In [2]:

```
# Ayush Sharma 209303312
# AIML Mini Project
# Model for Earthquake Prediction using Machine Learning and the Python
import datetime
import time
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from mpl_toolkits.basemap import Basemap
```

In [3]:

```
url = r'https://raw.githubusercontent.com/amankharwal/Website-data/master/database.
df = pd.read_csv(url)
df.head()
```

Out[3]:

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations	Magnitude	Ma
0	01/02/1965	13:44:18	19.246	145.616	Earthquake	131.6	NaN	NaN	6.0	
1	01/04/1965	11:29:49	1.863	127.352	Earthquake	80.0	NaN	NaN	5.8	
2	01/05/1965	18:05:58	-20.579	-173.972	Earthquake	20.0	NaN	NaN	6.2	
3	01/08/1965	18:49:43	-59.076	-23.557	Earthquake	15.0	NaN	NaN	5.8	
4	01/09/1965	13:32:50	11.938	126.427	Earthquake	15.0	NaN	NaN	5.8	

5 rows × 21 columns

In [4]:

```
df.columns
```

Out[4]:

```
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')
```

In [5]:

```
df = df[['Date', 'Time', 'Latitude', 'Longitude', 'Depth', 'Magnitude']]
df.head()
```

Out[5]:

	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

In [6]:

```
timestamp = []
for d, t in zip(df['Date'], df['Time']):
    try:
        ts = datetime.datetime.strptime(d+' '+t, '%m/%d/%Y %H:%M:%S')
        epoch = datetime.datetime.utcfromtimestamp(0)
        delta = ts - epoch
        timestamp.append(int(delta.total_seconds()))
    except ValueError:
        timestamp.append('ValueError')
timeStamp = pd.Series(timestamp)
df['Timestamp'] = timeStamp.values
fdf = df.drop(['Date', 'Time'], axis=1)
fdf = fdf[fdf.Timestamp != 'ValueError']
fdf.head()
```

Out[6]:

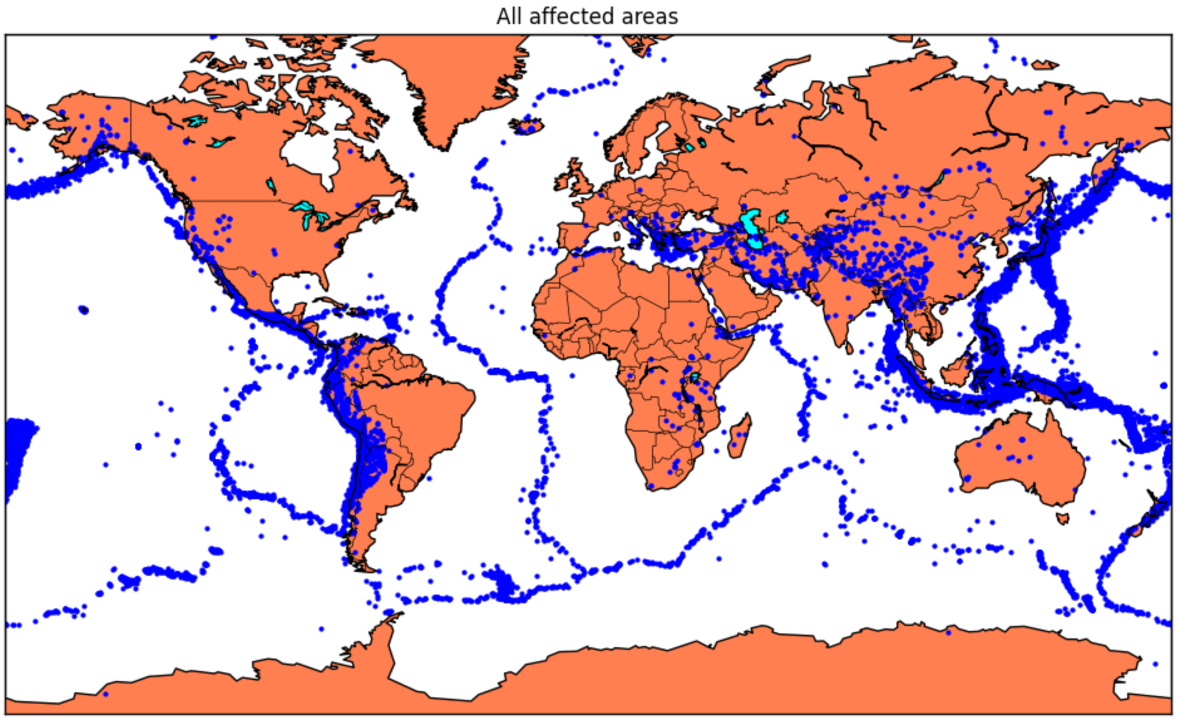
	Latitude	Longitude	Depth	Magnitude	Timestamp
0	19.246	145.616	131.6	6.0	-157630542
1	1.863	127.352	80.0	5.8	-157465811
2	-20.579	-173.972	20.0	6.2	-157355642
3	-59.076	-23.557	15.0	5.8	-157093817
4	11.938	126.427	15.0	5.8	-157026430

In [7]:

```
m = Basemap(projection='mill',llcrnrlat=-80,urcrnrlat=80, llcrnrlon=-180,urcrnrlon=

longitudes = df["Longitude"].tolist()
latitudes = df["Latitude"].tolist()
x,y = m(longitudes,latitudes)

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()
```



In [8]:

```
X = fdf[['Timestamp', 'Latitude', 'Longitude']]
y = fdf[['Magnitude', 'Depth']]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
print(X_train.shape, X_test.shape, y_train.shape, X_test.shape)
```

```
(18727, 3) (4682, 3) (18727, 2) (4682, 3)
```

In [9]:

```
X_train=np.asarray(X_train).astype(int)
y_train = np.array(y_train).astype(int)
X_test=np.asarray(X_train).astype(int)
y_test = np.array(y_train).astype(int)
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_loss, test_acc] = model.evaluate(X_test, y_test)
print("Evaluation result on Test Data : Loss = {}, accuracy = {}".format(test_loss,
Epoch 1/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 2/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 3/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 4/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 5/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 6/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 7/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 8/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 9/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 10/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 11/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 12/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 13/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 14/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 15/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 16/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 17/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 18/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 19/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
```

```
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
Epoch 20/20
1873/1873 [==============================] - 6s 3ms/step - loss: 0.5039 - accurac
y: 0.9801 - val_loss: 0.5039 - val_accuracy: 0.9801
586/586 [==============================] - 1s 2ms/step - loss: 0.5039 - accuracy:
0.9801
Evaluation result on Test Data : Loss = 0.5039247870445251, accuracy = 0.980135619
6403503
```

In [ ]:

In [ ]: