IBM NAAN MUDHALVAN

Technology name: Artificial Intelligence

Project title: Earthquake prediction model using python

Phase 2: Innovation

Dataset link: **https://www.kaggle.com/datasets/usgs/earthquake-database**

Notebook link:
https://colab.research.google.com/drive/1X09h3D1-JiF4Bkfu6bNfKIIkkArOjJtd?usp=sharing

Introduction:

An earthquake prediction model using Python is a data-driven approach aimed at forecasting or estimating the occurrence of earthquakes based on historical seismic data, geological information, and various machine learning techniques. This will provide an overview of the concept and its relevance, highlighting the fundamental components and challenges involved in creating such a model. Here we explore innovative techniques such as ensemble methods and deep learning architectures to improve the prediction system's accuracy and robustness.

Google Colab:

We have used Google Colab notebooks to perform certain tasks like data cleaning and analysis, to attain the accuracy in Earthquake prediction using Python.It's a Jupyter Notebook-based environment that offers several features and advantages. Google Colab comes pre-installed with many commonly used data science and machine learning libraries, including NumPy, Pandas, Matplotlib, TensorFlow, and PyTorch, making it convenient for data analysis and model development. You can easily connect your Google Colab environment with a GitHub repository to pull in or push out code, making version control and collaboration more efficient. You can use libraries like Matplotlib and Seaborn to create interactive visualizations within your notebooks. For a detailed walkthrough of the data cleaning and analysis process, refer to the Notebook on Google Colab, click here

Detailed explanation of the design:

Dataset:

Importing the dataset is the foundational step in our Earthquake Prediction using ML project. We seamlessly fetched seismic data from reliable sources, ensuring its accuracy and relevance. Leveraging the versatility of Python, we employed libraries like Pandas to efficiently read and organize the dataset for subsequent analysis. The chosen dataset encompasses essential seismic parameters, forming the basis for training and validating our machine learning model. The streamlined import process lays the groundwork for a comprehensive exploration into earthquake prediction methodologies.

Dataset location:

/content/drive/MyDrive/dataset/database.csv

File location: https://colab.research.google.com/drive/1X09h3D1-JiF4Bkfu6bNfKIIkkArOjJtd?usp=sharing

Program:

 # Importing the Libraries

```
import pandas as pd
import numpy as np
```

 # Loading the Dataset

```
data = pd.read_csv('database.csv')
data.head()
```

Output:

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	...	Magnitude Seismic Stations	Azimuthal Gap	Horizontal Distance	Horizontal Error	Root Mean Square	ID	Source	Location Source	Magnitude Source	Status
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM860706	ISCGEM	ISCGEM	ISCGEM	Automatic
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM860737	ISCGEM	ISCGEM	ISCGEM	Automatic
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM860762	ISCGEM	ISCGEM	ISCGEM	Automatic
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM860856	ISCGEM	ISCGEM	ISCGEM	Automatic
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM860890	ISCGEM	ISCGEM	ISCGEM	Automatic

5 rows × 21 columns

Data Analysis:

Data analysis in our Earthquake Prediction using ML project involves a meticulous exploration of seismic patterns and trends. Employing Python-based tools like NumPy and Pandas, we conducted descriptive statistics, revealing key insights into the dataset's characteristics. Visualization techniques, implemented with libraries such as Matplotlib and Seaborn, aided in uncovering spatial and temporal aspects of seismic activity. Correlation analysis provided a deeper understanding of feature relationships, guiding the model development process. The comprehensive data analysis phase contributes crucial inputs for building a robust machine learning model for earthquake prediction.

Program:

```
# Checking the Shape of the Dataset
data.shape

# Checking the Number of Entities
data.columns

# Checking Descriptive Structure of the data
data.describe()

# Checking Duplicated Rows.
data.duplicated()
```

```
# Checking the Data Information
data.info()
dataframe = pd.DataFrame(data)

# Checking Categorical and Numerical Columns
# Categorical columns
categorical_col = [col for col in dataframe.columns if
dataframe[col].dtype == 'object']
print('Categorical columns :',categorical_col)

# Numerical columns
numerical_col = [col for col in dataframe.columns if
dataframe[col].dtype != 'object']
print('Numerical columns :',numerical_col)

# Checking total number of Values in Categorical Columns
dataframe[categorical_col].nunique()

# Checking total number of Values in Numerical Columns
dataframe[numerical_col].nunique()

# Checking the Missing Values Percentage
round((dataframe.isnull().sum()/dataframe.shape[0])*100,2)
```

Output:

```
data.shape
```

```
(23412, 21)
```

```
data.columns
```

```
Index(['Date', 'Time', 'Latitude', 'Longitude', 'Type', 'Depth', 'Depth Error',
       'Depth Seismic Stations', 'Magnitude', 'Magnitude Type',
       'Magnitude Error', 'Magnitude Seismic Stations', 'Azimuthal Gap',
       'Horizontal Distance', 'Horizontal Error', 'Root Mean Square', 'ID',
       'Source', 'Location Source', 'Magnitude Source', 'Status'],
      dtype='object')
```

```
data.describe()
```

	Latitude	Longitude	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Error	Magnitude Seismic Stations	Azimuthal Gap	Horizontal Distance	Horizontal Error	Root Mean Square
count	23412.000000	23412.000000	23412.000000	4461.000000	7097.000000	23412.000000	327.000000	2564.000000	7299.000000	1604.000000	1156.000000	17352.000000
mean	1.679033	39.639961	70.767911	4.993115	275.364098	5.882531	0.071820	48.944618	44.163532	3.992660	7.662759	1.022784
std	30.113183	125.511959	122.651898	4.875184	162.141631	0.423066	0.051466	62.943106	32.141486	5.377262	10.430396	0.188545
min	-77.080000	-179.997000	-1.100000	0.000000	0.000000	5.500000	0.000000	0.000000	0.000000	0.004505	0.085000	0.000000
25%	-18.653000	-76.349750	14.522500	1.800000	146.000000	5.600000	0.046000	10.000000	24.100000	0.968750	5.300000	0.900000
50%	-3.568500	103.982000	33.000000	3.500000	255.000000	5.700000	0.059000	28.000000	36.000000	2.319500	6.700000	1.000000
75%	26.190750	145.026250	54.000000	6.300000	384.000000	6.000000	0.075500	66.000000	54.000000	4.724500	8.100000	1.130000
max	86.005000	179.998000	700.000000	91.295000	934.000000	9.100000	0.410000	821.000000	360.000000	37.874000	99.000000	3.440000

```
data.duplicated()
```

```
0        False
1        False
2        False
3        False
4        False
         ...  
23407    False
23408    False
23409    False
23410    False
23411    False
Length: 23412, dtype: bool
```

```
data.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  Root Mean Square            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
```

Feature Engineering:

Feature engineering is a critical aspect of machine learning where raw data is transformed or new features are created to enhance model performance. It involves techniques like polynomial expansion, interaction terms, and domain-specific transformations to extract meaningful information. Dimensionality reduction methods, such as PCA, help manage high-dimensional data, preventing overfitting and improving model efficiency. Handling categorical variables through encoding methods ensures effective utilization of non-numeric data. Feature engineering is an iterative process, guided by continuous evaluation and refinement to build models that accurately capture underlying patterns in the data.

Program:

```
# Creating Timestamp Column from Data and Time Column
import datetime
import time
timestamp = []
for d, t in zip(data['Date'], data['Time']):
try:
ts = datetime.datetime.strptime(d+' '+t, '%m/%d/%Y %H:%M:%S')
timestamp.append(time.mktime(ts.timetuple()))
except ValueError:
print('ValueError')
timestamp.append('ValueError')

# Converting the Tuple values into Series Values
```

```
timeStamp = pd.Series(timestamp)
data['Timestamp'] = timeStamp.values

# Droping the Date and Time Columns.
final_data = dataframe.drop(['Date', 'Time'], axis=1)
final_data = final_data[final_data.Timestamp != 'ValueError']
final_data.head()
```

Output:

	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	Magnitude Error	Magnitude Seismic Stations	Azimuthal Gap	Horizontal Distance	Horizontal Error	Root Mean Square	ID	Source	Location Source	Magnitude Source	Status	Timestamp
0	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	MW	NaN	NaN	NaN	NaN	NaN	NaN	ISCGEM860706	ISCGEM	ISCGEM	ISCGEM	Automatic	-157630542.0
1	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	NaN	NaN	NaN	ISCGEM860737	ISCGEM	ISCGEM	ISCGEM	Automatic	-157465811.0
2	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	MW	NaN	NaN	NaN	NaN	NaN	NaN	ISCGEM860762	ISCGEM	ISCGEM	ISCGEM	Automatic	-157355642.0
3	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	NaN	NaN	NaN	ISCGEM860856	ISCGEM	ISCGEM	ISCGEM	Automatic	-157093817.0
4	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	NaN	NaN	NaN	ISCGEM860890	ISCGEM	ISCGEM	ISCGEM	Automatic	-157026430.0

Data Cleaning:

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled

Program:

```
# Removal Of Unwanted Columns
dataframe1 = dataframe.drop(columns=['Depth Error','Depth
Seismic Stations', 'Magnitude Type',
'Magnitude Error', 'Magnitude Seismic Stations', 'Azimuthal Gap',
'Horizontal Distance', 'Horizontal Error', 'Root Mean Square', 'ID',
'Source', 'Location Source', 'Magnitude Source', 'Status', 'Date',
'Time'])

# Checking the Shape of Dataset after Removing the Columns
dataframe1.shape
dataframe1.head(10)

# Checking Columns
dataframe1.columns

# Checking the Missing Values Percentage
round((dataframe1.isnull().sum()/dataframe1.shape[0])*100,2)

# Checking the Data Information After droping the Unwanted
Columns
dataframe1.info()
```

```
# Checking the Descriptive Structure of the Data after the removal of Unwanted Columns
dataframe1.describe()

# Checking Categorical and Numerical Columns
# Categorical columns
categorical_col = [col for col in dataframe1.columns if
dataframe1[col].dtype == 'object']
print('Categorical columns :',categorical_col)

# Numerical columns
numerical_col = [col for col in dataframe1.columns if
dataframe1[col].dtype != 'object']
print('Numerical columns :',numerical_col)

# Checking total number of Values in Categorical Columns
dataframe1[categorical_col].nunique()

# Checking total number of Values in Numerical Columns
dataframe[numerical_col].nunique()

# Let's check the null values again
dataframe1.isnull().sum()
```

Output:

```
(23412, 6)
```

```
dataframe1.head(5)
```

	Latitude	Longitude	Type	Depth	Magnitude	Timestamp
0	19.246	145.616	Earthquake	131.6	6.0	-157630542.0
1	1.863	127.352	Earthquake	80.0	5.8	-157465811.0
2	-20.579	-173.972	Earthquake	20.0	6.2	-157355642.0
3	-59.076	-23.557	Earthquake	15.0	5.8	-157093817.0
4	11.938	126.427	Earthquake	15.0	5.8	-157026430.0

```
dataframe1.columns
```

```
Index(['Latitude', 'Longitude', 'Type', 'Depth', 'Magnitude', 'Timestamp'], dtype='object')
```

```
round((dataframe1.isnull().sum()/dataframe1.shape[0])*100,2)
```

```
Latitude     0.0
Longitude    0.0
Type         0.0
Depth        0.0
Magnitude    0.0
Timestamp    0.0
dtype: float64
```

```
Latitude     0.0
Longitude    0.0
Type         0.0
Depth        0.0
Magnitude    0.0
Timestamp    0.0
dtype: float64
```

```
dataframe1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Latitude   23412 non-null  float64
 1   Longitude  23412 non-null  float64
 2   Type       23412 non-null  object 
 3   Depth      23412 non-null  float64
 4   Magnitude  23412 non-null  float64
 5   Timestamp  23412 non-null  object 
dtypes: float64(4), object(2)
memory usage: 1.1+ MB
```

```
dataframe1.describe()
```

	Latitude	Longitude	Depth	Magnitude
count	23412.000000	23412.000000	23412.000000	23412.000000
mean	1.679033	39.639961	70.767911	5.882531
std	30.113183	125.511959	122.651898	0.423066
min	-77.080000	-179.997000	-1.100000	5.500000
25%	-18.653000	-76.349750	14.522500	5.600000
50%	-3.568500	103.982000	33.000000	5.700000
75%	26.190750	145.026250	54.000000	6.000000

```python
categorical_col = [col for col in dataframe1.columns if dataframe1[col].dtype == 'object']
print('Categorical columns :',categorical_col)
numerical_col = [col for col in dataframe1.columns if dataframe1[col].dtype != 'object']
print('Numerical columns :',numerical_col)
```

```
Categorical columns : ['Type', 'Timestamp']
Numerical columns : ['Latitude', 'Longitude', 'Depth', 'Magnitude']
```

```python
dataframe1[categorical_col].nunique()
```

```
Type             4
Timestamp    23391
dtype: int64
```

```python
dataframe1[numerical_col].nunique()
```

```
Latitude     20676
Longitude    21474
Depth         3485
Magnitude       64
dtype: int64
```

```python
dataframe1.isnull().sum()
```

```
Latitude     0
Longitude    0
Type         0
Depth        0
Magnitude    0
Timestamp    0
dtype: int64
```