Phase 5: Project Documentation & Submission

Earthquake prediction model

Problem Statement Earthquakes are natural disasters that can have devastating consequences, and there is a significant need for accurate prediction models to help mitigate their impact. This project aims to develop an earthquake prediction model using Python, which can forecast the occurrence of earthquakes based on historical seismic data and relevant features.

Objectives Create a machine learning model to predict the likelihood of earthquakes in a specific region. Utilize seismic data, geological features, and historical earthquake records for training. Implement a user-friendly interface for inputting data and obtaining predictions. Design Thinking Data Collection Gather a comprehensive dataset containing seismic data, geological information, and historical earthquake records. Sources may include geological surveys, seismic monitoring organizations, and publicly available datasets. Data Preprocessing Clean and preprocess the data to handle missing values, outliers, and inconsistencies. Feature engineering: Extract relevant features that could influence earthquake occurrence, such as fault lines, tectonic plate boundaries, depth, and magnitude of previous earthquakes. Model Selection Choose appropriate machine learning algorithms for earthquake prediction. Options may include decision trees, random forests, support vector machines, or deep learning models. Train the selected model on the preprocessed dataset, using historical earthquake occurrence as the target variable. Model Evaluation Evaluate the model's performance using appropriate metrics such as accuracy, precision, recall, and F1-score. Utilize cross-validation techniques to ensure the model's generalization to unseen data. User Interface Develop a user-friendly interface that allows users to input relevant data, such as location, geological features, and seismic data. Provide real-time predictions or forecasts based on the trained machine learning model. Visualization Implement data visualization tools to display seismic data, earthquake probabilities, and historical earthquake records on interactive maps or graphs. Visualize the model's predictions to make them easily understandable to users. Deployment Deploy the earthquake prediction model as a web application or desktop software, ensuring accessibility to users. Continuously update the model with new data to improve prediction accuracy. Usage Install the required Python libraries using pip install -r requirements.txt. Run the application using python earthquake_prediction_app.py. Input relevant data, such as location and geological features. Obtain earthquake predictions and visualize results. Conclusion This project aims to contribute to earthquake prediction efforts by developing a Python-based model that considers various factors influencing seismic activity. By following the design thinking outlined above, we intend to create a valuable tool for early earthquake warning and risk assessment.

Step 1: *First of All needs to download above files given

Step 2: To compile the program needed requirements to be installed to run program source code
*Numpy
*pandas
*Seaborn
*Scikit_learn
*Matplatlib

Step 3: Next you need to download dataset that was provided above dataset to be imported in your python program.

PROGRAM

```
# Step 1: Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt

# Step 2: Load the dataset from "AEP_hourly.csv"
data = pd.read_csv('AEP_hourly.csv')

# Step 3: Data Analysis

# Display the first few rows of the dataset to understand its structure
print("First few rows of the dataset:")
print(data.head())

# Get basic statistics of the dataset
data_statistics = data.describe()
print("\nBasic statistics of the dataset:")
print(data_statistics)

# Step 4: Data Visualization

# Create a figure for the plot
plt.figure(figsize=(12, 6))

# Plot energy consumption over time
plt.plot(data['Datetime'], data['AEP_MW'], label='Energy Consumption (MW)')
```

```
# Customize the plot with labels and title
plt.xlabel('Date and Time')
plt.ylabel('Energy Consumption (MW)')
plt.title('Hourly Energy Consumption')

# Add a grid to the plot
plt.grid()

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Add a legend to the plot
plt.legend()

# Display the plot
plt.show()

# Step 5: Display the data statistics
print("\nData Statistics:")
print(data_statistics)
```