EARTHQUAKE PREDICTION MODEL USING PYTHON

Introduction

Machine learning has the ability to advance our knowledge of earthquakes and enable more accurate forecasting and catastrophe response. It's crucial to remember that developing accurate and dependable prediction models for earthquakes still needs more study as it is a complicated and difficult topic.

In order to anticipate earthquakes, machine learning may be used to examine seismic data trends. Seismometers capture seismic data, which may be used to spot changes to the earth's surface, like seismic waves brought on by earthquakes. Machine learning algorithms may utilize these patterns to forecast the risk of an earthquake happening in a certain region by studying these patterns and learning to recognize key traits that are linked to seismic activity.So we will be predicting the earthquake fromDate and Time, Latitude, and Longitude from previous data is not a trend that follows like other things. It is naturally occurring.

Necessary steps to follow:

Predicting earthquakes is a highly complex and challenging task, and it's important to note that there's no reliable method for accurately predicting the time, location, and magnitude of earthquakes. However, you can create earthquake forecasting or early warning systems to provide alerts when seismic activity is detected. Here are the necessary steps for building a basic earthquake prediction model using Python:

1. Data Collection:

- Gather earthquake-related data from sources like the USGS (United States Geological Survey) or other relevant seismic organizations.

- Collect data on seismic events, geological features, and environmental conditions.

2. Data Preprocessing:

- Clean and preprocess the data, handling missing values and outliers.

- Convert location information into geographic coordinates (latitude and longitude).

- Engineer relevant features, such as seismic intensity, depth, and historical seismic activity in the region.

3. Data Visualization:

- Use libraries like Matplotlib or Seaborn to visualize the data and gain insights.

- Create plots and maps to understand the spatial and temporal distribution of earthquakes.

4. Feature Selection:

- Identify the most relevant features for earthquake prediction.

- Use techniques like feature correlation analysis to select the best predictors.

5. Model Selection:

- Choose an appropriate machine learning or statistical model. Common choices include:

  - Time-series analysis (e.g., ARIMA, LSTM)

  - Regression models (e.g., Linear Regression, Random Forest)

  - Clustering algorithms (e.g., K-Means for identifying earthquake clusters)

6. Model Training:

- Split the data into training and testing sets.

- Train the model on the training data.

7. Model Evaluation:

- Evaluate the model's performance using appropriate metrics (e.g., Mean Squared Error for regression models).

- Perform cross-validation to ensure model robustness.

8. Prediction and Alerting:

- Use the trained model to make predictions on real-time or historical seismic data.

- Set up alerts or notifications based on certain prediction thresholds.

9. Continuous Improvement:

- Continuously update and retrain the model as new data becomes available.

- Incorporate feedback and improve model performance over time.

10. Deployment:

- Deploy the model in a real-world environment or as part of an early warning system.

- Integrate the prediction system with communication channels to relay alerts.

11. Monitoring and Maintenance:

- Regularly monitor the system's performance and make necessary adjustments.

- Keep the system up to date with the latest data and research in earthquake prediction.

Remember that earthquake prediction is a challenging field, and even the most advanced models have limitations. Earthquake preparedness and response are crucial aspects of managing seismic risk, alongside any predictive efforts. Additionally, ethical considerations and public safety must be prioritized in any prediction system.

Importance of loading and preprocessing the dataset

1.Importing Libraries

Import numpy as np

Import pandas as pd

Import matplotlib.pyplot as plt

Import os

Print(os.listdir("../input"))

Output:

[‘database.csv’]

2.Read the Dataset

Now we will read the dataset and look for the various features in the dataset.

Data = pd.read_csv(“../input/database.csv”)

Data.head()

Output:

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

Data.columns

Output:

We need to select the features that will be useful for our prediction.

```
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 = 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

We will try to frame the time and place of the earthquake that has happened in the past on the world map.

```
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')


timeStamp = pd.Series(timestamp)
```

```
data['Timestamp'] = timeStamp.values
final_data = data.drop(['Date', 'Time'], axis=1)
final_data = final_data[final_data.Timestamp != 'ValueError']
final_data.head()
```

Output:

	Latitude	Longitude	Depth	Magnitude	Timestamp
0	19.246	145.616	131.6	6.0	-1.57631e+08
1	1.863	127.352	80.0	5.8	-1.57466e+08
2	-20.579	-173.972	20.0	6.2	-1.57356e+08
3	-59.076	-23.557	15.0	5.8	-1.57094e+08
4	11.938	126.427	15.0	5.8	-1.57026e+08

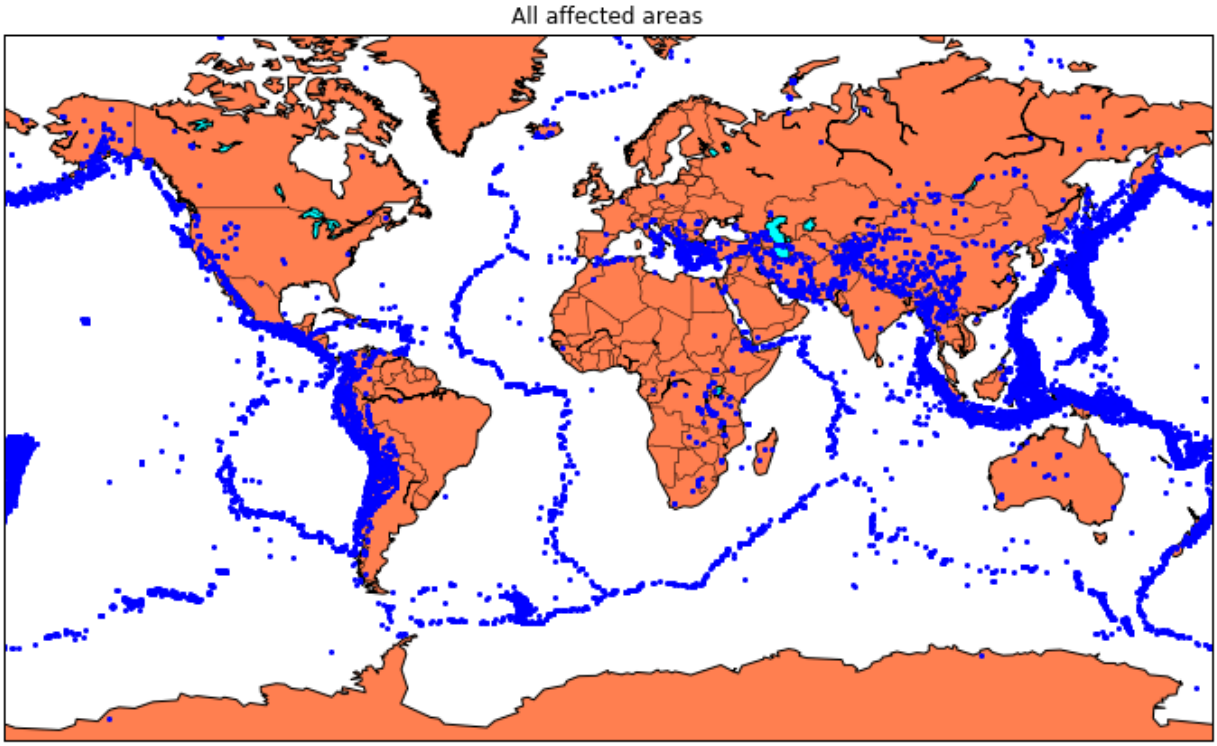
Visualization

Here, we will visualize the earthquakes that have occurred all around the world.

```
From mpl_toolkits.basemap import Basemap

M = Basemap(projection='mill',llcrnrlat=-80,urcrnrlat=80,
llcrnrlon=-180,urcrnrlon=180,lat_ts=20,resolution='c')
```

```
Longitudes = data[“Longitude”].tolist()
Latitudes = data[“Latitude”].tolist()
#m =
Basemap(width=12000000,height=9000000,projection=’l
cc’,

#resolution=None,lat_1=80.,lat_2=55,lat_0=80,lon_0=-
107.)
X,y = m(longitudes,latitudes)

Fig = plt.figure(figsize=(12,10))
Plt.title(“All affected areas”)
m.plot(x, y, “o”, markersize = 2, color = ‘blue’)
m.drawcoastlines()
m.fillcontinents(color=’coral’,lake_color=’aqua’)
m.drawmapboundary()
m.drawcountries()
plt.show()
```

Output:



Splitting The Dataset

Now we will split the dataset into a training and testing set.

X = final_data[[‘Timestamp’, ‘Latitude’, ‘Longitude’]]

Y = final_data[[‘Magnitude’, ‘Depth’]]

From sklearn.cross_validation import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

Print(X_train.shape, X_test.shape, y_train

Output:

Earthquake Prediction Using Machine Learning

We will be using the RandomForestRegressor model to predict the earthquake, here will look for its accuracy.

Reg = RandomForestRegressor(random_state=42)

Reg.fit(X_train, y_train)

Reg.predict(X_test)

Output:

```
array([[  5.96,  50.97],
       [  5.88,  37.8 ],
       [  5.97,  37.6 ],
       ...,
       [  6.42,  19.9 ],
       [  5.73, 591.55],
       [  5.68,  33.61]])
```

Conclusion

Understanding earthquakes and effectively responding to them remains a complex and challenging task, even with the latest technological advancements. However, leveraging the capabilities of machine learning can greatly enhance our comprehension of seismic events. By

employing machine learning techniques to analyze seismic data, we can uncover valuable insights and patterns that contribute to a deeper understanding of earthquakes. These insights can subsequently inform more effective strategies for mitigating risks and responding to seismic event