CS596 Machine Learning Project: Classifying Earthquakes

Stephen Barrack, Julia Cai, Jake Champagne, Joseph Couri, Preethi Narayanan, Megan Respicio
San Diego State University
Undergraduate Studies of Computer Science
GitHub Repository

Abstract

With earthquakes being a potentially devastating event, people might have concerns over where earthquakes may occur and how strongly those earthquakes will be. We initially did Principal Component analysis (PCA) on our raw data after doing some basic cleaning.We created programs utilizing K-Nearest Neighbors, Logistic Regression and Linear Regression to analyze and predict earthquakes based on the earthquake features such as latitude, longitude, depth, time, and magnitude. KNN determines the likely magnitude that will be felt by taking into account the magnitude of earthquakes that had occurred near the location. For the Logistic Regression model, by inputting a valid zip code, the program predicts the likelihood an earthquake will be felt in the area and the range of the magnitude. Lastly, the Linear Regression model attempts to see the validity of depth as a feature for determining an earthquake's magnitude. With the results of our models, in the paper we will discuss in further detail about our earthquake predictions, major challenges and solutions, the methods and results, data used and models, comparison of the models, and explain how our models work.

1. Introduction

Every year more than 3 million earthquakes take place, most of these unnoticed by us. In contrast, a severe earthquake is the most frightening and catastrophic event of nature which can occur anywhere on the surface of our planet and destroy almost everything. Although usually lasting only seconds, a severe earthquake in a densely populated area may have catastrophic effects causing the death of hundreds of thousands of people, injuries, destruction and enormous damage to the economies of the affected area [7]. Hundreds of thousands of people have been killed by earthquakes despite scientists being able to predict and forewarn in advance and engineers construct earthquake-safe buildings. Unfortunately earthquakes occur often in countries which are unable to afford earthquake-safe construction. So we need to predict and be prepared for earthquake well in advance itself.

1.1 Task Description

For our project, we must use machine learning techniques we have learned. This project is to get us familiar with the common pipelines of ML tasks as well as understanding the process and procedures of conducting a research through the use of ML. We also were able to gain hands-on experience using ML techniques while writing a deliverable report.

We chose to study the topic of earthquakes and earthquakes predictions. As residents of California, earthquakes happen every day but most of them go unnoticed. However, every couple years a devastating earthquake hits. We would like to figure out a way if we could predict the next big earthquake along with its probable magnitude so we can prepare for the future.

1.2 Major Challenges and Solutions

One major challenge we had was determining which models we would use for our earthquake predictions. It was difficult finding multiple models that would help find a solution to the question "at a given location, what magnitude would an individual feel if an earthquake were to occur?", which was our first problem statement. Fortunately, we were able to use KNN to come up with a solution for that particular question. As for finding multiple models to address that question, we ended up having to formulate different questions that would fit with the features and output for various models.

One model we considered using, and worked on for a while, was Naive Bayes. We found that this model would not work in our case because each feature needs to be independent. In our research, we needed latitude and longitude to be considered together and not as separate features. Our subject did not fit the conditions for naive bayes, so we could not use this model.

Another model we considered using was support vector machine (SVM). We wanted to use this model for classification. However, we couldn't get our code to work properly with classifying more than 2 classes, which in this case is if an earthquake would be felt or not and its probable magnitude.

2. Methods and Results

2.1 Data and Models

A. *Principal Component Analysis*

When beginning research for any machine learning study, it's important to determine which features and variables will be the most relevant. We proposed several questions regarding earthquake prediction, but we first had to ensure that we are not including insignificant variables that could distract us from the original goal. We used PCA in order to know the important variables so that we can focus on them and ignore the other components.

The data used for our model was Xpca.csv and Y_query.csv as a predictor and response respectively. The parameters used as predictors in the model were latitude, longitude, depth, gap, dmin, rms, horizontalError, depthError, magError, magNST. The magnitude of earthquake is considered as response variable. Data was split into training and test sets. Then variables were also standardised to have uniformity across values. Through dimensionality reduction, we were able to then deal with smaller data. Since the new shape is almost 5x smaller than the old shape, this shows that PCA was successful in eliminating the unimportant variables.

```
Old shape: (8582, 10)
New shape: (1717, 10)
```

We have also observed the PC variance

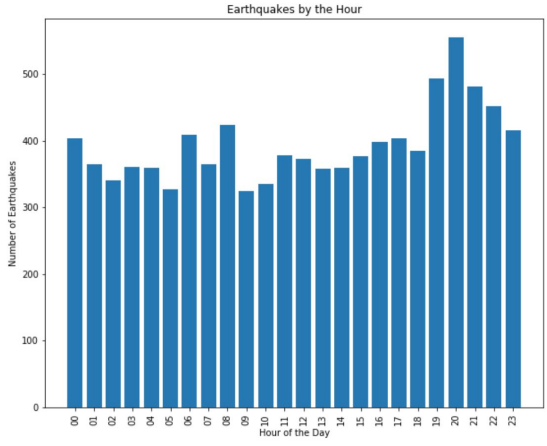
```
PC1 explains 55.706578180073215% of the total variance
PC2 explains 28.847944363330598% of the total variance
PC3 explains 10.978440765551639% of the total variance
PC4 explains 3.5617322922138546% of the total variance
PC5 explains 0.7706529851187569% of the total variance
PC6 explains 0.066027836096193% of the total variance
PC7 explains 0.042811930244362514% of the total variance
PC8 explains 0.025472501699184025% of the total variance
PC9 explains 0.0003247839096153602% of the total variance
PC10 explains 1.436176256889365e-05% of the total variance
First 10 PCs explains 100.0% of the total variance
```

B. *Linear Regression*

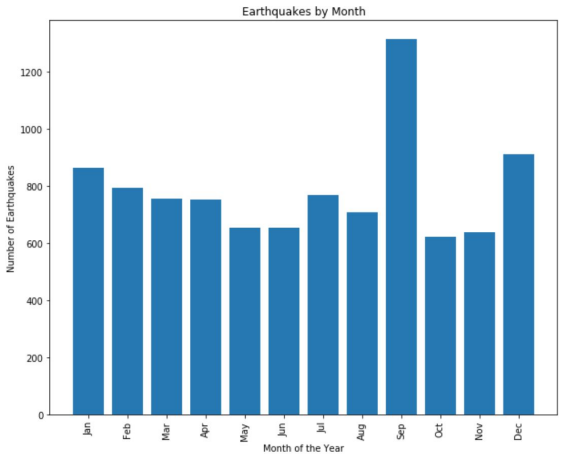
We furthered our understanding of earthquake prediction by using a linear regression model. We thought it was necessary to look for any possible relationship between some of the many features that USGS provided regarding earthquakes. Our linear regression model attempts to see if the depth of an earthquake is a valid feature when predicting earthquakes.

Our findings show a correlation coefficient of 0.00013435 and a mean squared error of 0.15794925052. Since the correlation coefficient is so close to 0, it indicates that depth and magnitude have no linear relationship. The scatter plot shows earthquakes with high magnitudes and low depths, low magnitudes with low depths, and even low magnitudes with high depths. From this study, we found that depth would not be a necessary feature to use in later in our analysis.

We also wanted to look at other features in our data and see if there was anything else we could use for our study. We decided that time of day and month of the year would be interesting features to analyze. After closer examination of the time of day that an earthquake was recorded, we found that earthquakes are dispersed pretty evenly throughout the day. However, it does seem that there is an uptick of earthquakes during the evening hours from about 7:00 to 9:59 pm.



Our analysis of the months of the year that earthquakes hit shows September to be a very highly impacted month for earthquakes in the last 3 years. September has over 400 more earthquakes recorded than than the next highest month of December. It would be interesting to find out why September is such a popular month for earthquakes in the United States.



C. *Logistic Regression*

We formulated the question "given a user's location, how likely will they will feel an earthquake and what magnitude would be felt". To answer this question, we implemented a supervised classification model - logistic regression.

The parameters used in the Logistic Model are the latitude, longitude, and to what degree the earthquake was "felt". The latitude and longitude are used to predict what "felt" will be. The values for "felt" are shown below:

- **felt = 0.** NOT-LIKELY feel an earthquake (mag: 0 - 2.4)
- **felt = 1.** MAYBE feel an earthquake (mag: 2.5 - 3.9)
- **felt = 2.** LIKELY feel an earthquake (mag: 4 - 4.9)
- **felt = 3.** LIKELY feel an earthquake (mag: 5 - 5.9) and MIGHT see some MINOR-SERIOUS damage from it
- **felt = 4.** LIKELY feel an earthquake (mag: 6 - 8) and LIKELY see some SERIOUS

The Logistic Model makes the calculations for future predictions based on 77,161 feature vectors, one for each earthquake event recorded by USGS. The user would input their zip code and the model would provide the likelihood of feeling an earthquake and a probable magnitude at the given zip code.

```
Train Accuracy =  0.8866918462563874
Test Accuracy =  0.8840986651691217

Predicting if will feel an earthquake for the lat and lon values: [[32.764998,
-117.073641]]

You will MAYBE feel an earthquake (mag: 2.5 - 3.9) at the zipcode: 92115
```

Using '92115' as a zip code input

D. *K-Nearest Neighbor*

The first problem statement that came to our mind was "what magnitude would an individual feel if an earthquake were to happen at their location?". To find a solution for this question, we implemented a supervised and regression model. We found that KNN would answer this solution.

The k-nearest neighbor model was written in jupyter notebook. The three parameters it takes from the data are latitude, longitude, and magnitude. There are five functions that are used by the main function, and each of these functions is tested individually in the program.

The first function called in the main function is loadDataset, which takes as parameters the name of the file with the data, the training and testing set, and the proportion that goes in the training set. It reads each line in the file specified, and stores the latitude, longitude, and magnitude of 80% of the lines in the training set. The remaining 20% of the lines go in the testing set.

The main function then calls the getNeighbors and getResponse for each line in the testing set. The getNeighbors function takes that line from the testing set, as well as the training set and the k value. With k equal to 3, it finds the 3 lines from the training set with the closest distance to the line from the testing set. It finds the distance by calling the function euclideanDistance and selects the three lines with the smallest distance. These three lines are passed as a set into the getResponse function. The getResponse function returns the average of the magnitudes from the 3 nearest neighbors, which is the predicted magnitude for that line from the testing set. This process is repeated for each line in the testing set, and all of the predictions are stored in a prediction set.

The main function then prints out the predicted magnitude and actual magnitude of each line in the testing set. The testing set and prediction set are then passed into the getError method. The get error method takes the percent error of each magnitude in the testing set with the corresponding magnitude in the prediction set. It then returns the average of the percent errors.

After running the code, as it prints out the predicted values next to the actual values, you can see that it is very accurate at predicting magnitudes.

```
Train set: 12029
Test set: 3020
> prediction = 5.1, actual = 5.1
> prediction = 5.1, actual = 5.3
> prediction = 5.1, actual = 5.1
> prediction = 5.7, actual = 5.7
> prediction = 5.0, actual = 5.0
> prediction = 5.1, actual = 5.0
> prediction = 5.5, actual = 5.1
> prediction = 5.2, actual = 5.2
> prediction = 5.1, actual = 5.1
> prediction = 5.3, actual = 5.5
> prediction = 6.0, actual = 5.9
> prediction = 6.0, actual = 5.9
> prediction = 5.1, actual = 5.0
> prediction = 5.1, actual = 5.0
> prediction = 5.1, actual = 5.0
> prediction = 5.1, actual = 5.1
> prediction = 5.5, actual = 5.7
> prediction = 5.2, actual = 5.2
> prediction = 5.7, actual = 5.8
> prediction = 5.4, actual = 5.5
> prediction = 5.4, actual = 5.6
> prediction = 6.1, actual = 6.8
Average Percent Error: 0.003408648227502922%
```

Using earthquakes-asia.csv dataset

It returns a very low average percent error of approximately .0034%.

2.2 Evaluation Metrics

Before comparing our models, it's important to note that each model helps us find a solution for different problems and questions regarding earthquake predictions. An in depth reasoning behind our logic for using each model for each question was discussed in the previous section. A short summary for each model is stated here for convenience.

PCA: Find which variables are the most relevant in earthquake prediction
Linear Regression: Finding if there is a significant relationship between depth and magnitude of an earthquake
Logistic Regression: Given a user's location, how likely will they will feel an earthquake and what magnitude would be felt
KNN: What magnitude would an individual feel if an earthquake were to happen at their location

Despite all these varying problems and solutions each of our models have found, there are some similarities between them.

We used linear regression and PCA to find trends in our data. Specifically, we wanted to find which features and variables that would be most relevant to our understanding of earthquake prediction. In PCA we wanted to test the relevancy of all features whereas in linear regression we focused on seeing if depth was an important feature in earthquake predicting. Both models revealed to us that the most significant features to consider when predicting earthquakes are latitude, longitude, and magnitude.

Both our logistic regression model and KNN predict a probable earthquake magnitude at a given location. KNN relies on calculating a magnitude based off of the history of the closest previous earthquakes while logistic regression predicts the change of feeling an earthquake along with the probable earthquake's magnitude using a training and testing dataset.

2.3 Analysis and Discussion

A. *PCA*

PCA helps reduce the number of variables in a dataset by finding which variables are the most important [10]. This is very useful in our project because there can be many variables to consider when predicting earthquakes. We approached dimensionality reduction with a feature elimination technique. This model and technique works for us because we picked a few variables that we deemed the most important - latitude, longitude, and magnitude. We were able to apply these features to PCA and see the relevance of them to each other along with other variables like depth, gap, and more. We were able to measure how each variable is associated with one another, find the directions in which our data is dispersed, and found the relative importance of these different directions [10].

B. *Linear Regression*

This model analyzed the relationship between depth and magnitude to see if there was a correlation between them.

This model works for our problem because our problem fits the condition and assumptions for linear regression. We assume that depth and magnitude already have a linear relationship, then we would use the model to test whether or not the linear relationship is significant enough for our variables to be considered dependent of each other. Secondly, both depth and

magnitude are are multivariable normal. Thirdly, there is little or not multicollinearity in our data. Finally, there is little autocorrelation in the data. This last assumption may be debatable since one earthquake may lead to another earthquake [8].

C. *Logistic Regression*

Similar to linear regression, the problem statement we've created for our logistic regression model fits the conditions. The variables we have used for this model is date and time, latitude and longitude, depth, magnitude, and felt or not. Our dependent variable, felt, is binary since an individual will only receive a "felt" or "not felt" prediction. This is why the logistic regression model works for our problem. The observations of our data are independent of each other. However, similarly to linear regression, earthquakes may cause other earthquakes to occur. There is little multicollinearity among the independent variables. We assume the linearity of independent variables and off logs. Finally, we have obtained a large sample size for our research [9].

D. *KNN*

KNN was the first model we found that would fit in our research for predicting earthquakes. The training phase is very minimal and does not make any generalizations [4]. This model works because our problem statement was to find a predicted magnitude an individual would feel at their given location. We based our predictions off of the distance between an individual and previous earthquakes and their magnitudes, thus, we technically don't need to have a training phase, hence why we chose this model.

3. Conclusion and Future Works

1. Conclusion

In our findings of applying machine learning techniques to earthquake predictions, we were able to conclude several things. We found that magnitude, latitude and longitude were the most important variables, among the ones we've tested, when predicting an earthquake's magnitude in our PCA model. We also found that depth wasn't relevant to an earthquake's magnitude in our linear regression model. With our logistic regression model, we are able to find the likelihood of an individual will feel an earthquake and what magnitude will be felt given a zip code. In our KNN model, we implemented the algorithm to predict an earthquake's magnitude at a given location using previous earthquake data.

2. Future Works

In a report by the Centre for Research on the Epidemiology of Disasters (CRED), in the period between 1994 and 2013, there has been 6,873 occurrences of natural disasters worldwide. Their report also claimed that 1.35 million lives were lost, which is almost 68,000 lives average each year [6].

Through our project, we hope that earthquake predicting through the use of machine learning techniques can expand to predict other natural disasters as well. With having the knowledge to know when a natural disaster may occur, and to what degree as to how much damage it can cause, we can be more readily prepared to prevent damages and lives being lost or people being affected.

References

[1] USGS Earthquake Catalog
[2] Scikit Learn Linear Regression Example
[3] PCA Using Python
[4] A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm
[5] NPTEL Dynamics of Earthquake Analysis
[6] PreventionWeb The human cost of natural disasters: a global perspective
[7] Earthquakes: Introduction
[8] Assumptions of Linear Regression
[9] Assumptions of Logistic Regression
[10] A One-Stop Shop for Principal Component Analysis

Author Contributions

All group members has equally contributed to the completion of this problem. They have worked very hard to ensure that the project is completed efficiently and presentable.

Stephen Barrack: Contributed to the design aspect of the program. Coordinated the development of each machine learning model used and proposed several methodologies for developing those models, both implemented and unimplemented. Maintained the repository for the code and instructed the group on using Git in command-line interface and GitHub GUI. Reviewed code to ensure it conformed to regular programming conventions and to keep a standard naming convention.

Julia Cai: Organized the meetings for the group project and created the role list to clarify which tasks that need to be completed and make the workload more evenly divided among all the group members. Julia implemented the Logistic Regression model onto the US earthquake data. Julia also wrote the abstract for the paper and information on logistic regression.

Jake Champagne: Worked on the Linear Regression model as well as attempting the Naives Bayes model. He also put together graphs which we used to analyzed the time of day of an earthquake as well as the month of year. He organized the csv files after retrieving the data from the USGS earthquake catalog.

Joseph Couri: Designed and coded the KNN model. Also helped to plan out the models we will use and how to code them. Joseph also attempted to implement the SVM model.

Preethi Narayanan: Implemented PCA on earthquake data to find the most relevant variables. Also attempted feature selection to find other important variables using Kbest and other techniques. Preethi also worked on the Naive Bayes model. She also wrote the PCA and introduction part of the report.

Megan Respicio: Pitched the initial idea of our subject being natural disaster predictor but then narrowed down to earthquake predicting. Created the zip-code lookup function for the logistic model. Megan has also made large contributions to the report in writings and proofreading to ensure that the report covers everything needed to be said in a professional manner.