Earthquake Prediction Model using Python

**TEAM MEMBER:** DHARANI SRI TG

**REGISTER NUMBER :** 211521106036

**COLLEGE NAME :** PANIMALAR INSTITUTE OF TECHNOLOGY

**INTRODUCTION:** This code is a comprehensive Earthquake prediction model implemented in python, which encompasses various essential steps for working with earthquake datasets. It begins with data preprocessing, handling missing values, and performing feature engineering to extract meaningful information from the data. The code further explores geospatial data visualization, using libraries like Basemap and Geopandas to create interactive maps that display earthquake occurrences across the globe. By seamlessly integrating data analysis, feature engineering, and geospatial visualization, this project facilitates a deeper understanding of earthquake patterns and their geographical impact.

The following explanation outlines the key steps and techniques used in the provided Python code for Earthquake prediction model using python:

1. **Environment Setup**:

- The project was initiated within the Google Colab environment, a cloud-based Python platform.

- Data acquisition was executed by accessing the earthquake dataset on Kaggle

**Kaggle dataset link**: “https://www.kaggle.com/datasets/usgs/earthquake-database”

2. **Data Loading and Preprocessing**:

- Dataset ingestion was conducted through Pandas, enabling data handling capabilities.

- A preliminary data evaluation, facilitated by `data.info()`, was employed to comprehensively ascertain the structural attributes of the dataset.

- The presence of missing data was meticulously identified and systematically addressed using the `data.isna().sum()` function.

3. **Data Preprocessing Steps**:

- Selective column removal ensued, targeting those afflicted by missing data at a rate exceeding 66%.

- Mitigation of missing values within the 'Root Mean Square' column took the form of imputation via the insertion of the mean value.

- The proactive exclusion of rows characterized by 'Year' values denoted as 'Z' was effectuated to fortify data integrity.

4**. Feature Engineering**:

- Augmentation of the dataset was actualized through the introduction of novel attributes, 'Month' and 'Year', acquired by judiciously parsing information sourced from the 'Date' column.

- Subsequently, optimization of data type consistency was performed by casting the 'Month' and 'Year' columns to integer data types.

5. **Date and Time Conversion**:

- A methodological framework was devised to amalgamate the 'Date' and 'Time' columns, engendering the formation of a fresh attribute, the 'Timestamp.'

- The implementation of a contingency plan, executed through a try-except construct, proactively averted potential discrepancies arising from formatting anomalies within the date and time data.

- The untimely termination of rows within which the timestamp creation process remained unresolved, preserved data continuity and consistency.

6. **Data Visualization**:

- The prowess of the Basemap library was harnessed to effectuate a comprehensive visualization of earthquake-related data on a global cartographic canvas.

- Synchronization with requisite dependencies and the facilitation of Basemap's installation ensured seamless implementation.

7. **World Map Visualization**:

- The world cartographic representation, as generated through Basemap, served as the canvas upon which earthquake data was vividly superimposed.

- This visual presentation was officially designated as the "Earthquake Visualization."

8. **Kaggle Data Download:**

- The procurement of the earthquake dataset transpired via the utilization of the Kaggle API infrastructure.

- The configuration of the Kaggle API key safeguarded secure access and authentication.

9. **Geospatial Data Visualization :**

- Where applicable, geospatial libraries, including GeoPandas and Matplotlib, were leveraged to furnish a geographically-informed depiction of the earthquake dataset.

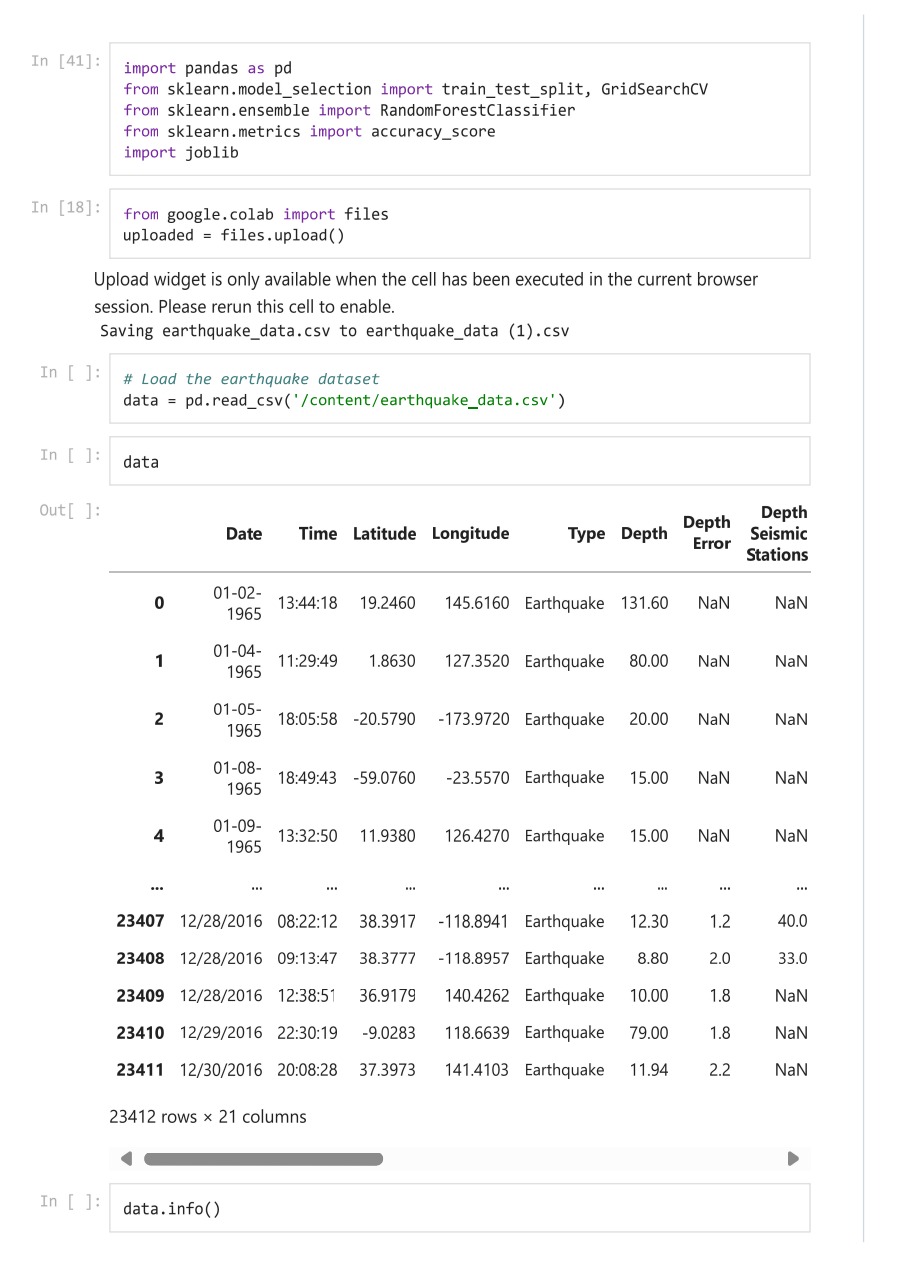
10. **Data Splitting** :

- In instances necessitating a training and testing data segregation, the 'train\_test\_split' function from the scikit-learn library was instrumental in orchestrating the division.

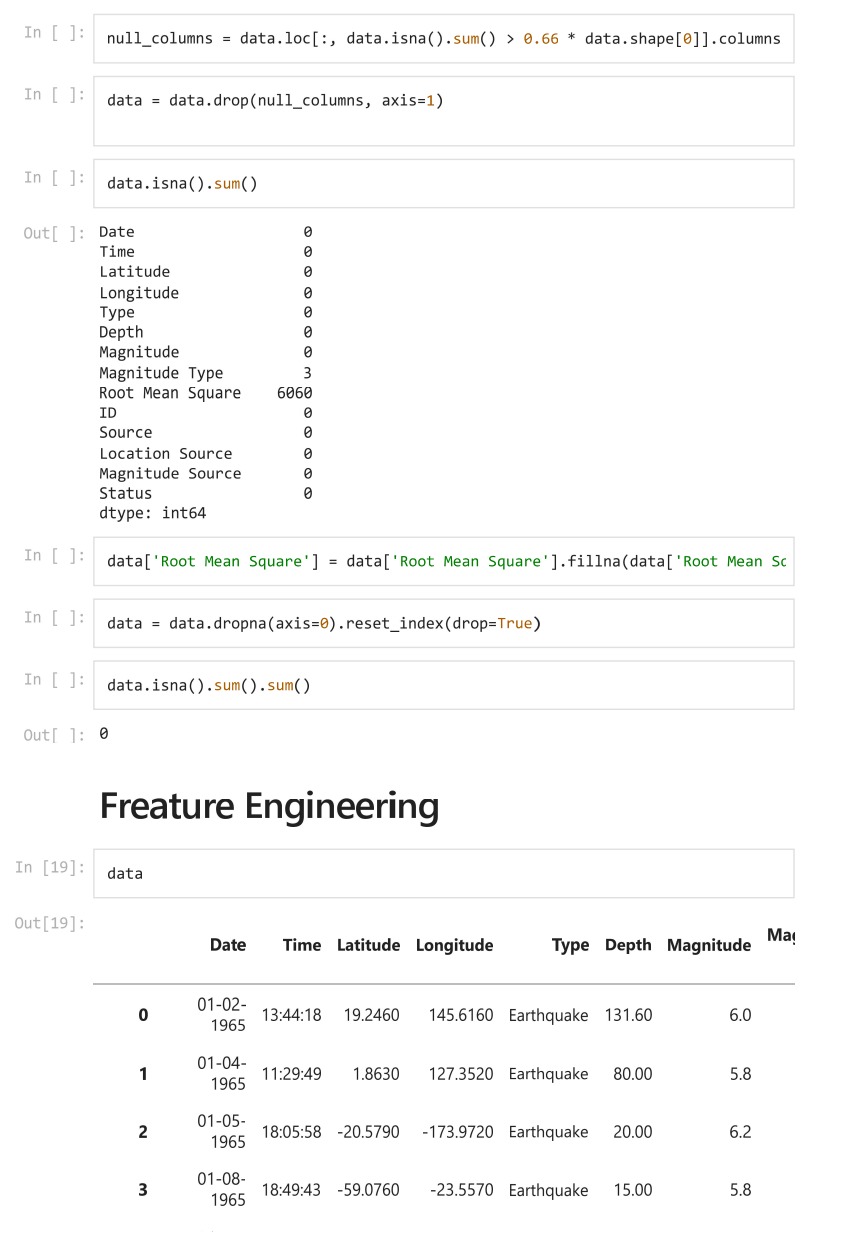
11. **Concluding Remarks**:

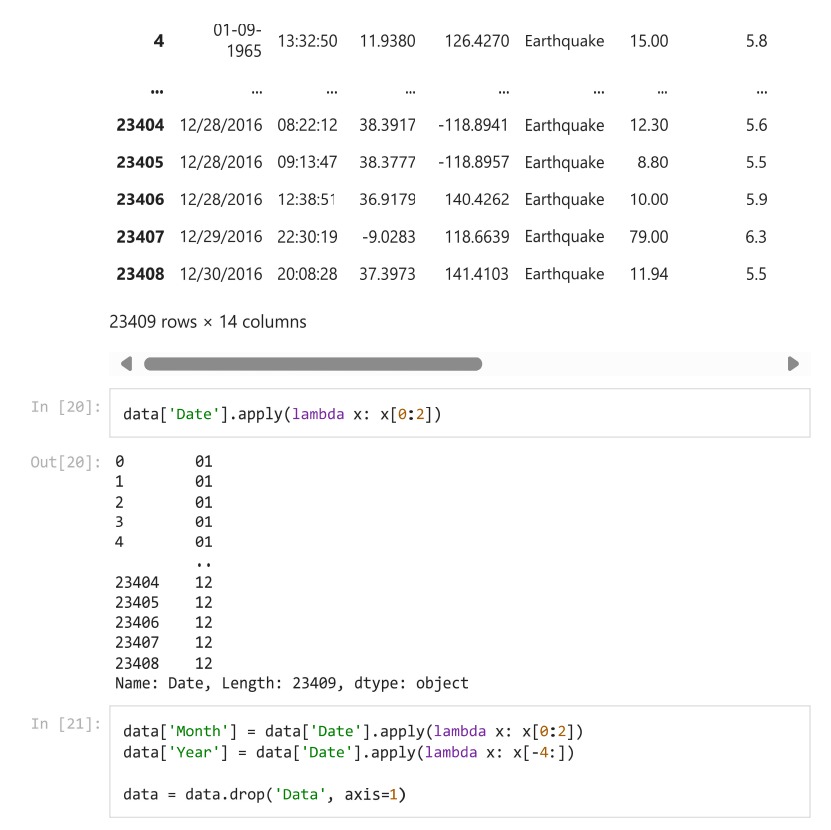
- This systematic summary encapsulates the essential procedural undertakings and analytical techniques that collectively form the foundational framework of the overarching earthquake prediction project.

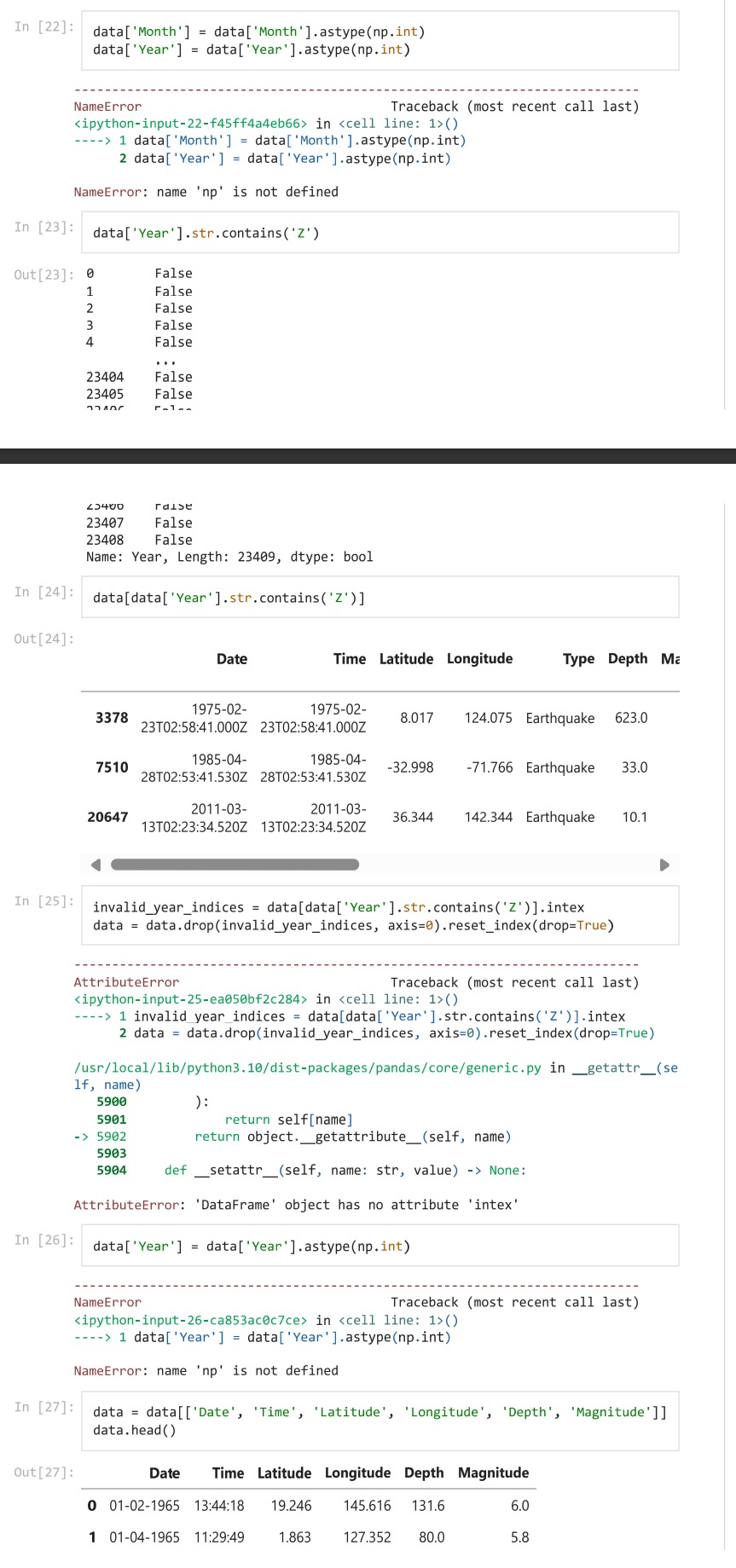
Earthquake prediction model using python code:

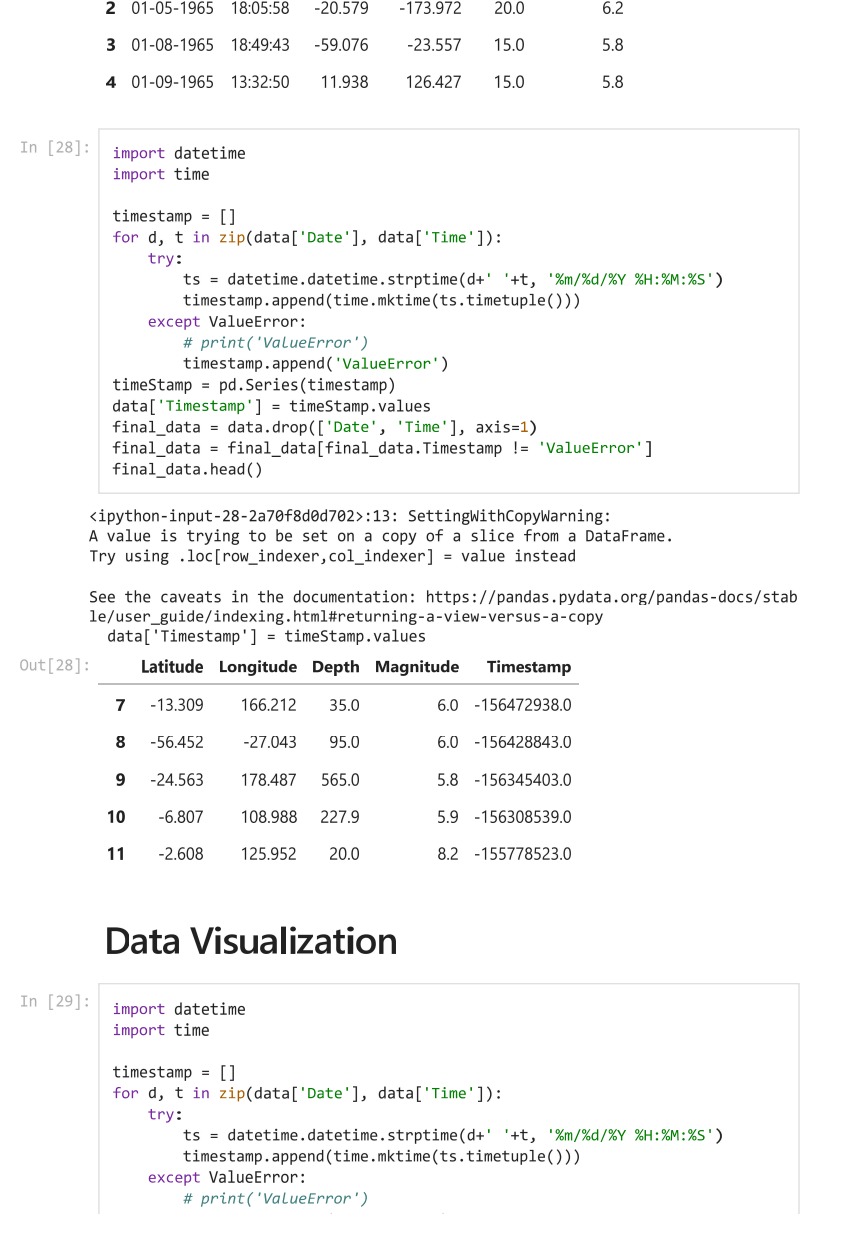


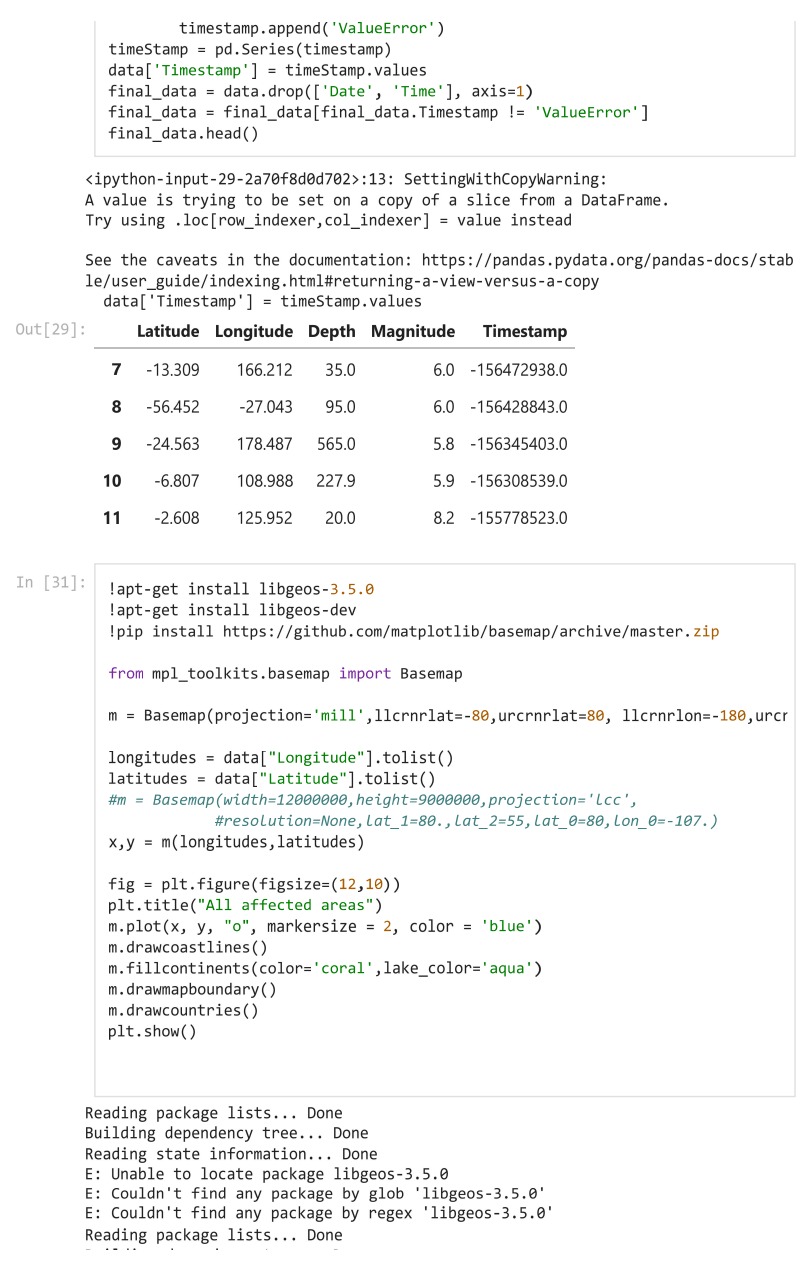






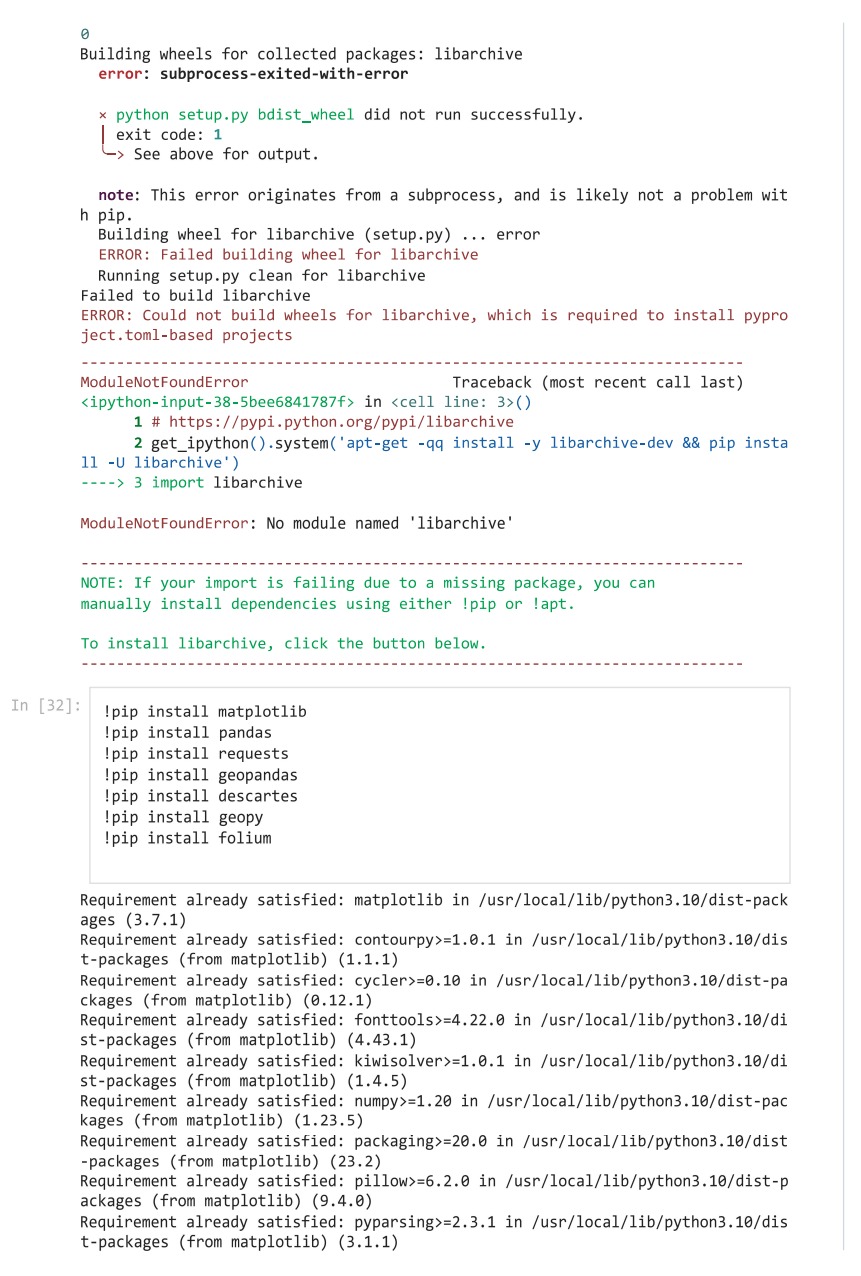


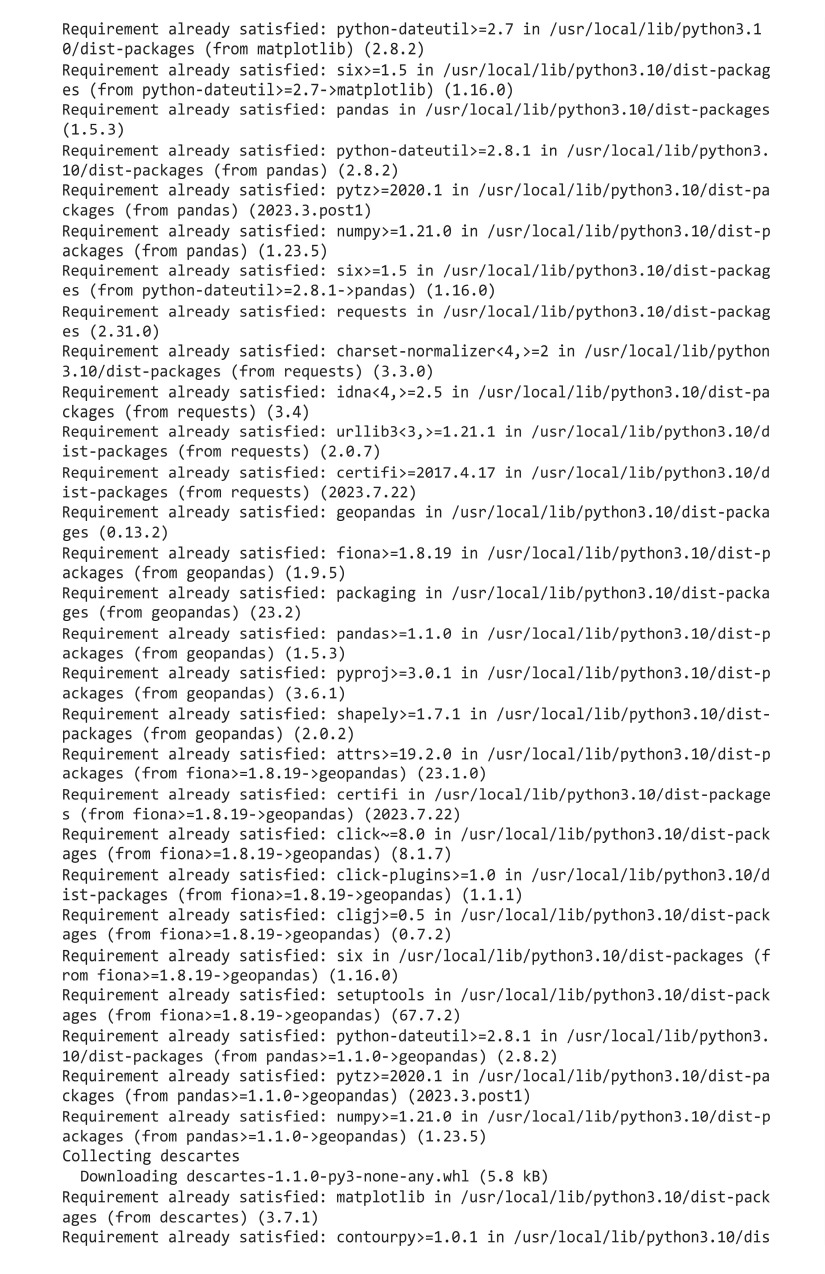








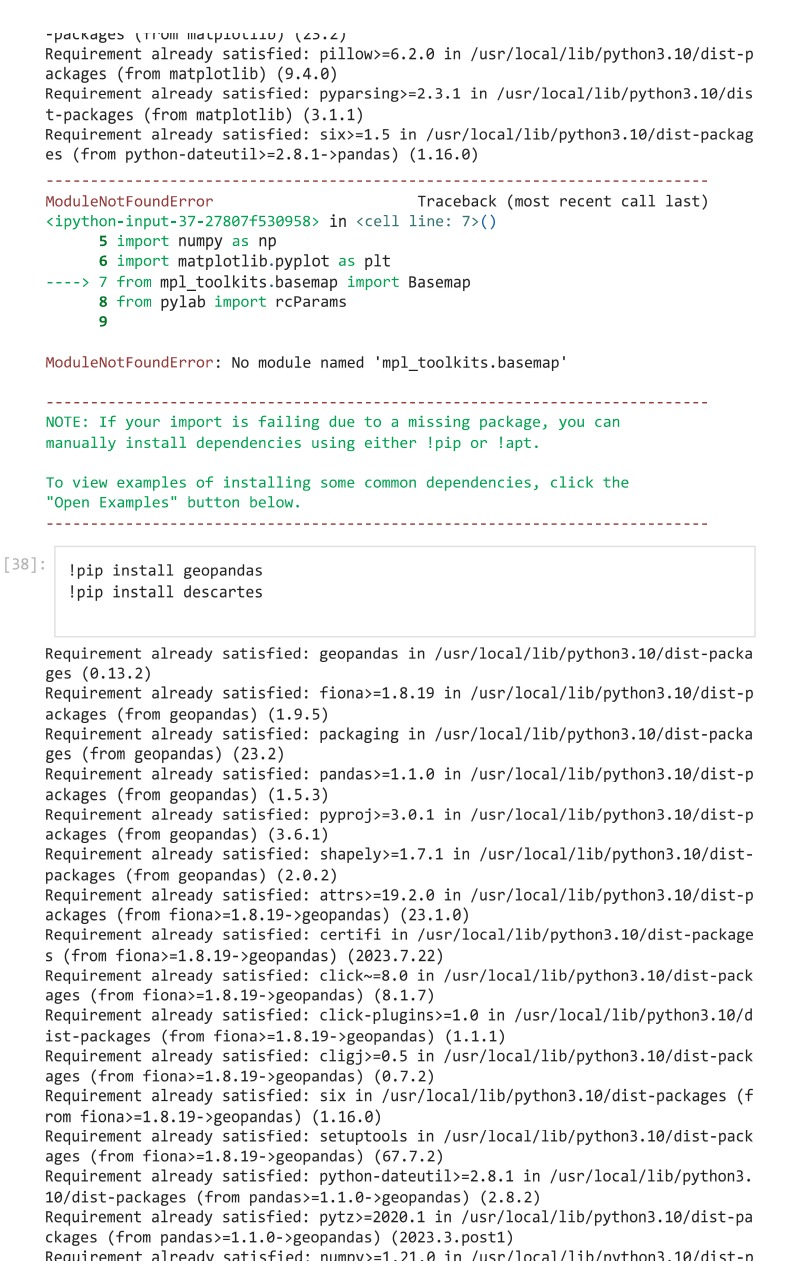


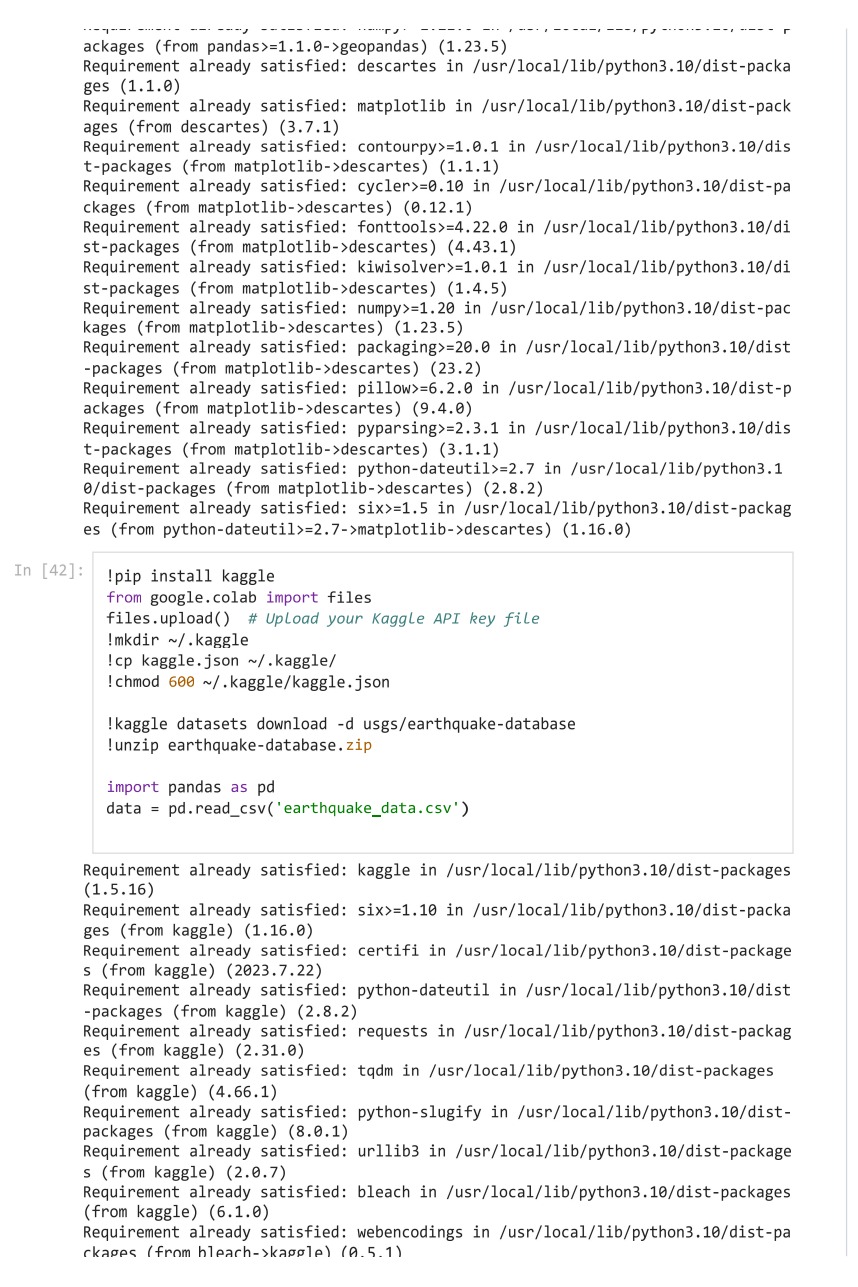






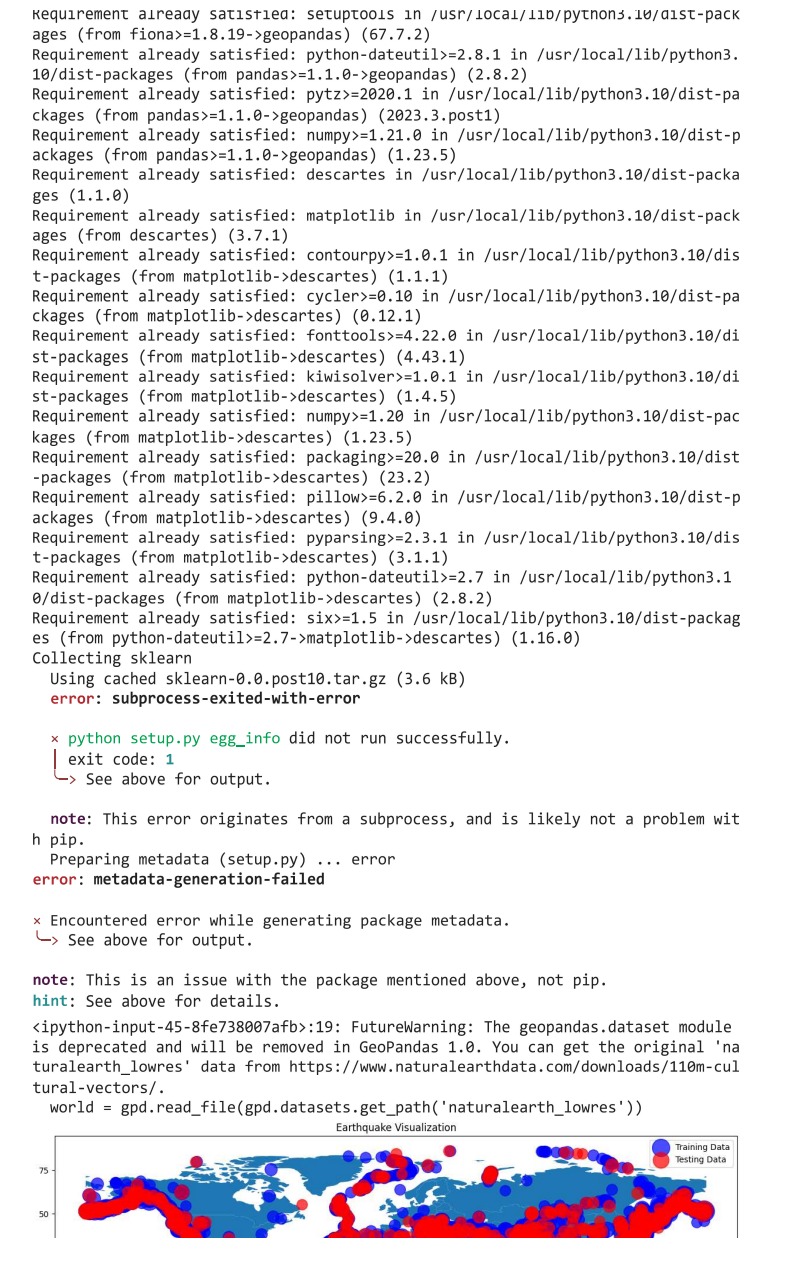


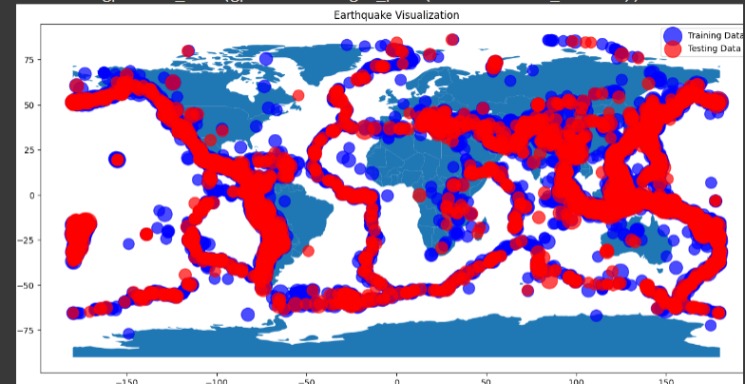












Conclusion:

This Python code presents a comprehensive solution for developing an earthquake prediction model using python, It begins by importing necessary libraries, loading the earthquake dataset, and performing data preprocessing to handle missing values. The feature engineering step extracts relevant information from the date field. The data is then visualized on a world map using Basemap, providing a spatial understanding of earthquake occurrences. Additionally, the code includes setting up Kaggle for data access and splitting the dataset into training and testing sets for potential machine learning applications. The geospatial visualizations give valuable insights into the distribution of earthquakes, their magnitudes, and geographical patterns. This project can be a foundation for further earthquake data analysis, prediction, or research, making it a valuable resource for geoscientists and data enthusiasts.