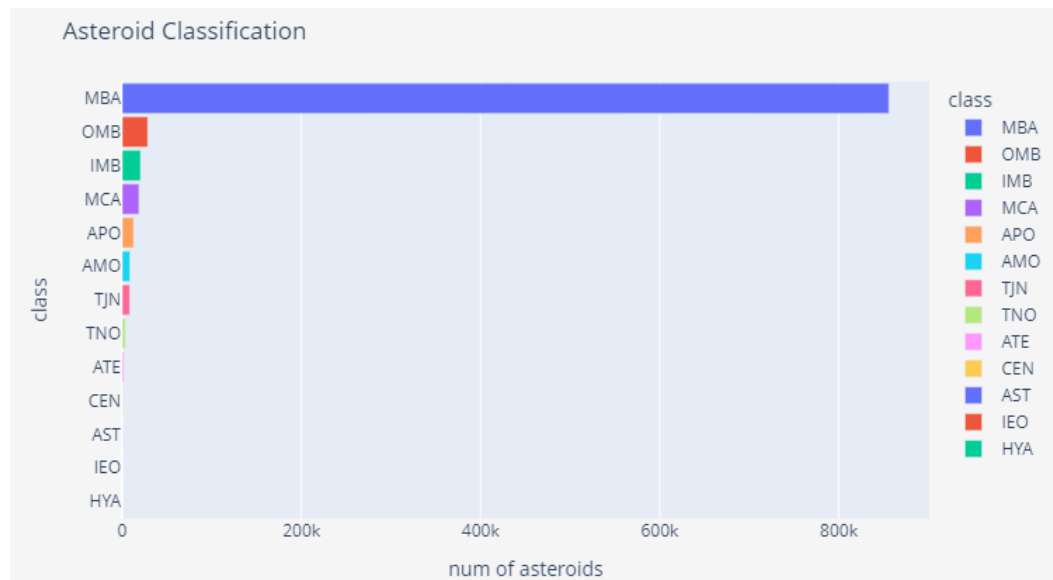
Fourth Diagram:

After analyzing the bar chart, this metric demonstrates the percentage of measurable asteroid diameters. That is, only 14.2% of asteroids have diameter values measured and recorded. This can be due to either excessive sized asteroids unable to be measured or lack of prioritizing the diameter metric in analysis.



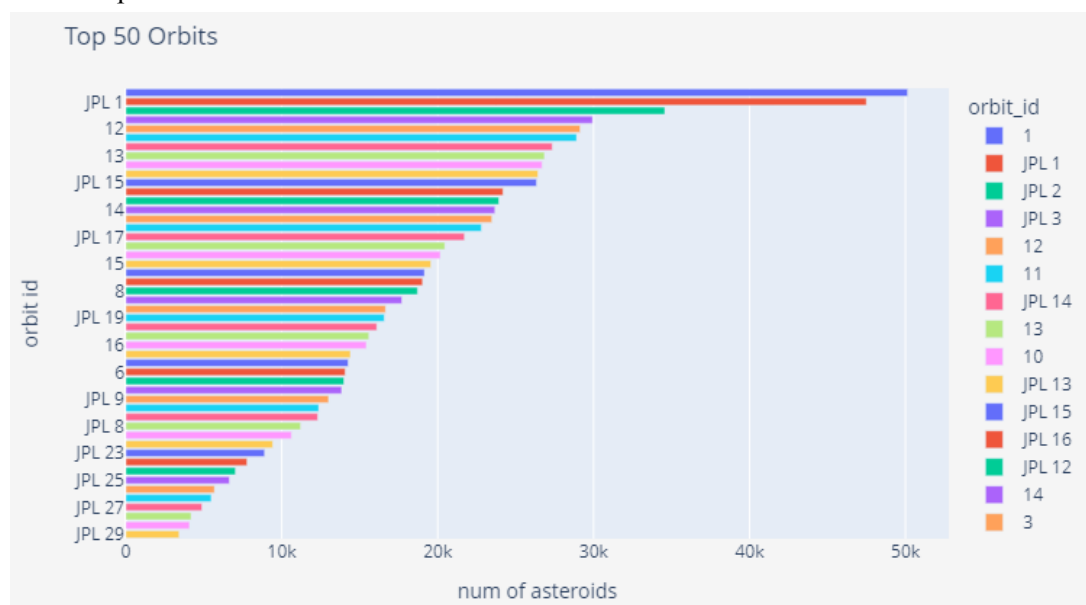
Fifth Diagram:

This bar chart shows asteroid classification by “class” in descending order. The class consisting of over 800 thousand asteroids(the majority) is the MBA class. Upon further scientific analysis, we can find that MBA means “main-belt asteroids”. A large portion of asteroids in our dataset are orbiting between Mars and Jupiter in the main portion of the asteroid belt. These are the asteroids orbiting closest to Earth as well. This puts a microscope on specific asteroids to analyze and continually monitor that are revolving on the main portion of the asteroid belt which is closest to Earth.



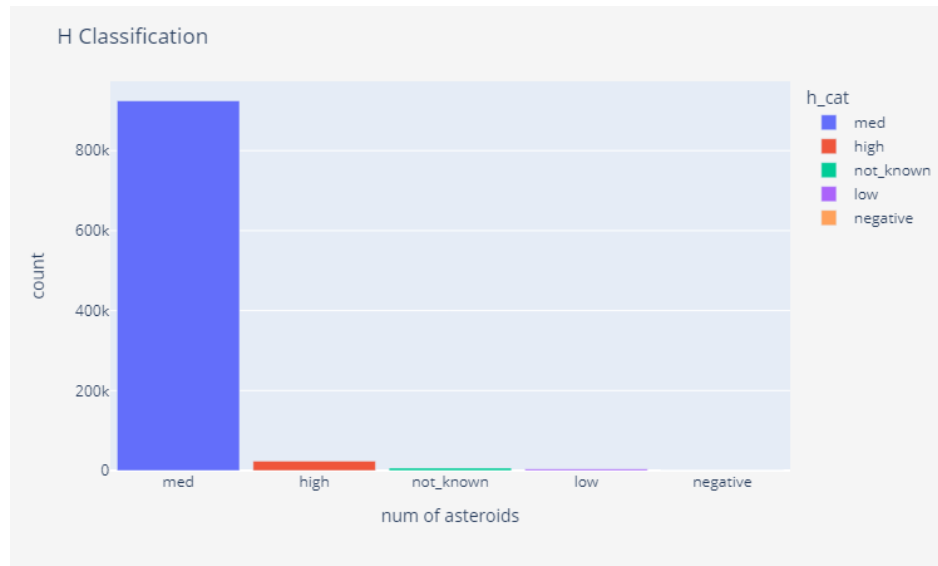
Sixth Diagram:

This bar chart depicts the top 50 orbit ID's. This demonstrates the orbits that had the most asteroids revolving on them. Because there is a significantly large number of orbit ID's, by focusing on the top 50 orbits, we are able to also narrow down our focus pertaining to the number of asteroids to be analyzed. Using the orbit ID's with the most asteroids, we can find the orbit coordinates and analyze corresponding asteroids proximities to Earth.



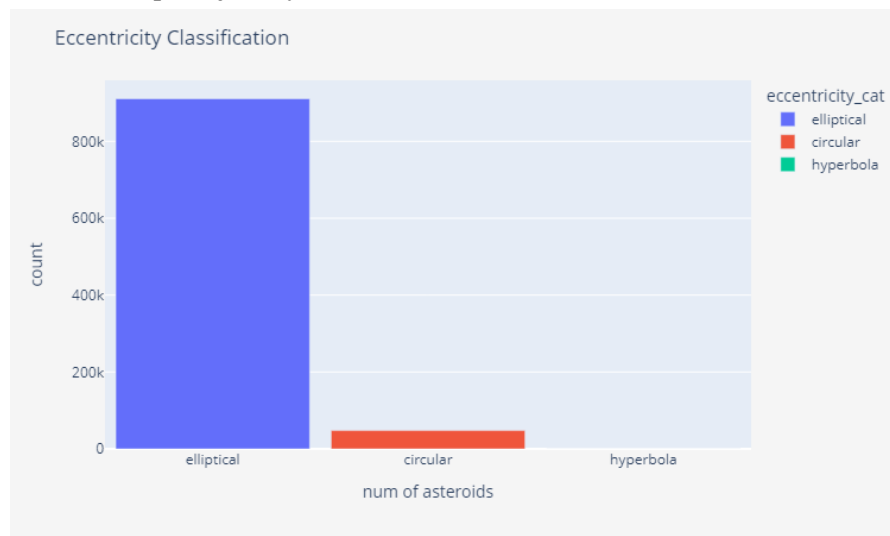
Seventh Diagram:

The bar graph depicts the classification of asteroids based on their absolute magnitude. Over 900 thousand asteroids have a medium magnitude i.e. between the range of 10-20 while 23 thousand asteroids have high magnitudes of over 20, only 6000 asteroids have an unknown absolute magnitude. The absolute magnitude provides astronomers information regarding the luminosity of asteroids which can hence be used to calculate their distance relative to other objects like the earth.



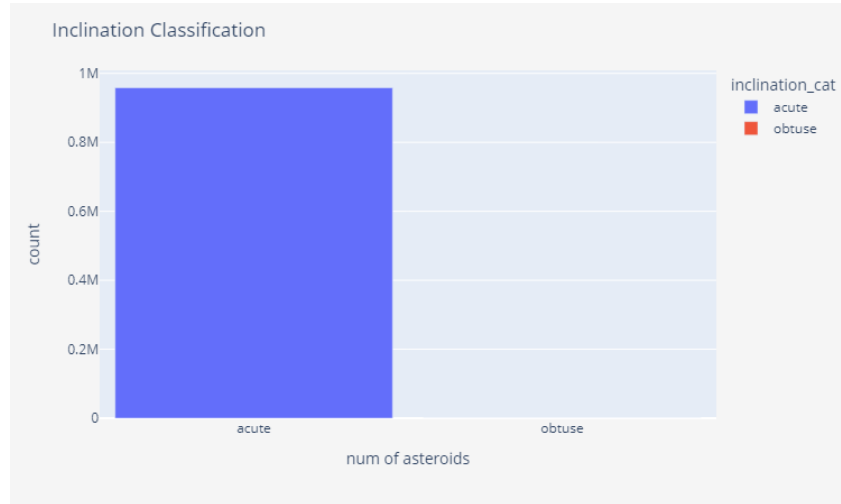
Eighth Diagram:

The visualization below shows the classification of asteroids based on their eccentricity. Eccentricity values are used to trace the path of asteroids and hence perceive the objects in path or at a distance that could be relatively at risk of collision. 900 thousand asteroids follow an elliptical path having eccentricity values between 0.05 and 1 while 47 thousand have a circular path ($e < 0.05$) and only 4 asteroids follow a hyperbolic path ($e > 1$). The asteroids having a parabolic path of exactly 1 can be dangerous since they have an escape trajectory.



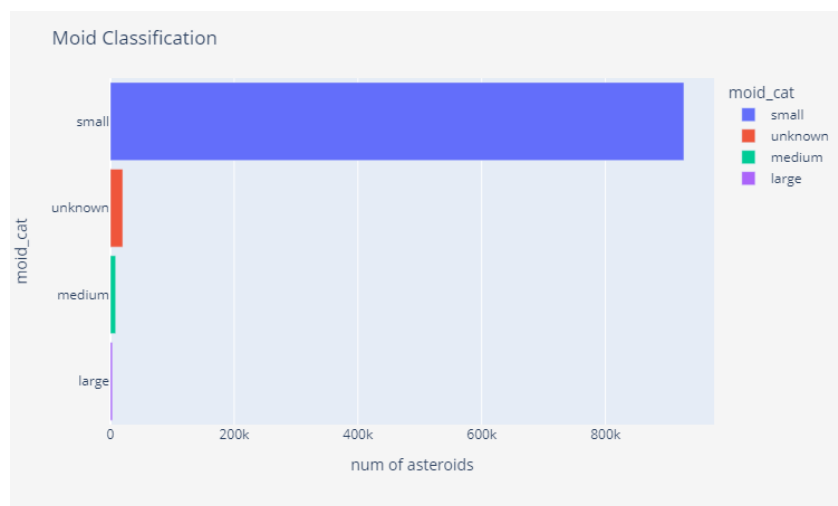
Ninth Diagram:

The bar graph depicts that only 116 Asteroids have an obtuse angle of inclination with their axis, all other 950 thousand asteroids have an acute angle of inclination. The angle of inclination is used to define the orbital path inclination and hence help in better calculating the deviation from current orbit. These measures help astronomers account for deviation while calculating the asteroid's path relative to the earth.



Tenth Diagram:

The bar graph shows the Earth Minimum Orbit Intersection Distance. This is one of the most essential pieces of information from the dataset and the visualization helps clearly communicate about how many asteroids pose high risk to the earth because of being in close vicinity. About 920 thousand asteroids are in the small moid category which is due to their small orbital intersection with earth. 20 thousand asteroids have an unknown moid which would require these asteroids to be further investigated to ensure collision with earth can be predicted. While the remaining asteroids having medium to large moid pose less to no risk level to earth.



Eleventh Diagram:

The visualization shows the asteroids classified by their angular speeds. 593 thousand asteroids have an extremely slow angular speed, 350 thousand have a slow speed and only 12 thousand have a fast speed. The angular speed is an important factor for classifying an asteroid as hazardous or not since a small asteroid at very high speed can be more dangerous than a bigger asteroid at a very low speed which might not make it through the orbit of other planets or the earth.

