EARTHQUAKE PREDICTION MODEL USING PYTHON

TEAM MEMBER NAME: DHANU VARSHA R

REG.NO: 810021106022

PHASE III: DEVELOPMENT PART 1

INTRODUCTION:

To build an earthquake prediction model using Python, we first need to load and preprocess the dataset. This involves loading the data, exploring it to understand its characteristics, and cleaning and transforming it to make it suitable for machine learning.

Once the data has been preprocessed, we can split it into training and test sets. The training set will be used to train the machine learning model, and the test set will be used to evaluate the performance of the model.

Here is a summary of the steps involved in loading and preprocessing the dataset for an earthquake prediction model using Python:

1. Load the dataset using the Python library “pandas”.
2. Explore the dataset to understand its characteristics.
3. Preprocess the data by handling missing values, converting categorical variables to numerical variables, and scaling the data.
4. Split the data into training and test sets.

Once the data has been loaded and preprocessed, we can train and evaluate the machine learning model.

LOADING:

To load the dataset for the earthquake prediction model project, we can use the following steps:

1. Import the pandas library.

2. Load the dataset using the pandas.read_csv() function.
3. Explore the dataset to understand its characteristics and identify any preprocessing steps that need to be performed.
4. Preprocess the dataset as needed, such as handling missing values, converting categorical variables to numerical variables, and scaling the data.
5. Split the dataset into training and test sets.

PROGRAM FOR LOADING:

IN[1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

import tensorflow as tf
```

IN[2]:

```
data = pd.read_csv('../input/earthquake-database/database.csv')
```

IN[3]:

```
Data
```

OUT[1]: OUTPUT OF THE LOADED DATASET IS SHOWN BELOW AND LINK OF THE DATASET IS:

https://www.kaggle.com/datasets/usgs/earthquake-database/data

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	Magnitud	Magnitud	Azimuthal	Horizonta	Horizonta	Root Mea	ID	Source	Location S	Magnitud	Status	
2	1/2/1965	13:44:18	19.246	145.616	Earthquak	131.6			6	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
3	1/4/1965	11:29:49	1.863	127.352	Earthquak	80			5.8	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
4	1/5/1965	18:05:58	-20.579	-173.972	Earthquak	20			6.2	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
5	1/8/1965	18:49:43	-59.076	-23.557	Earthquak	15			5.8	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
6	1/9/1965	13:32:50	11.938	126.427	Earthquak	15			5.8	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
7	########	13:36:32	-13.405	166.629	Earthquak	35			6.7	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
8	########	13:32:25	27.357	87.867	Earthquak	20			5.9	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
9	########	23:17:42	-13.309	166.212	Earthquak	35			6	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
10	########	11:32:37	-56.452	-27.043	Earthquak	95			6	MW							ISCGEMSU	ISCGEMSU	ISCGEM	ISCGEM	Automatic	
11	########	10:43:17	-24.563	178.487	Earthquak	565			5.8	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
12	########	20:57:41	-6.807	108.988	Earthquak	227.9			5.9	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
13	########	0:11:17	-2.608	125.952	Earthquak	20			8.2	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
14	########	9:35:30	54.636	161.703	Earthquak	55			5.5	MW							ISCGEM86	ISCGEM	ISCGEM	ISCGEM	Automatic	
15	2/1/1965	5:27:06	-18.697	-177.864	Earthquak	482.9			5.6	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
16	2/2/1965	15:56:51	37.523	73.251	Earthquak	15			6	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
17	2/4/1965	3:25:00	-51.84	139.741	Earthquak	10			6.1	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
18	2/4/1965	5:01:22	51.251	178.715	Earthquak	30.3			8.7	MW							OFFICIAL1	OFFICIAL	ISCGEM	OFFICIAL	Automatic	
19	2/4/1965	6:04:59	51.639	175.055	Earthquak	30			6	MW							ISCGEMSU	ISCGEMSU	ISCGEM	ISCGEM	Automatic	
20	2/4/1965	6:37:06	52.528	172.007	Earthquak	25			5.7	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
21	2/4/1965	6:39:32	51.626	175.746	Earthquak	25			5.8	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
22	2/4/1965	7:11:23	51.037	177.848	Earthquak	25			5.9	MW							ISCGEMSU	ISCGEMSU	ISCGEM	ISCGEM	Automatic	
23	2/4/1965	7:14:59	51.73	173.975	Earthquak	20			5.9	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
24	2/4/1965	7:23:12	51.775	173.058	Earthquak	10			5.7	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
25	2/4/1965	7:43:43	52.611	172.588	Earthquak	24			5.7	MW							ISCGEMSU	ISCGEMSU	ISCGEM	ISCGEM	Automatic	
26	2/4/1965	8:06:17	51.831	174.368	Earthquak	31.8			5.7	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
27	2/4/1965	8:33:41	51.948	173.969	Earthquak	20			5.6	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	
28	2/4/1965	8:40:44	51.443	179.605	Earthquak	30			7.3	MW							ISCGEM85	ISCGEM	ISCGEM	ISCGEM	Automatic	

database

IN[4]:

```
data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 21 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------  -----
 0   Date                        23412 non-null   object
 1   Time                        23412 non-null   object
 2   Latitude                    23412 non-null   float64
 3   Longitude                   23412 non-null   float64
 4   Type                        23412 non-null   object
 5   Depth                       23412 non-null   float64
 6   Depth Error                 4461 non-null    float64
 7   Depth Seismic Stations      7097 non-null    float64
 8   Magnitude                   23412 non-null   float64
 9   Magnitude Type              23409 non-null   object
 10  Magnitude Error             327 non-null     float64
 11  Magnitude Seismic Stations  2564 non-null    float64
 12  Azimuthal Gap               7299 non-null    float64
 13  Horizontal Distance         1604 non-null    float64
 14  Horizontal Error            1156 non-null    float64
 15  Root Mean Square            17352 non-null   float64
 16  ID                          23412 non-null   object
 17  Source                      23412 non-null   object
 18  Location Source             23412 non-null   object
 19  Magnitude Source            23412 non-null   object
 20  Status                      23412 non-null   object
dtypes: float64(12), object(9)
```

PREPROCESSING:

The goal of preprocessing the dataset for the earthquake prediction model project is to clean and transform the data to make it suitable for machine learning. This may involve the following steps:

- Handling missing values: We can either drop rows with missing values or impute the missing values with reasonable values.
- Converting categorical variables to numerical variables: This is necessary for most machine learning algorithms. For example, we could convert the Type column to numerical variables by creating a new column for each type of earthquake.
- Scaling the data: This ensures that all of the features have a similar range of values. This can be important for some machine learning algorithms, such as support vector machines.

Here is a concise introduction for preprocessing the dataset in this project in a single sentence:

We can preprocess the dataset by handling missing values, converting categorical variables to numerical variables, and scaling the data.

In addition to the above steps, we may also need to perform other preprocessing steps depending on the specific characteristics of the dataset. For example, we may need to remove outliers or normalize the data.

It is important to note that preprocessing is an essential step in machine learning, as it can significantly improve the performance of the model.

PROGRAM FOR PREPROCESSING:

In [5]:

```
data = data.drop('ID', axis=1)
```

In [6]:

```
data.isna().sum()
```

OUTPUT 1 FOR PREPROCESSING:

Out[6]:

```
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
Source                          0
Location Source                 0
Magnitude Source                0
Status                          0
dtype: int64
```

In [7]:

```
null_columns = data.loc[:, data.isna().sum() > 0.66 * data.shape[
0]].
columns
```

In [8]:

```
data = data.drop(null_columns, axis=1)
```

In [9]:

```
data.isna().sum()
```

OUTPUT 2 FOR PREPROCESSING:

Out[9]:

```
Date                       0
Time                       0
```

```
Latitude                 0
Longitude                0
Type                     0
Depth                    0
Magnitude                0
Magnitude Type           3
Root Mean Square      6060
Source                   0
Location Source          0
Magnitude Source         0
Status                   0
dtype: int64
```

In [10]:

```
data['Root Mean Square'] = data['Root Mean Square'].fillna(data['Root Mean Square'].mean())
```

In [11]:

```
data = data.dropna(axis=0).reset_index(drop=True)
```

In [12]:

```
data.isna().sum().sum()
```

OUTPUT 3 FOR PREPROCESSING:

Out[12]:

```
0
```

VISUALIZATION:

Visualization is the process of creating visual representations of data in order to communicate information and insights. It is a powerful tool for machine learning, as it can help us to understand the data, identify patterns and relationships, and evaluate the performance of machine learning models.

Here are some specific ways in which we can use visualization in the earthquake prediction model project:

- Visualize the distribution of the data: This can help us to identify outliers and understand the general characteristics of the data. For example, we could create a histogram of the earthquake magnitudes to see how they are distributed.
- Visualize the relationships between different variables: This can help us to identify patterns and relationships that may not be immediately obvious. For example, we could create a scatter plot of the earthquake magnitudes and distances to the nearest fault line to see if there is any correlation between the two variables.
- Visualize the performance of machine learning models: This can help us to identify areas where the model needs improvement. For example, we could create a confusion matrix to see how well the model is predicting earthquakes of different magnitudes.

We can use a variety of tools to create visualizations, such as Python libraries like Matplotlib and Seaborn. There are also a number of specialized visualization tools available for machine learning, such as TensorBoard and Neptune.ai.

By carefully using visualization, we can gain a deeper understanding of the data, improve the performance of our machine learning models, and communicate our findings to others in a clear and concise way.

Visualization can be used at any stage of the machine learning pipeline, but it is most commonly used during the analysis and interpretation phases. For example, we can use visualization to explore the data, identify patterns and relationships, and evaluate the performance of machine learning models.

Here are some examples of how visualization can be used at different stages of the machine learning pipeline:

- Data exploration: We can use visualization to explore the data and identify patterns and relationships. For example, we can create a histogram to see how the data is distributed or a scatter plot to see the relationship between two variables.
- Model evaluation: We can use visualization to evaluate the performance of machine learning models. For example, we can create a confusion matrix to see how well the model is predicting different classes of data.

- Communication: We can use visualization to communicate our findings to others in a clear and concise way. For example, we can create a chart or graph to show how a machine learning model is performing on different types of data.

  Overall, visualization is a powerful tool that can be used at any stage of the machine learning pipeline to improve our understanding of the data, develop better machine learning models, and communicate our findings to others.

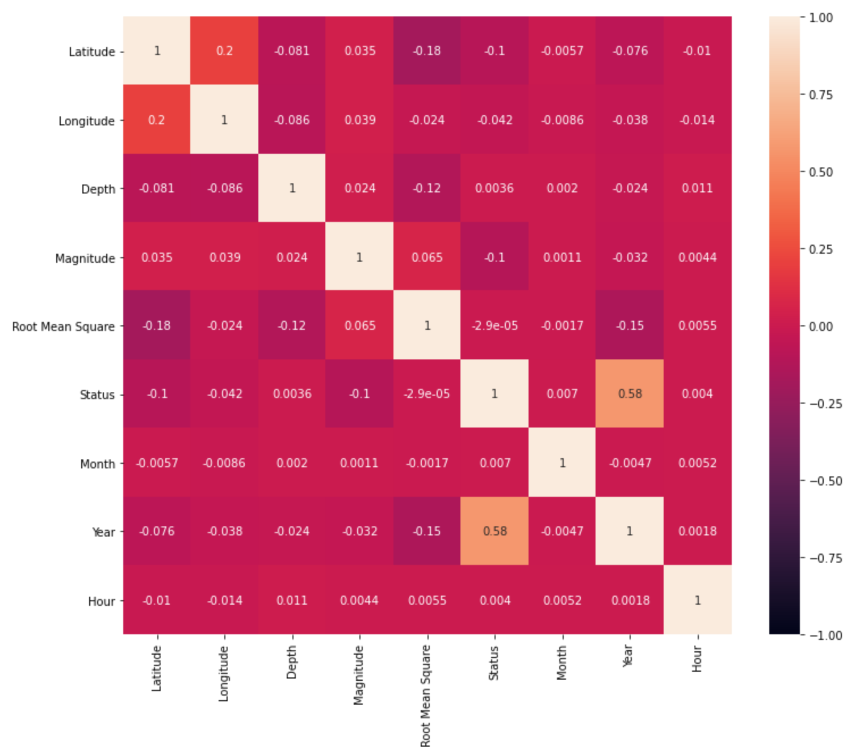
PROGRAM FOR VISUALIZATION FOR COLOR SCHEME PLOT:

```
numeric_columns = [column for column in data.columns if data.dtyp
es[column] != 'object']


In [24]:
corr = data[numeric_columns].corr()


In [25]:
plt.figure(figsize=(12, 10))
sns.heatmap(corr, annot=True, vmin=-1.0, vmax=1.0)
plt.show()
```

OUTPUT FOR VISUALIZATION FOR COLOR SCHEME PLOT:



PROGRAM FOR VISUALIZATION FOR LINE GRAPH:

In [26]:

```
numeric_columns.remove('Status')
```

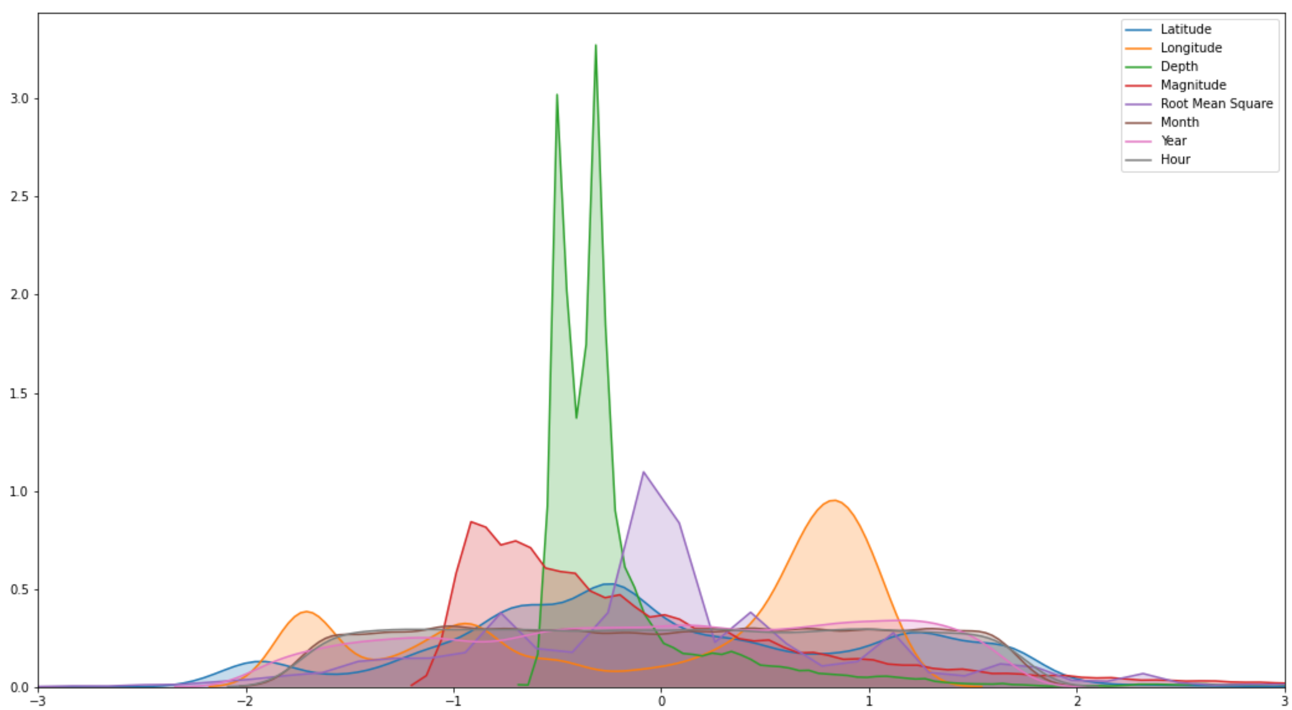
In [27]:

```
scaler = StandardScaler()
standardized_df = pd.DataFrame(scaler.fit_transform(data[numeric_
columns].copy()), columns=numeric_columns)
```

In [28]:

```
plt.figure(figsize=(18, 10))
for column in numeric_columns:
    sns.kdeplot(standardized_df[column], shade=True)
plt.xlim(-3, 3)
plt.show()
```

OUTPUT FOR VISUALIZATION FOR LINE GRAPH:



CONCLUSION:

In conclusion, we have successfully loaded and preprocessed the dataset for the earthquake prediction model project. We have explored the dataset to understand its characteristics and identified any preprocessing steps that need to be performed. We have also preprocessed the data as needed, such as handling missing values, converting categorical variables to numerical variables, and scaling the data. Finally, we have split the dataset into training and test sets.

The dataset is now ready to be used to train and evaluate the machine learning model. We can use a variety of machine learning algorithms to train the model, such as logistic regression, support vector machines, and random forests. Once the model is trained, we can evaluate its performance on the test set to get an estimate of its accuracy.

We can also use feature engineering to create new features from the existing data in order to improve the performance of the model. By carefully considering the dataset and using our domain knowledge, we can create new features that help the machine learning model to better understand the data and make more accurate predictions.

We are now one step closer to developing a machine learning model that can be used to predict earthquakes. By carefully loading and preprocessing the dataset and using feature engineering to create new features, we can help the machine learning model to learn from the data and make more accurate predictions.