Data-Science 1

What is our GOAL for this MODULE?

The goal of this module is to plot charts and find interesting insights about exo-planets

What did we ACHIEVE in the class TODAY?

- We plotted different charts in this class
- We learned about Gravity and how we can calculate gravity of a planet

Which CONCEPTS/CODING BLOCKS did we cover today?

- Plotly
- Python
- Logical reasoning

How did we DO the activities?

1. Take a look at the final data we came up with in the last class

   https://raw.githubusercontent.com/whitehatjr/Data-cleaning/master/main.csv

2. Let's recall the meaning of all the columns again:

   - ***name -*** name of the planet.
   - ***light_years_from_earth*** - Distance of the exo-planet from earth in light years.
   - ***planet_mass*** - Mass of the planet.
   - ***stellar_magnitude -*** This is the brightness of the host star of the planet when observed from Earth (just as the sun is our host star).
   - ***discovery_date -*** This is the year of discovery for the exo-planet.
   - ***planet_type -*** This is the type of the planet (Gas Giant, Super Earth, etc.).
   - ***planet_radius -*** This is the radius of the exo-planet with respect to Earth or Jupiter.
   - ***orbital_radius -*** This is the average distance of this exo-planet from its sun. Just like our solar system has 1 sun, there are multiple solar systems that contain many planets and sun(s).
   - ***orbital_period -*** This is the time it takes to complete one orbit of it's sun.
   - ***eccentricity -*** This denotes how circular the orbit is. It might be oval in shape too. The lower the eccentricity, the more circular is the orbit.
   - ***solar_system_name -*** The name of the host solar system.
   - ***planet_discovery_method -*** This is the discovery method which was used to find this exo-planet.
   - ***planet_orbital_inclination -*** This is the orbital inclination, which means that it is the tilt of the exo-planet's orbit when it revolves around its sun.
   - ***planet_density -*** This is the density of the planet.
   - ***right_ascension -*** This is the right ascension of the planetary system, which is the east-west coordinate by which the position of this planet is measured.
   - ***declination -*** This is the north-south coordinate by which the position of the planet is measured.
   - ***host_temperature -*** This is the temperature of the host star in Kelvin.

- ***host_mass* -** This is the amount of mass contained in the host star.
- ***host_radius* -** This is the radius of the host star.

3. Now, just like we have 8 planets in our solar system (Mercury, Venus, Earth, Mars, Jupiter, Saturn, Neptune, Uranus), let's try to find which other solar system has the maximum number of planets.
4. For this, we have a column named as - **solar_system_name** in our CSV. We will create a dictionary to maintain the count of planets each solar system has!
5. Before we do that, we can see that we have index mentioned in the rows as the first element, but the header's first element is empty:

```
import csv
rows = []

with open("main.csv", "r") as f:
 csvreader = csv.reader(f)
 for row in csvreader:
   rows.append(row)

headers = rows[0]
planet_data_rows = rows[1:]
print(headers)
print(planet_data_rows[0])
```

```
['', 'name', 'light_years_from_earth', 'planet_mass', 'stellar_magnitude', 'discovery_date', 'planet_type', 'planet_radius',
['0', '11 Comae Berenices b', '305.0', '19.4 Jupiters', '4.74', '2007', 'Gas Giant', '1.08 x Jupiter', '1.29 AU', '326 days',
```


6. We will quickly fix this and then find the solar system with the maximum number of planets.

```
headers[0] = “row_num”
solar_system_planet_count = {}
for planet_data in planet_data_rows:
 if solar_system_planet_count.get(planet_data[11]):
   solar_system_planet_count[planet_data[11]] += 1
 else:
   solar_system_planet_count[planet_data[11]] = 1
```

```
max_solar_system = max(solar_system_planet_count, key=solar_system_planet_count.get)
print("Solar system {} has maximum planets {} out of all the solar systems we have discovered so far!".format(max_solar_system, solar_system_planet_count[max_solar_system]))
```

7. Here, we will see that the solar system **HD 10180** has maximum planets (6). Could this be our next home? Let's get the list of all the planets in this solar system.

```
hd_10180_planets = []
for planet_data in planet_data_rows:
 if max_solar_system == planet_data[11]:
   hd_10180_planets.append(planet_data)

print(len(hd_10180_planets))
print(hd_10180_planets)
```

8. Great! This will give us the list of all the planets in it's solar system.
9. Before we plot charts, let's look at our data closely. We can see that the columns **planet_mass** and **planet_radius** have values with reference to **Jupiter** or **Earth.** There are also few values "**Unknown**". Let's make our data uniform.

```
temp_planet_data_rows = list(planet_data_rows)
for planet_data in temp_planet_data_rows:
 planet_mass = planet_data[3]
 if planet_mass.lower() == "unknown":
   planet_data_rows.remove(planet_data)
   continue
 else:
   planet_mass_value = planet_mass.split(" ")[0]
   planet_mass_ref = planet_mass.split(" ")[1]
   if planet_mass_ref == "Jupiters":
     planet_mass_value = float(planet_mass_value) * 317.8
   planet_data[3] = planet_mass_value

 planet_radius = planet_data[7]
 if planet_radius.lower() == "unknown":
```

```
   planet_data_rows.remove(planet_data)
   continue
 else:
   planet_radius_value = planet_radius.split(" ")[0]
   planet_radius_ref = planet_radius.split(" ")[2]
   if planet_radius_ref == "Jupiter":
     planet_radius_value = float(planet_radius_value) * 11.2
   planet_data[7] = planet_radius_value
```

10. Here, we are removing all unknown planets and are converting all the planets mass and radius in reference to Earth for our ease. We are left with 4,251 planets who's value we know with reference to Earth with following metrics:

    1 Jupiter Mass = 317.8 Earth Mass
    1 Jupiter Radius = 11.2 Earth Radius

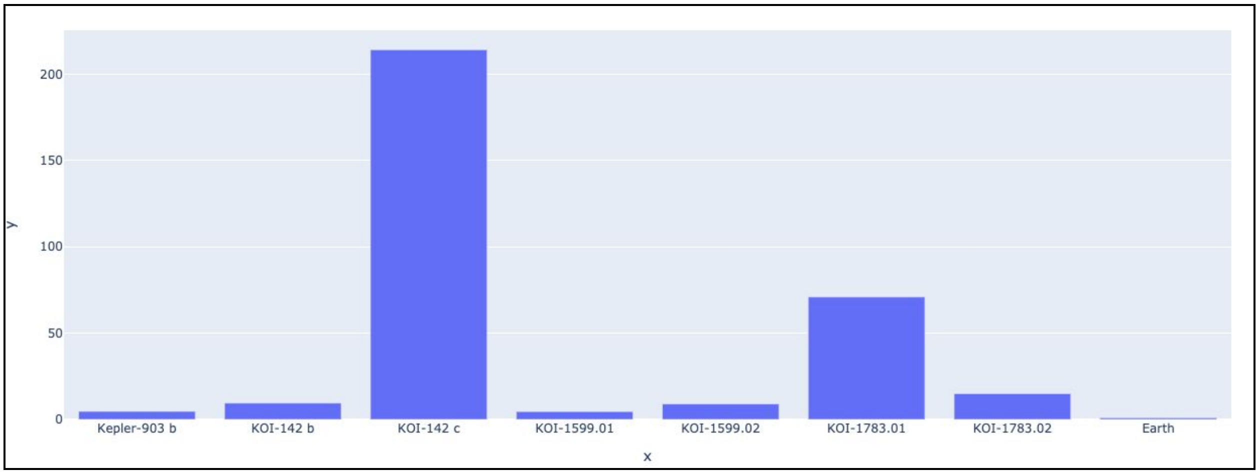
11. Let's plot a bar chart on planet mass. We will also add the mass of Earth (1) in our case as the values are in reference with Earth's mass now.

```
import plotly.express as px

hd_10180_planet_masses = []
hd_10180_planet_names = []
for planet_data in hd_10180_planets:
 hd_10180_planet_masses.append(planet_data[3])
 hd_10180_planet_names.append(planet_data[1])

hd_10180_planet_masses.append(1)
hd_10180_planet_names.append("Earth")

fig = px.bar(x=hd_10180_planet_names, y=hd_10180_planet_masses)
fig.show()
```



12. We can plot the same with all the planets, but let's understand one thing before that:
    - Great Scientist Albert Einstein gave us a formula with which we can calculate the gravity of any planet.

$$g = \frac{G * M_{earth}}{d^2}$$

- Here, G is a gravitational constant, which means that it will always be the same. ***The value of G (Gravitational Constant) is 6.674e-11.***
- M(earth) is the mass of Earth (or any other planet if we are calculating it for another planet). **Mass of Earth = 5.972e+24**
- d is the radius of the planet! **Radius of Earth = 6371000**

13. Here, we can see an inverse relation between the radius of the planet and the gravity. The more the radius (and bigger the planet), the lesser would be Gravity.
14. But then, we also see a direct relation between the mass of the planet and the gravity. The more the mass of the planet, the more will be the gravity.
    Fact - Our Earth's gravity is **9.8 m/s,** and we as humans are accustomed to it.
    In order for us to exist on any other planet, the gravity should be close to what we have here.
    Mars has a gravity of ***3.711 m/s*** and ***Moon*** has a gravity of ***1.62 m/s***.
15. Given what we have just learned, let's try to plot a scatter plot for all the planets, where we will keep the mass of the planet as the Y-Coordinate, the Radius of the Planet as X-Coordinate, and the size of the blob as the gravity of it.

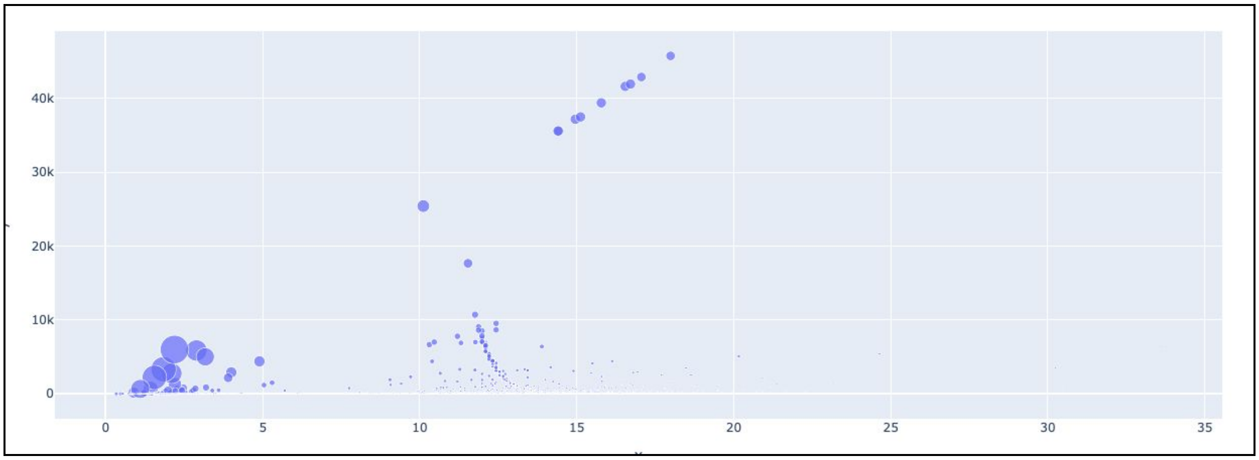
```
planet_masses = []
```

```
planet_radiuses = []
planet_names = []
for planet_data in planet_data_rows:
 planet_masses.append(planet_data[3])
 planet_radiuses.append(planet_data[7])
 planet_names.append(planet_data[1])
planet_gravity = []
for index, name in enumerate(planet_names):
 gravity = (float(planet_masses[index])*5.972e+24) /
(float(planet_radiuses[index])*float(planet_radiuses[index])*6371000*63710
00) * 6.674e-11
 planet_gravity.append(gravity)

fig = px.scatter(x=planet_radiuses, y=planet_masses, size=planet_gravity,
hover_data=[planet_names])
fig.show()
```

16. As we can see, there is this one humongous planet plotted on the top right of the chart with name **HD 100546 b**. Let's remove this planet from our main planet_list to have a better plotted chart.

```
temp_planet_data_rows = list(planet_data_rows)
for planet_data in temp_planet_data_rows:
 if planet_data[1].lower() == "hd 100546 b":
   planet_data_rows.remove(planet_data)
```



Fun Fact - Our standing human bodies can withstand a gravitational force 90 times stronger than earth!

17. Although that is going to be a bit extreme, we can still survive at 10 times the gravity we have at Earth. Let's list down all the names of the planets that have Gravity of 100 or less!

```
low_gravity_planets = []
for index, gravity in enumerate(planet_gravity):
 if gravity < 100:
   low_gravity_planets.append(planet_data_rows[index])

print(len(low_gravity_planets))
```

18. This will give us 3,951 planets. Let's see how many are there with Gravity less than 10?

```
low_gravity_planets = []
for index, gravity in enumerate(planet_gravity):
 if gravity < 10:
   low_gravity_planets.append(planet_data_rows[index])

print(len(low_gravity_planets))
```

19. 1,012 planets! Amazing.

What’s NEXT?

In the next class, we will dive a little deeper into our data try to compare their attributes to that of Earth’s.

EXTEND YOUR KNOWLEDGE:

You can read the following blog on data processing to understand more:
https://exoplanets.nasa.gov/news/?page=0&per_page=40&order=publish_date+desc%2C+created_at+desc&search=