Project title - Earthquake Prediction Model using Python

Phase 3 – Development Part-1

Colab Notebook link: Click here

Dataset Loading:

The earthquake dataset in the csv format is loaded as a dataframe.

```
#importing the required libraries
import numpy as np
import pandas as pd
```

```
[ ] #importing the dataset
df = pd.read_csv("/content/drive/MyDrive/database.csv")
data = df
df.head(5)
```

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	...
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	MW	...
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	MW	...
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	MW	...
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	MW	...
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	MW	...

5 rows × 21 columns

Data Preprocessing:

1. Checking for the total rows and columns:

```
[ ] #Finding the shape of the dataset
df.shape
```

```
(23412, 21)
```

2. Checking for duplicated values in the instances of the dataset:

```
[ ] #Checking for duplicated values in the rows of the dataset
    df.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
```

3. Statistical information about the dataset:

```
[ ] #Description about the dataset
    df.describe()
```

	Latitude	Longitude	Depth	Depth Error	Depth Seismic Stations
count	23412.000000	23412.000000	23412.000000	4461.000000	7097.000000
mean	1.679033	39.639961	70.767911	4.993115	275.364098
std	30.113183	125.511959	122.651898	4.875184	162.141631
min	-77.080000	-179.997000	-1.100000	0.000000	0.000000
25%	-18.653000	-76.349750	14.522500	1.800000	146.000000
50%	-3.568500	103.982000	33.000000	3.500000	255.000000
75%	26.190750	145.026250	54.000000	6.300000	384.000000
max	86.005000	179.998000	700.000000	91.295000	934.000000

4. Categorizing the columns based on their datatypes:

```
[ ] #Printing the numerical and categorical features
    # Categorical columns
    cat_col = [col for col in df.columns if df[col].dtype == 'object']
    print('Categorical columns :',cat_col)
    # Numerical columns
    num_col = [col for col in df.columns if df[col].dtype != 'object']
    print('Numerical columns :',num_col)

    Categorical columns : ['Date', 'Time', 'Type', 'Magnitude Type', 'ID',
    Numerical columns : ['Latitude', 'Longitude', 'Depth', 'Depth Error', '
```

5. Uniqueness check in categorical columns:

```
#Checking number of unique values in categorical columns
df[cat_col].nunique()
```

```
Date                12401
Time                20472
Type                    4
Magnitude Type         10
ID                  23412
Source                 13
Location Source        48
Magnitude Source       24
Status                  2
dtype: int64
```

6. Finding number of missing values in the columns:

```
[ ] #Finding number of missing values in each column
    print(df.isnull().sum())
```

```
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
Root Mean Square             6060
ID                              0
Source                          0
Location Source                 0
Magnitude Source                0
Status                          0
dtype: int64
```

7. Percentage of missing values:

```
[ ] #Finding the percentage of missing values in each column
    miss_percent = (df.isnull().sum()/df.shape[0])*100
    print(round(miss_percent,2))
```

```
Date                          0.00
Time                          0.00
Latitude                      0.00
Longitude                     0.00
Type                          0.00
Depth                         0.00
Depth Error                  80.95
Depth Seismic Stations       69.69
Magnitude                     0.00
Magnitude Type                0.01
Magnitude Error              98.60
Magnitude Seismic Stations   89.05
Azimuthal Gap                68.82
Horizontal Distance          93.15
Horizontal Error             95.06
Root Mean Square             25.88
ID                            0.00
Source                        0.00
Location Source               0.00
Magnitude Source              0.00
Status                        0.00
dtype: float64
```

8. Checking the column named “type” for unique values:

```
#Checking the number of instances in each class of the type attribute
df['Type'].value_counts()
```

```
Earthquake           23232
Nuclear Explosion      175
Explosion                4
Rock Burst               1
Name: Type, dtype: int64
```

9. Selecting the rows containing only the Earthquake type:

```
[ ] #Selecting the rows which contains the column name "type" with value "Earthquake"
    df.loc[df['Type']=="Earthquake"]
```

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations
0	01/02/1965	13:44:18	19.2460	145.6160	Earthquake	131.60	NaN	NaN
1	01/04/1965	11:29:49	1.8630	127.3520	Earthquake	80.00	NaN	NaN
2	01/05/1965	18:05:58	-20.5790	-173.9720	Earthquake	20.00	NaN	NaN
3	01/08/1965	18:49:43	-59.0760	-23.5570	Earthquake	15.00	NaN	NaN
4	01/09/1965	13:32:50	11.9380	126.4270	Earthquake	15.00	NaN	NaN

10. Dropping unnecessary columns and handling missing values:

```
[ ] #Dropping unnecessary columns and missing values from our dataset
    df.drop(['Type', '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'],axis=1,inplace=True)

[ ] #Printing the shape of the dataset after applying feature engineering
    print(df.shape)

    (23412, 6)

[ ] #Significant columns
    df.columns

    Index(['Date', 'Time', 'Latitude', 'Longitude', 'Depth', 'Magnitude'], dtype='object')
```

11. Checking for missing values after feature engineering:

```
[ ] #Checking for null values after feature engineering
    df.isnull().sum()

    Date         0
    Time         0
    Latitude     0
    Longitude    0
    Depth        0
    Magnitude    0
    dtype: int64
```

12. Creating a new column called ‘Timestamp’ from columns ‘Date’ and ‘Time’:

```
[ ] # We convert given Date and Time to Unix time which is in seconds and a
    import datetime
    import time

    timestamp = []
    for d, t in zip(df['Date'], df['Time']):
        try:
            ts = datetime.datetime.strptime(d+' '+t, '%m/%d/%Y %H:%M:%S')
            timestamp.append(time.mktime(ts.timetuple()))
        except ValueError:
            timestamp.append('ValueError')
    timeStamp = pd.Series(timestamp)
    df['Timestamp'] = timeStamp.values
```

❖ Dropping the columns date and time after creating the column Timestamp.

```
df.drop(['Date', 'Time'], axis=1,inplace=True)
df = df[df.Timestamp != 'ValueError']
print(df.head(5))
```

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

Conclusion:

Thus the loading and preprocessing of the given earthquake dataset is done successfully.