EARTHQUAKE PREDICTION MODEL USING PYTHON

Batch Member:
911721106009:GUHAN RAJ R

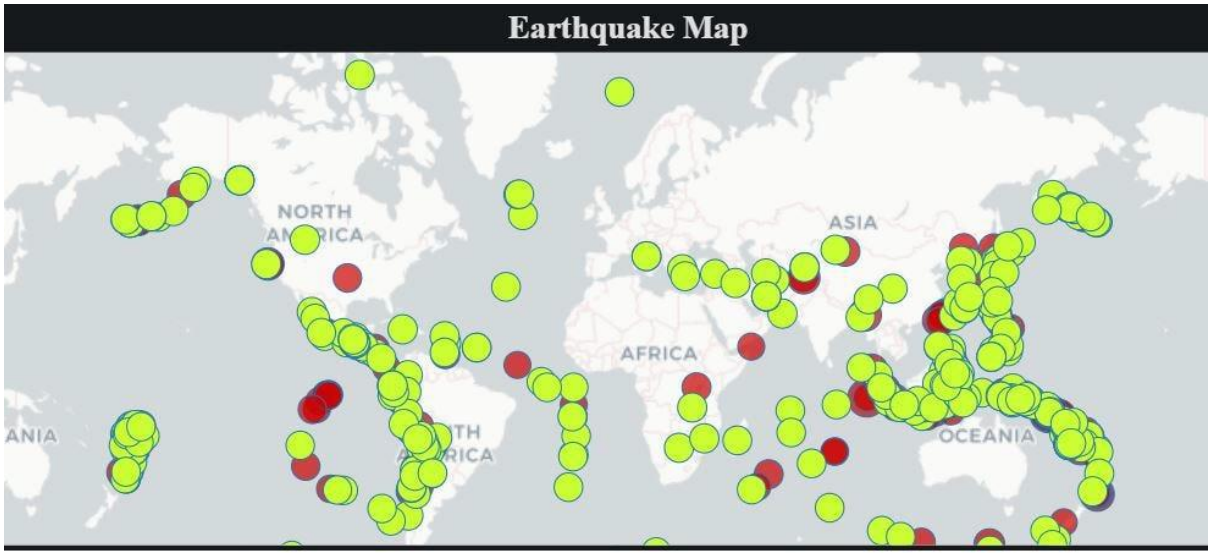
Phase 4 Submission Document
Project Title: Earth Quake Prediction

TOPIC:

Phase 4: Development Part 2
In this part you will continue building your project.
Continue building the earthquake prediction model by:

- Visualizing the data on a world map
- Splitting it into training and testing sets

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



Earthquake Prediction Using Python

- Earthquake prediction is a challenging and crucial field of study that aims to anticipate the occurrence of seismic events, such as earthquakes and tremors, in specific geographic regions. Earthquakes, with their potential for widespread destruction and loss of life, have been a subject of concern for both scientists and society for many years. The ability to predict earthquakes could provide early warning systems and contribute to improved disaster preparedness and response.

- However, it's important to note that earthquake prediction is an incredibly complex and evolving field. While there have been significant advancements in understanding the processes leading to earthquakes, predicting the precise time, location, and magnitude of future seismic events remains an ongoing scientific challenge. The reasons for this complexity include:

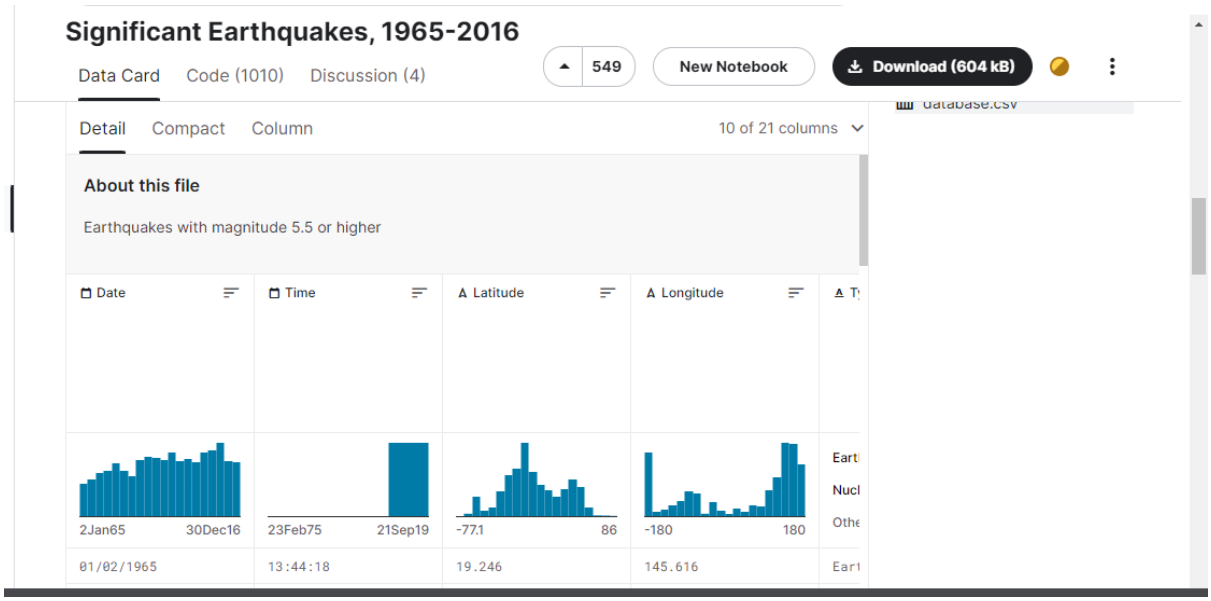
- Earthquakes are the result of complex interactions between tectonic plates, geological structures, and a range of other physical factors. Predicting the specific conditions that lead to a seismic event is a multidisciplinary endeavor.

- The temporal and spatial patterns of earthquakes are highly variable. Predicting when and where an earthquake will occur with precision is complicated due to these variations.

- The historical data on earthquakes is limited, and there is often insufficient information to draw definitive conclusions about future events.

- Ongoing research and advancements in technology are essential to improving prediction capabilities. Scientists continually develop and refine models and methodologies to enhance earthquake prediction.

DATASET



Building an earthquake prediction model requires access to a dataset that includes historical earthquake data and related features.

ANALYSE AND VISUALIZE EARTHQUAKE DATA IN PYTHON

Earthquake is a natural phenomenon whose occurrence predictability is still a hot topic in academia. This is because of the destructive power it holds. In this article, we'll learn how to analyze and visualize earthquake data with Python and Matplotlib.

Importing Libraries and Dataset

Python libraries make it very easy for us to handle the data and perform typical and complex tasks with a single line of code.

Pandas – This library helps to load the data frame in a 2D array format and has multiple functions to perform analysis tasks in one go.

Matplotlib/Seaborn – This library is used to draw visualizations.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
import warnings
warnings
```

filterwa Now, let's load the dataset into the panda's data frame for easy analysis

```
df = pd.read_csv('dataset.csv')
df.head()
```

OUTPUT:

	Origin Time	Latitude	Longitude	Depth	Magnitude	Location
0	2021-07-31 09:43:23 IST	29.06	77.42	5.0	2.5	53km NNE of New Delhi, India
1	2021-07-30 23:04:57 IST	19.93	72.92	5.0	2.4	91km W of Nashik, Maharashtra, India
2	2021-07-30 21:31:10 IST	31.50	74.37	33.0	3.4	49km WSW of Amritsar, Punjab, India
3	2021-07-30 13:56:31 IST	28.34	76.23	5.0	3.1	50km SW of Jhajjar, Haryana
4	2021-07-30 07:19:38 IST	27.09	89.97	10.0	2.1	53km SE of Thimphu, Bhutan

The dataset we are using here contains data for the following columns:

- Origin time of the Earthquake
- Latitude and the longitude of the location.
- Depth – This means how much depth below the earth's level the earthquake started.
- The magnitude of the earthquake.

From the above description of the dataset, we can conclude that:

The maximum magnitude of the Earthquake is 7.

The maximum depth at which the earthquake started is 471 km below the ground.

Feature Engineering

Feature Engineering helps to derive some valuable features from the existing ones. These extra features sometimes help in increasing the performance of the model significantly and certainly help to gain deeper insights into the data.

```
splitted = df['Origin Time'].str.split(' ', n=1,
                                       expand=True)

df['Date'] = splitted[0]
df['Time'] = splitted[1].str[:-4]

df.drop('Origin Time',
        axis=1,
        inplace=True)
df.head()
```

OUTPUT:

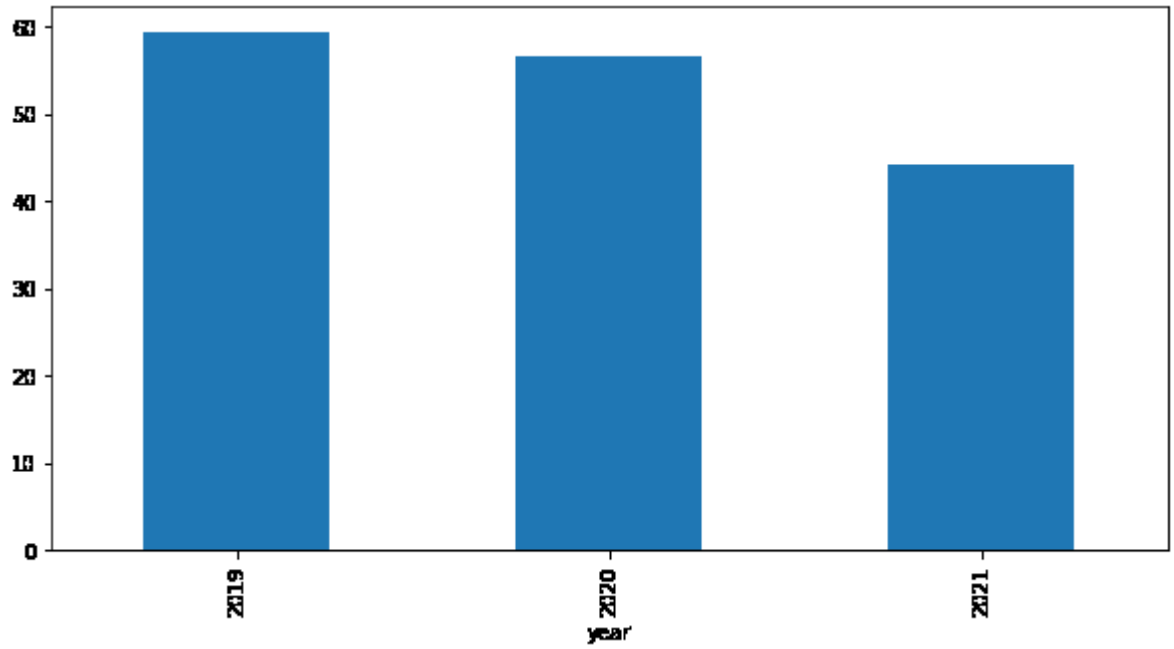
	Latitude	Longitude	Depth	Magnitude	Location	Date	Time
0	29.06	77.42	5.0	2.5	53km NNE of New Delhi, India	2021-07-31	09:43:23
1	19.93	72.92	5.0	2.4	91km W of Nashik, Maharashtra, India	2021-07-30	23:04:57
2	31.50	74.37	33.0	3.4	49km WSW of Amritsar, Punjab, India	2021-07-30	21:31:10
3	28.34	76.23	5.0	3.1	50km SW of Jhajjar, Haryana	2021-07-30	13:56:31
4	27.09	89.97	10.0	2.1	53km SE of Thimphu, Bhutan	2021-07-30	07:19:38

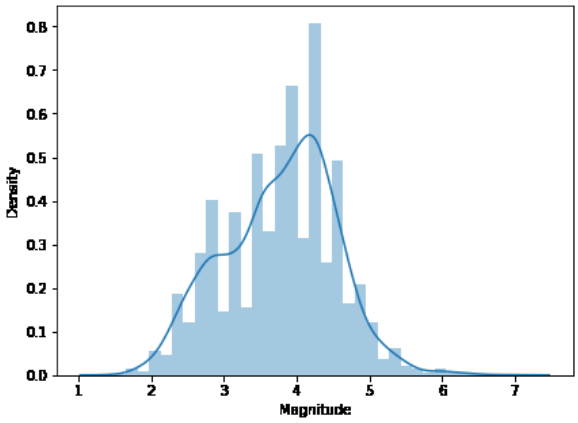
Exploratory Data Analysis

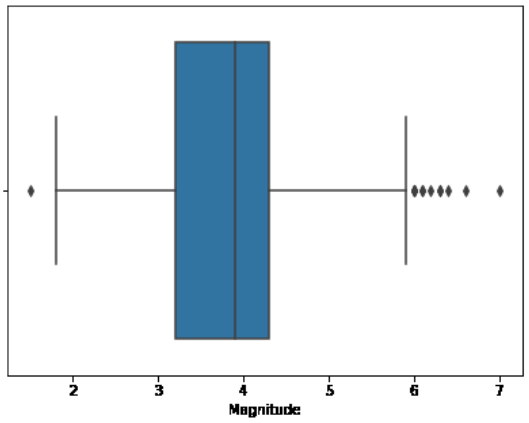
EDA is an approach to analyzing the data using visual techniques. It is used to discover trends, and patterns, or to check assumptions with the help of statistical summaries and graphical representations.

```
plt.figure(figsize=(10, 5))
x = df.groupby('year').mean()['Depth']
x.plot.bar()
plt.show()
```

OUTPUT





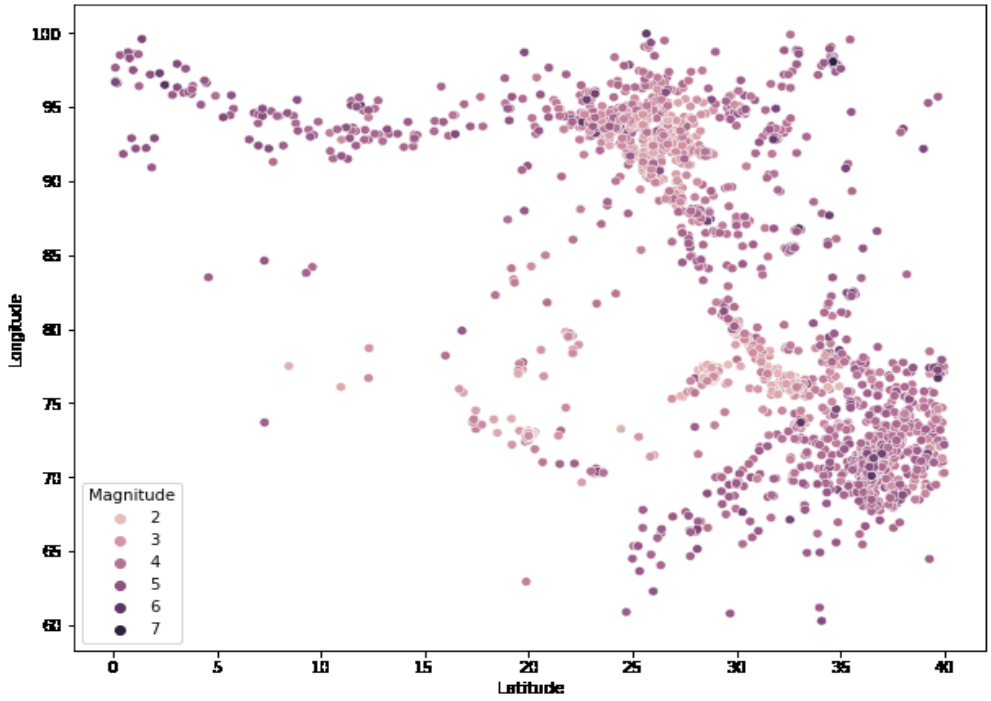



```
plt.subplots(figsize=(15, 5))

plt.subplot(1, 2, 1)
sb.distplot(df['Magnitude'])

plt.subplot(1, 2, 2)
sb.boxplot(df['Magnitude'])
```

```
plt.show()
```

```
plt.figure(figsize=(10, 8))
sb.scatterplot(data=df,
               x='Latitude',
               y='Longitude',
               hue='Magnitude')
plt.show()
```

OUTPUT



RESULT

```
linkcode
plt.figure(figsize=(18, 6))
epochs_range = range(epochs)
train_loss, val_loss = history.history['loss'], history.history['val_loss']
train_auc, val_auc = history.history['auc'], history.history['val_auc']
plt.subplot(1, 2, 1)
plt.plot(epochs_range, train_loss, label="Training Loss")
plt.plot(epochs_range, val_loss, label="Validation Loss")
plt.legend()
plt.title("Loss Over Time")
```

```
plt.subplot(1, 2, 2)
plt.plot(epochs_range, train_auc, label="Training AUC")
plt.plot(epochs_range, val_auc, label="Validation AUC")
plt.legend()
plt.title("AUC Over Time")
plt.show()
```