EARTHQUAKE PREDICTION MODEL USING PYTHON

AI_PHASE3

NAME: SUBASH RAJ V

REG.NO: 610821205052

DEVELOPMENT – PART 1

LOADING AND PREPROCESSING THE DATASET

1. Dataset Loading and printing dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# Dataset from https://www.kaggle.com/datasets/usgs/earthquake-database
data = pd.read_csv("earthquake-database/database.csv")
data
```

[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	
0	01/02/1965	13:44:18	19.2460	145.6160	Earthquake	131.60	NaN	NaN	6.0	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM8
1	01/04/1965	11:29:49	1.8630	127.3520	Earthquake	80.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM8
2	01/05/1965	18:05:58	-20.5790	-173.9720	Earthquake	20.00	NaN	NaN	6.2	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM8
3	01/08/1965	18:49:43	-59.0760	-23.5570	Earthquake	15.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM8
4	01/09/1965	13:32:50	11.9380	126.4270	Earthquake	15.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM8
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
23407	12/28/2016	08:22:12	38.3917	-118.8941	Earthquake	12.30	1.2	40.0	5.6	ML	...	18.0	42.47	0.120	NaN	0.1898	NN005
23408	12/28/2016	09:13:47	38.3777	-118.8957	Earthquake	8.80	2.0	33.0	5.5	ML	...	18.0	48.58	0.129	NaN	0.2187	NN005
23409	12/28/2016	12:38:51	36.9179	140.4262	Earthquake	10.00	1.8	NaN	5.9	MWW	...	NaN	91.00	0.992	4.8	1.5200	US100
23410	12/29/2016	22:30:19	-9.0283	118.6639	Earthquake	79.00	1.8	NaN	6.3	MWW	...	NaN	26.00	3.553	6.0	1.4300	US100
23411	12/30/2016	20:08:28	37.3973	141.4103	Earthquake	11.94	2.2	NaN	5.5	MB	...	428.0	97.00	0.681	4.5	0.9100	US100

23412 rows × 21 columns

2. DATASET PREPROCESSING

THE COLUMNS THAT ARE NEED FOR MODEL AND PREDICTION

 # THE COLUMNS THAT ARE NEED FOR MODEL AND PREDICTION

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	II
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM86070
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM86073
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM86076
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM86085
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN	NaN	ISCGEM86089

5 rows × 21 columns

Column Names

#Column Names

data.columns

[OUTPUT]

```
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')
```

The main features from earthquake data creating a object namely, Date, Time, Latitude, Longitude, Depth, Magnitude

data = data[['Date', 'Time', 'Latitude', 'Longitude', 'Depth', 'Magnitude']]

data.head()

[OUTPUT]

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

data.describe()

[OUTPUT]

	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
max	86.005000	179.998000	700.000000	9.100000

Here, the data is random we need to scale according to inputs to the model. So, we convert given Date and Time to Unix time which is in seconds and a numeral. This can be easily used as input for the network we built

```
import datetime

import time

# Create a list to store Unix timestamps

timestamp = []

# Iterate through the "Date" and "Time" columns
```

```
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:
        timestamp.append('ValueError')

# Create a Pandas Series from the timestamp list
timeStamp = pd.Series(timestamp)

# Add the "Timestamp" column to the DataFrame
data['Timestamp'] = timeStamp.values

# Drop the "Date" and "Time" columns
final_data = data.drop(['Date', 'Time'], axis=1

# Remove rows with 'ValueError' in the "Timestamp" column
final_data = final_data[final_data['Timestamp'] != 'ValueError']

# Display the first few rows of the final dataset
final_data.head()
```

[OUTPUT]

	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