Analysis of USGS Earthquake Data

NANDITA SATHE[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: nsathe@iu.edu*

project-001, April 24, 2017

Big Data Analytics tools allow us to analyze the huge volumes of geo-spatial data. Data of earthquakes that take place globally is a major part of crucial geo-spatial data. US Geological Survey's (USGS) Earthquake Hazards Program monitor and report earthquakes, assess earthquake impacts and hazards, and research the causes and effects of earthquakes [1]. Machine learning algorithms are used for advanced data analysis and earthquakes prediction.

Keywords: I524, geospatial, MongoDB, Plotly, K-means clustering, DBSCAN, Python, pymongo, USGS, Ansible, Cherrypy

https://github.com/cloudmesh/sp17-i524/blob/master/project/S17-IO-3017/report/report.pdf

1. INTRODUCTION

USGS collects volumes of geospatial data pertaining to earthquakes and makes it available for analysis. The project obtains a chunk of the data using GeoJSON web service provided by USGS. The data is saved locally in MongoDB database. Data is analysed using machine learning algorithms. The output is plotted (rendered) on web browser. A light weight web server is used to respond to the web requests. Project is capable of running in cloud environment. Deployment is automated using Ansible.

2. TECHNOLOGIES USED

Technologies used for development and deployment of this project are listed below.

1. Cloudmesh - For connecting to different cloud environments.
2. Ansible -For deploying software and associated packages.
3. Python - Writing script for data analysis and data processing
4. Mongo-DB - For storing Geo-spatial data
5. Plotly - As a visualization tool
6. Scikit-learn - For classification of data into clusters
7. Cherrypy - For web server development

3. BACKGROUND

3.1. USGS Web Service and Data Set

USGS earthquake data is fetched using the web service and passing relevant parameters to it. Out of the voluminous world-wide data of decades this project uses data of earthquakes appeared in North and South America for duration of 2 years from 2015-1-1 till 2017-1-1, having magnitude over 4. To select North and South America region data, its Latitude and Longitude are taken from Search Earth Catalog tool provided by USGS. [2]. Web service returns data in JSON format.

3.2. Data Processing and Visualization

Severity of earthquakes. Severity of earthquakes is analysed by their magnitude and depth. Data is classified into clusters using K-Means clustering algorithm (Section 4.1). Clusters are plotted on a interactive scatter plot.

Region affected the most by earthquakes. Analysing Latitude (lat)-Longitude (lon) of epicenter of the earthquake shows the regions where earthquakes are frequent. Lat lon data is clustered using DBSCAN algorithm (Section 4.2). Clusters are plotted on a geo-scatter plot on a world map.

4. IMPLEMENTATION

4.1. K-Means Clustering

Given a target number, k, of clusters to find, K-means algorithm locates the centers of each of those k clusters and the boundaries between them. It does this using the following algorithm [3].

- Step 1: Start with a randomly selected set of k centroids
- Step 2: Determine which observation is in which cluster, based on which centroid it is closest to (using the squared Euclidean distance.

$$\sum_{j=1}^{p}(x_{ij} - X_{i^1j})^2$$

where p is the number of dimensions)

- Re-calculate the centroids of each cluster by minimizing the squared Euclidean distance to each observation in the cluster

- Repeat 2. and 3. until the members of the clusters (and hence the positions of the centroids) no longer change.

4.2. DBSCAN

DBSCAN (Density-Based Spatial Clustering of Application with Noise) is a clustering algorithm. Given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), and marks points as outliers if they lie alone in low-density regions. Unlike K-Means Clustering, DBSCAN doesn't need to specify the number of clusters as it finds all the clusters that satisfy the requirement. Following points summarize the algorithm [4].

- Step 1: For each point in the data set, an n-dimensional sphere of radius *e*psilon is drawn around the point (if you have n-dimensional data).

- Step 2: If the number of points inside the sphere is larger than *m*in-samples, the center of the sphere is set as a cluster, and all the points within the sphere belong to this cluster.

- Step 3: Loop through all the points within the sphere with the above 2 steps, and expand the cluster whenever it satisfy the 2 rules.

- Step 4: For the points, which do not belong to any cluster, are treated as outliers.

4.3. Ansible

Project is deployed using ansible. Ansible is an automation platform, that automates cloud provisioning, configuration management, application deployments, etc. Ansible jobs are collected in a playbook and run on virtual clusters provided by Chameleon cloud. The tasks include cloning the git repository, installing dependencies and installing MongoDB.

4.4. Application

Application has a Cherrypy web server and python scripts that use the web server to send the output in the form of response so that users can see the output of the application in web browser.

Figure 1 shows default page showing application usage.

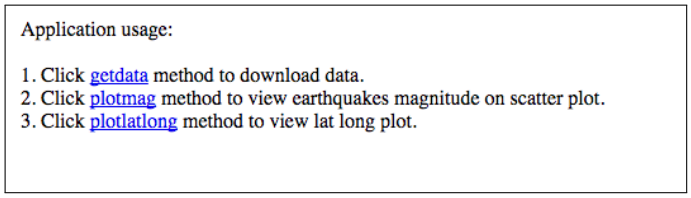

Fig. 1. Application usage

4.5. Steps to Execute

Steps to execute the project are explained in detail in README.md file [5].

5. EXECUTION PLAN

This is how the project is executed on week-by-week basis.

1. **13 Mar 2017 - 19 Mar 2017** Research on USGS data and fetching it. Deploy Mongo DB and Mongo Express tool to fetch data from Mongo DB. Data analysis and finalizing the analysis for project.

2. **20 Mar 2017 - 26 Mar 2017** Write script in Python for downloading USGS data at run-time and storing in local MongoDB. Research on clustering algorithms.

3. **27 Mar 2017 - 02 Apr 2017** Implement K-Means clustering and DBSCAN clustering in sample project. Understand the result. Create a VM on Chameleon.

4. **03 Apr 2017 - 09 Apr 2017** Write Python script to read data from MongoDB, implement K-Means clustering algorithm. Research on Plotly. Plot clustering result using Plotly.

5. **10 Apr 2017 - 16 Apr 2017** Implement DBSCAN clustering and plot clustering result using Plotly. Write ansible scripts for deployment.

6. **17 Apr 2017 - 23 Apr 2017** Deploy using ansible and run the project on various clouds. Observe and document benchmarks. Update and finalize the Project Report

6. BENCHMARKS

The Table 'Benchmark Identification' lists the benchmarks against which performance is tested. Application is tested on multiple environments to observe the performance against identified benchmarks.

Table **??** shows performance benchmarks.

Table 1. Benchmark Identification

Benchmark	Data Volume	Expected Result (sec)
Data Download Time	5000 records	1 sec
Data Read Time	5000 records	0.10 sec
Data processing Time	5000 records	0.10 sec

6.1. Benchmark Result

Performance tests are included in the python scripts themselves. As the scripts are executed, results are written in text files in 'benchmark' folder at project directory.

Performance tests are done on local VM, Chameleon cloud and on jetstream cloud. Results are shown in Table 'Benchmark Results'.

7. RESULT AND ANALYSIS

TODO

Table **??** shows performance tests on different environment.

Table 2. Benchmark Results. (Reading values are in seconds)

Benchmark	Data Volume	Local VM	Chameleon	Jetstream
Data Download Time	7034 records	2.16	0.70	9.18
Data Read Time-K-Means	7034 records	0.05	0.07	0.07
Data Read Time-DBSCAN	7034 records	0.10	0.06	0.07
Data processing-KMeans	7034 records	0.10	0.08	0.08
Data processing-DBSCAN	7034 records	0.30	0.09	0.02

8. ACKNOWLEDGEMENTS

This project is undertaken as part of the I524: Big Data and Open Source Software Projects course at Indiana University. The author would like to thank Prof. Gregor von Laszewski and his associates from the School of Informatics and Computing for providing all the technical support and assistance.

9. LICENSING

Project uses Apache license ver 2.0.

10. CONCLUSION

TBD

REFERENCES

[1] USGS, “Earthquake hazards program,” Web Page. [Online]. Available: https://earthquake.usgs.gov/

[2] ——, “Search earthquake catalog,” Web Page. [Online]. Available: https://earthquake.usgs.gov/earthquakes/search/

[3] B. Govan, “Clustering using scikit-learn,” Web Page, May 2013. [Online]. Available: http://fromdatawithlove.thegovans.us/2013/05/clustering-using-scikit-learn.html

[4] Q. Kong, “Clustering with dbscan,” Web Page, August 2016. [Online]. Available: http://qingkaikong.blogspot.in/2016/08/clustering-with-dbscan.html

[5] N. Sathe, “Readme,” Code Repository, April 2017, accessed: 2017-4-25. [Online]. Available: https://github.com/cloudmesh/earth/blob/master/README.md

11. WORK BREAKDOWN

The work on this project was distributed as follows between the authors:

Nandita Sathe. She completed all the work related to development of this application including research, testing and writing the project report.