Earthquake Prediction using Python

Phase 2 Innovation：

In this phase, we can explore innovative techniques such as ensemble methods and deep learning architectures to improve the prediction system's accuracy and robustness.

Consider advanced techniques such as hyperparameter tuning and feature engineering to improve the prediction model's performance

1.1 Our Innovation :

Let’s explore our innovation in detail

1.1.1 Ensemble Methods :

Predictions from numerous models (including Logistic Regression, Support Vector Machines, and Random Forest) are combined in voting ensembles, which might be hard or soft voting. Hard voting selects the class with the majority of votes, whereas soft voting considers the probability associated with each class.

In our model logistic regression ,we have used soft voting as an ensemble method. Below is the code for our model

voting_clf =VotingClassifier(estimators=[('cnn1',cnn1),('cnn2',cnn2)],voting='soft')

We have created three individual CNNs and combined them into an ensemble using soft voting. Soft voting allows each model to predict probabilities, and the final prediction is based on the average of these probabilities. The accuracy score is used to evaluate the ensemble's performance

1.1.2 Deep Learning Architectures :

Deep Learning Architecture which we used is Convolution Neural Network . We have created two individual models CNN_1, CNN_2.Later these two models are compiled with help of adam optimizer and binary loss entropy.

cnn1.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
cnn2.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

1.1.3 Hyper parameter tuning :

Hyperparameter tuning is done with the help of GridSearchCV to find the best combination of hyperparameters for the individual CNN models.

Defining hyper parameters for tuning:

```
param_grid = {
        'cnn1__activation': ['relu', 'tanh'],
        'cnn1__filters': [32, 64],
        'cnn2__activation': ['relu', 'tanh'],
         'cnn2__filters': [32, 64]
}
```

Grid search for hyper parameter tuning :

```
grid_search = GridSearchCV(ensemble, param_grid, cv=2)
grid_search.fit(X_train, y_train)
```

The model accuracy is increased by following code :

```
best_model = grid_search.best_estimator_
```

1.1.4 Feature Engineering :

The feature engineering stage of machine learning and data modelling is crucial. The performance of a machine learning model can be improved by modifying or adding additional features.

With the use of PCA, scaling, and a soft voting ensemble of CNN models with hyperparameter adjustment, we have applied feature engineering to our model.

```
from sklearn.decomposition import PCA
```

With the help of the following code we can applying feature engineering to our model :

```
pca = PCA(n_components=10)
 X_train = pca.fit_transform(X_train)
 X_test = pca.transform(X_test)
```

Above mentioned is our innovations added to our model. By this the predictive performance of our machine learning model can increase accordingly.