Over the Moon - A Microsoft Data Space Odissey (Part 1)

Pablo de la Asunción Cumbrera Conde

Introducción

Las rocas lunares son un componente esencial de nuestra odisea científica del descubrimiento y comprensión del universo. Sobre ellas reposan, pacientemente, multitud de respuestas, pinceladas sobre como los planetas y las estrellas han sido formadas, como preguntas que guían nuestra forma de enfocar la exploración del espacio.

El desafío para el departamento que se encuentra realizando la investigación de estas rocas es asegurarse de que hay suficientes muestras de cada tipo de roca disponible para poder continuar encontrando respuestas. Por tanto, ¿qué dificultades enfrentamos?

- La dificultad de la recolección de las diferentes piezas.
- La dificultad de que los astronautas identifiquen las muestras necesarias.

Dado este escenario, en el desarrollo del siguiente documento, tomaremos el rol de un científico de la NASA, que tomando como referencia las muestras recolectadas por la misión espacial 'Apolo 11' realizará una orientación sobre la recolección de minerales para los astronautas del programa espacial 'Artemis 2024' en dirección Marte.

Intentaremos dar respuesta a las siguientes cuestiones:

- ¿Cualés son los tipos de roca lunar cuyas existencias se nos están agotando?
- De estas rocas, ¿cuántas muestras de cada tipo pueden traer los astronautas en la misión 'Artemis' para continuar la investigación?

Para ello, dividiremos el documento en las siguientes partes:

- Visualización básica y limpieza de datos
- Preparando la misión
- Insights finales

Visualización básica y limpieza de datos

In [2]:
```
import pandas as pd
```

In [3]:
```
rock_data_path = ('file:///C:/Users/pablo/OneDrive/Escritorio/Data/Over%20the%20moon%20-%20Microsoft/rocksamples.csv')
rock_data_path
```

Out[3]: 'file:///C:/Users/pablo/OneDrive/Escritorio/Data/Over%20the%20moon%20-%20Microsoft/rocksamples.csv'

In [4]:
```
rock_samples = pd.read_csv(rock_data_path)
rock_samples.head()
```

Out[4]:

	ID	Mission	Type	Subtype	Weight (g)	Pristine (%)
0	10001	Apollo11	Soil	Unsieved	125.8	88.36
1	10002	Apollo11	Soil	Unsieved	5629.0	93.73
2	10003	Apollo11	Basalt	Ilmenite	213.0	65.56
3	10004	Apollo11	Core	Unsieved	44.8	71.76
4	10005	Apollo11	Core	Unsieved	53.4	40.31

Como podemos observar, en la tabla encontramos los siguientes elementos:

- ID: Identificador de la misión
- Mission: Misión encargada de recoger la muestra
- Type: Tipo de roca
- Subtype: Clasificación tipo de roca más específico
- Weight(g): Peso original de la muestra, en gramos.
- Pristine(%): El porcentaje de la muestra que queda trás la investigación

In [5]:
```
rock_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2229 entries, 0 to 2228
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   ID            2229 non-null   int64
 1   Mission       2229 non-null   object
 2   Type          2229 non-null   object
 3   Subtype       2226 non-null   object
 4   Weight (g)    2229 non-null   float64
 5   Pristine (%)  2229 non-null   float64
dtypes: float64(2), int64(1), object(3)
memory usage: 104.6+ KB
```

Realizamos una homogeneización de las unidades de medida de la muestra, ya que las rocas se miden en gramos, y el cohete, en kilos.

In [6]:
```
rock_samples['Weight (g)'] = rock_samples['Weight (g)'].apply(lambda x : x * 0.001)
rock_samples.rename(columns={'Weight (g)':'Weight (kg)'}, inplace=True)
rock_samples.head()
```

Out[6]:

	ID	Mission	Type	Subtype	Weight (kg)	Pristine (%)
0	10001	Apollo11	Soil	Unsieved	0.1258	88.36
1	10002	Apollo11	Soil	Unsieved	5.6290	93.73
2	10003	Apollo11	Basalt	Ilmenite	0.2130	65.56
3	10004	Apollo11	Core	Unsieved	0.0448	71.76
4	10005	Apollo11	Core	Unsieved	0.0534	40.31

Diseñamos un nuevo set de datos sobre las seis misiones del 'Apolo 11' que trajeron muestras.

In [7]:
```
missions_data = pd.DataFrame()
missions_data ['Mission'] = rock_samples['Mission'].unique()
missions_data
```

Out[7]:

	Mission
0	Apollo11
1	Apollo12
2	Apollo14
3	Apollo15
4	Apollo16
5	Apollo17

Añadimos los datos referentes a la recogida de muestras a nuestro nuevo set de datos

In [8]:
```
sample_total_weight = rock_samples.groupby('Mission')['Weight (kg)'].sum()
missions_data = pd.merge (missions_data, sample_total_weight, on='Mission')
missions_data.rename(columns={'Weight (kg)':'Sample weight (kg)'}, inplace=True)
missions_data
```

Out[8]:

	Mission	Sample weight (kg)
0	Apollo11	21.55424
1	Apollo12	34.34238
2	Apollo14	41.83363
3	Apollo15	75.39910
4	Apollo16	92.46262
5	Apollo17	109.44402

Comprobamos que estén todos los datos del set de datos anterior

In [9]:
```
sample_total_weight = rock_samples.groupby('Mission')['Weight (kg)'].sum()
sample_total_weight
```

Out[9]:
```
Mission
Apollo11     21.55424
Apollo12     34.34238
Apollo14     41.83363
Apollo15     75.39910
Apollo16     92.46262
Apollo17    109.44402
Name: Weight (kg), dtype: float64
```

Visualizamos las diferencias de peso entre las recogidas de cada misión y hacemos de estos resultados una nueva columna

In [10]:
```
missions_data['Weight diff'] = missions_data['Sample weight (kg)'].diff()
missions_data
```

Out[10]:

	Mission	Sample weight (kg)	Weight diff
0	Apollo11	21.55424	NaN
1	Apollo12	34.34238	12.78814
2	Apollo14	41.83363	7.49125
3	Apollo15	75.39910	33.56547
4	Apollo16	92.46262	17.06352
5	Apollo17	109.44402	16.98140

In [11]:
```
missions_data['Weight diff'] = missions_data['Weight diff'].fillna(value=0)
missions_data
```

Out[11]:

	Mission	Sample weight (kg)	Weight diff
0	Apollo11	21.55424	0.00000
1	Apollo12	34.34238	12.78814
2	Apollo14	41.83363	7.49125
3	Apollo15	75.39910	33.56547
4	Apollo16	92.46262	17.06352
5	Apollo17	109.44402	16.98140

Posteriormente, para introducir la información sobre el cochete, me gustaría señalar que en la misión 'Apolo 11' y 'Saturn V' se requieren dos importantes módulos para el aterrizaje lunar:

- Módulo de Comando (CM): Donde viven los astronautas. Dos bajan a la superficie lunar, el tercero permanece en ella.
- Módulo Lunar (LM): Estte módulo se separa tras alcanzar la órbita de la luna y puede trasnportar dos astronautas.

Añadiremos estos datos accesibles y públicos a nuestro set.

Fuente de datos: NASA - (https://nssdc.gsfc.nasa.gov/nmc/SpacecraftQuery.jsp)

In [12]:
```
missions_data['Lunar module (LM)'] = {'Eagle (LM-5)', 'Intrepid (LM-6)', 'Antares (LM-8)', 'Falcon (LM-10)', 'Orion (LM-11)', 'Challenger (LM-12)'}
missions_data['LM mass (kg)'] = {15103, 15235, 15264, 16430, 16445, 16456}
missions_data['LM mass diff'] = missions_data['LM mass (kg)'].diff()
missions_data['LM mass diff'] = missions_data['LM mass diff'].fillna(value=0)

missions_data['Command module (CM)'] = {'Columbia (CSM-107)', 'Yankee Clipper (CM-108)', 'Kitty Hawk (CM-110)', 'Endeavor (CM-112)', 'Casper (CM-113)', 'America (CM-114)'}
missions_data['CM mass (kg)'] = {5560, 5609, 5758, 5875, 5840, 5960}
missions_data['CM mass diff'] = missions_data['CM mass (kg)'].diff()
missions_data['CM mass diff'] = missions_data['CM mass diff'].fillna(value=0)

missions_data
```

Out[12]:

	Mission	Sample weight (kg)	Weight diff	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15264	0.0	Columbia (CSM-107)	5960	0.0
1	Apollo12	34.34238	12.78814	Orion (LM-11)	15235	-29.0	Casper (CM-113)	5609	-351.0
2	Apollo14	41.83363	7.49125	Antares (LM-8)	16456	1221.0	Yankee Clipper (CM-108)	5840	231.0
3	Apollo15	75.39910	33.56547	Intrepid (LM-6)	16430	-26.0	America (CM-114)	5875	35.0
4	Apollo16	92.46262	17.06352	Falcon (LM-10)	16445	15.0	Endeavor (CM-112)	5560	-315.0
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	15103	-1342.0	Kitty Hawk (CM-110)	5758	198.0

Añadimos también columnas de peso total

In [13]:
```
missions_data['Total weight (kg)'] = missions_data['LM mass (kg)'] + missions_data['CM mass (kg)']
missions_data['Total weight diff'] = missions_data['LM mass diff'] + missions_data['CM mass diff']
missions_data
```

Out[13]:

	Mission	Sample weight (kg)	Weight diff	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff	Total weight (kg)	Total weight diff
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15264	0.0	Columbia (CSM-107)	5960	0.0	21224	0.0
1	Apollo12	34.34238	12.78814	Orion (LM-11)	15235	-29.0	Casper (CM-113)	5609	-351.0	20844	-380.0
2	Apollo14	41.83363	7.49125	Antares (LM-8)	16456	1221.0	Yankee Clipper (CM-108)	5840	231.0	22296	1452.0
3	Apollo15	75.39910	33.56547	Intrepid (LM-6)	16430	-26.0	America (CM-114)	5875	35.0	22305	9.0
4	Apollo16	92.46262	17.06352	Falcon (LM-10)	16445	15.0	Endeavor (CM-112)	5560	-315.0	22005	-300.0
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	15103	-1342.0	Kitty Hawk (CM-110)	5758	198.0	20861	-1144.0

Para responder a nuestra primera cuestión, es necesario saber la capacidad de carga que podrá soportar al cruzar la órbita (concepto 'payload') en base a otras misiones espaciales ('Saturn V' - 43,500 kg)

Fuente de datos: NASA - (https://www.nasa.gov/sites/default/files/atoms/files/0080_sls_fact_sheet_sept2020_09082020_final_0.pdf)

In [14]:
```
# Sample-to-weight ratio

saturnVPayload = 43500

missions_data['Crewed area : Payload'] = missions_data['Total weight (kg)'] / saturnVPayload
missions_data['Sample : Crewed area'] = missions_data['Sample weight (kg)'] / missions_data['Total weight (kg)']
missions_data['Sample : Payload'] = missions_data['Sample weight (kg)'] / saturnVPayload

missions_data
```

Out[14]:

	Mission	Sample weight (kg)	Weight diff	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff	Total weight (kg)	Total weight diff	Crewed area : Payload	Sample : Crewed area	Sample : Payload
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15264	0.0	Columbia (CSM-107)	5960	0.0	21224	0.0	0.487908	0.001016	0.000495
1	Apollo12	34.34238	12.78814	Orion (LM-11)	15235	-29.0	Casper (CM-113)	5609	-351.0	20844	-380.0	0.479172	0.001648	0.000789
2	Apollo14	41.83363	7.49125	Antares (LM-8)	16456	1221.0	Yankee Clipper (CM-108)	5840	231.0	22296	1452.0	0.512552	0.001876	0.000962
3	Apollo15	75.39910	33.56547	Intrepid (LM-6)	16430	-26.0	America (CM-114)	5875	35.0	22305	9.0	0.512759	0.003380	0.001733
4	Apollo16	92.46262	17.06352	Falcon (LM-10)	16445	15.0	Endeavor (CM-112)	5560	-315.0	22005	-300.0	0.505862	0.004202	0.002126
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	15103	-1342.0	Kitty Hawk (CM-110)	5758	198.0	20861	-1144.0	0.479563	0.005246	0.002516

Guardamos la media de estos ratios a lo largo de las diferentes misiones.

In [15]:
```
crewedArea_payload_ratio = missions_data['Crewed area : Payload'].mean()
sample_crewedArea_ratio = missions_data['Sample : Crewed area'].mean()
sample_payload_ratio = missions_data['Sample : Payload'].mean()

print(crewedArea_payload_ratio)
print(sample_crewedArea_ratio)
print(sample_payload_ratio)
```

```
0.49630268199233724
0.0028946732226251396
0.0014369195019157093
```

Preparando la misión

Desconocemos aún muchos detalles de la misión espacial 'Artemis' pero sabemos que cada misión realizara tres iteraciones del cohete. Cada cohete tendrá una version para el mantenimiento de la tripulación y otra exclusiva para carga. Para poder realizar predicciones sobre esta misión a partir de los datos del 'Apolo 11' debemos enfocarnos en el proyecto más parecido posible: en el vehículo espacial destinados a albergar la tripulación.

In [16]:
```
artemis_crewedArea = 26520
artemis_mission = pd.DataFrame({'Mission':['Artemis1','Artemis1b','Artemis2'],
                                 'Total weight (kg)':[artemis_crewedArea,artemis_crewedArea,artemis_crewedArea],
                                 'Payload (kg)':[26988, 37965, 42955]})

# Añadimos los ratios anteriores

artemis_mission['Sample weight from total (kg)'] = artemis_mission['Total weight (kg)'] * sample_crewedArea_ratio
artemis_mission['Sample weight from payload (kg)'] = artemis_mission['Payload (kg)'] * sample_payload_ratio

#Y calculamos el promedio de ambos

artemis_mission['Estimated sample weight (kg)'] = (artemis_mission['Sample weight from payload (kg)'] + artemis_mission['Sample weight from total (kg)'])/2

artemis_mission
```

Out[16]:

	Mission	Total weight (kg)	Payload (kg)	Sample weight from total (kg)	Sample weight from payload (kg)	Estimated sample weight (kg)
0	Artemis1	26520	26988	76.766734	38.779584	57.773159
1	Artemis1b	26520	37965	76.766734	54.552649	65.659691
2	Artemis2	26520	42955	76.766734	61.722877	69.244806

Con estos datos ya podemos saber cuanta carga es capaz de transportar 'Artemis' en cada misión: 57.77 kg, 65.65 kg y 69.24 kg.

Ahora la pregunta es: ¿qué recogida de tipos de roca debemos priorizar? - Para ello primero nos basaremos en los datos de las misiones 'Apolo 11' para saber cuánta muestra de cada roca lunar trajeron.

In [19]:
```
rock_samples ['Remaining (kg)']=rock_samples['Weight (kg)']*(rock_samples['Pristine (%)']*0.1)
rock_samples.head()
```

Out[19]:

	ID	Mission	Type	Subtype	Weight (kg)	Pristine (%)	Remaining (kg)
0	10001	Apollo11	Soil	Unsieved	0.1258	88.36	1.111569
1	10002	Apollo11	Soil	Unsieved	5.6290	93.73	52.760617
2	10003	Apollo11	Basalt	Ilmenite	0.2130	65.56	1.396428
3	10004	Apollo11	Core	Unsieved	0.0448	71.76	0.321485
4	10005	Apollo11	Core	Unsieved	0.0534	40.31	0.215255

In [20]:
```
rock_samples.describe()
```

Out[20]:

	ID	Weight (kg)	Pristine (%)	Remaining (kg)
count	2229.000000	2229.000000	2229.000000	2229.000000
mean	52058.432032	0.168253	84.512764	1.381034
std	26207.651471	0.637286	22.057299	5.259540
min	10001.000000	0.000000	0.000000	0.000000
25%	15437.000000	0.003000	80.010000	0.024323
50%	65527.000000	0.010200	92.300000	0.085300
75%	72142.000000	0.093490	98.140000	0.782400
max	79537.000000	11.729000	180.000000	111.695267

Es necesario usar la función `describe` para poder tener una vista general de los datos de alrededor de 2000 muestras. Sobre estos datos buscaremos cual de las muestras está agotando existencias debido a un alto uso para investigación.

In [21]:
```
low_samples = rock_samples.loc[(rock_samples['Weight (kg)'] >= .16) & (rock_samples['Pristine (%)'] <= 50)]
low_samples.head()
```

Out[21]:

	ID	Mission	Type	Subtype	Weight (kg)	Pristine (%)	Remaining (kg)
11	10017	Apollo11	Basalt	Ilmenite	0.973	43.71	4.252983
14	10020	Apollo11	Basalt	Ilmenite	0.425	27.88	1.184900
15	10021	Apollo11	Breccia	Regolith	0.250	30.21	0.755250
29	10045	Apollo11	Basalt	Olivine	0.185	12.13	0.224405
37	10057	Apollo11	Basalt	Ilmenite	0.919	35.15	3.230285

Ahora buscamos las coincidencias entre los tipos de roca lunar entre los dos conjuntos de datos con la función `Type.unique`

In [25]:
```
rock_samples.Type.unique()
```

Out[25]:
```
array(['Soil', 'Basalt', 'Core', 'Breccia', 'Special', 'Crustal'],
      dtype=object)
```

In [26]:
```
low_samples.Type.unique()
```

Out[26]:
```
array(['Basalt', 'Breccia', 'Soil', 'Core'], dtype=object)
```

In [27]:
```
low_samples.groupby('Type')['Weight (kg)'].count()
```

Out[27]:
```
Type
Basalt     14
Breccia     8
Core        1
Soil        4
Name: Weight (kg), dtype: int64
```

Creamos un nuevo conjunto de datos incluyendo todos estos nuevos insights.

In [30]:
```
needed_samples = low_samples[low_samples['Type'].isin(['Basalt', 'Breccia'])]
needed_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22 entries, 11 to 2183
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   ID              22 non-null     int64
 1   Mission         22 non-null     object
 2   Type            22 non-null     object
 3   Subtype         22 non-null     object
 4   Weight (kg)     22 non-null     float64
 5   Pristine (%)    22 non-null     float64
 6   Remaining (kg)  22 non-null     float64
dtypes: float64(3), int64(1), object(3)
memory usage: 1.4+ KB
```

Estos datos nos indican que necesitamos traer piedras del tipo 'Basalt' y 'Breccia' (los valores más altos en la lista que indica bajo en reservas). Pero, ¿es esto solo lo que debemos buscar? Volveremos con nuestro neuvo conocimiento al set de datos original y realizaremos algunas comprobaciones...

In [28]:
```
rock_samples.groupby('Type')['Weight (kg)'].sum()
```

Out[28]:
```
Type
Basalt      93.14077
Breccia    168.88075
Core        19.93587
Crustal      4.74469
Soil        87.58981
Special      0.74410
Name: Weight (kg), dtype: float64
```

Podemos observar que hay dos valores que nos pueden llamar la atención, el tipo 'Special' (cuya naturaleza de la etiqueta SUPONGO viene del hecho de encontrar valores o tipos inesperados en otras muestras de roca) o el tipo 'Crustal' del que nunca hicimos una gran recolección, por lo que incluiremos en nuestra búsqueda.

In [31]:
```
needed_samples = needed_samples.append(rock_samples.loc[rock_samples['Type'] == 'Crustal'])
needed_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 68 entries, 11 to 2189
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   ID              68 non-null     int64
 1   Mission         68 non-null     object
 2   Type            68 non-null     object
 3   Subtype         68 non-null     object
 4   Weight (kg)     68 non-null     float64
 5   Pristine (%)    68 non-null     float64
 6   Remaining (kg)  68 non-null     float64
dtypes: float64(3), int64(1), object(3)
memory usage: 4.2+ KB
```

Insights finales

Sumario de las rocas necesarias:

In [32]:
```
needed_samples_overview = pd.DataFrame()
needed_samples_overview['Type'] = needed_samples.Type.unique()
needed_samples_overview
```

Out[32]:

	Type
0	Basalt
1	Breccia
2	Crustal

In [33]:
```
needed_sample_weights = needed_samples.groupby('Type')['Weight (kg)'].sum().reset_index()
needed_samples_overview = pd.merge(needed_samples_overview, needed_sample_weights, on='Type')
needed_samples_overview.rename(columns={'Weight (kg)':'Total weight (kg)'}, inplace=True)
needed_samples_overview
```

Out[33]:

	Type	Total weight (kg)
0	Basalt	17.42340
1	Breccia	10.11850
2	Crustal	4.74469

In [34]:
```
needed_sample_ave_weights = needed_samples.groupby('Type')['Weight (kg)'].mean().reset_index()
needed_samples_overview = pd.merge(needed_samples_overview, needed_sample_ave_weights, on='Type')
needed_samples_overview.rename(columns={'Weight (kg)':'Average weight (kg)'}, inplace=True)
needed_samples_overview
```

Out[34]:

	Type	Total weight (kg)	Average weight (kg)
0	Basalt	17.42340	1.244529
1	Breccia	10.11850	1.264812
2	Crustal	4.74469	0.103145

Para poder lograr nuestro objetivo de traer un número de rocas lunares de cada tipo de vuelta a casa, debemos calcular cual es la proporción y dificultad de entonctrar cada tipo de roca.

In [35]:
```
total_rock_count = rock_samples.groupby('Type')['ID'].count().reset_index()
needed_samples_overview = pd.merge(needed_samples_overview, total_rock_count, on='Type')
needed_samples_overview.rename(columns={'ID':'Number of samples'}, inplace=True)
total_rocks = needed_samples_overview['Number of samples'].sum()
needed_samples_overview['Percentage of rocks'] = needed_samples_overview['Number of samples'] / total_rocks
needed_samples_overview
```

Out[35]:

	Type	Total weight (kg)	Average weight (kg)	Number of samples	Percentage of rocks
0	Basalt	17.42340	1.244529	351	0.258850
1	Breccia	10.11850	1.264812	959	0.707227
2	Crustal	4.74469	0.103145	46	0.033923

Para finalizar, como recomendación al programa 'Artemis'determinaremos cual es el peso medio que podrán transportar acorde a nuestras estimaciones.

In [36]:
```
artemis_ave_weight = artemis_mission['Estimated sample weight (kg)'].mean()
artemis_ave_weight
```

Out[36]: 64.22588520079607

Y utilizamos estos datos para dar instrucciones a los astronautas de cuantas rocas luanares de cada tipo deben recoger.

In [38]:
```
needed_samples_overview['Weight to collect'] = needed_samples_overview['Percentage of rocks'] * artemis_ave_weight
needed_samples_overview['Rocks to collect'] = needed_samples_overview['Weight to collect'] / needed_samples_overview['Average weight (kg)']
needed_samples_overview
```

Out[38]:

	Type	Total weight (kg)	Average weight (kg)	Number of samples	Percentage of rocks	Weight to collect	Rocks to collect
0	Basalt	17.42340	1.244529	351	0.258850	16.624842	13.358345
1	Breccia	10.11850	1.264812	959	0.707227	45.422289	35.912271
2	Crustal	4.74469	0.103145	46	0.033923	2.178754	21.123128

Los astronautas deberían coger 13 rocas lunares de tipo 'Basalt', 25 de 'Breccia' y 21 de 'Crustal'.