PROCEDURE TO WORK KAAGGLE FOR EARTH QUAKE PREDICTION MODEL USING PYTHON

```
IMPORT NUMPY AS NP
IMPORT PANDAS AS PD
IMPORT MATPLOTLIB.PYPLOT AS PLT

IMPORT OS
PRINT(OS.LISTDIR("../INPUT"))
```

['DATABASE.CSV']

READ THE DATA FROM CSV AND ALSO COLUMNS WHICH ARE NECESSARY FOR THE MODEL AND THE COLUMN WHICH NEEDS TO BE PREDICTED.

INPUT [2]:

```
DATA = PD.READ_CSV("../INPUT/DATABASE.CSV")
DATA.HEAD()
```

OUTPUT[2]:

	DATE	TIME	LATITUDE	LONGITUDE	TYPE	DEPTH	DEPTH ERROR	DEPTH SEISMIC STATIONS	MAGNITUDE	MAGNITUDE TYPE	MAGNITUDE ERROR	MAGNITUDE SEISMIC STATIONS	AZIMUTHAL GAP	HORIZONTAL DISTANCE	HORIZONTAL ERROR	ROOT MEAN SQUARE	ID	SOURCE	LOCATION SOURCE	MAGNITUDE SOURCE	STATUS
0	01/02/1	13:44	19.2	145.6	EARTHQU	131	NAN	NAN	6.0	MW	NAN	NAN	NAN	NAN	NAN	NAN	ISCGEM860	ISCG	ISCG	ISCG	AUTOM

	DATE	TIME	LATITUDE	LONGITUDE	TYPE	DEPTH	DEPTHERROR	DEPTHSEISMICSTATIONS	MAGNITUDE	MAGNITUDETYPE	MAGNITUDEERROR	MAGNITUDESEISMICSTATIONS	AZIMUTHALGAP	HORIZONTALDISTANCE	HORIZONTALERROR	ROOTMEANSQUARE	ID	SOURCE	LOCATIONSOURCE	MAGNITUDESOURCE	STATUS
	965	:18	46	16	AKE	.6											706	EM	EM	EM	ATIC
1	01/04/1965	11:29:49	1.863	127.352	EARTHQUAKE	80.0	NAN	NAN	5.8	MW	NAN	NAN	NAN	NAN	NAN	NAN	ISCGEM860737	ISCGEM	ISCGEM	ISCGEM	AUTOMATIC
2	01/05/1965	18:05:58	-20.579	-173.972	EARTHQUAKE	20.0	NAN	NAN	6.2	MW	NAN	NAN	NAN	NAN	NAN	NAN	ISCGEM860762	ISCGEM	ISCGEM	ISCGEM	AUTOMATIC

	DATE	TIME	LATITUDE	LONGITUDE	TYPE	DEPTH	DEPTH ERROR	DEPTH SEISMIC STATIONS	MAGNITUDE	MAGNITUDE TYPE	MAGNITUDE ERROR	MAGNITUDE SEISMIC STATIONS	AZIMUTHAL GAP	HORIZONTAL DISTANCE	HORIZONTAL ERROR	ROOT MEAN SQUARE	ID	SOURCE	LOCATION SOURCE	MAGNITUDE SOURCE	STATUS
3	01/08/1965	18:49:43	-59.076	-23.557	EARTHQUAKE	15.0	NAN	NAN	5.8	MW	NAN	NAN	NAN	NAN	NAN	NAN	ISCGEM860856	ISCGEM	ISCGEM	ISCGEM	AUTOMATIC
4	01/09/1965	13:32:50	11.938	126.427	EARTHQUAKE	15.0	NAN	NAN	5.8	MW	NAN	NAN	NAN	NAN	NAN	NAN	ISCGEM860890	ISCGEM	ISCGEM	ISCGEM	AUTOMATIC

INPUT [3]:

DATA.COLUMNS

OUTPUT[3]:

INDEX(['DATE', 'TIME', 'LATITUDE', 'LONGITUDE', 'TYPE', 'DEPTH', 'DEPTH ERROR',
'DEPTH SEISMIC STATIONS', 'MAGNITUDE', 'MAGNITUDE TYPE',

```
'MAGNITUDE ERROR', 'MAGNITUDE SEISMIC STATIONS', 'AZIMUTHAL GAP',
 'HORIZONTAL DISTANCE', 'HORIZONTAL ERROR', 'ROOT MEAN SQUARE', 'ID',
 'SOURCE', 'LOCATION SOURCE', 'MAGNITUDE SOURCE', 'STATUS'],
DTYPE='OBJECT')
```

FIGURE OUT THE MAIN FEATURES FROM EARTHQUAKE DATA AND CREATE A OBJECT OF THAT FEATURES, NAMELY, DATE, TIME, LATITUDE, LONGITUDE, DEPTH, MAGNITUDE.

INPUT[4]:

```
DATA = DATA[['DATE', 'TIME', 'LATITUDE', 'LONGITUDE', 'DEPTH', 'MAGNITUDE']]
DATA.HEAD()
```

OUTPUT[4]:

	DATE	TIME	LATITUDE	LONGITUDE	DEPTH	MAGNITUDE
0	01/02/1965	13:44:18	19.246	145.616	131.6	6.0
1	01/04/1965	11:29:49	1.863	127.352	80.0	5.8
2	01/05/1965	18:05:58	-20.579	-173.972	20.0	6.2
3	01/08/1965	18:49:43	-59.076	-23.557	15.0	5.8
4	01/09/1965	13:32:50	11.938	126.427	15.0	5.8

HERE, THE DATA IS RANDOM WE NEED TO SCALE ACCORDING TO INPUTS TO THE MODEL. IN THIS, WE CONVERT GIVEN DATE AND TIME TO UNIX TIME WHICH IS IN SECONDS AND A NUMERAL. THIS CAN BE EASILY USED AS INPUT FOR THE NETWORK WE BUILT.

INPUT [5]:

```
IMPORT DATETIME
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(ts.timetuple()))
  except ValueError:
    # print('ValueError')
    timestamp.append('ValueError')
```

In [6]:

```
timeStamp = pd.Series(timestamp)
data['Timestamp'] = timeStamp.values
```

In [7]:

```
final_data = data.drop(['Date', 'Time'], axis=1)
final_data = final_data[final_data.Timestamp != 'ValueError']
final_data.head()
```

Out[7]:

	Latitude	Longitude	Depth	Magnitude	Timestamp
0	19.246	145.616	131.6	6.0	-1.57631E+08
1	1.863	127.352	80.0	5.8	-1.57466E+08
2	-20.579	-173.972	20.0	6.2	-1.57356E+08
3	-59.076	-23.557	15.0	5.8	-1.57094E+08
4	11.938	126.427	15.0	5.8	-1.57026E+08

Visualization

Here, all the earthquakes from the database in visualized on to the world map which shows clear representation of the locations where frequency of the earthquake will be more.

Input [8]:

```
from mpl_toolkits.basemap import Basemap
```

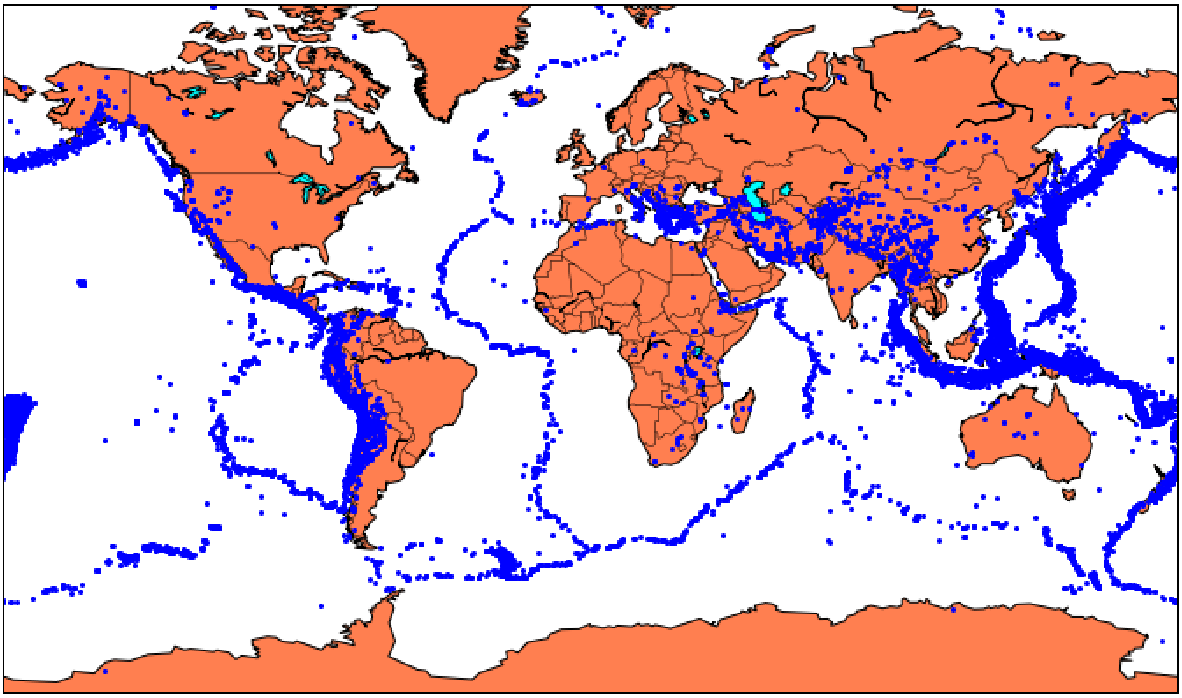
```
M = BASEMAP(PROJECTION='MILL',LLCRNRLAT=-80,URCRNRLAT=80, LLCRNRLON
=-180,URCRNRLON=180,LAT_TS=20,RESOLUTION='C')

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(LONGITUDES,LATITUDES)
```

INPUT[9]:

```
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()
```

```
/OPT/CONDA/LIB/PYTHON3.6/SITE-PACKAGES/MPL_TOOLKITS/BASEMAP/__INIT
__.PY:1704: MATPLOTLIBDEPRECATIONWARNING: THE AXESPATCH FUNCTION WA
S DEPRECATED IN VERSION 2.1. USE AXES.PATCH INSTEAD.
  LIMB = AX.AXESPATCH
/OPT/CONDA/LIB/PYTHON3.6/SITE-PACKAGES/MPL_TOOLKITS/BASEMAP/__INIT
__.PY:1707: MATPLOTLIBDEPRECATIONWARNING: THE AXESPATCH FUNCTION WA
S DEPRECATED IN VERSION 2.1. USE AXES.PATCH INSTEAD.
  IF LIMB IS NOT AX.AXESPATCH:
```

All affected areas



SPLITTING THE DATA

FIRSTLY, SPLIT THE DATA INTO XS AND YS WHICH ARE INPUT TO THE MODEL AND OUTPUT OF THE MODEL RESPECTIVELY. HERE, INPUTS ARE TIMESTAMP, LATITUDE AND LONGITUDE AND OUTPUTS ARE MAGNITUDE AND DEPTH. SPLIT THE XS AND YS INTO TRAIN AND TEST WITH VALIDATION. TRAINING DATASET CONTAINS 80% AND TEST DATASET CONTAINS 20%.

INPUT [10]:

```
X = FINAL_DATA[['TIMESTAMP', 'LATITUDE', 'LONGITUDE']]
Y = FINAL_DATA[['MAGNITUDE', 'DEPTH']]
```

INPUT [11]:

```
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)
```

```
(18727, 3) (4682, 3) (18727, 2) (4682, 3)
/OPT/CONDA/LIB/PYTHON3.6/SITE-PACKAGES/SKLEARN/CROSS_VALIDATION.PY:41: DEPRECATIONWARNING: THIS MODULE WAS DEPRECATED IN VERSION 0.18 IN FAVOR OF THE MODEL_SELECTION MODULE INTO WHICH ALL THE REFACTORED CLASSES AND FUNCTIONS ARE MOVED. ALSO NOTE THAT THE INTERFACE OF THE NEW CV ITERATORS ARE DIFFERENT FROM THAT OF THIS MODULE. THIS MODULE WILL BE REMOVED IN 0.20.
  "THIS MODULE WILL BE REMOVED IN 0.20.", DEPRECATION WARNING)
```

HERE, WE USED THE RANDOM FOREST REGRESSOR MODEL TO PREDICT THE OUTPUTS, WE SEE THE STRANGE PREDICTION FROM THIS WITH SCORE ABOVE 80% WHICH CAN BE ASSUMED TO BE BEST FIT BUT NOT DUE TO ITS PREDICTED VALUES.