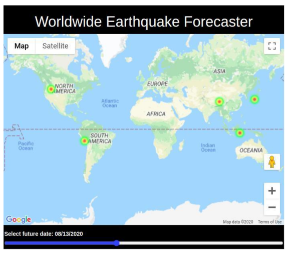
project: Earthquake prediction   using python


Worldwide Earthquake Forecaster
Map
Satellite
NORTH AMERICA
EUROPE
ASIA
AFRICA
SOUTH AMERICA
OCEANIA
Select future date: 08/13/2020

- **`Data`**`/` : Notebook and HTML file `ETL_USGS_EarthQuake.ipybn` for ETL and EDA part of the project, and it also contains cleaned data in Earthquake.db & Earthquake_data.db format saved after ETL process
- **`models`**`/` : Notebook and HTML file `Earthquake-prediction-ML-workflow.ipynb` which has all the implementation after related to Prediction steps and Machine Learning pipeline.
- **`Web App`**`/` : all the necessary routing python files in `main.py` for flask application i.e from data extraction to modelling application and convert prediction coordinates to google maps api format

I have implemented all the neccesary steps in these IPYBN notebooks. I recommend for project walkthrough follow -

1. For ETL walkthrough open `Data/ETL_USGS_EarthQuake.ipybn` or `Data/ETL_USGS_EarthQuake.html`
2. Next, go to `models/Earthquake-prediction-ML-workflow.ipynb` or `models/Earthquake-prediction-ML-workflow.html` for ML and workflow.

Requirements

1. click==7.1.2
2. Flask==1.1.2
3. gunicorn==20.0.4
4. itsdangerous==1.1.0
5. Jinja2==2.11.2
6. joblib==0.16.0
7. MarkupSafe==1.1.1
8. numpy==1.19.1
9. pandas==1.1.0
10. python-dateutil==2.8.1
11. pytz==2020.1
12. scikit-learn==0.23.1
13. scipy==1.5.2
14. six==1.15.0
15. sklearn==0.0

16. SQLAlchemy==1.3.18
17. threadpoolctl==2.1.0
18. Werkzeug==1.0.1
19. xgboost==1.1.1
20. python 3.x

Linux/Mac Users

Note for **windows user** : install git bash and proceed with same instruction as linux.

step 1 : `$ git clone https://github.com/aditya-167/Realtime-Earthquake-forecasting.git`

step 2 : `$ cd Real Time-Earthquake-forecasting`

step 3 : `$ python3 -m venv <<any environment name>>` (If error occurs, download virtual environment for python)

step 4 : `$ source <<any environment name>>/bin/activate`

step 5 : `$ pip install --upgrade pip`

step 6 : `$ pip install -r requirements.txt` (If error occurs in xgboost installation, upgrade pip using step 5)

step 7 : Run application with `$ python application.py` i.e in root directory of project repo.

step 8 : Go to local host when application starts and use slider to choose dates for prediction in app.

- .

I have implemented all the neccesary steps in these IPYBN notebooks. I recommend for project walkthrough follow -

1. For ETL walkthrough open `Data/ETL_USGS_EarthQuake.ipybn` or `Data/ETL_USGS_EarthQuake.html`
2. Next, go to `models/Earthquake-prediction-ML-workflow.ipybn` or `models/Earthquake-prediction-ML-workflow.html` for ML and workflow.

Linux/Mac Users

Note for **windows user** : install gitbash and proceed with same instruction as linux.

`step 1:$ git clone https://github.com/aditya-167/Realtime-Earthquake-forecasting.git`

`step 2:$ cd Realtime-Earthquake-forecasting`

`step 3:$ python3 -m venv <<any environment name>>` (If error occurs, download virtual environment for python)

`step 4:$ source <<any environment name>>/bin/activate`

`step 5:$ pip install --upgrade pip`

`step 6:$ pip install -r requirements.txt` (If error occurs in xgboost installation, upgrade pip using step 5)

`step 7`: Run application with `$ python application.py` i.e in root directory of project repo.

`step 8`: Go to local host when application starts and use slider to choose dates for prediction in app.

content:

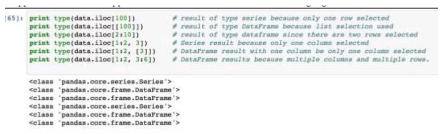
Project Overview

Countless dollars and entire scientific careers have been dedicated to predicting where and when the next big earthquake will strike. But unlike weather forecasting, which has significantly improved with the use of better satellites and more powerful mathematical models, earthquake prediction has been marred by repeated failure due to highly uncertain conditions of earth and its surroundings. Now, with the help of artificial intelligence, a growing number of scientists say changes in the way they can analyze massive amounts of seismic data can help them better understand earthquakes, anticipate how they will behave, and provide quicker and more accurate early warnings. This helps in hazzard assessments for many builders and real estate business for infrastructure planning from business perspective. Also many lives can be saved through early warning. This project aims a simple solution to above problem by predicting or forecasting likely places to have earthquake in next 7 days. For user-friendly part, this project has a web application that extracts live data updated every minute by USGS.gov and predicts next likely place world wide to get hit by an earthquake, hence a realtime solution is provided.

Problem Statement and approach to solution

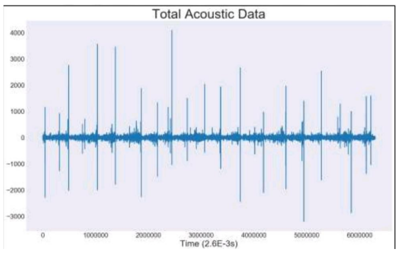
Anticipating seismic tremors is a pivotal issue in Earth science because of their overwhelming and huge scope outcomes. The goal of this project is to predict where likely in the world and on what dates the earthquake will happen. Application and impact of the project includes potential to improve earthquake hazard assessments that could spare lives and billions of dollars in infrastructure and planning. Given geological locations, magnitude and other factors in dataset from https://earthquake.usgs.gov/earthquakes/feed/v1.0/csv.php for 30 days past which is updated every minute, we predict or forecast 7 days time in future that is yet to come, the places where quake would likely happen. Since this is event series problem type, proposed solution in this project follows considering binary classification of earthquake occurance with training period includes fixed rolling window moving averages of past days while for which its labels, a fixed window size shifted ahead in time. The model will be trained with Adaboost classifier (RandomForestClassifier and DecisionTreeClassifier) and compared with XGBoost based on AUC ROC score and recall score due to the nature of problem (i.e binary classification). Model with better AUC score and recall will be considered for web app that uses Google maps api to predict places where earthquake might occur.

The problem addressed above is about binary classification, `Earthquake occur = 1` and `Earthquake not occur = 0` and with these prediction we try to locate co-cordinates corrosponding to the predictions and display it on the google maps api web app. More suitable metrics for binary clsssification problems are **ROC (Reciever operator characteristics), AUC (Area Under Curve), Confusion matrix for Precision, recall, accuracy and sensitivity**. One important thing about choosing metrics and model is what exactly we need from predictions and what not. To be precise, we need to **minimize or get less False negative predictions** since we dont want our model to predict as `0` or `no earthquake occured` at particular location when in reality it had actually happend as this is more dangerous than the prediction case in which prediction is `true/1` or `earthquake occured` but in reality it did not because its always **better safe than sorry!!!**. Hence apart from `roc_auc score`, I have considered `Recall` as well for evaluation and model selection with `higher auc_roc score and recall`, where `recall = (TP/TP+FN)`.

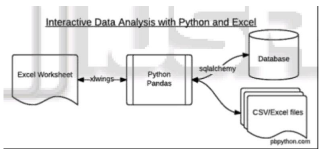


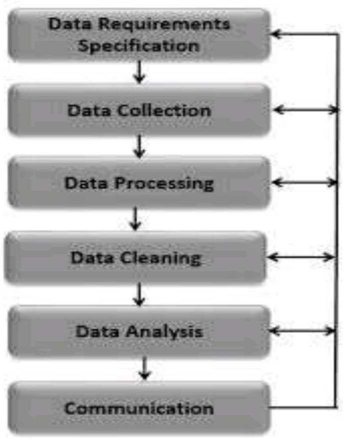
```
[65]: print type(data.iloc[100])          # result of type series because only one row selected
      print type(data.iloc[[100]])        # result of type DataFrame because list selection used
      print type(data.iloc[2:10])         # result of type dataframe since there are two rows selected
      print type(data.iloc[1:2, 3])       # Series result because only one column selected
      print type(data.iloc[1:2, [3]])     # DataFrame result with one column be only one column selected
      print type(data.iloc[1:2, 3:6])     # DataFrame results because multiple columns and multiple rows.

      <class 'pandas.core.series.Series'>
      <class 'pandas.core.frame.DataFrame'>
      <class 'pandas.core.frame.DataFrame'>
      <class 'pandas.core.series.Series'>
      <class 'pandas.core.frame.DataFrame'>
      <class 'pandas.core.frame.DataFrame'>
```



working construction and flow diagaram:




Data Requirements Specification
Data Collection
Data Processing
Data Cleaning
Data Analysis
Communication