PHASE-4

EARTHQUAKE PREDICTION MODEL USING PYTHON

TEAM MEMBERS:

1.SANGEETHA.ML

2.AISHWARYA LAKSHMI.M

3.ABINAYA.P

4.NISHAANTHI.R

5.DHARSHINI.Y

Step 1: Data Collection

You'll need a dataset containing seismic features and corresponding labels indicating whether an earthquake occurred or not. Obtaining a reliable dataset is crucial. Unfortunately, I can't provide a real dataset due to limitations, but you can explore resources like the U.S. Geological Survey (USGS) for seismic data.

Step 2: Data Preprocessing

Preprocess your dataset by cleaning the data, handling missing values, and normalizing features. Here's a basic example of data preprocessing:

```
import pandas as pd

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

# Load your seismic data into a DataFrame
data = pd.read_csv('earthquake_data.csv')

# Split features and labels
X = data.drop('label', axis=1)
y = data['label']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Step 3: Model Selection and Training

Choose an appropriate machine learning algorithm. For simplicity, let's use a Support Vector Machine (SVM) classifier in this example:

```
    from sklearn.svm import SVC

    # Create an SVM classifier
    svm_classifier = SVC(kernel='linear', random_state=42)
    # Train the classifier
    svm_classifier.fit(X_train, y_train)
```

Step 4: Model Evaluation

Evaluate the model's performance on the test dataset:

```
from sklearn.metrics import accuracy_score


# Make predictions
predictions = svm_classifier.predict(X_test)


# Calculate accuracy
accuracy = accuracy_score(y_test, predictions)
print('Accuracy:', accuracy)
```

CODING:

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
```

```
# Load your seismic data (hypothetical example)
# Replace 'earthquake_data.csv' with the path to your dataset
# Your dataset should have features (e.g., magnitude, depth, location) and labels (1 for earthquake, 0 for no earthquake)
data = pd.read_csv('earthquake_data.csv')

# Extract features and labels
X = data.drop('label', axis=1)  # Features
y = data['label']  # Labels

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize features by removing the mean and scaling to unit variance
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Create a Support Vector Machine (SVM) classifier
svm_classifier = SVC(kernel='linear', random_state=42)

# Train the classifier
svm_classifier.fit(X_train, y_train)

# Make predictions on the test set
predictions = svm_classifier.predict(X_test)
```

```
# Calculate accuracy

accuracy = accuracy_score(y_test, predictions)

print('Accuracy:', accuracy)
```