# Earthquake Prediction Model using Python

Here is where your presentation begins

## Problem Statement

Explore the key features of earthquake data and design an object for those features, such as date, time, latitude, longitude, depth, and magnitude. Before developing the prediction model, visualize the data on a world map to display a complete overview of where the earthquake frequency will be higher. Split the data into a training set and a test set for validation. Lastly, build a neural network to fit the data from the training set.

# Earthquake Prediction - the problem

- The physics that control earthquakes are at this time poorly understood
- Good Sensors are hard to deploy
- Earthquakes are very long term - it is hard to collect large amount of statistics

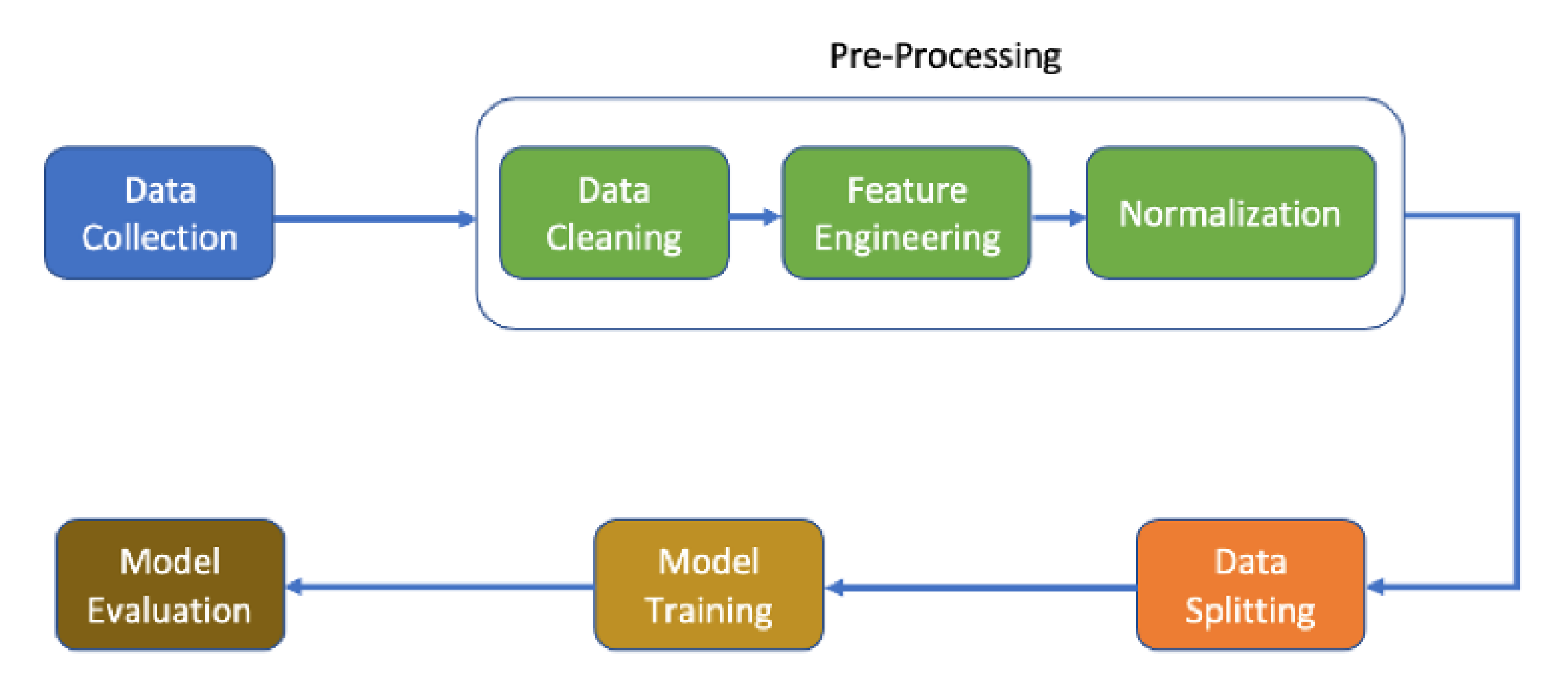
# Our Approach

- Changes in the eather and Seismic Events are very long term - use long term information.
- Good features are extracted by seismologists - use them.
- Use a more visual presentation of the data in order to choose the location , features and classifiers to work with.

# Hypothesis

- We Assume two main types of events:
- A periodic release of pressure - where the local region of the earth is in a “loop” and releases its pressure periodically.
- A sudden release of a large amount of pressure - this event happens after a period of local “silence”.  The earth can not release the pressure because of some blocking, pressure is accumulated, and then in one instance a large amount of pressure is released. This usually causes a large earthquake.

# Flow of the Methodology



# Data Collection

- We have gathered our dataset from the publicly available domain Kaggle.
- In particular we have used the "Significant Earthquakes, 1965-2016" dataset from Kaggle.
- It includes a record of the date, time, location, depth, magnitude, and source of every earthquake with a reported magnitude 5.5 or higher since 1965.

# Data Preprocessing

- Several methods to clean the data were implemented to preprocess the data before use.
- The data was preprocessed by handling missing values, then scaling and normalization were done.
- The dates in the data were parsed, character encoding was done and inconsistent data entry was handled.

# Methodology

- The data was divided (after randomly shuffling it) into two separate sets:
- A training set (80%) - Used for constructing the classification model
- A testing set (20%) - Used for testing the model
- The model was constructed using only the training set, and tested on the testing set

# Methodology

- We have chosen to use the Random Forest algorithm to build our data model.
- Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result.

# Methodology

- We have chosen to use the Random Forest algorithm to build our data model.
- Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result.

# Methodology

- Here are the four steps in the Random Forest algorithm:
- 1. Choose a random bootstrap sample of size n (randomly taken n samples from the training set with replacement).

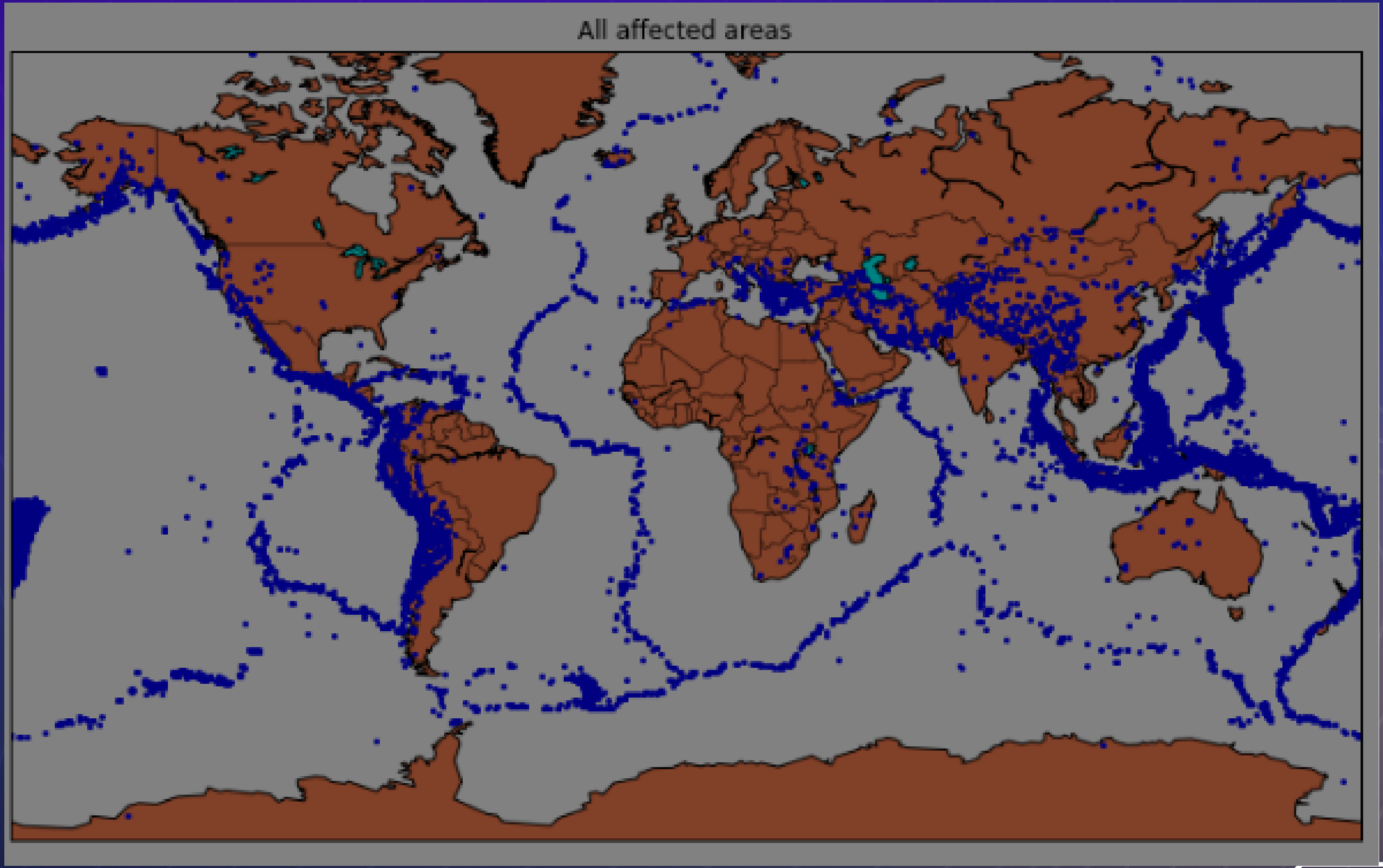
# Methodology

- 2. Build a decision tree from the bootstrap sample. On each node:
- a. Randomly select d features without replacement
- b. Split the nodes using the feature that provides the best split according to the objective function.
- 3. Repeat steps 1) and 2) k
- 4. Combine predictions based on each tree to assign a class label based on the most votes.

# Data Visualization

- We have built a heatmap of the earthquake prone zones on the world map using data visualization methods provided in Matplotlib.
- The heatmap can be seen in the following slide.

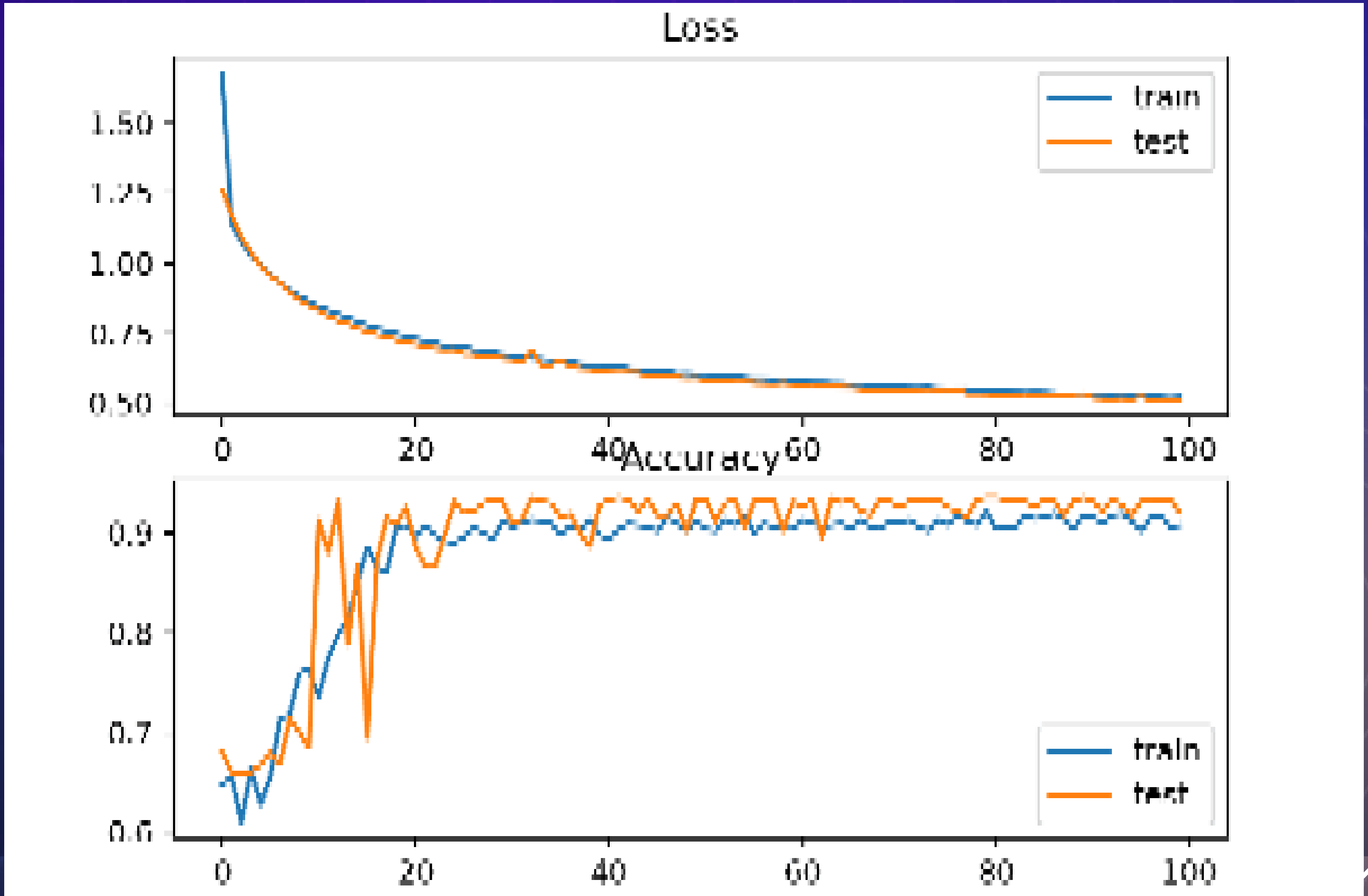
# Data Visualization



# Data Model

- The Random Forest data model was built using Tensorflow and Keras over 20 epochs of training.
- We were able to achieve an accuracy of 92.41% and a loss of 0.50 in our testing.
- The accuracy and loss graph can be seen in the following slide.

# Data Model



Loss
train
test
1.50
1.25
1.00
0.75
0.50
0
20
40
60
80
100
Accuracy
train
test
0.9
0.8
0.7
0.6
0
20
40
60
80
100

# Conclusion

- We have successfully built a data model for the prediction of earthquakes in a given area with the latitudinal and longitudinal data of the area.
- The model performs well in our test with an accuracy of about 92.41%.
- This could prove to be helpful for people deciding what cities to move into knowing the safety issues in certain areas.