ARTIFICIAL INTELLIGENCE,ML & DL

Project Number 1:

Project Title: Geospatial Data Analysis and Visualization in Geology

Developed By: Konain Ahmed

CNIC No:82203-9139285-7

Project supervisor: Muhammad Rizwan Khan

Project Overview:

In this project, you will analyze and visualize geological data using Python. You will work with **NumPy** and **pandas** for data manipulation, and **Matplotlib** for creating visualizations. The goal is to interpret geological phenomena, such as earthquake patterns, rock formation distributions, and soil composition, to aid in geological research and decision-making.

Objectives:

- Utilize **Matplotlib**, **NumPy**, and **pandas** in the context of geological data analysis.
- Load, clean, and manipulate geospatial and geological data.
- Perform basic statistical analysis using **NumPy**.
- Create various visualizations to interpret geological patterns and trends.
- Combine multiple visualizations to offer a comprehensive view of geological data.

1. Introduction

This project analyzes and visualizes geological data using Python, showing patterns in earthquakes, rock types, and soil composition.

2. Data Overview

Dataset:

The dataset includes geospatial and geological data collected from different regions. The dataset contains the following columns:

- **Region**: The name or identifier of the geographical region.
- **Latitude**: Latitude coordinate of the data point.
- **Longitude**: Longitude coordinate of the data point.
- **Elevation (m)**: Elevation of the region in meters.
- **Rock Type**: Dominant rock type in the region (e.g., sedimentary, igneous, metamorphic).
- **Soil Composition**: Percentage composition of different soil types (e.g., clay, sand, silt).
- **Earthquake Frequency**: Number of earthquakes recorded in the region over a specified period.
- **Average Temperature (°C)**: Average annual temperature of the region.

2.Project Tasks

1. **Installation and Setup**:

    - Install the required libraries
    - Type the following commands in your command promp,power shell,terminal or directly into cell to install libraries.

pip install numpy pandas matplotlib

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\misba> pip install numpy pandas matplotlib
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\users\misba\appdata\roaming\python\python312\site-packages (2.0.1)
Requirement already satisfied: pandas in c:\users\misba\appdata\roaming\python\python312\site-packages (2.2.2)
Requirement already satisfied: matplotlib in c:\users\misba\appdata\roaming\python\python312\site-packages (3.9.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\misba\appdata\roaming\python\python312\site-packages (from pandas)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\misba\appdata\roaming\python\python312\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\misba\appdata\roaming\python\python312\site-packages (from pandas) (2024.1)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (1
.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (0.12.
1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (
4.53.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (
1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (24
.1)
Requirement already satisfied: pillow>=8 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\misba\appdata\roaming\python\python312\site-packages (from matplotlib) (3
.1.2)
Requirement already satisfied: six>=1.5 in c:\users\misba\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.8.2
->pandas) (1.16.0)
PS C:\Users\misba>
```

2. **Importing Libraries**:

- Import Matplotlib, NumPy, and pandas

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

3. **Loading and Exploring Data**:

    - Load the geological data and examine its structure.

Code

LOADING AND EXPLORING DATA

```
df=pd.read_csv("geospatial_geological_data_500_entries.csv")
print(df)
```

✓ 0.3s

Output

```
      Region   Latitude   Longitude  Elevation (m)    Rock Type  \
0       West  28.876757 -171.735313    8407.665470  Sedimentary
1    Central -24.113510 -137.988411    3327.163550  Metamorphic
2       East  85.100246   82.351161     635.887001  Metamorphic
3    Central -66.195677  168.107903    2300.531636  Sedimentary
4    Central -33.023112  159.474712    4656.207375  Sedimentary
..       ...        ...         ...            ...          ...
495    North -58.023480  133.053298    7564.156793      Igneous
496     East  36.001756  -32.713754    3537.115535  Metamorphic
497    North  19.894278 -171.805841    8668.562188      Igneous
498    North -47.363460  150.346976    8575.555706  Sedimentary
499     West  63.355240   62.834869    7634.937038      Igneous
```

```
                           Soil Composition  Earthquake Frequency  \
0    Clay: 29.40%, Sand: 16.83%, Silt: 53.77%                     3
1    Clay: 37.83%, Sand: 11.66%, Silt: 50.50%                    12
2    Clay: 49.50%, Sand: 24.40%, Silt: 26.10%                    49
3    Clay: 17.85%, Sand: 35.78%, Silt: 46.36%                    89
4    Clay: 11.44%, Sand: 24.10%, Silt: 64.46%                    82
..                                        ...                   ...
495  Clay: 34.35%, Sand: 16.98%, Silt: 48.68%                    13
496   Clay: 49.77%, Sand: 49.27%, Silt: 0.96%                    51
497  Clay: 41.47%, Sand: 27.81%, Silt: 30.72%                    88
498  Clay: 27.07%, Sand: 40.68%, Silt: 32.25%                    25
499  Clay: 15.15%, Sand: 21.91%, Silt: 62.94%                    67
...
498              -26.795340
499               49.984498

[500 rows x 8 columns]
```

We used pandas to load the data and completed some of initial cleaning tasks such as handling missing values, categorizing BMI into standard category etc.

4. **Data Cleaning and Manipulation**:
    - Handle missing values, categorize data, and calculate additional metrics (e.g., soil type classifications).

Code & Output

Data Cleaning and Manipulation:

```python
# Checking Null Values In Data
print(df.isnull().sum())
# Categorization of data..
df['Rock Type'] = df['Rock Type'].astype('category')
```

[19] ✓ 0.0s

```
Region                       0
Latitude                     0
Longitude                    0
Elevation (m)                0
Rock Type                    0
Soil Composition             0
Earthquake Frequency         0
Average Temperature (°C)     0
dtype: int64
```

+ Code + Markdown

The DataSet do not have any null value so no need to clarify the null values.

5.Visualizing Earthquake Patterns:

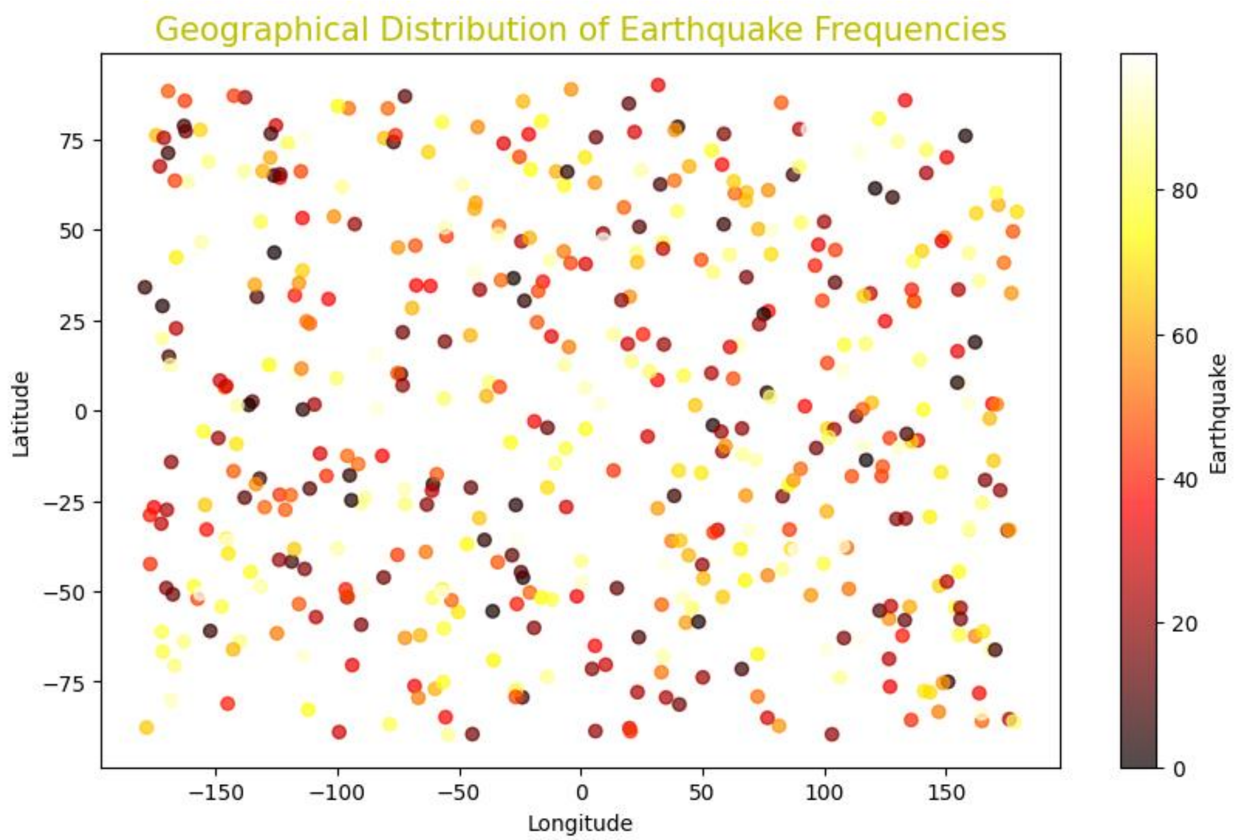
- Plot the geographical distribution of earthquake frequencies using scatter plots and geospatial maps.

Code

```python
x=df["Longitude"]
y=df["Latitude"]
plt.figure(figsize=(10, 6))
plt.scatter(df['Longitude'], df['Latitude'],
            c=df['Earthquake Frequency'], cmap='hot', alpha=0.7)
plt.colorbar(label='Earthquake')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.title('Geographical Distribution of Earthquake Frequencies',color="y",size=15)
plt.show()
```

[51] ✓ 6.6s

Graph



6.Rock Type Distribution Analysis:

- Create pie charts to visualize the distribution of different rock types across various regions.

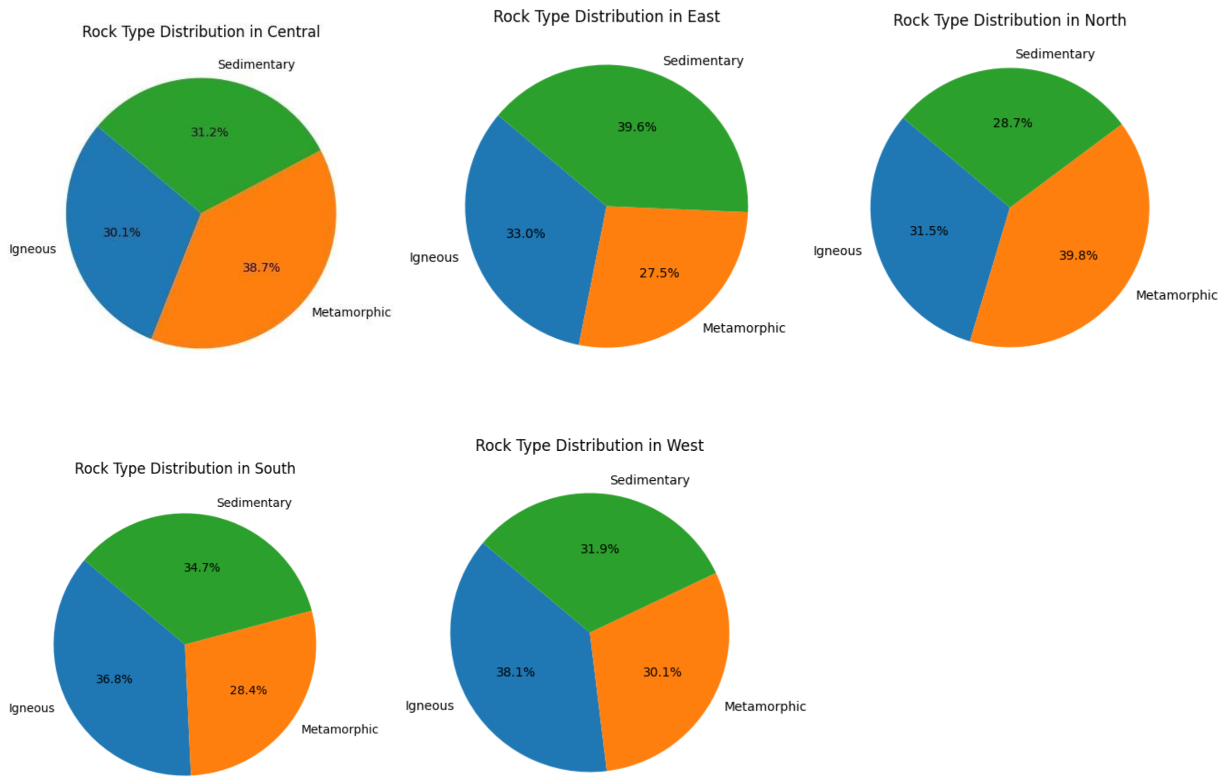
Code

Rock Type Distribution Analysis

```python
# Group by Region and Rock Type
rock_distribution = df.groupby(['Region', 'Rock Type']).size().unstack()

# Plot pie charts for each region
for region in rock_distribution.index:
    plt.figure(figsize=(5, 5))
    rock_distribution.loc[region].plot(kind='pie', autopct='%1.1f%%', startangle=140)
    plt.title(f'Rock Type Distribution in {region}')
    plt.ylabel('')
    plt.show()
```

[22] ✓ 0.7s

Graph



7.Elevation and Temperature Relationship:

- Generate scatter plots to analyze the relationship between elevation and average temperature.

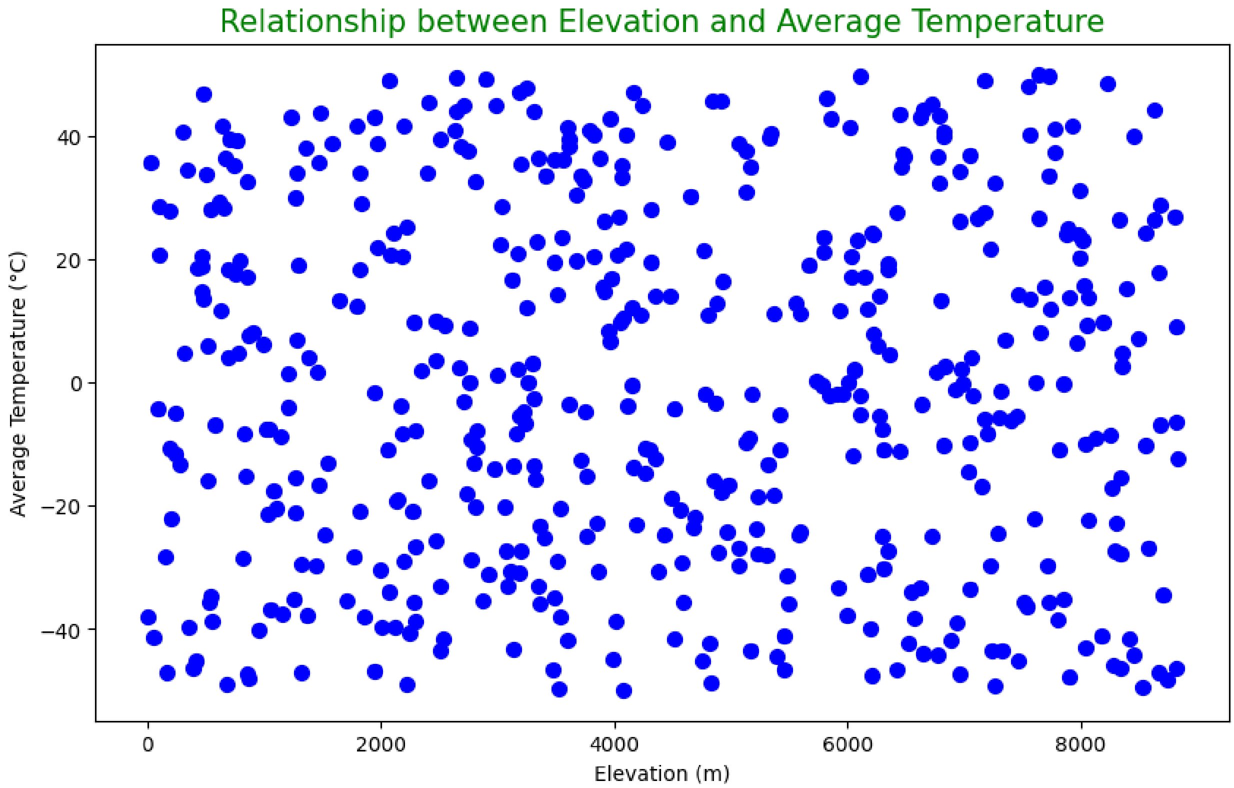
Code

Elevation and Temperature Relationship

```
plt.figure(figsize=(10, 6))
plt.scatter(df['Elevation (m)'], df['Average Temperature (°C)'], c='blue', s=50)
plt.xlabel('Elevation (m)')
plt.ylabel('Average Temperature (°C)')
plt.title('Relationship between Elevation and Average Temperature',fontdict={'size':15,'color':"green"})
plt.show()
```

[49] ✓ 0.4s

Graph



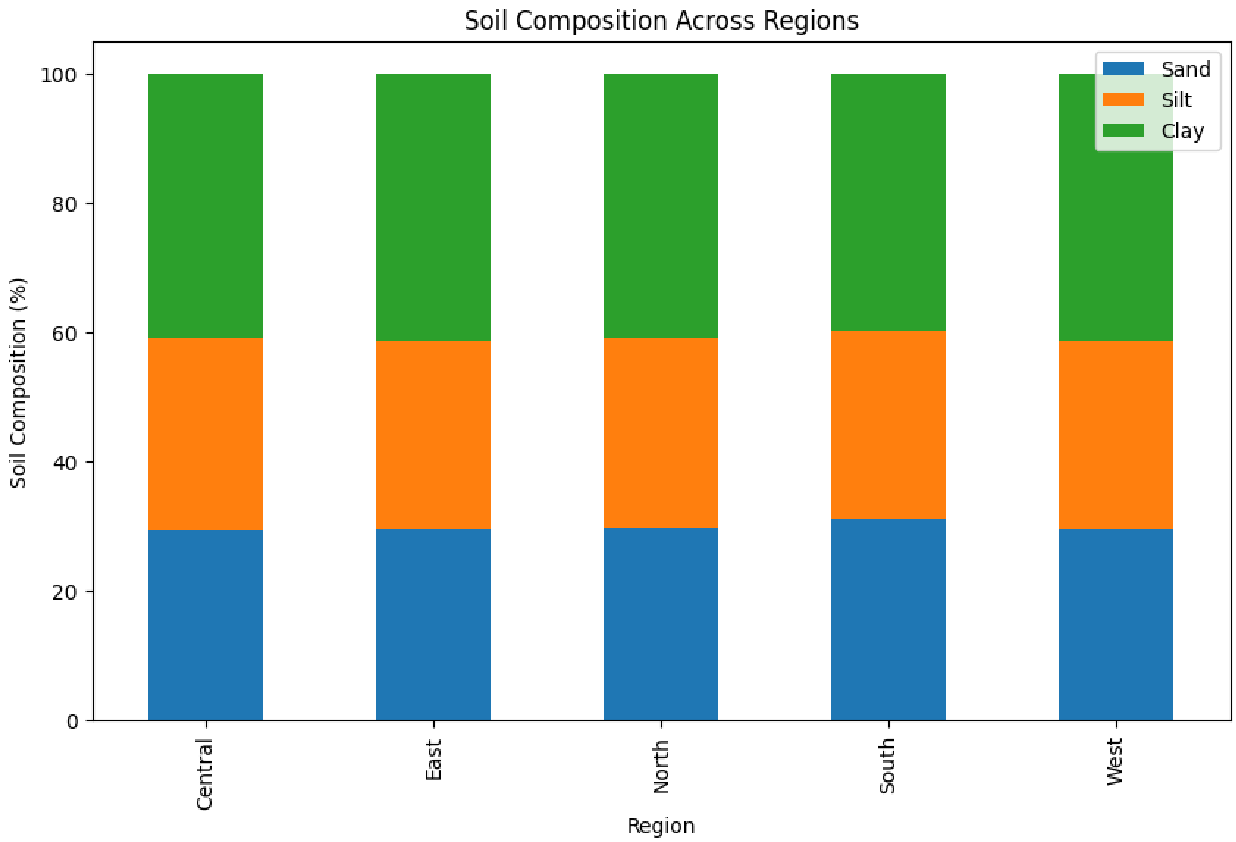
8.Soil Composition Analysis:

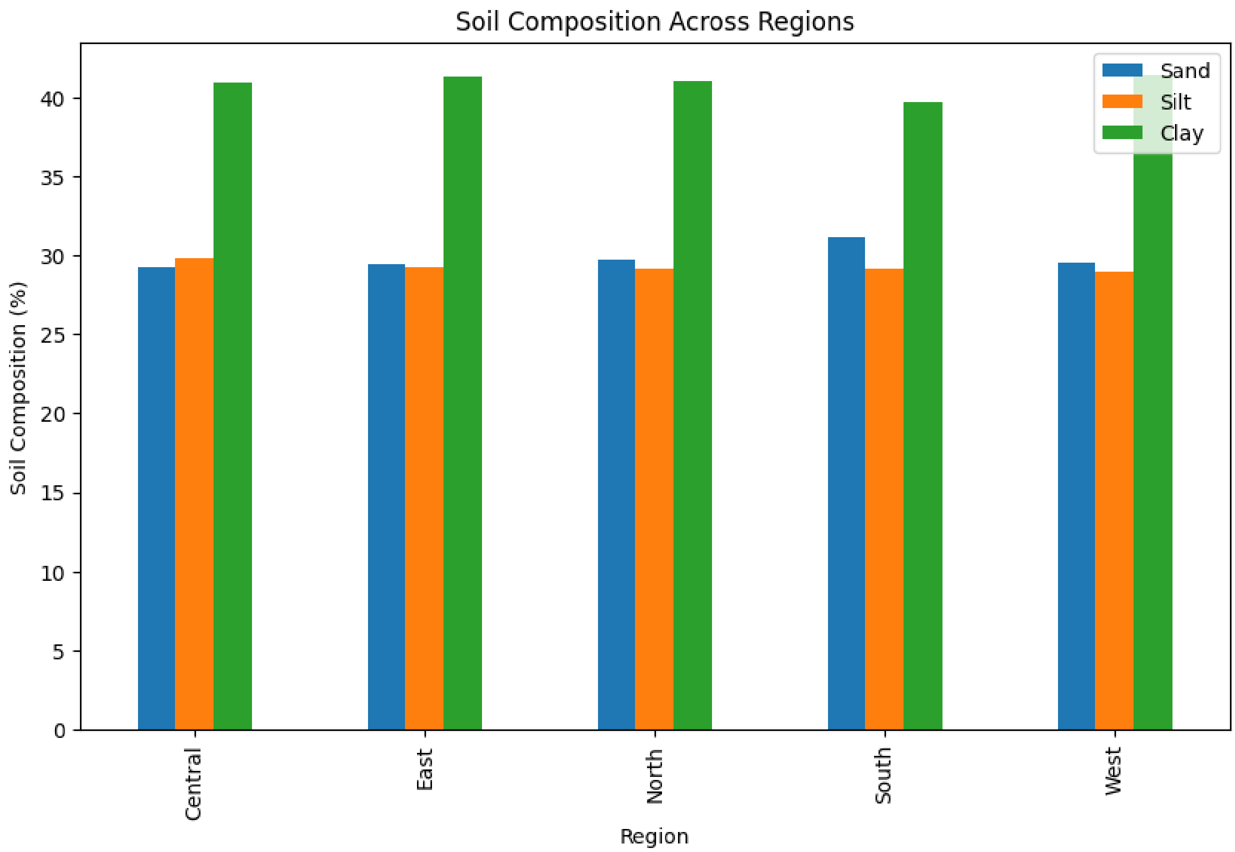
- Use bar charts to compare soil composition percentages across different regions.

Code

```
x= df["Region"],
y=df["Soil Composition"]

soil_composition_split = y.str.split(',', expand=True)

sand = []
silt = []
clay = []

for i in range(len(soil_composition_split)):
    sand.append(float(soil_composition_split[0][i].split()[1].replace('%', '')))
    silt.append(float(soil_composition_split[1][i].split()[1].replace('%', '')))
    clay.append(float(soil_composition_split[2][i].split()[1].replace('%', '')))


composition_df = pd.DataFrame({
    'Region': df['Region'],
    'Sand': sand,
    'Silt': silt,
    'Clay': clay
})
soil_composition_by_region = composition_df.groupby('Region').mean()

soil_composition_by_region.plot(kind='bar',stacked=True, figsize=(10, 6))
plt.title('Soil Composition Across Regions')
plt.xlabel('Region')
plt.ylabel('Soil Composition (%)')
plt.show()
```

Graph


Soil Composition Across Regions
Sand
Silt
Clay
Soil Composition (%)
0
20
40
60
80
100
Central
East
North
South
West
Region



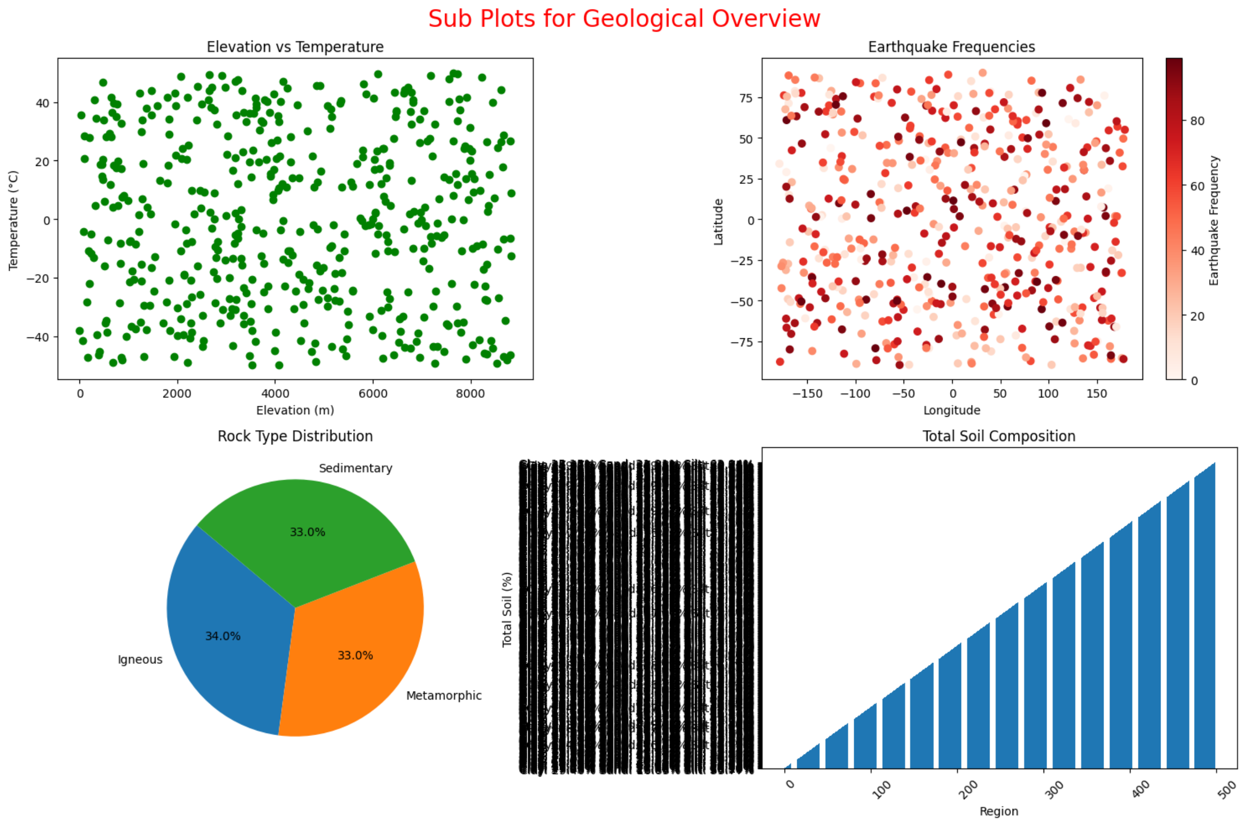
1. **Subplots for Geological Overview**:
   - Create a figure with multiple subplots to provide an overview of key geological indicators, such as elevation, rock types, and earthquake frequencies.

```python
plt.figure(figsize=(15, 10))

# Elevation vs Temperature
plt.subplot(2, 2, 1)
plt.scatter(df['Elevation (m)'], df['Average Temperature (°C)'], c='green')
plt.title('Elevation vs Temperature')
plt.xlabel('Elevation (m)')
plt.ylabel('Temperature (°C)')

# Earthquake Frequencies
plt.subplot(2, 2, 2)
plt.scatter(df['Longitude'], df['Latitude'], c=df['Earthquake Frequency'], cmap='Reds')
plt.title('Earthquake Frequencies')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.colorbar(label='Earthquake Frequency')
```

```python
# Rock Type Distribution
plt.subplot(2, 2, 3)
rock_distribution = df['Rock Type'].value_counts()
plt.pie(rock_distribution, labels=rock_distribution.index, autopct='%1.1f%%', startangle=140)
plt.title('Rock Type Distribution')

# Soil Composition
plt.subplot(2, 2, 4)
plt.bar(soil_composition.index, soil_composition.sum(axis=1)) # type: ignore
plt.title('Total Soil Composition')
plt.xlabel('Region')
plt.ylabel('Total Soil (%)')
plt.xticks(rotation=45, ha='left')

plt.tight_layout()
plt.show()
```

[46] ✓ 8.2s

Output:



Conclusions:

The conclusion is that learning to analyze and visualize geological data using Python is a powerful way to uncover important patterns in nature. This project demonstrates how tools like NumPy, pandas, and Matplotlib can be used to transform raw data into meaningful insights, helping you build a strong foundation in data analysis and contribute to real-world applications in geology.

Thank You