EARTHQUAKE PREDICTION MODEL USING PYTHON

Phase 2 Submission Document:

Project: Earthquake Prediction

INTRODUCTION: ✓

✓ Earthquake Prediction is a way of predicting the magnitude of an earthquake based on parameters such as longitude, latitude, depth, and duration magnitude, country, and depth using machine learning to give warnings of potentially damaging earthquakes early enough to allow appropriate response to the disaster, enabling people to minimize loss of life and property

- ✓ Earthquake prediction is a complex and challenging task that involves anticipating the occurrence, magnitude, and location of seismic events before they occur

.

- ✓ Earthquakes, caused by the sudden release of energy in the Earth's crust, can have devastating consequences, making prediction efforts crucial for risk mitigation and disaster preparedness.

- ✓ Earthquake prediction aims to provide advanced warning systems and insights into seismic activities, enabling communities to take precautionary measures and potentially save lives. Traditional methods involve analyzing historical seismic data and identifying patterns that may precede significant events.

- ✓ However, predicting earthquakes with high precision remains an ongoing scientific challenge.

Key Aspects of Earthquake Prediction:

1.Seismic Data Analysis:

1. Utilizing datasets that include information on earthquake occurrences, such as time, date, location (latitude and longitude), depth, and magnitude.

2. Exploring temporal and spatial patterns within the data to identify potential precursors to seismic events.

2.Feature Engineering:

1. Extracting meaningful features from the data, such as temporal trends, spatial relationships, and interactions between different parameters.

2. Incorporating domain-specific knowledge and external factors that may influence seismic activities.

3.Machine Learning Models:

1. Developing predictive models, often based on machine learning algorithms, to analyze and learn patterns from historical earthquake data.

2. Common models include neural networks, support vector machines, and decision trees.

4.Hyperparameter Tuning:

1. Fine-tuning model parameters to optimize predictive performance through techniques like grid search or Bayesian optimization.

5.Validation and Evaluation:

1. Splitting the dataset into training and testing sets to validate the model's performance.

2. Evaluating the model's accuracy, precision, recall, and other relevant metrics.

6.Challenges in Earthquake Prediction:

1. Earthquakes are inherently unpredictable due to the dynamic and complex nature of tectonic processes.

2. Limited historical data for rare, large-magnitude earthquakes makes it challenging to train accurate models.

7.Innovation and Future Directions:

1. Advancing techniques such as deep learning, ensemble methods, and integrating real-time sensor data for improved prediction capabilities.

2. Collaborative efforts between scientists, researchers, and data scientists to explore innovative solutions.

Content For Project PHASE 2:

Preprocessing of Dataset:

Data preprocessing is a crucial step in earthquake prediction, ensuring that the dataset is clean, structured, and suitable for training machine learning models. Here's a general guide for preprocessing a dataset for earthquake prediction:

DATASET Link:

https://www.kaggle.com/datasets/usgs/earthquakedatabase

date	depth	mag	place	latitude	longitude	depth_avg_22	depth_avg_15	depth_avg_7	mag_avg_22	mag_avg_15	ma
2020-07-14	6.70	1.58	Oklahoma	36.171483	-97.718347	6.717727	6.560000	7.100000	1.352273	1.271333	[illegible]
2020-07-14	7.55	2.07	Oklahoma	36.171483	-97.718347	6.730000	6.682667	7.132857	1.372727	1.334667	[illegible]
2020-07-14	7.39	1.89	Oklahoma	36.171483	-97.718347	6.747727	6.708667	6.940000	1.396818	1.377333	[illegible]
2020-07-15	7.75	1.48	Oklahoma	36.171483	-97.718347	6.834545	6.764000	6.848571	1.383182	1.388667	[illegible]
2020-07-15	7.81	1.50	Oklahoma	36.171483	-97.718347	6.841364	6.854667	6.964286	1.404545	1.385333	[illegible]

[1]:
```
import numpy as np
import pandas as pd
import os
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVR
from catboost import CatBoostRegressor, Pool
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

[2]:
```
train=pd.read_csv("H:\database.csv", nrows=6000000,dtype={'Magnitude':np.
↪float16,'Depth':np.float64})
train.head(10)
```

[2]:
```
         Date      Time  Latitude  Longitude        Type  Depth  Depth Error  \
0  01/02/1965  13:44:18    19.246    145.616  Earthquake  131.6          NaN
1  01/04/1965  11:29:49     1.863    127.352  Earthquake   80.0          NaN
2  01/05/1965  18:05:58   -20.579   -173.972  Earthquake   20.0          NaN
3  01/08/1965  18:49:43   -59.076    -23.557  Earthquake   15.0          NaN
4  01/09/1965  13:32:50    11.938    126.427  Earthquake   15.0          NaN
5  01/10/1965  13:36:32   -13.405    166.629  Earthquake   35.0          NaN
6  01/12/1965  13:32:25    27.357     87.867  Earthquake   20.0          NaN
7  01/15/1965  23:17:42   -13.309    166.212  Earthquake   35.0          NaN
8  01/16/1965  11:32:37   -56.452    -27.043  Earthquake   95.0          NaN
9  01/17/1965  10:43:17   -24.563    178.487  Earthquake  565.0          NaN

   Depth Seismic Stations  Magnitude Magnitude Type  ...  \
0                     NaN   6.000000             MW  ...
1                     NaN   5.800781             MW  ...
2                     NaN   6.199219             MW  ...
3                     NaN   5.800781             MW  ...
4                     NaN   5.800781             MW  ...
5                     NaN   6.699219             MW  ...
6                     NaN   5.898438             MW  ...
7                     NaN   6.000000             MW  ...
8                     NaN   6.000000             MW  ...
9                     NaN   5.800781             MW  ...
```

```
   Magnitude Seismic Stations  Azimuthal Gap  Horizontal Distance  \
0                         NaN            NaN                  NaN
1                         NaN            NaN                  NaN
2                         NaN            NaN                  NaN
3                         NaN            NaN                  NaN
4                         NaN            NaN                  NaN
5                         NaN            NaN                  NaN
6                         NaN            NaN                  NaN
7                         NaN            NaN                  NaN
8                         NaN            NaN                  NaN
9                         NaN            NaN                  NaN

   Horizontal Error  Root Mean Square               ID    Source  \
0               NaN               NaN     ISCGEM860706    ISCGEM
1               NaN               NaN     ISCGEM860737    ISCGEM
2               NaN               NaN     ISCGEM860762    ISCGEM
3               NaN               NaN     ISCGEM860856    ISCGEM
4               NaN               NaN     ISCGEM860890    ISCGEM
5               NaN               NaN     ISCGEM860922    ISCGEM
6               NaN               NaN     ISCGEM861007    ISCGEM
7               NaN               NaN     ISCGEM861111    ISCGEM
8               NaN               NaN  ISCGEMSUP861125 ISCGEMSUP
9               NaN               NaN     ISCGEM861148    ISCGEM

  Location Source Magnitude Source     Status
0          ISCGEM           ISCGEM  Automatic
1          ISCGEM           ISCGEM  Automatic
2          ISCGEM           ISCGEM  Automatic
3          ISCGEM           ISCGEM  Automatic
4          ISCGEM           ISCGEM  Automatic
5          ISCGEM           ISCGEM  Automatic
6          ISCGEM           ISCGEM  Automatic
7          ISCGEM           ISCGEM  Automatic
8          ISCGEM           ISCGEM  Automatic
9          ISCGEM           ISCGEM  Automatic

[10 rows x 21 columns]
```

[3]:

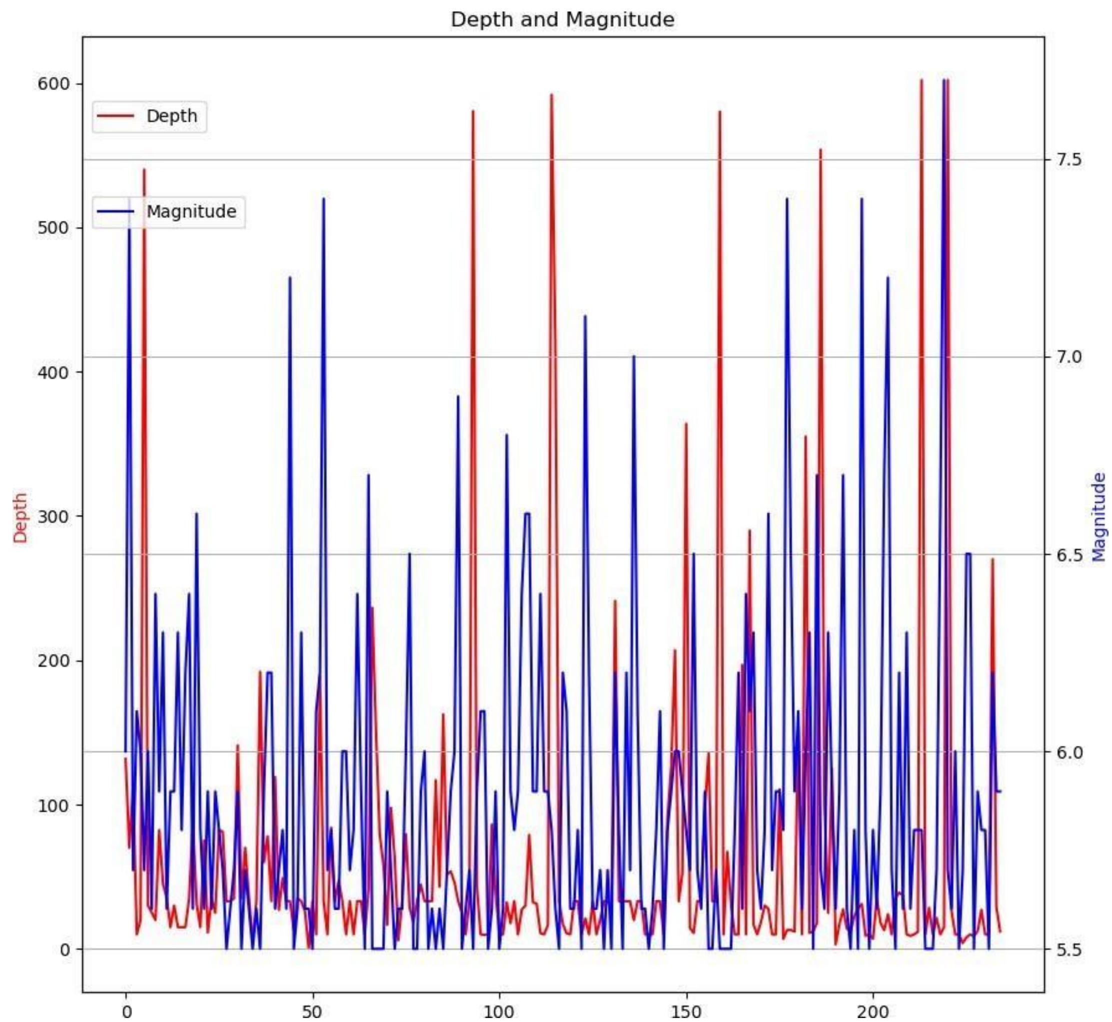
```
train_acoustic_df = train['Depth'].values[::100]
train_time_to_failure_df = train['Magnitude'].values[::100]

fig, ax1 = plt.subplots(figsize=(10,10))
plt.title('Depth and Magnitude')
plt.plot(train_acoustic_df, color='r')

ax1.set_ylabel('Depth', color='r')
plt.legend(['Depth'], loc=(0.01, 0.9))
```

```
ax2 = ax1.twinx()
plt.plot(train_time_to_failure_df, color="b")
ax2.set_ylabel("Magnitude", color="b")
plt.legend(["Magnitude"], loc=(0.01, 0.8))

plt.grid(True)
```




```
[4]: def gen_features(X):
        fe = []
        fe.append(X.mean())
        fe.append(X.std())
        fe.append(X.min())
        fe.append(X.max())
        fe.append(X.kurtosis())
        fe.append(X.skew())
```

```
    fe.append(np.quantile(X,0.01))
    fe.append(np.quantile(X,0.05))
    fe.append(np.quantile(X,0.95))
    fe.append(np.quantile(X,0.99))
    fe.append(np.abs(X).max())
    fe.append(np.abs(X).mean())
    fe.append(np.abs(X).std())
    return pd.Series(fe)
```

[5]:
```
train = pd.read_csv("H:\database.csv", iterator=True, chunksize=150_000,␣
 ↪dtype={"Depth": np.float64, "Magnitude": np.float64})

X_train = pd.DataFrame()
y_train =  pd.Series()
for df in train:
    ch = gen_features(df["Depth"])
    X_train = X_train.append(ch, ignore_index=True)
    y_train = y_train.append(pd.Series(df["Magnitude"].values[-1]))
```

```
C:\Users\Ragu\AppData\Local\Temp\ipykernel_8256\66685793.py:4: FutureWarning:
The default dtype for empty Series will be 'object' instead of 'float64' in a
future version. Specify a dtype explicitly to silence this warning.
  y_train = pd.Series()
C:\Users\Ragu\AppData\Local\Temp\ipykernel_8256\66685793.py:7: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  X_train = X_train.append(ch, ignore_index=True)
C:\Users\Ragu\AppData\Local\Temp\ipykernel_8256\66685793.py:8: FutureWarning:
The series.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  y_train = y_train.append(pd.Series(df['Magnitude'].values[-1]))
```

[6]:
```
X_train.head(10)
```

[6]:

	0	1	2	3	4	5	6	7	\
0	70.767911	122.651898	-1.1	700.0	10.456851	3.290683	3.13964	10.0	

	8	9	10	11	12
0	386.835	606.8513	700.0	70.76802	122.651835

[7]:
```
submission = pd.read_csv("database.csv", index_col="ID")
```

[8]:
```
scaler = StandardScaler()
scaler.fit(X_train)

X_train_scaled = pd.DataFrame(scaler.transform(X_train))
X_train_scaled.head(10)
```

[8]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

[9]:
```
parameters = [{"gamma": [0.001, 0.005, 0.01, 0.02, 0.05, 0.1],
              "C": [0.1, 0.2, 0.25, 0.5, 1, 1.5, 2]}]

reg1 = GridSearchCV(SVR(kernel="rbf", tol=0.01), parameters, cv=5,␣
↪scoring="neg_mean_absolute_error")
```

[10]:
```
test = pd.read_csv("database.csv", iterator=True, chunksize=150_000,␣
↪dtype={"Depth": np.float64, "Magnitude": np.float64})
X_test = pd.DataFrame()
for df in test:
    ch = gen_features(df["Depth"])
    X_test = X_test.append(ch, ignore_index=True)
```

```
C:\Users\Ragu\AppData\Local\Temp\ipykernel_8256\4001832223.py:5: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  X_test = X_test.append(ch, ignore_index=True)
```

[11]:
```
X_test.head(10)
```

[11]:

	0	1	2	3	4	5	6	7	\
0	70.767911	122.651898	-1.1	700.0	10.456851	3.290683	3.13964	10.0	

	8	9	10	11	12
0	386.835	606.8513	700.0	70.76802	122.651835

[12]:
```
X_test_scaled = pd.DataFrame(scaler.transform(X_test))
X_test_scaled.head(10)
```

[12]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

[13]:
```
submission
```

[13]:

	Date	Time	Latitude	Longitude	Type	Depth	\
ID							
ISCGEM860706	01/02/1965	13:44:18	19.2460	145.6160	Earthquake	131.60	
ISCGEM860737	01/04/1965	11:29:49	1.8630	127.3520	Earthquake	80.00	
ISCGEM860762	01/05/1965	18:05:58	-20.5790	-173.9720	Earthquake	20.00	
ISCGEM860856	01/08/1965	18:49:43	-59.0760	-23.5570	Earthquake	15.00	
ISCGEM860890	01/09/1965	13:32:50	11.9380	126.4270	Earthquake	15.00	
...	...	...	...	...	...	...	
NN00570710	12/28/2016	08:22:12	38.3917	-118.8941	Earthquake	12.30	
NN00570744	12/28/2016	09:13:47	38.3777	-118.8957	Earthquake	8.80	

US10007NAF	12/28/2016	12:38:51	36.9179	140.4262	Earthquake	10.00
US10007NL0	12/29/2016	22:30:19	-9.0283	118.6639	Earthquake	79.00
US10007NTD	12/30/2016	20:08:28	37.3973	141.4103	Earthquake	11.94

ID	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type \
ISCGEM860706	NaN	NaN	6.0	MW
ISCGEM860737	NaN	NaN	5.8	MW
ISCGEM860762	NaN	NaN	6.2	MW
ISCGEM860856	NaN	NaN	5.8	MW
ISCGEM860890	NaN	NaN	5.8	MW
...	...	...	...	...
NN00570710	1.2	40.0	5.6	ML
NN00570744	2.0	33.0	5.5	ML
US10007NAF	1.8	NaN	5.9	MWW
US10007NL0	1.8	NaN	6.3	MWW
US10007NTD	2.2	NaN	5.5	MB

ID	Magnitude Error	Magnitude Seismic Stations	Azimuthal Gap \
ISCGEM860706	NaN	NaN	NaN
ISCGEM860737	NaN	NaN	NaN
ISCGEM860762	NaN	NaN	NaN
ISCGEM860856	NaN	NaN	NaN
ISCGEM860890	NaN	NaN	NaN
...	...	...	...
NN00570710	0.320	18.0	42.47
NN00570744	0.260	18.0	48.58
US10007NAF	NaN	NaN	91.00
US10007NL0	NaN	NaN	26.00
US10007NTD	0.029	428.0	97.00

ID	Horizontal Distance	Horizontal Error	Root Mean Square	Source \
ISCGEM860706	NaN	NaN	NaN	ISCGEM
ISCGEM860737	NaN	NaN	NaN	ISCGEM
ISCGEM860762	NaN	NaN	NaN	ISCGEM
ISCGEM860856	NaN	NaN	NaN	ISCGEM
ISCGEM860890	NaN	NaN	NaN	ISCGEM
...	...	...	...	...
NN00570710	0.120	NaN	0.1898	NN
NN00570744	0.129	NaN	0.2187	NN
US10007NAF	0.992	4.8	1.5200	US
US10007NL0	3.553	6.0	1.4300	US
US10007NTD	0.681	4.5	0.9100	US

Location Source Magnitude Source Status

```
ID
ISCGEM860706          ISCGEM          ISCGEM   Automatic
ISCGEM860737          ISCGEM          ISCGEM   Automatic
ISCGEM860762          ISCGEM          ISCGEM   Automatic
ISCGEM860856          ISCGEM          ISCGEM   Automatic
ISCGEM860890          ISCGEM          ISCGEM   Automatic
...                      ...             ...      ...
NN00570710                NN              NN    Reviewed
NN00570744                NN              NN    Reviewed
US10007NAF                US              US    Reviewed
US10007NL0                US              US    Reviewed
US10007NTD                US              US    Reviewed

[23412 rows x 20 columns]
```

[14]:
```
submission.to_csv("submission.csv",index=True)
```

CONCLUSION:

the development of an earthquake prediction machine learning model using Python involves a systematic and multidimensional approach. Here's an elaborative summary of key aspects: In summary, the development of an earthquake prediction model using Python is a multifaceted process that requires a combination of domain knowledge, data science expertise, and continuous improvement. Through careful preprocessing, model development, and collaboration, Python serves as a versatile tool for addressing the complexities of earthquake prediction and contributing to advancements in the field.