ARTIFICAL INTELLIGENCE

PHASE-3 SUBMISSION

EARTHQUAKE PREDICTION USING PYTHON

For machine learning algorithms to work, it's necessary to convert **raw data** into a **clean data** set, which means we must convert the data set to **numeric data**. We do this by encoding all the **categorical labels** to column vectors with binary values. **Missing values**, or NaNs (not a number) in the data set is an annoying problem. You have to either drop the missing rows or fill them up with a mean or interpolated values.

Preprocess data in Python – Step by step:

1. Load data in Pandas.
2. Drop columns that aren't useful.
3. Drop rows with missing values.
4. Create dummy variables.
5. Take care of missing data.
6. Convert the data frame to NumPy.
7. Divide the data set into training data and test data.

1.Load data in Pandas:

To work on the data, you can either load the CSV in Excel or in Pandas. For the purposes of this tutorial, we'll load the CSV data in Pandas.

```
[ ] import pandas as pd
    df = pd.read_csv("database.csv")
```

Let's take a look at the data format below:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 21 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Date                        23412 non-null  object 
 1   Time                        23412 non-null  object 
 2   Latitude                    23412 non-null  float64
 3   Longitude                   23412 non-null  float64
 4   Type                        23412 non-null  object 
 5   Depth                       23412 non-null  float64
 6   Depth Error                 4461 non-null   float64
 7   Depth Seismic Stations      7097 non-null   float64
 8   Magnitude                   23412 non-null  float64
 9   Magnitude Type              23409 non-null  object 
 10  Magnitude Error             327 non-null    float64
 11  Magnitude Seismic Stations  2564 non-null   float64
 12  Azimuthal Gap               7299 non-null   float64
 13  Horizontal Distance         1604 non-null   float64
 14  Horizontal Error            1156 non-null   float64
 15  Root Mean Square            17352 non-null  float64
 16  ID                          23412 non-null  object 
 17  Source                      23412 non-null  object 
 18  Location Source             23412 non-null  object 
 19  Magnitude Source            23412 non-null  object 
 20  Status                      23412 non-null  object 
dtypes: float64(12), object(9)
memory usage: 3.8+ MB
```

2. Drop Columns That Aren't Useful: Let's try to drop some of the columns which won't contribute much to our machine learning model. We'll start with Date and Time.

```
cols=['Date','Time']
df=df.drop(cols, axis=1)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23412 entries, 0 to 23411
Data columns (total 19 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Latitude                    23412 non-null  float64
 1   Longitude                   23412 non-null  float64
 2   Type                        23412 non-null  object 
 3   Depth                       23412 non-null  float64
 4   Depth Error                 4461 non-null   float64
 5   Depth Seismic Stations      7097 non-null   float64
 6   Magnitude                   23412 non-null  float64
 7   Magnitude Type              23409 non-null  object 
 8   Magnitude Error             327 non-null    float64
 9   Magnitude Seismic Stations  2564 non-null   float64
 10  Azimuthal Gap               7299 non-null   float64
 11  Horizontal Distance         1604 non-null   float64
 12  Horizontal Error            1156 non-null   float64
 13  Root Mean Square            17352 non-null  float64
 14  ID                          23412 non-null  object 
 15  Source                      23412 non-null  object 
 16  Location Source             23412 non-null  object 
 17  Magnitude Source            23412 non-null  object 
 18  Status                      23412 non-null  object 
dtypes: float64(12), object(7)
memory usage: 3.4+ MB
```

3. Drop Rows With Missing Values: Next we can drop all rows in the data that have missing values (NaNs). Here's how:

```
[ ] df=df.dropna()
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 14 entries, 565 to 22238
Data columns (total 19 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Latitude                    14 non-null     float64
 1   Longitude                   14 non-null     float64
 2   Type                        14 non-null     object
 3   Depth                       14 non-null     float64
 4   Depth Error                 14 non-null     float64
 5   Depth Seismic Stations      14 non-null     float64
 6   Magnitude                   14 non-null     float64
 7   Magnitude Type              14 non-null     object
 8   Magnitude Error             14 non-null     float64
 9   Magnitude Seismic Stations  14 non-null     float64
 10  Azimuthal Gap               14 non-null     float64
 11  Horizontal Distance         14 non-null     float64
 12  Horizontal Error            14 non-null     float64
 13  Root Mean Square            14 non-null     float64
 14  ID                          14 non-null     object
 15  Source                      14 non-null     object
 16  Location Source             14 non-null     object
 17  Magnitude Source            14 non-null     object
 18  Status                      14 non-null     object
dtypes: float64(12), object(7)
memory usage: 2.2+ KB
```

4. Creating Dummy Variables

Instead of wasting our data, let's convert the Latitude and Longitude to columns in Pandas and drop them after conversion.

```
[ ] dummies=[]
    cols=['Latitude', 'Longitude']
    for col in cols:
      dummies.append(pd.get_dummies(df[col]))
```

Then..

```
database_dummies=pd.concat(dummies, axis=1)
```

Finally we **concatenate** to the original data frame, column-wise:

```
df=pd.concat((df,database_dummies), axis=1)
```

Now that we converted Latitude and Longitude values into columns, we drop the redundant columns from the data frame.

```
df=df.drop(['Latitude', 'Longitude'], axis=1)
```

Let's take a look at the new data frame:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 14 entries, 565 to 22238
Data columns (total 45 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Type                        14 non-null     object
 1   Depth                       14 non-null     float64
 2   Depth Error                 14 non-null     float64
 3   Depth Seismic Stations      14 non-null     float64
 4   Magnitude                   14 non-null     float64
 5   Magnitude Type              14 non-null     object
 6   Magnitude Error             14 non-null     float64
 7   Magnitude Seismic Stations  14 non-null     float64
 8   Azimuthal Gap               14 non-null     float64
 9   Horizontal Distance         14 non-null     float64
 10  Horizontal Error            14 non-null     float64
 11  Root Mean Square            14 non-null     float64
 12  ID                          14 non-null     object
 13  Source                      14 non-null     object
 14  Location Source             14 non-null     object
 15  Magnitude Source            14 non-null     object
 16  Status                      14 non-null     object
 17  18.045                      14 non-null     uint8
 18  30.25                       14 non-null     uint8
 19  37.2315                     14 non-null     uint8
 20  37.245                      14 non-null     uint8
 21  37.2788333                  14 non-null     uint8
 22  37.2901667                  14 non-null     uint8
 23  37.2953333                  14 non-null     uint8
 24  37.2965                     14 non-null     uint8
 25  37.3005                     14 non-null     uint8
 26  37.3021667                  14 non-null     uint8
 27  37.3141667                  14 non-null     uint8
 28  38.1383333                  14 non-null     uint8
 29  41.1444                     14 non-null     uint8
 30  46.2073333                  14 non-null     uint8
```

```
 31  -122.188                    14 non-null     uint8
 32  -118.3913333                14 non-null     uint8
 33  -116.5341667                14 non-null     uint8
 34  -116.4736667                14 non-null     uint8
 35  -116.4606667                14 non-null     uint8
 36  -116.4556667                14 non-null     uint8
 37  -116.4115                   14 non-null     uint8
 38  -116.4083333                14 non-null     uint8
 39  -116.3686667                14 non-null     uint8
 40  -116.346                    14 non-null     uint8
 41  -116.3331667                14 non-null     uint8
 42  -114.8721                   14 non-null     uint8
 43  -114.8                      14 non-null     uint8
 44  -68.3509                    14 non-null     uint8
dtypes: float64(10), object(7), uint8(28)
memory usage: 2.4+ KB
```

Let’s compute a median or interpolate() all the ages and fill those missing age values. Pandas has an interpolate() function that will replace all the missing NaNs to interpolated values.

4. Take Care of Missing Data

```
df['Type']=df['Type'].interpolate()
```

Now let's observe the data columns. Notice 'Close' is now interpolated with imputed new values.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 14 entries, 565 to 22238
Data columns (total 45 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   Type                         14 non-null     object 
 1   Depth                        14 non-null     float64
 2   Depth Error                  14 non-null     float64
 3   Depth Seismic Stations       14 non-null     float64
 4   Magnitude                    14 non-null     float64
 5   Magnitude Type               14 non-null     object 
 6   Magnitude Error              14 non-null     float64
 7   Magnitude Seismic Stations   14 non-null     float64
 8   Azimuthal Gap                14 non-null     float64
 9   Horizontal Distance          14 non-null     float64
 10  Horizontal Error             14 non-null     float64
 11  Root Mean Square             14 non-null     float64
 12  ID                           14 non-null     object 
 13  Source                       14 non-null     object 
 14  Location Source              14 non-null     object 
 15  Magnitude Source             14 non-null     object 
 16  Status                       14 non-null     object 
 17  18.045                       14 non-null     uint8  
 18  30.25                        14 non-null     uint8  
 19  37.2315                      14 non-null     uint8  
 20  37.245                       14 non-null     uint8  
 21  37.2788333                   14 non-null     uint8  
 22  37.2901667                   14 non-null     uint8  
 23  37.2953333                   14 non-null     uint8  
 24  37.2965                      14 non-null     uint8  
 25  37.3005                      14 non-null     uint8  
 26  37.3021667                   14 non-null     uint8  
 27  37.3141667                   14 non-null     uint8  
 28  38.1383333                   14 non-null     uint8  
 29  41.1444                      14 non-null     uint8  
 30  46.2073333                   14 non-null     uint8  
```

```
 31  -122.188                   14 non-null     uint8
 32  -118.3913333               14 non-null     uint8
 33  -116.5341667               14 non-null     uint8
 34  -116.4736667               14 non-null     uint8
 35  -116.4606667               14 non-null     uint8
 36  -116.4556667               14 non-null     uint8
 37  -116.4115                  14 non-null     uint8
 38  -116.4083333               14 non-null     uint8
 39  -116.3686667               14 non-null     uint8
 40  -116.346                   14 non-null     uint8
 41  -116.3331667               14 non-null     uint8
 42  -114.8721                  14 non-null     uint8
 43  -114.8                     14 non-null     uint8
 44  -68.3509                   14 non-null     uint8
dtypes: float64(10), object(7), uint8(28)
memory usage: 2.4+ KB
```

6. Convert the Data Frame to NumPy: Now that we've converted all the data to integers, it's time to prepare the data for machine learning models. This is where scikit-learn and NumPy come into play: X= Input set with 14 attributes y = Small y output, in this case Survived

Now we convert our data frame from Pandas to NumPy and we assign input and output:

```
X=df.values
y=df['Root Mean Square'].values
```

still has Root Mean Square values in it, which should not be there. So we drop in the NumPy column, which is the first column.

X

```
import numpy as np
X=np.delete(x, 1, axis=1)
```

7. Divide the Data Set Into Training Data and Test Data

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```