Classification Project

Data Analytics I

Atirek Kumar | 20171060
Debojit Das | 20171129

The problem

In this classification project we build a classifier for predicting earthquakes in a geographical region using KNN and decision trees and determine which works better.

WHAT WE DID

1. Removing extra information

We removed the initial rows from the csv file containing the metadata to make the row containing the labels as the first row. Then using pandas we also removed the following:

- The 2nd row which contained the sub-labels such as **Mw**, **Mb**, etc.
- Removed all the columns which do not contribute to predicting an earthquake. That included all columns except:
    - `LAT (N)`
    - `LONG (E)`
    - `DEPTH (km)`
    - `MAGNITUDE` (the result to be predicted)

2. Cleaning the data

To use the longitude and latitude as numbers, they had to be converted from type **string** to **float**. To do this special symbols such as degree(**°**) and alphabets indicating the directions **N**,**S**,**E**,**W** had to be removed and the value had to be changed accordingly.

Rows containing **NaN** or **infinity** had to be iteratively checked and removed which brought the number of rows in the dataset from **52989** to **40107**.

Taking the threshold value as **4.20**, the entries in **MAGNITUDE** column were changed to 0 or 1 depending on whether the threshold value.

3. Splitting the data

First the data is divided into **X** (the parameters latitude, longitude and depth of earthquake which determine whether an earthquake occured) and **y** (the value to be determined). Then **X** and **y** are split into training data and testing data in a **70:30** ratio.

4. K Nearest Neighbours (KNN)

The first classifier used on the dataset is K-nearest neighbours. Using the **KNeighborsClassifier** from **sklearn** module we use the K-nearest neighbours method to train the training set using **knn.fit(X_train, y_train)** and then predict the result on the test set. For **K=3** or **K=4** we got the highest AUC score given the chosen threshold.

5. Decision tree

The second classifier used on the dataset is a decision tree. Using the **DecisionTreeRegressor** from **sklearn** module we make a decision tree using the training set and then predict the result on the test set. For **depth=25** we got the highest AUC score given the chosen threshold.

5. ROC curve

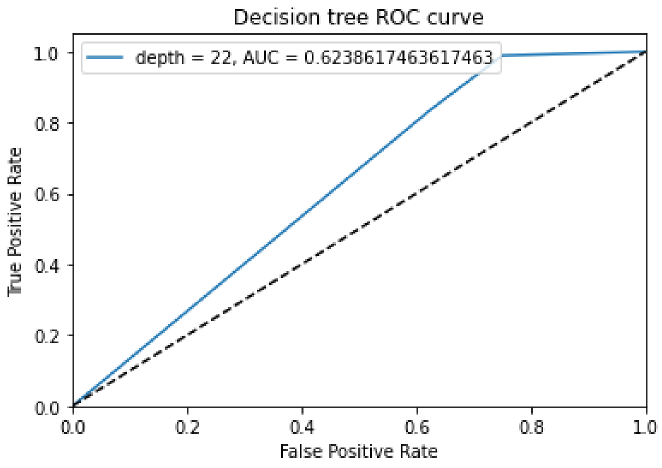
After determining the best values of **K** and **depth** we plot the **ROC curve** after training the dataset and testing it for different values of parameters **K** and **depth** for **KNN** and **Decision Tree** respectively.

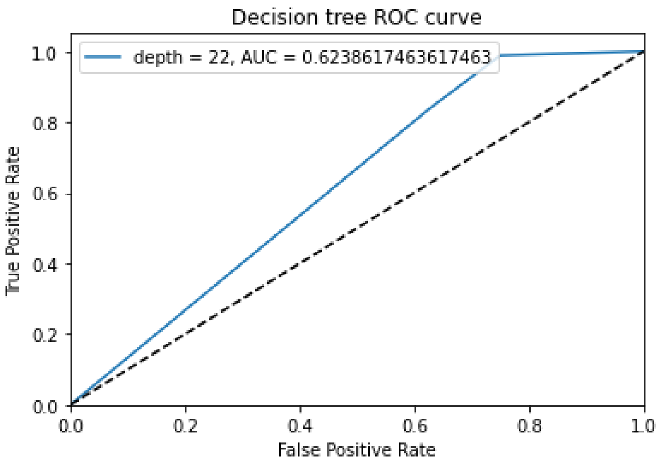
TASKS

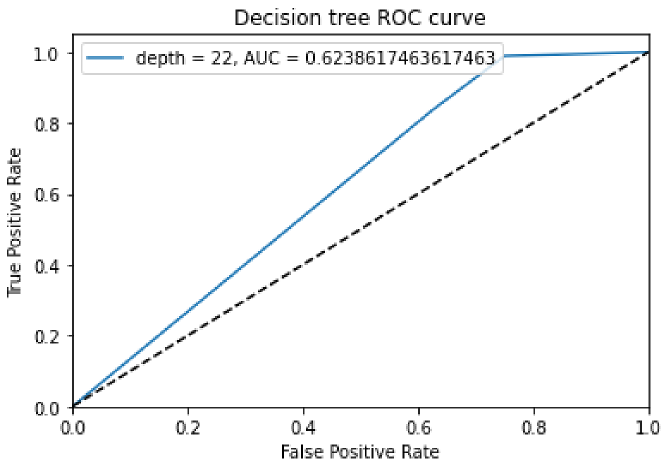
We used a threshold of magnitude 4.20 to determine whether an earthquake is major or minor. Earthquakes with magnitude more than 4.20 were assigned the label 1, and the ones with magnitude less than that were assigned the label 0.

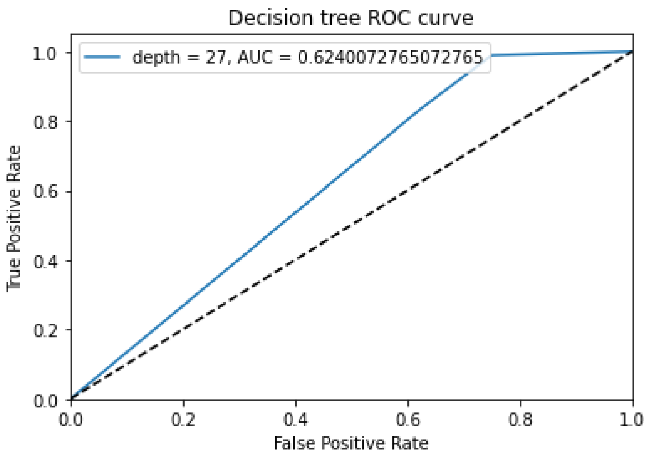
Our results are as follows:

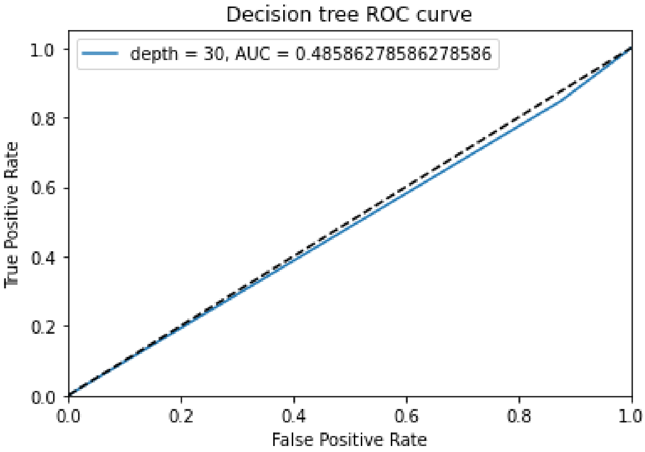
1.

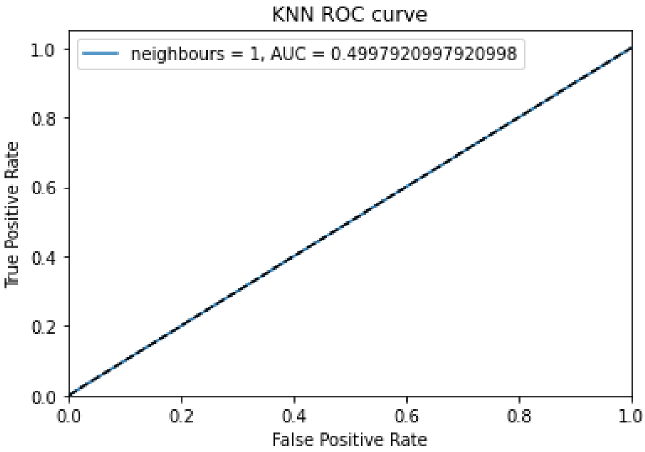


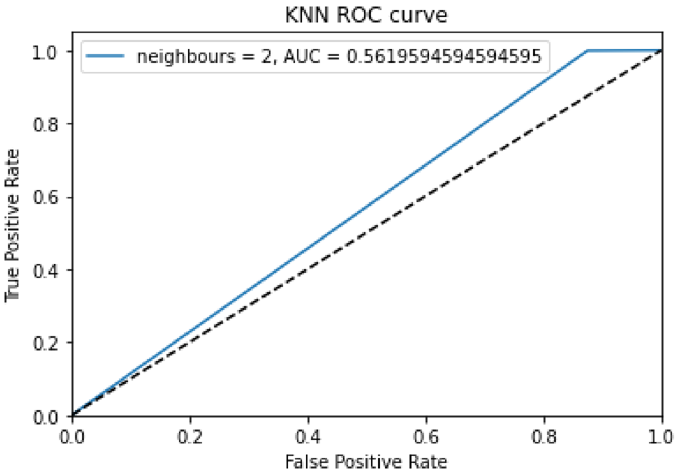


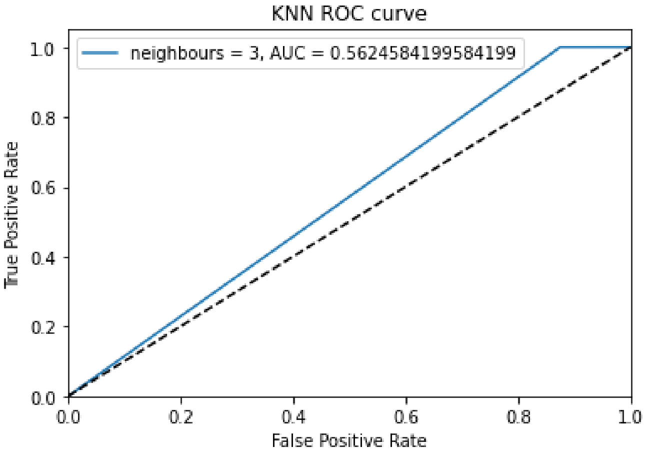


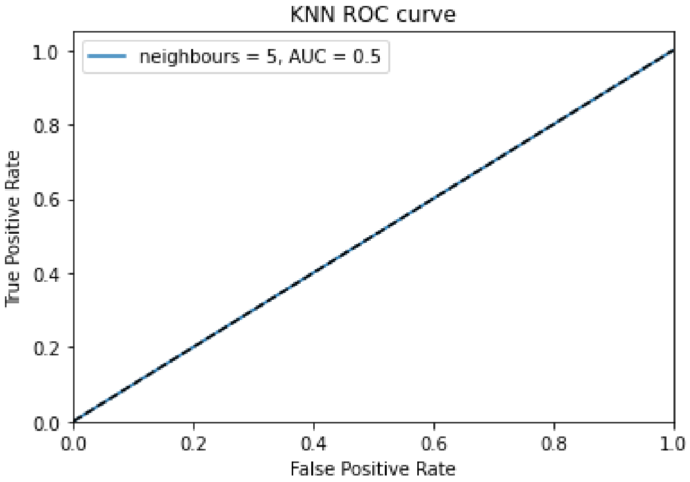


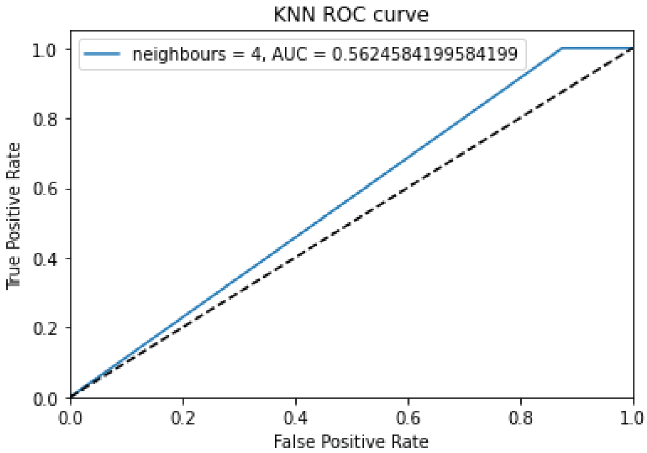

Decision tree ROC curve
depth = 30, AUC = 0.48586278586278586
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


KNN ROC curve
neighbours = 1, AUC = 0.4997920997920998
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


KNN ROC curve
neighbours = 2, AUC = 0.5619594594594595
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

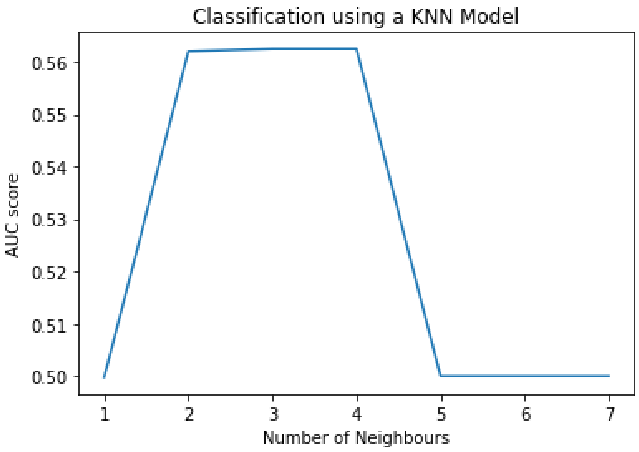

KNN ROC curve
neighbours = 3, AUC = 0.5624584199584199
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


KNN ROC curve
neighbours = 5, AUC = 0.5
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

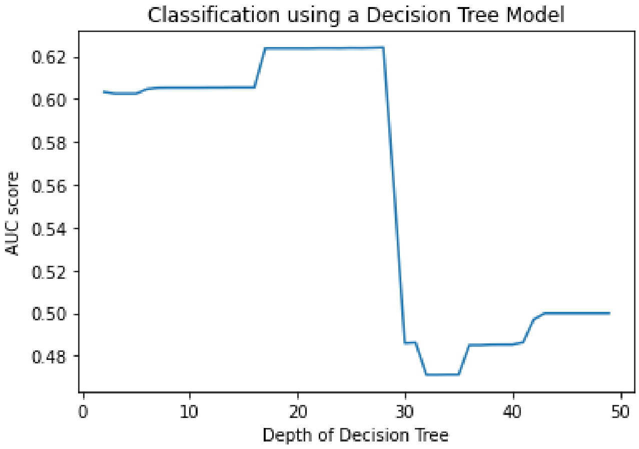

KNN ROC curve
neighbours = 4, AUC = 0.5624584199584199
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

2. The **Decision Tree model seems to perform better than the KNN model** for given data. KNN is designed to work with distances, and in our case we can not design an efficient measure for distances since we are working with different types (coordinates, depth). Hence, Decision Tree gives better results since it does not rely on any such relations.

3. For **KNN**, **K=3 and K=4** seem to be giving the best results. Given the diversity of the given case, K > 4 leads to overfitting. K < 3 leads to underfitting.



For **Decision Tree**, **depth=25** seems to be giving the best result. Depth of 18-27 performs well, but increasing the depth beyond 27 leads to overfitting. Depth < 18 leads to underfitting.



4. Experimentally (using Decision Trees), we found that Latitude and Depth have the most impact in predicting the Magnitude of an earthquake. Longitude and Depth also give good predictions. Clearly, depth is the most important factor and that was expected since Latitude and Longitude are both required, together, to know the location, and hence knowing only one is not enough information. Depth, however, gives us a lot of information since the magnitude depends directly on depth, and indirectly on latitude and longitude (it depends directly on the location).

5. We used multiple techniques to account for the missing data. First of all, we only considered Latitude, Longitude, Depth and Magnitude, since the other fields do not have much significance in predicting magnitude, and can very easily lead to overfitting. We tried the following techniques on Decision Tree model:
    a. Removing columns which have null values (we soon realised it leaves us with an empty dataset)
    b. Removing rows which have null values (this performed the best for us)
    c. Taking mean for all values that were missing (roc = 0.58)
    d. Taking median for all values that were missing (roc = 0.56)
    e. Taking most-frequent for all values that were missing (roc = 0.53)
    f. Taking mean for all values that were missing, but taking 180° as default for Longitude and Latitude if missing (roc = 0.58)