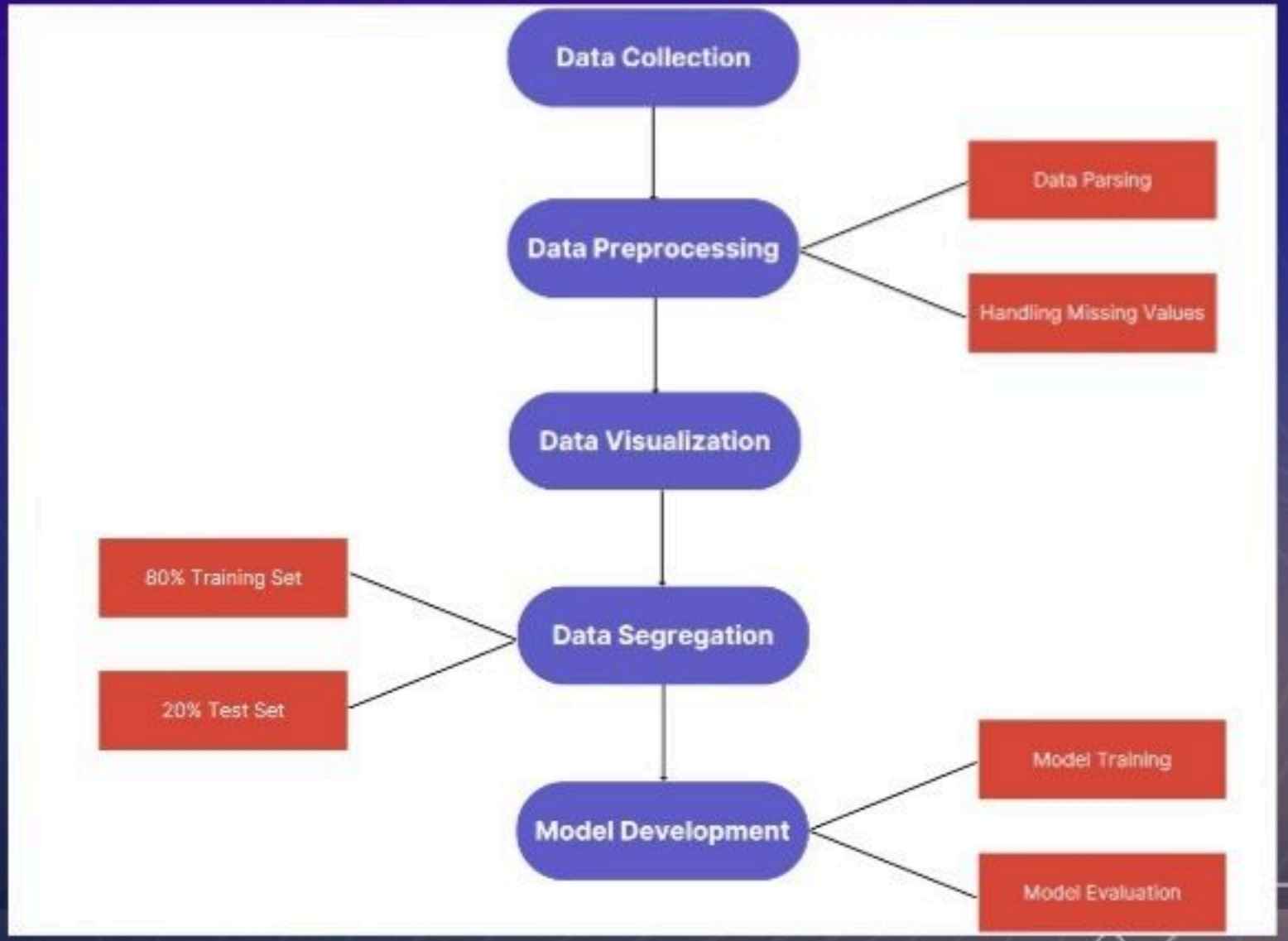

Earthquake Prediction Model using Python – Steps

Introduction

- The earthquake prediction data model aims to predict the likelihood of an earthquake occurring in various regions in the world.
- We have collected a publicly available dataset and have built two data models to perform ensemble learning on the dataset.
- The steps followed in our methodology are outlined in this presentation.

Flow of the Methodology



Data Collection – Step 1

- We have gathered our dataset from the publicly available domain Kaggle.
- In particular we have used the “Significant Earthquakes, 1965–2016” dataset from Kaggle in the CSV format.
- 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 – Step 2

- 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 was parsed, and inconsistent data entry was handled.

Datetime Parsing

- An environment was setup in form of a python notebook file.
- The required modules (Pandas, Numpy, Seaborn, Datetime) were downloaded and imported into the environment.
- The date columns were converted into datetime.

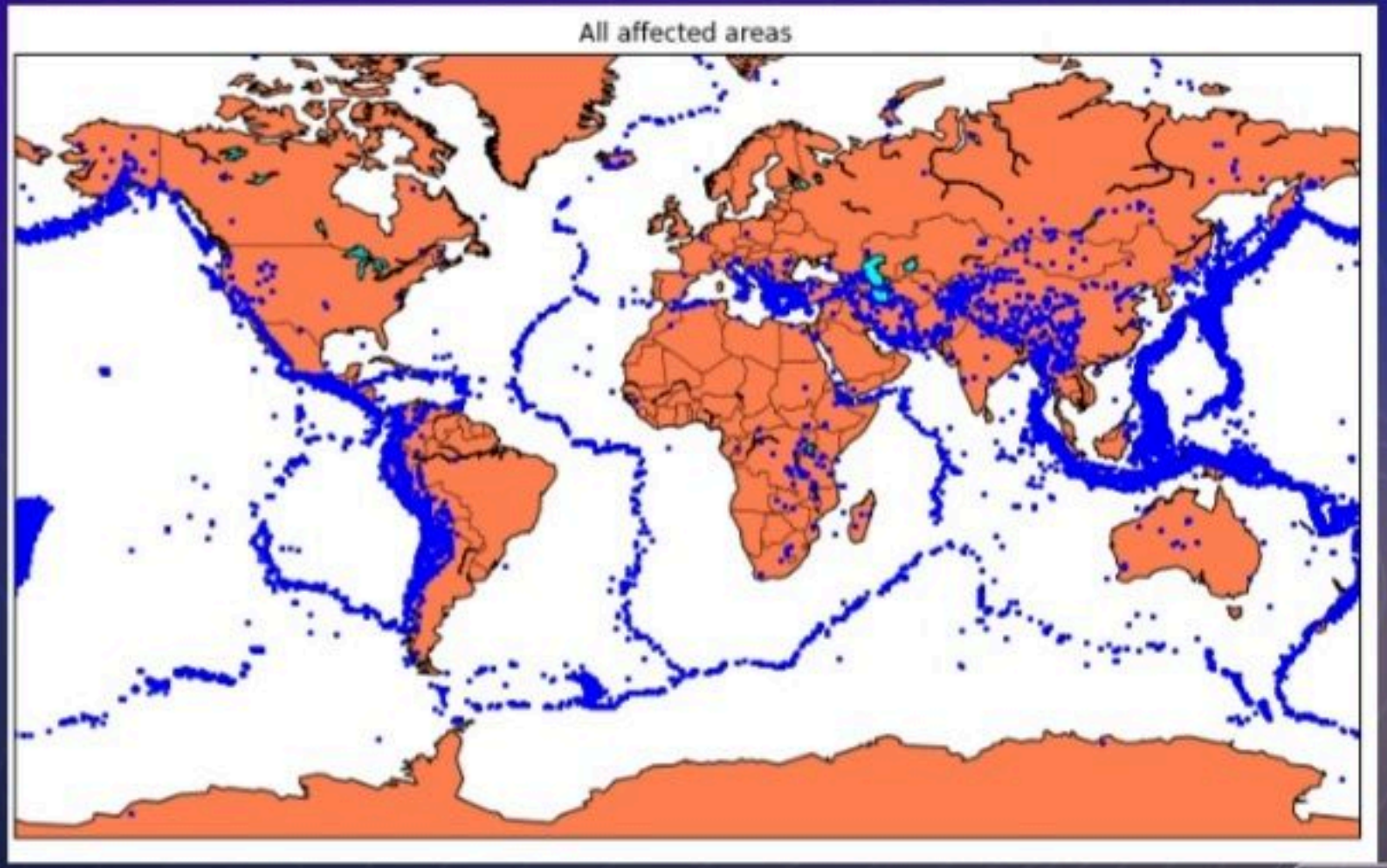
Handling Missing Values

- The pandas module was used to load the dataset.
- A check was done to check for missing values.
- The missing values were replaced with the mean or most frequent value per each column, depending on the exact data that was missing.

Data Visualization – Step 3

- 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.

Visualization of Affected Areas



Data Segmentation – Step 4

- After being randomly shuffled, the data was split into two distinct sets:
- A training set is used to build the classification model (80%).
- A test group (20%) – Used to evaluate the model.
- Only the training set was used to build the model, and the testing set was used to evaluate it.

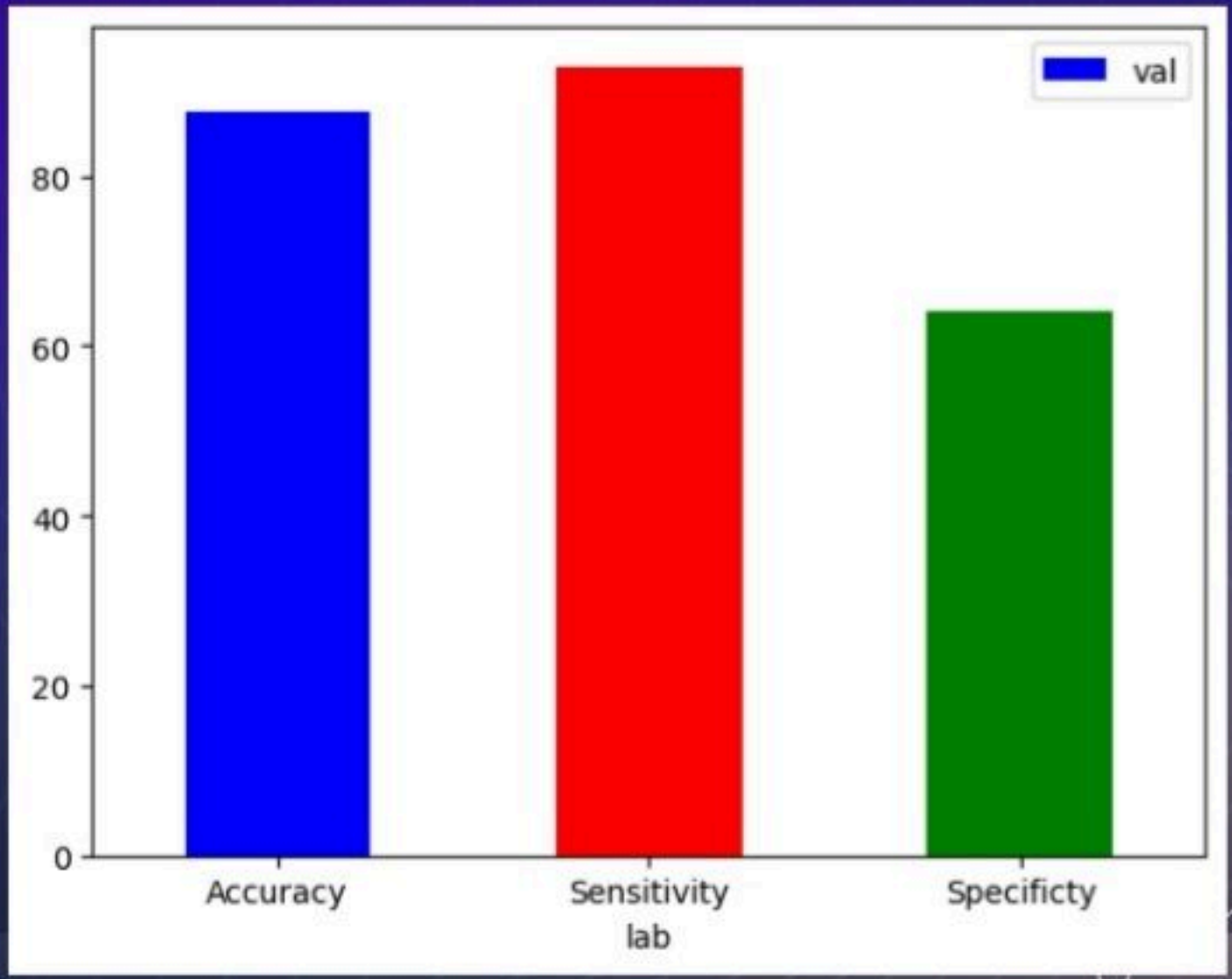
Training on the Data – Step 5

- To create our data model, we have opted to use the Random Forest algorithm and a Neural Network algorithm.
- Leo Breiman and Adele Cutler are the creators of the widely used machine learning technique known as random forest, which mixes the output of several decision trees to produce a single outcome.

Random Forest Algorithm Implementation

- We have used Scikit-Learn Python module to train our dataset using the Random Forest algorithm.
- We have fit the X and Y values on the regressor and built the data model and test it against the test dataset to achieve an accuracy of 87.49%, a sensitivity of 92.9% and a specificity of 64.1% as seen in the graph in the following slide.

Random Forest Algorithm Test



Neural Network Algorithm

- We have used the Keras library to build a Neural Network for the dataset.
- A neural network is a collection of algorithms that aims to identify underlying links in a piece of data using a method that imitates how the human brain functions.

Neural Network Algorithm Implementation

- We have created a sequential model which a linear stack of layers for the neural network.
- The first layer is a Dense layer with neurons neurons, using the specified activation function (activation). It also specifies an input shape of (3,), which means that the model expects input data with three features.

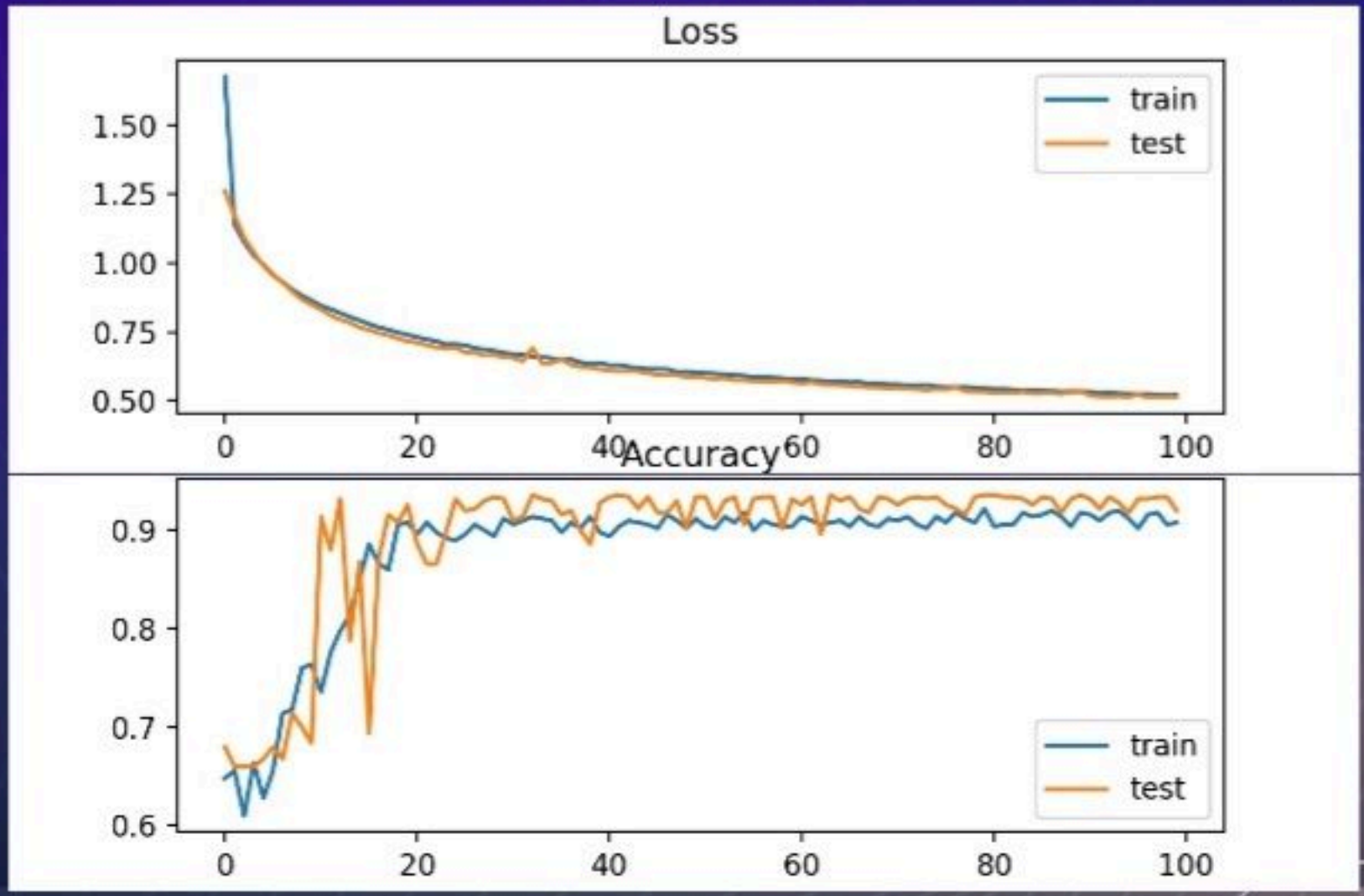
Neural Network Algorithm Implementation

- The second layer is also a Dense layer with neurons neurons and the same activation function.
- The third layer is a Dense layer with 2 neurons (assuming you are building a classification model) and uses the softmax activation function, which is commonly used for multi-class classification problems.

Neural Network Algorithm Implementation

- We have designed the grid search to consider two optimizer choices: Stochastic Gradient Descent (SGD) and Adadelta.
- We then train the data model for 20 epochs with a batch size of 10 and test it against the test dataset to achieve an accuracy of 92.42%.

Neural Network Algorithm Test Graphs



Conclusion

- We have collected and preprocessed a publicly available dataset for the prediction of earthquakes and have visualized the data on a world map.
- We have successfully built two data models for the prediction of earthquakes in a given area with the latitudinal and longitudinal data of the area.
- The neural network data model is able to outperform the random forest data model.
- The neural model performs well in our test with an accuracy of about 92.41%.