EARTHQUAKE PREDICTION MODEL USING PYTHON

BATCH MEMBER

TEAM-ID : Proj_272172_Team_2

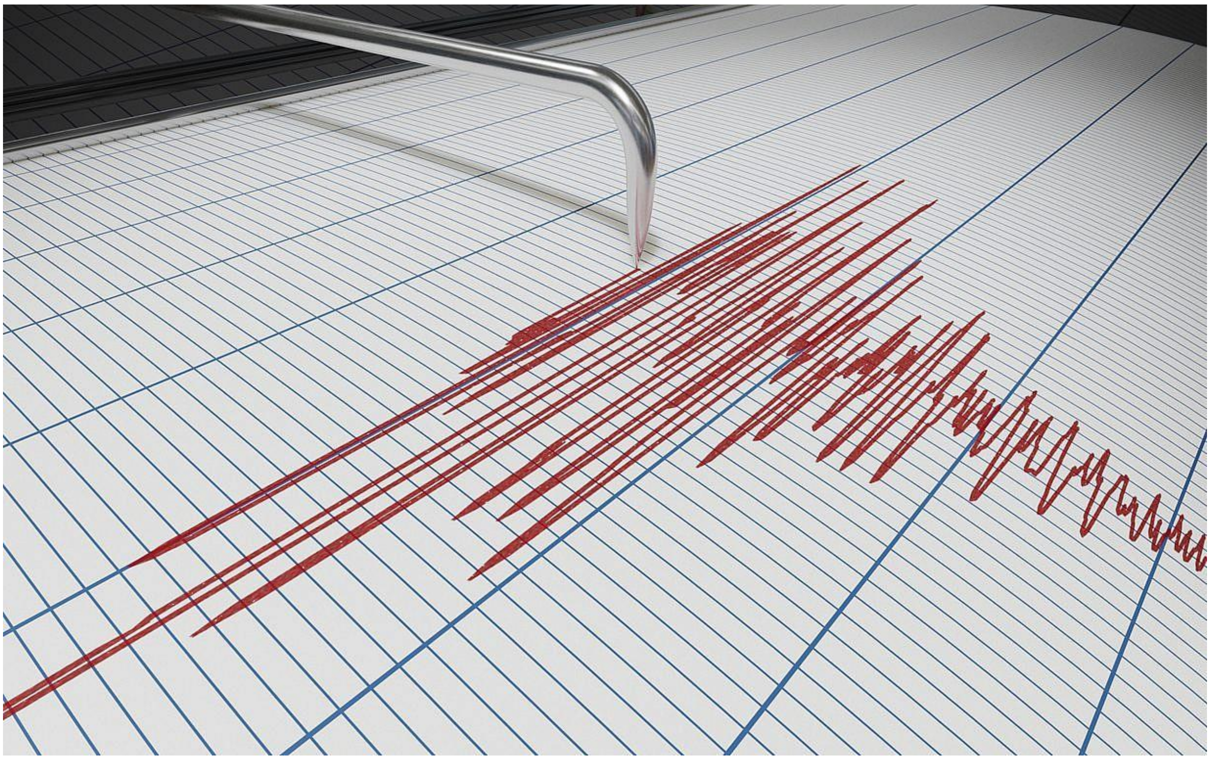
950621104097 : S.SOBIKA

Phase 4 Submission Document

Project Title: Earthquake Prediction Model Using Python.

Phase 3: Development Part 2.

Topic: Continue building the earthquake prediction model by feature engineering, model training and evaluation.



EARTHQUAKE PREDICTION

Introduction :

- The data scientist aiming to build a predictive model, the foundation of this endeavour lies in loading and preprocessing the dataset.
- In this section continue building the project by performing different activities like features engineering, model training, evaluation, etc.

GIVEN DATASET :

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	...	Magnitude Seismic Stations	Azimuthal Gap	Horizontal Distance	Horizontal Error
0	01/02/1965	13:44:18	19.2460	145.6160	Earthquake	131.60	NaN	NaN	6.0	MW	...	NaN	NaN	NaN	NaN
1	01/04/1965	11:29:49	1.8630	127.3520	Earthquake	80.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN
2	01/05/1965	18:05:58	-20.5790	-173.9720	Earthquake	20.00	NaN	NaN	6.2	MW	...	NaN	NaN	NaN	NaN
3	01/08/1965	18:49:43	-59.0760	-23.5570	Earthquake	15.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN
4	01/09/1965	13:32:50	11.9380	126.4270	Earthquake	15.00	NaN	NaN	5.8	MW	...	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
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
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
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
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
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

23412 rows × 21 columns

Overview of the process :

The following is an overview of the process of building a earthquake prediction model used by feature selection, model training, and evaluation.

1. **Prepare the data:**

   This includes cleaning the data, removing outliers, and handling missing values.

2. **Perform feature selection :**

   This can be done using a variety of methods, such as correlation analysis, information gain, and recursive features elimination.

3. **Train the model :**

   There are many different machine learning algorithms that can be used for house price prediction. Some popular choices include linear regression, random forests, SVR.

4. **Evaluate the model :**

   This can be done by calculating the mean squared error(MSE) or the root mean squared error (RMSE) of the model's predictions on the held-out test set.

5. **Deploy the model :**

   Once the model has been evaluating and found to be performing well, it can be deployed to production so that it can be used to predict the earthquake.

Features Selection :

Checking for missing values

In[1]:

```
print("Missing values")

print("-" *30)

print(df.isna().sum())

print("-"*30)

print("Total missing values", df.isna().sum())
```

Out[1]:

```
Missing values
------------------------------
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
------------------------------
Total missing values 145439
```

Model Training :

1. **Choose a machine learning algorithm :**
   There are a number of different machine learning algorithm that can be for earthquake prediction, such as linear regression, ridge regression, lasso regression, decision trees, and random forests are covered.

Machine Learning Models:

In[2]:

```
new_row = {"Model": "Ridge", "MAE":mae, "MSE": mse,"RMSE":rmse, "R2 Score": r_squared, "RMSE(Cross-Validation)":rmse_cross_val}

models = models.append(new_row, ignore_index=True)
```

In[3]:

```
def evaluation(y_true, y_pred):
    # calculate MAE
    mae = mean_absolute_error(y_true, y_pred)
    # calculate MSE
    mse = mean_squared_error(y_true, y_pred)
    # calculate RMSE
    rmse = np.sqrt(mse)
    rmse_cross_val = np.mean(rmse)
    # calculate R-squared score
    r_squared = r2_score(y_true, y_pred)
    # return the four metrics as a tuple
    return mae, mse, rmse, r_squared, rmse_cross_val
```

Linear Regression :

In[4]:

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

lin_reg = LinearRegression()

lin_reg.fit(X_train, y_train)

predictions = lin_reg.predict(X_test)

mae, mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("-" *30)

print("RMSE Cross-Validation:",rmse_cross_val)

Out[4]:

```
MAE: 16.214208564591
MSE: 413.6507308565237
RMSE: 20.33840531744128
R2 Score: -0.15997292842810484
------------------------------
RMSE Cross-Validation: 20.33840531744128
```

Ridge Regression :

In[5]:

ridge = Ridge()

ridge.fit(X_train, y_train)

```
predictions = ridge.predict(X_test)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("RMSE Cross-Validation:",rmse_cross_val)
```

Out[5]:

```
MAE: 16.221156759713875
MSE: 413.6262826215744
RMSE: 20.33780427237843
R2 Score: -0.15990436988686807
RMSE Cross-Validation: 20.33780427237843
```

<u>Lasso Regression:</u>

In[6]:

```
lasso = Lasso()

lasso.fit(X_train, y_train)

predictions = lasso.predict(X_test)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("-" *30)
```

print("RMSE Cross-Validation:",rmse_cross_val)

Out[6]:

```
MAE: 10.864336073458082
MSE: 195.0097579097878
RMSE: 13.96458942861507
R2 Score: 0.4531472494046366
------------------------------
RMSE Cross-Validation: 13.96458942861507
```

Elastic Net:

In[7]:

```
elasticnet = ElasticNet()

elasticnet.fit(X_train, y_train)

predictions = elasticnet.predict(X_test)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("-" *30)

print("RMSE Cross-Validation:",rmse_cross_val)
```

Out[7]:

```
MAE: 10.872423700794576
MSE: 195.23917220459506
RMSE: 13.972801158128425
R2 Score: 0.4525039183247668
------------------------------
RMSE Cross-Validation: 13.972801158128425
```

Support Vector Machines:

In[8]:

```
svr = SVR(C=100000)

svr.fit(X_train,y_train)

predictions = svr.predict(X_test)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("-" *30)

print("RMSE Cross-Validation:",rmse_cross_val)
```

Out[8]:

```
MAE: 60.364276908953464
MSE: 3877.4242177347583
RMSE: 62.26896673090664
R2 Score: -9.873199994813712
------------------------------
RMSE Cross-Validation: 62.26896673090664
```

Random Forest Regressor:

In[9]:

```
random_forest = RandomForestRegressor(n_estimators= 100)

random_forest.fit(X_train, y_train)

predictions = random_forest.predict(X_test)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)
```

```
print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)

print("R2 Score:",r_squared)

print("-" *30)

print("RMSE Cross-Validation:",rmse_cross_val)
```

Out[9]:

```
MAE: 10.295796132468222
MSE: 198.72930732017593
RMSE: 14.097138267044697
R2 Score: 0.44271676711570895
------------------------------
RMSE Cross-Validation: 14.097138267044697
```

Polynomial Regression (Degree= 2):

In[10]:

```
poly_reg = PolynomialFeatures(degree =2)

X_train_2d = poly_reg.fit_transform(X_train)

X_test_2d = poly_reg.transform(X_test)

lin_reg = LinearRegression()

lin_reg.fit(X_train_2d, y_train)

predictions = lin_reg.predict(X_test_2d)

mae,mse, rmse, r_squared,rmse_cross_val = evaluation(y_test, predictions)

print("MAE:",mae)

print("MSE:",mse)

print("RMSE:",rmse)
```

print("R2 Score:",r_squared)

print("-" *30)

print("RMSE Cross-Validation:",rmse_cross_val)

Out[10]:

```
MAE: 39.11674027433722
MSE: 1563.7117106065875
RMSE: 39.5437948432695
R2 Score: -3.3850115976195143
------------------------------
RMSE Cross-Validation: 39.5437948432695
```

Model Training :

- Model training is the process of teaching a machine learning model to predict earthquake.
- Once the model is trained, it can be used to predict earthquake for new data.

1. Prepare the data.
2. Split the data into training and test sets.
3. Choose a machine learning algorithm.
4. Tune the hyperparameters of the algorithm.
5. Train the model on the training set.
6. Evaluate the model on the test set.

Split the data into train and test :

In[11]:

X = df[['Latitude', 'Longitude', 'Magnitude','Magnitude Error', 'Magnitude Seismic Stations', 'Azimuthal Gap', 'Horizontal Distance', 'Horizontal Error', 'Root Mean Square', 'Depth Error']]

Y = df['Depth']

In[12]:

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

In[13]:

y_train.head()

Out[13]:

```
count    23412.000000
max        700.000000
std        122.651898
25%         14.522500
min         -1.100000
Name: Depth, dtype: float64
```

In[14]:

y_train.shape

Out[14]:

```
(18729,)
```

In[15]:

y_test.head()

Out[15]:

```
mean    70.767911
50%     33.000000
Name: Depth, dtype: float64
```

In[16]:

Y_test.shape

Out[16]:

```
(4683,)
```

Model Evaluation:

- It is the process of assessing the performance of a machine learning model on the unseen data.
- There are a number of different metrices that can be used to evaluate the performance of a earthquake prediction model.

Some of the most common metrics are:

- ✓ **Mean Squared Error(MSE):**
  This metric measures the average squared difference between the predicted and actual earthquake.
- ✓ **Root Mean Squared Error(RMSE):**
  This metric is the square root of the MSE.
- ✓ **Mean Absolute Error:**
  This metric measures the average absolute difference between the predicted and actual earthquake.
- ✓ **R-Squared:**
  This metric measures how well the model explains the variation in the actual earthquake happened.

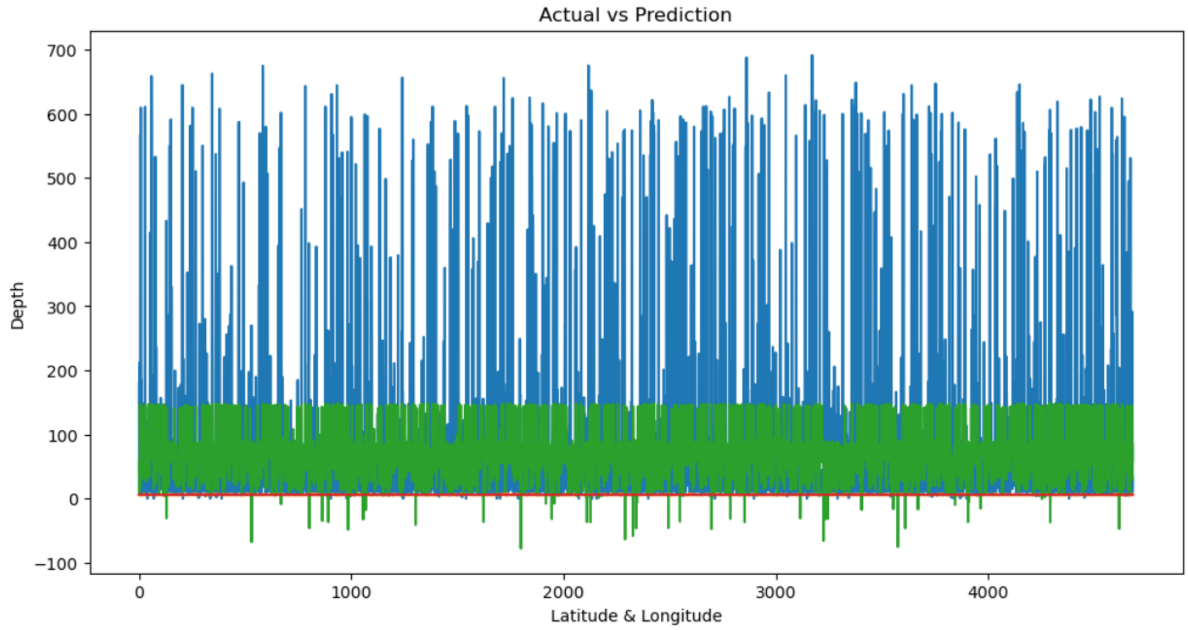
Evaluation of Predicted Data :

In[17]:

```
plt.figure(figsize=(12,6))

plt.plot(np.arange(len(y_test)), y_test)

plt.plot(np.arange(len(y_test)),predictions)

plt.xlabel("Latitude & Longitude")

plt.ylabel("Depth")

plt.title("Actual vs Prediction")
```

Out[17]:

```
Text(0.5, 1.0, 'Actual vs Prediction')
```

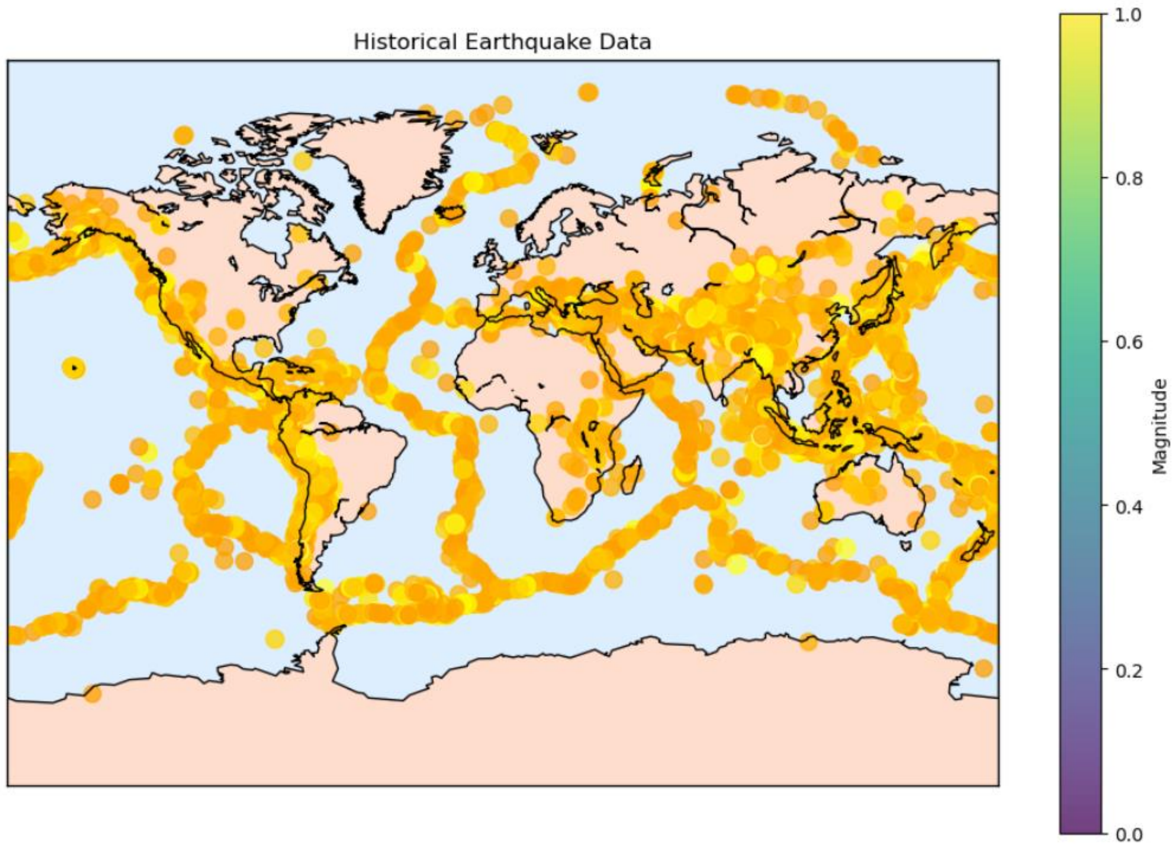


In[18]:

```
lons = df["Longitude"]

lats = df["Latitude"]

mags = df["Magnitude"]

depths = df["Depth"]

fig, ax = plt.subplots(figsize=(12,8))

m = Basemap(projection="mill", llcrnrlat=-90, urcrnrlat=90,

            llcrnrlon=-180, urcrnrlon=180, resolution="c")

m.drawcoastlines()

m.fillcontinents(color="#FFDDCC", lake_color="#DDEEFF")

m.drawmapboundary(fill_color="#DDEEFF")
```

```
x,y = m(lons, lats)

cmap = plt.get_cmap("hot")

colors = [cmap(i / max(mags)) for i in mags]

m.scatter(x, y, marker="o", c=colors, s=[i * 15 for i in mags], alpha=0.75)

plt.colorbar(label="Magnitude")

plt.title("Historical Earthquake Data")

plt.show()
```

Out[18]:



In[19]:

```
print(r2_score(y_test, predictions))

print(mean_absolute_error(y_test,predictions))
```

print(mean_squared_error(y_test,predictions))

Out[19]:

```
-0.15997292842810484
16.214208564591
413.6507308565237
```

Features Engineering :

It is a crucial aspect of predicting earthquake model using machine learning. It involves creating new features, transforming existing ones, and selecting the most relevant variables to improve the model's predictive power. Here are some feature engineering ideas for earthquake prediction.

1. **Auto-recognition of diurnal periodic waveform:**
   These are electromagnetic disturbances (ED) that synchronize with sunrise and sunset. They can be used to filter out the background noise and focus on the anomalous signals that may precede earthquakes.
2. **Higuchi Fractal Dimension:**
   This is a measure of the complexity or irregularity of a time series. It can be used to capture the non-linear features of ED data and quantify the degree of chaos or order in the system. A higher fractal dimension indicates a more chaotic system, which may imply a higher probability of earthquake occurrence.
3. **Sliding interquartile range:**
   This is a robust measure of variability or dispersion in a time series. It can be used to detect outliers or spikes in ED data that may indicate seismic precursors.
4. **Gutenberg-Richter Law:**
   This is a statistical law that relates the frequency and magnitude of earthquakes in a given region. It can be used to estimate the

probability of occurrence and the expected magnitude of future earthquakes based on historical seismic events.

5. **Geo- sound:**

   This is the sound generated by the movement of tectonic plates or faults. It can be measured by microphones or acoustic sensors and can provide information about the stress state and deformation of the crust.

<u>Various features of perform model training :</u>

1. **Seismic waveforms:**

   These are the signals recorded by seismometers that measure the ground motion caused by earthquakes. They can be used to extract features such as amplitude, frequency, duration, phase, and polarity of the waves, which can indicate the location, magnitude, and mechanism of the earthquake. Seismic waveforms can also be transformed into different domains, such as time-frequency, wavelet, or spectral, to capture more information.

2. **Earthquake catalog:**

   This is a collection of historical earthquake data that includes parameters such as date, time, latitude, longitude, depth, magnitude, and fault type of each event. Earthquake catalog can be used to analyze the spatial and temporal patterns of seismic activity, such as clustering, recurrence intervals, and aftershock sequences. Earthquake catalog can also be used to estimate the probability and expected magnitude of future earthquakes based on statistical models, such as the Gutenberg-Richter law or the Poisson distribution.

3. **Geological features:**

   These are the characteristics of the earth's crust and mantle that affect the generation and propagation of seismic waves. Geological features include parameters such as rock type, density, porosity, permeability, elasticity, viscosity, and stress state. Geological features can be derived from various sources, such as borehole logs, geophysical surveys, or satellite imagery. Geological features can be used to model the structure and dynamics of the earth's interior and to simulate the ground motion at specific locations or regions.

4. **Environmental factors:**

   These are the external factors that may have an impact on earthquake occurrence or detection. Environmental factors include parameters such as temperature, pressure, humidity, precipitation, wind speed, solar radiation, and geomagnetic field. Environmental factors can be measured by various sensors or instruments, such as thermometers, barometers, hygrometers, rain gauges, anemometers, pyranometers, and magnetometers. Environmental factors can be used to identify potential precursors or anomalies that may indicate seismic activity or to filter out noise or interference in seismic data.

Conclusion :

- Earthquake prediction is a challenging and important task that aims to forecast the occurrence, location, magnitude, and impact of future earthquakes based on various types of data and models.
- Earthquake prediction can help reduce the loss of life and property, improve the preparedness and resilience of communities, and advance the scientific understanding of the earth's processes.
- However, earthquake prediction is also subject to many uncertainties, limitations, and ethical issues that need to be addressed.
- Earthquake data is often noisy, incomplete, inconsistent, or unreliable. For example, seismic waveforms may be affected by environmental factors, instrument errors, or human interference.

- Earthquake catalog may be biased, incomplete, or inaccurate due to different reporting standards, detection thresholds, or measurement methods.
- Geological features may be difficult to measure or estimate due to the complexity and variability of the earth's structure and dynamics. Environmental factors may be irrelevant, redundant, or misleading as potential precursors or indicators of seismic activity.
- Earthquake models are often based on simplifying assumptions, approximations, or empirical rules that may not capture the true physics or statistics of the earthquake phenomenon.
- Earthquake prediction is inherently probabilistic and uncertain due to the randomness and complexity of the earthquake process.
- Earthquake prediction is not a perfect science but a continuous learning process that requires collaboration, innovation, and evaluation.
- By improving the data quality and availability, developing more realistic and robust models, enhancing the prediction accuracy and uncertainty quantification, and considering the ethical and social implications, earthquake prediction can become more feasible and beneficial for society.