
# Earthquake Analysis and Prediction using Neural Networks

Winter Internship Project
TEQIP-III Programme

## Indian Institute of Technology
Indore

Department of Mathematics

Supervisor: Dr. Santanu Manna
Head of Department:

Arjita Basu
Department of Mathematics and Computing
Birla Institute of Technology, Mesra

Monalisha Ojha
Department of Mathematics and Computing
Birla Institute of Technology, Mesra

# Acknowledgement

We have taken great efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to express our deepest appreciation to all of them.

We would also like to extend our heartfelt gratitude to Dr. Santanu Manna for his guidance and constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project. Also, we are highly indebted to TEQIP for giving us the opportunity to do a project at IIT Indore. We would like to express our gratitude towards our parents for their kind co-operation and encouragement which help us in completion of this project.

# Earthquake Analysis and Prediction using Neural Networks

Indian Institute of Technology

**Abstract**

Earthquake Analysis and magnitude prediction has been carried out in this report using the temporal sequence of historic seismic activities in combination with the Neural Network and other Machine Learning Models. The analysis gave us a clear picture of the earthquakes of the selected regions and the best machine learning model for predicting magnitude of earthquake, using the maximum likelihood estimate, had a 75% accuracy.

## 1 Introduction

Natural disasters keep happening from time to time and cause massive casualties, loss of life and property. Such events can't be prevented but timely assessment of conditions and prediction patterns may help prevent loss of human lives. Earthquake is one of the major catastrophes with tsunamis also resulting from underwater earthquakes.

The early history of earthquake predictions were done by studying unusual animal behavior or watching the night skies for strange lights. Today, many respected scientists in seismology and other fields are actively working on this problem. Even when recent seismic studies provide huge amount of new and relevant information, predictions were more often wrong than right. Increased knowledge of the earthquake source has however encouraged seismologists to believe that earthquakes are preceded by phenomena which indicates the coming of an earthquake within hours,days, months or maybe years.

An earthquake is caused by a sudden slip on a fault that creates seismic waves. The tectonic plates are always slowly moving, but they get stuck at their edges due to friction. When the stress on the edge overcomes the friction, there is an earthquake that releases energy in waves that travel through the earth's crust and cause the shaking that we feel. Tectonic earthquakes occur anywhere in the earth where there is sufficient stored elastic strain energy to drive fracture propagation along a fault plane. The spot underground where the rock breaks is called the focus of the earthquake. The place right above the focus (on top of the ground) is called the epicenter of the earthquake.

This project is focused on the proper visual analysis of huge data from ISC catalogues to get a clear picture of magnitude and depth of earthquakes of selected region. We will study the Gutenberg-Richter law and see how well it fits with real seismic events. We will also see the potential of b-value, which describes the relative number of smaller and larger earthquakes in a given area, as an earthquake precursor for both small and large events. We will use some machine learning algorithms for determining the magnitude and depth of an earthquake given the latitude longitude of earthquake.As machine learning is

an application of artificial intelligence that provides systems the ability to automatically learn and improve from experience without being explicitly programmed, the "experience" in this case would be the earthquake data from previous years.

Neural networks will be used for the same. A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal.

It is important that the relationship between seismic activity and geophysical facts is modeled, irrespective of the degree of the non-linearity that exists among them, for timely disaster mitigation in far future.

# 2 Related Work

Earthquake occurrence is considered to be a random or highly nonlinear phenomenon, and there is no such existing model capable of predicting exact time, location and magnitude of earthquake. Researchers have carried out various studies over earthquake occurrences and predictions, which lead to various conclusions regarding the aspects under consideration. Famous Gutenberg and Richter mathematical model (Dahmen et al. 1998) proposed a relationship between earthquake magnitude and frequency of occurrences; this earthquake probability distribution model is useful for structural designing. Petersen et al. (2007) carried out research under the umbrella of California Geological Survey (CGS) and proposed a time-independent model showing that probability of earthquake occurrence follows poison's distribution model.

Panakkat and Adeli (2007) introduced a remarkable approach for earthquake prediction using mathematically calculated seismic indicators from temporal distribution of recorded seismic events for Southern California and San Francisco bay regions. The model makes prediction on monthly basis, and the association between earthquake occurrence and the parameters are modeled using different ANNs. The calculation of these parameters assumed completeness of earthquake catalog, and fixed number of events are used to calculate the seismic parameters before the month under consideration. Following this research, Adeli and Panakkat (2009) used the same seismic parameters in combination with the Probablistic Neural Network (PNN) for earthquake prediction.

float

# 3 Earthquake Study

## 3.1 Gujarat Earthquake

The 2001 Gujarat earthquake, also known as the Bhuj earthquake, occurred on 26 January, India's 52nd Republic Day, at 08:46 am IST and lasted for over 2 minutes. The epicentre was about 9 km south-southwest of the village of Chobari in Bhachau Taluka of Kutch District of Gujarat, India. For the analysis of this region, we took the data of ISC catalogue from 1995 to 2005. We found out the mean magnitude of all seismic activity in this region to be 3.5, maximum magnitude to be 6.9 and maximum depth of origination of a seismic activity was 59.7km.

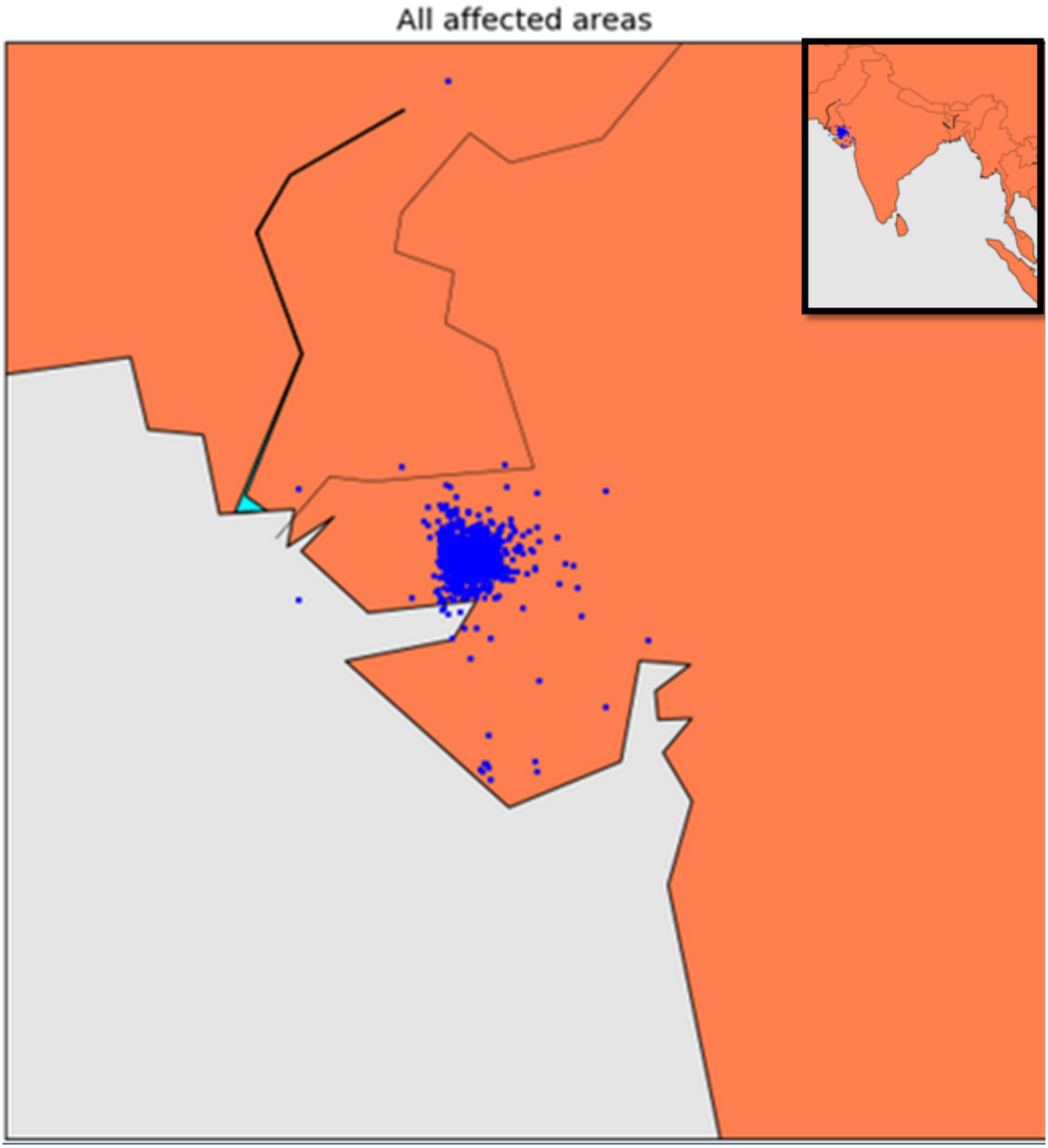



Figure 1: Location map of earthquake (magnitude¿=3)in western part of India (including Gujarat) during 1995-2005

## 3.2 Indian Ocean Earthquake

The 2004 Indian Ocean earthquake and tsunami (also known as the Boxing Day Tsunami) occurred at 00:58:53 GMT on 26 December, with an epicentre off the west coast of northern Sumatra, Indonesia. The earthquake was the third largest ever recorded and had the longest duration of faulting ever observed; between eight and ten minutes. For the analysis of this region, we took ISC catalogue data from 2000 to 2010. We found out the mean magnitude of all seismic activity in this region to be 4.1, maximum magnitude to be 9.1 and maximum depth of origination of a seismic activity was 700km.

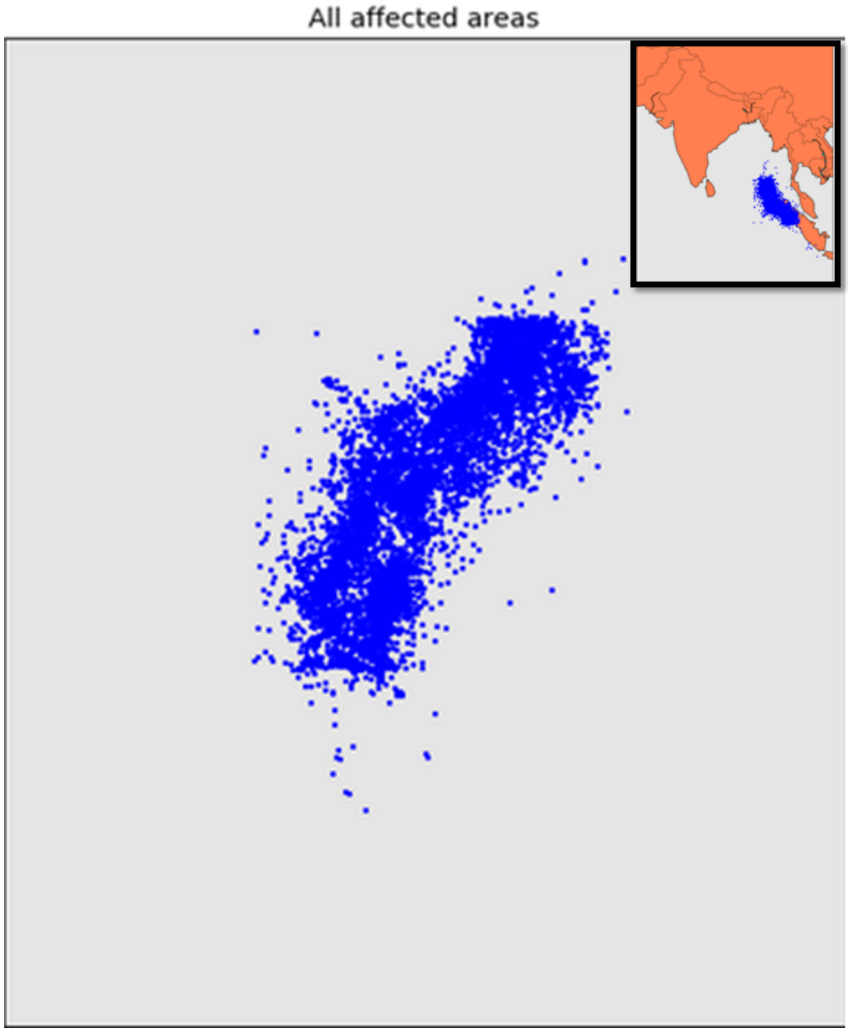



Figure 2: Location map of earthquake (magnitude¿=3)in Indian ocean region (including Andaman and Sumatra) during 2000-2010

```
In [17]: from mpl_toolkits.basemap import Basemap

         m = Basemap(projection='mill',llcrnrlat=20.0,urcrnrlat=-8.0, llcrnrlon=83.0,urcrnrlon=107.0,lat_ts=20,resolution='c'

         longitudes = df["Lon"].tolist()
         latitudes = df["Lat"].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(longitudes,latitudes)

In [18]: 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()
```

Figure 3: Python code to generate above map

## 3.3 Nepal Earthquake

The April 2015 Nepal earthquake (also known as the Gorkha earthquake) killed nearly 9,000 people and injured nearly 22,000. It occurred at 11:56 Nepal Standard Time on 25

April 2015. Its epicenter was east of Gorkha District at Barpak, Gorkha, and its hypocenter was at a depth of approximately 8.2 km.The earthquake triggered an avalanche on Mount Everest. The earthquake triggered another huge avalanche in the Langtang valley. For the analysis of this region, we took the ISC catalogue data from 2010 to 2018. We found out the mean magnitude of all seismic activity in this region to be 3.88, maximum magnitude to be 7.8 and maximum depth of origination of a seismic activity was 100km.

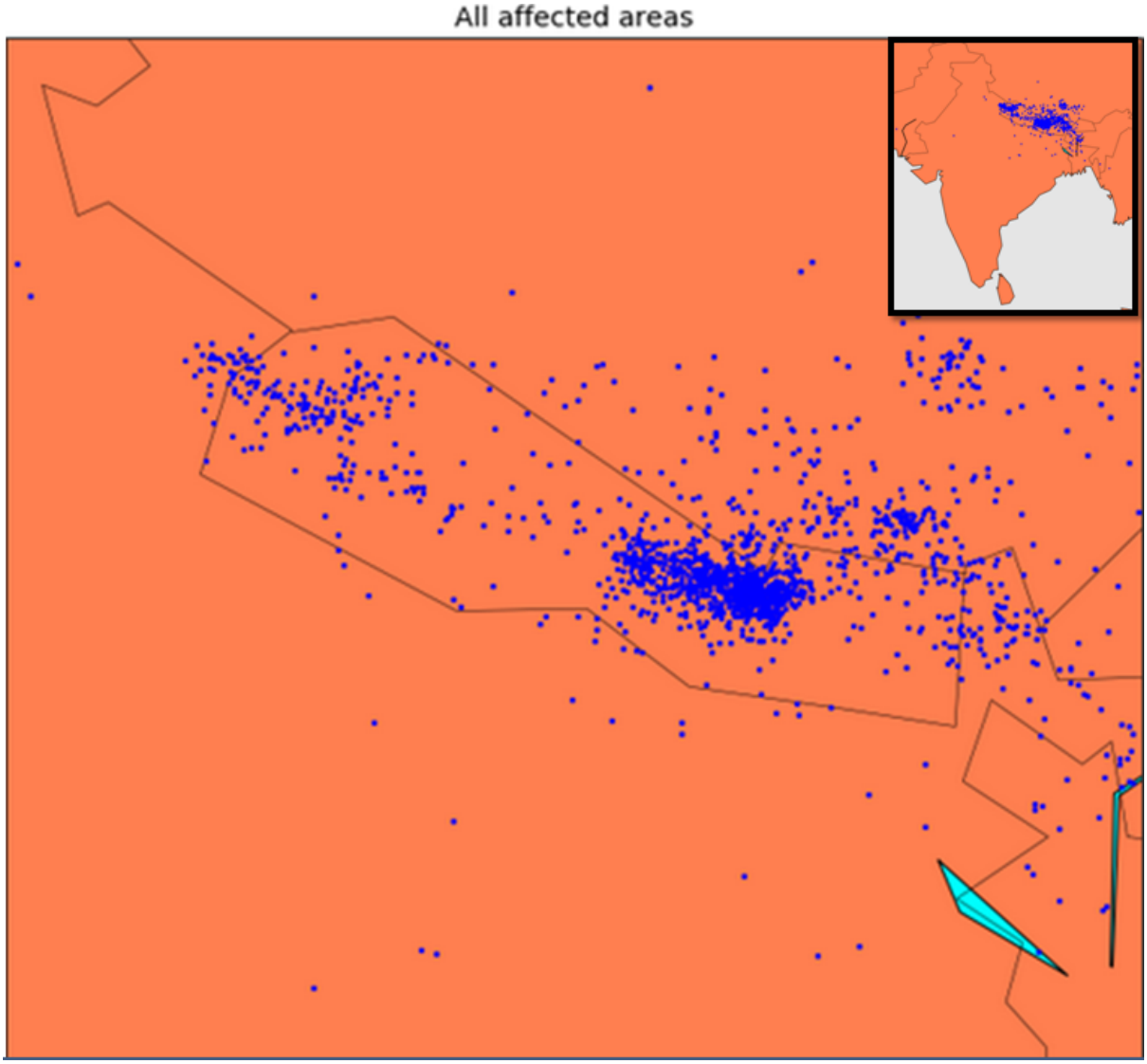



Figure 4: Location map of earthquake (magnitude¿=3)in Nepal and adjoining regions in India during 2010-2019

# 4 Plate Tectonics of the Region

## 4.1 Gujarat

Gujarat lies 300–400 km from the plate boundary between the Indian Plate and the Eurasian Plate, but the current tectonics are still governed by the effects of the continuing continental collision along this boundary. During the break-up of Gondwana in the Jurassic, this area was affected by rifting with a roughly west–east trend. During the

collision with Eurasia the area has undergone shortening, involving both reactivation of the original rift faults and development of new low-angle thrust faults. The related folding has formed a series of ranges, particularly in central Kutch. The 2001 Gujarat earthquake was caused by movement on a previously unknown south-dipping fault, trending parallel to the inferred rift structures.

## 4.2 Indian Ocean

The 2004 Indian Ocean earthquake was unusually large in geographical and geological extent. An estimated 1,600 kilometers of fault surface slipped about 15 meters along the subduction zone where the Indian Plate slides under the overriding Burma Plate. The slip did not happen instantaneously but took place in two phases over several minutes: Seismographic and acoustic data indicate that the first phase involved a rupture about 400 kilometers long and 100 km wide, 30 km beneath the sea bed—the largest rupture ever known to have been caused by an earthquake. The rupture proceeded at about 2.8 kilometers per second, beginning off the coast of Aceh and proceeding north-westerly over about 100 seconds. After a pause of about another 100 seconds, the rupture continued northwards towards the Andaman and Nicobar Islands. The northern rupture occurred more slowly than in the south, at about 2.1 km/s, continuing north for another five minutes to a plate boundary where the fault type changes from subduction to strike-slip (the two plates slide past one another in opposite directions). The Indian Plate is part of the great Indo-Australian Plate, which underlies the Indian Ocean and Bay of Bengal, and is moving north-east at an average of 60 millimeters per year. The India Plate meets the Burma Plate (which is considered a portion of the great Eurasian Plate) at the Sunda Trench. At this point the India Plate subducts beneath the Burma Plate, which carries the Nicobar Islands, the Andaman Islands, and northern Sumatra. As well as the sideways movement between the plates, the 2004 Indian Ocean earthquake resulted in a rise of the sea floor by several metres, displacing an estimated 30 cubic kilometers of water and triggering devastating tsunami waves.

## 4.3 Nepal

The earthquake and its aftershocks were the result of thrust faulting (i.e., compression-driven fracturing) in the Indus-Yarlung suture zone, a thin east-west region spanning roughly the length of the Himalayan ranges. The earthquake relieved compressional pressure between the Eurasian tectonic plate and the Indian section of the Indo-Australian Plate, which subducts (underthrusts) the Eurasian Plate. Subduction in the Himalayas occurs at an average rate of 4–5 cm annually. Such tectonic activity adds more than 1 cm to the height of the Himalayan mountains every year.

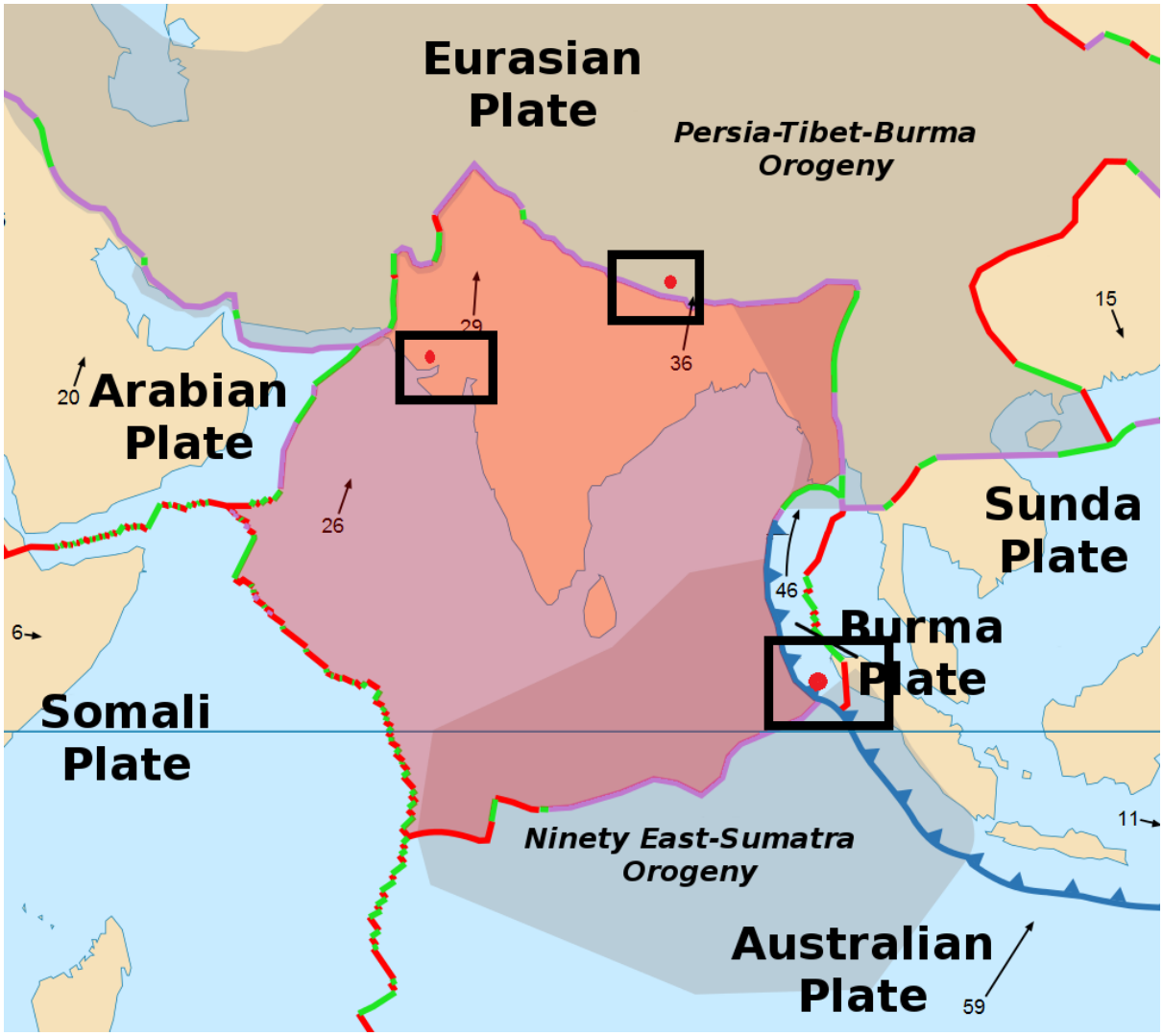



Figure 5: Plate tectonics of the Indian subcontinent
The regions marked in black boxes are the regions which will be under observation in this report. Also, the red points indicate the epicenters of the Gujarat 2001, Indian ocean 2004 and Nepal 2015 major earthquakes.

float

# 5 Comparative Studies of the Above Three Earthquakes

Now let us do some comparative visual data analysis of the available earthquake data. The following are the plots of the date vs. magnitude of seismic activity of the three regions.

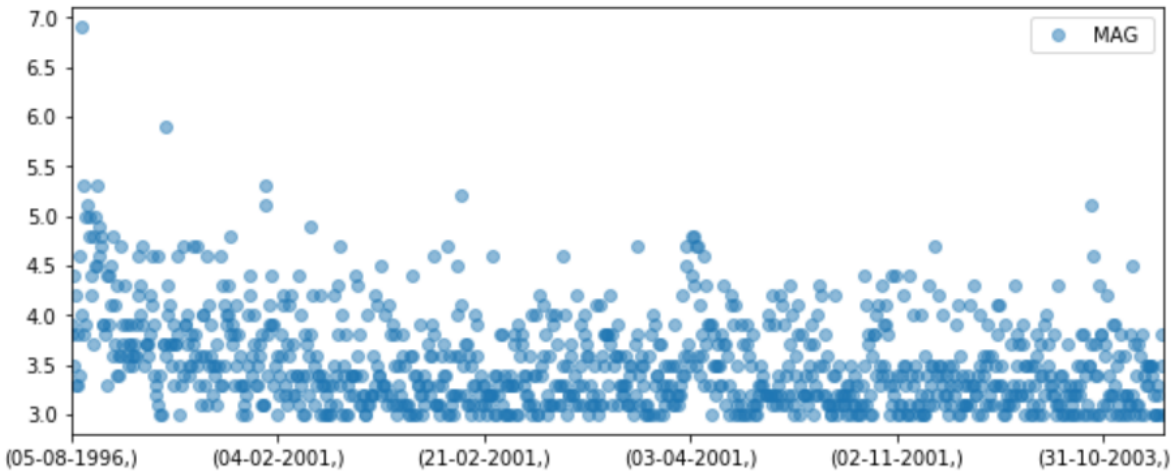



Figure 6: Gujarat Date vs Mag plot
From the figure we can observe that most of the activity range from 3 to 4.5 and very less seismic activities happen above that.

```
In [11]:  import numpy as np
          cumulative_df = df1.apply(np.cumsum).copy()

In [13]:  cumulative_df.iloc[:, [7]].plot(figsize=(10,4))

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0xeb38af0208>
```

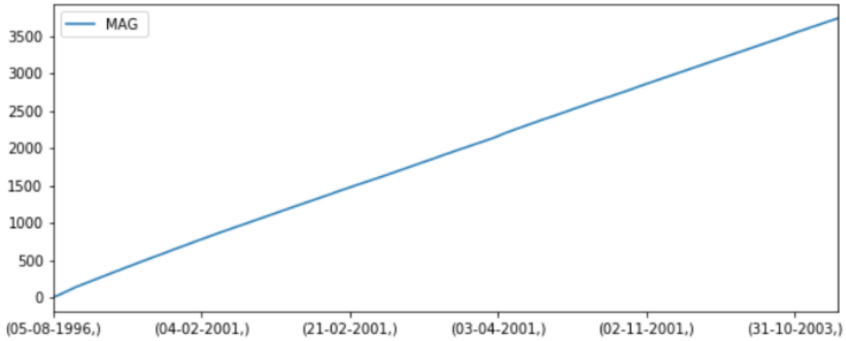



Figure 7: Gujarat Date vs Mag Cumulative plot

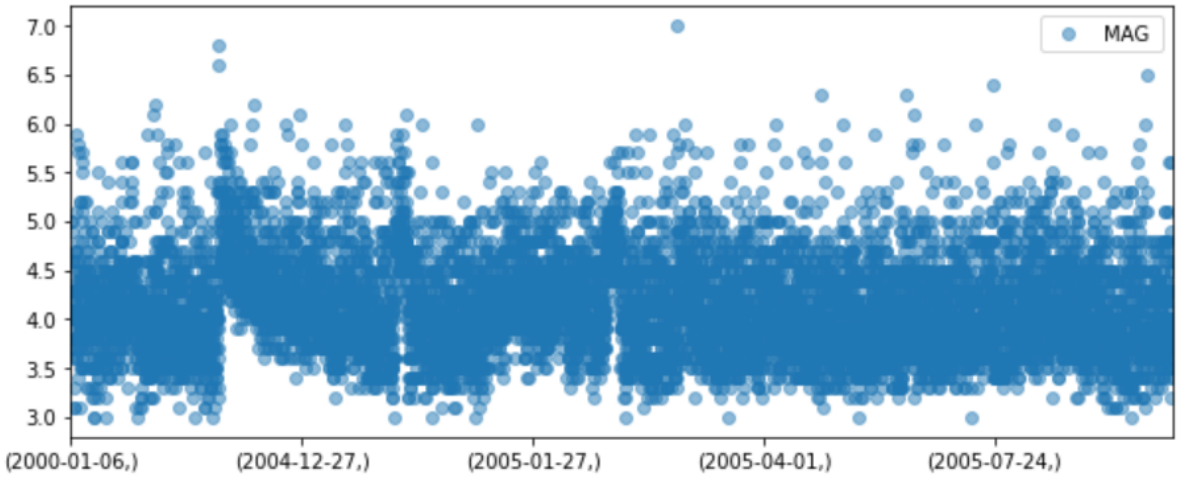



Figure 8: Indian Ocean Date vs Mag plot

From the figure we can observe that many seismic activities have taken place in the range of 3 to 5.5. This plot happens to be the most dense among the three, indicating that it happens to be the most earthquake prone zone among the three.

```
In [10]:  import numpy as np
          cumulative_df = df1.apply(np.cumsum).copy()

In [12]:  cumulative_df.iloc[:, [7]].plot(figsize=(10,4))

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xab9acec898>
```

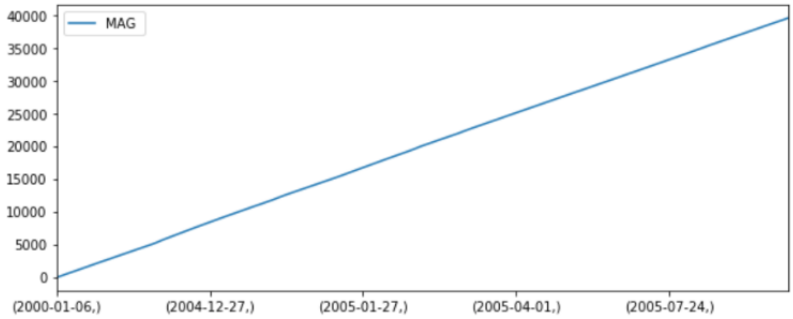



Figure 9: Indian Ocean Date vs Mag Cumulative plot

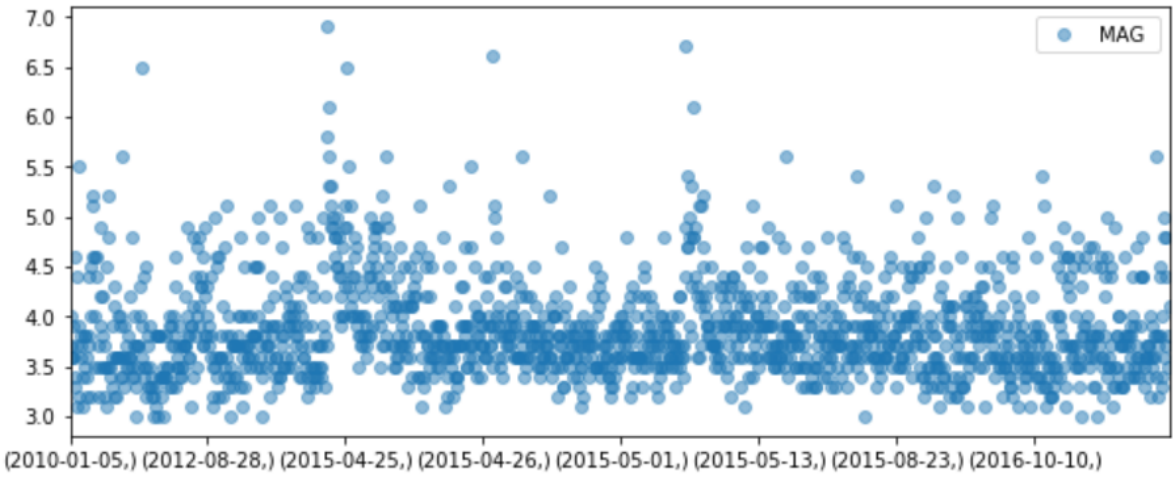



Figure 10: Nepal Date vs Mag plot
From the figure we can observe that most of the activity range from 3 to 4 and quite a few seismic activities have happened above that.

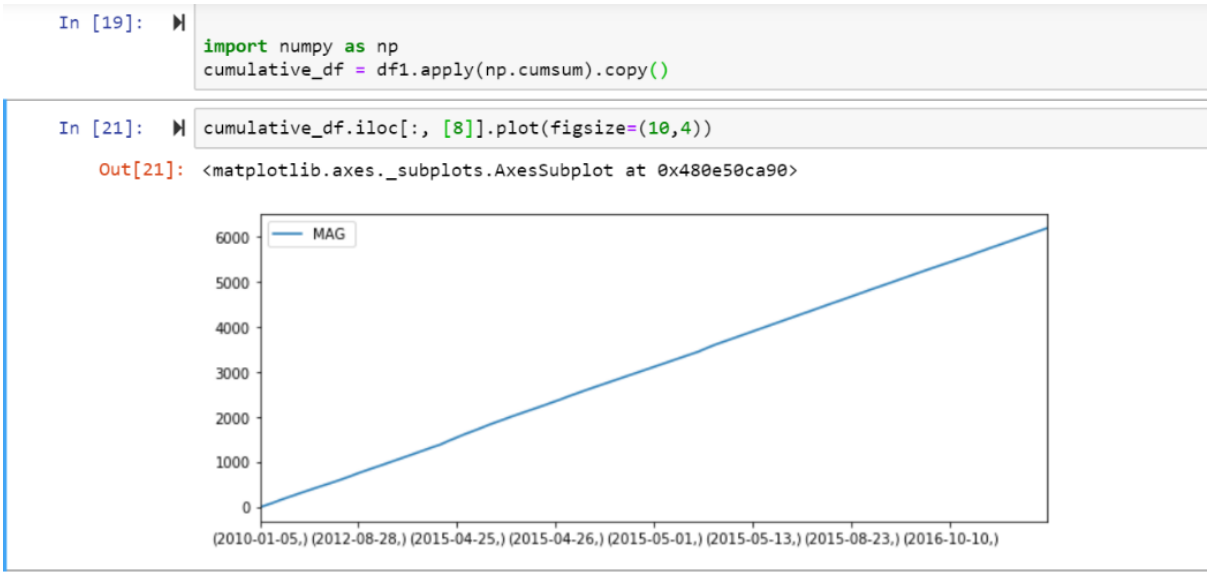



Figure 11: Nepal Date vs Mag Cumulative plot

The following are the plots of the date vs. depth of seismic activity of the three regions.

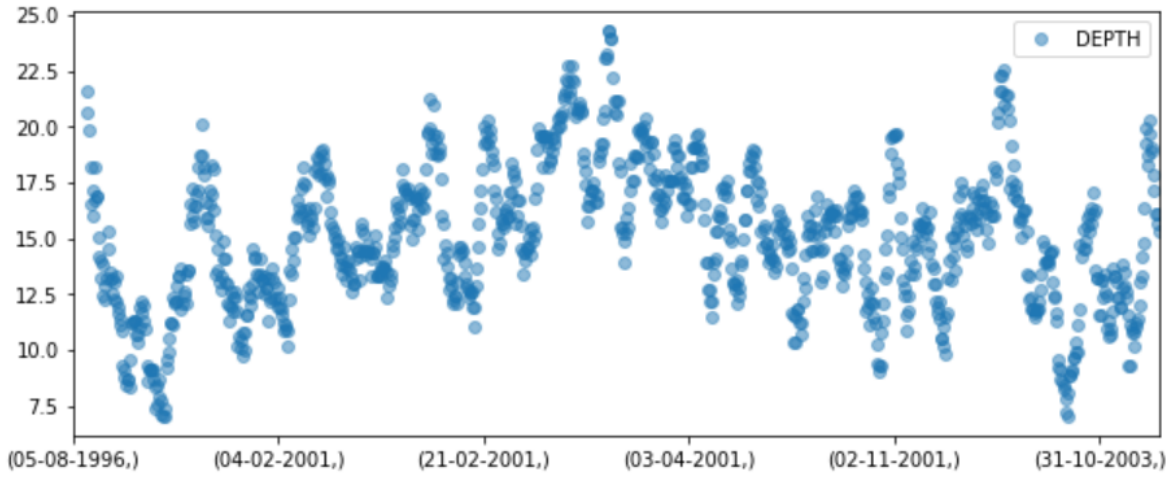



Figure 12: Gujarat Date vs Depth plot
From the figure we can observe that depth of seismic activities range from around 10 to 20kms.

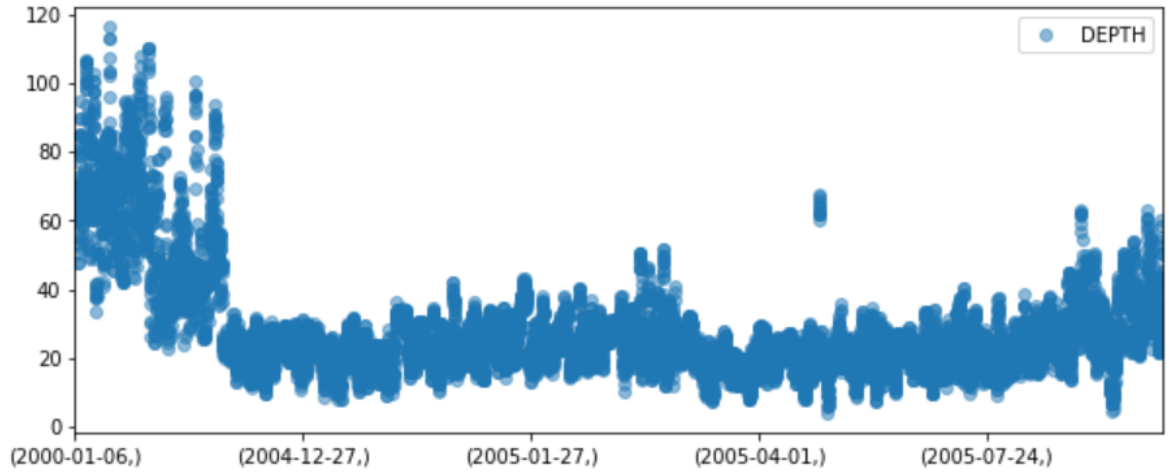



Figure 13: Indian Ocean Date vs Mag plot
From the figure we can observe that earlier the depth of seismic activities ranged from 20 to 120 kms and later it went to around 10 to 40 kms.

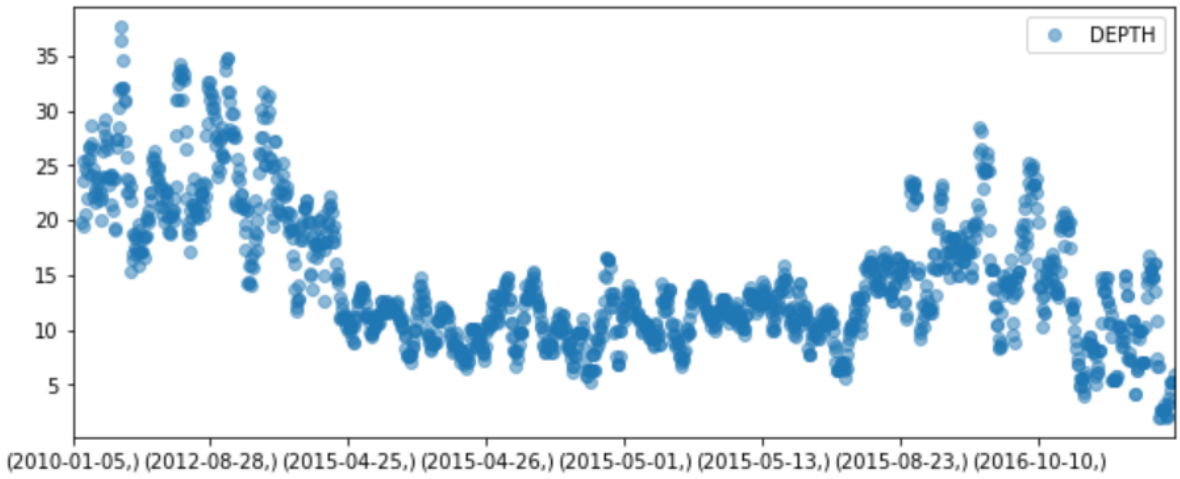



Figure 14: Nepal Date vs Mag plot
From the figure we can observe that depth of seismic activities ranged from 15 to 35 kms had a dip in between from 5 to 20 kms and again increased from 10 to 30 kms.

# 6 Gutenberg Richter Law

In seismology, the Gutenberg–Richter law expresses the relationship between the magnitude and total number of earthquakes in any given region and time period of at least that magnitude. The formula is as given below.

$$\log_{10} N = a - bM \quad \text{or} \quad N = 10^{a-bM}$$

Figure 15: Gutenberg Richter Relation

where N is the number of events having a magnitude ¿=M and a and b are constants. This relationship between event magnitude and frequency of occurrence is remarkably common, although the values of a and b may vary significantly from region to region or over time. The parameter b (commonly referred to as the "b-value") is commonly close to 1.0 in seismically active regions. This means that for a given frequency of magnitude 7.0 or larger events there will be 10 times as many magnitude 6.0 or larger quakes and 100 times as many magnitude 5.0 or larger quakes.

Now we will compare the Gutenberg Richter curves of the three regions.

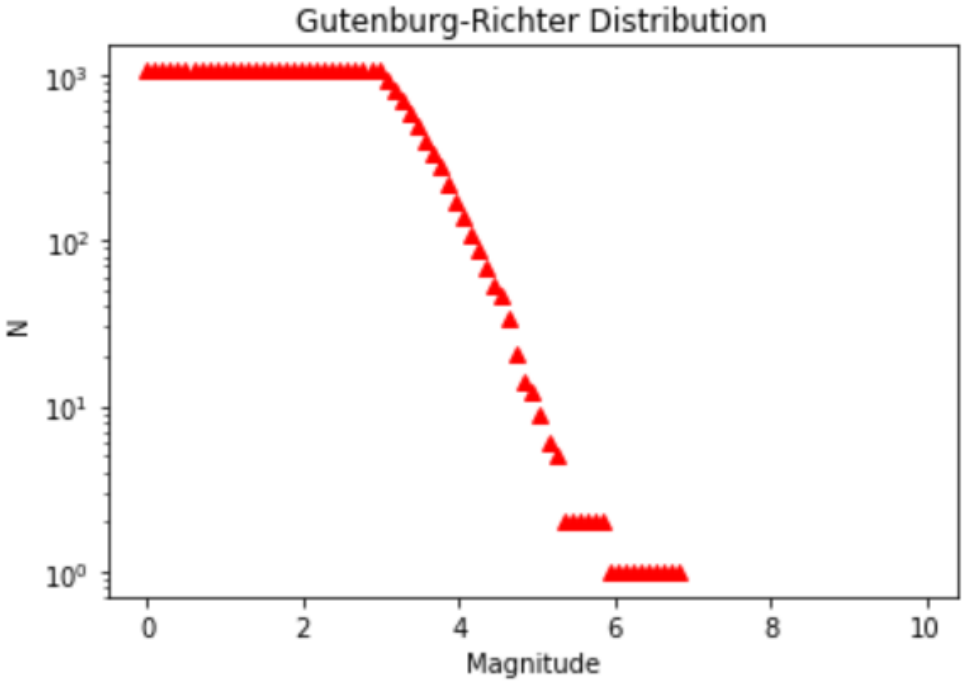



Figure 16: M vs N of Gujarat

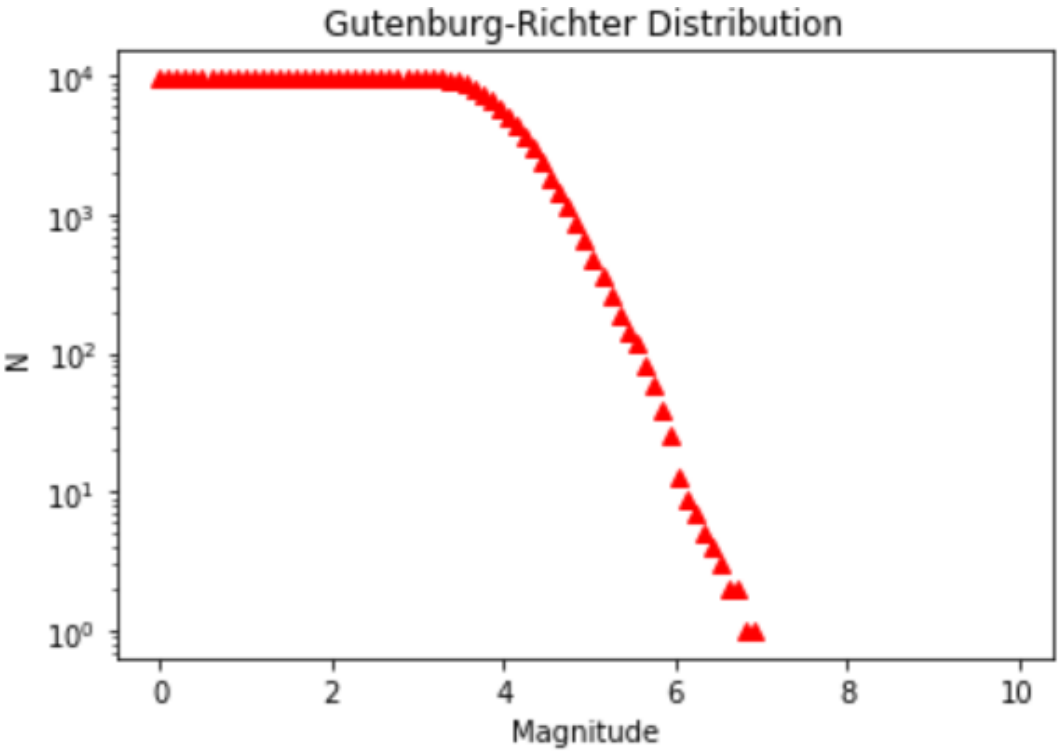



Figure 17: M vs N of Indian Ocean

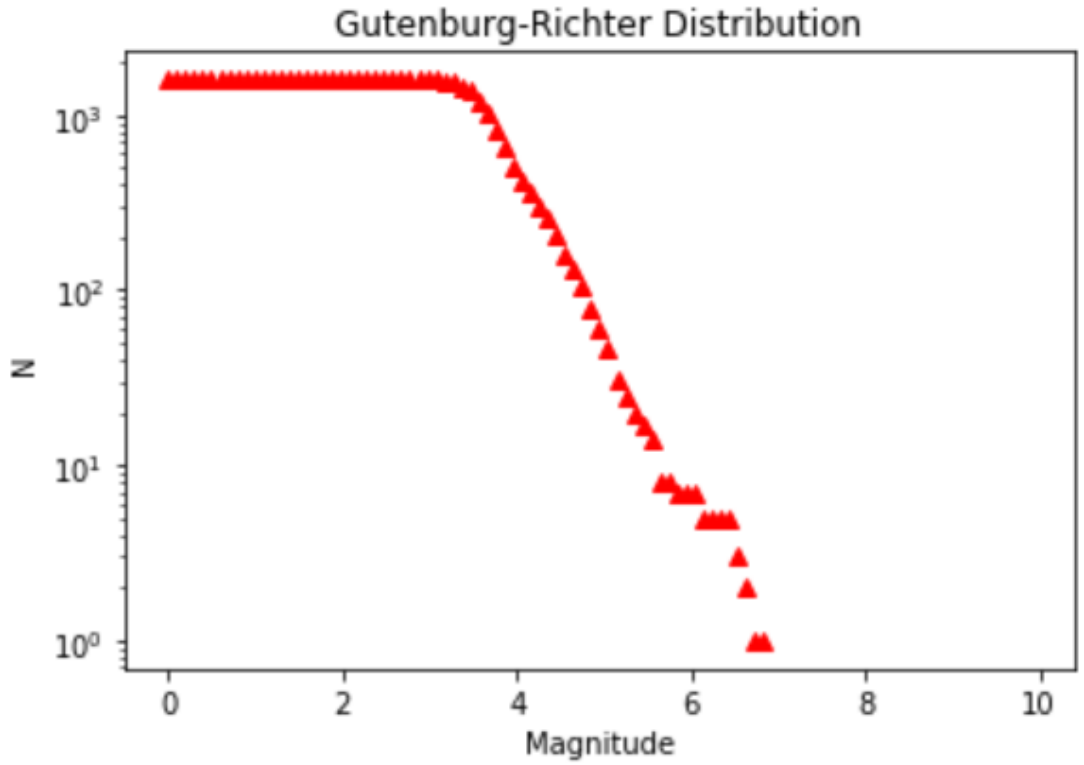
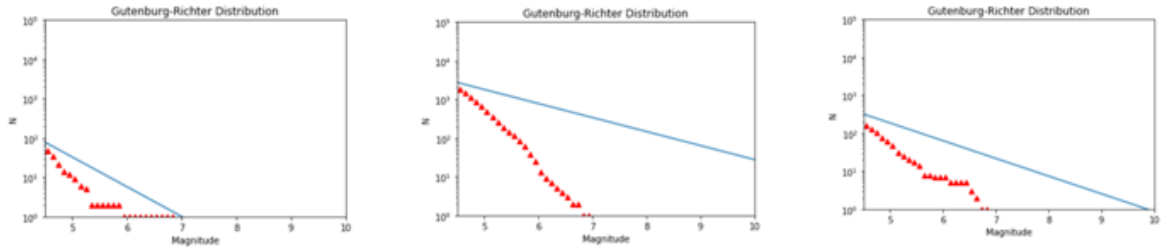



Figure 18: M vs N of Nepal

Below is the Gutenberg-Richter distribution curve derived from actual values in red and what should have been according to the Gutenberg Richter formula in blue.




# 7 Methods of determining b-values

b-values of earthquakes have been measured using linear least square fit and maximum likelihood estimator. Linear least squares is the least squares approximation of linear functions to data. It is a set of formulations for solving statistical problems involved in linear regression. In statistics, maximum likelihood estimation is a method of estimating the parameters of a probability distribution by maximizing a likelihood function, so that under the assumed statistical model the observed data is most probable. In this report, we are going to take the maximum likelihood estimate approach.

## 7.1 Maximum Likelihood Estimate

The maximum likelihood method has been suggested as preferable for calculating the b value because it yields a more robust estimate when the number of the infrequent large

earthquakes changes. There will be cases, however, such as estimating the probability of the largest magnitude of earthquakes, where the least-squares method is more suitable.

$$\beta = \frac{1}{\overline{m} - m_{\min}},$$

$$\beta = b \ln(10).$$

Figure 19: The above formulae are used to calculate the b-value using Maximum Likelihood estimate

```
In [201]:
def fmd_values(magnitudes, bin_width=0.1):
    """
    params magnitudes : numpy.array
    params bin_width : float

    returns a,b,bstd, n-values if above the earthquake count threshold
    else returns np.nans
    """
    length = magnitudes.shape[0]
    minimum = magnitudes.min()
    average = magnitudes.mean()
    b_value = (1 / (average - (minimum - (bin_width/2)))) * np.log10(np.exp(1))
    square_every_value = np.vectorize(lambda x: x**2)
    b_stddev = square_every_value((magnitudes - average).sum()) / (length * (length - 1))
    b_stddev = 2.3 * np.sqrt(b_stddev) * b_value**2
    a_value = np.log10(length) + b_value * minimum

    return a_value, b_value, b_stddev, length

In [202]: fmd_values(df1.iloc[:, [8]].values)
Out[202]: (4.602455445297751, 0.4664741860943467, 2.044949354866672e-16, 1596)
```

Figure 20: The above code calculates the b-value using Maximum Likelihood estimate

# 8 Machine learning techniques for earthquake prediction

float It is well known that if a disaster has happened in a region, it is likely to happen there again. Some regions really have frequent earthquakes, but this is just a comparative quantity compared to other regions. So, predicting the earthquake with Date and Time, Latitude and Longitude from previous data is not a trend which follows like other things, it is natural occuring.

In this report, three machine learning techniques including neural network, random forest and KNN(K-Nearest Neighbors)Regression are separately applied to model relationships between different seismic parameters and future earthquake occurrences.Accuracy is major performance measure considered for analyzing the results. Earthquake magnitude prediction using these aforementioned techniques show significant and encouraging results, thus constituting a step forward toward the final robust prediction mechanism which is not available so far.

## 8.1 Earthquake Catalogue

The main source of earthquake catalogue, International Seismological Centre.The International Seismological Centre (ISC) was set up in 1964 with the assistance of UNESCO as a successor to the International Seismological Summary (ISS) to follow up the pioneering work of Prof. John Milne and Sir Harold Jeffreys in collecting, archiving and processing seismic station and network bulletins and preparing and distributing the definitive summary of world seismicity. In the following figure, exponential rise in the occurrence of earthquakes with decreasing magnitudes shows that it follows Gutenberg–Richter's relationship, hence the catalogue is complete from magnitude 4.0 and onward. Therefore for mathematical parameters calculation, seismic events of magnitude greater than and equal to 4.0 are considered in this study. There are a total of 1596 seismic events recorded from January,2010 to December, 2019 in Nepal region and all of these are considered for this study. In this research,analysis is carried out yearly basis and seismic parameters are calculated for every year.

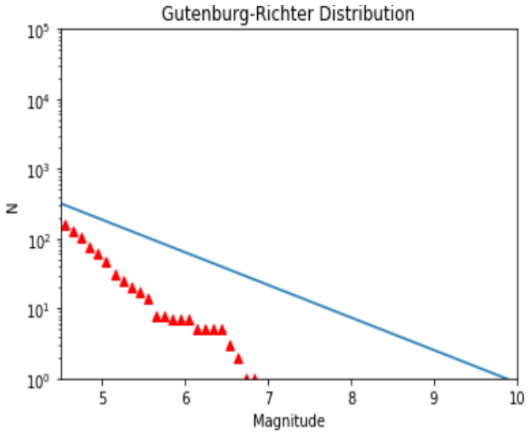



Figure 21: Gutenberg-Ritcher Relationship of the dataset
From the figure we can observe the Gutenberg-Ritcher Relationship of the Nepal Earthquake Database from ISC

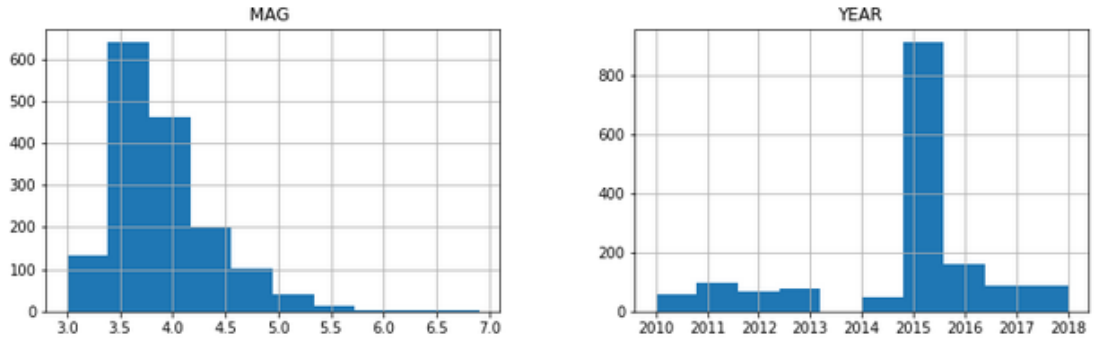



Figure 22: Distribution of Earthquake
From the above picture we can infer that most earthquakes have happened (above 600) of magnitude 3.5 to 4. Also most earthquakes have taken place in the year 2015.

## 8.2 Data Preprocessing

In this paper, we have used Nepal Earthquake data of 9 year from 2010 to 2019.The data is consisted of 1596 observations and 9 features.For the initial preprocessing , we have inspected each feature of the dataset to 1) remove features with frequent and irreperable missing fields or set the missing values to zero where appropriate ,2) remove irrelevant or uninformative features or duplicate features .We have split the data into train , validation , and test sets.Consequently , several feature selection techniques were used to find the features with the most predictive values to both reduce the model variances and reduce the computation time.

```
import tensorflow as tf
import matplotlib.pyplot as plt
from mpl_toolkits.basemap import Basemap
import numpy as np
import pandas as pd
import matplotlib
import sklearn
from sklearn_pandas import DataFrameMapper
from functools import partial
matplotlib.style.use('ggplot')
%matplotlib inline
```

Figure 23: Importing important libraries

```
df = pd.read_csv(r'C:\Users\User\Downloads\nepal_earthquake.csv')
df
```

| | EVENTID | AUTHOR | YEAR | DATE | TIME | LAT | LON | DEPTH | DEPFIX | AUTHOR.1 | ... | Unnamed: 92 | Unnamed: 93 | Unnamed: 94 | Unnamed: 95 | Unnam |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 15622976 | ISC | 2010 | 2010-01-05 | 04:25.8 | 30.0907 | 80.1698 | 4.5 | | ISC | ... | NaN | NaN | NaN | NaN | N |
| 1 | 15623393 | ISC | 2010 | 2010-01-11 | 15:13.5 | 29.7659 | 80.4999 | 13.7 | | ISC | ... | NaN | NaN | NaN | NaN | N |
| 2 | 17146398 | IDC | 2010 | 2010-01-16 | 58:02.5 | 28.7977 | 81.9623 | 0.0 | TRUE | IDC | ... | NaN | NaN | NaN | NaN | N |

Figure 24: Importing the nepal earthquake dataset

```
X = df[['Timestamp','Lat', 'Lon']]
y = df[['Mag', 'Depth km']]
```

Figure 25: Spliting the dataset into test and train

```
from sklearn.model_selection 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)
```

```
(1275, 3) (319, 3) (1275, 2) (319, 3)
```

Figure 26: Spliting the dataset into test and train

# 9 Methods

Neural Network consists of three Dense layer with each 16, 16, 2 nodes and relu, relu and softmax as activation function.Deep Feedforward Network(Neural Network) was set as a baseline model on the dataset using all of the features as model inputs. After selecting a set of features, Magnitude and Depth, several machine learning models were considered in order to find the optimal one. All of the models were implemented using scikit-learn and Keras library.

## 9.1 Deep feedforward networks

Deep feedforward networks, also called feedforward neural networks, or multilayer perceptrons(MLPs), are the quintessential deep learning models.The goal of a feed forward network is to approximate some function

$$f * (x)$$

For example,for a classier,

$$y = f(x)$$

maps an input x to a category. A feedforward network denes a mapping

$$y = f(x, t)$$

and learns the value of the parameters that result in the best function approximation. Feedforward neural networks are called networks because they are typically represented by composing together many dierent functions. The model is associated with a directed a cyclic graph describing how the functions are composed together. For example, we might have three functions

$$f(1)$$

$$f(2)$$

, and

$$f(3)$$

connected in a chain, to form

$$f(x) = f(1) f(2) f(3)$$

. These chain structures are the most commonly used structures of neural networks. In this case,

$$f(1)$$

is called the rst layer of the network,

$$f(2)$$

is called the second layer, and so on. The overall length of the chain gives the depth of the model. The name "deep learning"arose from this terminology. The nal layer of a feedforward network is called the output layer. During neural network training, we drive

$$f(x)$$

to match

$$f*(x)$$

.The training data provides us with noisy, approximate examples of

$$f(x,t)$$

evaluate dierent training points. Each example x is accompanied by a label

$$y = f(x)$$

.The training examples specify directly what the output layer must do at each point x; it must produce a value that is close toy. The behavior of the other layers is not directly specied by the training data. The learning algorithm must decide how to use those layers to produce the desired output, but the training data do not say what each individual layer should do. Instead, the learning algorithm must decide how to use these layers to best implement an approximation of

$$f*(x)$$

. Because the training data does not show the desired output for each of these layers, they are called hidden layers.

## 9.2 ReLU Activation

'ReLU' is an activation function that captures non-linearity in the output of another function. Mathematically, it is defined as So, it always returns the positive value. We

$$f(x) = max(0, x)$$

Figure 27: RELU Function

can say, it is a 'positive filter'

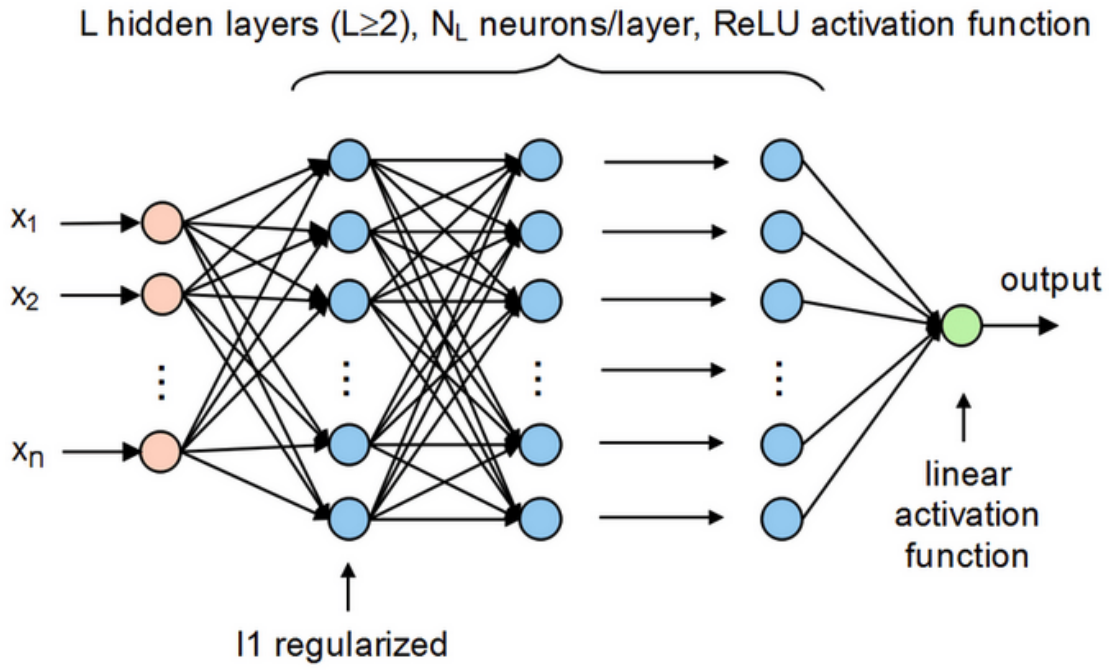



Figure 28: Architecture of Deep Feedforward Network

```
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
```

Figure 29: Neural Network consists of three Dense layer with each 16, 16, 2 nodes and relu, relu and softmax as activation function.

```
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', 'Nadam']
optimizer = ['SGD', 'Adadelta']
loss = ['squared_hinge']

param_grid = dict(neurons=neurons, batch_size=batch_size, epochs=epochs, activation=activation, optimizer=optimizer,
```

Figure 30: we define the hyperparameters with two or more options to find the best fit.

```
from sklearn.model_selection import GridSearchCV

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']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
    print("%f (%f) with: %r" % (mean, stdev, param))
```

```
/home/monalisha/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_split.py:1978: FutureWarning: The de
fault value of cv will change from 3 to 5 in version 0.22. Specify it explicitly to silence this warning.
  warnings.warn(CV_WARNING, FutureWarning)
```

```
Best: 0.861176 using {'activation': 'relu', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons': 16,
'optimizer': 'SGD'}
0.000000 (0.000000) with: {'activation': 'sigmoid', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neuro
ns': 16, 'optimizer': 'SGD'}
0.000784 (0.001109) with: {'activation': 'sigmoid', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neuro
ns': 16, 'optimizer': 'Adadelta'}
0.861176 (0.123807) with: {'activation': 'relu', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons
': 16, 'optimizer': 'SGD'}
0.386667 (0.306186) with: {'activation': 'relu', 'batch_size': 10, 'epochs': 10, 'loss': 'squared_hinge', 'neurons
': 16, 'optimizer': 'Adadelta'}
```

Figure 31: we find the best fit of the above model and get the mean test score and standard deviation of the best fit model.

```
model = Sequential()
model.add(Dense(16, activation='relu', input_shape=(3,)))
model.add(Dense(16, activation='relu'))
model.add(Dense(2, activation='softmax'))

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

```
model.fit(X_train, y_train, batch_size=10, epochs=20, verbose=1, validation_data=(X_test, y_test))
```

Figure 32: The best fit parameters are used for same model to compute the score with training data and testing data.

## 9.3 Random Forest Regression

Here, we used the Random Forest Regressor model to predict the outputs.Random Forest is far more flexible than a Linear Regression model. This means lower bias, and it can fit the data better. Complex models can often memorize the underlying data and hence will not generalize well. Parameter tuning is used to avoid this problem.

```
In [85]: from sklearn.ensemble import RandomForestRegressor

         reg = RandomForestRegressor(random_state=42)
         reg.fit(X_train, y_train)
         reg.predict(X_test)
Out[85]: array([[ 3.76, 12.9 ],
                [ 3.7 , 20.46],
                [ 3.97, 27.92],
                [ 4.  , 17.25],
                [ 3.9 ,  5.91],
                [ 3.82, 10.07],
                [ 3.57,  6.  ],
                [ 3.88, 14.62],
                [ 4.03, 16.34],
                [ 3.83, 27.46],
                [ 3.81,  8.84],
                [ 3.73, 14.34],
                [ 4.16, 15.04],
                [ 4.1 , 15.65],
                [ 3.92,  8.34],
                [ 3.93, 17.46],
                [ 3.79, 12.  ],
                [ 3.88, 33.62],
                [ 3.85,  8.29],
                [ 3.71,  1.38],
```

Figure 33: Random Forest Regressor model to predict the outputs

```
In [86]: reg.score(X_test, y_test)

/home/monalisha/anaconda3/lib/python3.7/site-packages/sklearn/base.py:420: FutureWarning: The default value of mult
ioutput (not exposed in score method) will change from 'variance_weighted' to 'uniform_average' in 0.23 to keep con
sistent with 'metrics.r2_score'. To specify the default value manually and avoid the warning, please either call 'm
etrics.r2_score' directly or make a custom scorer with 'metrics.make_scorer' (the built-in scorer 'r2' uses multiou
tput='uniform_average').
  "multioutput='uniform_average').", FutureWarning)

Out[86]: 0.13587407173150406
```

Figure 34: R2 score of the model

### 9.3.1 KNN Regression

KNN(K-Nearest Neighbours) can be used for both classification and regression problems. The algorithm uses 'feature similarity' to predict values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.

```
In [78]: from sklearn.preprocessing import StandardScaler
         sc = StandardScaler()
         X_train = sc.fit_transform(X_train)
         X_test = sc.transform(X_test)

In [47]: from sklearn.neighbors import KNeighborsRegressor

         knn = KNeighborsRegressor(5)
         knn.fit(X_train ,y_train )

Out[47]: KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
                             metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                             weights='uniform')
```

Figure 35: KNN model to predict the magnitude


Figure 36: KNN model to predict the magnitude

# 10 Discussion and Results

It shows Test accuracy of 73 percent. We see that the above model performs better but it also has lot of noise (loss) which can be neglected for prediction and use it for furthur prediction.

```
Train on 1275 samples, validate on 319 samples
Epoch 1/20
1275/1275 [==============================] - 1s 423us/step - loss: 0.6161 - acc: 0.7592 - val_loss: 0.6349 - val_a
cc: 0.7179
Epoch 2/20
1275/1275 [==============================] - 0s 209us/step - loss: 0.6161 - acc: 0.7592 - val_loss: 0.6349 - val_a
cc: 0.7179
Epoch 3/20
1275/1275 [==============================] - 0s 198us/step - loss: 0.6161 - acc: 0.7592 - val_loss: 0.6349 - val_a
cc: 0.7179
Epoch 4/20
1275/1275 [==============================] - 0s 203us/step - loss: 0.6161 - acc: 0.7592 - val_loss: 0.6349 - val_a
cc: 0.7179
Epoch 5/20
1275/1275 [==============================] - 0s 197us/step - loss: 0.6161 - acc: 0.7592 - val loss: 0.6349 - val a
```

Figure 37: 75 percent accuracy

The above model is saved for furthur prediction.

Considering what is and what is not accounted for in the models built in this study, their predicting results are fairly accurate. To further improve the prediction accuracy, more variabilities need to be considered and modeled. This project attempts to come up with the best model for predicting the magnitude and Depth based on a set of features including locations, longitudes and latitudes, Time and Date. Machine learning techniques including Random Forest Regression, k nearest neighbors and Neural Networks along with feature importance analyses are employed to achieve the best results in terms of Accuracy and R2 score.The initial experimentation with the baseline model proved that the abundance of features leads to high variance and weak performance of the model on the validation set compared to the training set. This level of accuracy is a promising outcome given the heterogeneity of the dataset and the involved hidden factors , which were impossible to consider.

We have identified a couple of area where we can make improvements. Currently there are some steps that we need to perform manually. The future works on this project can include (i) studying other features, (ii) further experimentation with neural net architectures.

# 11 Conclusion

We saw some beautiful pictorial representations by analysing thousands of data which will help us to visualize as to which zones are earthquake prone. Three machine learning techniques have been used to predict earthquakes in three regions on the edges of the Indian tectonic plate, which are some of the most active seismic regions of the world. Every applied classifier shows slightly different results from each other. Prediction of the magnitude and Depth based on a set of features including locations, longitudes and latitudes, Time and Date has been successfully performed. The study shows, although earthquake occurrence is supposed to be decidedly nonlinear and appears to be a random phenomenon, yet it can be modeled on the basis of geophysical facts of the seismic region along with highly sophisticated modeling and learning approaches of machine learning.

# 12 References


- K. M. Asim, F Martinez-Alvarez, A. Basit, T. Iqbal(2016) Earthquake magnitude prediction in Hindukush region using machine learning techniques:472-473
- PAIBOON NUANNIN(2006) The Potential of b-value Variations as Earthquake Precursors for Small and Large Events:12-13
- YAOLIN SHI AND BRUCE A. BOLT(1982) THE STANDARD ERROR OF THE MAGNITUDE-FREQUENCY b VALUE:1-2
- Andrzej Kijko and Ansie Smit(2012) Extension of the Aki-Utsu b-Value Estimator for Incomplete Catalogs
- Keitii AKI(1965) Maximum Likelihood estimate of b in the formula log N=a-bM and its confidence limits
- 2001 Gujarat earthquake wikipedia
- April 2015 Nepal earthquake wikipedia
- 2004 Indian Ocean earthquake and tsunami wikipedia