Earthquake prediction model using python

1. Data Collection: Gather seismic data from reliable sources. This could include earthquake event data, ground motion data, fault line information, and any other relevant geospatial or geological data.

2.Data Preprocessing: Prepare and preprocess your data. This might involve cleaning, normalizing, and feature engineering. Feature engineering could be critical in extracting relevant information from seismic data.

3.Feature Selection: Use techniques to select the most relevant features for your prediction task.

4.Data Splitting: Split your data into training and testing sets to evaluate your model's performance.

5.

python

Copy code

```
# Import necessary libraries
import xgboost as xgb
from xgboost import plot_importance
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

# Split the data into features (X) and target labels (y)
X = ...
y = ...

# 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)

# Create an XGBoost classifier
model = xgb.XGBClassifier(objective='binary:logistic')

# Fit the model on the training data
model.fit(X_train, y_train)
```

```
# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")

# Plot feature importances
plot_importance(model)
```

6.Hyperparameter Tuning: You can fine-tune the model's hyperparameters to improve its performance. Tools like GridSearchCV or RandomizedSearchCV can be useful.

7.Evaluation: Evaluate your model's performance using appropriate metrics, such as accuracy, precision, recall, or F1-score. Additionally, consider using domain-specific metrics for earthquake-related tasks.

8.Visualization: Visualize the results and relevant features using libraries like Matplotlib and Seaborn.

Santhosh N

CSE III year

810621104025