Project Report

On

Global Earthquake Prediction

Submitted in partial fulfilment of the requirements for the award of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

(Artificial Intelligence & Machine Learning)

by

Ms. P. SNEHA – 23WH5A6603

Ms. P. KEERTHANA – 23WH5A6605

Ms. P. SUSHMA – 23WH5A6606

Ms. T. RAJESHWARI – 23WH5A6607

Under the esteemed guidance of

Ms. A Naga Kalyani

Assistant Professor, CSE(AI&ML)

VISHNU
UNIVERSAL LEARNING
BVRIT H
Estd. 2012

BVRIT HYDERABAD College of Engineering for Women

(UGC Autonomous Institution | Approved by AICTE | Affiliated to JNTUH)

(NAAC Accredited - A Grade | NBA Accredited B.Tech. (EEE, ECE, CSE and IT)

Bachupally, Hyderabad – 500090

2024-25

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with A Grade

Bachupally, Hyderabad – 500090

2024-25

CERTIFICATE

This is to certify that the major project entitled **"Global earthquake prediction using python"** is a bonafide work carried out **by Ms. P. Sneha (23wh5a6603), Ms. P. Keerthana (23wh5a6605), Ms. P. Sushma(23wh5a6606), Ms. T. Rajeshwari (23wh5a6607)** in partial fulfillment for the award of B. Tech degree in **Computer Science & Engineering (AI&ML), BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad,** affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor	**Head of the Department**
Ms. A Naga Kalyani	**Dr. B. Lakshmi Praveena**
Assistant Professor	**HOD & Professor**
Dept of CSE(AI&ML)	**Dept of CSE(AI&ML)**

External Examiner

DECLARATION

We hereby declare that the work presented in this project entitled **"Global earthquake prediction using python"** submitted towards completion of Project work in IV Year of B.Tech of CSE(AI&ML) at **BVRIT HYDERABAD College of Engineering for Women,** Hyderabad is an authentic record of our original work carried out under the guidance of **Ms. A Naga Kalyani, Assistant Professor, Department of CSE(AI&ML).**

Sign with Date:

P. Sneha

(23wh5a6603)

Sign with Date:

P. Keerthana

(23wh5a6605)

Sign with Date:

P. Sushma

(23wh5a6606)

Sign with Date:

T. Rajeshwari

(23wh5a6607)

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women**, for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. B. Lakshmi Praveena, Head of the Department, Department of CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide**, Ms. A Naga Kalyani, Assistant Professor, CSE(AI&ML), BVRIT HYDERABAD College of Engineering for Women,** for her constant guidance and encouragement throughout the project.

Finally, we would like to thank our Major Project Coordinator, all Faculty and Staff of CSE(AI&ML) department who helped us directly or indirectly. Last but not least, we wish to acknowledge our **Parents and Friends** for giving moral strength and constant encouragement.

P. Sneha (23wh5a6603)

P. Keerthana (23wh5a6605)

P. Sushma(23wh5a6606)

T. Rajeshwari (23wh5a6607)

ABSTRACT

The project aims to build a machine-learning model to predict earthquake magnitudes using a Random Forest Regression approach. Accurate earthquake predictions are vital for mitigating the devastating impact of seismic events on human lives and infrastructure. By leveraging historical seismic data and advanced regression techniques, this project strives to provide reliable magnitude predictions, which can aid in preparedness and early warning systems. Additionally, a classification model is employed to categorize earthquakes above or below critical thresholds, enabling prioritization of resources for high-risk areas. To ensure robustness, the project evaluates model performance using metrics like RMSE, accuracy, and a confusion matrix.

PROBLEM STATEMENT

Earthquakes pose significant risks to human safety, economic stability, and infrastructure. Current systems for predicting earthquake magnitude often lack precision and depend on traditional statistical models, which may fail to capture complex patterns in seismic data.

This project seeks to address the challenge of predicting earthquake magnitudes with high accuracy by employing a machine learning-based approach. The primary goals include:

1. Develop a regression model based on historical seismic data to predict earthquake magnitudes.
2. Creating a classification framework to identify high-risk earthquakes exceeding a critical magnitude threshold.
3. Evaluating the effectiveness of the models through error metrics and classification reports, ensuring applicability in real-world early warning systems.

The ultimate objective is to enhance prediction accuracy, reduce false alarms, and support proactive disaster management strategies.

DATA SET

Global Earthquake prediction – Kaggle

https://www.kaggle.com/datasets/shreyasur965/recent-earthquakes

SOURCE CODE

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor,
RandomForestClassifier  # Import RandomForestClassifier
from sklearn.metrics import mean_squared_error, classification_report,
accuracy_score
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score, accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler
# Load the dataset
data = pd.read_csv('earthquakes.csv')
# Display the first few rows
print(data.describe())
print(data.columns)
```

```
# Correlation Heatmap
plt.figure(figsize=(10, 6))
# Select only numerical features for correlation calculation
numerical_data = data.select_dtypes(include=np.number)
sns.heatmap(numerical_data.corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
#Distribution of target variable
plt.figure(figsize=(8, 5))
sns.histplot(data['magnitude'], kde=True, color='blue')
plt.title("Distribution of Earthquake Magnitudes")
plt.xlabel("magnitude")
plt.ylabel("frequency")
plt.show()
#Scatter plot of Two Features
plt.figure(figsize=(8, 5))
sns.scatterplot(x=data['depth'], y=data['magnitude'])
plt.title("Depth vs Magnitude")
plt.xlabel("Depth")
plt.ylabel("Magnitude")
plt.show()
# Top 10 earthquake-prone locations by frequency
top_locations = data['place'].value_counts().head(10)
plt.figure(figsize=(10,6))
sns.barplot(y=top_locations.index, x=top_locations.values, palette='coolwarm')
plt.title('Top 10 Earthquake-Prone Locations')
plt.xlabel('Number of Earthquakes')
```

```
plt.ylabel('Location')
plt.show()
# get correlation matrix
corr_matrx = data.corr(numeric_only=True)
plt.figure(figsize=(10,10))
# Applying Threshold bases mask
threshold = 0.4
mask = (corr_matrx < threshold) & (corr_matrx != 1)
sns.heatmap(corr_matrx[corr_matrx !=
1],annot=True,fmt='0.2f',cmap='coolwarm',mask=mask)
plt.show()
#threshold uding mean
print("Mean of Green Alert Magnitude  = ",data[data.alert ==
'green']['magnitude'].mean())
print("Mean of Red Alert Magnitude  = ",data[data.alert ==
'red']['magnitude'].mean())
print("Mean of Orange Alert Magnitude  = ",data[data.alert ==
'orange']['magnitude'].mean())
print("Mean of Yellow Alert Magnitude  = ",data[data.alert ==
'yellow']['magnitude'].mean())
# Data preprocessing
# 1. Handle missing values (example: fill with mean)
for col in data.select_dtypes(include=np.number):
    data[col] = data[col].fillna(data[col].mean())

# 2. Convert 'alert' and 'magType' to numerical (example: one-hot encoding)
data = pd.get_dummies(data, columns=['alert', 'magType'], drop_first=True)

# 3. Select features (X) and target (y)
```

```
# Exclude 'type' column or any other columns containing strings
features = ['depth', 'latitude', 'longitude', 'mag', 'nst', 'gap', 'dmin', 'rms',
'horizontalError', 'depthError', 'magError', 'magNst']
# Remove features not in dataset
features = [col for col in features if col in data.columns] #Ensure features are in
the dataset
X = data[features]
y = data['magnitude']

# 4. Scale numerical features
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize and train the model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model (Regression Metrics)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
```

```
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"MSE: {mse}")
print(f"MAE: {mae}")
print(f"R-squared: {r2}")

# For Accuracy and Confusion Matrix (Convert to Classification Problem)
# You'll need to define a threshold to categorize predictions
threshold = 5.0
y_test_class = (y_test >= threshold).astype(int)
y_pred_class = (y_pred >= threshold).astype(int)

# Calculate Accuracy
accuracy = accuracy_score(y_test_class, y_pred_class)
print(f"Accuracy: {accuracy}")

# Calculate and Display Confusion Matrix
cm = confusion_matrix(y_test_class, y_pred_class)
print("Confusion Matrix:")
print(cm)
from sklearn.metrics import classification_report, confusion_matrix,
ConfusionMatrixDisplay
import matplotlib.pyplot as plt
import numpy as np
# Classification Report
class_report = classification_report(y_test_class, y_pred_class)
print("Classification Report:\n", class_report)
```

```
# Confusion Matrix
cm = confusion_matrix(y_test_class, y_pred_class)
# Display the confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[0, 1])
disp.plot()
plt.title("Confusion Matrix")
plt.show()
plt.figure(figsize=(8, 5))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.7)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red',
linestyle='--')  # Line of perfect prediction
plt.title("True vs Predicted Magnitudes")
plt.xlabel("True Magnitude")
plt.ylabel("Predicted Magnitude")
plt.show()
```

OUTPUT

Displaying the Dataset

	magnitude	time	updated	felt	cdi \
count	1137.000000	1.137000e+03	1.137000e+03	1137.000000	1137.000000
mean	4.856675	1.712109e+12	1.716593e+12	414.408091	2.925242
std	1.047840	1.143033e+10	9.671955e+09	5746.971362	2.562707
min	3.000000	1.687542e+12	1.693083e+12	0.000000	0.000000
25%	3.800000	1.701663e+12	1.707609e+12	0.000000	0.000000
50%	5.300000	1.713810e+12	1.719958e+12	2.000000	3.000000
75%	5.600000	1.722885e+12	1.725384e+12	24.000000	5.000000
max	7.600000	1.726661e+12	1.726672e+12	183786.000000	9.000000

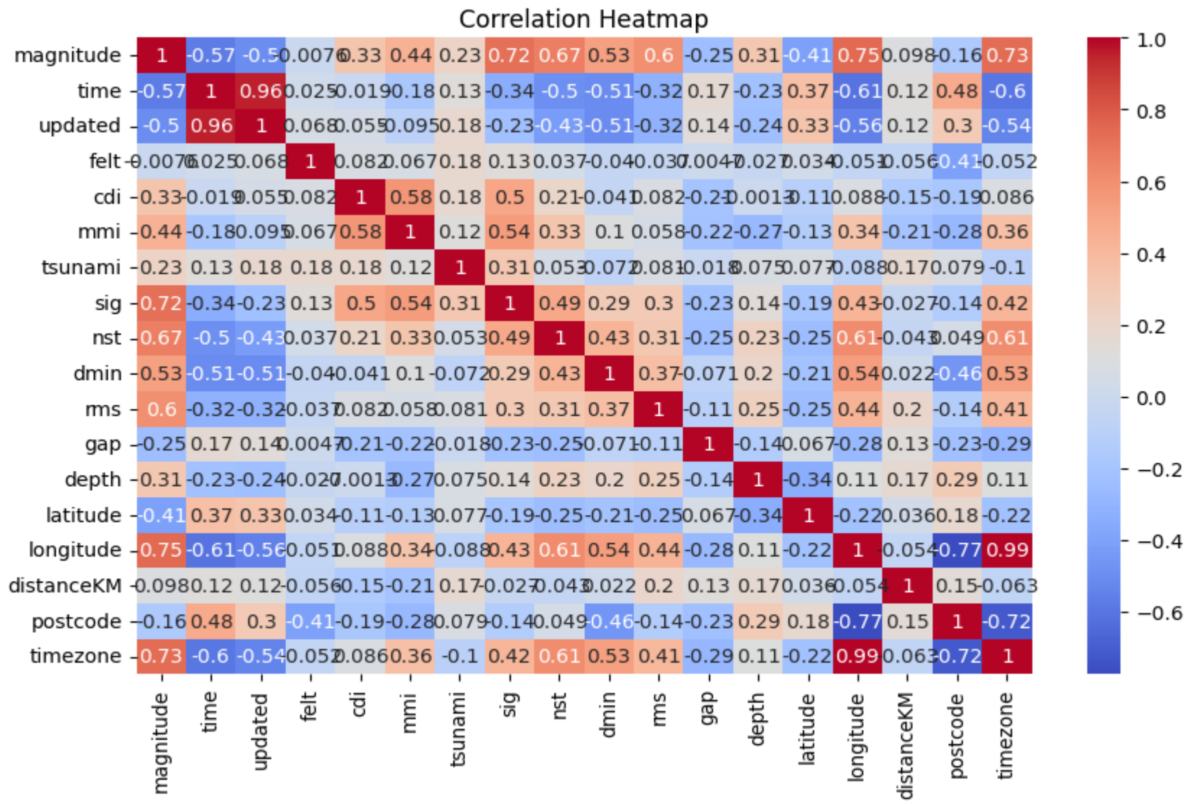
	mmi	tsunami	sig	nst	dmin \
count	1137.000000	1137.000000	1137.000000	1137.000000	1137.000000
mean	4.320141	0.059807	432.698329	115.094107	1.342604
std	1.453949	0.237232	256.177844	91.877870	1.704364
min	1.000000	0.000000	138.000000	0.000000	0.000000
25%	4.000000	0.000000	234.000000	37.000000	0.100000
50%	4.000000	0.000000	449.000000	102.000000	0.680000
75%	5.000000	0.000000	518.000000	157.000000	2.061000
max	9.000000	1.000000	2419.000000	619.000000	12.457000

	rms	gap	depth	latitude	longitude \
count	1137.000000	1137.000000	1137.000000	1137.000000	1137.000000
mean	0.585974	55.055286	41.287300	27.308909	-3.930635
std	0.308556	37.609237	87.866489	20.133139	118.043697

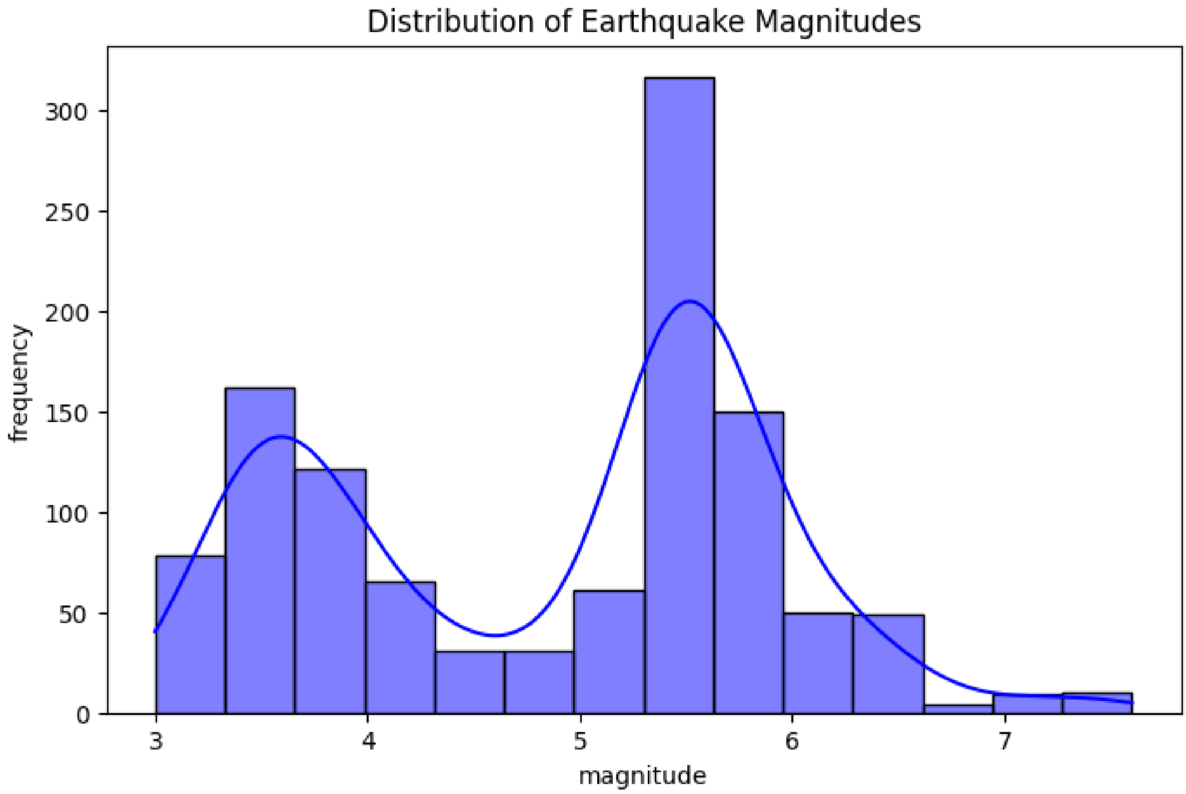
```
min      0.000000    0.000000    -0.250000   -43.706400  -179.807000
25%      0.300000   30.000000     7.550000    24.195400  -104.452000
50%      0.630000   49.000000    10.000000    31.667700   -68.682000
75%      0.780000   68.000000    34.723000    37.497600   126.628000
max      2.520000  256.000000   639.503000    68.176100   179.972000

        distanceKM      postcode     timezone
count  1137.000000    197.000000  1137.000000
mean     52.289358  83086.131980    21.741425
std      56.027469  12812.555204   440.864430
min       0.000000   8833.000000  -720.000000
25%      15.000000  79331.000000  -360.000000
50%      37.000000  79772.000000  -180.000000
75%      61.000000  92530.000000   480.000000
max     298.000000  99827.000000   780.000000
Index(['id', 'magnitude', 'type', 'title', 'date', 'time', 'updated', 'url',
       'detailUrl', 'felt', 'cdi', 'mmi', 'alert', 'status', 'tsunami', 'sig',
       'net', 'code', 'ids', 'sources', 'types', 'nst', 'dmin', 'rms', 'gap',
       'magType', 'geometryType', 'depth', 'latitude', 'longitude', 'place',
       'distanceKM', 'placeOnly', 'location', 'continent', 'country',
       'subnational', 'city', 'locality', 'postcode', 'what3words', 'timezone',
       'locationDetails'],
      dtype='object')
```

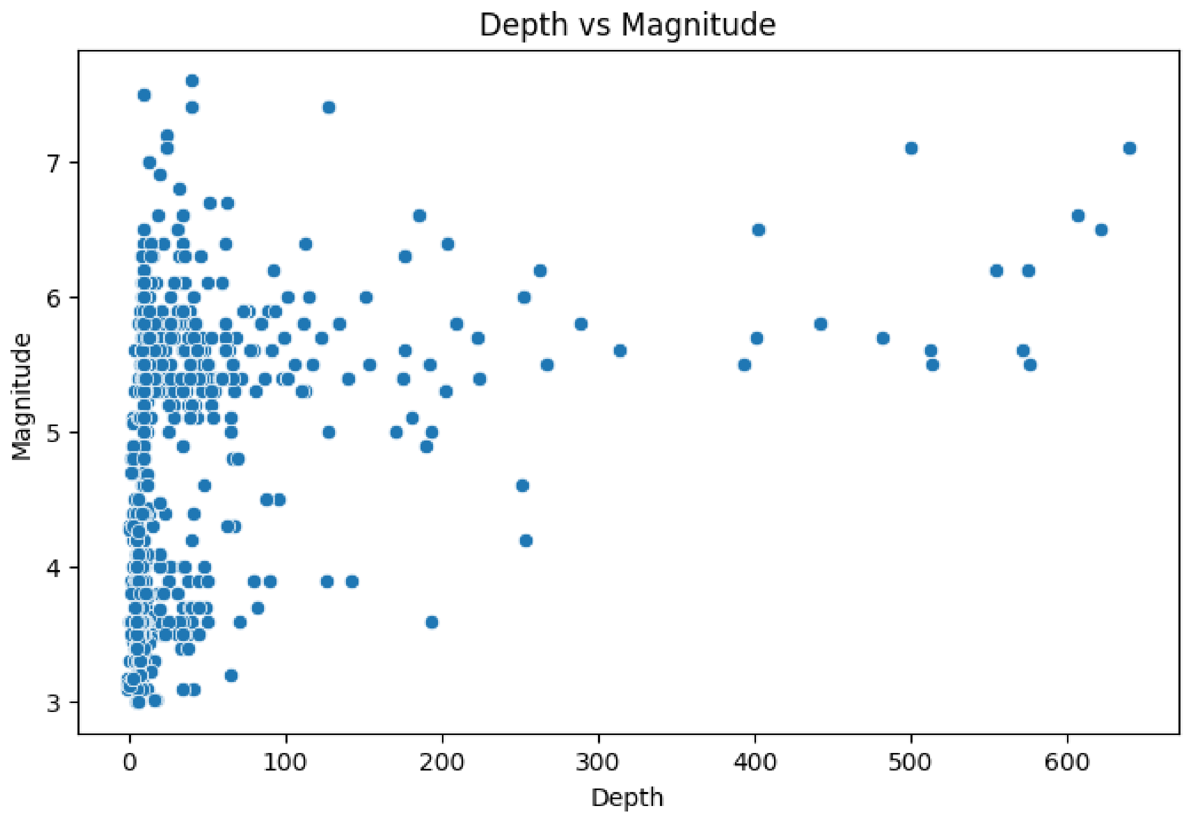
Co-relation Heat Map



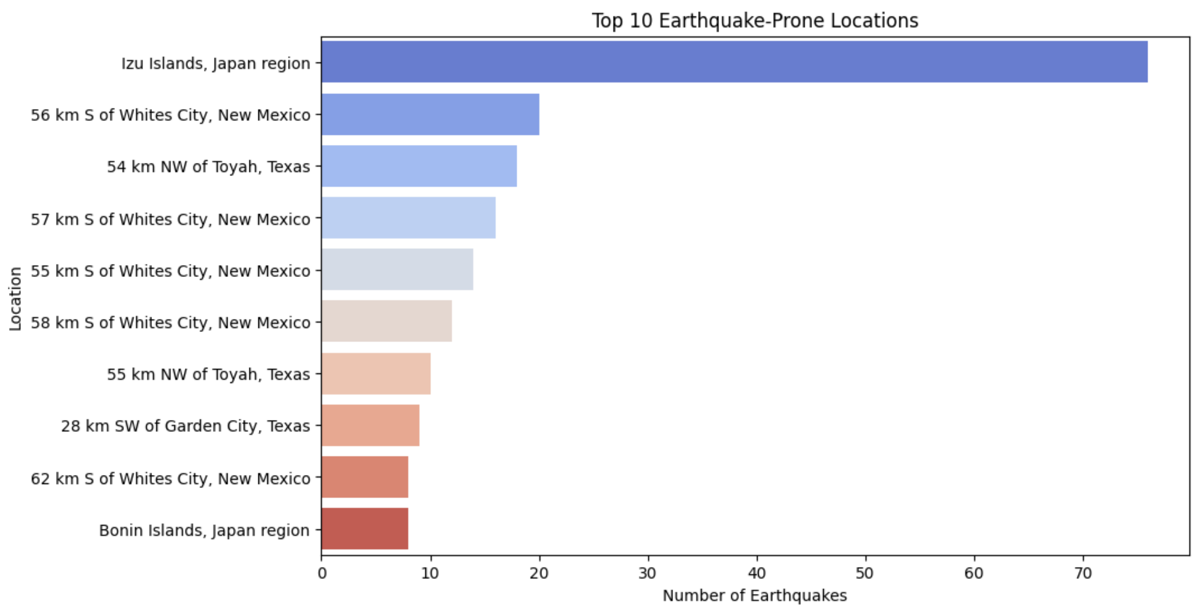
Distribution of Target Variable



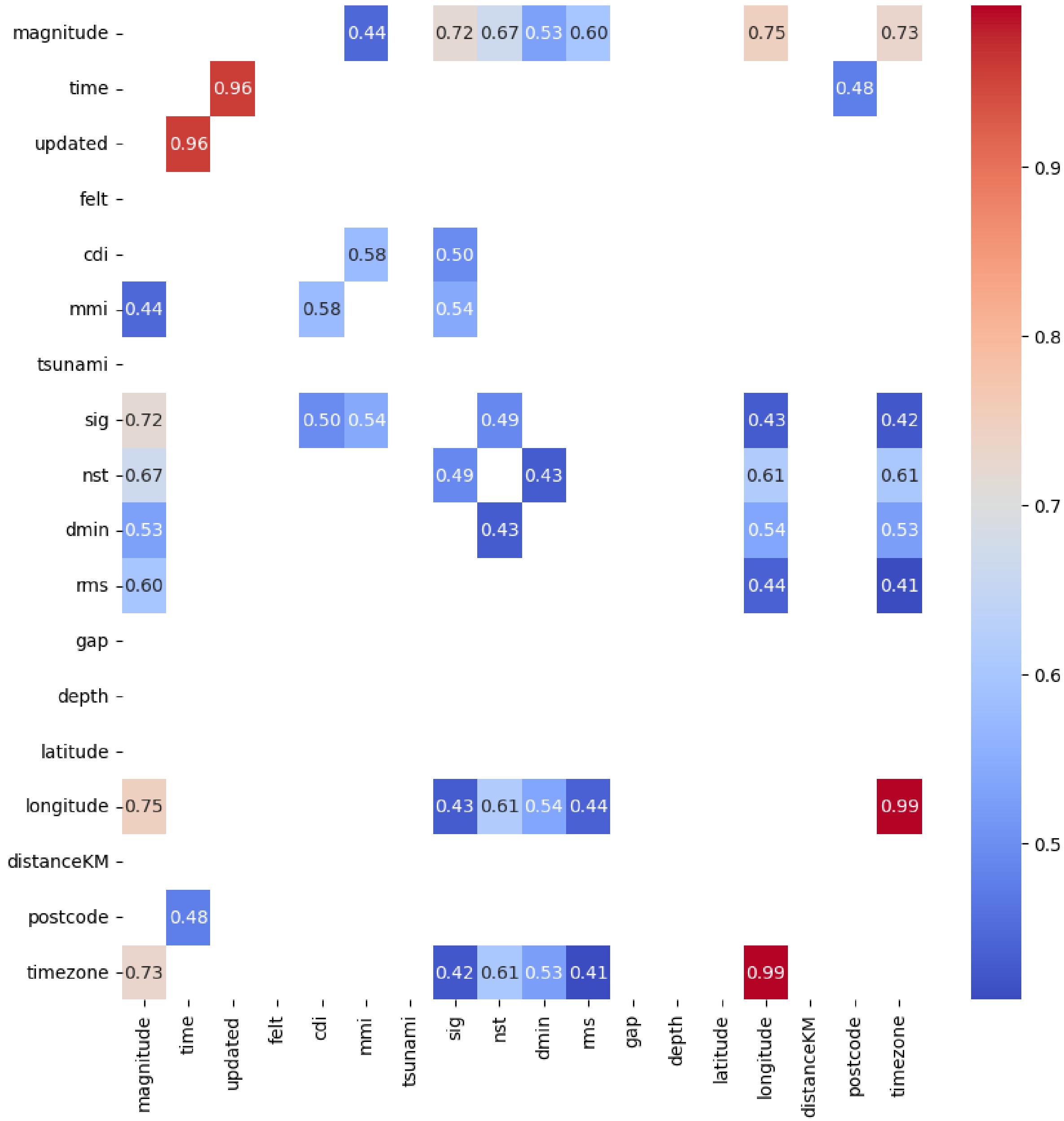
Scatter plot of two features



Top 10 earthquake prone locations by frequency



Correlation Matrix



Threshold using Mean

Mean of Green Alert Magnitude = 5.388537271448664

Mean of Red Alert Magnitude = 6.888888888888889

Mean of Orange Alert Magnitude = 6.3

Mean of Yellow Alert Magnitude = 6.273684210526317

Accuracy and confusion matrix

RMSE: 0.2665577587914667

MSE: 0.07105303877192974

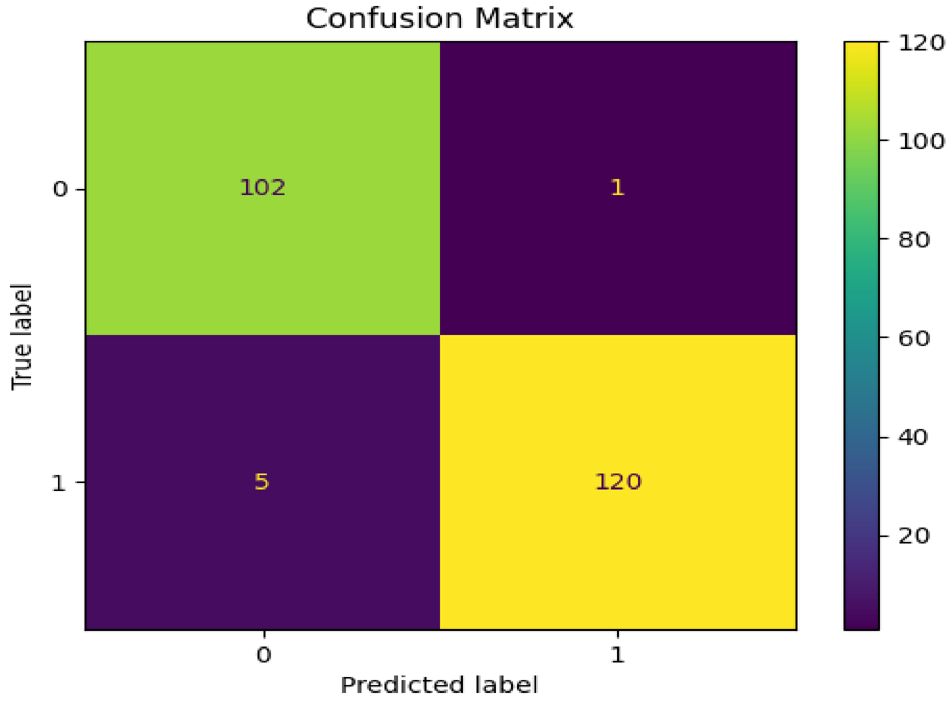
MAE: 0.15059122807017541

R-squared: 0.9341574941340103

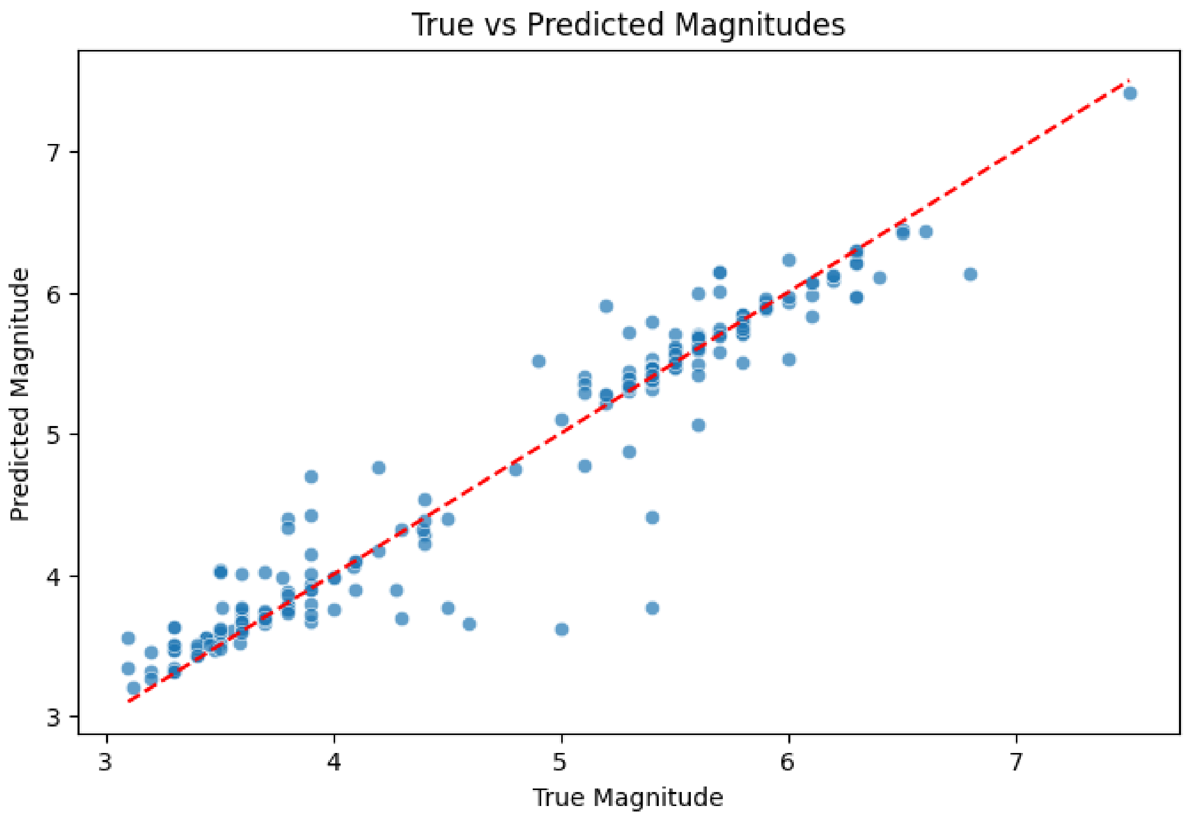
Accuracy: 0.9736842105263158

Classification Report:

	precision	recall	f1-score	support
0	0.95	0.99	0.97	103
1	0.99	0.96	0.98	125
accuracy			0.97	228
macro avg	0.97	0.98	0.97	228
weighted avg	0.97	0.97	0.97	228



Visualize predictions



Github Link

https://github.com/Sneha-sharma-20/GlobalEarthquake_Predictions