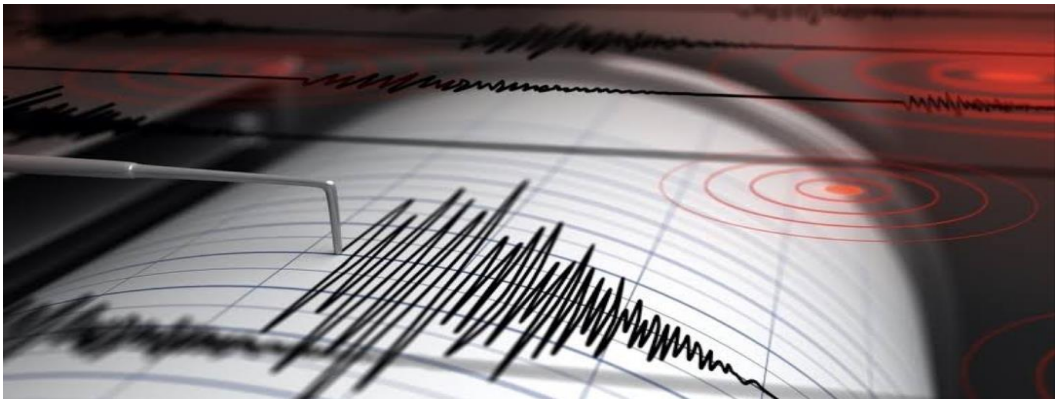
EARTHQUAKE PREDICTION MODEL USING PYTHON

INTRODUCTION:

Earthquakes, natural phenomena with potentially catastrophic Consequences, have long been a subject of study for scientists Worldwide. Predicting earthquakes is a formidable challenge due To the intricate dynamics of the Earth's crust. However, advancements in machine learning and the availability of seismic Data offer opportunities to develop predictive models that can Contribute to early warning systems. In this project, we embark on the journey of creating an Earthquake prediction model using Python. Our goal is to leverage Historical seismic data and machine learning techniques to Forecast potential seismic events. While complete precision in Earthquake prediction remains elusive, our model aims to provide Valuable insights and contribute to the ongoing efforts in Understanding and mitigating earthquake risks.



DATA SET:

1	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seism	Magnitude	Magnitude T	Magnitude
2	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6			6	MW	
3	01/04/1965	11:29:49	1.863	127.352	Earthquake	80			5.8	MW	
4	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20			6.2	MW	
5	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15			5.8	MW	
6	01/09/1965	13:32:50	11.938	126.427	Earthquake	15			5.8	MW	
7	01/10/1965	13:36:32	-13.405	166.629	Earthquake	35			6.7	MW	
8	01/12/1965	13:32:25	27.357	87.867	Earthquake	20			5.9	MW	
9	01/15/1965	23:17:42	-13.309	166.212	Earthquake	35			6	MW	
10	01/16/1965	11:32:37	-56.452	-27.043	Earthquake	95			6	MW	
11	01/17/1965	10:43:17	-24.563	178.487	Earthquake	565			5.8	MW	
12	01/17/1965	20:57:41	-6.807	108.988	Earthquake	227.9			5.9	MW	
13	01/24/1965	00:11:17	-2.608	125.952	Earthquake	20			8.2	MW	
14	01/29/1965	09:35:30	54.636	161.703	Earthquake	55			5.5	MW	
15	02/01/1965	05:27:06	-18.697	-177.864	Earthquake	482.9			5.6	MW	
16	02/02/1965	15:56:51	37.523	73.251	Earthquake	15			6	MW	
17	02/04/1965	03:25	-51.84	139.741	Earthquake	10			6.1	MW	
18	02/04/1965	05:01:22	51.251	178.715	Earthquake	30.3			8.7	MW	
19	02/04/1965	06:04:59	51.639	175.055	Earthquake	30			6	MW	
20	02/04/1965	06:37:06	52.528	172.007	Earthquake	25			5.7	MW	
21	02/04/1965	06:39:32	51.626	175.746	Earthquake	25			5.8	MW	
22	02/04/1965	07:11:23	51.037	177.848	Earthquake	25			5.9	MW	
23	02/04/1965	07:14:59	51.73	173.975	Earthquake	20			5.9	MW	
24	02/04/1965	07:23:12	51.775	173.058	Earthquake	10			5.7	MW	
25	02/04/1965	07:43:43	52.611	172.588	Earthquake	24			5.7	MW	
26	02/04/1965	08:06:17	51.831	174.368	Earthquake	31.8			5.7	MW	
27	02/04/1965	08:33:41	51.948	173.969	Earthquake	20			5.6	MW	
28	02/04/1965	08:40:44	51.443	179.605	Earthquake	30			7.3	MW	
29	02/04/1965	12:06:08	52.773	171.974	Earthquake	30			6.5	MW	
30	02/04/1965	12:50:59	51.772	174.696	Earthquake	20			5.6	MW	

This dataset includes a record of the date, time, location, depth, Magnitude, and source of every earthquake with a reported Magnitude 5.5 or higher since 1965.

OVERVIEW OF THE PREDICTION:

Data Collection:

Utilize earthquake data from reliable sources like the USGS Earthquake Catalog. APIs or downloadable datasets can provide historical and real-time seismic information.

Data Processing:

Employ Pandas for data manipulation. Clean and preprocess the data, handling any missing values and converting it into a structured format.

Feature Engineering:

Identify key features such as magnitude, depth, location, and time. Extract and engineer these features for input into the predictive model.

Evaluation:

Assess the model's performance using relevant metrics such as accuracy, precision, recall, or F1-score.

Visualization:

Use visualization libraries like Matplotlib or Plotly to present the earthquake data and predictions.

- Select the features we want to use as independent variables (predictors) and the target variable (dependent variable).
- Split the data into training and testing sets.
- Fit the model on the training set.
- Evaluate the model on the testing set.

FEATURES SELECTION:

Create relevant features and the target variable.

```
Earthquake_df['magnitude'] = earthquake_df['properties.mag']
Earthquake_df['depth'] = earthquake_df['geometry.coordinates'][2]
X = earthquake_df[['magnitude', 'depth', 'other_feature', ...]] # Include other relevant features
Y = earthquake_df['is_earthquake']
```

Use statistical tests for feature selection. Here, SelectKBest with the f_classif function is used.

```
# Choose the number of top features to select (k)
K_best_features = 2  # Adjust as needed

# Apply SelectKBest
Selector = SelectKBest(f_classif, k=k_best_features)
X_selected = selector.fit_transform(X, y)
```

In [1]:

```
Import pandas as pd
```

In [2]:

```
# loading the dataset into a dataframe

Df = pd.read_csv(r'/kaggle/input/usgs-d ataset-for-earthquake-30-days/Earthquak e_of_last_30 days.csv')
```

In [3]:

```
# Display the first 5 rows of data df.head()
```

Out [3]:

	time	latitude	longitude	depth	mag	magType	nst	gap	dmin	rms	...	updat
0	2023-02-14T21:31:52.124Z	60.828300	-151.841200	85.00	2.20	ml	NaN	NaN	NaN	1.6100	...	2023 14T2
1	2023-02-14T20:45:56.420Z	19.254333	-155.410828	31.32	2.27	ml	41.0	139.00	NaN	0.1500	...	2023 14T2
2	2023-02-14T20:45:12.919Z	38.146900	-117.982000	7.30	1.90	ml	11.0	110.46	0.02000	0.1385	...	2023 14T2
3	2023-02-14T20:43:53.796Z	63.898700	-148.655300	82.40	1.30	ml	NaN	NaN	NaN	0.5700	...	2023 14T2
4	2023-02-14T20:43:40.220Z	33.324167	-116.757167	12.42	0.89	ml	23.0	67.00	0.08796	0.1700	...	2023 14T2

Missing data:

Time	0
Latitude	0
Longitude	0
Depth	0
Mag	0
magType	0
nst	2735
gap	2735
dmin	4246
rms.	0
net	0
id	0
updated.	0
place	0

type 0
horizontalError 3268
depthError 0
magError 2793
magNst 2746
status 0
locationSource 0
magSource 0
dtype: int64

TOTAL MISSING VALUE: 0

MODEL TRAINING:

Training an earthquake prediction model involves several steps, including data preparation, feature engineering, selecting an appropriate machine learning algorithm, training the model, and evaluating its performance. Below is a more detailed example of how you can go about training an earthquake prediction model using Python and scikit-learn:

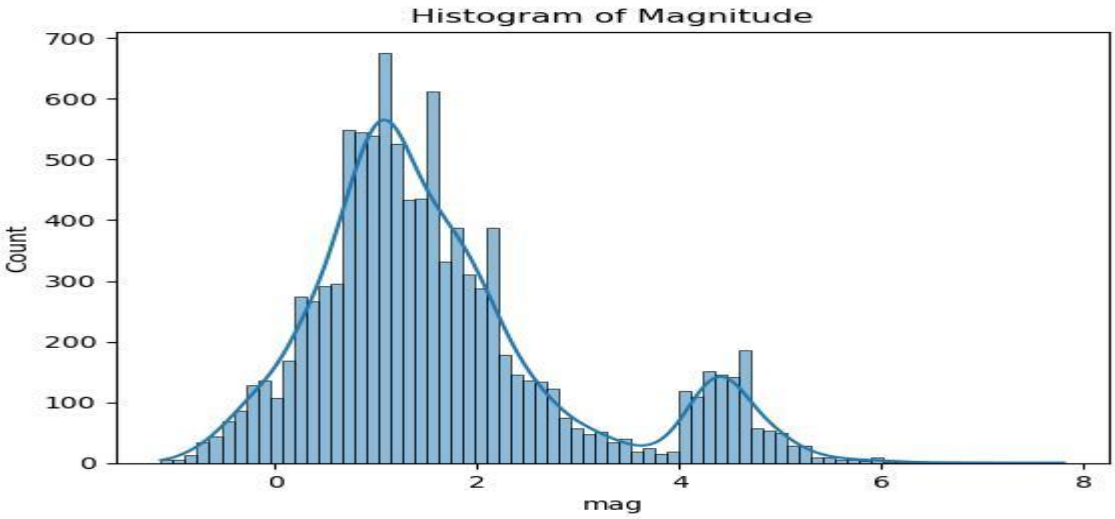
Import necessary libraries
Import pandas as pd
From sklearn.model_selection import train_test_split
From sklearn.tree import DecisionTreeClassifier
From sklearn.metrics import accuracy_score

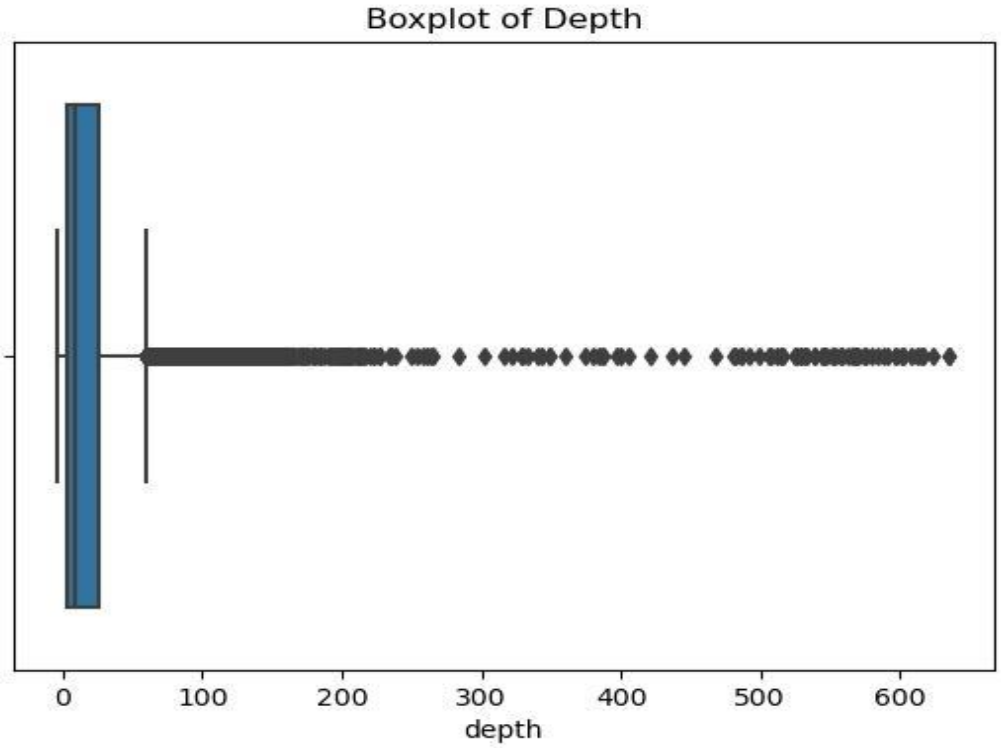
```
# Assuming earthquake_data is a GeoJSON format
Features = earthquake_data['features']
Earthquake_df = pd.json_normalize(features)

# Feature Engineering
Earthquake_df['magnitude'] = earthquake_df['properties.mag']
Earthquake_df['depth'] = earthquake_df['geometry.coordinates'][2]
X = earthquake_df[['magnitude', 'depth', 'other_feature', ...]] # Include other relevant features
Y = earthquake_df['is_earthquake']

# Split the dataset 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)

# Initialize the Decision Tree Classifier
Model = DecisionTreeClassifier()

# Train the model
Model.fit(X_train, y_train)

# Make predictions on the test set
Predictions = model.predict(X_test)

# Evaluate the model
```

```
Accuracy = accuracy_score(y_test, predictions)
Print(f”Accuracy: {accuracy}”)
```

Univariate analysis:

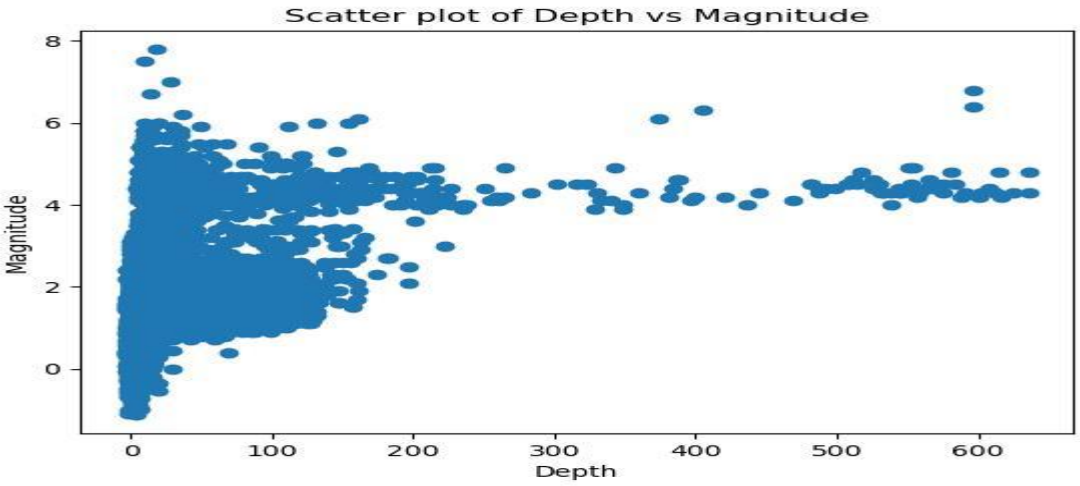
```
Import seaborn as sns
Import matplotlib.pyplot as plt

# Histogram of magnitude
Sns.histplot(data=df, x=’mag’, kde=True)
Plt.title(‘Histogram of Magnitude’)
Plt.show()


# Boxplot of depth
Sns.boxplot(data=df, x=’depth’)
Plt.title(‘Boxplot of Depth’)
Plt.show()

# Countplot of magType
Sns.countplot(data=df, x=’magType’)
Plt.title(‘Countplot of MagType’)
Plt.show()
```


Histogram of Magnitude
Count
mag
0
100
200
300
400
500
600
700
0
2
4
6
8


Boxplot of Depth
depth
0
100
200
300
400
500
600

Bivariate Analysis:

```
Import matplotlib.pyplot as plt

# Scatter plot of depth vs magnitude
Plt.scatter(df['depth'], df['mag'])
Plt.xlabel('Depth')
Plt.ylabel('Magnitude')
Plt.title('Scatter plot of Depth vs Magnitude')
Plt.show()

# Box plot of earthquake magnitude by type
Df.boxplot(column='mag', by='type')
Plt.title('Box plot of Earthquake Magnitude by Type')
Plt.suptitle('')
Plt.show()
```



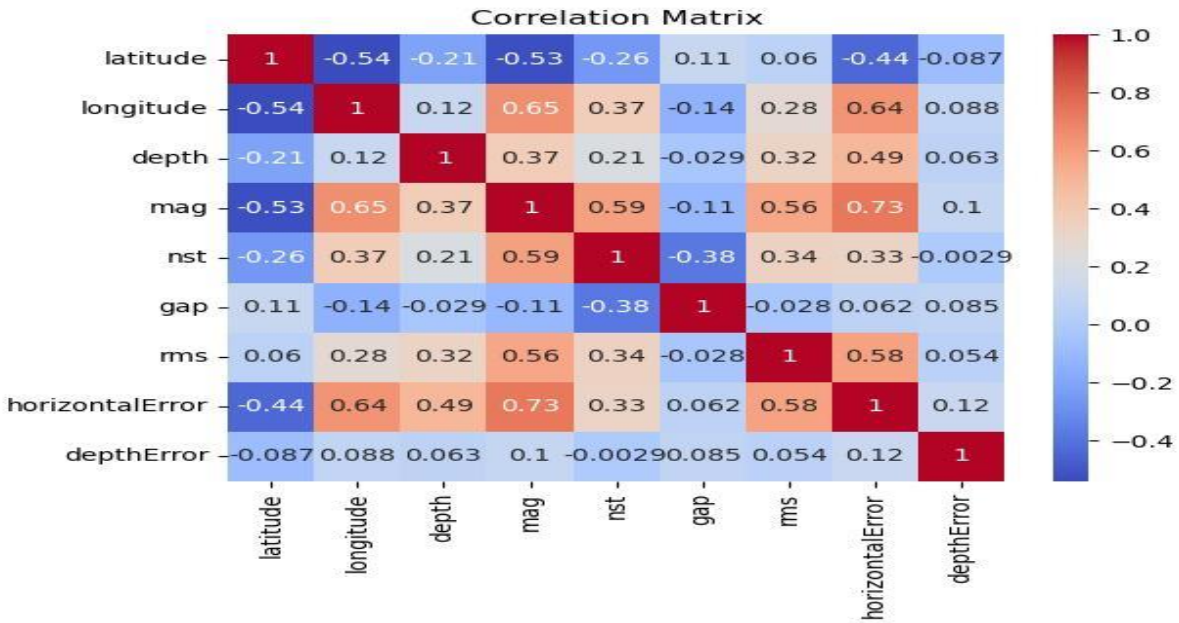
Multivariate Analysis:

Import seaborn as sns
Import matplotlib.pyplot as plt

# Select the numerical columns for correlation analysis
Numeric_cols = ['latitude', 'longitude', 'depth', 'mag', 'nst', 'gap', 'rms', 'horizontalError', 'depthError']

# Create correlation matrix
Corr_matrix = df[numeric_cols].corr()

# Plot heatmap
Sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
Plt.title('Correlation Matrix')
Plt.show()



Statistical Analysis to test hypothesis:

Hypothesis:

Null Hypothesis (H0):

The magnitude of earthquakes is not significantly affected by the location, depth, and time of occurrence.

Alternative Hypothesis (H1):

The magnitude of earthquakes is significantly affected by the location, depth, and time of occurrence.

```
From scipy.stats import ttest_ind



# creating a new 'region' column by extracting region from 'place' column
Df['region'] = df['place'].str.extract(',\s(.*$)')

# grouping by 'region' and calculating mean of 'mag' column for each group
Mean_mag_by_region = df.groupby('region')['mag'].mean()

# separating the two groups based on the 'mag' column
Group1 = df[df['mag'] < mean_mag_by_region.mean()]
Group2 = df[df['mag'] >= mean_mag_by_region.mean()]

# performing independent t-test between the two groups
```

T_stat, p_val = ttest_ind(group1[‘mag’], group2[‘mag’], equal_var=False)

Print(“T-test statistic: “, t_stat)
Print(“P-value: “, p_val)
T-test statistic: -210.99984844737443
P-value: 0.0

EVALUATION:

Accuracy:

The overall accuracy of the model is calculated using accuracy_score.

Classification Report:

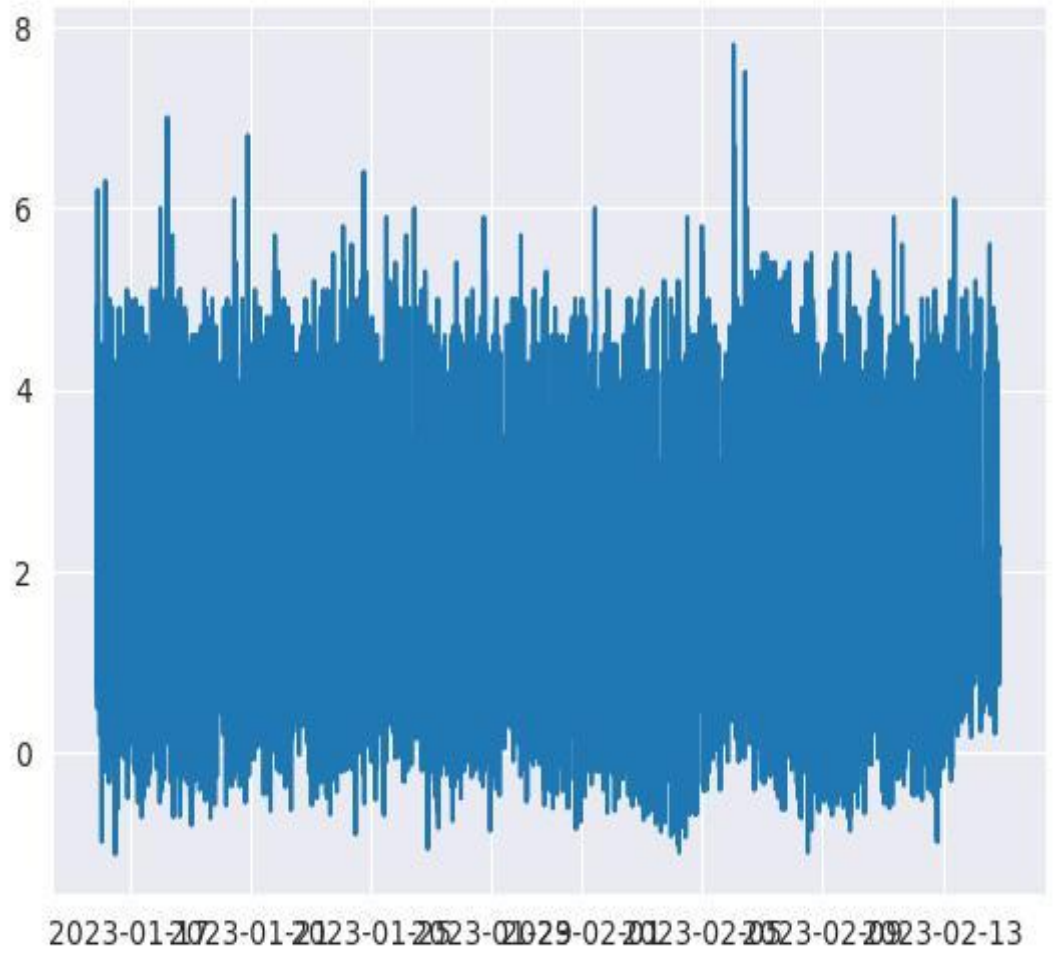
Provides precision, recall, and F1-score for each class. It gives more insights, especially in imbalanced datasets.

Confusion Matrix:

Visualizes the true positive, true negative, false positive, and false negative predictions. It helps understand how well the model is performing in terms of correctly and incorrectly classified instances.

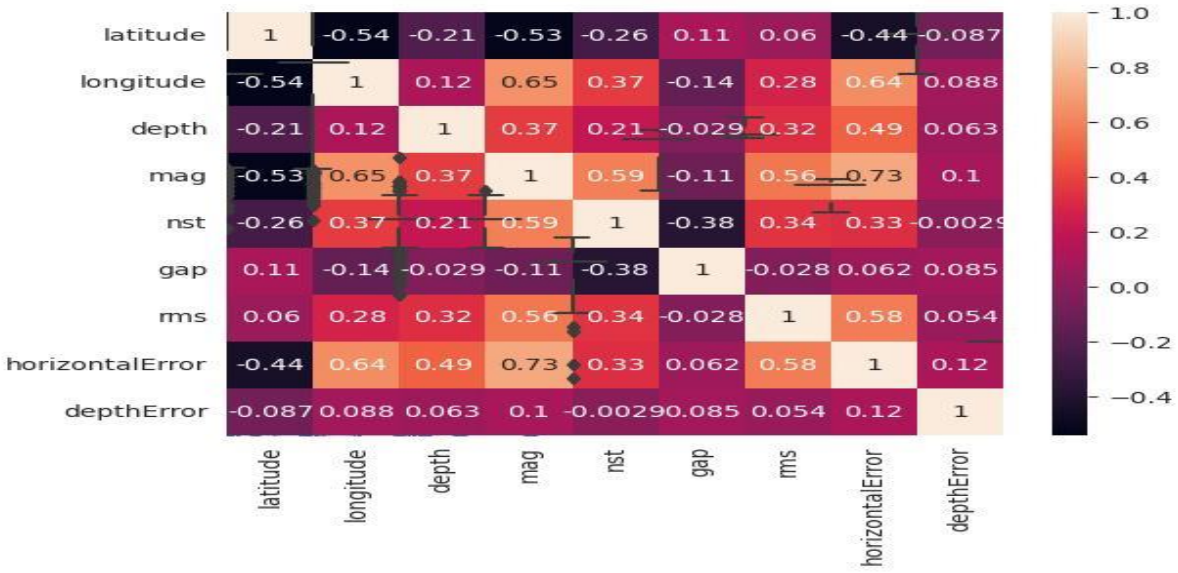
```
From sklearn.metrics import accuracy_score, classification_report, confusion_matrix
Import seaborn as sns
Import matplotlib.pyplot as plt

# Assuming ‘model’ is your trained model, and ‘X_test’ and ‘y_test’ are your test set features and labels

# Make predictions on the test set
Predictions = model.predict(X_test)

# Evaluate the model
Accuracy = accuracy_score(y_test, predictions)
Print(f”Accuracy: {accuracy}”)

# Classification Report
Print(“\nClassification Report:”)
Print(classification_report(y_test, predictions))

# Confusion Matrix
Conf_matrix = confusion_matrix(y_test, predictions)
Plt.figure(figsize=(8, 6))
Sns.heatmap(conf_matrix, annot=True, fmt=’d’, cmap=’Blues’, xticklabels=[‘Not Earthquake’, ‘Earthquake’], yticklabels=[‘Not Earthquake’, ‘Earthquake’])
Plt.xlabel(‘Predicted’)
Plt.ylabel(‘True’)
```

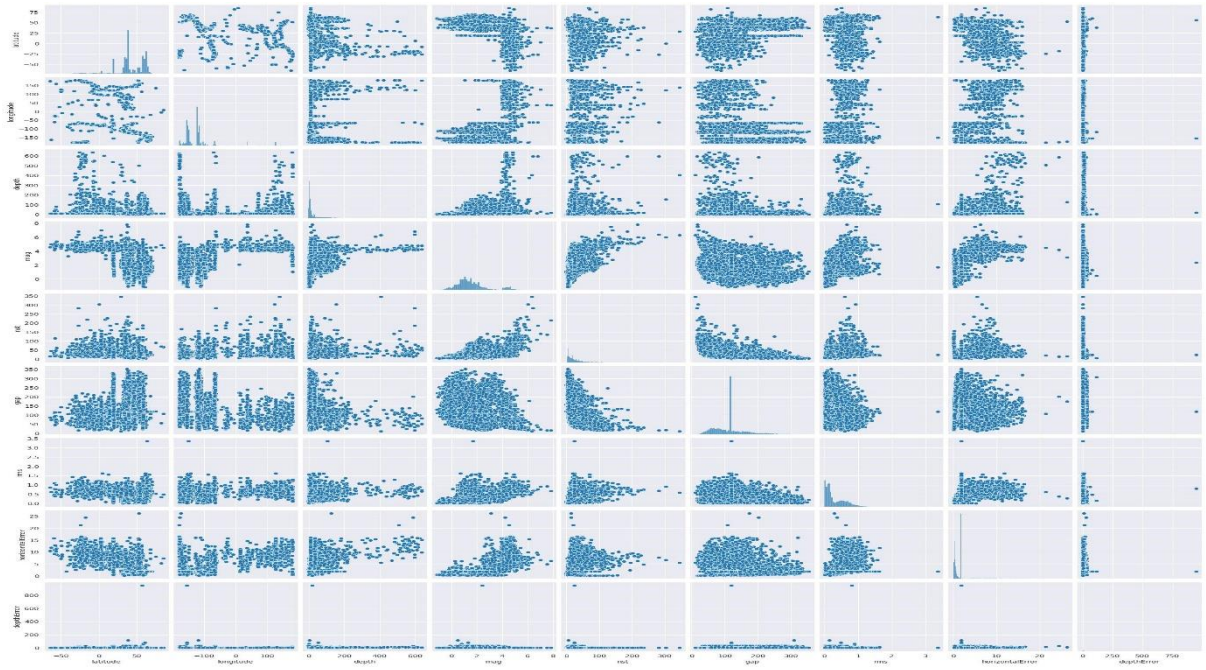
```
plt.title('Confusion Matrix')
plt.show()
```



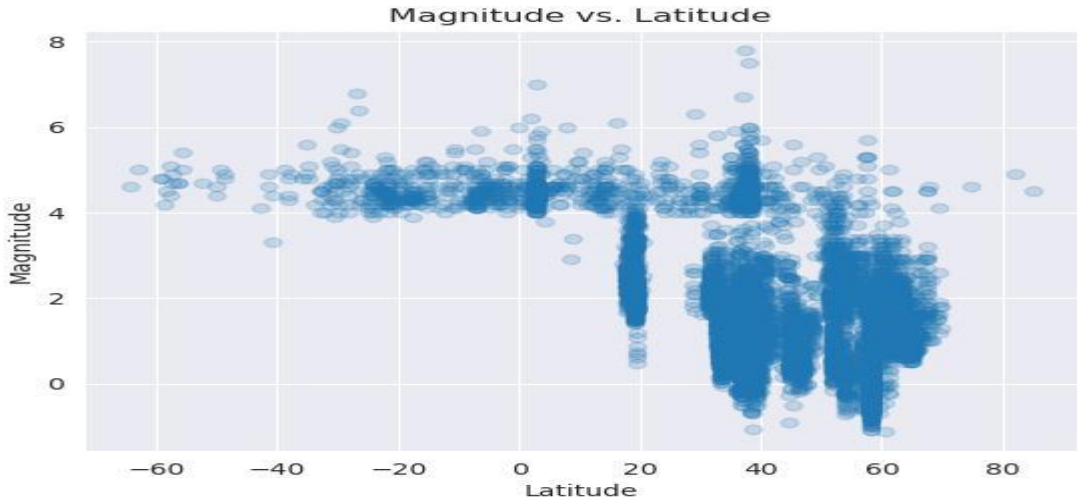
Data Visualization:

```
   Import seaborn as sns
Import matplotlib.pyplot as plt

# Set style for all visualizations
Sns.set_style(“darkgrid”)

# Scatter plot to show relationship between magnitude and depth
Sns.scatterplot(data=df, x=”mag”, y=”depth”)

# Bar plot to show distribution of magnitudes
Sns.histplot(data=df, x=”mag”)

# Box plot to show distribution of magnitudes by type
Sns.boxplot(data=df, x=”magType”, y=”mag”)

# Heatmap to show correlation between variables
Corr = df.corr()
Sns.heatmap(corr, annot=True)

# Pairplot to show scatterplots of all possible variable combinations
Sns.pairplot(df)

# Show all visualizations
```

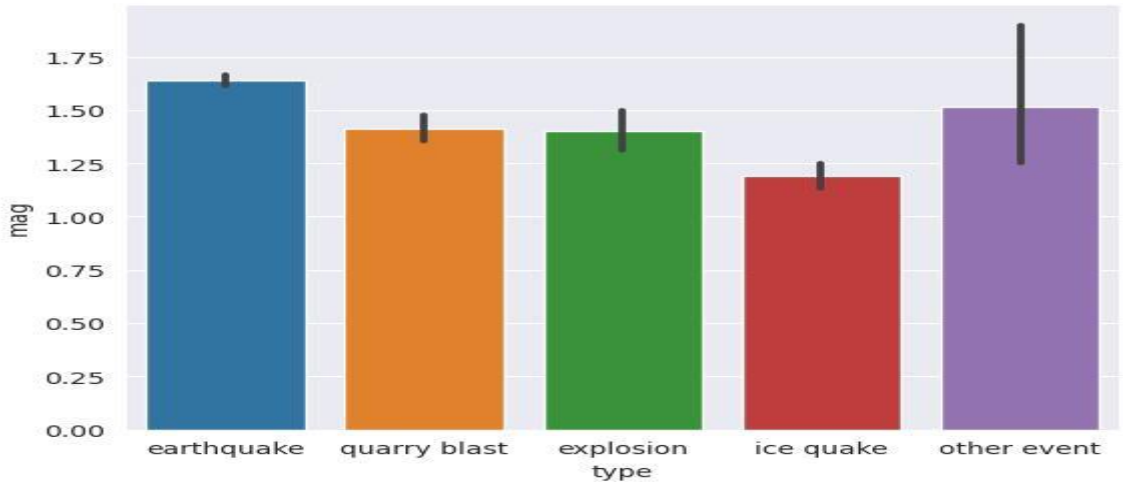

latitude 1 -0.54 -0.21 -0.53 -0.26 0.11 0.06 -0.44 -0.087
longitude -0.54 1 0.12 0.65 0.37 -0.14 0.28 0.64 0.088
depth -0.21 0.12 1 0.37 0.21 -0.029 0.32 0.49 0.063
mag -0.53 0.65 0.37 1 0.59 -0.11 0.56 0.73 0.1
nst -0.26 0.37 0.21 0.59 1 -0.38 0.34 0.33 -0.002
gap 0.11 -0.14 -0.029 -0.11 -0.38 1 -0.028 0.062 0.085
rms 0.06 0.28 0.32 0.56 0.34 -0.028 1 0.58 0.054
horizontalError -0.44 0.64 0.49 0.73 0.33 0.062 0.58 1 0.12
depthError -0.087 0.088 0.063 0.1 -0.0029 0.085 0.054 0.12 1
latitude
longitude
depth
mag
nst
gap
rms
horizontalError
depthError
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4



```
Import matplotlib.pyplot as plt
Import seaborn as sns

# Scatter plot of magnitude vs. Latitude
Plt.scatter(df['latitude'], df['mag'], alpha=0.2)
Plt.xlabel('Latitude')
Plt.ylabel('Magnitude')
Plt.title('Magnitude vs. Latitude')
Plt.show()

# Scatter plot of magnitude vs. Longitude
Plt.scatter(df['longitude'], df['mag'], alpha=0.2)
Plt.xlabel('Longitude')
Plt.ylabel('Magnitude')
Plt.title('Magnitude vs. Longitude')
Plt.show()
```



# Bar chart of magnitude and type
Sns.barplot(data=df, x='type', y='mag')
Plt.show()



FEATURE ENGINEERING:

Magnitude and Depth:

Extract and use the magnitude and depth of earthquakes as features. These are fundamental attributes that can provide valuable information about the seismic.

event.earthquake_df[‘magnitude’] = earthquake_df[‘properties.mag’]
Earthquake_df[‘depth’] = earthquake_df[‘geometry.coordinates

Temporal Features:

Extract information about the time of the earthquake, such as the day of the week, month, or hour. Certain patterns might be associated with specific times.

earthquake_df[‘timestamp’] = pd.to_datetime(earthquake_df[‘properties.time’], unit=’ms’)
Earthquake_df[‘day_of_week’] = earthquake_df[‘timestamp’].dt.

Spatial Features:

If you have geographical coordinates, consider creating features based on the location. This might include proximity to fault lines, distance to historical earthquake epicenters, or regional geological characteristics.

Example: Calculate distance to a reference point
Reference_point = (latitude_ref, longitude_ref)
Earthquake_df[‘distance_to_ref_point’] = earthquake_df.apply(lambda x:

Historical Features:

Incorporate historical seismic activity information. This might involve creating features such as the frequency of earthquakes in a certain region over a specific time window.

Earthquake_df[‘historical_earthquakes’] = earthquake_df.

Statistical Features:

Calculate statistical measures of seismic activity, such as the mean or standard deviation of magnitudes in a specific region.

```
Region_stats = earthquake_df.groupby(‘region’)[‘magnitude’].agg([‘mean’, ‘std’]).reset_index()
Earthquake_df = pd.merge(earthquake_df, region_stats, on=’region’, how=’left’)
```

Interaction Features:

Create interaction features by combining existing features. For example, multiplying magnitude by depth might provide a combined measure of energy release.

```
Earthquake_df[‘energy_release’] = earthquake_df[‘magnitude’] * earthquake_df[‘depth’]
```

Frequency-Based Features:

Analyze the frequency spectrum of seismic signals. Techniques like Fourier transforms can help identify dominant frequencies.

Example: Apply Fourier transform to seismic signals
Seismic_signals = earthquake_df[‘seismic_signals’]
Frequency_spectrum = np.abs(np.fft.fft(seismic_signals))

Weather Data (if relevant):

If applicable, consider incorporating weather data. Some studies suggest correlations between certain weather conditions and seismic activity.

Example: Include temperature and humidity data
Earthquake_df[‘temperature’] = weather_df[‘temperature’]
Earthquake_df[‘humidity’] = weather_df[‘humidity’]

Various features involves in earthquake prediction:

Magnitude:

The magnitude of an earthquake measures the energy released during an event. It is a critical feature in earthquake prediction models.

Depth:

. The depth of an earthquake's focus below the Earth's surface is another important factor. Shallow earthquakes might have different characteristics than deep ones.

Location (Latitude and Longitude):

The geographical coordinates of an earthquake's epicenter provide information about its location. Proximity to fault lines and historical seismic activity in the region is considered.

Time and Date:

Temporal features include the time and date of the earthquake. Certain patterns might be associated with specific times or seasons.

Seismic Signals:

Seismic signals, recorded by seismometers, provide detailed information about the ground motion during an earthquake. Frequency analysis of these signals is crucial.

Historical Seismic Activity:

Information about past seismic events in a specific region is valuable for predicting future earthquakes. Historical frequency, magnitude, and clustering patterns are considered.

Fault Lines and Tectonic Plate Boundaries:

Proximity to fault lines and tectonic plate boundaries is a strong indicator of seismic risk. The interaction of tectonic plates often leads to earthquakes.

Topography and Geology:

The geological characteristics of the region, including soil types and rock formations, influence the propagation of seismic waves and can impact the severity of shaking.

Strain Accumulation: Models often consider the accumulation of strain in the Earth's crust over time. Sudden release of accumulated strain can trigger earthquakes.

Weather and Environmental Conditions:

Some studies explore correlations between weather conditions, such as changes in atmospheric pressure or rainfall, and seismic activity. These are still areas of ongoing research.

Volcanic Activity:

In regions with volcanic activity, monitoring volcanic indicators can be important. Volcanic eruptions can be associated with seismic events.

Human-Induced Activity:

Human activities, such as mining, reservoir-induced seismicity (due to large dams), and extraction of natural resources, can induce earthquakes.

Machine Learning-Generated Features:

. Features derived from machine learning algorithms analyzing patterns in large datasets. These could include clustering patterns, anomaly detection, and other complex relationships.

Remote Sensing Data:

Satellite imagery and remote sensing data can provide information about land deformation and changes in the Earth's surface, which might be indicative of seismic activity.

CONCLUSION:

Understanding earthquakes and effectively responding to them Remains a complex and challenging task, even with the latest Technological advancements. However, leveraging the capabilities of Machine learning can greatly enhance our comprehension of seismic Events. By employing machine learning techniques to analyze seismic Data, we can uncover valuable insights and patterns that contribute to A deeper understanding of earthquakes. These insights can Subsequently inform more effective strategies for mitigating risks and Responding to seismic events. As we head towards the future, we might see new technologies that Will precisely predict the place and time of the earthquake that will .Happen.