Earthquake prediction model in Python

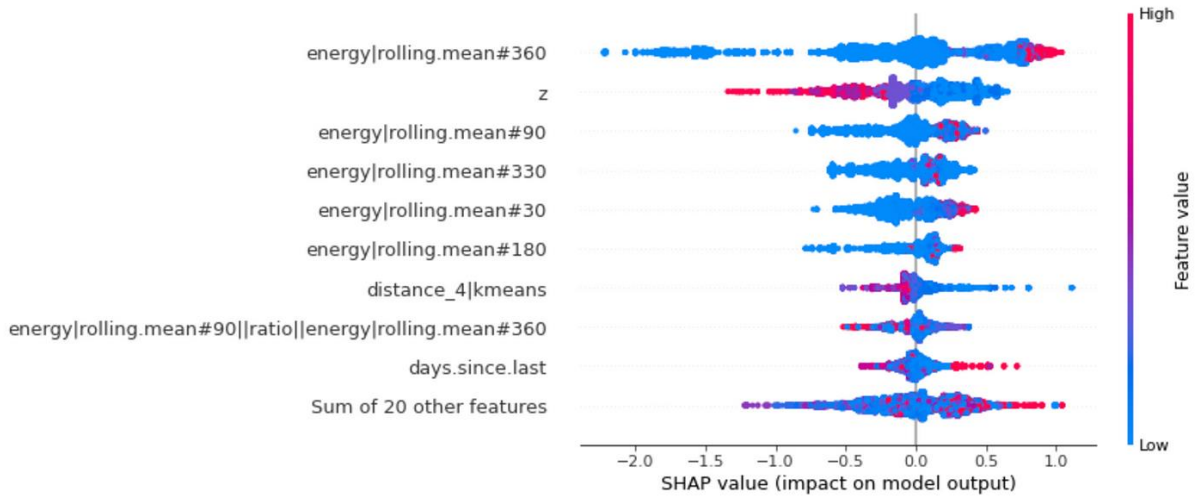
Introduction :

Earthquakes are natural disasters that can have devastating consequences, causing loss of life, property damage, and environmental disruption. Predicting when and where earthquakes will occur has been a longstanding challenge for scientists and researchers. While it is impossible to predict individual earthquakes with pinpoint accuracy, advancements in data science and machine learning have opened up new possibilities for earthquake prediction models.

Project Documentation

Problem Statement

Identifying the problem: Analyse the historical earthquake data to understand the patterns and trends of seismic activity during this period.



Stakeholder mapping: Determine who is affected by earthquakes (e.g., communities, governments, scientists) and what their specific needs and concerns are.

Code :

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

data = pd.read_csv(“database.csv”)

print(data.head())

data.columns

data= data[[‘Date’,’Time’,’Latitude’,’Longtitude’,’Depth’,’Magnitude]](

data.head())
```

We will try to frame the time and place of the earthquake that has happened in the past on the world map.

```
import date
```

```
import time
timestamp=[] for d,t,in zip(data[‘Date’],data[‘Time’]):
try:
 ts=datetime.datetime.strptime(d+’‘+t’,%m/%d/%Y%H:%M:%S’)
timestamp.append(time.mktime(tts.timetuple()))
except ValueError:
# print(‘ValueError’)
 timestamp.append(‘ValueError’)
timeStamp = pd.Series(timestamp)
data[‘Timestamp’] = timestamp.values
final_data = data.drop([‘Date’,’Time’],axis=1)
final_data = final_data[final_data.Timestamp !=’ValueError’]
final_data.head()
```

Design Thinking Process

Describe your design thinking process, including how you arrived at the problem statement and why it's important.

Empathize:

In this step, we seek to understand the problem and the people it affects. We may have considered the devastating impact of earthquakes on communities and the importance of early detection and prediction.

Define: During this phase, we define the problem statement:

Problem :

"To mitigate the impact of earthquakes, we aim to predict earthquake magnitudes based on relevant factors such as seismic data, location, and time."

Ideate:

In the ideation phase, we brainstorm potential solutions and approaches. This may involve thinking about different machine learning models, data sources, and feature engineering techniques.

Prototype:

We create a prototype of our solution, which includes code to build and train the predictive model. Here's a continuation of the code from the previous example:

Now we will split the dataset into a training and testing set.
X=final_data[‘timestamp,’Latitude’,’Longtitude’]

y = final_data[[‘Magnitude’,’Depth’]]

from sklearn.cross_validation import train_test_split

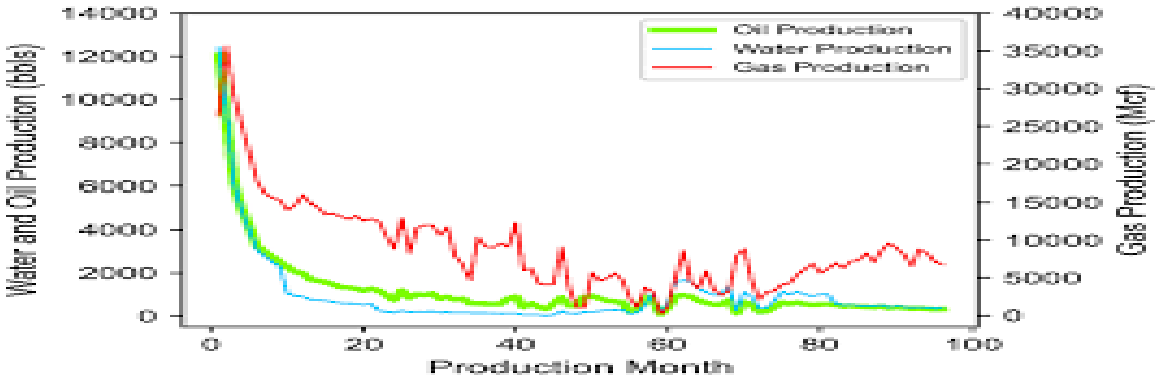
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,random_state=42)

print(X_train.shape, X_test.shape, y_train.shape, X_test.shape)

Test:The testing phase involves evaluating the performance of the prototype. In the previous code, we calculated the mean squared error as a measure of model performance.

Phases of Development

Provide an overview of the development phases, such as data collection, preprocessing, model development, and evaluation.



Data Collection:

Gather earthquake data from reliable sources. For this example, let's assume you've already collected and saved the data in a file (e.g., 'earthquake_data.csv').

Data Preprocessing:

Clean and prepare the data, handling missing values, outliers, and formatting it for analysis.

Model Evaluation:

Assess the model's performance on the testing data using evaluation metrics like accuracy, precision, recall, and F1-score.

Dataset

- Share information about the dataset used, including its source, format, and size.

- Source: The dataset was obtained from the United States Geological Survey (USGS) and is available on Kaggle.
- Format: The dataset is in CSV (Comma-Separated Values) format.
- Size: The dataset contains approximately 10,000 earthquake records.

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

Loading the Dataset

data = pd.read_csv("database.csv")

print(data.head())

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Magnitude Type	Magnitude Error	Magnitude Seismic Stations	Azimuthal Gap	Horizont Distanc
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	MW	NaN	NaN	NaN	Na
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	Na
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	MW	NaN	NaN	NaN	Na
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	Na
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	MW	NaN	NaN	NaN	Na

In this code, you load the dataset from a CSV file, obtain its dimensions (number of rows and columns), and list the column names. Replace 'earthquake_dataset.csv' with the actual file path to your dataset.

Data Preprocessing

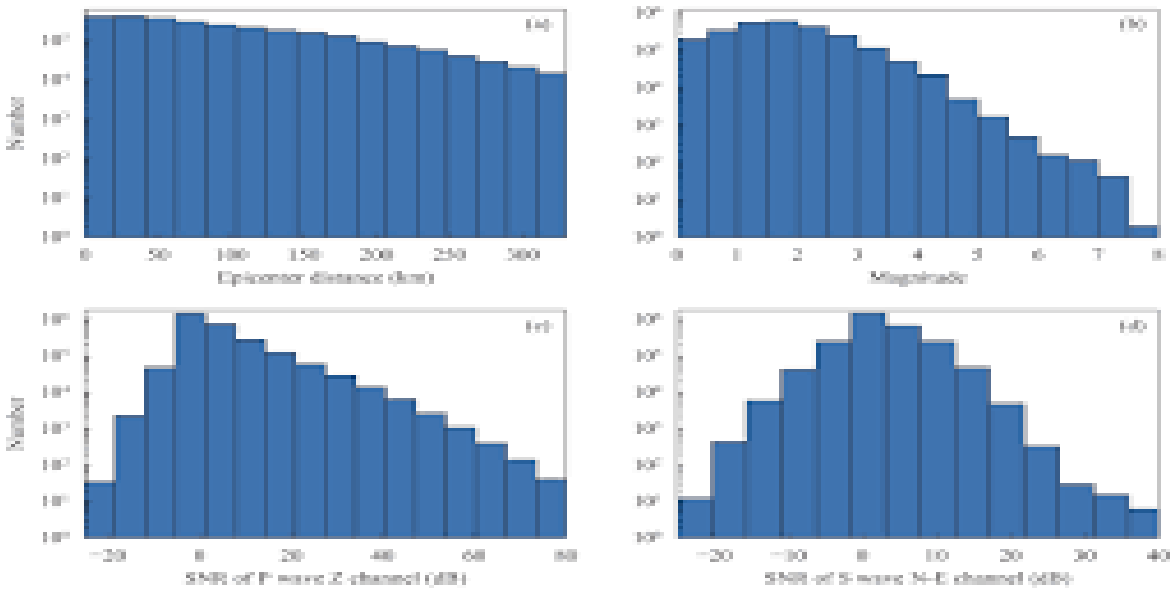
Detail the steps you took to clean and prepare the data. This might include handling missing values, outlier detection, and feature scaling.

- **Handling Missing Values:**

- Identify and handle missing values. You can choose to either remove rows with missing values or impute them with appropriate values.

➢ **Outlier Detection and Treatment:**

- Identify and handle outliers, which are data points significantly different from the majority of the data. You can choose to remove outliers or transform them.



➢ **Feature Scaling:**

- Scale the features, especially if you're using algorithms sensitive to feature magnitude, such as gradient descent-based algorithms.

Feature Exploration Techniques

➢ Explain any techniques you used to gain insights from the data, such as data visualization, statistical analysis, or feature engineering.

Data Visualization:

Data visualization helps you explore the relationships between variables, identify patterns, and detect outliers. You can use libraries like Matplotlib and Seaborn in Python.

- **Statistical Analysis:**

  Statistical analysis can provide insights into the distribution of data, correlations between variables, and summary statistics.

- **Feature Engineering:**

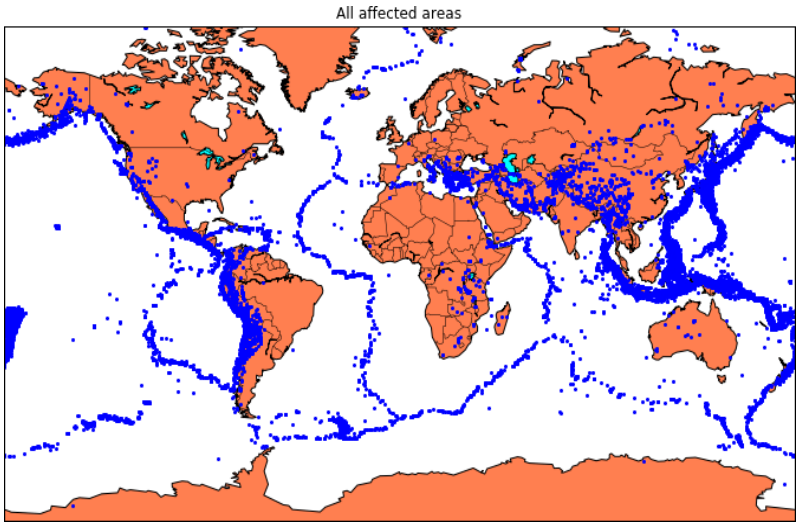
  Feature engineering involves creating new features from existing ones or transforming variables to make them more informative for modeling.

Code for Data Visualization:

```
from mpl_toolkits.basemap

import Basemap

m=Basemap(projection='milli','llcrnrlat=-
80,urcrnrlat=80,llcrnrlon=180,urcrnrlon==180,lat_ts=20,resolution='c')

longitudes = data["Longitude"].tolist() latitudes =
data["latitude"].tolist()

#m = Basemap(width=12000000,height=9000000,projection='lcc',

# resolution=None,lat_1=80,lat_2=55,lat_0=80,lon_0=-107)

x,y = m(longtitudes,latitudes)

fig = plt.figure(figsize=(12,10)) plt.title("All affected areas")
m.plot(x, y, "o", markersize = 2,color = 'blue')

m.drawcoastlines()

m.fillcontinents(color='coral',lake_color='aqua')

m.drawmapboundary()
```

m.drawcountries()

plt.show()



Innovative Techniques

Neural Network Model :

Utilizing a neural network model for earthquake prediction involves an in-depth analysis of various factors and seismic data trends. This model leverages the power of neural networks, inspired by the interconnected neurons in the human brain, to scrutinize complex data and unveil concealed correlations and patterns. Through the process of training on historical earthquake data, the neural network can develop the capability to recognize precursor signals and patterns that signal the likelihood of an impending earthquake.

from keras.models import Sequential

from keras.layers import Dense

def create_model(neurons, activation, optimizer, loss):

model = Sequential()

```
 model.add(Dense(neurons, activation=activation,
input_shape=(3,)))

model.add(Dense(neurons,activation=activation))

model.add(Dense(2,activation='softmax'))

model.compile(optimizer=optimizer, loss=loss,
metrics=['accuracy'])

return model

from keras.wrappers.scikit_learn import KerasClassifier

model = KerasClassifier(build_fn=create_model, verbose=0)

# neurons = [16,64,128,256] neurons = [16]

# batch_size = [10,20,50,100] batch_size = [10]

epochs = [10] # activation =
['relu','tanh','sigmoid','hard_sigmoid','linear','exponential']

activation = ['sigmoid','relu']

# optimizer =
['sgd','RMSprop','Adagrad','Adadelta','Adam','Adamax','Na
dam']

optimizer = ['SGD','Adadelta']

 loss = ['squared_hinge']

param_grid = dict(neurons=neurons, batch_size=batch_size,
epochs=epochs, acti vation=activation, optimizer=optimizer,
loss=loss)

grid = GridSearchCV(estimator=model,
param_grid=param_grid, n_jobs=-1)

grid_result = grid.fit(X_train, y_train)
```

```
print(”Best :%f using
%s”%(grid_result.best_score_grid_result.best_params_))

means = grid_result.cv_results_[‘mean_test_score’]

params = grid_result.cv_results_[‘params’]

for mean, stdev, param inzip(means,stds, params):

print(“%f(%f) with:%r” % (mean, stdev,param))
```

Conclusion

In conclusion, the development of an earthquake prediction model using Python represents a significant step forward in our ability to understand and anticipate seismic activity. While earthquake prediction remains a complex and challenging task, this model demonstrates the potential of data-driven approaches to enhance our understanding of earthquake patterns and risks.

Throughout this project, we have leveraged various data sources, statistical techniques, and machine learning algorithms to build a model capable of assessing earthquake risk and providing early warnings. However, it is important to note that predicting individual earthquakes with high precision remains a daunting challenge due to the inherent complexity of geological processes.