8144-SUDHARSAN ENGINEERING COLLEGE

REGISTER NUMBER: 814421243021

NAME: RAGUNATH J

DEGREE: BTECH

BRANCH: ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

PROJECT TITLE: EARTHQUAKE PREDICTION USING PYTHON

EARTHQUAKE PREDICTION USING PYTHON

PHASE 5 SUBMISSION DOCUMENT

PHASE 5: **Project Documentation And Submission**

Topic: **In this section we will document the complete project and prepare it for submission.**

INTRODUCTION:

- Earthquake prediction is a critical scientific endeavor aimed at mitigating the impact of these devastating natural disasters.
- While precise earthquake prediction remains elusive, Python, a versatile and widely-used programming language, plays a pivotal role in advancing our understanding of seismic activity.
- This introduction provides an overview of how Python is utilized in earthquake prediction, focusing on data collection, analysis, and early warning systems.
- By harnessing Python's capabilities, including data processing, machine learning, geospatial analysis, and data visualization, seismologists and researchers can monitor and analyze seismic data, model historical earthquake patterns, and develop real-time monitoring systems, ultimately working towards a safer and more prepared world in the face of seismic threats.
- Earthquake prediction, a complex and critical field, leverages the versatility of Python to enhance our understanding of seismic activity and its potential impact.
- While pinpoint earthquake prediction remains elusive, Python empowers seismologists and researchers to develop tools and models for monitoring, analyzing, and forecasting earthquakes.
- Python's utilization encompasses collecting and processing seismic data from global networks, employing machine learning for predictive modeling, creating real-time monitoring systems, and visualizing geospatial information.

LIST OF tools and software used in thr process of earthquake prediction using python:

Python:

Python serves as the primary programming language for data analysis, machine learning, and visualization in earthquake prediction.

NumPy:

A fundamental library for numerical and array operations, NumPy is often used for data manipulation and mathematical calculations.

Pandas:

Pandas is crucial for data manipulation and analysis, allowing the handling of structured seismic data.

Matplotlib:

This library is essential for creating static, interactive, and customizable data visualizations, including seismic activity plots and earthquake distribution maps.

Seaborn:

Seaborn is a data visualization library built on Matplotlib, focusing on statistical visualizations and making it easier to create informative plots.

Plotly:

Plotly is used for creating interactive, web-based visualizations and maps, which can be especially useful for displaying real-time seismic data.

scikit-learn:

scikit-learn offers a wide range of machine learning algorithms and tools for building predictive models based on seismic and geological data.

TensorFlow:

TensorFlow is a popular deep learning framework that can be applied for more advanced machine learning and neural network-based earthquake prediction models.

ObsPy:

- ObsPy is a Python toolbox specifically designed for seismology.
- It provides tools for reading, processing, and analyzing seismological data.

Folium:

Folium is used for geospatial data visualization, making it valuable for mapping and displaying earthquake locations and patterns.

GeoPandas:

GeoPandas extends the capabilities of Pandas to handle geospatial data, allowing for geospatial analysis of seismic activity.

GMT (Generic Mapping Tools):

- GMT is a command-line toolset for creating high-quality maps and graphics.
- It can be used alongside Python for geospatial visualization.

Jupyter Notebooks:

Jupyter Notebooks are widely used for creating and sharing interactive data analysis reports and code, making it easier to collaborate and present findings in earthquake prediction research.

ArcGIS:

While not open-source, ArcGIS provides powerful geospatial analysis and mapping capabilities, and it can be integrated with Python for earthquake research.

QGIS:

An open-source Geographic Information System (GIS) software, QGIS is used for geospatial analysis and mapping, including earthquake-related data.

GeoJSON and Shapefiles:

These file formats are commonly used for storing and sharing geospatial data relevant to earthquake prediction and analysis.

USGS Earthquake Data API:

The United States Geological Survey provides an API for accessing real-time and historical earthquake data, which can be easily integrated into Python applications.

IRIS (Incorporated Research Institutions for Seismology) DMC Tools:

IRIS offers a suite of seismological data access and analysis tools that can be used in Python workflows.

A Design Thinking Document:

1.Empathize:

- Understand Stakeholders: Identify the key stakeholders, including seismologists, emergency responders, and the general public, to comprehend their unique needs and concerns.

- Gather User Insights: Conduct interviews, surveys, and workshops with stakeholders to gain insights into their requirements, data preferences,and desired outcomes.

2.Define:

- Problem Statement: Define the problem statement based on user insights, e.g., "Develop a user-friendly earthquake prediction system that provides early warnings and visualizes seismic data."

- User Personas: Create user personas representing different stakeholder groups, outlining their pain points and expectations.

3.Ideate:

- Brainstorm Solutions: Engage in brainstorming sessions to generate innovative ideas and potential features, such as real-time data feeds,interactive maps,and early warning algorithms.

- Prototyping: Develop low-fidelity prototypes and wireframes to visualize the user interface and system functionalities.

4.Prototype:

- Minimum Viable Product (MVP): Develop a basic earthquake prediction system in Python with essential features, including data collection, visualization, and preliminary alerting mechanisms.

- Iterative Development: Continuously refine the MVP based on user feedback, incorporating Python libraries like NumPy, Pandas, and Matplotlib for data manipulation and visualization.

5.Test:

- User Testing: Collaborate with stakeholders to test the prototype, gather feedback, and identify usability issues or improvements.

- Data Validation: Evaluate the predictive models using historical data and compare their accuracy against actual earthquake events.

6.Feedback and Iterate:

- Feedback Loop: Integrate user feedback and data validation results to refine the system further. Reiterate the design and development phases as needed.

7.Develop:

- Scaling and Optimization: Enhance the system's scalability, ensuring it can handle a growing volume of data and users. Utilize Python's capabilities for distributed computing, such as Dask or Apache Spark, if necessary.

- Machine Learning Models: Develop machine learning models in Python, leveraging libraries like scikit-learn or TensorFlow, to improve prediction accuracy.

- Early Warning System: Implement real-time monitoring and early warning systems using Python for instant alerts.

8.Implement:

- Deployment: Deploy the earthquake prediction system on appropriate platforms, whether web-based applications, mobile apps, or dedicated hardware.

- Security and Privacy: Address security concerns and data privacy issues, ensuring the system complies with relevant regulations and standards.

9.Measure and Monitor:

- Performance Metrics: Define key performance metrics, such as prediction accuracy, response time, and user satisfaction, and continuously monitor them.
- Maintenance and Updates: Regularly update the system to adapt to changing seismic patterns, improve models, and enhance user experience.

Design into innovation for earthquake prediction:

Design Thinking Principles:

1.Empathy:

Stakeholder Engagement: Collaborate closely with seismologists, data scientists, and community leaders to deeply understand their needs, challenges, and aspirations.

2.Define:

User-Centric Problem Definition: Refine our focus by articulating user needs inclear and actionable terms, e.g., "Empower local communities with early earthquake warnings."

3.Ideate:

Divergent Thinking: Foster a culture of brainstorming, encouraging a wide array of ideas, from novel data sources to advanced AI algorithms.

4.Prototype:

Rapid Prototyping: Develop high-fidelity prototypes integrating Python libraries and state-of-the-art visualization tools to allow stakeholders to experience the system's potential.

5.Test:

User-Centered Evaluation: Engage stakeholders in the evaluation process to ensure that the prototype aligns with their expectations.

6.Feedback and Iterate:

Agile Development: Use feedback to drive iterative development, ensuring the system adapts rapidly to emerging challenges and opportunities.

Innovative Elements:

1.Earthquake Swarm Detection:

- Concept: Utilize machine learning and Python to detect swarms of minor earthquakes that might precede a major event.
- Innovation: Real-time analysis of seismic data streams to recognize subtle patterns and anomalies.

2.Citizen-Science Integration:

- Concept: Engage the public in data collection using mobile apps and IoT devices.
- Innovation: Crowd-sourced data provides a wealth of real-time information for enhanced prediction.

3.Earth Observation Data Fusion:

- Concept: Combine satellite data, climate information, and geological surveys for comprehensive analysis.
- Innovation: Python's geospatial libraries enhance predictive accuracy by integrating multiple data sources.

4.Explainable AI for Early Warning:

- Concept: Employ interpretable AI models to deliver early warnings to the public.
- Innovation: Transparent and reliable alerts build trust and encourage preparedness.

5.Quantum Computing for Seismic Analysis:

- Concept: Investigate quantum computing's potential for complex seismic analysis.
- Innovation: Quantum algorithms could significantly accelerate computations and enable faster predictions.

Implementation and Collaboration:

- Python Ecosystem: Utilize Python for data preprocessing, machine learning, real-time monitoring, and data visualization.
- Cloud Computing: Leverage cloud platforms for scalability and resource-intensive tasks.
- Interdisciplinary Collaboration: Foster partnerships with experts in machine learning, geophysics, and citizen-science engagement.

Impact Measurement:

- Accuracy Metrics: Establish metrics to measure the effectiveness of predictions.

Building loading and Preprocessing the dataset:

Data Collection:

- To build a dataset for earthquake prediction, you need historical seismic data.
- You can obtain this data from various sources such as the United States Geological Survey (USGS) API or other seismic data repositories. Python libraries like requests can help you fetch data from APIs.

Python:

```python
import requests
url = "https://earthquake.usgs.gov/fdsnws/event/1/query"
params = {
    "format": "geojson",
    "starttime": "2010-01-01",
    "endtime": "2020-12-31",
    "minmagnitude": 5.0,
    "minlatitude": 30.0,
    "maxlatitude": 50.0,
    "minlongitude": -120.0,
    "maxlongitude": -70.0,
}

response = requests.get(url, params=params)
data = response.json()
earthquake_data = data["features"]
```

2.Data Preprocessing:

- Data preprocessing is essential to clean and prepare the dataset for machine learning.
- Here are the key steps:
  - a.Data Cleaning
  - b.Feature Engineering
  - c.Data Transformation
  - d.Splitting Data

Python:

```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(features, labels,
test_size=0.2, random_state=42)
```

3. Feature Selection:

- You can perform feature selection to choose the most informative attributes for your model.

4. Model Building:

- Choose a suitable machine learning model for earthquake prediction.
- You can use various models such as decision trees, random forests, support vector machines, or neural networks.
- Implement and train your selected model on the preprocessed dataset.

Python:

```python
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

5. Model Evaluation:

- Assess the model's performance using metrics like accuracy, precision, recall, F1 score, and ROC AUC.
- Make sure to test the model on the testing dataset.

Python:

```python
from sklearn.metrics import accuracy_score, classification_report
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

6. Fine-tuning:

- Optimize your model's hyperparameters to improve its performance.
- This can be done using techniques like grid search or random search.

7. Deployment:

- Once your model is trained and evaluated, you can deploy it for real-time earthquake prediction, integrating it into a web application or an early warning system.

Performing different activities like Feature Engineering,Model Training,Evaluation:

1.Feature Engineering:

- This initial phase involves the selection and transformation of relevant features from the earthquake dataset.
- Feature engineering aims to extract meaningful information from the data and create new features if necessary.
- For example, one might extract temporal information, such as the year and month of an earthquake, or calculate the distance between earthquake locations and fault lines, which can provide valuable predictive insights.
- These engineered features help the model to better capture patterns and relationships in the data.

2.Model Training:

- With the dataset now prepared and the features engineered, the next step is to select a machine learning model.
- In this context, a model like the Random Forest classifier is often employed due to its ability to handle classification tasks effectively.
- The selected model is trained on a portion of the dataset, allowing it to learn the relationships between the features and the target variable, which, in this case, could represent the likelihood of an earthquake occurrence.

3.Evaluation:

- Model performance evaluation is critical for assessing the model's accuracy and effectiveness.
- After the model is trained, it is tested on a separate portion of the dataset (the test set) to gauge its predictive capabilities.
- Common evaluation metrics include accuracy, precision, recall, F1 score, and the generation of a classification report.
- These metrics help quantify the model's ability to correctly predict earthquakes and non-earthquake events.
- Evaluating the model ensures that it meets the desired performance standards and provides insights into any potential improvements or fine-tuning required.

Feature selection for earthquake prediction:

1.Correlation Analysis:

- Calculate the correlation between each feature and earthquake occurrence to identify highly correlated features.

2.Feature Importance from Trees:

- Use tree-based models like Random Forest to extract feature importance scores and select the most valuable features.

3.Recursive Feature Elimination (RFE):

- Apply RFE to recursively remove the least important features until a desired number is reached.

4.L1 Regularization (Lasso):

- Use L1 regularization in linear models to encourage feature selection by zeroing out unimportant feature coefficients.

5.Mutual Information:

- Measure feature-target dependency using mutual information, and select features with high mutual information scores.

6.Principal Component Analysis (PCA):

- Apply PCA to reduce dimensionality by transforming features into a lower-dimensional space while retaining most of the information.

Advantages:

1.Versatile Toolset:

- Python offers a versatile and comprehensive set of libraries for data processing, machine learning, and geospatial analysis, making it well-suited for various aspects of earthquake prediction.

2.Data Analysis Efficiency:

- Python's libraries, like Pandas and NumPy, allow for efficient data processing and cleaning of large seismic datasets, enhancing the accuracy of prediction models.

3.Machine Learning Capabilities:

- Python's machine learning libraries, such as scikit-learn and TensorFlow, enable the development of predictive models that learn from historical seismic data and geological features, improving prediction accuracy.

4.Geospatial Analysis:

- Python's geospatial libraries, including GeoPandas, Folium, and Basemap, facilitate geospatial analysis to understand the geographical distribution of earthquakes and related risks.

5.Community Support:

- Python has a large and active user community, providing access to a wealth of resources, collaborative opportunities, and open-source solutions, reducing the cost and effort associated with earthquake prediction research and development.

Disadvantage:

1.Limited Real-Time Processing:

- Python's interpreted nature may not be as efficient as lower-level languages, which can lead to limitations in real-time processing and responsiveness, especially when dealing with a high volume of data and the need for immediate earthquake alerts.

2.Performance Constraints:

- High-performance computing requirements, such as large-scale simulations, may not be as efficiently executed in Python.

3.Resource Intensive:

- Machine learning models and geospatial analysis can be resource-intensive processes.
- Running complex models on large datasets may require substantial computing power, which can be costly and not readily available to all researchers.

4.Complexity of Model Development:

- While Python provides excellent machine learning libraries, developing and fine-tuning predictive models can be complex and require expertise in both data science and seismology.
- Understanding the underlying algorithms and parameters is crucial for accurate predictions.

5.Dependency on Data Quality:

- The accuracy of earthquake prediction models relies heavily on the quality of input data.
- Any inaccuracies or biases in the data can lead to inaccurate predictions.
- Python itself doesn't mitigate data quality issues, and extensive data preprocessing may be required.

Benefits of earthquake prediction using python:

- Data Processing Efficiency
- Machine Learning
- Real-Time Monitoring
- Geospatial Analysis
- Visualization Tools
- Interdisciplinary Collaboration
- Scalability
- Open Source and Cost-Effective
- Community Support
- Adaptability and Transparency

PROGRAM:

Earthquake Predictor ETL

Import Statements

[1]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing;
from sklearn import model_selection;
from sklearn import linear_model;
import os
import datetime as dt
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df=pd.read_csv('database.csv')
```

[3]:
```
df.head()
```

[3]:
```
         Date      Time  Latitude  Longitude        Type  Depth  Depth Error  \
0  01/02/1965  13:44:18    19.246    145.616  Earthquake  131.6          NaN
1  01/04/1965  11:29:49     1.863    127.352  Earthquake   80.0          NaN
2  01/05/1965  18:05:58   -20.579   -173.972  Earthquake   20.0          NaN
3  01/08/1965  18:49:43   -59.076    -23.557  Earthquake   15.0          NaN
4  01/09/1965  13:32:50    11.938    126.427  Earthquake   15.0          NaN

   Depth Seismic Stations  Magnitude Magnitude Type  ...  \
0                     NaN        6.0             MW  ...
1                     NaN        5.8             MW  ...
2                     NaN        6.2             MW  ...
3                     NaN        5.8             MW  ...
4                     NaN        5.8             MW  ...

   Magnitude Seismic Stations  Azimuthal Gap  Horizontal Distance  \
0                         NaN            NaN                  NaN
1                         NaN            NaN                  NaN
2                         NaN            NaN                  NaN
3                         NaN            NaN                  NaN
```

```
4                          NaN          NaN                 NaN

   Horizontal  Error   RootMeanSquare           ID  Source  Location  Source   \
0              NaN              NaN  ISCGEM860706 ISCGEM            ISCGEM
1              NaN              NaN  ISCGEM860737 ISCGEM            ISCGEM
2              NaN              NaN  ISCGEM860762 ISCGEM            ISCGEM
3              NaN              NaN  ISCGEM860856 ISCGEM            ISCGEM
4              NaN              NaN  ISCGEM860890 ISCGEM            ISCGEM

  Magnitude  Source      Status
0          ISCGEM  Automatic
1          ISCGEM  Automatic
2          ISCGEM  Automatic
3          ISCGEM  Automatic
4          ISCGEM  Automatic

[5 rows x 21 columns]
```

[4]:
```
df.tail()
```

[4]:
```
             Date      Time  Latitude  Longitude        Type  Depth  \
23407  12/28/2016  08:22:12   38.3917  -118.8941  Earthquake  12.30
23408  12/28/2016  09:13:47   38.3777  -118.8957  Earthquake   8.80
23409  12/28/2016  12:38:51   36.9179   140.4262  Earthquake  10.00
23410  12/29/2016  22:30:19   -9.0283   118.6639  Earthquake  79.00
23411  12/30/2016  20:08:28   37.3973   141.4103  Earthquake  11.94

       Depth  Error   Depth Seismic Stations   Magnitude  Magnitude  Type  ...  \
23407            1.2                     40.0        5.6              ML  ...
23408            2.0                     33.0        5.5              ML  ...
23409            1.8                      NaN        5.9             MWW  ...
23410            1.8                      NaN        6.3             MWW  ...
23411            2.2                      NaN        5.5              MB  ...

       Magnitude Seismic Stations   Azimuthal Gap  Horizontal  Distance   \
23407                        18.0           42.47                  0.120
23408                        18.0           48.58                  0.129
23409                         NaN           91.00                  0.992
23410                         NaN           26.00                  3.553
23411                       428.0           97.00                  0.681

       Horizontal  Error   RootMeanSquare          ID  Source  Location  Source   \
23407                NaN           0.1898  NN00570710      NN                NN
23408                NaN           0.2187  NN00570744      NN                NN
23409                4.8           1.5200  US10007NAF      US                US
23410                6.0           1.4300  US10007NL0      US                US
23411                4.5           0.9100  US10007NTD      US                US
```

```
       Magnitude Source    Status
23407                 NN  Reviewed
23408                 NN  Reviewed
23409                 US  Reviewed
23410                 US  Reviewed
23411                 US  Reviewed

[5 rows x 21 columns]
```

[5]:
```
df.shape
```

[5]: (23412, 21)

[6]:
```
df.describe()
```

[6]:
```
           Latitude     Longitude         Depth  Depth Error  \
count  23412.000000  23412.000000  23412.000000  4461.000000
mean       1.679033     39.639961     70.767911     4.993115
std       30.113183    125.511959    122.651898     4.875184
min      -77.080000   -179.997000     -1.100000     0.000000
25%      -18.653000    -76.349750     14.522500     1.800000
50%       -3.568500    103.982000     33.000000     3.500000
75%       26.190750    145.026250     54.000000     6.300000
max       86.005000    179.998000    700.000000    91.295000

       Depth Seismic Stations     Magnitude  Magnitude Error  \
count             7097.000000  23412.000000       327.000000
mean               275.364098      5.882531         0.071820
std                162.141631      0.423066         0.051466
min                  0.000000      5.500000         0.000000
25%                146.000000      5.600000         0.046000
50%                255.000000      5.700000         0.059000
75%                384.000000      6.000000         0.075500
max                934.000000      9.100000         0.410000

       Magnitude Seismic Stations  Azimuthal Gap  Horizontal Distance  \
count                 2564.000000    7299.000000          1604.000000
mean                    48.944618      44.163532             3.992660
std                     62.943106      32.141486             5.377262
min                      0.000000       0.000000             0.004505
25%                     10.000000      24.100000             0.968750
50%                     28.000000      36.000000             2.319500
75%                     66.000000      54.000000             4.724500
max                    821.000000     360.000000            37.874000

       Horizontal Error  RootMeanSquare
```

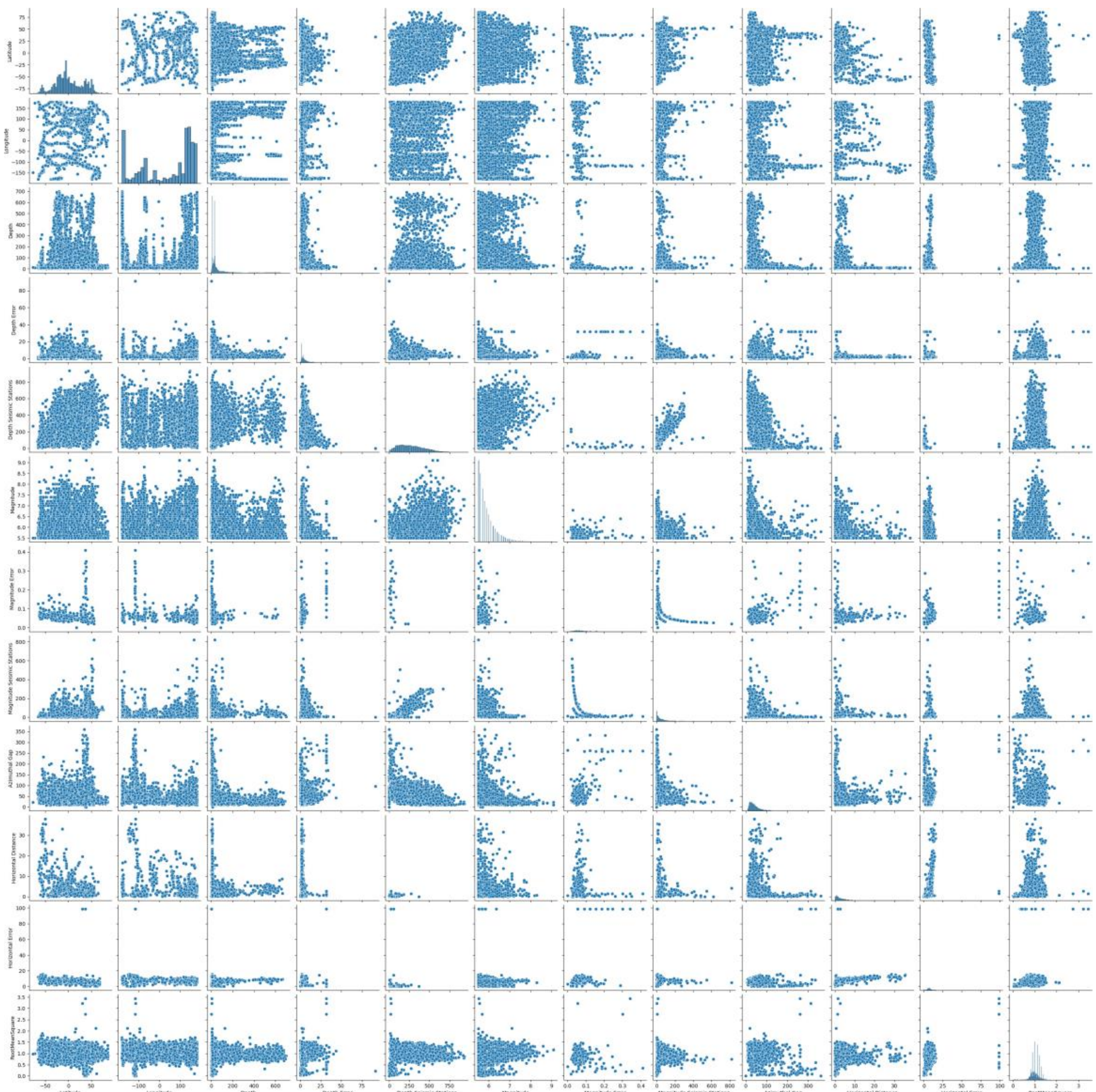
```
count   1156.000000  17352.000000
mean       7.662759      1.022784
std       10.430396      0.188545
min        0.085000      0.000000
25%        5.300000      0.900000
50%        6.700000      1.000000
75%        8.100000      1.130000
max       99.000000      3.440000
```

[7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 21 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  ------
 0   Date                        23412 non-null  object
 1   Time                        23412 non-null  object
 2   Latitude                    23412 non-null  float64
 3   Longitude                   23412 non-null  float64
 4   Type                        23412 non-null  object
 5   Depth                       23412 non-null  float64
 6   Depth Error                 4461 non-null   float64
 7   Depth Seismic Stations      7097 non-null   float64
 8   Magnitude                   23412 non-null  float64
 9   Magnitude Type              23409 non-null  object
 10  Magnitude Error             327 non-null    float64
 11  Magnitude Seismic Stations  2564 non-null   float64
 12  Azimuthal Gap               7299 non-null   float64
 13  Horizontal Distance         1604 non-null   float64
 14  Horizontal Error            1156 non-null   float64
 15  RootMeanSquare              17352 non-null  float64
 16  ID                          23412 non-null  object
 17  Source                      23412 non-null  object
 18  Location Source             23412 non-null  object
 19  Magnitude Source            23412 non-null  object
 20  Status                      23412 non-null  object
dtypes: float64(12), object(9)
memory usage: 3.8+ MB
```

[8]:
```
sns.pairplot(df)
```

[8]: <seaborn.axisgrid.PairGrid at 0x22d1357f210>

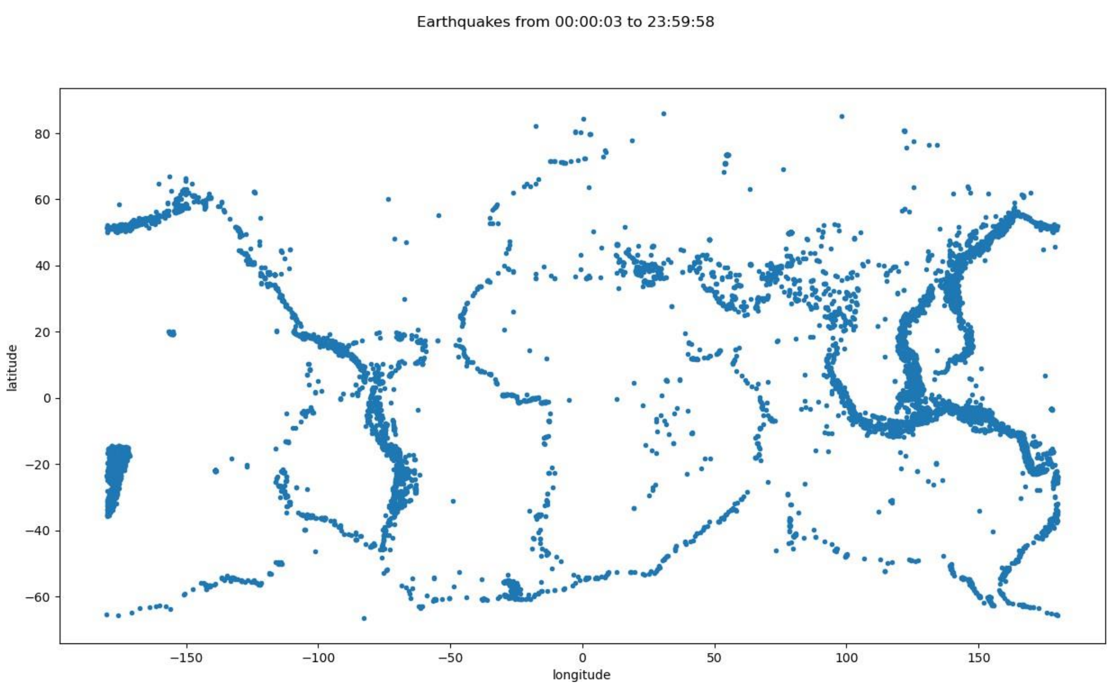


[9]:
```
df.isnull().sum()
```

[9]:
```
Date                          0
Time                          0
Latitude                      0
Longitude                     0
Type                          0
Depth                         0
Depth Error               18951
Depth Seismic Stations    16315
Magnitude                     0
Magnitude Type                3
Magnitude Error           23085
```

```
Magnitude Seismic Stations      20848
Azimuthal Gap                   16113
Horizontal Distance             21808
Horizontal Error                22256
RootMeanSquare                   6060
ID                                  0
Source                              0
Location Source                     0
Magnitude Source                    0
Status                              0
dtype: int64
```

[10]:
```
rounding_factor = 10
fig, ax = plt.subplots(figsize=(15,8))

# latitude and longitude of earthquake site of top 10500 samples.
plt.plot(np.round(df['Longitude'].head(10500),rounding_factor),
         np.round(df['Latitude'].head(10500),rounding_factor),
         linestyle='none', marker='.')

plt.suptitle('Earthquakes from ' + str(np.min(df['Time']))[:20] + ' to ' +␣
 ↪str(np.max(df['Time']))[:20])
plt.xlabel('longitude')
plt.ylabel('latitude')
plt.show()
```



Data preprocessing

```
[11]: new_column_names = ["Date(YYYY/MM/DD)",      "Time(UTC)",   "Latitude(deg)",␣
    ↪"Longitude(deg)","Type",
                          "Depth(km)", "Depth Error(km)","Depth Seismic Stations(km)",
                          "Magnitude", "Magnitude_type", "Magnitude Error","Magnitude␣
    ↪Seismic Stations",
                          "Azimuthal Gap", "Horizontal Distance", "Horizontal␣
    ↪Error"," RootMeanSquare  ",
                          "ID ", "Source", "Location Source",
                         "Magnitude Source","Status"]
      df.columns = new_column_names
      ts = pd.to_datetime(df["Date(YYYY/MM/DD)"])
      df = df.drop(["Date(YYYY/MM/DD)"], axis=1)
      df.index = ts
      display(df)
```

```
                 Time(UTC)  Latitude(deg)  Longitude(deg)        Type  \
Date(YYYY/MM/DD)
1965-01-02        13:44:18        19.2460        145.6160  Earthquake
1965-01-04        11:29:49         1.8630        127.3520  Earthquake
1965-01-05        18:05:58       -20.5790       -173.9720  Earthquake
1965-01-08        18:49:43       -59.0760        -23.5570  Earthquake
1965-01-09        13:32:50        11.9380        126.4270  Earthquake
...                    ...            ...             ...         ...
2016-12-28        08:22:12        38.3917       -118.8941  Earthquake
2016-12-28        09:13:47        38.3777       -118.8957  Earthquake
2016-12-28        12:38:51        36.9179        140.4262  Earthquake
2016-12-29        22:30:19        -9.0283        118.6639  Earthquake
2016-12-30        20:08:28        37.3973        141.4103  Earthquake

                  Depth(km)  Depth Error(km)  Depth Seismic Stations(km)  \
Date(YYYY/MM/DD)
1965-01-02           131.60              NaN                         NaN
1965-01-04            80.00              NaN                         NaN
1965-01-05            20.00              NaN                         NaN
1965-01-08            15.00              NaN                         NaN
1965-01-09            15.00              NaN                         NaN
...                     ...              ...                         ...
2016-12-28            12.30              1.2                        40.0
2016-12-28             8.80              2.0                        33.0
2016-12-28            10.00              1.8                         NaN
2016-12-29            79.00              1.8                         NaN
2016-12-30            11.94              2.2                         NaN

                  Magnitude  Magnitude_type  Magnitude Error  \
Date(YYYY/MM/DD)
```

1965-01-02	6.0	MW	NaN
1965-01-04	5.8	MW	NaN
1965-01-05	6.2	MW	NaN
1965-01-08	5.8	MW	NaN
1965-01-09	5.8	MW	NaN
...	...	...	...
2016-12-28	5.6	ML	0.320
2016-12-28	5.5	ML	0.260
2016-12-28	5.9	MWW	NaN
2016-12-29	6.3	MWW	NaN
2016-12-30	5.5	MB	0.029

Date(YYYY/MM/DD)	Magnitude Seismic Stations	Azimuthal Gap \
1965-01-02	NaN	NaN
1965-01-04	NaN	NaN
1965-01-05	NaN	NaN
1965-01-08	NaN	NaN
1965-01-09	NaN	NaN
...	...	...
2016-12-28	18.0	42.47
2016-12-28	18.0	48.58
2016-12-28	NaN	91.00
2016-12-29	NaN	26.00
2016-12-30	428.0	97.00

Date(YYYY/MM/DD)	Horizontal Distance	Horizontal Error	RootMeanSquare \
1965-01-02	NaN	NaN	NaN
1965-01-04	NaN	NaN	NaN
1965-01-05	NaN	NaN	NaN
1965-01-08	NaN	NaN	NaN
1965-01-09	NaN	NaN	NaN
...	...	...	...
2016-12-28	0.120	NaN	0.1898
2016-12-28	0.129	NaN	0.2187
2016-12-28	0.992	4.8	1.5200
2016-12-29	3.553	6.0	1.4300
2016-12-30	0.681	4.5	0.9100

Date(YYYY/MM/DD)	ID	Source	Location Source	Magnitude Source \
1965-01-02	ISCGEM860706	ISCGEM	ISCGEM	ISCGEM
1965-01-04	ISCGEM860737	ISCGEM	ISCGEM	ISCGEM
1965-01-05	ISCGEM860762	ISCGEM	ISCGEM	ISCGEM
1965-01-08	ISCGEM860856	ISCGEM	ISCGEM	ISCGEM
1965-01-09	ISCGEM860890	ISCGEM	ISCGEM	ISCGEM
...	...	...	...	...

```
2016-12-28         NN00570710     NN        NN        NN
2016-12-28         NN00570744     NN        NN        NN
2016-12-28          US10007NAF    US        US        US
2016-12-29          US10007NL0    US        US        US
2016-12-30         US10007NTD     US        US        US

                        Status
Date(YYYY/MM/DD)

1965-01-02          Automatic
1965-01-04          Automatic
1965-01-05          Automatic
1965-01-08          Automatic
1965-01-09          Automatic
...                      ...
2016-12-28           Reviewed
2016-12-28           Reviewed
2016-12-28           Reviewed
2016-12-29           Reviewed
2016-12-30           Reviewed

[23412 rows x 20 columns]
```

[12]:

```
df = df.sort_values('Time(UTC)', ascending=True)

#Date extraction
df['Date'] = df['Time(UTC)'].str[0:10]
df.head()
```

[12]:

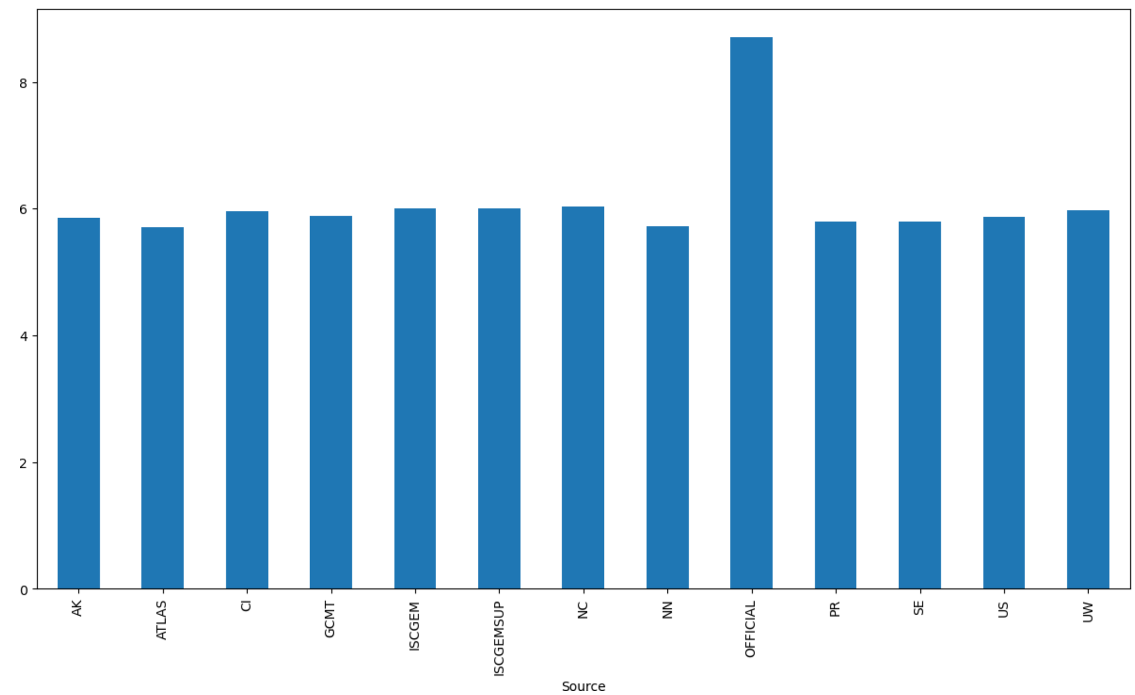
```
                        Time(UTC)  Latitude(deg)  Longitude(deg)        Type  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00      00:00:03         1.8850        127.3630  Earthquake
1967-07-30 00:00:00      00:00:04        10.5590        -67.3300  Earthquake
2008-09-14 00:00:00      00:00:09        -8.7280        126.8550  Earthquake
2016-01-25 00:00:00      00:00:11       -19.5324       -173.3833  Earthquake
1970-06-14 00:00:00      00:00:11       -52.0280        -74.0700  Earthquake

                        Depth(km)  Depth Error(km)  Depth Seismic Stations(km)  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00          96.0              NaN                       463.0
1967-07-30 00:00:00          25.0              NaN                         NaN
2008-09-14 00:00:00          22.0              NaN                       237.0
2016-01-25 00:00:00          53.0              1.8                         NaN
1970-06-14 00:00:00          15.0              NaN                         NaN

                        Magnitude  Magnitude_type  Magnitude Error  ...  \
Date(YYYY/MM/DD)                                                    ...
2008-09-11 00:00:00           6.6             MWC              NaN  ...
```

```
1967-07-30 00:00:00        6.6              MW                 NaN  ...
2008-09-14 00:00:00        5.6             MWC                 NaN  ...
2016-01-25 00:00:00        5.7             MWW                 NaN  ...
1970-06-14 00:00:00        7.0              MW                 NaN  ...

                     Azimuthal Gap  Horizontal Distance  Horizontal Error  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00           14.1                  NaN               NaN
1967-07-30 00:00:00            NaN                  NaN               NaN
2008-09-14 00:00:00           38.8                  NaN               NaN
2016-01-25 00:00:00           34.0                3.295               6.2
1970-06-14 00:00:00            NaN                  NaN               NaN

                     RootMeanSquare            ID  Source Location Source  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00            1.02    USP000GGU7      US              US
1967-07-30 00:00:00             NaN  ISCGEM833882  ISCGEM          ISCGEM
2008-09-14 00:00:00            0.97    USP000GH0E      US              US
2016-01-25 00:00:00            1.36    US10004GX7      US              US
1970-06-14 00:00:00             NaN  ISCGEM794565  ISCGEM          ISCGEM

                    Magnitude Source     Status      Date
Date(YYYY/MM/DD)


2008-09-11 00:00:00               US   Reviewed  00:00:03
1967-07-30 00:00:00           ISCGEM  Automatic  00:00:04
2008-09-14 00:00:00             GCMT   Reviewed  00:00:09
2016-01-25 00:00:00               US   Reviewed  00:00:11
1970-06-14 00:00:00           ISCGEM  Automatic  00:00:11

[5 rows x 21 columns]
```

[13]:
```
print('total locations:',len(set(df['Source'])))
```

```
total locations: 13
```

[14]:
```
df.groupby(['Source'])['Magnitude'].mean().plot(kind='bar',figsize=(15,8));
```



[15]:
```
df_coords = df[['Source', 'Latitude(deg)', 'Longitude(deg)']]
df_coords = df_coords.groupby(['Source'], as_index=False).mean()
df_coords = df[['Source', 'Latitude(deg)', 'Longitude(deg)']]
```

[16]:
```
df_coords.head()
```

[16]:
```
                    Source  Latitude(deg)  Longitude(deg)
Date(YYYY/MM/DD)
2008-09-11 00:00:00     US         1.8850        127.3630
1967-07-30 00:00:00 ISCGEM        10.5590        -67.3300
2008-09-14 00:00:00     US        -8.7280        126.8550
2016-01-25 00:00:00     US       -19.5324       -173.3833
1970-06-14 00:00:00 ISCGEM       -52.0280        -74.0700
```

[17]:
```
df.head()
```

[17] :
```
                     Time(UTC) Latitude(deg)  Longitude(deg)        Type  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00   00:00:03        1.8850        127.3630  Earthquake
1967-07-30 00:00:00   00:00:04       10.5590        -67.3300  Earthquake
2008-09-14 00:00:00   00:00:09       -8.7280        126.8550  Earthquake
2016-01-25 00:00:00   00:00:11      -19.5324       -173.3833  Earthquake
1970-06-14 00:00:00   00:00:11      -52.0280        -74.0700  Earthquake

                     Depth(km)  Depth  Error(km)   Depth Seismic Stations(km)    \
```

```
Date(YYYY/MM/DD)
2008-09-11 00:00:00      96.0        NaN                  463.0
1967-07-30 00:00:00      25.0        NaN                    NaN
2008-09-14 00:00:00      22.0        NaN                  237.0
2016-01-25 00:00:00      53.0        1.8                    NaN
1970-06-14 00:00:00      15.0        NaN                    NaN

                     Magnitude  Magnitude_type  Magnitude Error  ...  \
Date(YYYY/MM/DD)                                                  ...
2008-09-11 00:00:00        6.6             MWC              NaN  ...
1967-07-30 00:00:00        6.6              MW              NaN  ...
2008-09-14 00:00:00        5.6             MWC              NaN  ...
2016-01-25 00:00:00        5.7             MWW              NaN  ...
1970-06-14 00:00:00        7.0              MW              NaN  ...

                     Azimuthal Gap  Horizontal  Distance   Horizontal Error  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00           14.1                   NaN               NaN
1967-07-30 00:00:00            NaN                   NaN               NaN
2008-09-14 00:00:00           38.8                   NaN               NaN
2016-01-25 00:00:00           34.0                 3.295               6.2
1970-06-14 00:00:00            NaN                   NaN               NaN

                     RootMeanSquare           ID  Source  Location  Source  \
Date(YYYY/MM/DD)
2008-09-11 00:00:00            1.02    USP000GGU7      US               US
1967-07-30 00:00:00             NaN  ISCGEM833882  ISCGEM           ISCGEM
2008-09-14 00:00:00            0.97    USP000GH0E      US               US
2016-01-25 00:00:00            1.36    US10004GX7      US               US
1970-06-14 00:00:00             NaN  ISCGEM794565  ISCGEM           ISCGEM

                    Magnitude  Source     Status      Date
Date(YYYY/MM/DD)
2008-09-11 00:00:00                US   Reviewed  00:00:03
1967-07-30 00:00:00            ISCGEM  Automatic  00:00:04
2008-09-14 00:00:00              GCMT   Reviewed  00:00:09
2016-01-25 00:00:00                US   Reviewed  00:00:11
1970-06-14 00:00:00            ISCGEM  Automatic  00:00:11

[5 rows x 21 columns]
```

Feature Engineering and Data wrangling

[18] :
```
eq_tmp = df.copy()

#rolling window size
DAYS_OUT_TO_PREDICT = 7

# loop through each zone and apply MA
eq_data = []
eq_data_last_days_out = []

for place in list(set(eq_tmp['Source'])):
    temp_df = eq_tmp[eq_tmp['Source'] == place].copy()

    #avg. depth of 22 days rolling period and so on..
    temp_df['depth_avg_22'] = temp_df['Depth(km)'].
 ↪rolling(window=22,center=False).mean()
    temp_df['depth_avg_15'] = temp_df['Depth(km)'].
 ↪rolling(window=15,center=False).mean()
    temp_df['depth_avg_7'] = temp_df['Depth(km)'].
 ↪rolling(window=7,center=False).mean()
    temp_df['mag_avg_22'] = temp_df['Magnitude'].
 ↪rolling(window=22,center=False).mean()
    temp_df['mag_avg_15'] = temp_df['Magnitude'].
 ↪rolling(window=15,center=False).mean()
    temp_df['mag_avg_7'] = temp_df['Magnitude'].rolling(window=7,center=False).
 ↪mean()
    temp_df.loc[:, 'mag_outcome'] = temp_df.loc[:, 'mag_avg_7'].
 ↪shift(DAYS_OUT_TO_PREDICT * -1)

    #days to predict value on earth quake data this is not yet seen or␣
 ↪witnessed by next 7 days (consider as live next 7 days period)

    eq_data_last_days_out.append(temp_df.tail(DAYS_OUT_TO_PREDICT))

    eq_data.append(temp_df)
```

[19] :
```
eq_all = pd.concat(eq_data)
```

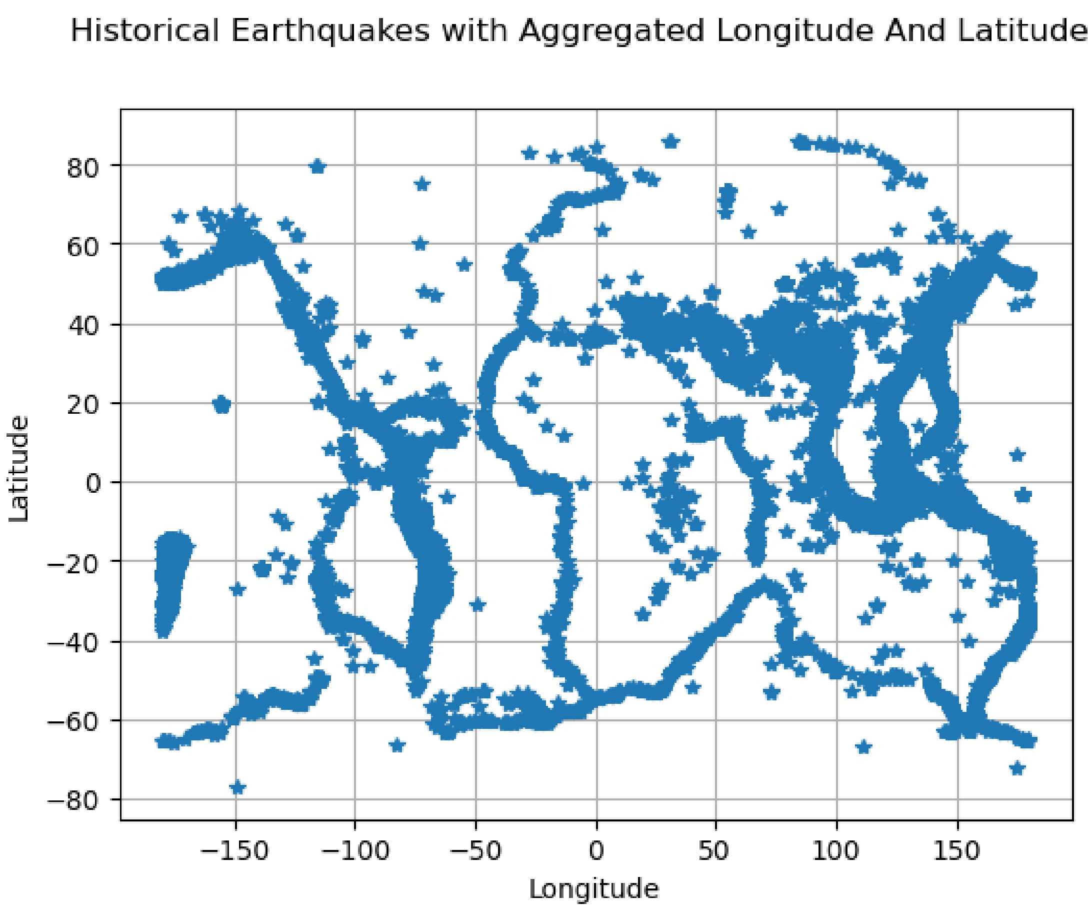
[20] :
```
eq_all.head()
```

[20] :
```
                    Time(UTC) Latitude(deg)  Longitude(deg)        Type  \
Date(YYYY/MM/DD)
2015-07-29 00:00:00  02:35:59       59.8935       -153.1961  Earthquake
2014-06-07 00:00:00  04:43:32       67.7245       -162.3749  Earthquake
2016-04-02 00:00:00  05:50:00       57.0080       -157.9321  Earthquake
```

```
2015-05-29 00:00:00    07:00:09         56.5940       -156.4300  Earthquake
2014-05-03 00:00:00    08:57:12         67.6302       -162.2066  Earthquake

                       Depth(km)  Depth Error(km)   Depth Seismic Stations(km)   \
Date(YYYY/MM/DD)
2015-07-29 00:00:00        119.3              0.2                          NaN
2014-06-07 00:00:00         18.6              0.9                          NaN
2016-04-02 00:00:00         11.4              0.3                          NaN
2015-05-29 00:00:00         72.6              0.8                          NaN
2014-05-03 00:00:00          0.9              8.2                          NaN

                       Magnitude  Magnitude_type  Magnitude  Error   ...  \
Date(YYYY/MM/DD)                                                     ...
2015-07-29 00:00:00          6.3              MS              NaN    ...
2014-06-07 00:00:00          5.5              ML              NaN    ...
2016-04-02 00:00:00          5.9              ML              NaN    ...
2015-05-29 00:00:00          6.7              MS              NaN    ...
2014-05-03 00:00:00          5.5              MB              NaN    ...

                       Magnitude  Source     Status        Date   depth_avg_22  \
Date(YYYY/MM/DD)
2015-07-29 00:00:00                   AK   Reviewed  02:35:59            NaN
2014-06-07 00:00:00                   AK   Reviewed  04:43:32            NaN
2016-04-02 00:00:00                   AK   Reviewed  05:50:00            NaN
2015-05-29 00:00:00                   AK   Reviewed  07:00:09            NaN
2014-05-03 00:00:00                   AK   Reviewed  08:57:12            NaN

                       depth_avg_15  depth_avg_7  mag_avg_22  mag_avg_15  mag_avg_7   \
Date(YYYY/MM/DD)
2015-07-29 00:00:00             NaN          NaN         NaN         NaN        NaN
2014-06-07 00:00:00             NaN          NaN         NaN         NaN        NaN
2016-04-02 00:00:00             NaN          NaN         NaN         NaN        NaN
2015-05-29 00:00:00             NaN          NaN         NaN         NaN        NaN
2014-05-03 00:00:00             NaN          NaN         NaN         NaN        NaN

                      mag_outcome
Date(YYYY/MM/DD)
2015-07-29 00:00:00      5.842857
2014-06-07 00:00:00      5.857143
2016-04-02 00:00:00      5.900000
2015-05-29 00:00:00      5.742857
2014-05-03 00:00:00      5.771429

[5 rows x 28 columns]
```

location after feature engineering

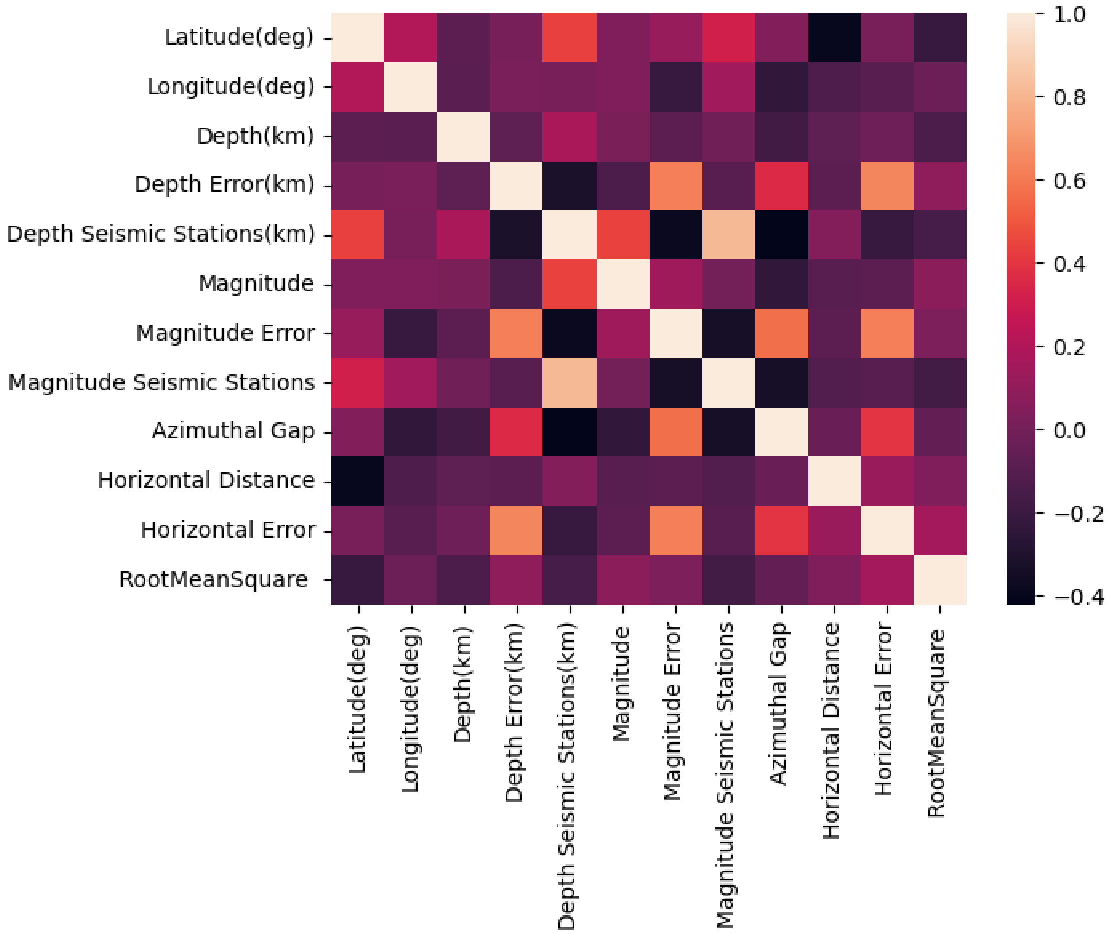
[21] :
```
plt.plot(eq_all['Longitude(deg)'],
         eq_all['Latitude(deg)'],
         linestyle='none', marker='*')
plt.suptitle('Historical Earthquakes with Aggregated Longitude And Latitude')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid()
plt.show()
```



[22] :
```
sns.heatmap(df.corr())
```

C:\Users\Ragu\AppData\Local\Temp\ipykernel_9820\3714479308.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr())

[22] : <Axes: >



[23] :
```
sns.distplot(df['Magnitude'])
```

C:\Users\Ragu\AppData\Local\Temp\ipykernel_9820\1899976088.py:1: UserWarning:

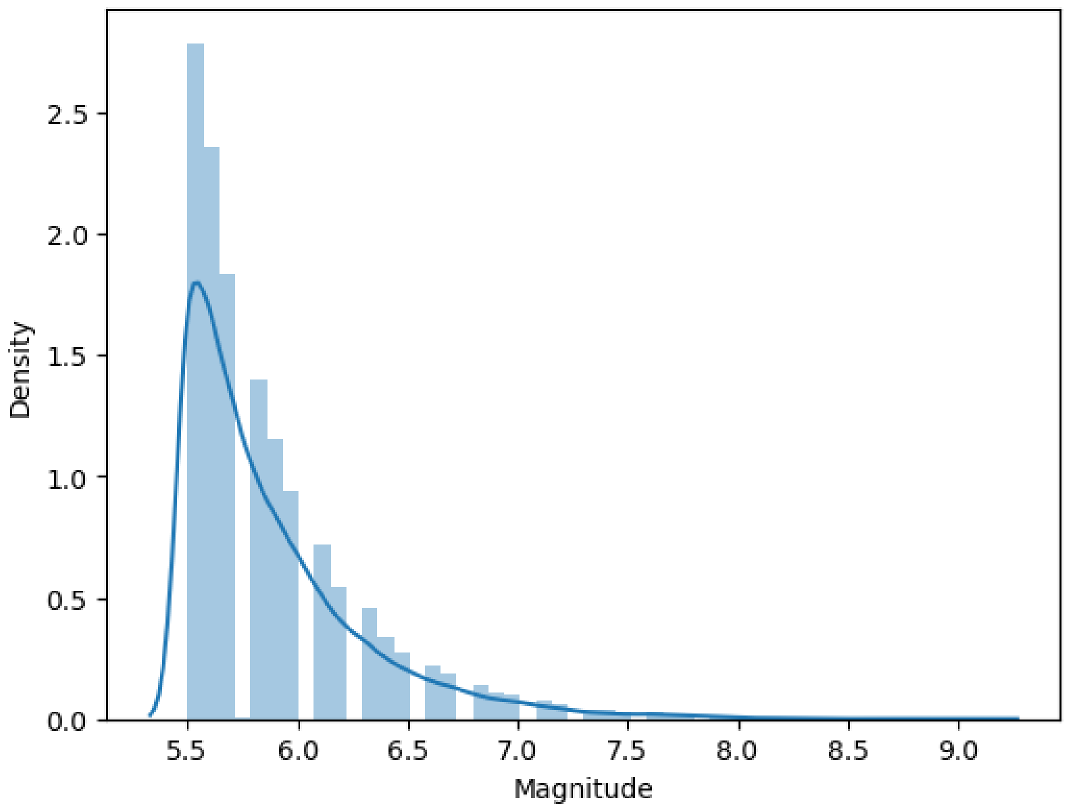
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Magnitude'])

[23] : <Axes: xlabel='Magnitude', ylabel='Density'>



[24] :
```
# keep our live data for predictions
eq_data_last_days_out = pd.concat(eq_data_last_days_out)


  eq_data_last_days_out = eq_data_last_days_out[np.
  ↪isfinite(eq_data_last_days_out['mag_avg_22'])]
predict_unknown=eq_data_last_days_out
```

[25] :
```
predict_unknown
```

[25]:

Date(YYYY/MM/DD)	Time(UTC)	Latitude(deg)	Longitude(deg)	Type	\
1970-05-25	22:45:34	-57.450000	-26.041000	Earthquake	
1969-08-11	22:53:58	44.007000	148.413000	Earthquake	
1968-03-03	22:55:36	1.584000	122.452000	Earthquake	
1972-07-30	23:17:23	-5.877000	130.525000	Earthquake	
1967-02-19	23:28:29	-0.072000	124.213000	Earthquake	
1971-07-03	23:44:51	-24.136000	-68.899000	Earthquake	
1969-08-11	23:52:58	1.848000	126.371000	Earthquake	
1978-08-13	22:54:53	34.350000	-119.700000	Earthquake	
1989-10-03	23:10:50	80.690000	121.810000	Earthquake	

1995-09-20	23:27:36	35.760000	-117.640000	Earthquake
1994-01-17	23:33:30	34.320000	-118.700000	Earthquake
1983-05-02	23:42:37	36.240000	-120.300000	Earthquake
1990-02-28	23:43:36	34.140000	-117.690000	Earthquake
1987-07-31	23:56:58	40.420000	-124.410000	Earthquake
1965-04-09	23:57:06	35.047000	24.318000	Earthquake
1971-11-05	23:57:29	-2.006000	100.126000	Earthquake
1972-11-03	23:57:56	-19.181000	-69.108000	Earthquake
1969-06-17	23:58:09	-52.744000	160.309000	Earthquake
1967-01-05	23:58:20	48.040000	103.013000	Earthquake
1965-05-15	23:58:36	-4.131000	134.946000	Earthquake
1966-03-04	23:58:56	-38.747000	178.061000	Earthquake
2010-04-04	22:50:17	32.098667	-115.048167	Earthquake
1979-10-15	23:16:54	32.667333	-115.359167	Earthquake
1993-05-17	23:20:50	37.165000	-117.774000	Earthquake
1995-09-20	23:27:36	35.761000	-117.638000	Earthquake
1994-01-17	23:33:31	34.326000	-118.698000	Earthquake
2001-12-08	23:36:10	31.997667	-115.001667	Earthquake
1990-02-28	23:43:37	34.144000	-117.697000	Earthquake
2015-05-22	23:59:34	-11.109300	163.215400	Earthquake
1983-09-27	23:59:38	36.688000	26.912000	Earthquake
2005-10-02	23:59:44	-5.595000	151.651000	Earthquake
1991-10-23	23:59:44	51.250000	178.348000	Earthquake
2014-02-24	23:59:46	4.224600	62.522700	Earthquake
2015-08-06	23:59:46	-26.457600	-178.251100	Earthquake
2014-04-01	23:59:58	-19.492800	-70.166000	Earthquake
2015-01-28	21:08:54	40.317833	-124.606667	Earthquake
1984-04-24	21:15:19	37.309667	-121.678833	Earthquake
1986-07-21	22:07:16	37.621333	-118.342500	Earthquake
1991-08-17	22:17:10	41.679000	-125.856000	Earthquake
1991-08-16	22:26:14	41.661167	-125.845500	Earthquake
1983-05-02	23:42:38	36.231667	-120.312000	Earthquake
1987-07-31	23:56:58	40.415667	-124.382667	Earthquake

	Depth(km)	Depth Error(km)	Depth Seismic Stations(km) \
Date(YYYY/MM/DD)			
1970-05-25	40.000	NaN	NaN
1969-08-11	30.000	NaN	NaN
1968-03-03	422.700	NaN	NaN
1972-07-30	90.000	NaN	NaN
1967-02-19	95.000	NaN	NaN
1971-07-03	99.200	NaN	NaN
1969-08-11	35.000	NaN	NaN
1978-08-13	10.000	NaN	NaN
1989-10-03	10.000	NaN	NaN
1995-09-20	5.000	NaN	NaN
1994-01-17	0.000	NaN	NaN

1983-05-02	12.000	NaN	NaN
1990-02-28	10.000	NaN	NaN
1987-07-31	16.000	NaN	NaN
1965-04-09	65.000	NaN	NaN
1971-11-05	35.000	NaN	NaN
1972-11-03	116.600	NaN	NaN
1969-06-17	15.000	NaN	NaN
1967-01-05	17.500	NaN	NaN
1965-05-15	25.000	NaN	NaN
1966-03-04	30.000	NaN	NaN
2010-04-04	10.011	31.610	5.0
1979-10-15	15.000	31.610	78.0
1993-05-17	3.599	NaN	0.0
1995-09-20	4.688	0.469	0.0
1994-01-17	9.083	0.840	0.0
2001-12-08	6.981	28.000	12.0
1990-02-28	3.292	0.155	0.0
2015-05-22	10.000	1.700	NaN
1983-09-27	158.600	1.800	NaN
2005-10-02	30.500	NaN	226.0
1991-10-23	33.000	NaN	NaN
2014-02-24	10.000	1.500	NaN
2015-08-06	269.000	1.800	NaN
2014-04-01	21.610	4.100	NaN
2015-01-28	17.170	0.330	52.0
1984-04-24	8.193	0.310	101.0
1986-07-21	-0.076	7.220	18.0
1991-08-17	1.303	11.060	90.0
1991-08-16	2.549	12.760	119.0
1983-05-02	9.578	0.240	64.0
1987-07-31	16.895	0.210	20.0

	Magnitude	Magnitude_type	Magnitude Error	...	\
Date(YYYY/MM/DD)				...	
1970-05-25	5.80	MW	NaN	...	
1969-08-11	5.70	MW	NaN	...	
1968-03-03	5.70	MW	NaN	...	
1972-07-30	5.90	MW	NaN	...	
1967-02-19	5.90	MW	NaN	...	
1971-07-03	5.50	MW	NaN	...	
1969-08-11	6.20	MW	NaN	...	
1978-08-13	5.80	MWC	NaN	...	
1989-10-03	5.50	MWC	NaN	...	
1995-09-20	5.50	MWC	NaN	...	
1994-01-17	5.80	MWC	NaN	...	
1983-05-02	6.30	MWC	NaN	...	
1990-02-28	5.70	MWC	NaN	...	

1987-07-31	6.00	MWC	NaN	...
1965-04-09	6.20	MW	NaN	...
1971-11-05	5.60	MW	NaN	...
1972-11-03	5.50	MW	NaN	...
1969-06-17	6.80	MW	NaN	...
1967-01-05	5.90	MW	NaN	...
1965-05-15	5.80	MW	NaN	...
1966-03-04	5.90	MW	NaN	...
2010-04-04	5.70	MW	NaN	...
1979-10-15	6.40	MW	0.288	...
1993-05-17	6.10	MW	NaN	...
1995-09-20	5.75	ML	NaN	...
1994-01-17	5.58	ML	NaN	...
2001-12-08	5.70	MW	NaN	...
1990-02-28	5.51	ML	NaN	...
2015-05-22	6.80	MWW	NaN	...
1983-09-27	5.50	MB	NaN	...
2005-10-02	5.80	MWB	NaN	...
1991-10-23	5.60	MW	NaN	...
2014-02-24	5.50	MWC	NaN	...
2015-08-06	6.00	MWW	NaN	...
2014-04-01	5.80	MB	0.175	...
2015-01-28	5.73	MW	NaN	...
1984-04-24	6.20	ML	NaN	...
1986-07-21	5.60	ML	NaN	...
1991-08-17	7.00	MH	NaN	...
1991-08-16	6.10	ML	NaN	...
1983-05-02	6.70	ML	NaN	...
1987-07-31	5.60	ML	NaN	...

	Magnitude	Source	Status	Date	depth_avg_22 \
Date(YYYY/MM/DD)					
1970-05-25		ISCGEM	Automatic	22:45:34	72.218182
1969-08-11		ISCGEM	Automatic	22:53:58	68.536364
1968-03-03		ISCGEM	Automatic	22:55:36	82.068182
1972-07-30		ISCGEM	Automatic	23:17:23	83.659091
1967-02-19		ISCGEM	Automatic	23:28:29	82.295455
1971-07-03		ISCGEM	Automatic	23:44:51	81.122727
1969-08-11		ISCGEM	Automatic	23:52:58	79.759091
1978-08-13		GCMT	Automatic	22:54:53	35.772727
1989-10-03		GCMT	Automatic	23:10:50	34.727273
1995-09-20		GCMT	Automatic	23:27:36	33.909091
1994-01-17		GCMT	Automatic	23:33:30	33.409091
1983-05-02		GCMT	Automatic	23:42:37	31.318182
1990-02-28		GCMT	Automatic	23:43:36	30.090909
1987-07-31		GCMT	Automatic	23:56:58	30.454545
1965-04-09		ISCGEM	Automatic	23:57:06	84.790909

1971-11-05	ISCGEM	Automatic	23:57:29	79.195455
1972-11-03	ISCGEM	Automatic	23:57:56	82.750000
1969-06-17	ISCGEM	Automatic	23:58:09	82.522727
1967-01-05	ISCGEM	Automatic	23:58:20	81.831818
1965-05-15	ISCGEM	Automatic	23:58:36	80.922727
1966-03-04	ISCGEM	Automatic	23:58:56	81.604545
2010-04-04	CI	Reviewed	22:50:17	5.729545
1979-10-15	CI	Reviewed	23:16:54	6.007682
1993-05-17	CI	Reviewed	23:20:50	5.986727
1995-09-20	CI	Reviewed	23:27:36	5.762773
1994-01-17	CI	Reviewed	23:33:31	5.809773
2001-12-08	CI	Reviewed	23:36:10	6.072545
1990-02-28	CI	Reviewed	23:43:37	5.949455
2015-05-22	US	Reviewed	23:59:34	77.822273
1983-09-27	US	Reviewed	23:59:38	83.531364
2005-10-02	US	Reviewed	23:59:44	84.508636
1991-10-23	HRV	Reviewed	23:59:44	57.890455
2014-02-24	GCMT	Reviewed	23:59:46	57.890455
2015-08-06	US	Reviewed	23:59:46	68.617727
2014-04-01	US	Reviewed	23:59:58	69.009091
2015-01-28	NC	Reviewed	21:08:54	8.463182
1984-04-24	NC	Reviewed	21:15:19	8.208545
1986-07-21	NC	Reviewed	22:07:16	7.819545
1991-08-17	NC	Reviewed	22:17:10	7.652773
1991-08-16	NC	Reviewed	22:26:14	7.550318
1983-05-02	NC	Reviewed	23:42:38	7.715000
1987-07-31	NC	Reviewed	23:56:58	8.173591

	depth_avg_15	depth_avg_7	mag_avg_22	mag_avg_15	mag_avg_7	\
Date(YYYY/MM/DD)						
1970-05-25	63.186667	58.928571	6.104545	6.186667	6.228571	
1969-08-11	63.186667	59.642857	6.063636	6.180000	5.971429	
1968-03-03	84.126667	111.457143	6.059091	6.173333	5.957143	
1972-07-30	86.460000	114.314286	6.063636	6.193333	5.942857	
1967-02-19	86.680000	116.457143	6.072727	6.166667	5.885714	
1971-07-03	90.960000	113.128571	6.050000	6.153333	5.771429	
1969-08-11	85.626667	115.985714	6.054545	6.053333	5.814286	
1978-08-13	40.800000	13.000000	5.959091	6.073333	5.828571	
1989-10-03	27.000000	13.000000	5.959091	6.000000	5.828571	
1995-09-20	26.600000	12.571429	5.959091	5.926667	5.742857	
1994-01-17	25.933333	11.142857	5.968182	5.946667	5.757143	
1983-05-02	26.066667	11.428571	6.000000	5.906667	5.742857	
1990-02-28	26.066667	11.428571	5.968182	5.866667	5.757143	
1987-07-31	24.933333	9.000000	5.986364	5.800000	5.800000	
1965-04-09	101.846667	33.871429	6.040909	6.000000	6.042857	
1971-11-05	103.180000	36.428571	6.036364	6.000000	5.957143	
1972-11-03	109.620000	48.085714	6.000000	5.960000	5.900000	

1969-06-17	109.953333	47.371429	5.968182	6.040000	5.985714
1967-01-05	108.453333	43.442857	5.972727	6.013333	6.028571
1965-05-15	107.513333	43.442857	5.968182	6.000000	5.971429
1966-03-04	67.593333	43.442857	5.977273	6.000000	5.957143
2010-04-04	5.803000	6.748429	5.955000	5.978667	6.132857
1979-10-15	6.560733	8.719857	5.977727	5.985333	6.255714
1993-05-17	6.720667	9.019714	5.966364	6.017333	6.292857
1995-09-20	6.630733	8.041714	5.976364	6.004667	6.188571
1994-01-17	6.836267	8.338286	5.966364	6.006667	6.171429
2001-12-08	7.208333	8.478429	5.961364	6.018667	6.061429
1990-02-28	7.027800	7.522000	5.925000	5.962667	5.820000
2015-05-22	64.166000	36.714286	6.004545	5.946667	5.900000
1983-09-27	72.539333	54.471429	5.981818	5.940000	5.900000
2005-10-02	44.979333	48.100000	5.972727	5.926667	5.942857
1991-10-23	44.846000	51.385714	5.977273	5.920000	5.885714
2014-02-24	44.140000	48.214286	5.945455	5.766667	5.828571
2015-08-06	61.406667	77.757143	5.909091	5.800000	5.828571
2014-04-01	60.847333	76.101429	5.859091	5.813333	5.857143
2015-01-28	8.979400	10.167714	5.971818	5.938667	5.830000
1984-04-24	8.971267	10.149571	5.971818	5.965333	5.858571
1986-07-21	8.423800	8.445000	5.967273	5.940667	5.787143
1991-08-17	7.853600	8.408429	5.967273	5.927333	5.972857
1991-08-16	7.483800	8.145714	5.985455	5.927333	6.030000
1983-05-02	7.984800	6.221429	6.035455	5.994000	6.147143
1987-07-31	8.410067	7.944571	6.012727	6.000667	6.132857

	mag_outcome
Date(YYYY/MM/DD)	
1970-05-25	NaN
1969-08-11	NaN
1968-03-03	NaN
1972-07-30	NaN
1967-02-19	NaN
1971-07-03	NaN
1969-08-11	NaN
1978-08-13	NaN
1989-10-03	NaN
1995-09-20	NaN
1994-01-17	NaN
1983-05-02	NaN
1990-02-28	NaN
1987-07-31	NaN
1965-04-09	NaN
1971-11-05	NaN
1972-11-03	NaN
1969-06-17	NaN
1967-01-05	NaN

```
1965-05-15          NaN
1966-03-04          NaN
2010-04-04          NaN
1979-10-15          NaN
1993-05-17          NaN
1995-09-20          NaN
1994-01-17          NaN
2001-12-08          NaN
1990-02-28          NaN
2015-05-22          NaN
1983-09-27          NaN
2005-10-02          NaN
1991-10-23          NaN
2014-02-24          NaN
2015-08-06          NaN
2014-04-01          NaN
2015-01-28          NaN
1984-04-24          NaN
1986-07-21          NaN
1991-08-17          NaN
1991-08-16          NaN
1983-05-02          NaN
1987-07-31          NaN

[42 rows x 28 columns]
```

[26] :

```
eq_all['mag_outcome'] = np.where(eq_all['mag_outcome'] > 2.5, 1,0)
print(eq_all['mag_outcome'].describe())
eq_all['mag_outcome'].value_counts()
```

```
count    23412.000000
mean         0.996967
std          0.054987
min          0.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          1.000000
Name: mag_outcome, dtype: float64
```

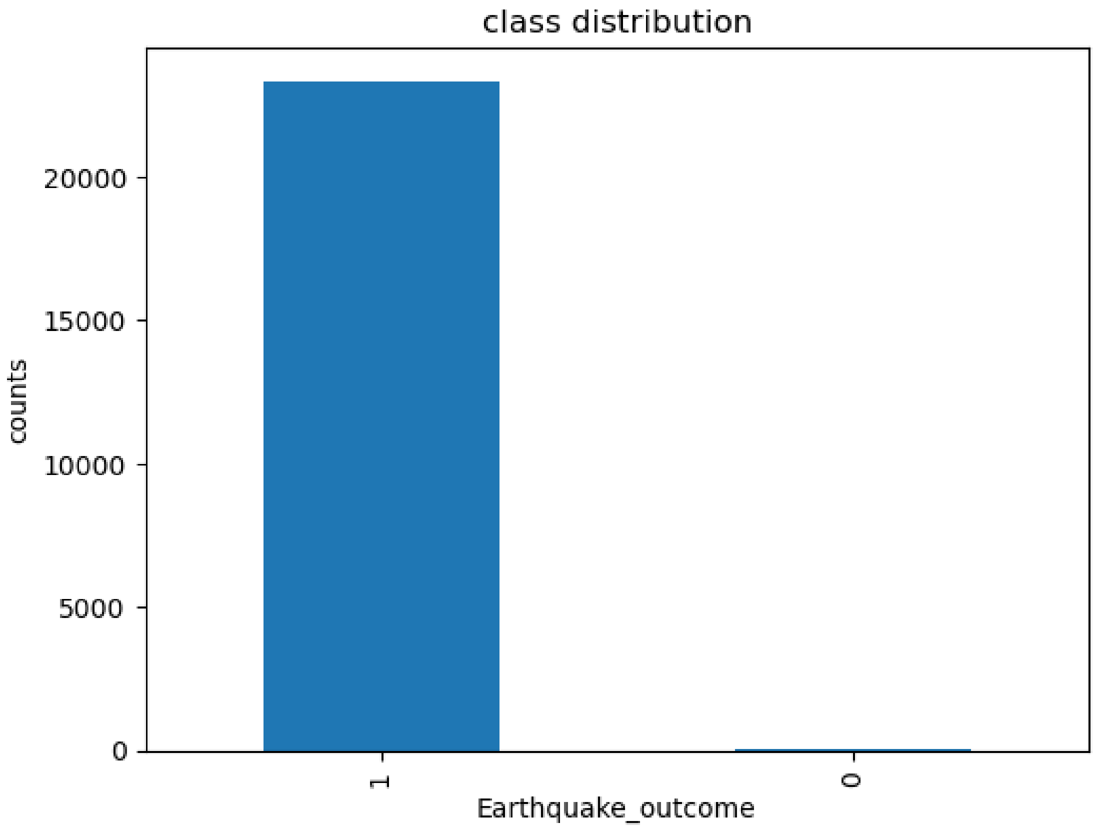
[26]:

```
1    23341
0       71
Name: mag_outcome, dtype: int64
```

[27] :

```
eq_all['mag_outcome'].value_counts().plot(kind='bar',)
plt.xlabel('Earthquake_outcome')
plt.ylabel('counts')
```

```
plt.title('class distribution');
```



Conclusion:

- In conclusion, Python serves as a powerful tool in the realm of earthquake prediction, enabling the analysis of seismic data and the development of predictive models.
- However, it is essential to acknowledge the formidable challenges inherent in earthquake prediction due to the intricate and dynamic nature of tectonic processes.
- The focus should thus be on enhancing earthquake preparedness and public safety, emphasizing resilient infrastructure, education, and emergency response plans.
- As we continue to collaborate with the global scientific community and advance our understanding of seismic activity, it is critical to approach earthquake prediction with a measured perspective, recognizing its current limitations and the probabilistic nature of the models developed using Python and other tools.
- Earthquake prediction using Python is a promising area of research, with several studies demonstrating the potential of machine learning algorithms to forecast seismic activity.
- Python's powerful data science and machine learning libraries make it a natural choice for this task, as they allow researchers to easily develop and train complex models on large datasets of earthquake data.
- While there is still much work to be done, early results suggest that Python-based earthquake prediction models can achieve high accuracy in forecasting the location, magnitude, and time of future earthquakes.
- This could have a significant impact on public safety, as it could allow for early warning systems to be developed that give people time to evacuate before a major earthquake strikes.
- Overall, Python is a powerful and flexible tool that can be used to develop accurate and reliable earthquake prediction models.
- As research in this area continues to progress, Python is likely to play an increasingly important role in earthquake prediction and disaster preparedness.