

데이터공학_기승전결_보고서_1팀_강성준_김재엽

1. 소개

2. 관련연구

2.1 Curiosity Rover

2.2 Mars Crater

2.3 Solar System

3. 제안내용

3.1 화성 크레이터 데이터에 대한 가설

3.2 화성 대기 데이터에 대한 가설

3.3 태양계 데이터에 대한 가설

4. 실험

4-1. 실험 데이터



사용 데이터



화성 대기질 데이터



화성 크레이터 데이터



태양계 데이터

4-2. 실험 방법



사용한 언어 및 라이브러리



Jupyter-notebook



실험 데이터 실험 방법

4-3. 실험 평가



화성 크레이터 데이터



화성 대기질 데이터



태양계 데이터

5. 토의

5-1. 가설 평가

5-2. 화성 크레이터 지도 만들기

6. 결론

1. 소개



데이터 분석 목적



최근 화성에 대한 space X와 같은 기관의 활동으로
국제적 관심이 매우 높아지고 있지만
일반인들은 화성에 대해 관심이 많지 않음



이를 통해 랭체인 기반 화성 가이드 챗봇을 개발하여
다양한 사람들이 화성에 대해 배울 수 있도록 돕고자 함!



소개

- 최근 일론 머스크에 의해 화성에 대한 관심이 국제적으로 늘어나고 있지만 일반인에게는 아직 닿지 못함 → 일반인에게도 화성에 대해 쉽게 관심을 가질 수 있도록 화성 가이드 챗봇을 개발하기 위한 데이터 분석을 진행함
- 이때 화성에 대한 데이터로 화성 대기질 데이터, 화성 분화구 데이터를 사용하여 화성의 특징을 분석하고 소개할 수 있도록 함

2. 관련연구

2.1 Curiosity Rover

화성 대기질 데이터

<논문 1>

화성과학연구소 REMS 압력 센서의 중력파 관측 및 MarsWRF를 이용한 중규모 대기 모델링과의 비교

- 논문 출처 : <https://ntrs.nasa.gov/citations/20210022894>
- 화성 표면의 기압을 이용하여 다양한 대기 현상을 분석한 논문
- 8년간의 데이터로 대기 압력의 파동을 식별하고 파동의 주기로 계절을 파악 함
- 2018년 화성의 글로벌 먼지 폭풍 동안 가장 강한 파동이 관찰되었다고 결론을 내림

<논문 2>

화성 대기질 데이터 - REMS: The Environmental Sensor Suite for the Mars Science Laboratory Rover

- 논문 출처 : <https://link.springer.com/article/10.1007/s11214-012-9921-1>
- 화성 표면의 거주 가능성에 직접적으로 관련된 세 가지 주요 환경 요소를 조사 함
- 주로 REMS (Rover Environmental Monitoring Station)의 역할을 설명하는 논문
- REMS가 측정한 요소와 센서 위치, 데이터 수집을 어떻게 진행했는지 나와있음
- 180분동안 모든 센서의 데이터를 동시 수집을 진행했다고 함

2.2 Mars Crater

화성 크레이터 데이터

<논문 1>

1km 크기의 화성 충돌 분화구에 대한 새로운 글로벌 데이터베이스: 1. 데이터베이스 생성, 속성 및 매개변수

- 논문 출처 :
<https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2011JE003966>
- 여러 위성 및 탐사선이 수집한 화성 글로벌 분화구 데이터베이스를 생성함
 - 화성의 직경 1km 이상의 분화구 384,343개를 포함하는 데이터베이스 생성함
- 데이터베이스에 포함된 분화구의 형태학적 및 형태 계량학적 데이터 분석함
- 분화구의 깊이, 림 높이, 방출물 특성 등 다양한 형태 계량학적 특성 조사함

<논문 2>

1km 이상의 화성 충돌 크레이터에 대한 새로운 글로벌 데이터베이스: 2. 전 세계 크레이터의 특성과 단순-복잡한 전이 직경의 지역적 변화

- 논문 출처 :
<https://agupubs.onlinelibrary.wiley.com/doi/epdf/10.1029/2011JE003967>
- 여러 위성 및 탐사선이 수집한 화성 글로벌 분화구 데이터베이스를 생성함
- 다양한 지형에 걸쳐 단순하고 복잡한 형태에 대한 크레이터 깊이 및 직경 비율 분석 함
- 고위도 지역에서 단순에서 복합으로의 전환이 약 11km 직경에서 발생함을 발견함
- 분화구 형성 중에 녹는 휘발성 물질의 존재가 고위도 분화구의 형태에 중요한 역할 을 함

2.3 Solar System

태양계 데이터

<EDA>

GalaxyGlimpse: Planetary Wisdom

- EDA 출처 : [Kaggle-EDA](#)
- 지구형 행성과 목성형 행성이라는 두 가지 주요 그룹으로 분류함
- 지구형 행성은 암석으로 이루어지고 표면이 단단함
- 목성형 행성은 단단한 표면이 없고 가스로 구성되어 있음

3. 제안내용

3.1 화성 크레이터 데이터에 대한 가설

화성 크레이터

1 화성 크레이터 데이터로 화성 크레이터 맵 만들기

- 위도 경도 데이터로 화성 크레이터 맵 만들기
 - 지도 3개 만들기
 - 층수가 많고 방출물이 많은 크레이터 맵
 - 기본 크레이터 맵
 - 이름이 있고 직경이 100km 이상인 크레이터 맵

2 분화구의 직경이 클수록 깊이도 깊을까?

- 직경이 크다는 것은 큰 운석이 충돌 될 가능성이 있음
 - 큰 운석이랑 충돌 시 당연히 깊이도 깊을 것이다.

3 분화구에 분출물 성분이 많을 수록 층수가 더 많아질 것

- 분화구에 분출물이 많다는 것은 충돌을 자주 했다는 것
 - 층수도 많을 것

3.2 화성 대기 데이터에 대한 가설

화성 대기질

1 화성이랑 지구는 비슷하기 때문에 사계절도 똑같을까?

- 화성에도 우리나라와 비슷한 사계절이 존재한다.
- 화성의 온도는 우리나라의 온도 변화 주기가 비슷하다.

2 인간이 화성에서 살 수 있을까?

- 화성에 인간이 생존할 수 있는 시기가 존재한다.

3 화성의 온도 변화가 큰 이유가 무엇일까?

- 화성의 낮은 기압은 온도 변화의 폭을 크게 만든다.

3.3 태양계 데이터에 대한 가설

태양계 데이터

1 데이터 셋을 pca로 표현하면 지구형 행성과 목성형 행성끼리 그룹으로 묶일까?

- 태양계는 지구형과 목성형으로 나뉘지기 때문에 태양계 행성을 대표하는 데이터를 주성분 분석을 진행하면 분명 지구형과 목성형으로 나뉘지 않을까?

4. 실험

4-1. 실험 데이터

✓ 사용 데이터

Data	Size	Source	DOI
화성 대기질 데이터	13×3198 dataset	NASA - Techinacal	DOI: 10.1029/2021JE006907

		Reports	
화성 크레이터 데이터	10×384344	<u>Stuart J. Robbins</u>	DOI: 10.1029/2011JE003966
태양계 데이터	9×27	Sourav Banerjee	

화성 대기질 데이터

Raw Data

[REMS_Mars_Dataset.csv](#)

Processing Data

[REMS_Mars_Dataset_Cleaned_last.csv](#)

 데이터 출처 : [kaggle](#)

 데이터 논문 : [paper](#)

화성 크레이터 데이터

Raw Data

[Mars Crater info.csv](#)

Processing Data

[Processed_Mars_Crater_info.csv](#)

 데이터 출처 : [kaggle](#)

 데이터 논문: [paper](#)



태양계 데이터



planets_updated.csv

☞ 데이터 출처 : [kaggle](#)

4-2. 실험 방법



사용한 언어 및 라이브러리

Software	Description
python	프로그래밍 언어
numpy	파이썬의 수치 연산 라이브러리
pandas	데이터 조작 및 분석 라이브러리
matplotlib	정적 그래프 생성 라이브러리
plotly.express	인터랙티브 플롯 고수준 인터페이스
seaborn	통계적 데이터 시각화 라이브러리
sklearn	머신 러닝 라이브러리
folium	지도 시각화 라이브러리

→ 사용한 언어 및 라이브러리는 위에 표와 같다. 실험을 진행하기 위해서 데이터마다 파트를 나눠

[Jupyter-notebook](#) 환경에서 실험을 진행했다.



Jupyter-notebook

화성대기질EDA.ipynb

화성크레이터_EDA.ipynb



실험 데이터 실험 방법

1. 화성 대기질 데이터의 실험 방법

- 전처리 진행
 - 타입 변환
 - 컬럼 제거
 - 결측치 처리
- 가설 검증 진행
 - 🤔 화성의 온도 변화가 큰 이유가 무엇일까?
 - 🤔 화성이랑 지구는 비슷하기 때문에 사계절도 똑같을까?
 - 🤔 인간이 화성에서 살 수 있을까?

2. 화성 크레이터 데이터의 실험 방법

- 전처리 진행
 - 타입 변환
 - 결측치 대체
- 가설 검증 진행
 - 🤔 분화구의 직경이 클수록 깊이도 깊을 것..?
 - 🤔 분화구에 분출물 성분이 많을 수록 층수가 더 많아질 것..?
- 화성 크레이터 맵 만들기
 - 😎 분화구 이름이 있고 직경이 100km 이상인 지도
 - 😎 크레이터 전체 맵

3. 태양계 데이터의 실험 방법

- 가설 검증 진행
 - 주성분 분석을 진행하면 지구형 행성과 목성형 행성끼리 그룹으로 묶일까?

4-3. 실험 평가

💿 화성 크레이터 데이터

▼ 데이터 EDA

▼ 전처리 및 데이터 기본 구조 파악

💻 컬럼 확인

Coulmn	Feature_explain
CRATER_ID	분화구의 고유 식별자
CRATER_NAME	분화구의 이름
LATITUDE_CIRCLE_IMAGE	분화구의 중심의 위도
LONGITUDE_CIRCLE_IMAGE	분화구의 중심의 경도
DIAM_CIRCLE_IMAGE	분화구의 직경
DEPTH_RIMFLOOR_TOPOG	분화구의 가운데에서 바닥까지의 깊이
MORPHOLOGY_EJECTA_1	분화구의 방출물의 첫 번째 형태학적 특징
MORPHOLOGY_EJECTA_2	분화구의 방출물의 두 번째 형태학적 특징
MORPHOLOGY_EJECTA_3	분화구의 방출물의 세 번째 형태학적 특징
NUMBER_LAYERS	분화구 층의 수

✓ 필드 값이 없는 행 파악

A	B	C	D	E	F	G	H	I	J
CRATER_ID	CRATER_N	LATITUDE	LONGITU	DIAM_CIR	DEPTH_RI	MORPHOI	MORPHOI	MORPHOI	NUMBER_L
01-000000		84.367	108.746	82.1	0.22				0
01-000001	Korolev	72.76	164.464	82.02	1.97	Rd/MLERS	HuBL		3
01-000002		69.244	-27.24	79.63	0.09				0
01-000003		70.107	160.575	74.81	0.13				0
01-000004		77.996	95.617	73.53	0.11				0
01-000005		68.547	137.849	72.66	0.19				0
01-000006		69.492	169.751	70.11	0.1				0
01-000007		78.716	-87.249	63.57	0.05				0
01-000008		75.539	13.829	58.4	0.11	Rd			0
01-000009		69.371	75.147	55.24	0				0
01-000010		77.707	-27.314	52.37	0.07				0
01-000011		70.319	139.936	51.31	0.13				0
01-000012	Dokka	77.17	-145.681	51.08	1.74	Rd			0

✓ 전처리 진행

1. 필드 값 X → NA (문자열)로 대체

```
df = pd.read_csv('data/Mars Crater info.csv')

df.replace(' ', 'NA', inplace=True)
df.replace('', 'NA', inplace=True)

df.to_csv('data/Processed_Mars_Crater_info.csv', index=False)

print("결측치 및 공백 전처리 완료")

na_counts_df = df.apply(lambda x: (x == 'NA').sum()).reset_index()
na_counts_df.columns = ['Column', 'NA_Count']

print("각 컬럼의 'NA' 값 개수:")
print(na_counts_df)
```

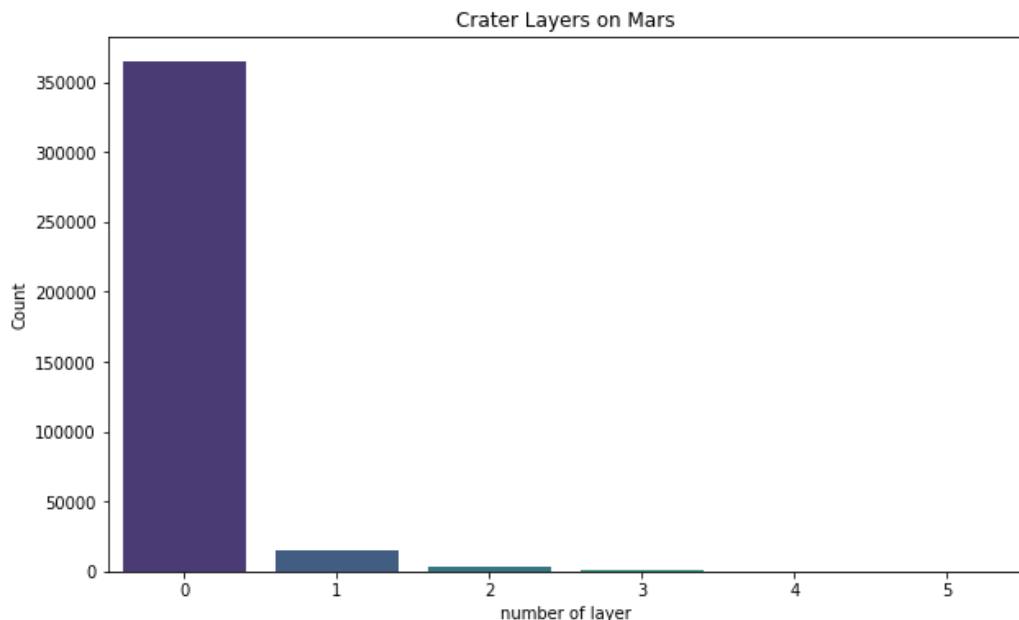
결측치 및 공백 전처리 완료		
각 컬럼의 'NA' 값 개수:		
	Column	NA_Count
0	CRATER_ID	0
1	CRATER_NAME	383356
2	LATITUDE_CIRCLE_IMAGE	0
3	LONGITUDE_CIRCLE_IMAGE	0
4	DIAM_CIRCLE_IMAGE	0
5	DEPTH_RIMFLOOR_TOPOG	0
6	MORPHOLOGY_EJECTA_1	339718
7	MORPHOLOGY_EJECTA_2	364867
8	MORPHOLOGY_EJECTA_3	383050
9	NUMBER_LAYERS	0

- **CRATER_ID**
 - 384344 (행)
 - 01-000000 ~ 30-013690
- **CRATER_NAME**
 - 986개

- NA: 383,356개

-
- NA 없는 컬럼
 - LATITUDE_CIRCLE_IMAGE
 - LONGITUDE_CIRCLE_IMAGE
 - DIAM_CIRCLE_IMAGE
 - DEPTH_RIMFLOOR_TOPOG
 - MORPHOLOGY_EJECTA_1: 분화구의 방출물(ejecta)의 첫 번째 형태학적 특징
 - NA: 339,718개
 - MORPHOLOGY_EJECTA_2: 분화구의 방출물(ejecta)의 두 번째 형태학적 특징
 - NA: 364,867개
 - MORPHOLOGY_EJECTA_3: 분화구의 방출물(ejecta)의 세 번째 형태학적 특징
 - NA: 383,3050개
-

- NUMBER_LAYERS



- 0층 364612개

- 1층 14567개
- 2층 3435개
- 3층 739개
- 4층 85개
- 5층 5개

- 각 직경 구간별 분화구 수 계산

```
# 각 구간별 분화구 수 계산
bins = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, float('inf')]
labels = ['0-10km', '10-20km', '20-30km', '30-40km', '40-50km', '50-60km', '60-70km', '70-80km', '80-90km', '90-100km', '100km & Above']

# 구간별로 직경 분류
df['DIAMETER_BIN'] = pd.cut(df['DIAM_CIRCLE_IMAGE'], bins, labels=labels)

bin_counts = df['DIAMETER_BIN'].value_counts().sort_index()

# 막대 그래프 시각화
plt.figure(figsize=(14, 8))
bar_plot = sns.barplot(x=bin_counts.index, y=bin_counts.values)

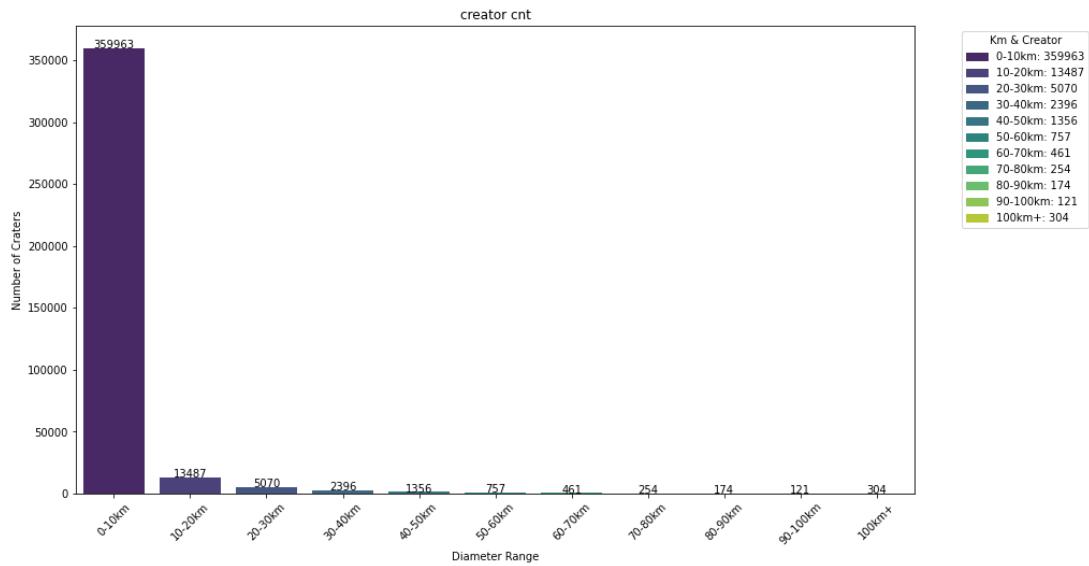
plt.title('creator cnt')
plt.xlabel('Diameter Range')
plt.ylabel('Number of Craters')
plt.xticks(rotation=45)

for index, value in enumerate(bin_counts.values):
    bar_plot.text(index, value + 2, str(value), color='black')

handles = [plt.Rectangle((0,0),1,1, color=bar_plot.patches[i].get_color()) for i in range(len(bin_counts))]
labels = [f'{labels[i]}: {bin_counts.values[i]}' for i in range(len(bin_counts))]

plt.legend(handles, labels, title='Km & Creator', bbox_to_anchor=(1.05, 1))

plt.show()
```



- 각 직경 구간별 이름만 있는 분화구 수 계산

```

# 분화구 이름이 있는 행만 선택
df = df[df['CRATER_NAME'].notna() & df['CRATER_NAME'].str.contains('name')]

# 직경 구간 설정
bins = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, float('inf')]
labels = ['0-10km', '10-20km', '20-30km', '30-40km', '40-50km', '50-60km', '60-70km', '70-80km', '80-90km', '90-100km', '100km+']

# 구간별로 직경 분류
df['DIAMETER_BIN'] = pd.cut(df['DIAM_CIRCLE_IMAGE'], bins, labels=labels)

# 각 구간별 분화구 수 계산
bin_counts = df['DIAMETER_BIN'].value_counts().sort_index()

plt.figure(figsize=(14, 8))
bar_plot = sns.barplot(x=bin_counts.index, y=bin_counts.values)

plt.title('name with diameter')
plt.xlabel('Diameter Range')
plt.ylabel('Number of Craters')
plt.xticks(rotation=45)

```

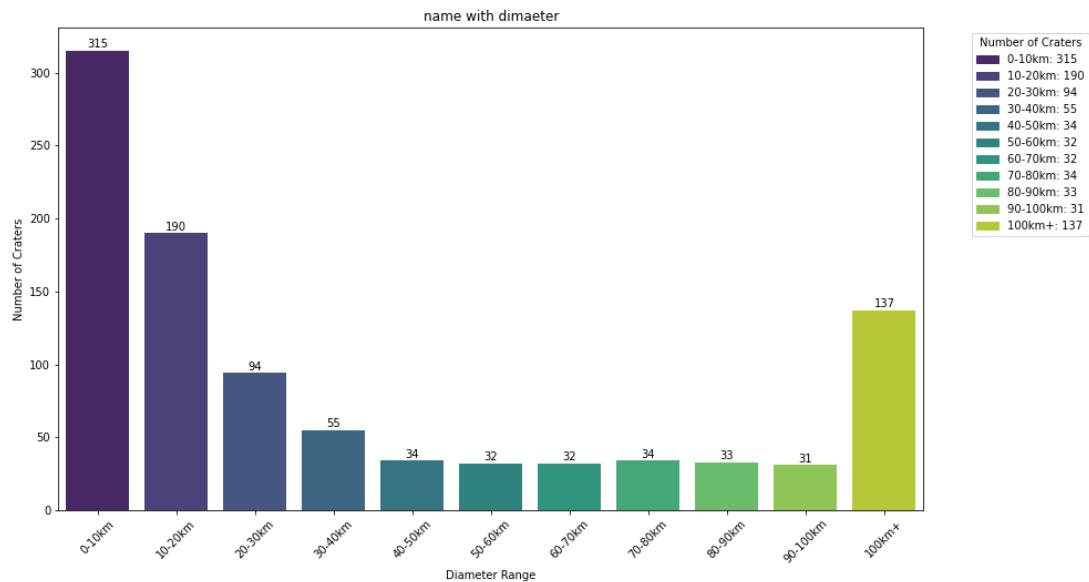
```

for index, value in enumerate(bin_counts.values):
    bar_plot.text(index, value + 2, str(value), color='white')

handles = [plt.Rectangle((0,0),1,1, color=bar_plot.patches[i].get_color())
           for i in range(len(bin_counts))]
labels = [f'{labels[i]}: {bin_counts.values[i]}' for i in range(len(bin_counts))]
plt.legend(handles, labels, title='Number of Craters',
           loc='upper right', frameon=False)

plt.show()

```



▼ 크레이터 수치형 컬럼 데이터 표준편차, 평균, 분산, 정규분포 파악

✓ 수치형 컬럼

Coulmn	Feature_explain
LATITUDE_CIRCLE_IMAGE	분화구의 중심의 위도
LONGITUDE_CIRCLE_IMAGE	분화구의 중심의 경도
DIAM_CIRCLE_IMAGE	분화구의 직경
DEPTH_RIMFLOOR_TOPOG	분화구의 가운데에서 바닥까지의 깊이

```

import seaborn as sns
from scipy.stats import norm

```

```

# 수치형 컬럼으로 표준편차, 평균, 분산, 정규분포를 계산

numeric_cols = ['LATITUDE_CIRCLE_IMAGE', 'LONGITUDE_CIRCLE_IMAGE']
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric)

# 계산
stats_df = df[numeric_cols].describe().transpose()
stats_df['variance'] = df[numeric_cols].var()
stats_df['std_dev'] = df[numeric_cols].std()

# df
print("Statistics DataFrame:")
print(stats_df)

# 그래프
for col in numeric_cols:
    plt.figure(figsize=(10, 6))
    sns.histplot(df[col], kde=False, stat="density", line_kde=True)

    mean = df[col].mean()
    std = df[col].std()
    xmin, xmax = plt.xlim()
    x = np.linspace(xmin, xmax, 100)
    p = norm.pdf(x, mean, std)

    plt.plot(x, p, 'k', linewidth=2, label=f'Normal distribution fit')

    title = f"Distribution of {col} with Normal Fit"
    plt.title(title)
    plt.xlabel(col)
    plt.ylabel('Density')
    plt.legend()

plt.show()

```

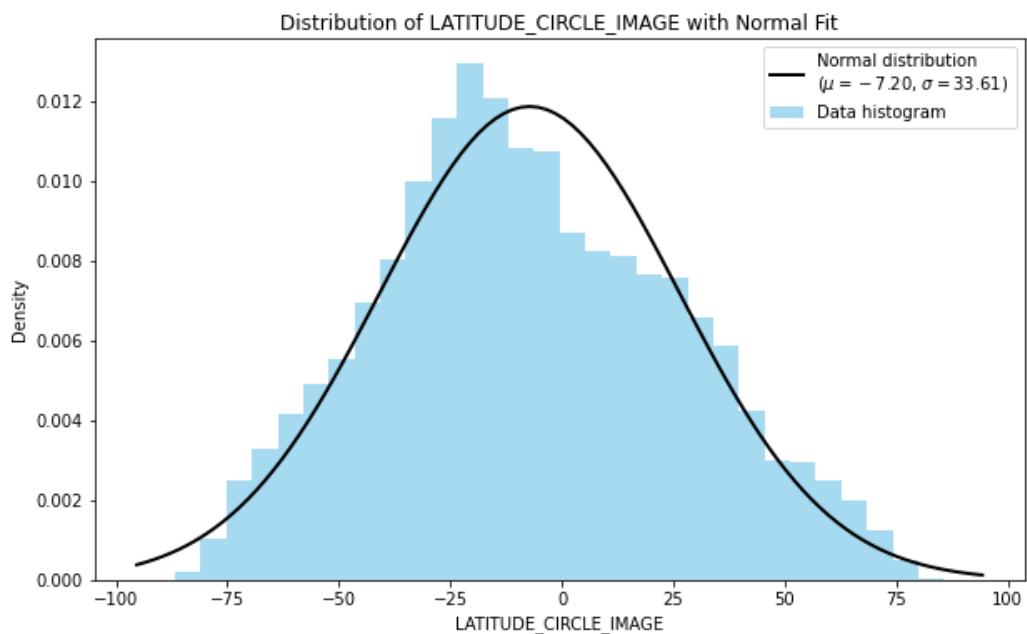
Statistics DataFrame:						
	count	mean	std	min	25%	\
LATITUDE_CIRCLE_IMAGE	384343.0	-7.199209	33.608966	-86.700	-30.935	
LONGITUDE_CIRCLE_IMAGE	384343.0	10.128021	96.641592	-179.997	-58.829	
DIAM_CIRCLE_IMAGE	384343.0	3.556686	8.591993	1.000	1.180	
DEPTH_RIMFLOOR_TOPOG	384343.0	0.075838	0.221518	-0.420	0.000	
	50%	75%	max	variance	std_dev	\
LATITUDE_CIRCLE_IMAGE	-10.079	17.2225	85.702	1129.562623	33.608966	
LONGITUDE_CIRCLE_IMAGE	12.740	89.2755	179.997	9339.597376	96.641592	
DIAM_CIRCLE_IMAGE	1.530	2.5500	1164.220	73.822340	8.591993	
DEPTH_RIMFLOOR_TOPOG	0.000	0.0000	4.950	0.049070	0.221518	

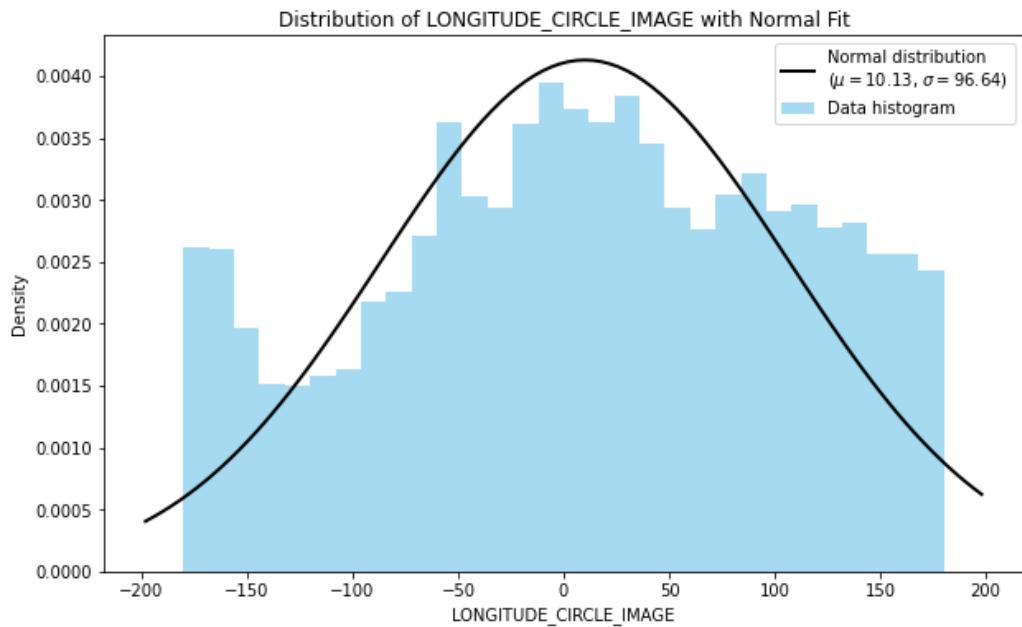


분화구의 지리적 분포 (위도와 경도)

Mean&Std

- Latitude Mean: -7.20, Longitude Mean: 10.13
 - 평균적으로 분화구는 화성의 적도 근처와 동쪽 반구에 분포함
 - 평균 위도가 -7.20이라는 점에서 분화구가 남반구에 더 많이 분포함
- Latitude Std: 33.61, Longitude Std: 96.64





- 위도의 표준편차는 33.61, 경도의 표준편차는 96.64
 - ➡ 분화구가 위도와 경도 모두에서 넓게 퍼져 있음을 나타냄
 - ➡ 위도는 히스토그램과 정규분포 곡선이 거의 일치함
 - ➡ 경도는 약간 불규칙 전체적으로는 데이터가 정규분포를 잘 따름

✓ MinMax

- Latitude Min: -86.70, Longitude Min: -179.99
 - 가장 남쪽에 위치한 분화구의 위도는 -86.70
 - 가장 서쪽에 위치한 분화구의 경도는 -179.99
- Latitude Max: 85.70, Longitude Max: 179.99
 - 가장 북쪽에 위치한 분화구의 위도는 85.70
 - 가장 동쪽에 위치한 분화구의 경도는 179.99
- Latitude 25% (1사분위수): -30.94, Longitude 25% (1사분위수): -58.83
 - 분화구의 25%가 위도 -30.94 이하, 경도 -58.83 이하에 위치함
 - ➡ 분화구가 남반구의 낮은 위도와 서쪽 반구에 많이 분포하고 있음을 의미

- Latitude 50% (Median): -10.08, Longitude 50% (Median): 12.74
 - 중앙값은 위도 -10.08, 경도 12.74로, 절반의 분화구가 이 값 이하에 위치
- Latitude 75% (3사분위수): 17.22, Longitude 75% (3사분위수): 89.28
 - 분화구의 75%가 위도 17.22 이하, 경도 89.28 이하에 위치
 - ➡ 북반구와 동쪽 반구에 분화구가 더 많이 분포하고 있음을 의미

Variance

- Latitude Variance: 1129.56, Longitude Variance: 9339.60
 - 분산이 높다는 것은 데이터가 평균으로부터 많이 퍼져 있음
 - 분화구는 화성의 동서 방향으로 고르게 퍼져 있고 특정 지역에 집중되지 않고 전역에 걸쳐 분포되어 있음
 - 그러면 실제로 좌표 찍어서 표현하면 고르게 이미지가 나와야 함
 - 가설: 그래프로 각 크레이터 좌표를 찍으면 고르게 잘 나올거다...!

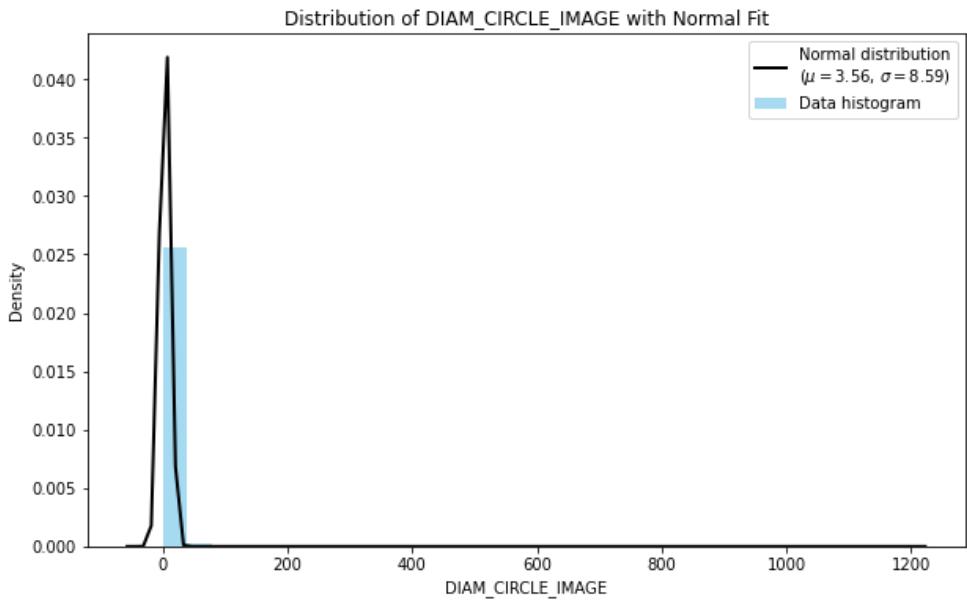
위도 경도 정리

1. 위도 경도는 정규분포 형태 X
 - a. 정규성 검증을 진행했는데 정규분포를 따르지 않음
2. 위도 경도가 고르게 분포되어 있음
3. 가설 각 좌표를 2D 그래프에 찍으면 전체에 고르게 좌표가 찍혀야 함

분화구의 직경 및 깊이

직경

- Mean: 3.56 km, Std: 8.59 km
 - 평균 직경은 3.56km로 대부분의 분화구가 작은 직경을 가지고 있음을 나타냄
 - 표준편차가 8.59km로 큰 편이므로 일부 큰 분화구가 평균을 있음



➡️ 그래프를 보면 정규분포를 따르지 않음 (왜곡이 큼)

✓ MinMax

- **Min: 1.00 km, Max: 1164.22 km**
 - 가장 작은 분화구의 직경은 1.00km 가장 큰 분화구의 직경은 1164.22km 임
 - 상당히 편차가 심함
- **25% (1사분위수): 1.18 km, 50% (Median): 1.53 km, 75% (3사분위수): 2.55 km**
 - 분화구의 25%가 직경 1.18km 이하
 - 50%가 1.53km 이하
 - 75%가 2.55km 이하

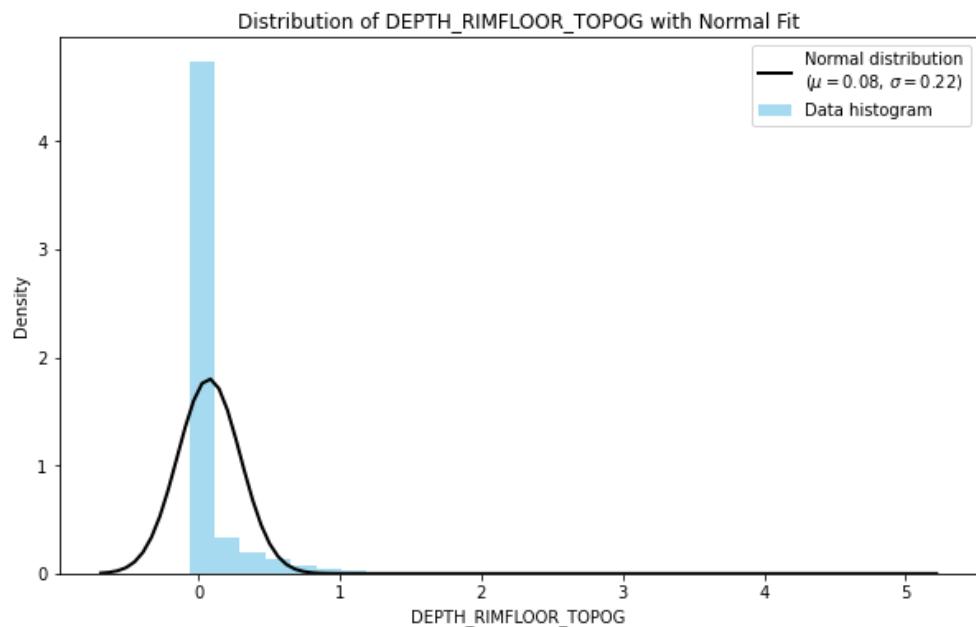
✓ Variance

- **Variance: 73.82**
 - 분산이 높아 데이터가 평균에서 많이 퍼져 있음을 나타냄

✓ 깊이(Depth)

- Mean: 0.08 km, Std: 0.22 km

- 평균 깊이는 0.08km로 대부분의 분화구가 얕음을 나타냄
- 표준편차는 0.22km로 데이터가 크게 퍼져 있지 않음을 의미함



➡️ 그래프를 보면 대부분의 데이터가 0 근처에 몰려있음..

➡️ 정규분포 곡선과 차이가 큼

✓ MinMax

- Min: -0.42 km, Max: 4.95 km

- 가장 얕은 분화구의 깊이는 -0.42km 가장 깊은 분화구의 깊이는 4.95km임

- 25% (1사분위수): 0.00 km, 50% (Median): 0.00 km 75% (3사분위수): 0.00 km

- 분화구의 25%가 깊이 0.00km 이하
- 50%가 0.00km 이하
- 75%가 0.00km 이하 ...!?!?

🛠️ 분화구 깊이 0 이하 그래프로 표시

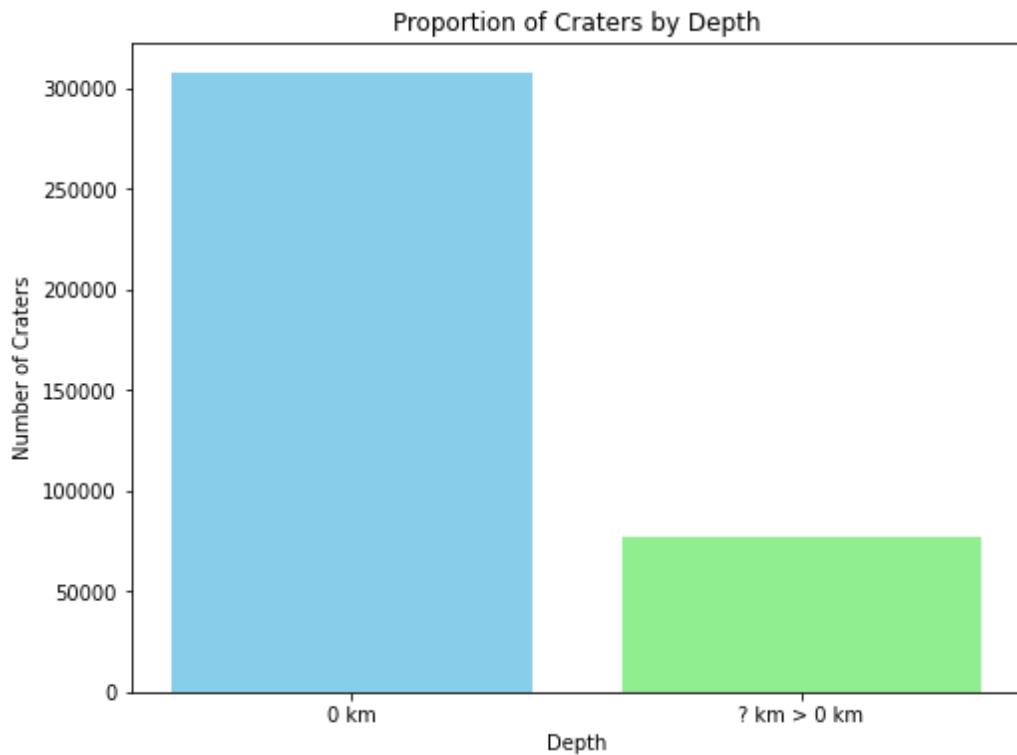
```
zero_depth_craters = df[df['DEPTH_RIMFLOOR_TOPOG'] == 0]
non_zero_depth_craters = df[df['DEPTH_RIMFLOOR_TOPOG'] > 0]
total_craters = df.shape[0]

# Create a DataFrame for the counts
depth_counts = pd.DataFrame({
    'Depth': ['0 km', '? km > 0 km'],
    'Count': [zero_depth_craters, non_zero_depth_craters]
})

# Plot the bar chart
plt.figure(figsize=(8, 6))
plt.bar(depth_counts['Depth'], depth_counts['Count'])
plt.title('Proportion of Craters by Depth')
plt.xlabel('Depth')
plt.ylabel('Number of Craters')
plt.show()

# Calculate the proportion of craters with depth 0
zero_depth_proportion = zero_depth_craters / total_craters

print(f"깊이가 0인 분화구의 비율: {zero_depth_proportion:.2f}")
```



➡ 깊이가 0인 분화구의 비율: 80.01%임 그래서 저렇게 비율이 나온것으로 분석

✓ Variance

- **Variance: 0.05**

- 분산이 낮아 데이터가 평균에서 크게 퍼져 있지 않음을 나타냄

✓ 정리

- 분화구의 크기와 깊이는 다양하지만 대부분의 분화구는 직경이 작고 얕음
- 일부 매우 큰 직경의 분화구가 존재하여 데이터 분포에 큰 영향을 미침
- 직경과 깊이 컬럼은 전혀 정규분포를 따르지 않음

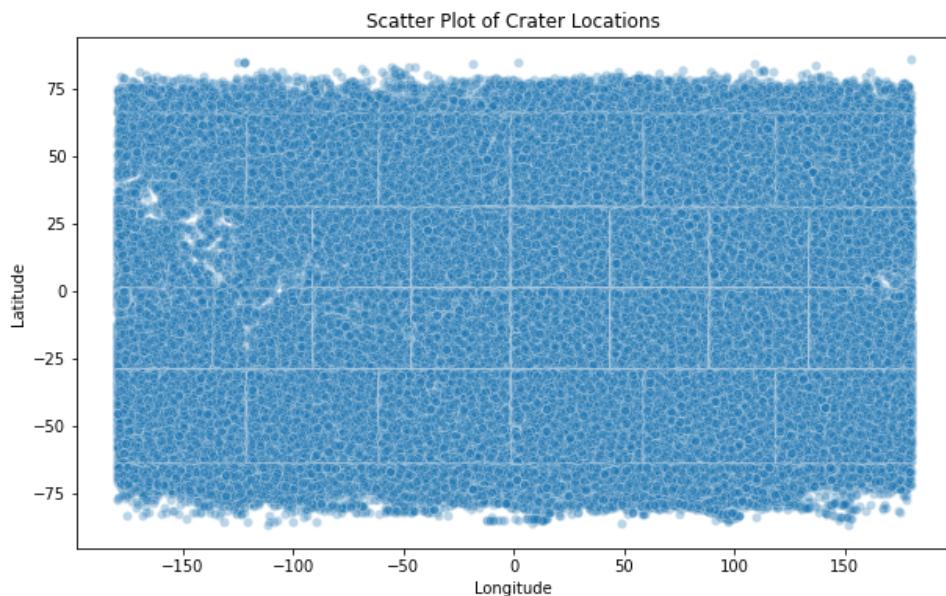
▼ 크레이터 수치형 컬럼 상관관계 분석, 밀도 분포, 선형 및 비선형 분석

💿 분화구의 위도 경도 깊이 면적 분석

✓ 위도와 경도가 함께 분포된 형태를 파악

위도와 경도가 함께 분포된 형태를 파악

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='LONGITUDE_CIRCLE_IMAGE', y='LATITUDE')
plt.title('Scatter Plot of Crater Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

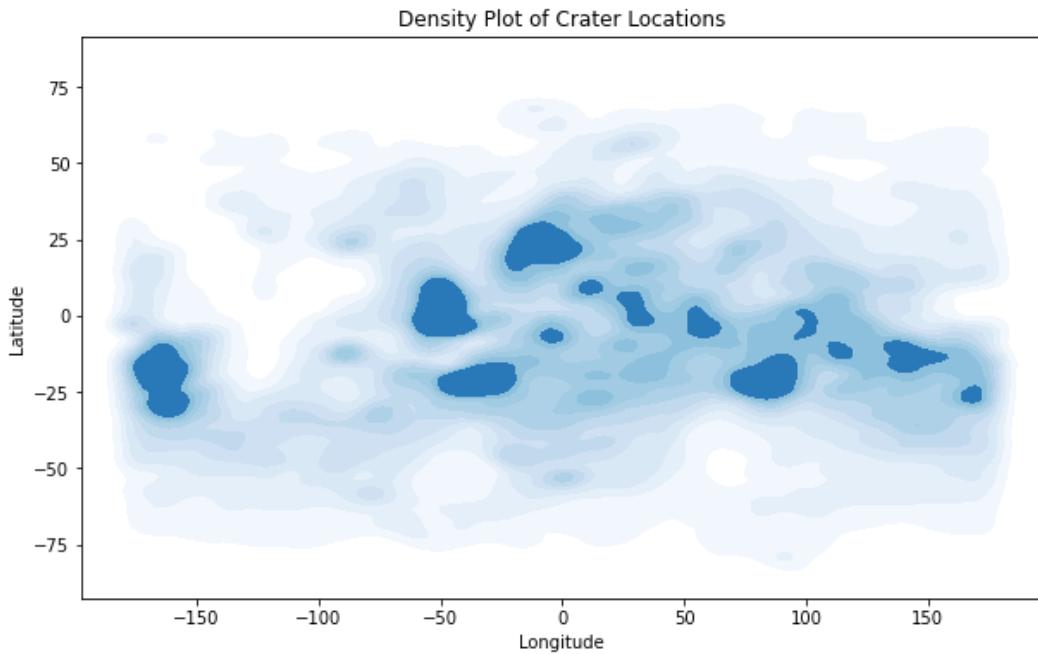


→ Latitude Variance: 1129.56, Longitude Variance: 9339.60 분산이 커서 엄청 고르게 분포되어

있는 것을 확인할 수 있음

✓ 위도와 경도 밀도 분석

```
plt.figure(figsize=(10, 6))
sns.kdeplot(x=df['LONGITUDE_CIRCLE_IMAGE'], y=df['LATITUDE'])
plt.title('Density Plot of Crater Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

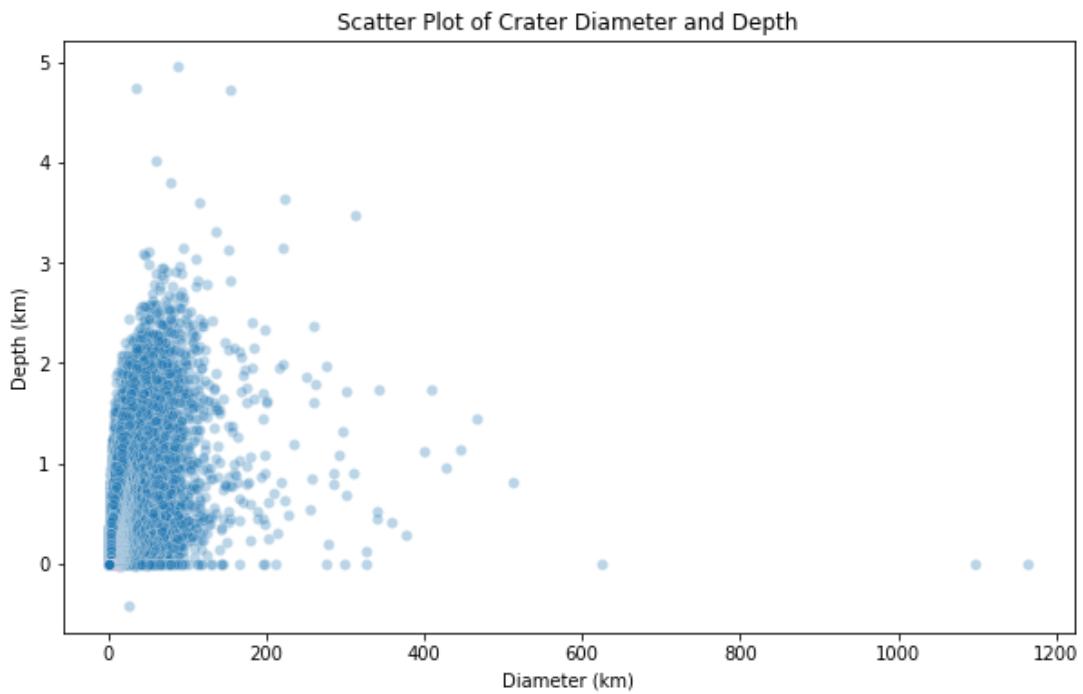


- **분화구의 자리적 분포**
 - 분화구는 화성의 적도와 중위도 지역에 주로 밀집되어 있음
 - 극지방에는 상대적으로 분포가 적음

- **주요 밀집 지역:**
 - 특정 경도와 위도에 분화구가 집중되어 있음
 - 특히 경도 -150도, -75도, 0도, 50도, 100도 근처에 밀집된 지역이 보임

✓ 깊이와 직경의 상관관계 및 밀도 분석

```
# 깊이와 직경의 상관관계 분석
plt.figure(figsize=(10, 6))
sns.scatterplot(x='DIAM_CIRCLE_IMAGE', y='DEPTH_RIMFLOOR')
plt.title('Scatter Plot of Crater Diameter and Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.show()
```



선형 상관관계 여부

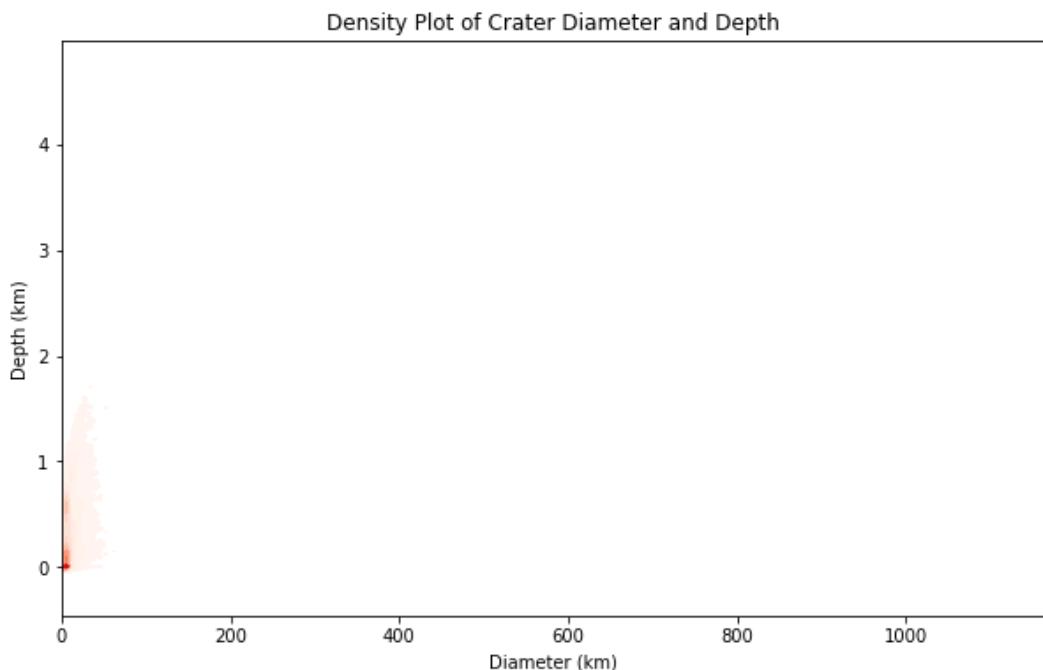
- 그래프를 보면 분화구의 직경과 깊이 사이에 명확한 선형 상관관계는 보이지 않음
- 대부분의 분화구가 직경이 작고 깊이가 얕으며 큰 직경의 분화구는 깊이의 변동이 크고 직경이 커질수록 깊이도 증가하는 패턴도 보임
- 큰 직경의 분화구에서도 깊이가 일정하지 않고 다양하게 분포되어 있어 직경과 깊이 사이의 관계는 선형 X

체크

- 작은 분화구
 - 작은 직경의 분화구는 깊이가 일정하거나 얕은 경우가 많음
- 큰 분화구
 - 큰 직경의 분화구는 깊이가 다양하게 분포하며 직경이 증가함에 따라 깊이도 증가하는 경향이 있지만 일관된 패턴은 보이지 않음 이상치도 존재함

깊이와 직경의 밀도 분포 분석

```
# 깊이와 직경의 밀도 분포 분석
plt.figure(figsize=(10, 6))
sns.kdeplot(x=df['DIAM_CIRCLE_IMAGE'], y=df['DEPTH_RIMF'])
plt.title('Density Plot of Crater Diameter and Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.show()
```



- 대부분의 분화구는 직경이 0에서 50km 사이에 집중되어 있음
- 깊이는 주로 0에서 1km 사이에 집중되어 있음
- 깊이가 3km 이상인 분화구는 거의 없고 대부분의 분화구 깊이는 2km 이하임

➡️ 분화구 그룹을 식별하기 위해 Kmeans ⇒ k는 3개로 지정

```
from sklearn.cluster import KMeans
import numpy as np

data_for_clustering = df[['DIAM_CIRCLE_IMAGE', 'DEPTH_RIMF']]
kmeans = KMeans(n_clusters=3)
```

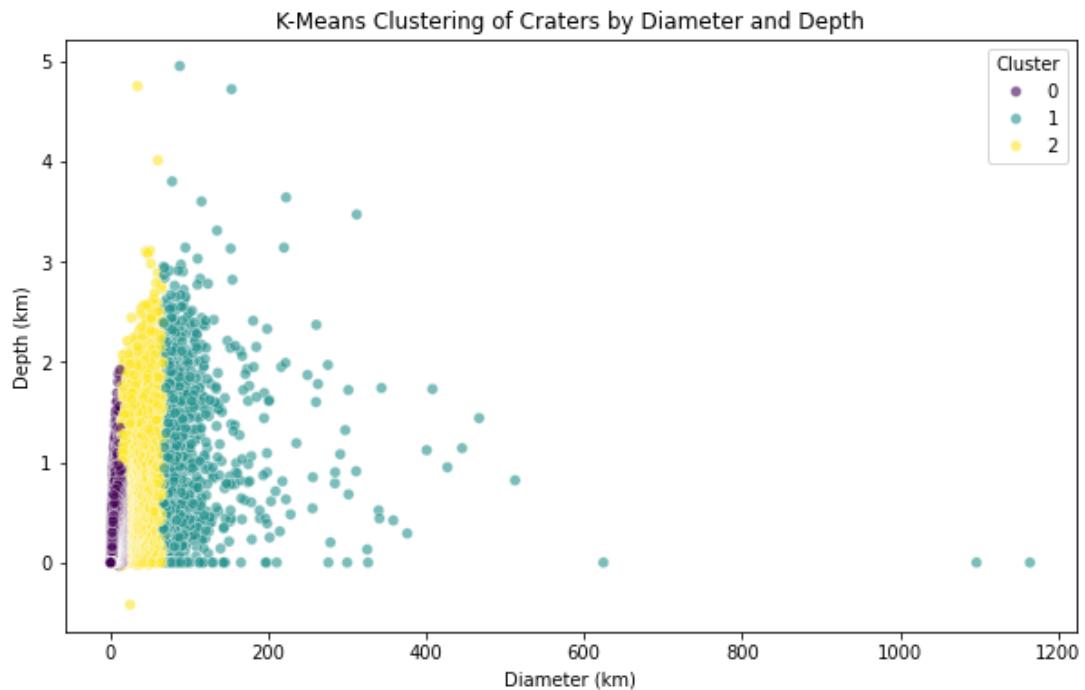
```

clusters = kmeans.fit_predict(data_for_clustering)

data_for_clustering['Cluster'] = clusters

plt.figure(figsize=(10, 6))
sns.scatterplot(x='DIAM_CIRCLE_IMAGE', y='DEPTH_RIMFLOOR')
plt.title('K-Means Clustering of Craters by Diameter and Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.legend(title='Cluster')
plt.show()

```



1. Cluster 0 (보라색)

- 특징:** 주로 직경이 50km 이하이며 깊이가 0.5km 이하인 분화구를 포함
- 분석:** 작은 크기와 얕은 깊이의 분화구로 상대적으로 덜 깊고 작은 충돌로 형성된 분화구일 가능성이 높음

2. Cluster 1 (청록색)

- 특징:** 직경이 다양한 분화구를 포함하며 깊이는 주로 0km ~ 3km 사이...
직경은 최대 1200km까지 다양함

- **분석:** 다양한 크기와 깊이의 분화구를 포함하며 큰 충돌 이벤트로 형성된 분화구도 포함될 수 있음 또한 깊이의 분포가 넓어 더 다양한 형성 조건을 가진 분화구를 나타냄

3. Cluster 2 (노란색)

- **특징:** 주로 직경이 50km 이하이고 깊이가 0.5km ~ 1km 사이인 분화구를 포함함
- **분석:** 중간 크기와 중간 깊이의 분화구로 보통 크기의 충돌 이벤트로 형성된 분화구일 가능성이 있음 또한 깊이가 약간 더 깊은 분화구를 포함함

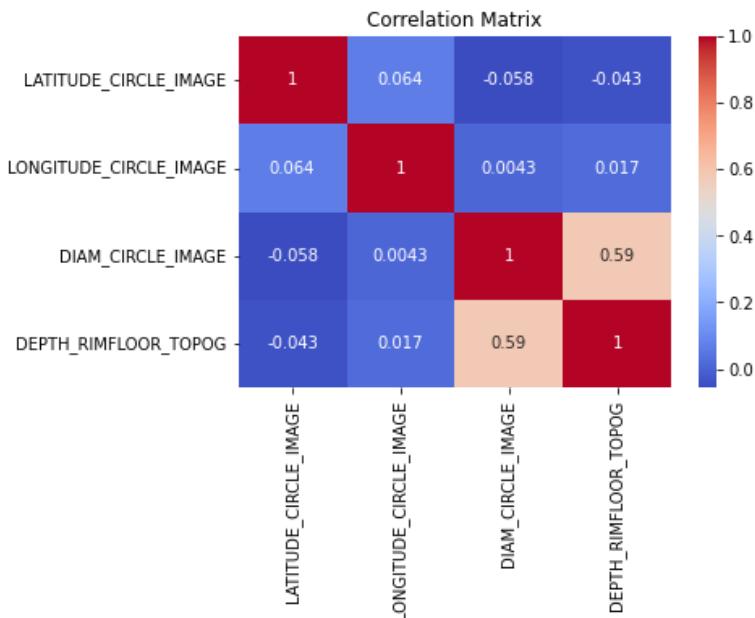
✓ 정리

- **작은 분화구 (Cluster 0)**
 - 작은 충돌 이벤트로 형성된 작은 분화구
- **중간 크기의 분화구 (Cluster 2)**
 - 중간 크기의 충돌 이벤트로 형성된 분화구로 깊이도 중간 정도
- **다양한 크기와 깊이의 분화구 (Cluster 1)**
 - 다양한 충돌 이벤트로 형성된 다양한 크기와 깊이의 분화구로 특히 큰 충돌 이벤트로 형성된 분화구도 포함됨

✓ 깊이와 직경의 상관계수 및 전체 상관계수

```
correlation_matrix = df.corr()
print("Correlation matrix:\n", correlation_matrix)
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

➡ 상관계수 (Correlation coefficient): 0.59



가설대로 상관계수가 있음

선형 관계가 아닐거지만 그래도 그래프로 파악

1. 이상치 제거

- a. 350km 이상이거나 깊이가 3km 이상인 데이터를 이상치로 간주
- b. 전에 그래프 보면 이상치가 많음

```

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
## 이상치 식별 및 제거
filtered_df = df[(df['DIAM_CIRCLE_IMAGE'] <= 350) & (df['DEPTH_RIMFLOOR_TOPOG'] <= 3000)]

# 선형 회귀 모델 학습
X = filtered_df[['DIAM_CIRCLE_IMAGE']]
y = filtered_df['DEPTH_RIMFLOOR_TOPOG']

linear_reg_model = LinearRegression()
linear_reg_model.fit(X, y)

# 예측

```

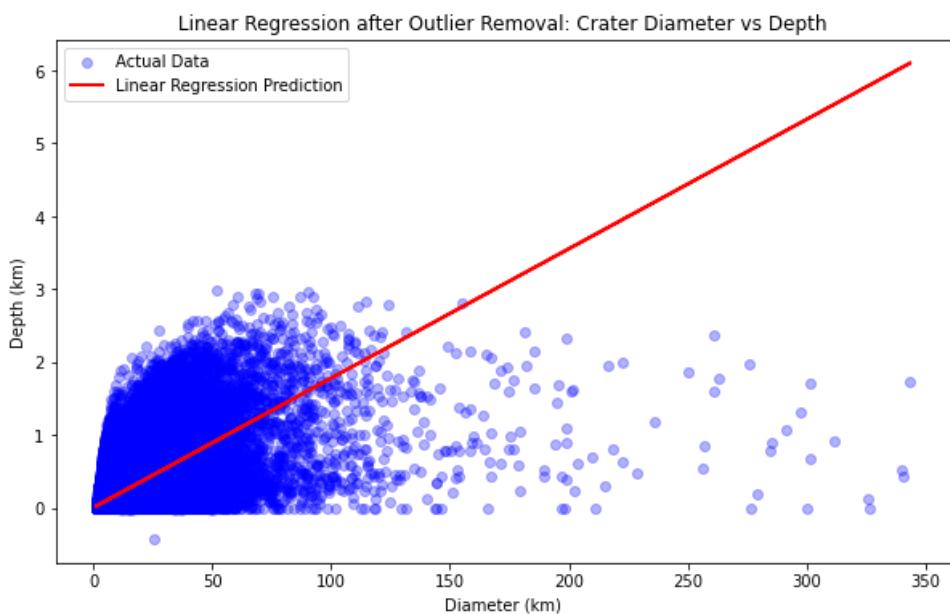
```

y_pred = linear_reg_model.predict(X)

# 모델 평가
rmse = np.sqrt(mean_squared_error(y, y_pred))
r2 = r2_score(y, y_pred)
print(f"Linear Regression RMSE: {rmse:.2f}")
print(f"Linear Regression R-squared (R2): {r2:.2f}")

# 결과 시각화
plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='blue', alpha=0.3, label='Actual Data')
plt.plot(X, y_pred, color='red', linewidth=2, label='Linear Regression Prediction')
plt.title('Linear Regression after Outlier Removal: Crater Diameter vs Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.legend()
plt.show()

```



➡ 이상치 제거 후 6% 정도 높아져 데이터 40% 설명 가능 하지만 전혀 선형적이지
않음

비선형인가?

비선형 패턴 파악

1 다항 회귀

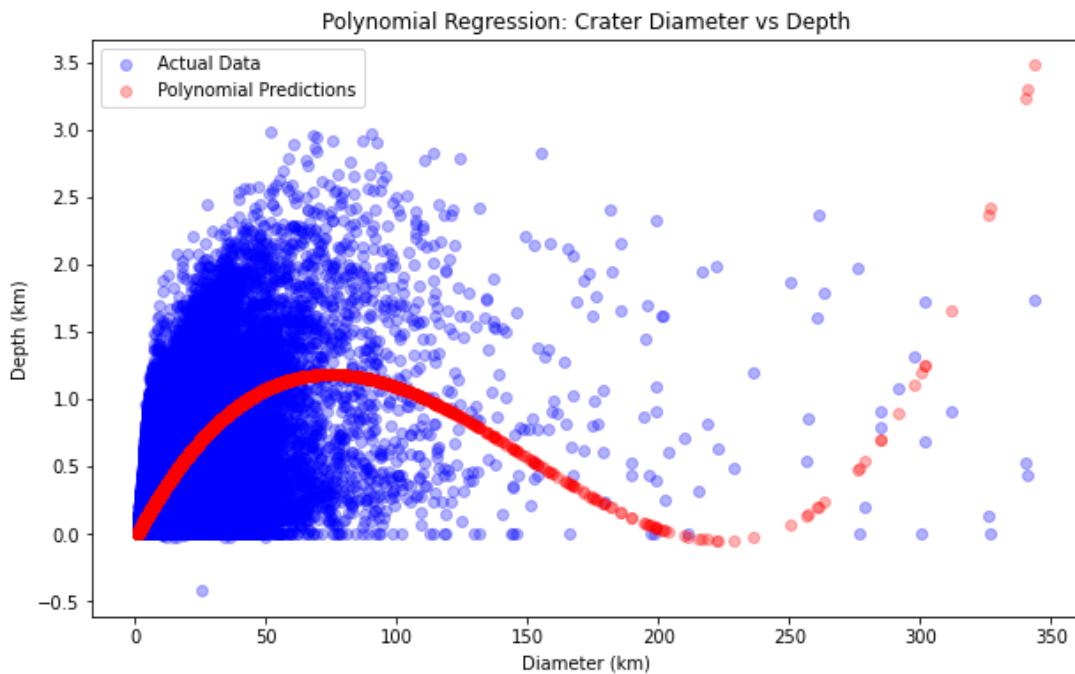
```
# 다항 회귀 모델 적용 (degree=3)
poly = PolynomialFeatures(degree=3)
X_poly = poly.fit_transform(X)

# 모델 학습
poly_reg_model = LinearRegression()
poly_reg_model.fit(X_poly, y)

# 예측
y_poly_pred = poly_reg_model.predict(X_poly)

# 모델 평가
rmse_poly = np.sqrt(mean_squared_error(y, y_poly_pred))
r2_poly = r2_score(y, y_poly_pred)
print(f"Polynomial Regression RMSE: {rmse_poly:.2f}")
print(f"Polynomial Regression R-squared (R2): {r2_poly:.2f}")

# 결과 시각화
plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='blue', alpha=0.3, label='Actual')
plt.scatter(X, y_poly_pred, color='red', alpha=0.3, label='Predicted')
plt.title('Polynomial Regression: Crater Diameter vs Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.legend()
plt.show()
```



Polynomial Regression RMSE: 0.15

Polynomial Regression R-squared (R2): 0.53

2 지수 회귀

```

log_y = np.log1p(y) # 로그 변환 적용

# 모델 학습
exp_reg_model = LinearRegression()
exp_reg_model.fit(X, log_y)

# 예측
y_exp_pred = np.expm1(exp_reg_model.predict(X)) # 예측

# 모델 평가
rmse_exp = np.sqrt(mean_squared_error(y, y_exp_pred))
r2_exp = r2_score(y, y_exp_pred)
print(f"Exponential Regression RMSE: {rmse_exp:.2f}")
print(f"Exponential Regression R-squared (R2): {r2_exp:.2f}")

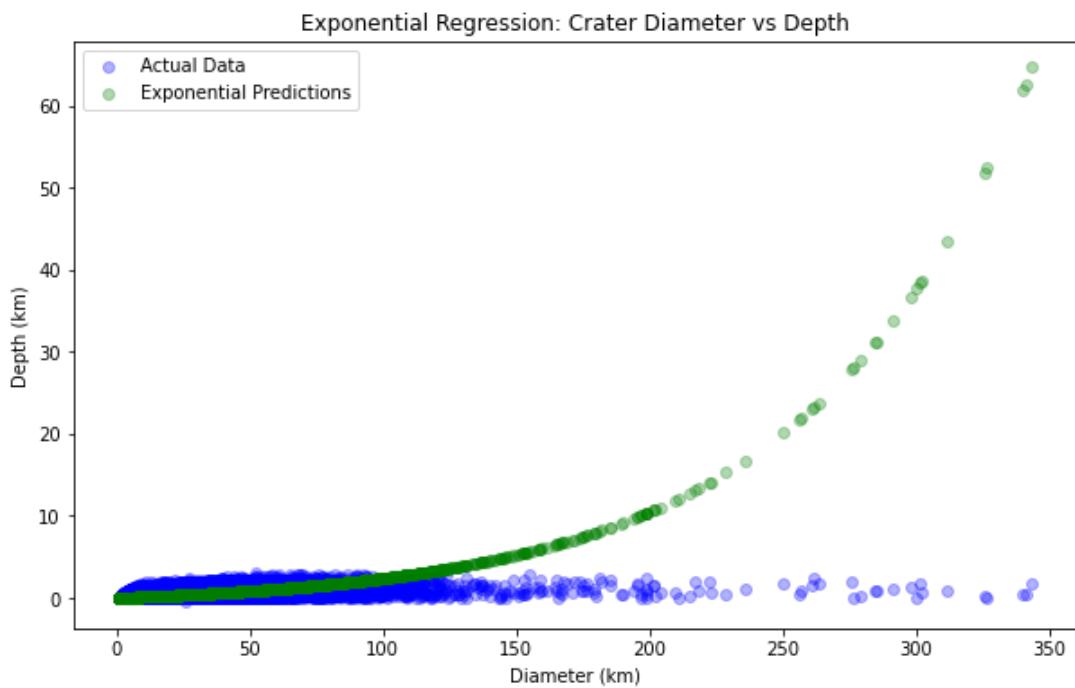
# 결과 시각화

```

```

plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='blue', alpha=0.3, label='Actual Data')
plt.scatter(X, y_exp_pred, color='green', alpha=0.3, label='Exponential Predictions')
plt.title('Exponential Regression: Crater Diameter vs Depth')
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.legend()
plt.show()

```



Exponential Regression RMSE: 0.36

Exponential Regression R-squared (R²): -1.61

3 로그 회귀

```

log_X = np.log1p(filtered_df['DIAM_CIRCLE_IMAGE']).values
y = filtered_df['DEPTH_RIMFL0OR_TOPOG']

# 모델 학습
log_reg_model = LinearRegression()
log_reg_model.fit(log_X, y)

```

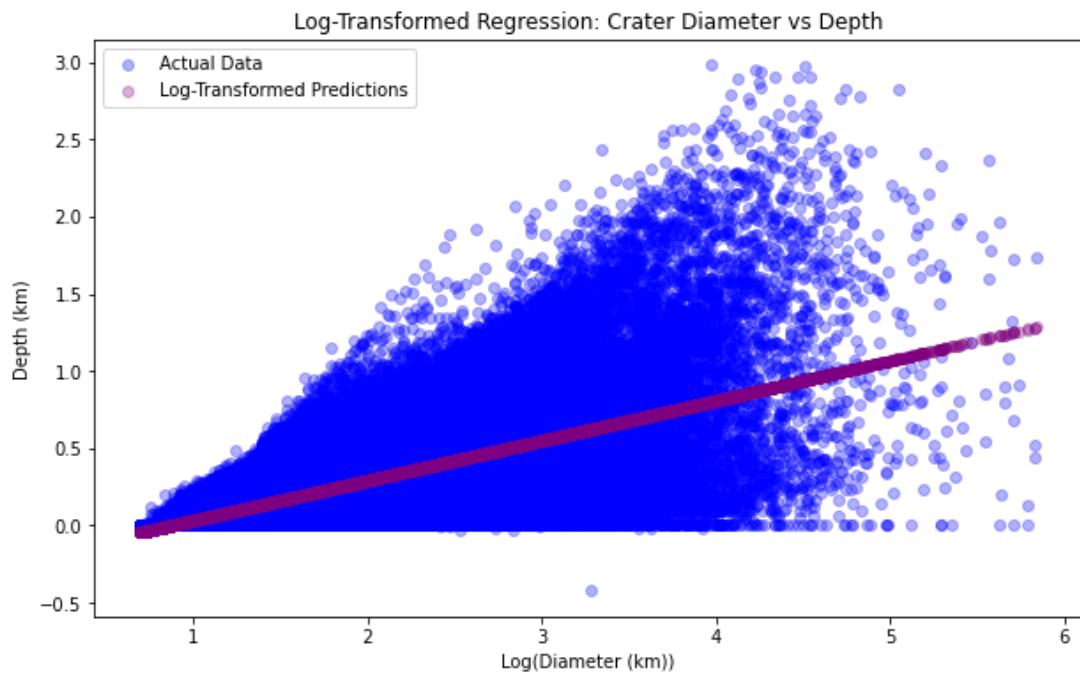
```

# 예측
y_log_pred = log_reg_model.predict(log_X)

# 모델 평가
rmse_log = np.sqrt(mean_squared_error(y, y_log_pred))
r2_log = r2_score(y, y_log_pred)
print(f"Log-Transformed Regression RMSE: {rmse_log:.2f}")
print(f"Log-Transformed Regression R-squared (R2): {r2_}

# 결과 시각화
plt.figure(figsize=(10, 6))
plt.scatter(log_X, y, color='blue', alpha=0.3, label='Actual Data')
plt.scatter(log_X, y_log_pred, color='purple', alpha=0.3, label='Log-Transformed Predictions')
plt.title('Log-Transformed Regression: Crater Diameter vs Depth')
plt.xlabel('Log(Diameter (km))')
plt.ylabel('Depth (km)')
plt.legend()
plt.show()

```



Log-Transformed Regression RMSE: 0.15
 Log-Transformed Regression R-squared (R2): 0.55

4 다변량 회귀

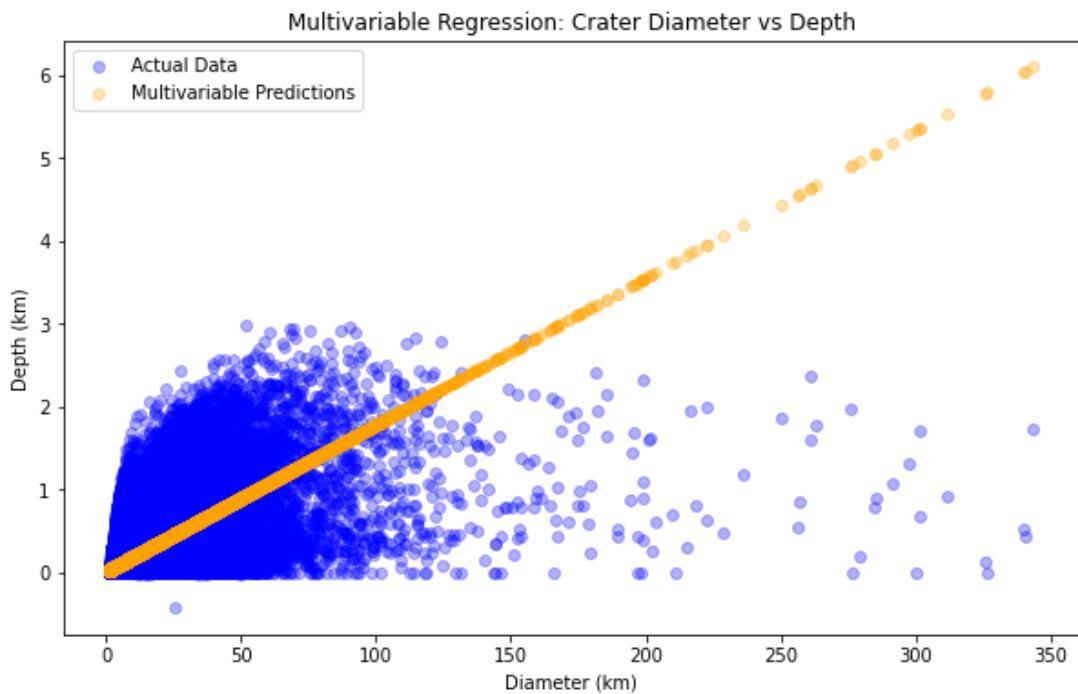
```
# 다변량 회귀 모델 학습
multi_X = filtered_df[['DIAM_CIRCLE_IMAGE', 'LATITUDE_C
multi_y = filtered_df['DEPTH_RIMFLLOOR_TOPOG']

multi_reg_model = LinearRegression()
multi_reg_model.fit(multi_X, multi_y)

# 예측
y_multi_pred = multi_reg_model.predict(multi_X)

# 모델 평가
rmse_multi = np.sqrt(mean_squared_error(multi_y, y_multi_
r2_multi = r2_score(multi_y, y_multi_pred)
print(f"Multivariable Regression RMSE: {rmse_multi:.2f}")
print(f"Multivariable Regression R-squared (R2): {r2_mu

# 결과 시각화
plt.figure(figsize=(10, 6))
plt.scatter(multi_X['DIAM_CIRCLE_IMAGE'], multi_y, colo
plt.scatter(multi_X['DIAM_CIRCLE_IMAGE'], y_multi_pred,
plt.title('Multivariable Regression: Crater Diameter vs
plt.xlabel('Diameter (km)')
plt.ylabel('Depth (km)')
plt.legend()
plt.show()
```



Multivariable Regression RMSE: 0.17

Multivariable Regression R-squared (R²): 0.40

✓ 비선형 모델 분석

- **다항 회귀와 로그 회귀**
 - 모델이 가장 높은 설명력을 가지고 있음
 - 두 모델이 직경과 깊이 간의 비선형 관계를 잘 설명하고 있음을 의미함
- **지수 회귀**
 - 모델은 예측값이 과도하게 커지고 실제 데이터와 맞지 않기 때문에 적합하지 않음
- **다면량 회귀** 모델은 추가 변수를 포함하지만, 여전히 설명력이 떨어짐

✓ 깊이와 직경의 분석

- 깊이와 직경 간의 관계가 비선형적임
- 큰 직경의 분화구는 예측이 더 어렵고 추가적인 변수가 필요함

- ▶ 복잡한 비선형 패턴이 존재함 ... 단순 선형 관계로 설명되지 않음을 확인
했음 분화구 형성 과정이 엄청 복합적이고 다양한 변수와 요인을 고려한
심층적인 지질학적 분석이 필요함

➡ 가설 → 분화구의 직경이 클수록 깊이도 깊을 것..?은 전혀 아니었던 걸로

Linear Regression RMSE: 0.17
Linear Regression R-squared (R²): 0.40



가설 “분화구의 직경이 클수록 깊이도 깊을 것” NO 😢

▼ 크레이터 분출물 및 총 수 컬럼 특징 분석

✓ 분화구 형태학적 특징 컬럼 파악

분화구 형태학적 특징 전체 파악

```
morphology_ejecta_1_unique = sorted_craters['MORPHOLOGY']
morphology_ejecta_2_unique = sorted_craters['MORPHOLOGY']
morphology_ejecta_3_unique = sorted_craters['MORPHOLOGY']

morphology_ejecta_1_unique, morphology_ejecta_2_unique,
```

▼ 특징 데이터

분화구 분출물 Top 10개

```
def plot_top_10_morphology(df, column_name):
    filtered_df = df[df[column_name] != 'NA']

    value_counts = filtered_df[column_name].value_counts()

    plt.figure(figsize=(10, 6))
    bars = plt.bar(value_counts.index, value_counts.values)
    plt.title(f'Top 10 {column_name}')
    plt.xlabel(column_name)
    plt.ylabel('Frequency')
    plt.xticks(rotation=45)

    for bar in bars:
        yval = bar.get_height()
        plt.text(bar.get_x() + bar.get_width()/2, yval - 5, str(yval))
```

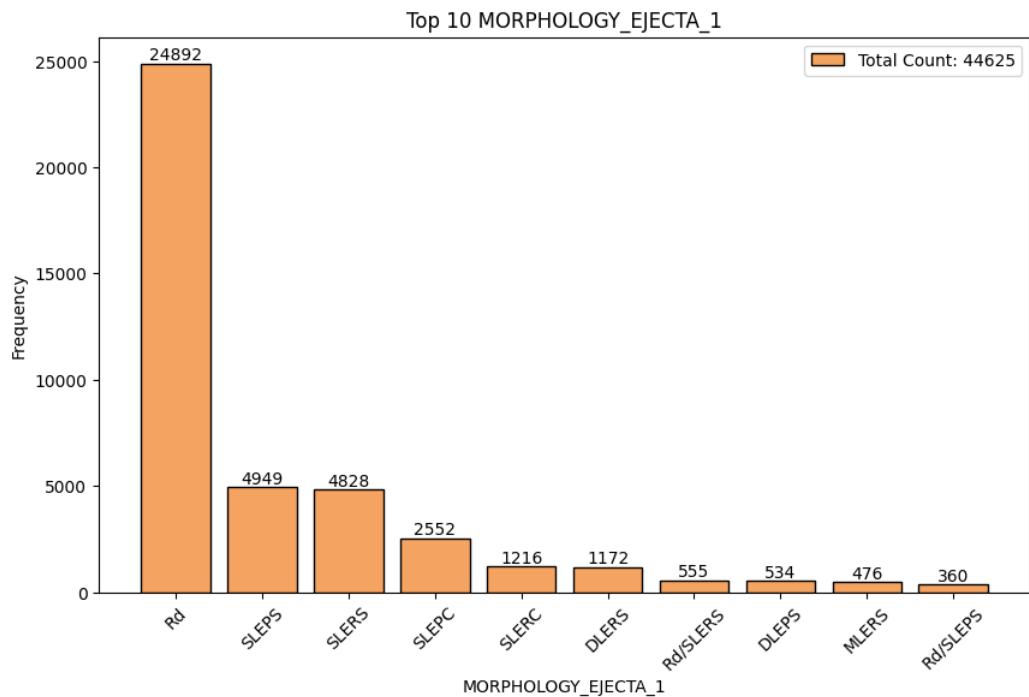
```
plt.legend([f'Total Count: {filtered_df[column_name]}'])
plt.show()
```

```
plot_top_10_morphology(df, 'MORPHOLOGY_EJECTA_1')
```

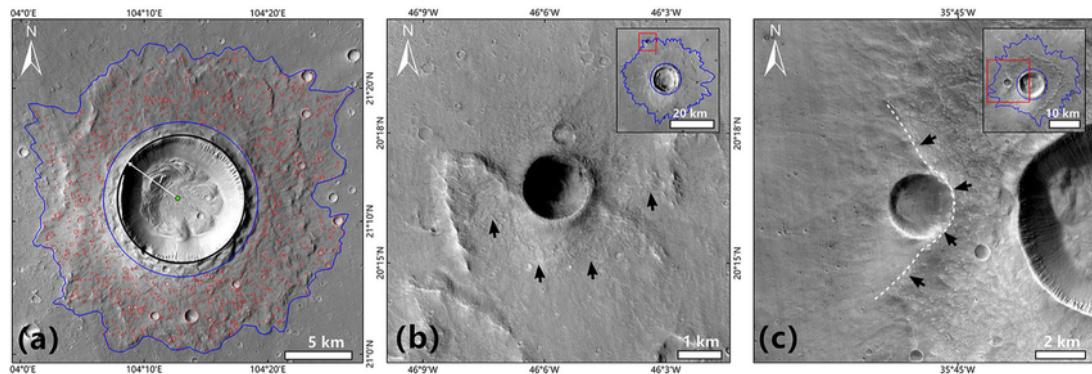
```
plot_top_10_morphology(df, 'MORPHOLOGY_EJECTA_2')
```

```
plot_top_10_morphology(df, 'MORPHOLOGY_EJECTA_3')
```

✓ MORPHOLOGY_EJECTA_1

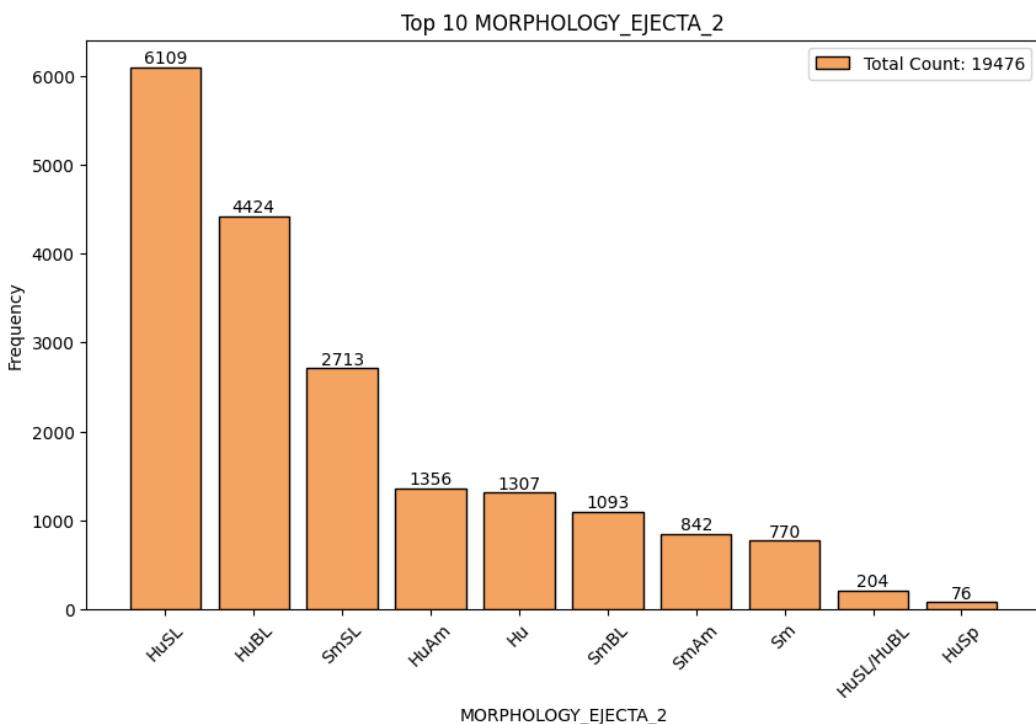


- Rd가 가장 많이 있음
 - Radial ejecta로 방출물의 물질이 분화구 주변으로 방사형으로 뻗어나간 형태
 - 분화구 형성 시 강한 폭발력에 의해 물질이 중심에서 방사형으로 퍼졌음

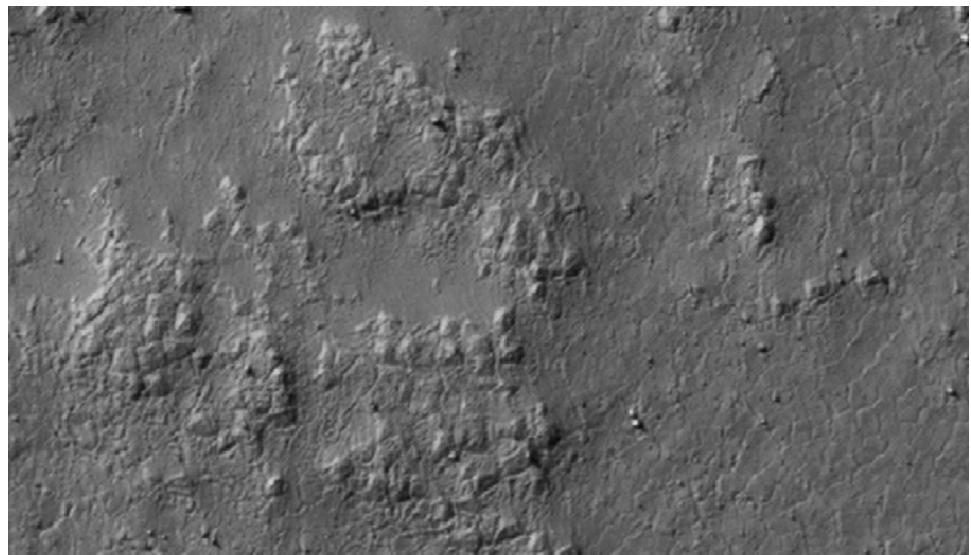


https://www.researchgate.net/figure/Layered-ejecta-craters-LEC-examples-showing-how-we-delineate-the-outline-of-ejecta_fig4_358728900

✓ MORPHOLOGY_EJECTA_2

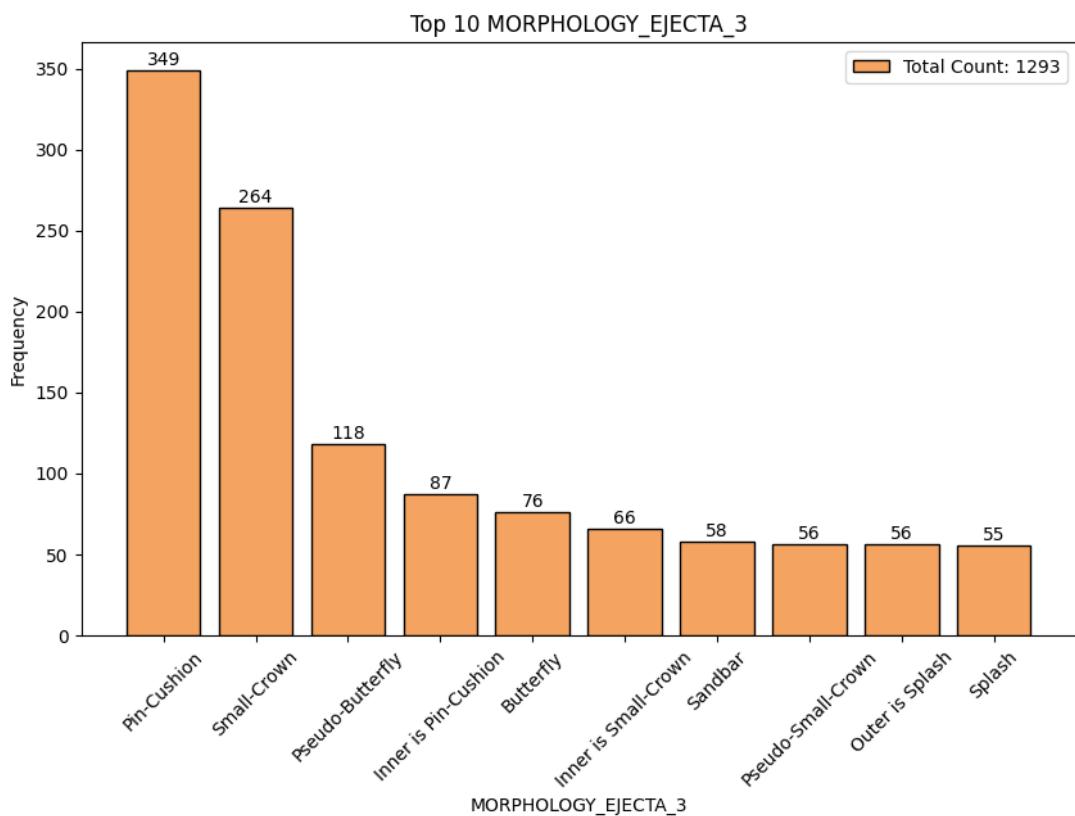


- HuSL가 (Hummocky Single Layer Ejecta) 가장 많았음
 - 율퉁불퉁한 방출물 구조가 주요 특징
 - 지형적으로 복잡한 방출물 분포



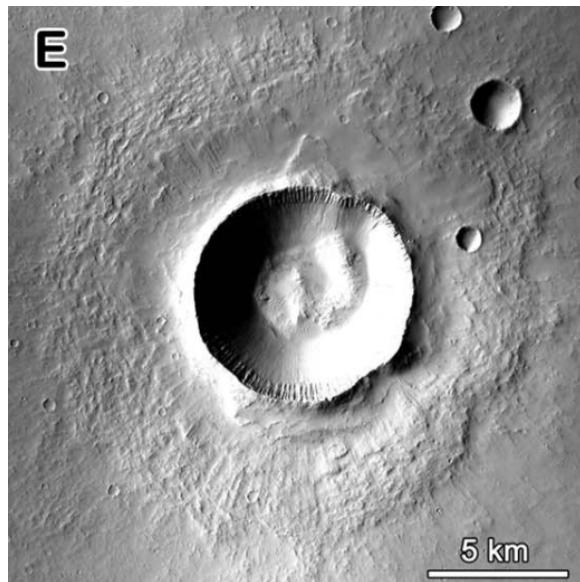
https://www.researchgate.net/figure/Hummocky-terrain-and-polygonal-textures-PSP-006605-1260-5391-N-13081-E-025_fig1_233962369

✓ MORPHOLOGY_EJECTA_3



- Pin-Cushion가 가장 많음
 - 분화구 주변의 방출물이 다양한 형태로 존재함을 알 수 있음

- 중앙에서 바깥쪽으로 향하는 방출물의 작은 돌기들이 여러 개 형성되어 있는 지형



<https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2011JE003966>

✓ 가설 → 분화구에 복잡한 방출물 형태가 더 많은 층수를 가질 것

1. 막대그래프로 Top 5에 드는 방출물 형태와 평균 층수를 표현

```
def analyze_ejecta_layers(df, column_name):
    filtered_df = df[df[column_name] != 'NA']

    ejecta_layers_mean = filtered_df.groupby(column_name).mean()

    ejecta_layers_mean = ejecta_layers_mean.sort_values

    colors = ['#b87333', '#cd7f32', '#e5aa70', '#d2691e']

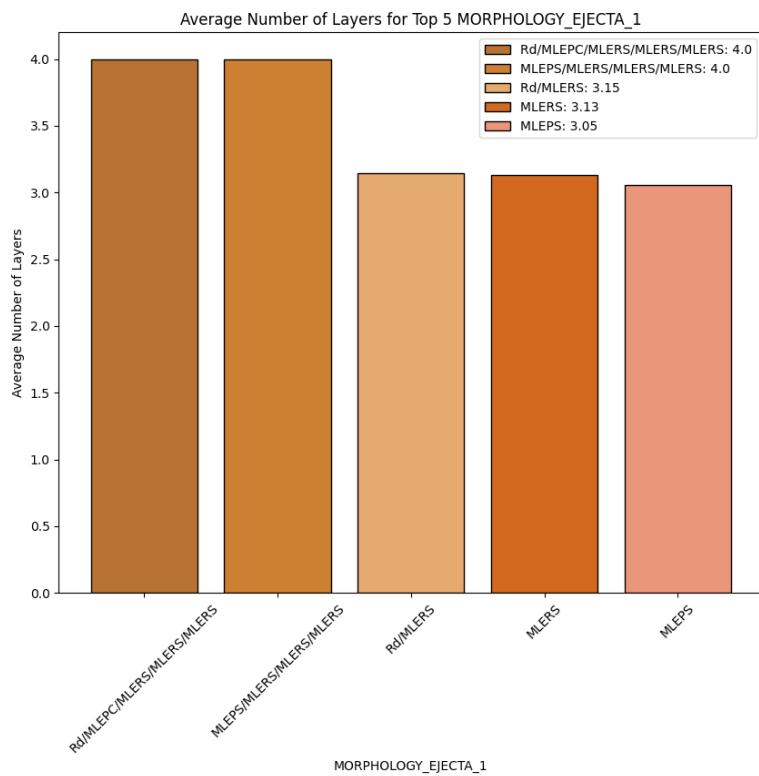
    plt.figure(figsize=(10, 8))
    bars = plt.bar(ejecta_layers_mean.index, ejecta_layers_mean)
    plt.title(f'Average Number of Layers for Top 5 {column_name}')
    plt.xlabel(column_name)
    plt.ylabel('Average Number of Layers')
    plt.xticks(rotation=45)

    plt.legend(bars, [f'{idx}: {round(val, 2)}' for idx
```

```
plt.show()
```

```
# 각 MORPHOLOGY_EJECTA 컬럼에 대해 분석
analyze_ejecta_layers(df, 'MORPHOLOGY_EJECTA_1')
analyze_ejecta_layers(df, 'MORPHOLOGY_EJECTA_2')
analyze_ejecta_layers(df, 'MORPHOLOGY_EJECTA_3')
```

✓ MORPHOLOGY_EJECTA_1

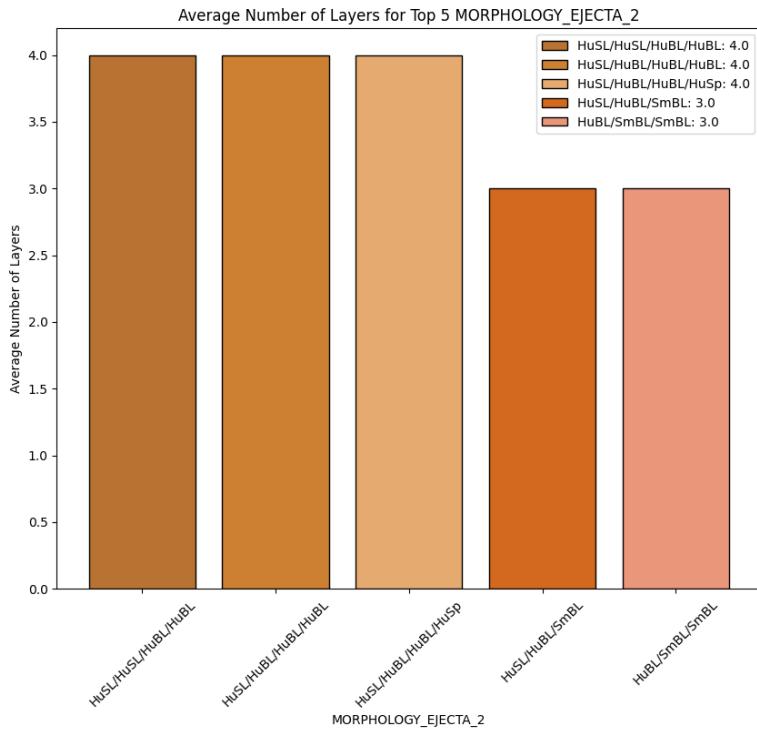


상위 5개의 방출물 형태

1. **Rd/MLEPC/MLERS/MLERS/MLERS:** 평균 층 수 4
2. **MLEPS/MLERS/MLERS/MLERS:** 평균 층 수 4
3. **Rd/MLERS:** 평균 층 수 3
4. **MLERS:** 평균 층 수 3
5. **MLEPS:** 평균 층 수 3

→ 복합적인 지질학적 형성 과정이 많은 층 수를 만들어냈을 가능성이 높음

✓ MORPHOLOGY_EJECTA_2

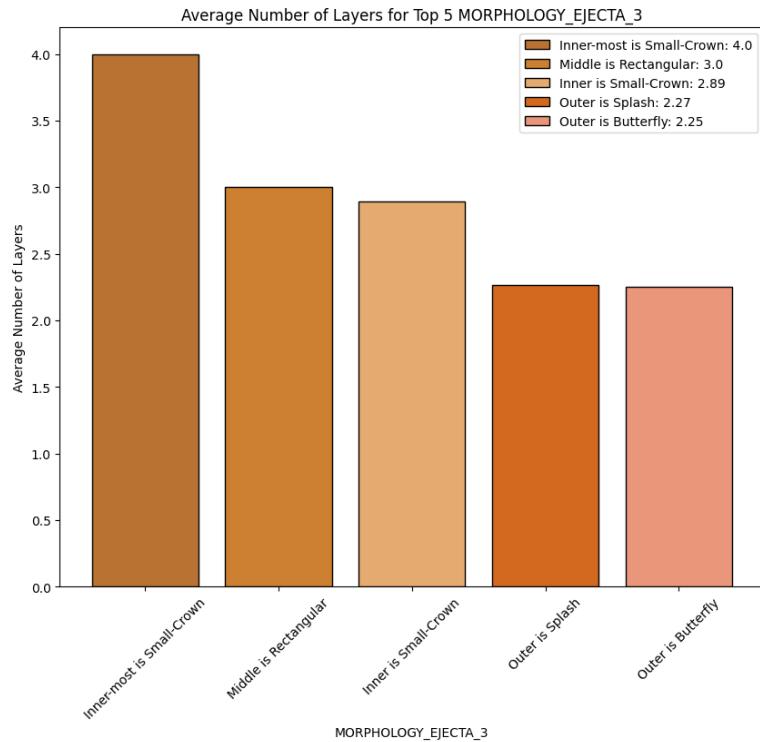


国旗 상위 5개의 방출물 형태

1. **HuSL/HuSL/HuBL/HuBL: 평균 층 수 4**
2. **HuSL/HuBL/HuBL/HuBL: 평균 층 수 4**
3. **HuSL/HuBL/HuBL/HuSp: 평균 층 수 4**
4. **HuSL/HuBL/SmBL: 평균 층 수 3**
5. **HuBL/SmBL/SmBL: 평균 층 수 3**

➡ 복합적인 지질학적 형성 과정이 많은 층 수를 만들어냈을 가능성이 높음

✓ MORPHOLOGY_EJECTA_3



纡상 위 5개의 방출물 형태

1. Inner-most is Small-Crown: 평균 층 수 4
2. Middle is Rectangular: 평균 층 수 3
3. **Inner is Small-Crown:** 평균 층 수 3
4. **Outer is Splash:** 평균 층 수 2
5. **Outer is Butterfly:** 평균 층 수 2

→ 중심부가 작은 왕관 형태인 방출물이 평균 층 수가 4층임

2. 상관관계 분석 (피어슨 상관계수)

```
from scipy.stats import pearsonr

def analyze_correlation(df):
    df = df.fillna('NA')

    # 각 행에서 방출물 형태의 개수 계산
    df['NUM_EJECTA_FORMS'] = df[['MORPHOLOGY_EJECTA_1',
                                   
    # 방출물 형태의 개수와 층 수 간의 상관관계 계산
```

```

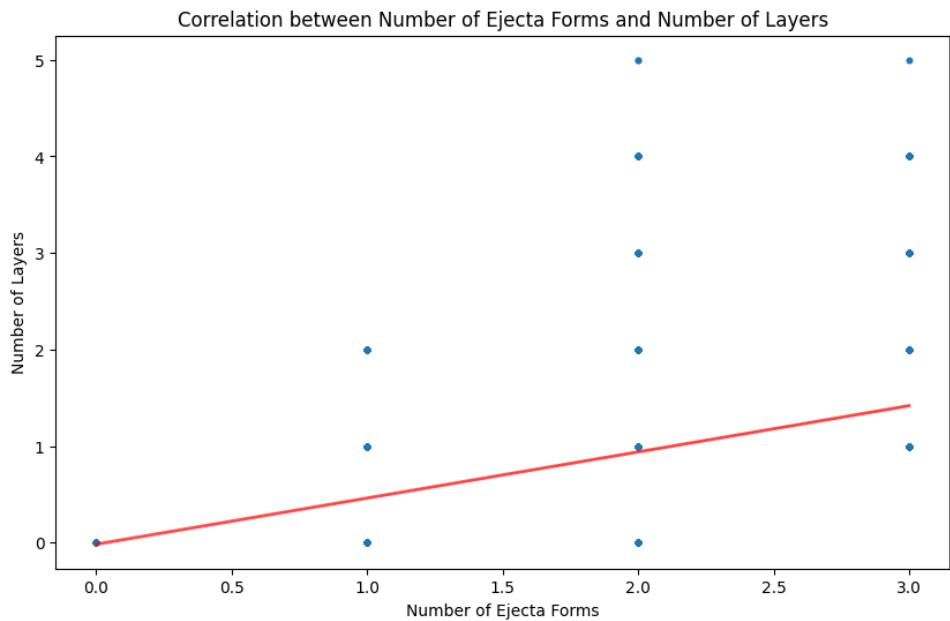
correlation, p_value = pearsonr(df['NUM_EJECTA_FORMS'],
                                 df['NUMBER_LAYERS'])

print(f"Correlation: {correlation}")
print(f"P-value: {p_value}")

# 산점도와 회귀선 시각화
plt.figure(figsize=(10, 6))
sns.regplot(x='NUM_EJECTA_FORMS', y='NUMBER_LAYERS')
plt.title('Correlation between Number of Ejecta Forms and Number of Layers')
plt.xlabel('Number of Ejecta Forms')
plt.ylabel('Number of Layers')
plt.show()

# 분석 수행
analyze_correlation(df)

```



Correlation: 0.7927468207486432 (양의 상관관계)

P-value: 0.0

● 방출물 형태의 개수가 증가할수록 분화구의 층 수도 증가하는 패턴



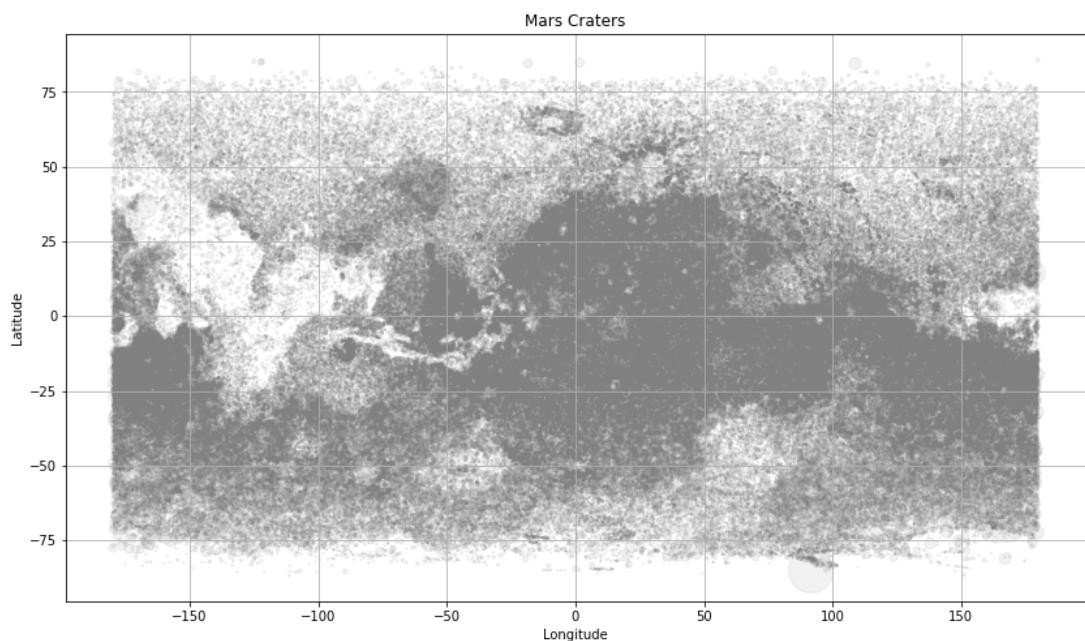
가설 “분화구에 분출물 성분이 많을 수록 층수가 더 많아질 것” YES 😎

▼ 화성 크레이터 맵 만들기

✓ 전체 크레이터 지도 만들기

전체 CREATOR 지도

```
plt.figure(figsize=(14,8))
plt.scatter(x=df.LONGITUDE_CIRCLE_IMAGE,
            y=df.LATITUDE_CIRCLE_IMAGE,
            s=df.DIAM_CIRCLE_IMAGE,
            color='sienna1',
            alpha=0.1)
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid()
plt.title('Mars Craters')
plt.show()
```



➡ 분석 결과 상당히 문화구가 고르게 분포되어있음

 분화구 이름이 있고 분화구 100km 이상 지도

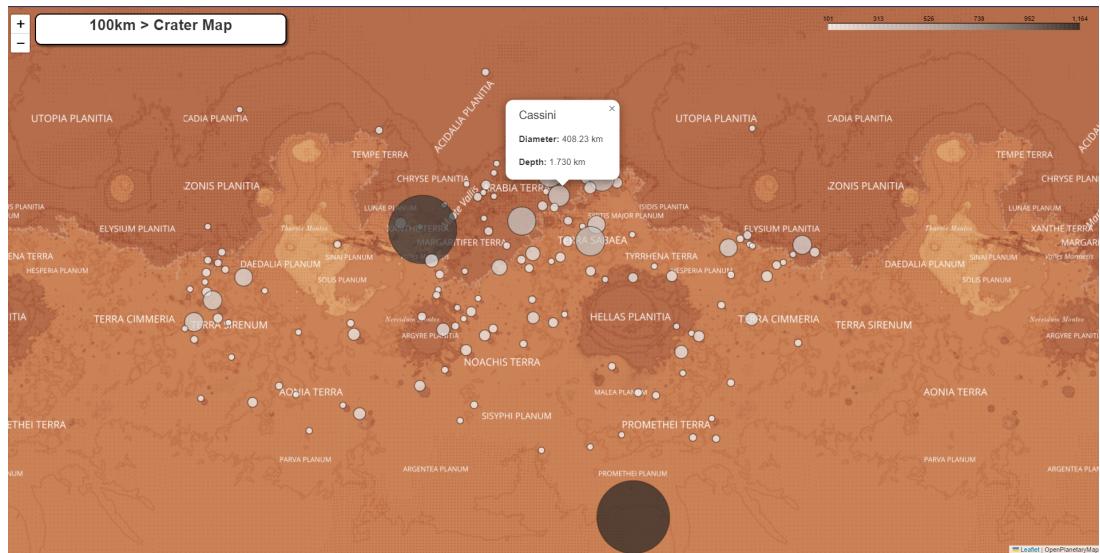
```

<div style="background-color:{colormap(row['DIA'
    """,
    icon_size=(icon_size, icon_size)
)
folium.Marker(
    location=[row['LATITUDE_CIRCLE_IMAGE'], row['LO
    icon=icon,
    popup=popup
).add_to(map1)

colormap.add_to(map1)

map1.save('craters_above_100km.html')

```



💿 화성 대기질 데이터

▼ 데이터 EDA

1. 데이터 구조 확인 - 화성 지면 온도, 공기중 온도, 기압 존재 확인

```

import pandas as pd

# 데이터 파일 경로
file_path = 'C:\\\\Users\\\\JY\\\\Desktop\\\\jy\\\\4grade\\\\수업\\\\데이터\\'

```

```

# 데이터 불러오기
data = pd.read_csv(file_path)

# "Value not available"을 결측치로 처리
data = data.replace("Value not available", pd.NA)

# 데이터 타입 변환
data = data.convert_dtypes()

# 데이터 구조 확인
print(data.head())
print(data.info())
print(data.describe())

```

```

      earth_date_time      mars_date_time sol_number \
0 Earth, 2022-01-26 UTC Mars, Month 6 - LS 163°   Sol 3368
1 Earth, 2022-01-25 UTC Mars, Month 6 - LS 163°   Sol 3367
2 Earth, 2022-01-24 UTC Mars, Month 6 - LS 162°   Sol 3366
3 Earth, 2022-01-23 UTC Mars, Month 6 - LS 162°   Sol 3365
4 Earth, 2022-01-22 UTC Mars, Month 6 - LS 161°   Sol 3364

      max_ground_temp(°C) min_ground_temp(°C) max_air_temp(°C) min_air_temp(°C) \
0                  -3                 -71                  10                 -84
1                  -3                 -72                  10                 -87
2                  -4                 -70                   8                 -81
3                  -6                 -70                   9                 -91
4                  -7                 -71                   8                 -92

      mean_pressure(Pa) wind_speed(m/h) humidity(%) sunrise sunset UV_Radiation \
0                  707           <NA>       <NA>    05:25 17:20   moderate
1                  707           <NA>       <NA>    05:25 17:20   moderate
2                  708           <NA>       <NA>    05:25 17:21   moderate
3                  707           <NA>       <NA>    05:26 17:21   moderate
4                  708           <NA>       <NA>    05:26 17:21   moderate

      weather
0  Sunny
1  Sunny
2  Sunny
3  Sunny
4  Sunny
<class 'pandas.core.frame.DataFrame'>

```

```

RangeIndex: 3197 entries, 0 to 3196
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
--- 
 0   earth_date_time    3197 non-null   string  
 1   mars_date_time     3197 non-null   string  
 2   sol_number          3197 non-null   string  
 3   max_ground_temp(°C) 3169 non-null   string  
 4   min_ground_temp(°C) 3169 non-null   string  
 5   max_air_temp(°C)    3168 non-null   string  
 6   min_air_temp(°C)    3168 non-null   string  
 7   mean_pressure(Pa)   3170 non-null   string  
 8   wind_speed(m/h)    0 non-null     object  
 9   humidity(%)        0 non-null     object  
 10  sunrise             3197 non-null   string  
 11  sunset              3197 non-null   string  
 12  UV_Radiation       3170 non-null   string  
 13  weather             3194 non-null   string  
dtypes: object(2), string(12)
memory usage: 349.8+ KB
None
      earth_date_time      mars_date_time  sol_number \
count           3197                  3197      3197
unique          3197                  360      3197
top      Earth, 2022-01-26 UTC  Mars, Month 3 - LS 86°  Sol 3368
freq                 1                   12         1

      max_ground_temp(°C)  min_ground_temp(°C)  max_air_temp(°C) \
count           3169                  3169          3168
unique          51                     39          50
top            -6                    -79            6
freq            130                  221          154

      min_air_temp(°C)  mean_pressure(Pa)  wind_speed(m/h)  humidity(%)  sunrise \
count           3168                  3170          0          0      3197
unique          77                     223          0          0      90
top            -78                  873        NaN        NaN  05:18
freq            165                  47        NaN        NaN      170

      sunset  UV_Radiation  weather
count      3197          3170      3194
unique        96             4          1
top      17:20      moderate    Sunny
freq        199          1450      3194

```

- wind_speed와 humidity는 컬럼만 존재하고 값이 없는것으로 보여짐
- non-null을 출력했지만 컬럼의 개수가 다른것으로 보아 결측치 혹은 이상치가 존재한다고 판단

2. 결측치 처리

2-1. 데이터 정제 - 풍속, 습도 삭제 과정

- 삭제하는 이유?

```
[3]: # "Value not available"을 결측값(NaN)으로 변환
data = data.replace("Value not available", pd.NA)

# 결측값 비율 확인
missing_values_ratio = data.isna().sum() / len(data)
print(missing_values_ratio)

earth_date_time      0.000000
mars_date_time       0.000000
sol_number           0.000000
max_ground_temp(°C) 0.008758
min_ground_temp(°C) 0.008758
max_air_temp(°C)    0.009071
min_air_temp(°C)    0.009071
mean_pressure(Pa)   0.008445
wind_speed(m/h)     1.000000
humidity(%)          1.000000
sunrise              0.000000
sunset               0.000000
UV_Radiation         0.008445
weather              0.000938
dtype: float64
```

- 결측치를 확인한 결과 컬럼의 모든 값이 결측치인 것을 확인
→ 컬럼 자체를 삭제 결정

2-2. UV_Radiation 값 결측치 대체

- UV_Radiation에는 결측치가 27개 존재

```
Missing values after filling:
earth_date_time      0
mars_date_time       0
sol_number           0
max_ground_temp(°C) 0
min_ground_temp(°C) 0
max_air_temp(°C)    0
min_air_temp(°C)    0
mean_pressure(Pa)   0
sunrise              0
sunset               0
UV_Radiation         27
```

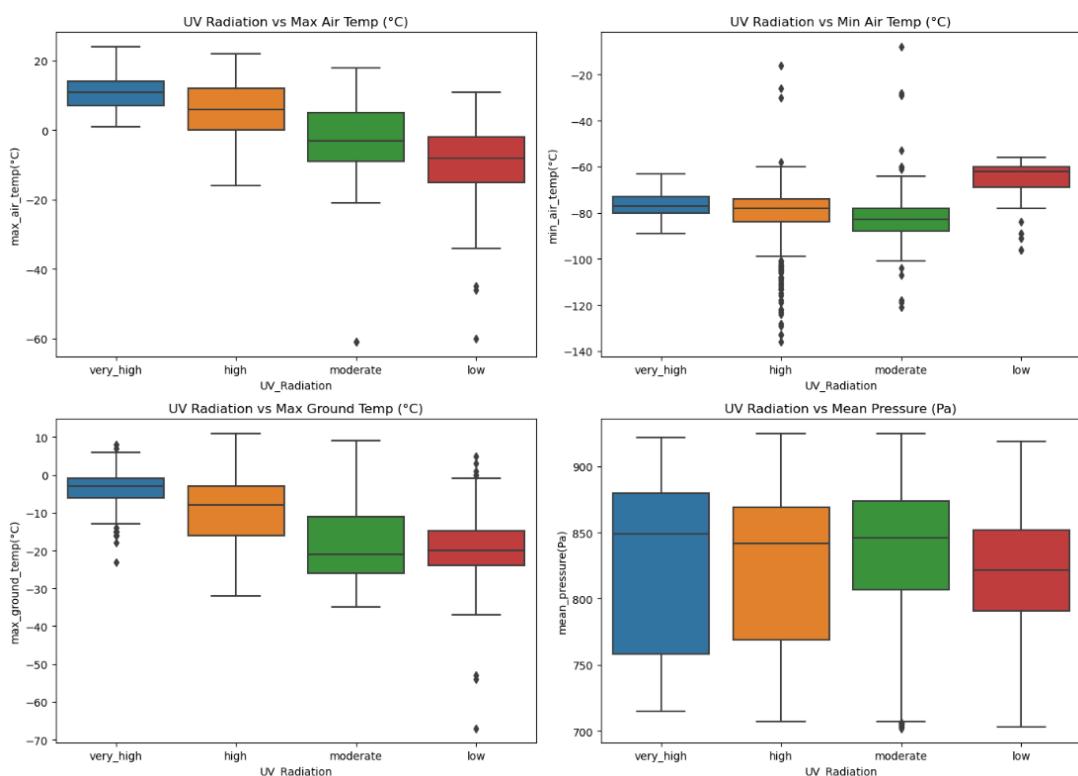
- 또한, UV_Radiation 값은 문자열로 되어있음

<input checked="" type="checkbox"/> (모두)	(3197)
<input checked="" type="checkbox"/> high	(1366)
<input checked="" type="checkbox"/> low	(84)
<input checked="" type="checkbox"/> moderate	(1463)
<input checked="" type="checkbox"/> very_high	(284)

- 모델을 돌리기 위해 변수를 수치형으로 변환

```
# UV_Radiation 변수를 수치형으로 변환
uv_radiation_mapping = {'low': 1, 'moderate': 2, 'high': 3, 'very_high': 4}
data_cleaned['UV_Radiation_numeric'] = data_cleaned['UV_Radiation'].map(uv_radiation_mapping)
```

- 다른 변수와의 관계를 알아보기 위한 BOX Plot



1. UV_Radiation과 Max Air Temp

- low 일 때 중앙값이 -20°C
- very high 일 때 중앙값이 20°C

→ Max Air Temp가 높을수록 UV_Radiation 값이 높아짐

2. UV_Radiation과 Min Air Temp

- low 일 때 중앙값이 -60°C

- very high 일 때 중앙 값이 -80°C

→ Min Air Temp가 낮을수록 UV_Radiation이 높아짐

3. UV_Radiation과 Max Ground Temp

- low 일 때 중앙값 -30°C
- very high 일 때 중앙값 0°C

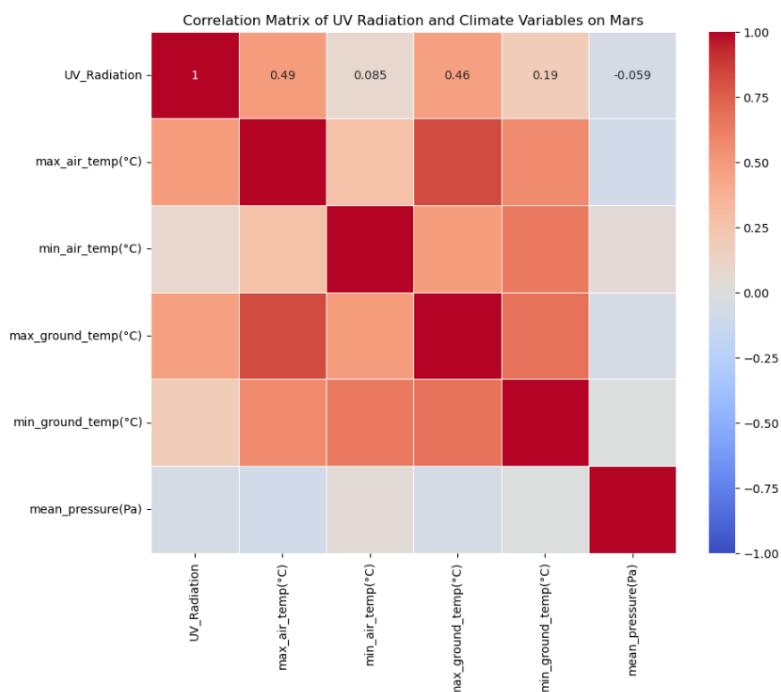
→ Max Ground Temp가 높을수록 UV_Radiation이 높아짐

4. UV Radiation과 Mean Pressure (Pa)

- low 일 때 중앙값은 750Pa
- very high 일 때 중앙값은 850Pa

→ Mean Pressure 가 높을수록 UV_Radiation 값이 높아짐

- UV_Radiation의 상관관계 계수



- max_air_temp가 가장 관계가 높은 것으로 파악

**해당 결과를 기반으로 랜덤포레스트 모델을 사용하여
UV_Radiation 결측치 예측**

```

precision    recall   f1-score   support
          1.0      1.00      0.75      0.86      20
          2.0      0.79      0.86      0.82     264
          3.0      0.78      0.77      0.77     282
          4.0      0.79      0.60      0.68      68

accuracy           0.79      634
macro avg       0.84      0.75      0.78      634
weighted avg     0.79      0.79      0.79      634

UV_Radiation missing values after filling: 0
    max_air_temp(°C)  min_air_temp(°C)  max_ground_temp(°C)  mean_pressure(Pa) \
0                  2                 -79                  -14                845
1                  2                 -79                  -14                845
2                  8                 -83                  -16                739
3                  9                 -83                  -11                740
4                  8                 -82                  -18                741

```

- 예측 결과 전체적인 정확도는 79%로 very high를 제외한 나머지 컬럼은 양호

→ 대체된 값에 very high 값이 없으므로 예측한 결과로 결측치 대체 결정

2-3. 온도, 습도, 기압 결측치 대체

- 월별 평균 값으로 대체

```

Missing values before filling:
earth_date_time      0
mars_date_time        0
sol_number            0
max_ground_temp(°C)  28
min_ground_temp(°C)  28
max_air_temp(°C)     29
min_air_temp(°C)     29
mean_pressure(Pa)    27
sunrise               0
sunset                0
UV_Radiation         27
weather               3
month                0
dtype: int64

```

- 대체한 결과

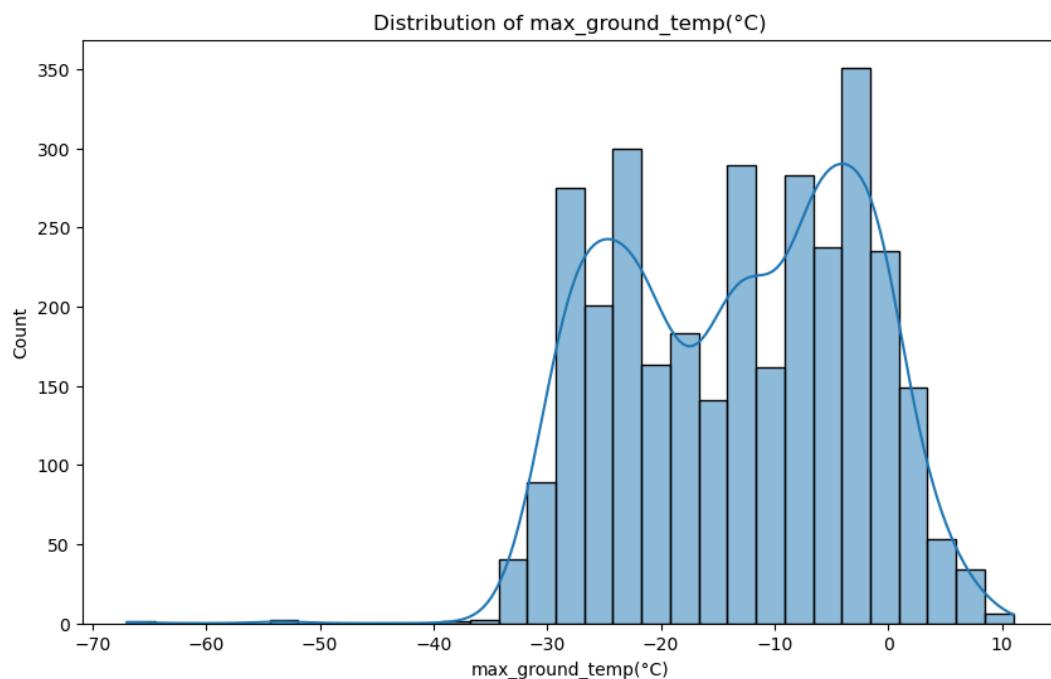
```

Missing values after filling:
earth_date_time          0
mars_date_time            0
sol_number                0
max_ground_temp(°C)      0
min_ground_temp(°C)      0
max_air_temp(°C)          0
min_air_temp(°C)          0
mean_pressure(Pa)         0
sunrise                   0
sunset                    0
UV_Radiation              27
weather                   3
month                     0
dtype: int64

```

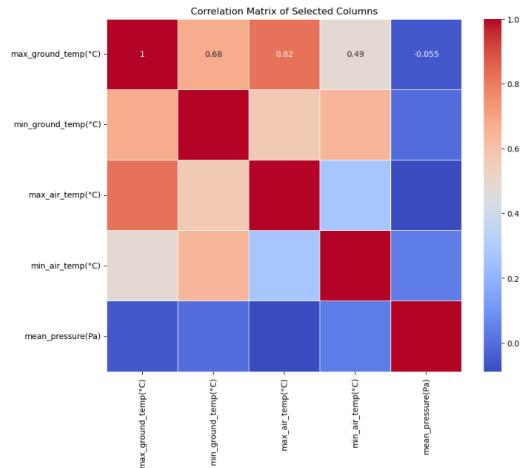
3. 데이터 분포 확인

- max_ground_temp

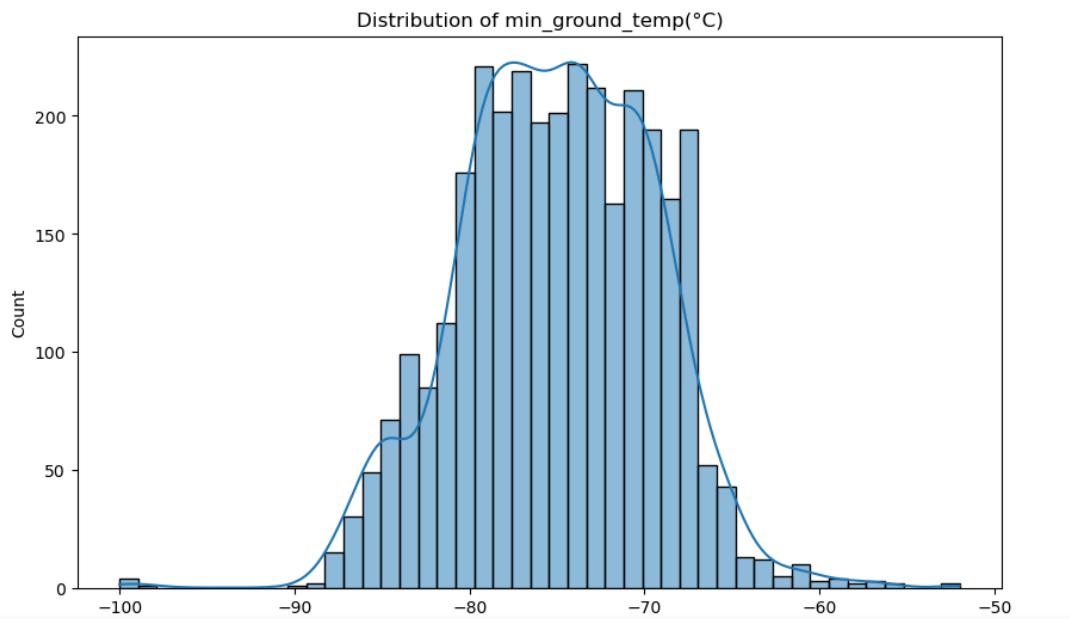


- 평균 : -13.21°C , 분산 : 109.16, 표준편차 : 10.45

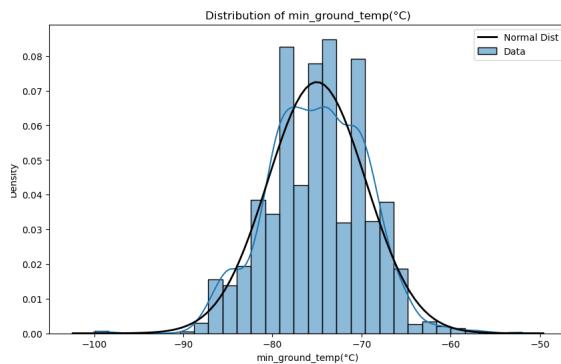
1. 화성의 최고온도의 평균은 -13°C 로 출다는 것을 확인할 수 있음
2. 표준 편차가 10.45로 화성 지면 온도는 변동성을 크게 띤
3. 정규 분포를 따르지 않음



- 지면 최고 온도는 대기 중 최고 온도에 영향을 가장 많이 받음
- 온도와 기압은 비교적 낮은 상관 관계를 보임
- **min_ground_temp**

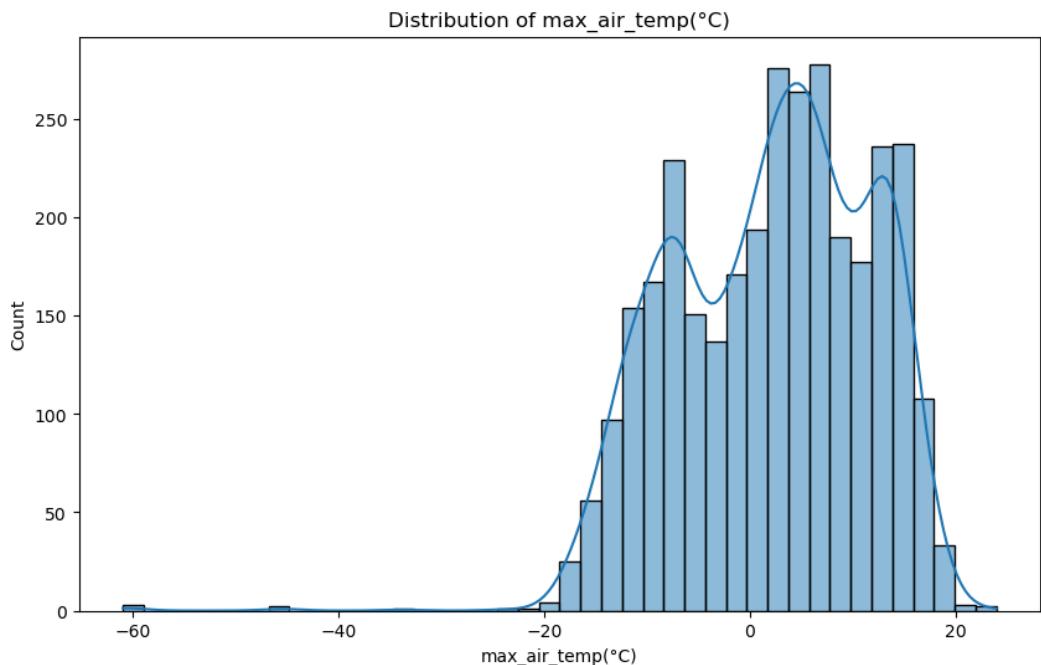


- 평균 : -75.02, 분산 : 30.31, 표준 편차 : 5.5
 1. 화성의 최저 온도는 평균 -75도로 매우 낮다는 것을 알 수 있음
 2. 검정선은 정규분포인데 대체로 정규분포를 따르는 것처럼 보임



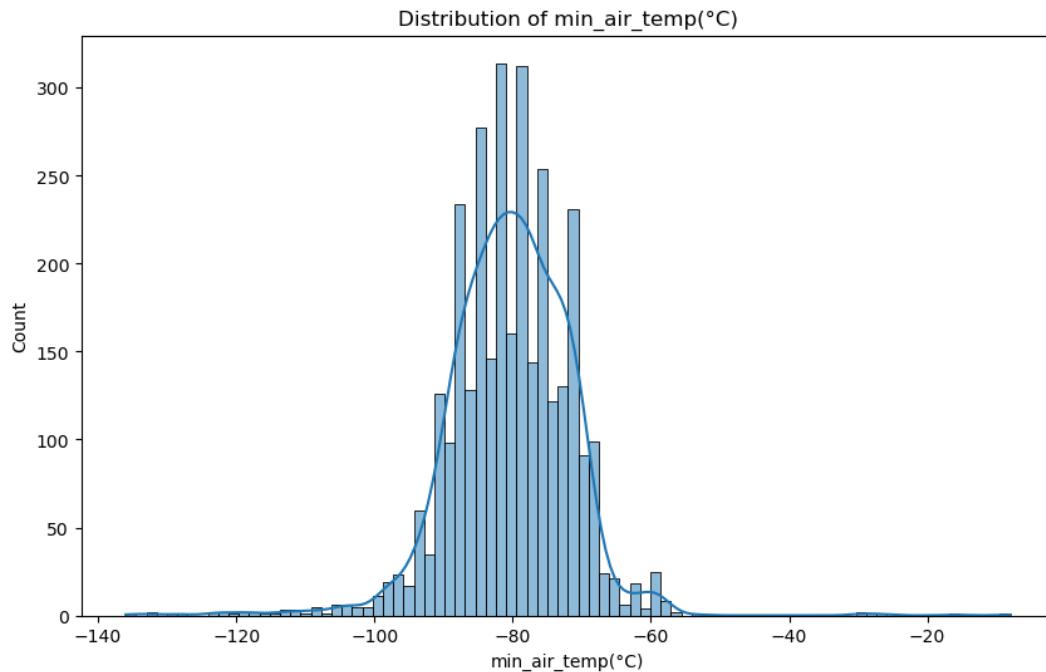
3. 정규성 검정

- Kolmogorov-Smirnov Test
- 통계량 : 0.053, p-value : 2.32e-08
→ p-value가 너무 작아 정규분포는 따르지 않음
- **max_air_temp**

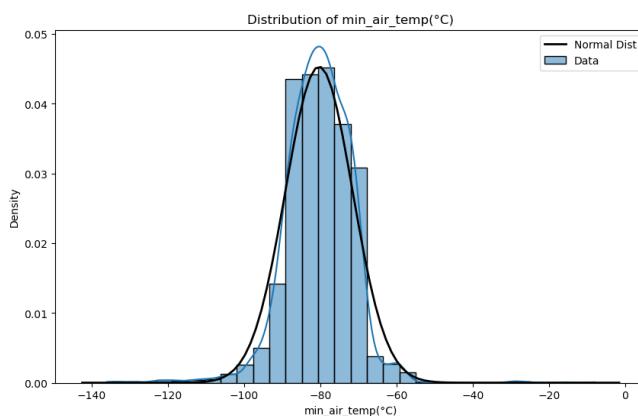


- 평균 : 2.02, 분산 : 87.60, 표준 편차 : 9.35
 - 공기 중 최고 온도의 평균은 2.02로 낮은편에 속하고 분산이 87로 평균 주위 값으로 온도가 이뤄지는 것을 알 수 있음
- max air temp는 정규 분포 형태를 따르지 않음

- **min_air_temp**



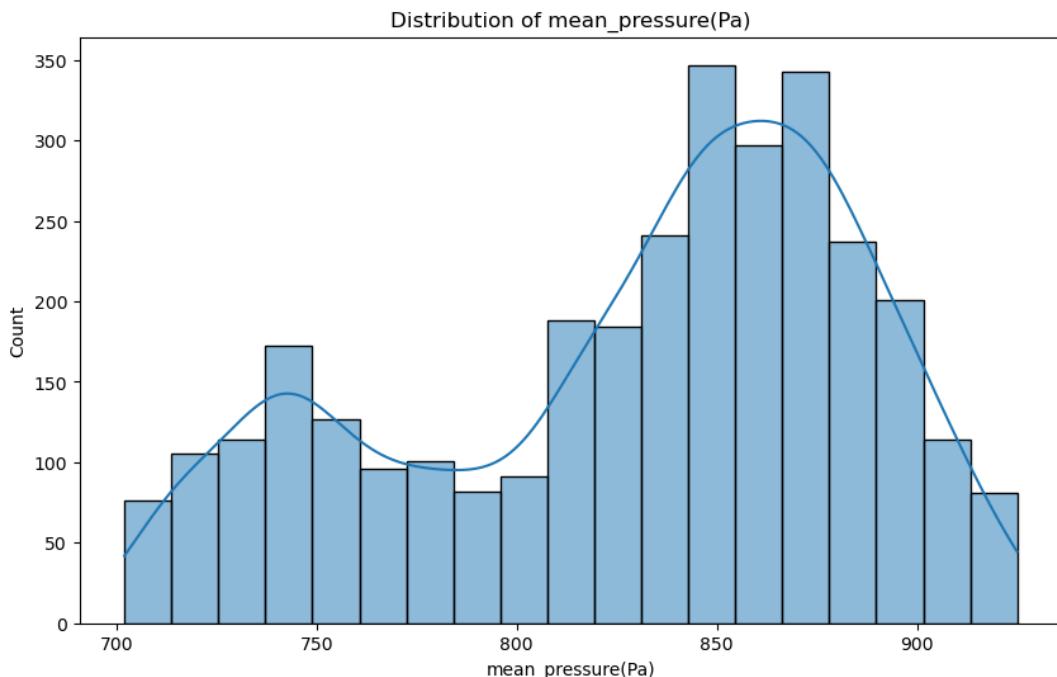
- 평균 -83.08, 분산 : 67.51, 표준 편차 : 8.22
- 화성의 최저 온도의 평균은 -83으로 매우 추운 것으로 확인
- 표준 편자는 8.22로 최저 기온이 영하에서 머무른다는 것을 확인 → 사람이 살 수 없음
- 정규성을 띠는 듯 하지만 정규성 검정을 했을 때 p-value 값이 매우 낮게 나와서 정규 분포를 따르지 않음



Kolmogorov-Smirnov Test:
Statistics=0.05499938116219416,

p=7.58841011245684e-09

- **mean_pressure**



- 평균 : 828.79 분산 : 3237.62, 표준 편차 : 56.90
- 지구의 평균 기압에 비해 화성의 기압은 매우 낮은 것을 알 수 있음

4. 변수 간 관계 분석 및 가설 세우기

- 확인한 데이터 분포를 바탕으로 변수 간의 관계를 분석하고 가설을 세움

가설 1. 화성의 낮은 기압은 온도 변화의 폭을 크게 만든다.

1. 온도 변화의 폭을 계산

- 지면 온도 변화 폭(ground_temp_range) = max_ground_temp - min_ground_temp
- 대기 온도 변화 폭(air_temp_range) = max_air_temp - min_air_temp

```
# 온도 변화 폭 계산
data['ground_temp_range'] = data['max_ground_temp(°C)'] - data['min_ground_temp(°C)']
data['air_temp_range'] = data['max_air_temp(°C)'] - data['min_air_temp(°C)']
```

2. 기압에 따른 온도 변화의 폭을 분석

- 기압을 여러 구간으로 나눔 - low, midium, high로 구분
- 각 기압의 구간별 온도 변화 폭의 통계를 분석
 - 평균, 표준편차, 분산을 계산
 - 시각화를 통해 구간별 온도 변화 폭의 분포 확인

```
# 기압 범주화
data['pressure_category'] = pd.cut(data['mean_pressu

# 기압 범주별 온도 변화 폭 통계 분석
pressure_groups = data.groupby('pressure_category').  

    'ground_temp_range': ['mean', 'std', 'var'],
    'air_temp_range': ['mean', 'std', 'var']
})
print(pressure_groups)
```

ground_temp_range		mean	std
var	pressure_category		
	low	64.246939	5.598919
1.347894			3
	medium	60.966825	10.368912
7.514336			10
	high	61.717949	7.863203
1.829957			6
	very_high	61.260086	7.493093
6.146437			5
air_temp_range		mean	
std	var		
pressure_category			
	low	83.602041	10.108349
2.178728			10
	medium	80.447867	12.874172
16			

```

5.744307
high           85.221851 11.617742 13
4.971933
very_high      80.587176 9.745812 9
4.980859

```

3. 통계적 검정

- 상관 분석
 - 기압과 온도 변화 폭 간의 상관관계를 계산
 - mean_pressure와 지면 온도 변화 폭, 대기 온도 변화 폭에 대한 피어슨 상관계수 계산

```

# 상관 분석
corr_ground_temp_range, _ = pearsonr(data['mean_pres
corr_air_temp_range, _ = pearsonr(data['mean_pressur
print(f'Correlation between pressure and ground temp
print(f'Correlation between pressure and air tempera

```

```

(print)
Correlation between pressure and ground temperature
Correlation between pressure and air temperature ran

```

- 기압이 낮을수록 온도 변화 폭이 증가하는 경향이 있지만 상관 관계는 매우 약함

→ **기압은 온도 변화에 영향을 크게 주지 못함**

- 회귀 분석
 - 기압을 독립 변수로 두고 온도 변화 폭을 종속 변수로 설정하여 회귀 분석 진행
 - 기압이 온도 변화 폭에 미치는 영향을 확인함

```

# 회귀 분석
X = sm.add_constant(data['mean_pressure(Pa)'])

```

```

y_ground = data['ground_temp_range']
y_air = data['air_temp_range']

model_ground = sm.OLS(y_ground, X).fit()
model_air = sm.OLS(y_air, X).fit()

print(model_ground.summary())
print(model_air.summary())

```

	coef	std err	t	P> t	[0.025	0.975]
const	69.8133	2.026	34.467	0.000	65.842	73.785
mean_pressure(Pa)	-0.0097	0.002	-3.962	0.000	-0.014	-0.005

- 회귀 분석 결과 1Pa 기압이 증가할 때 지표면 온도가 0.0097°C 감소하는 것으로 나타남
- 기압이 지상 온도에 미치는 영향이 통계적으로는 유의미하다고 하지만 매우 작은 영향력임

→ **기압은 지상 온도 변화에 많은 영향을 주지 못함**

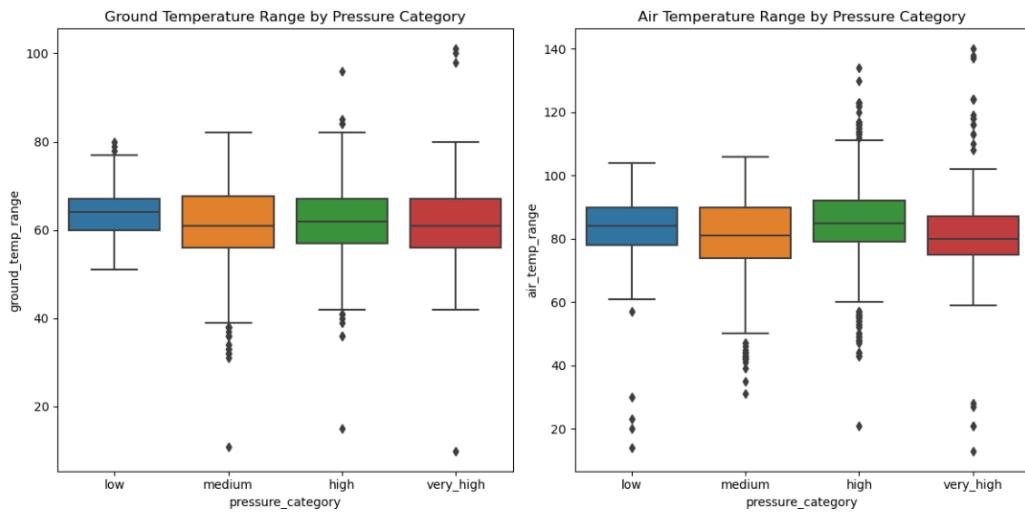
	coef	std err	t	P> t	[0.025	0.975]
const	98.5381	2.826	34.873	0.000	92.998	104.078
mean_pressure(Pa)	-0.0196	0.003	-5.749	0.000	-0.026	-0.013

- 회귀 분석 결과 1Pa 기압이 증가할 때 공기 온도는 0.0196°C 감소하는 것으로 나타남
- 기압이 공기 온도에 미치는 영향은 통계적으로는 유의미하지만 영향이 매우 작음

→ **기압은 공기 온도 변화에 많은 영향을 주지 못함**

4. 결과 해석 및 결론 도출

- 기압 구간별 온도 변화 폭의 BoxPlot으로 시각화



- 기압에 따른 온도 변화 폭에 대한 차이는 거의 없음

→ **기압이 워낙 낮아서 영향력이 없어보임**

→ 결론 : 화성의 낮은 기압은 온도 변화의 폭을 크게 만들지 않는다.

- 기압과 온도 폭 사이에 음의 상관 관계가 존재하고 회귀 분석 결과에서도 기압이 낮을수록 온도 폭이 증가하는 경향이 있지만, 매우 미미한 수준이기 때문에 기압은 온도 변화의 폭에 영향이 없는 것으로 판단됨.



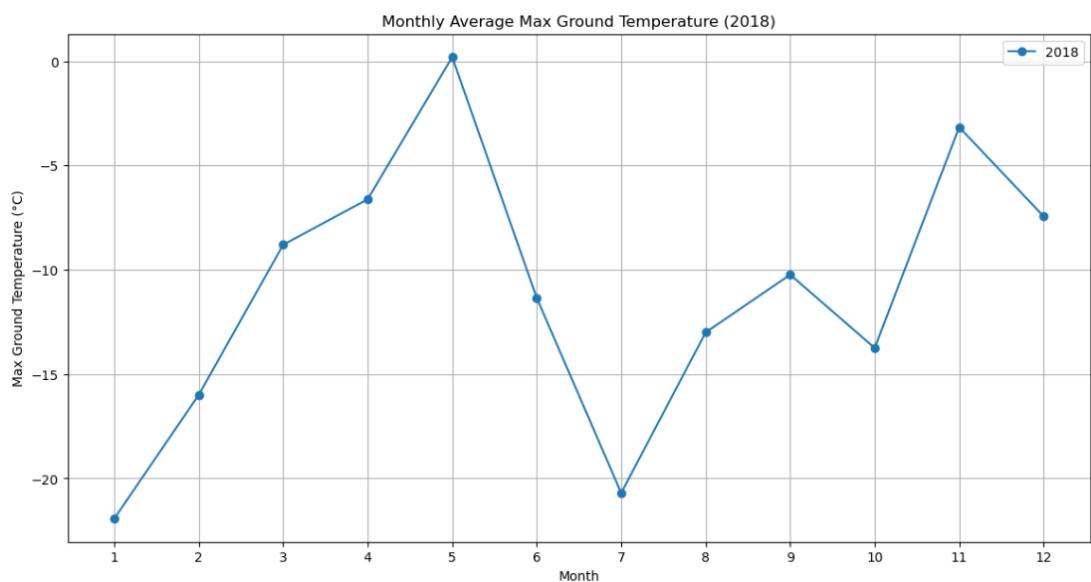
그렇다면 화성의 온도 변화가 큰 이유는 무엇일까?

ChatGPT 4o 답변:

- **얇은 대기:** 열을 보존하는 능력이 부족하여 낮에는 뜨거워지고 밤에는 급격히 식음.
- **대기의 구성:** 주로 이산화탄소로 구성되어 온실 효과가 제한적.
- **자전축 기울기:** 지구와 비슷한 자전축 기울기로 인해 계절 변화와 함께 온도 변동.
- **낮은 열용량:** 표면의 암석과 먼지가 열을 빨리 흡수하고 방출.
- **타원형 공전 궤도:** 태양과의 거리 변화로 인한 태양 복사 에너지의 변동.
- **극심한 낮과 밤의 온도 차이:** 자전 주기가 지구와 유사하지만 열 보존이 부족하여 큰 온도 차이를 초래.

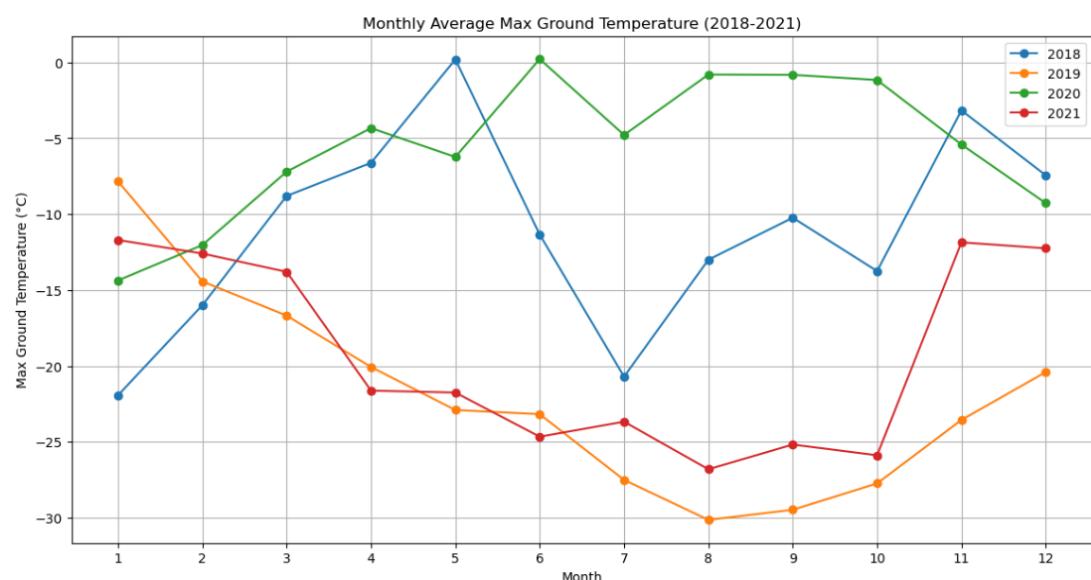
가설 2. 화성의 온도 변화는 우리나라의 온도 변화 주기와 비슷하다.

- 화성의 1년치 월별 지표면 온도를 그려보니 다음과 같이 그려짐



→ 온도 변화가 6개월마다 심한 것으로 추정, 우리 나라와 비슷하다고 생각하여 가설을 세우게 됨

- 화성의 2018년부터 2021년까지의 Max_Ground_Temperature를 월별로 나타낸 결과



- 화성은 2년 주기로 온도 변화가 크게 나타남
- 우리나라의 사계절 그래프



[https://data.kma.go.kr/climate/average30Years/selectAverage30YearsKoreaList.do?
pgmNo=188](https://data.kma.go.kr/climate/average30Years/selectAverage30YearsKoreaList.do?pgmNo=188)

- 우리나라는 그래프가 일정하게 나타남
 - 우리나라는 온도 변화가 3개월마다 일어나지만 화성은 2년을 주기로 온도 변화가 일어남
- 화성의 온도는 우리나라의 온도 변화 주기가 비슷하지 않음.



화성의 온도 변화는 어떻게 이루어질까요?

ChatGPT 4o :

- **일교차 (Diurnal Cycle):**

- 화성의 하루(일주기)는 약 24.6시간으로 지구와 비슷합니다.
- 그러나 화성의 얇은 대기로 인해 낮과 밤의 온도 변화가 극심합니다. 낮 동안 태양열로 인해 온도가 급격히 상승하고, 밤에는 대기가 열을 보존하지 못해 온도가 급격히 하락합니다.
- 예를 들어, 화성의 적도 지역에서 낮 기온은 20°C 까지 오를 수 있지만, 밤에는 -70°C 이하로 떨어질 수 있습니다.

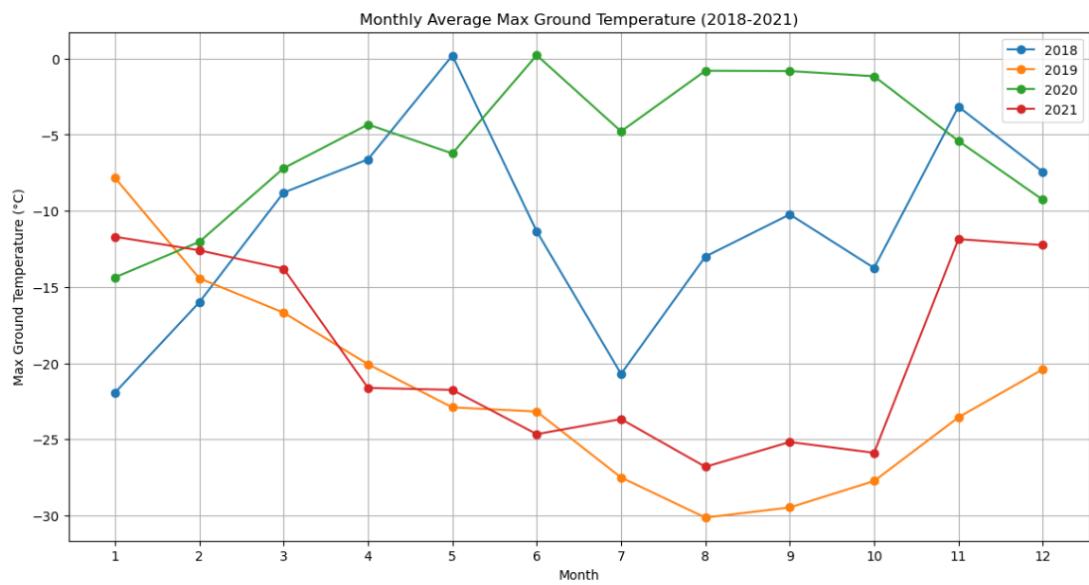
- **계절 변화 (Seasonal Cycle):**

- 화성의 자전축 기울기는 약 25도이며, 이는 지구와 유사하여 계절 변화를 경험합니다.
- 화성의 공전 주기는 약 687일로 지구의 약 1.88배입니다. 따라서 화성의 각 계절은 지구의 두 배 정도 길게 지속됩니다.
- 화성의 타원형 궤도로 인해 계절에 따른 태양 에너지의 변화가 더 극심합니다. 이는 특히 남반구에서 여름이 짧고 매우 뜨거우며, 겨울이 길고 매우 추운 결과를 초래합니다.

가설 3. 화성에 인간이 생존할 수 있는 시기가 존재한다.

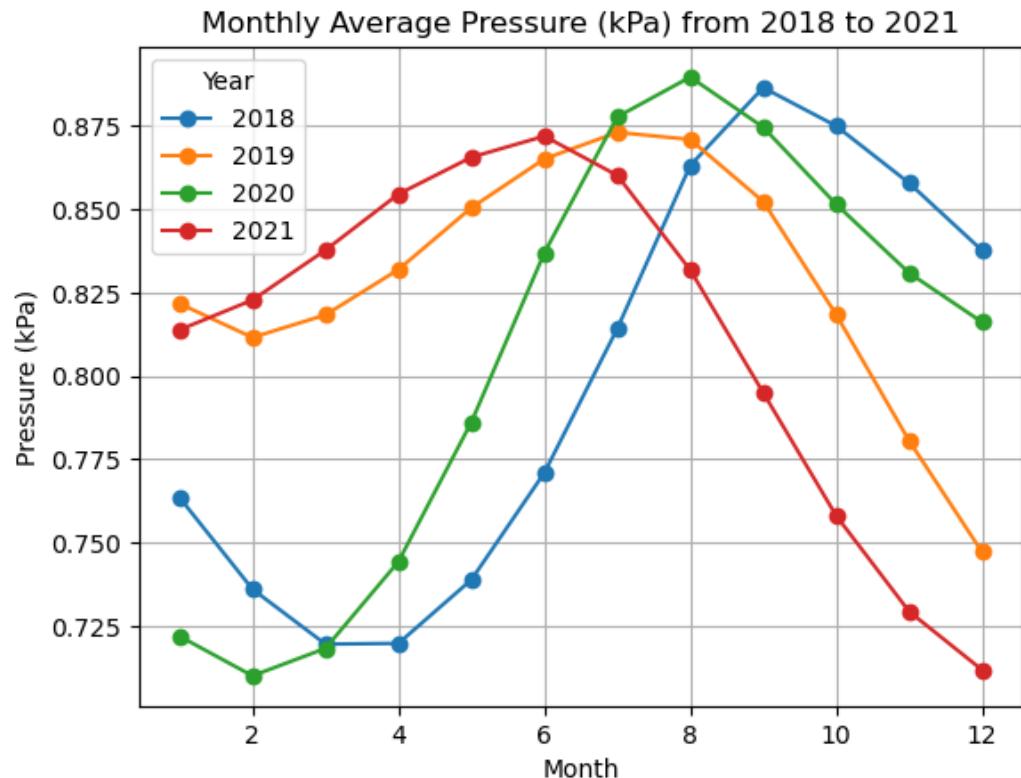
- 화성의 기후변화를 보면 사람이 살 수 있을 것 같은 시기가 있는 것 같아 가설을 세움

1. 온도 변화 분석



- 사람이 살 수 있는 최저 기온은 -10°C 로 2020년, 2018년도에 살 수 있는 기간이 보임

2. 기압 변화 분석

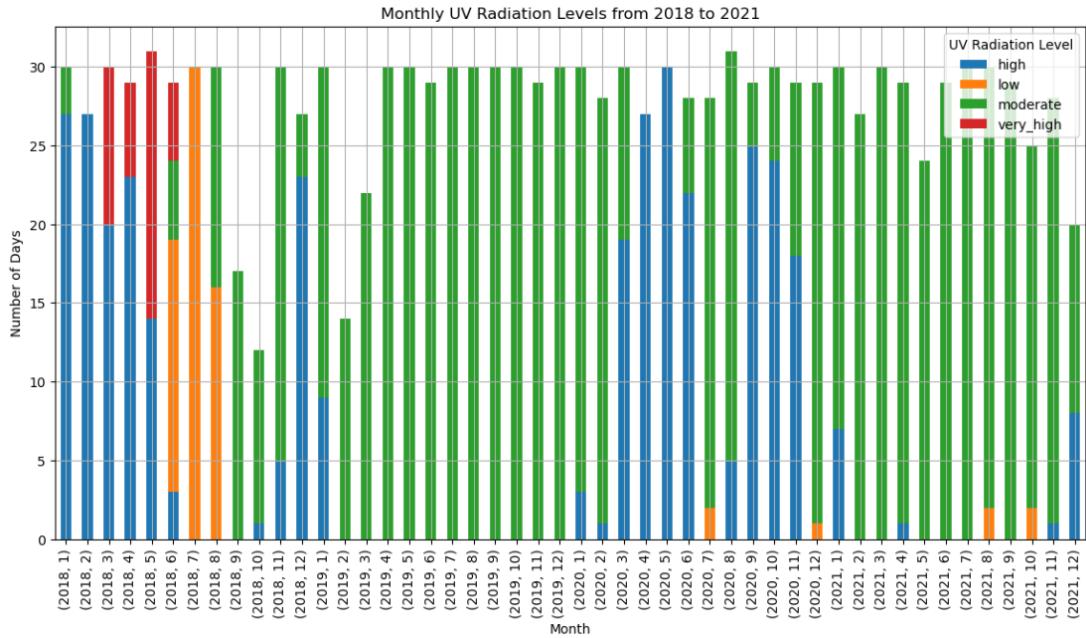


- 사람이 살 수 있는 기압은 60kpa ~ 103kpa라고 하는데 화성은 0.8부터 시작됨..

- 맨 몸으로는 절대 살 수 없는 기압으로 측정됨.

→ 사람이 살 수 있는 시기가 없다고 나왔지만 다른 요인들도 추가 분석 진행

3. 자외선 방사선 분석



- 사람이 살 수 있는 자외선은 low,moderate로 2018년 7월~9월, 2019년 2월 ~2019년 12월로 측정됨

4. 결합 분석

- 최종적으로 인간이 화성에서 생존할 수 있는 시기를 파악

```

import pandas as pd
import matplotlib.pyplot as plt

# 데이터 파일 경로
file_path = 'C:\\\\Users\\\\JY\\\\Desktop\\\\jy\\\\4grade\\\\수업

# 데이터 불러오기
data = pd.read_csv(file_path)

# 필요한 컬럼 선택
data = data[['earth_date_time', 'max_air_temp(°C)',

# 날짜 형식 변환

```

```

data['earth_date_time'] = pd.to_datetime(data['earth_date_time'])

# Pa를 kPa로 변환
data['mean_pressure(kPa)'] = data['mean_pressure(Pa)']

# 연도와 월 컬럼 추가
data['year'] = data['earth_date_time'].dt.year
data['month'] = data['earth_date_time'].dt.month

# 결측값 처리
data = data.dropna()

# 온도 조건: -10°C <= max_air_temp <= 30°C
temperature_condition = (data['max_air_temp(°C)'] >= -10) & (data['max_air_temp(°C)'] <= 30)

# 기압 조건: 60 kPa <= mean_pressure <= 101.3 kPa
pressure_condition = (data['mean_pressure(kPa)'] >= 60) & (data['mean_pressure(kPa)'] <= 101.3)

# 자외선 방사선 조건: low 또는 moderate
uv_condition = data['UV_Radiation'].isin(['low', 'moderate'])

# 모든 조건을 만족하는 데이터 필터링
survivable_conditions = data[temperature_condition & pressure_condition & uv_condition]

# 조건 충족 데이터 개수 확인
print(f"Number of days with survivable conditions: {survivable_conditions.shape[0]}")

# 월별로 조건을 충족하는 데이터 개수 확인
survivable_conditions['month'] = survivable_conditions['month'].astype(str)
monthly_survivable_counts = survivable_conditions.groupby('month').size()

# 결과 출력
print(monthly_survivable_counts)

# 시각화
plt.figure(figsize=(14, 7))
monthly_survivable_counts.plot(kind='bar', stacked=True)
plt.title('Monthly Count of Days with Survivable Conditions')

```

```
plt.xlabel('Month')
plt.ylabel('Number of Days')
plt.grid(True)
plt.legend(title='Year')
plt.show()
```

→ Number of days with survivable conditions: 0

- 아직까지 사람은 화성에 생존할 수 없다고 결론 도출

→ 화성에 인간이 생존할 수 있는 시기가 존재하지 않는다.



화성에 인간이 생존하려면?

ChatGPT 4o

- 거주지 건설:

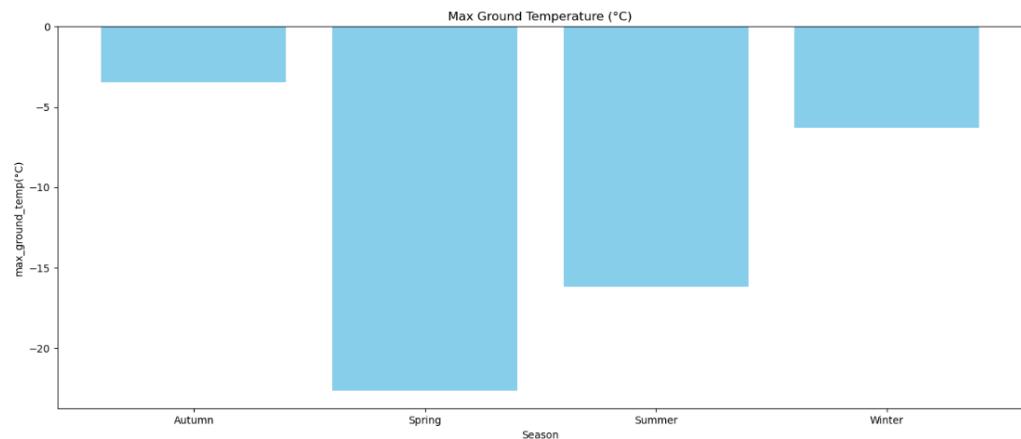
- 방사선과 극한의 온도 변화로부터 보호할 수 있는 **지하나
돔 형태**의 거주지 건설.
- 적절한 온도와 압력을 유지하는 생명 유지 시스템 구축.

가설 4. 화성에도 우리나라와 비슷한 사계절이 존재한다.

- 우리나라와 기후 변화의 주기는 다르지만 화성에도 기후 변화가 존재하므로 사계절이 존재할 수 있다.
- **화성의 사계절 계산**

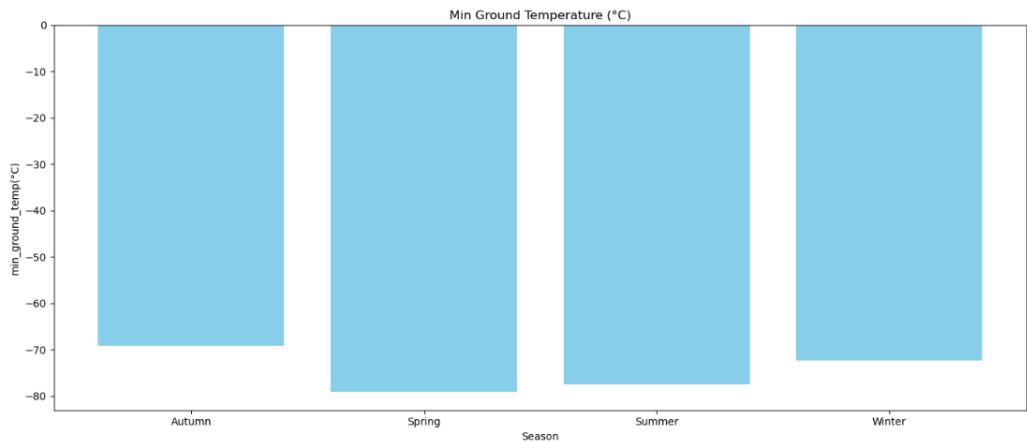
mars_date_time
Mars, Month 6 - LS 150°
Mars, Month 6 - LS 155°
Mars, Month 6 - LS 155°
Mars, Month 6 - LS 156°
Mars, Month 6 - LS 156°
Mars, Month 6 - LS 157°
Mars, Month 6 - LS 157°
Mars, Month 6 - LS 158°
Mars, Month 6 - LS 158°

- LS는 화성 중심 좌표계의 태양의 황경각으로 지구의 춘하추동이라고 생각하면 된다.
 - 0~90 사이가 봄, 90~180 사이가 여름, 180~270사이가 가을, 270~360 사이가 겨울
- 이를 통해 화성의 사계절 온도를 시각화 진행
 - **사계절 변화에 따른 지표면 최고 온도**



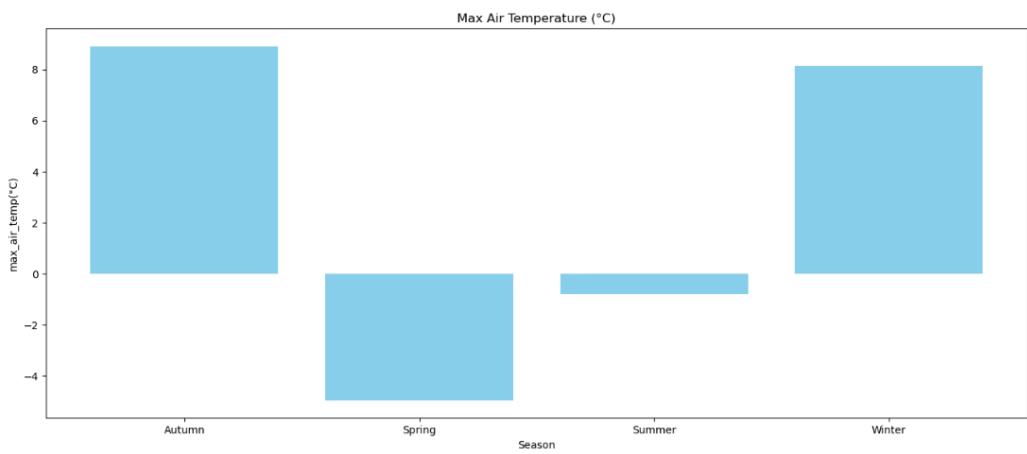
- 화성의 봄, 여름은 -15°C 가량으로 매우 추운것으로 나타남
 - 화성의 가을 겨울은 봄, 여름에 비해 비교적 따뜻한 것으로 나타남
- 우리나라와 정 반대의 성향을 지닌다는 것을 파악할 수 있음

- **사계절 변화에 따른 지표면 최저 온도**



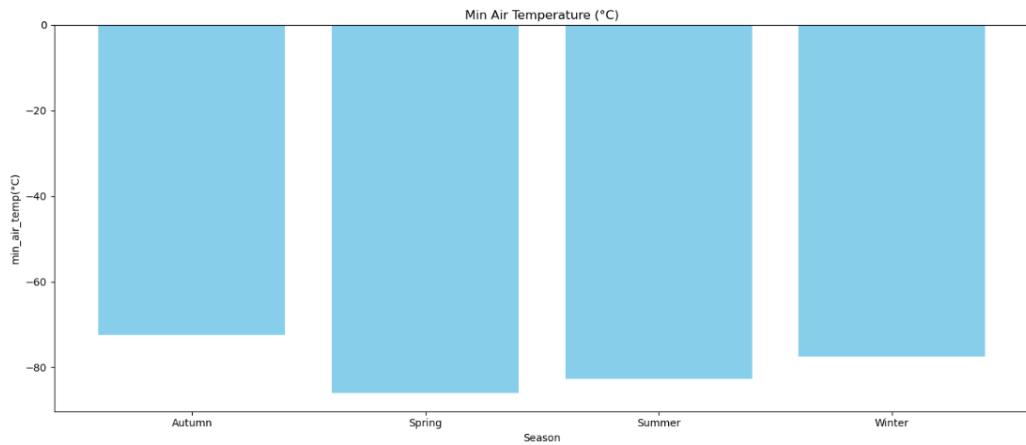
- 최저 온도는 사계절 가릴 것 없이 -70°C 가량으로 비슷한 온도를 가짐
→ 사계절은 존재하지만 열을 가지고 있을 수 없는 화성의 특성이 드러남

- **사계절 변화에 따른 공기중 최고 온도**



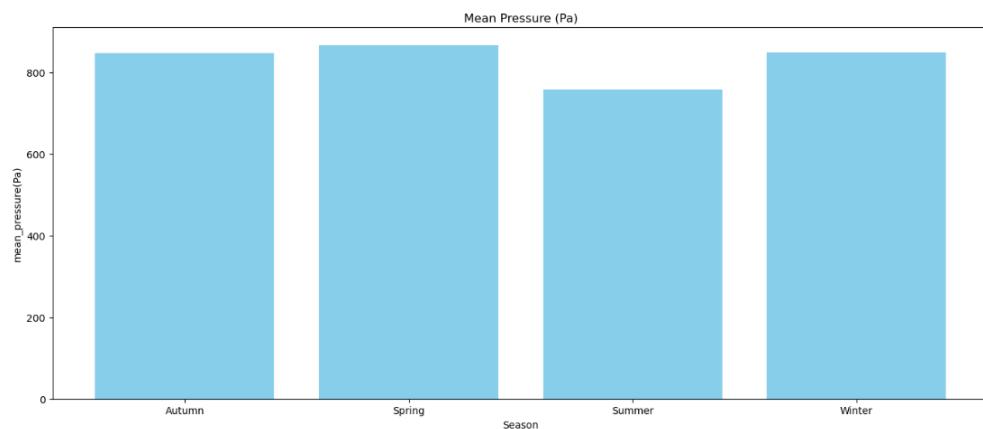
- 봄에는 -4°C , 여름에는 0°C 로 공기중 온도가 지표면 온도에 비해 따뜻함을 알 수 있음
- 가을, 겨울에는 8°C 가량으로 지구의 봄, 가을 날씨와 비슷하게 보임
→ 가을, 겨울에는 지구와 비슷한 환경이 구성 될 수 있다는 가능성이 존재함

- **사계절 변화에 따른 공기중 최저 온도**



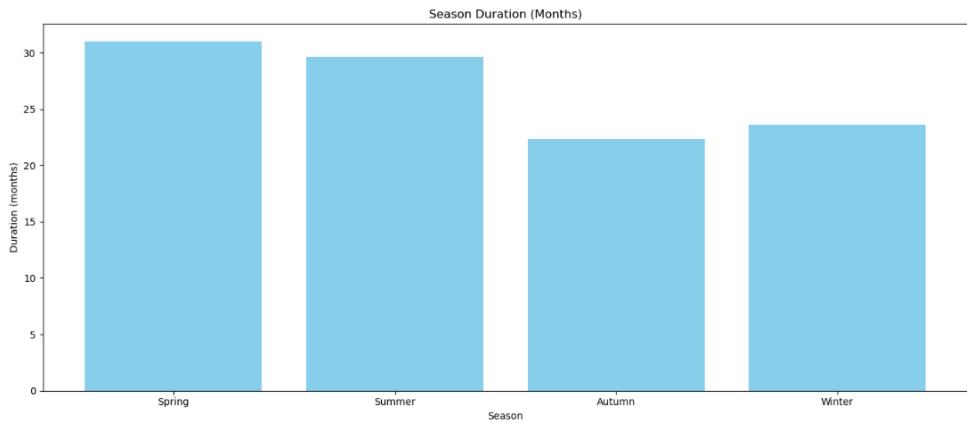
- 지표면 최저 온도와 다를 것 없이 최저 온도는 -70°C 가량으로 나타남
→ 위에서 분석했던 것처럼 지표면 온도와 공기중 온도는 서로 영향을 받기 때문에 지표면 온도가 차가워서 공기중 온도도 차가운 것으로 예상됨

■ 사계절 변화에 따른 기압



- 여름만 700대로 내려가게되고 봄, 가을, 겨울에는 800이상을 유지하는 것으로 보임
- 계절 변화에 따른 기압 변화는 별로 없는것으로 파악됨

■ 사용된 데이터의 계절별 기간



- 봄과 여름이 3개월 가량, 가을이 2개월, 겨울이 2.4개월 가량으로 측정됨
→ 실제 화성에서 1년동안 사계절의 기간은 봄이 7개월, 여름이 6개월, 가을이 5.3, 겨울이 4개월로 비슷한 수치를 보이는 것을 확인함
- 결론 : 화성에도 우리나라와 비슷한 사계절이 존재하지 않는다.
 - 우리나라와 정 반대의 온도인 사계절을 지니고 있음
 - 여름에는 춥고 가을 겨울이 오히려 따뜻한 상황임



지구와 화성이 비슷하다고 하는 이유는?

ChatGPT 4o:

1. 자전축 기울기

- **유사점:** 화성의 자전축 기울기는 약 25도이고, 지구의 자전축 기울기는 약 23.5도입니다. 이러한 유사한 자전축 기울기 때문에 두 행성 모두 계절 변화를 경험합니다.
- **의미:** 계절 변화는 다양한 기후 조건을 만들어내며, 생명체가 적응할 수 있는 다양한 환경을 제공합니다.

2. 자전 주기

- **유사점:** 화성의 하루 길이(자전 주기)는 약 24.6시간으로 지구의 하루 길이와 매우 유사합니다.
- **의미:** 지구와 유사한 하루 주기는 인간이 비교적 쉽게 적응할 수 있는 낮과 밤의 주기를 의미합니다.

→ 다양한 이유로 화성이 지구와 비슷하다고 하지만 자전축 기울기와 자전 주기가 유사하기 때문입니다. 하지만, 아직까지 화성과 지구 사이에는 공통점보단 차이점이 더 많으며 이러한 차이점을 극복하기 위한 기술과 연구가 필요합니다.



태양계 데이터

▼ 데이터 EDA

✓ 행성 데이터 분석

➡ 가설: 지구형 행성과 목성형 행성끼리 그룹으로 묶일 것임

🧑‍💻 실제 비교 분석

구분	지구형 행성	목성형 행성
	수성, 금성, 지구, 화성	목성, 토성, 천왕성, 해왕성

질량	작다	크다
반지름	작다	크다
밀도	크다	작다
크기	작다	크다
자전주기	길다	짧다
위성 수	없거나 적다	많다
고리 유무	없다	있다
표면 성분	흙이나 암석	기체(수소, 헬륨 등)
대기 성분	질소, 산소, 이산화탄소	수소, 헬륨, 메테인, 암모니아 등
편평도	작다	크다

1. 전체 컬럼을 넣어서 행성 간에 주성분 분석 진행

```

import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns
from mpl_toolkits.mplot3d import Axes3D

file_path = 'data/planets_updated.csv'
planets_data = pd.read_csv(file_path)

numerical_columns = [
    'Mass (10^24kg)', 'Diameter (km)', 'Density (kg/m^3)',
    'Surface Gravity(m/s^2)', 'Escape Velocity (km/s)',
    'Rotation Period (hours)', 'Length of Day (hours)',
    'Distance from Sun (10^6 km)', 'Orbital Period (days)'
    'Orbital Velocity (km/s)', 'Mean Temperature (C)',
    'Surface Pressure (bars)', 'Surface Temperature (C)'
]

object_columns = ['Orbital Period (days)', 'Surface Pressu
for column in object_columns:
    planets_data[column] = pd.to_numeric(planets_data[colu

```

```

X_filled = planets_data[numerical_columns].fillna(planets_)

pca_2d = PCA(n_components=2)
principal_components_2d = pca_2d.fit_transform(X_filled)

planets_data['PC1'] = principal_components_2d[:, 0]
planets_data['PC2'] = principal_components_2d[:, 1]

# Define colors for each planet
colors = {
    "Mercury": "gray", "Venus": "brown", "Earth": "blue",
    "Jupiter": "orange", "Saturn": "yellow", "Uranus": "lightblue",
    "Neptune": "darkblue"
}

plt.figure(figsize=(10, 6))
sns.scatterplot(data=planets_data, x='PC1', y='PC2', hue='Planet')
plt.title('2D PCA of Planetary Data')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend()
plt.grid()
plt.show()

pca_3d = PCA(n_components=3)
principal_components_3d = pca_3d.fit_transform(X_filled)
planets_data['PC3'] = principal_components_3d[:, 2]
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

for planet in planets_data['Planet'].unique():
    planet_data = planets_data[planets_data['Planet'] == planet]
    ax.scatter(planet_data['PC1'], planet_data['PC2'], planet_data['PC3'],
               label=planet, color=colors.get(planet, 'black'))

ax.set_title('3D PCA of Planetary Data')
ax.set_xlabel('Principal Component 1')
ax.set_ylabel('Principal Component 2')

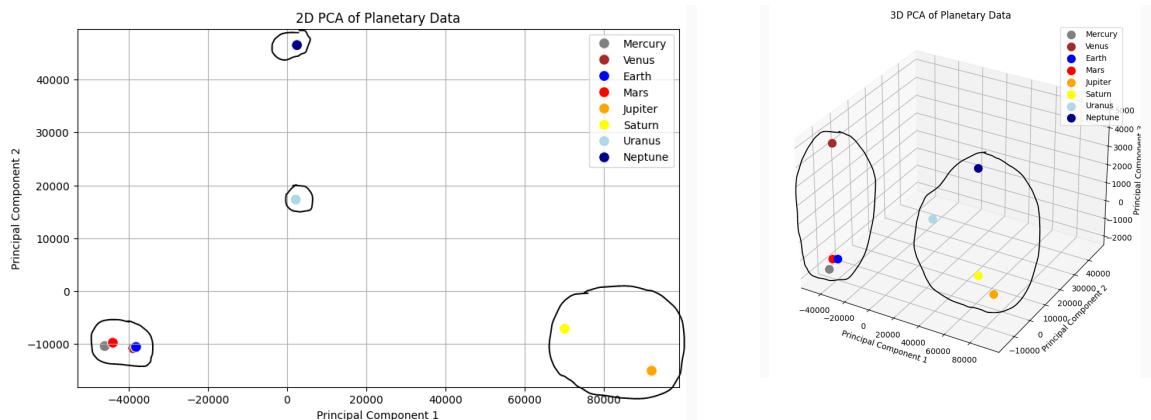
```

```

ax.set_zlabel('Principal Component 3')
ax.legend()

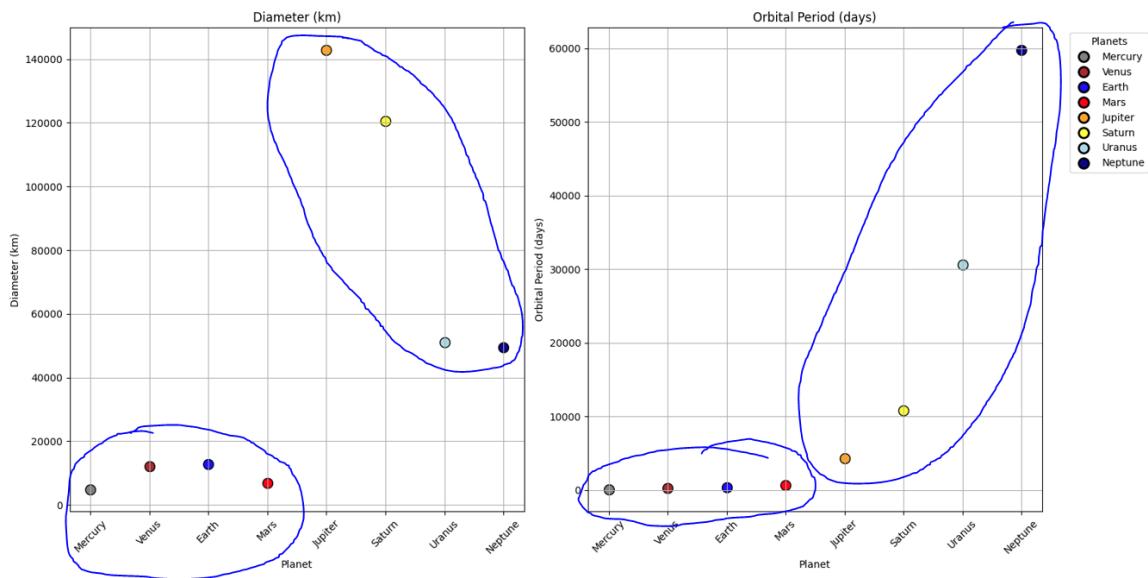
plt.show()

```



특성	주성분 1	주성분 2
직경 (km)	0.997	-0.066
궤도 주기 (일)	0.064	0.994
밀도 (kg/m^3)	-0.032	-0.043
하루 길이 (시간)	-0.015	-0.023
질량 (10^{24}kg)	0.011	-0.008
태양으로부터의 거리	0.008	0.073
자전 주기 (시간)	0.008	0.015
평균 온도 (C)	-0.002	-0.005
표면 온도 (C)	-0.001	-0.006

2. 직경과 궤도 주기 그래프



직경

지구형 행성

→ 직경이 매우 작음

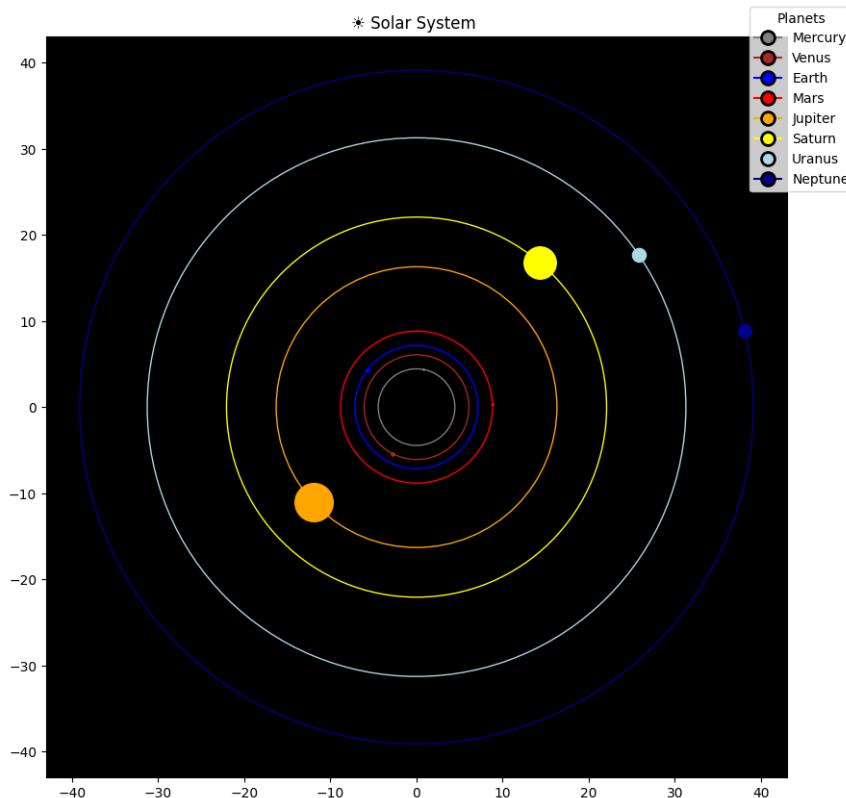
목성형 행성

→ 직경이 매우 큼 (목성 > 토성 > 천왕성 > 해왕성)

궤도주기

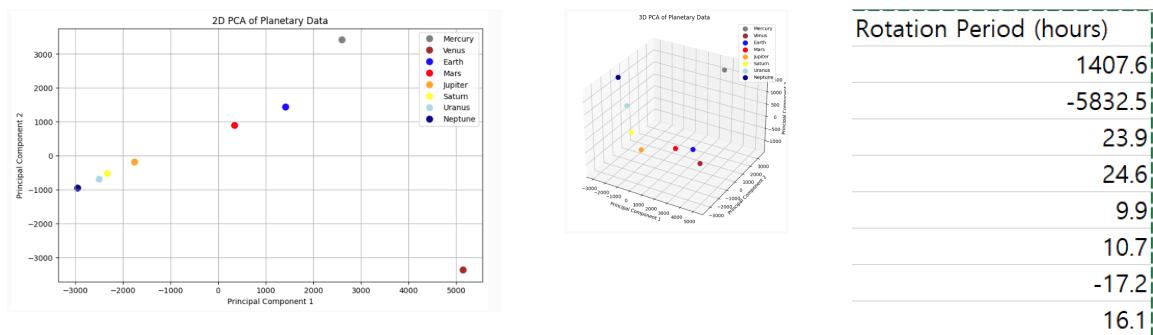
→ 태양계 위치마다 점점 기간이 길어짐

직경과 궤도주기를 태양계로 표현



➡ 밀도와 자전 주기

3. 직경과 궤도 주기를 빼고 PCA 진행



➡ 2차원 1 주성분은 밀도, 2 주성분은 자전 주기

데이터를 통해 목성형 행성은 자전 주기가 굉장히 짧고 밀도가 낮으며 지구형 행성은 밀도가 높고 자전 주기가 길다.

✓ 천왕성 금성이 이상한 값을 가짐

➡ 역회전 (반대방향)으로 자전하기 때문에 음수로 찍힘

➡ 3차원 1 주성분은 밀도, 2 주성분은 자전 주기, 3 주성분은 태양과의 거리

데이터를 통해 목성형 행성은 자전 주기가 굉장히 짧고 밀도가 낮으며 지구형 행성은 밀도가 높고 자전 주기가 길고 태양과의 거리가 수성은 정말 가까운 것을 알 수 있다.

5. 토의

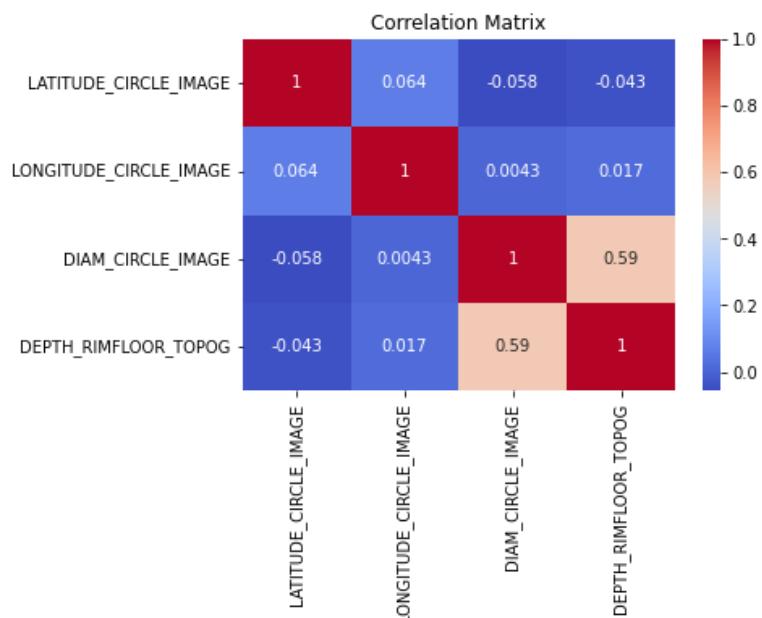
5-1. 가설 평가



화성 크레이터 데이터

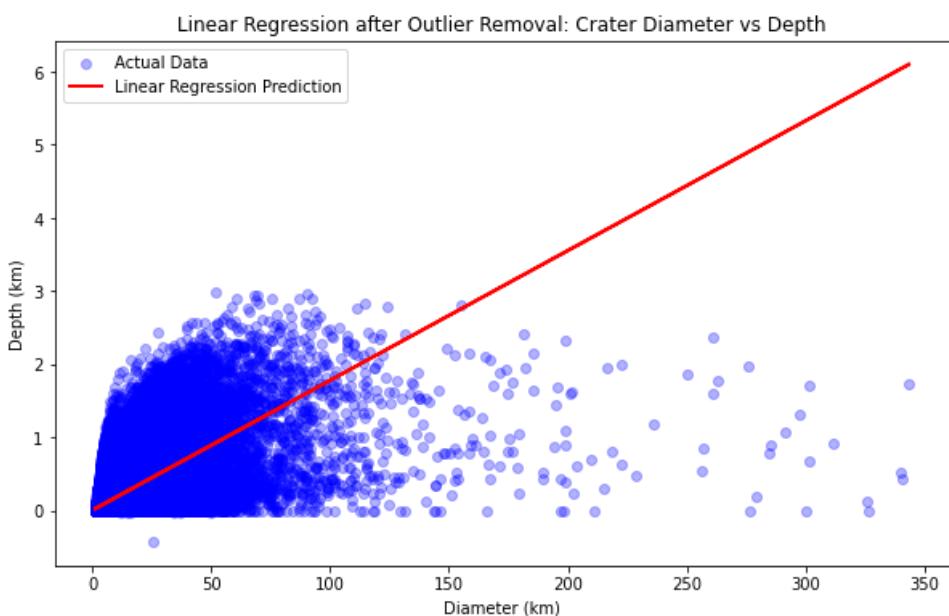
가설 “분화구의 직경이 클수록 깊이도 깊을 것” NO 😢

1. 전체 컬럼 상관계수 분석 진행



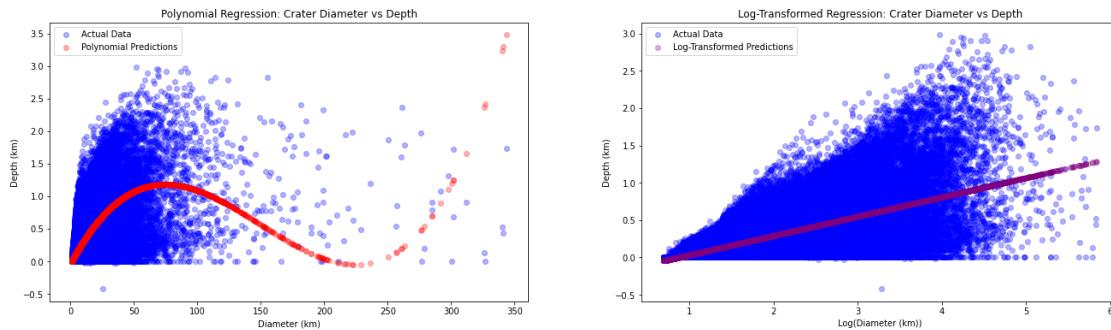
➡️ 분화구와 직경의 상관계수 (Correlation coefficient): 0.59로 가장 높았음

2. 직경과 분화구의 선형 패턴 파악



➡ 이상치 제거 후 분화구와 직경을 선형 회귀 모델로 돌려본 결과 선형 패턴이 보이지 않음

3. 직경과 분화구의 비선형 패턴 파악



- **다항 회귀와 로그 회귀**
 - 모델이 가장 높은 설명력을 가지고 있음
 - 두 모델이 직경과 깊이 간의 비선형 관계를 잘 설명하고 있음을 의미함

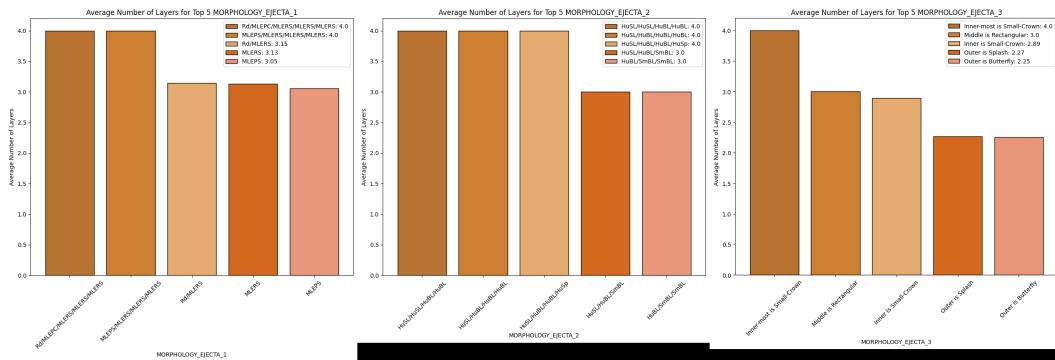
4. 결론

- 깊이와 직경 간의 관계가 **비선형적임**
- 큰 직경의 분화구는 **예측이 더 어렵고 추가적인 변수가 필요함**

▶ 복잡한 비선형 패턴이 존재하여 단순 선형 관계로 설명되지 않음을 확인했음 분화구 형성 과정이 엄청 복합적이고 **다양한 변수와 요인을 고려한 심층적인 지질학적 분석이 필요함**

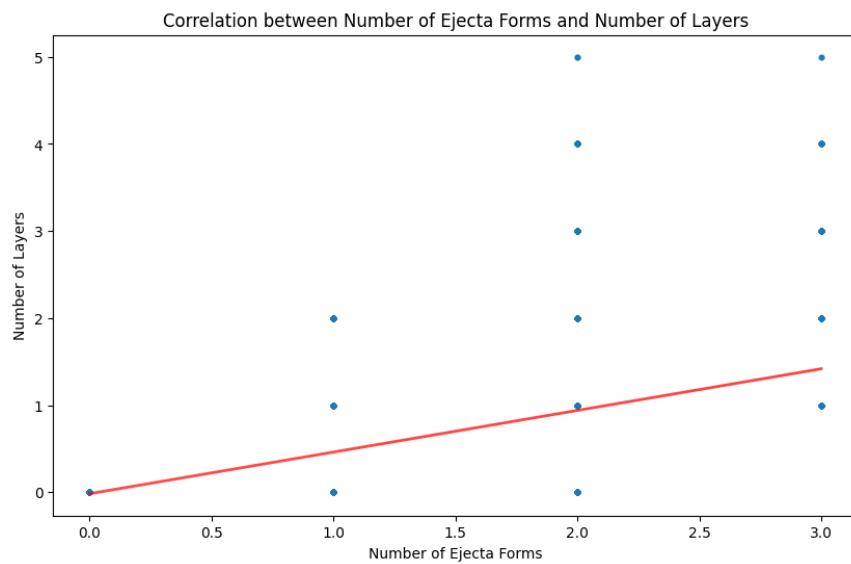
가설 2. “분화구에 분출물 성분이 많을 수록 층수가 더 많아질 것” YES 😎

1. 막대그래프로 Top 5에 드는 방출물 형태와 평균 층수를 표현



➡ 방출물 형태가 많을 수록 층수가 많아지는 것을 확인

2. 방출물 형태의 개수와 분화구 층 수 간의 피어슨 상관계산 분석



Correlation: 0.7927468207486432 (양의 상관관계)

P-value: 0.0

3. 결론

▶ 방출물 형태의 개수가 증가할수록 분화구의 총 수도 증가하는 패턴



화성 대기질 데이터

가설 1. “화성의 낮은 기압은 온도 변화의 폭을 크게 만든다.” → ✗

기압과 온도 변화에 대해 선형 회귀 분석을 진행

	coef	std err	t	P> t	[0.025	0.975]
<hr/>						
const	69.8133	2.026	34.467	0.000	65.842	73.785
mean_pressure(Pa)	-0.0097	0.002	-3.962	0.000	-0.014	-0.005

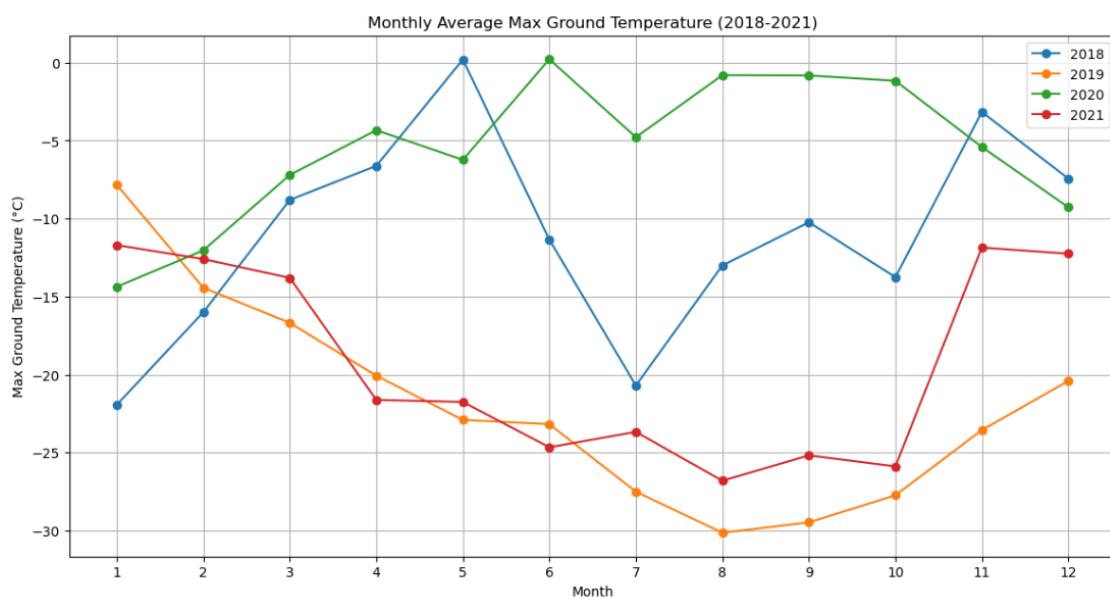
	coef	std err	t	P> t	[0.025	0.975]
<hr/>						
const	98.5381	2.826	34.873	0.000	92.998	104.078
mean_pressure(Pa)	-0.0196	0.003	-5.749	0.000	-0.026	-0.013



선형 회귀 분석 결과 1Pa 기압이 증가할 때 지표면 온도는 0.0097°C 감소, 공기 중 온도는 0.0196°C 감소하는 것으로 나타났지만 변화에 많은 영향을 주지 못하는 것으로 파악됨

가설 2 “화성의 온도 변화는 우리나라의 온도 변화 주기와 비슷하다.” → **X**

화성의 2018년도부터 2021년까지의 온도 변화



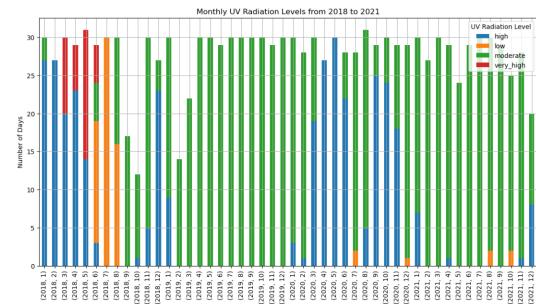
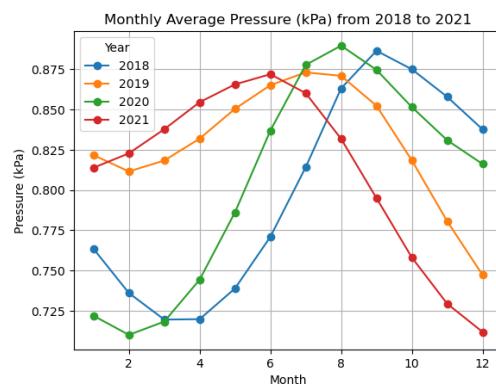
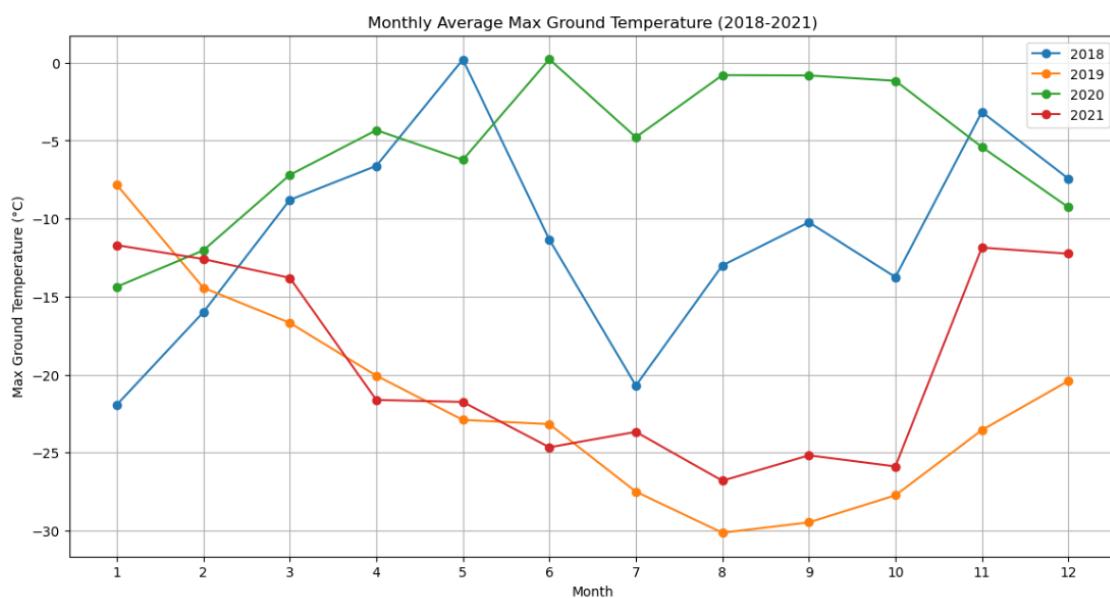
<https://data.kma.go.kr/climate/average30Years/selectAverage30YearsKoreaList.do?pgmNo=188>



우리나라의 기후 그래프와 화성의 기후 그래프가 다르게 나타나는 점을 확인하여 주기가 비슷하지 않음을 확인함

가설 3 “화성에 인간이 생존할 수 있는 시기가 존재한다.” → ✗

✓ 화성의 2018년도부터 2021년까지의 온도 변화



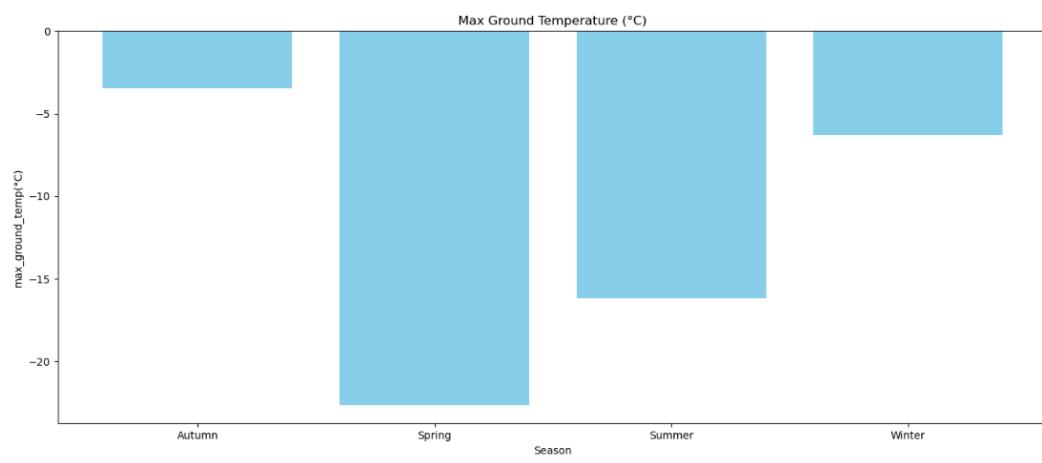
➡ 사람이 살 수 있는 최저 기온은 -10°C 로 2020년, 2018년도에 살 수 있는 기간이 종종 보이지만 기압이 사람이 살 수 없을 정도의 낮은 기압으로 형성되어 있어 사람이 살 수 있는 기간은 존재하지 않다고 파악됨.

가설 4 “화성에도 우리나라와 비슷한 사계절이 존재한다.” → **X**

화성에도 사계절은 존재

mars_date_time
Mars, Month 6 - LS 150°
Mars, Month 6 - LS 155°
Mars, Month 6 - LS 155°
Mars, Month 6 - LS 156°
Mars, Month 6 - LS 156°
Mars, Month 6 - LS 157°
Mars, Month 6 - LS 157°
Mars, Month 6 - LS 158°
Mars, Month 6 - LS 158°

- LS는 화성의 사계를 나누기 위해 존재, 360을 봄, 여름, 가을, 겨울로 나눠 가짐

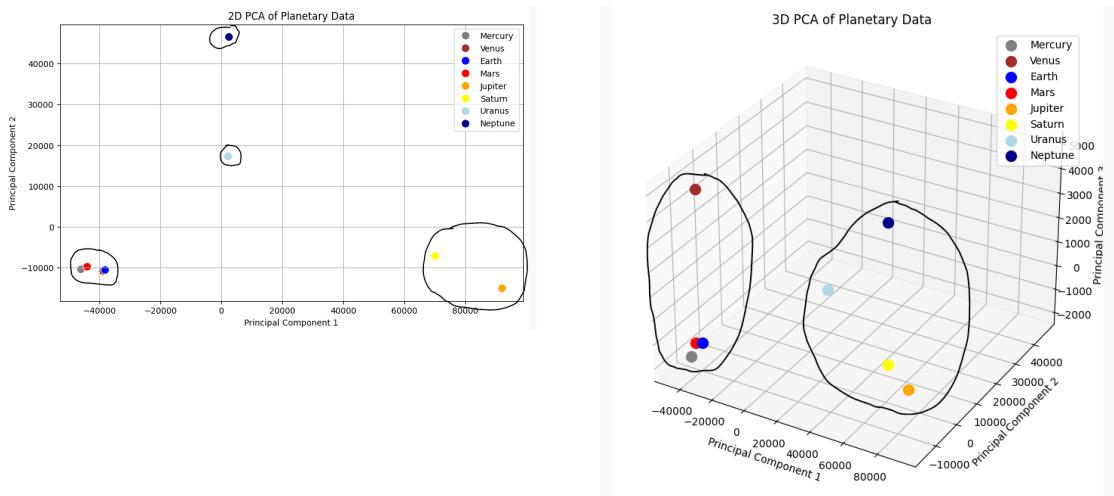


 하지만, 우리나라와 다르게 화성의 사계절은 봄, 여름이 더 춥고 가을, 겨울이 더 따뜻한것으로 파악됨

 태양계 데이터

가설 “지구형 행성과 목성형 행성끼리 그룹으로 묶일 것임” → ○

- 전체 컬럼을 넣어서 행성 간에 주성분 분석 진행

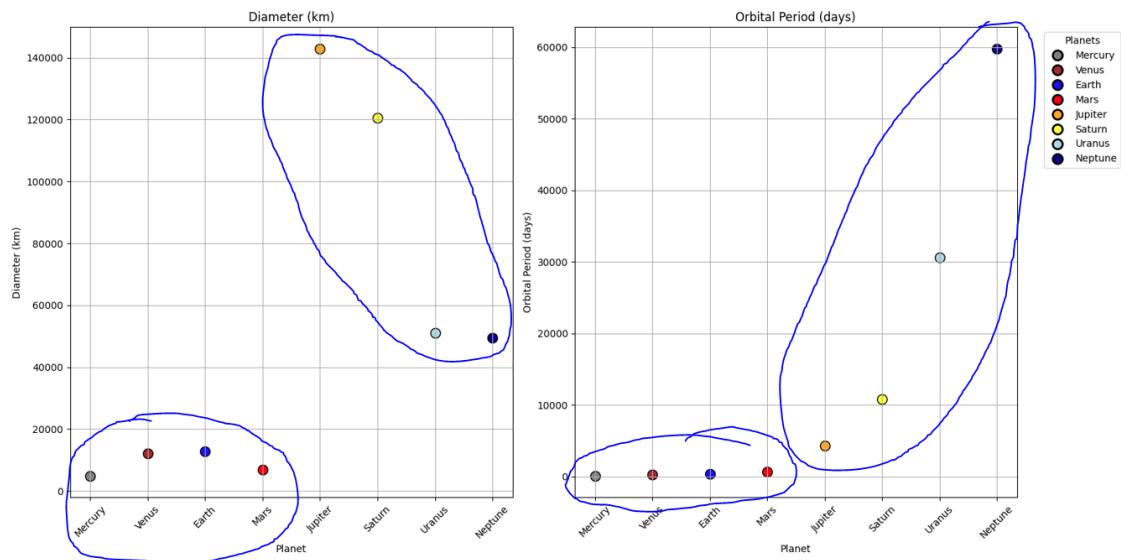


특성	주성분 1	주성분 2
직경 (km)	0.997	-0.066
궤도 주기 (일)	0.064	0.994
밀도 (kg/m^3)	-0.032	-0.043
하루 길이 (시간)	-0.015	-0.023
질량 (10^{24}kg)	0.011	-0.008
태양으로부터의 거리	0.008	0.073
자전 주기 (시간)	0.008	0.015
평균 온도 (C)	-0.002	-0.005
표면 온도 (C)	-0.001	-0.006



직경과 궤도 주기가 주성분

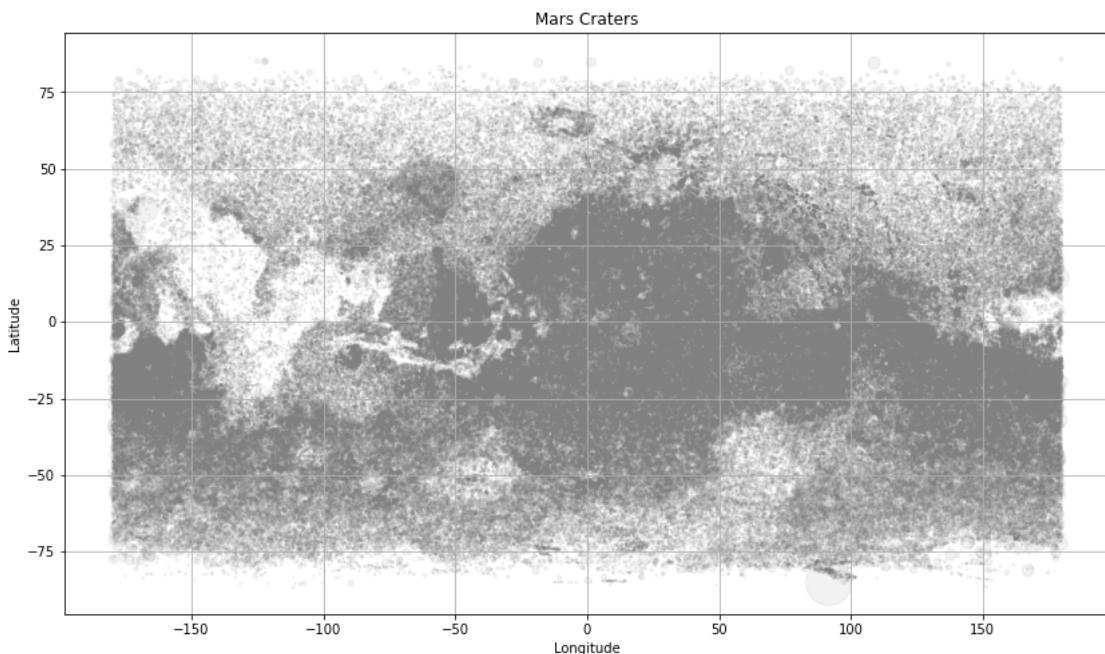
2. 직경과 궤도 주기 그래프



주성분으로 그래프를 그려 확인한 결과 지구형 행성과 목성형 행성으로 나뉘지는 것을 확인

5-2. 화성 크레이터 지도 만들기

1. 화성 크레이터 데이터 맵



➡ 분석 결과 상당히 분화구가 적도와 남반구 위주로 고르게 분포되어 있음

2. 분화구 100km 이상 크레이터 맵

<http://203.234.55.134:3000/>

<http://203.234.55.134:3000/>

<http://203.234.55.134:3000/>

➡ <https://www.openplanetary.org/opm-basemaps/opm-mars-basemap-v0-2> API <실제 화성 지도> 사용

➡ Folium 라이브러리 사용해서 지도 맵을 만듬

6. 결론



프로젝트를 통해 화성 분화구 데이터, 화성 대기질 데이터 태양계 행성 데이터를 사용하여 데이터 분석을 진행하고 다양한 가설을 검증하였음

1. 화성 분화구 데이터 분석

- **가설 1: 분화구의 직경이 클수록 깊이도 깊을 것이다.**
 - 분석 결과, 분화구의 깊이와 직경 간의 관계는 비선형적임을 확인하였음
단순한 직경 외에도 추가적인 변수가 깊이에 영향을 미친다는 것을 알 수 있었음
- **가설 2: 분화구에 분출물 성분이 많을수록 총수가 더 많아질 것이다.**
 - 피어슨 상관 관계 분석을 통해 분출물 형태의 개수가 증가할수록 분화구의 총 수도 증가하는 패턴을 발견하였음
 - 가설은 맞지만 조금 더 명확히 분석할 필요가 있음

2. 화성 대기질 데이터 분석

- **가설 1: 기압과 온도 변화의 폭 사이에는 상관관계가 있을 것이다.**
 - 선형 회귀 분석 결과 기압과 온도 변화 폭 사이에는 유의미한 상관관계가 없다는 사실을 밝혀냄
- **가설 2: 화성의 온도 변화 주기는 지구의 온도 변화 주기와 유사할 것이다.**
 - 분석 결과, 화성의 온도 변화 주기는 약 2년으로 지구의 온도 변화 주기와는 일치하지 않음을 확인하였음
- **가설 3: 인간이 화성에서 생존할 수 있는 시기가 있을 것이다.**
 - 화성의 평균 기압이 약 800Pa로 인간이 생존할 수 없는 조건임을 확인하였음 따라서 인간이 화성에서 생존하기에는 부적합한 환경임
- **가설 4: 화성에도 지구와 유사한 사계절이 있을 것이다.**
 - 분석 결과, 화성의 계절별 온도 변화는 지구와 상당히 다름 화성은 여름과 가을에 매우 춥고 봄과 겨울에 비교적 따뜻한 경향을 보여 지구의 계절 패턴과는 상반됨

3. 태양계 행성 데이터 분석

- **가설: 지구형 행성과 목성형 행성은 주성분 분석을 통해 그룹으로 묶일 것이다.**

- 주성분 분석 결과 지구형 행성과 목성형 행성은 주성분 공간에서 비교적 뭉쳐서 나오는 것을 확인하였음 두 그룹이 특정 특성에 따라 분류될 수 있음을 알 수 있었음

추가 작업: 화성 크레이터 지도 시각화

- **화성 크레이터 데이터 맵**

- 화성 분화구 데이터를 이용하여 분화구의 위치를 분석함
- 화성 표면의 분화구 분포를 시각적으로 보여주어 분화구의 위치를 파악 할 수 있음

- **분화구 100km 이상 크레이터 맵**

- 화성 분화구 데이터를 이용하여 분화구의 위치를 분석하고 Folium 라이브러리와 실제 화성 지도 API를 활용하여 화성 크레이터 맵을 제작함



화성의 환경은 현재까지는 인간이 생존하기에 적합하지 않지만 이러한 프로젝트를 진행함으로써 데이터를 분석하고 화성의 지질학적, 대기적 특성을 이해하는데 도움을 줄 수 있었음 하지만 이 정도의 데이터로는 설명할 수 없는 부분들이 너무 많았음 향후 연구로는 다양한 화성 탐사선이 수집한 데이터들을 분석하여 지금 설명할 수 없는 가설들을 해결할 수 있도록 할 것이고 화성에 대한 정보를 제공하는 챗봇을 만들 때 참고할 데이터로써 활용하여 사람들에게 화성에 대한 지식을 쉽게 공유하여 진입 장벽을 낮추는데 도움을 줄 것임