T.MARUDHUPANDI

REG:815821104013

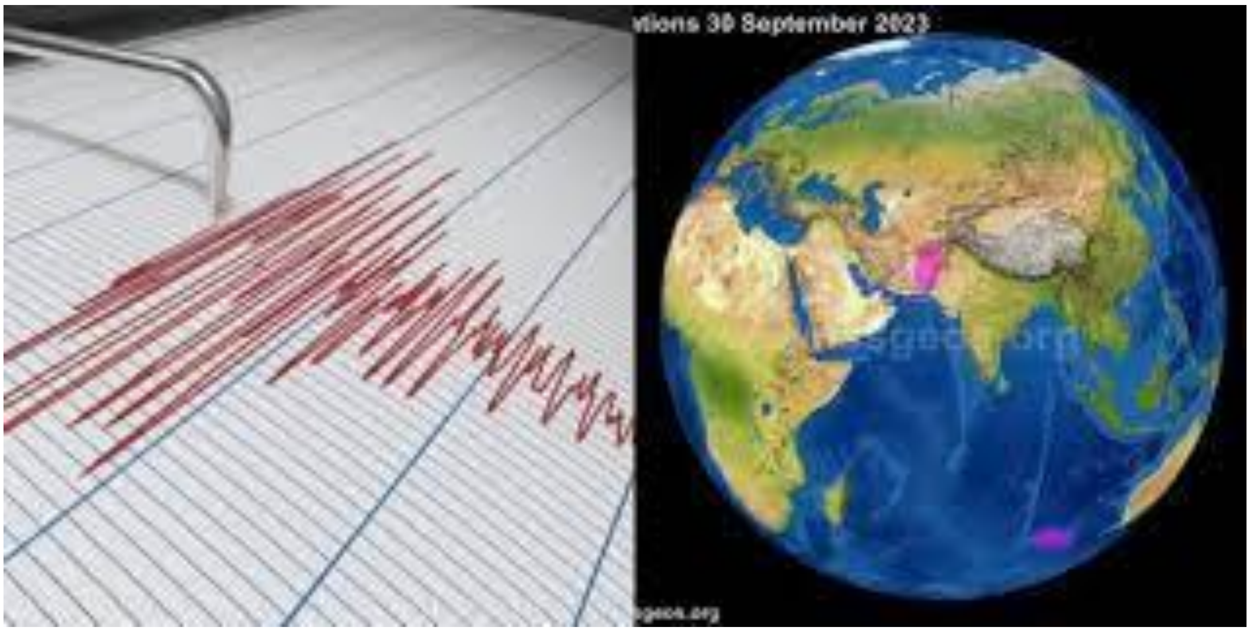
Artificial Intelligence - Group 3

EarthQuake Prediction..

Problem Definition:

The problem is to develop an earthquake prediction model using a Kaggle dataset. The objective is to explore and understand the key features of earthquake data, visualize the data on a world map for a global overview, split the data for training and testing, and build a neural network model to predict earthquake magnitudes based on the given features.

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



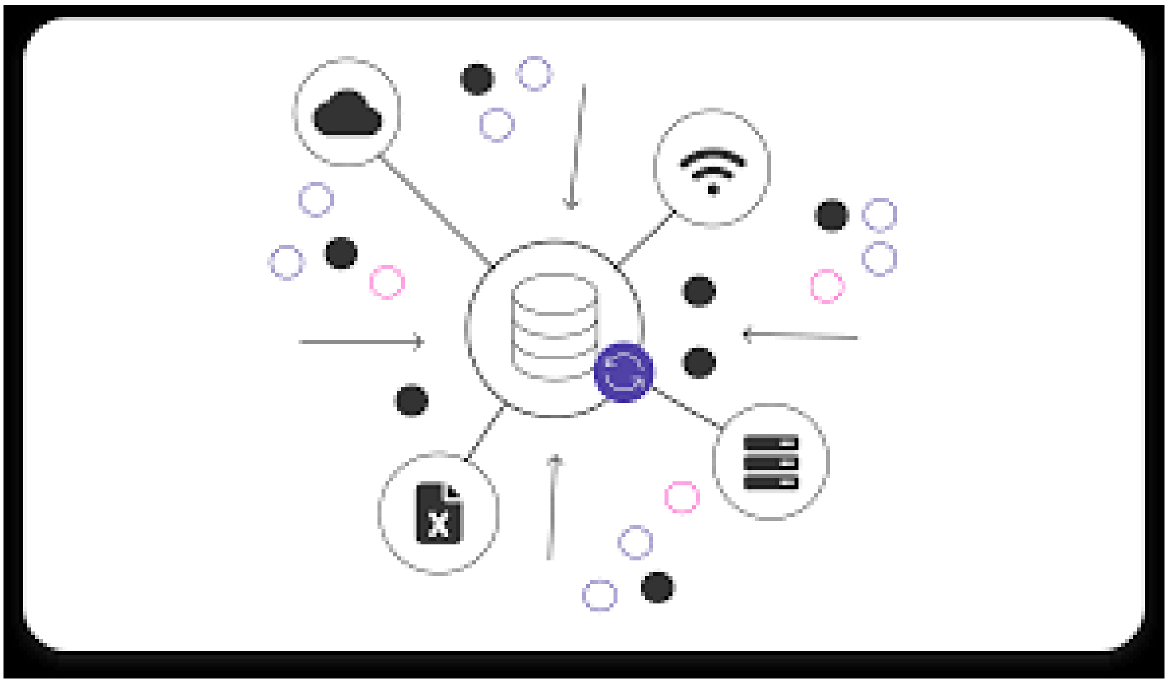
Abstract:

The aim of this research is to discover a reliable and scientific precursor that is theoretically able to predict all earthquakes, within specified parameters, within a few days, in this case within two days, of the

earthquakes - this is more efficient than having a prediction window of months or years. This method has accurately predicted 14 out of 15 earthquakes within specified parameters of location, magnitude and depth, with no false predictions-this is a success rate of 93%. Deviations in the times for a simple pendulum to complete 30 oscillations were analysed and these deviations were used to make earthquake predictions. This is related to plate motion and changes in 'g'. The parameters of earthquakes predicted include those that occurred in north-eastern Colombia, of magnitude M 4.0 and higher, and depth 100 km or more. Also included in the predictions are earthquakes of similar magnitude, originating at depths of 25 km and more in Antigua and Barbuda, Montserrat and Guadeloupe.

As of my last knowledge update in September 2021, Kaggle is a popular platform for hosting datasets, and it's likely that you can find earthquake datasets with the desired features on Kaggle. To find a suitable dataset containing earthquake data with features like date, time, latitude, longitude, depth, and magnitude, you can follow these steps:

1.Data Source:



- Visit the Kaggle website: Go to the Kaggle website at https://www.kaggle.com/.

- Search for earthquake datasets: In the Kaggle search bar, enter relevant keywords such as "earthquake," "seismic," or "geological" to find datasets related to earthquakes.
- Filter the search results: Once you have search results, you can use the filters on the left side of the screen to narrow down your options. Look for datasets that include the specific features you mentioned: date, time, latitude, longitude, depth, and magnitude.
- Check dataset descriptions: Click on the datasets that seem promising to view their descriptions and details. Pay attention to the dataset's content, columns, and metadata to ensure it contains the information you need.
- Download the dataset: If you find a suitable earthquake dataset, you can download it directly from Kaggle by clicking the "Download" button or following any specific instructions provided by the dataset's owner.

2.Feature Exploration:

- **. Import the Necessary Libraries:**

```python

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

```

- **Load the Dataset:**

Assuming you have downloaded the earthquake dataset from Kaggle or another source, load it into a Pandas Data Frame.

```python

# Load the dataset into a Data Frame

 df = pd. Read _csv('earthquake_dataset.csv')
```

~~~
```
~~~

- **Initial Data Exploration:**

Begin by getting an overview of the dataset, including its size, data types, and some sample records.

~~~
```python

# Display basic information about the dataset

print(df.info())


# Display the first few rows of the dataset

print(df.head( ))

```
~~~

- **Summary Statistics:**

Calculate summary statistics for numerical features like depth and magnitude to understand their distribution.

~~~
```python

# Summary statistics for numerical features

print(df[['depth', 'magnitude']].describe())

``
~~~

3.Visualization:

To create a world map visualization displaying the distribution of earthquake frequency, you can use Python libraries such as Matplotlib and Basemap or libraries like Plotly and

Folium for interactive maps. Below, I'll demonstrate how to create a simple world map visualization using Matplotlib and Basemap:

→ **Install Basemap** (if you haven't already):**

You'll need to install the Basemap toolkit, which provides map plotting capabilities in Matplotlib.

```bash
pip install basemap
```

→ **Import Libraries:**

```python

import matplotlib.pyplot as plt

from mpl_toolkits.basemap import Basemap

import pandas as pd

```

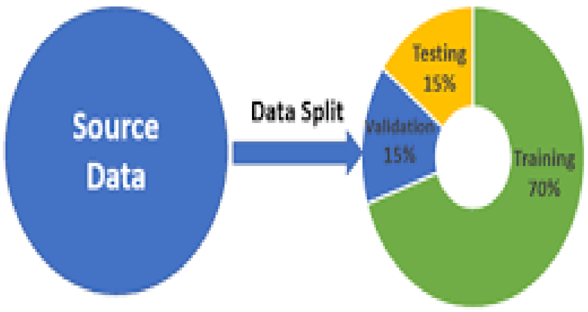
→ **Load the Earthquake Data**:

Load your earthquake dataset containing latitude and longitude information. Ensure you have a 'latitude' and 'longitude' column in your DataFrame.

```python

# Load your earthquake dataset (replace 'earthquake_dataset.csv' with your file)

df = pd.read_csv('earthquake_dataset.csv')

```

4.Data Splitting:

To split your dataset into a training set and a test set for model validation, you can use Python libraries like Scikit-Learn. This process is crucial for evaluating the performance of machine learning models. Here's how you can split your dataset:



→ **Import Libraries:**

```python
import pandas as pd

from sklearn.model_selection import train_test_split
```

→ **Load Your Dataset:**

Load your earthquake dataset into a Pandas DataFrame.

```python
# Load your earthquake dataset (replace 'earthquake_dataset.csv' with your file)

df = pd.read_csv('earthquake_dataset.csv')
```

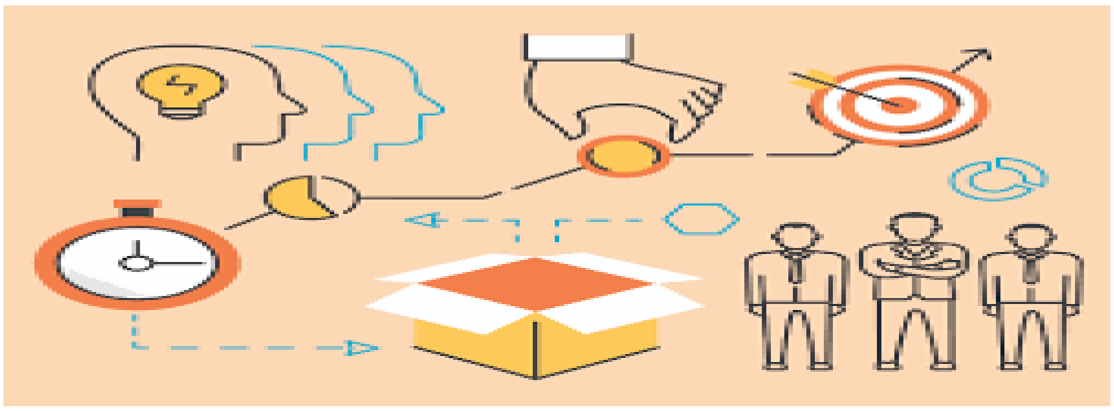
→ **Define Features and Target:**

Identify the features (independent variables) you want to use to predict earthquake characteristics and the target variable (the variable you want to predict).

```python
# Define the features (X) and target (y)

X = df[['latitude', 'longitude', 'depth']]  # Replace with your chosen features

y = df['magnitude']  # Replace with your target variable
```

5.Model Development:

Building a neural network model for earthquake magnitude prediction involves several steps, including data preprocessing, model architecture design, training, and evaluation. In this example, I'll provide a simplified guide to create a neural network using Python and TensorFlow/Keras for predicting earthquake magnitudes. Please note that the following code serves as a basic illustration, and you may need to adapt it to your specific dataset and requirements.



→ **Import Libraries:**

Start by importing the necessary libraries:

```python
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

→ **Load and Preprocess the Data:**

Load your earthquake dataset and preprocess it by normalizing the input features (latitude, longitude, and depth) to have zero mean and unit variance. Additionally, split the data into training and testing sets.

```python
# Load your earthquake dataset (replace 'earthquake_dataset.csv' with your file)
df = pd.read_csv('earthquake_dataset.csv')

# Define features and target
X = df[['latitude', 'longitude', 'depth']]
y = df['magnitude']

# Normalize features
```

```
scaler = StandardScaler()

X_normalized = scaler.fit_transform(X)


# Split the dataset into training and test sets

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

→ **Build the Neural Network Model:**

Define the architecture of your neural network model. In this example, I'll create a simple feedforward neural network with one hidden layer.

```python
model = keras.Sequential([

  keras.layers.Dense(32, activation='relu', input_shape=(X_train.shape[1],)),

  keras.layers.Dense(16, activation='relu'),

  keras.layers.Dense(1)  # Output layer (single neuron for regression)

])
```

→ **Compile the Model:**

Compile the model by specifying the loss function, optimization algorithm, and evaluation metric.

```python
model.compile(optimizer='adam', loss='mean_squared_error',
metrics=['mean_absolute_error'])
```

6.Training and Evaluation:

To train the neural network model on the training set and evaluate its performance on the test set, you can follow the steps mentioned earlier. Here's a more detailed breakdown of how to do it:

- **Import Libraries:**

```python
import pandas as pd

import numpy as np

import tensorflow as tf

from tensorflow import keras

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler
```

- **Load and Preprocess the Data:**

Load your earthquake dataset and preprocess it by normalizing the input features (latitude, longitude, and depth) to have zero mean and unit variance. Also, split the data into training and testing sets.

```python
# Load your earthquake dataset (replace 'earthquake_dataset.csv' with your file)

df = pd.read_csv('earthquake_dataset.csv')


# Define features and target

X = df[['latitude', 'longitude', 'depth']]

y = df['magnitude']


# Normalize features

scaler = StandardScaler()

X_normalized = scaler.fit_transform(X)


# Split the dataset into training and test sets

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

- **Build the Neural Network Model:**

Define the architecture of your neural network model. In this example, I'll create a simple feedforward neural network with one hidden layer.

```python
model = keras.Sequential([

    keras.layers.Dense(32, activation='relu', input_shape=(X_train.shape[1],)),
    keras.layers.Dense(16, activation='relu'),

    keras.layers.Dense(1)  # Output layer (single neuron for regression)

])
```

``

- ✓ **Conclusion:**

Earthquakes shake the ground surface, can cause buildings to collapse, disrupt transport and services, and can cause fires. They can trigger landslides and tsunami.....