

DATA ANALYTICS

PROJECT 1 - CLASSIFICATION

Anishka Sachdeva - 2018101112
Satyam Viksit Pansari - 2018101088

September 9, 2020

I. PROBLEM STATEMENT

Given a dataset about occurrences of earthquakes in a geographical region, use **KNN** and **Decision Trees** to build a classifier for predicting labels as Magnitude(Mw). Use an appropriate threshold that seem fit between [4, 5] inclusive. Consider a threshold of T . For $Mw < T$, label becomes 0 (no earthquake) and for $Mw \geq T$ becomes 1 (earthquake). Use appropriate features as input from the dataset that you seem fit. Mention the threshold used and what is the train-test split size used.

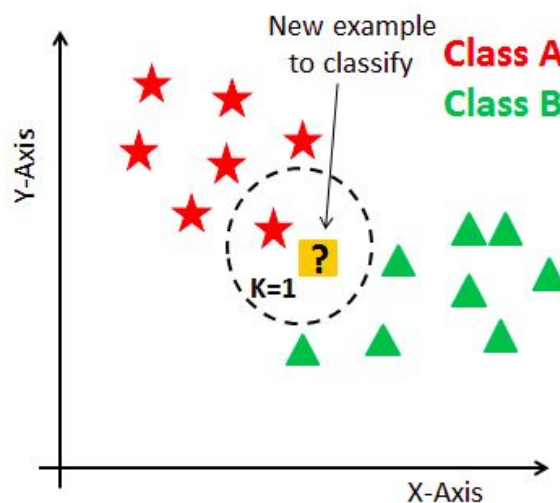
II. META DATA

1. **Sl. No.** : Serial Number
 2. **Year, Month, Day** : Date of a particular earthquake as per UTC (Coordinated Universal Time)
 3. **Origin Time** of earthquake in UTC and IST (Indian Standard Time) in [Hour: Minute: seconds] format
 4. **Magnitude of Earthquake** : There are different ways to represent the magnitude of an earthquake. For your study, you can consider Mw, since we are deriving other types from Mw only.
 5. **GPS Location** in terms of Latitude(Lat) and Longitude(Long) of earthquake
 6. **Depth** : Depth of occurrence of an earthquake in kilometre
 7. **Location** : Name of a region where an earthquake took place
 8. **Source** : The agency from which we have gathered the data, for e.g. IMD= Indian Meteorological Department, Min. of Earth Science, Government of India
-

III. ALGORITHMS USED

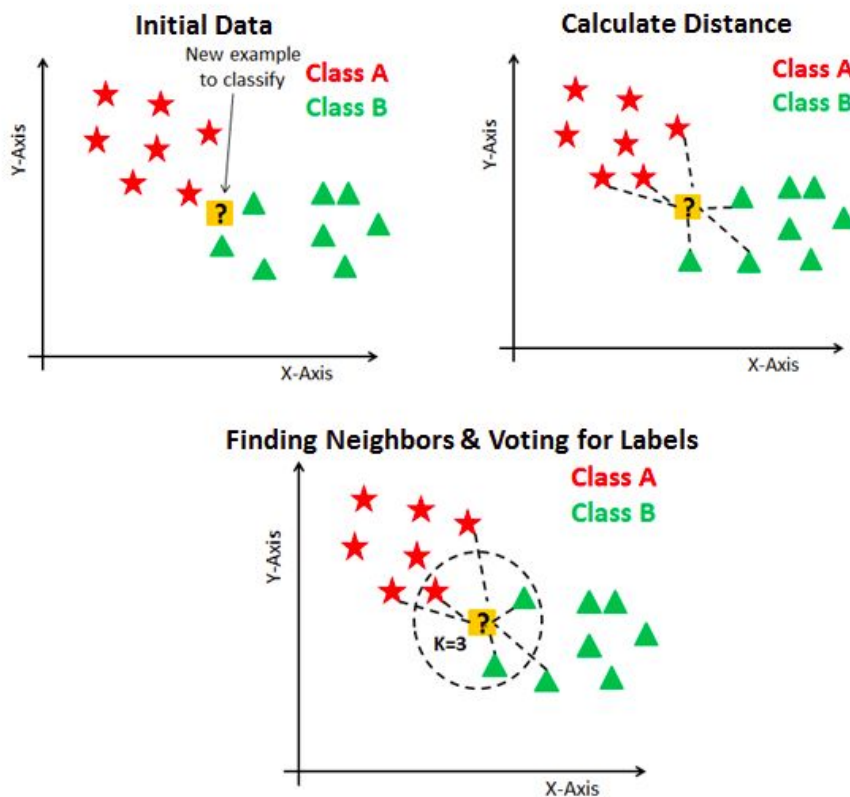
1. KNN (K Nearest Neighbours) Algorithm

In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2. When $K=1$, then the algorithm is known as the nearest neighbor algorithm. This is the simplest case. Suppose P_1 is the point, for which the label needs to predict. First, you find the one closest point to P_1 and then the label of the nearest point assigned to P_1 .



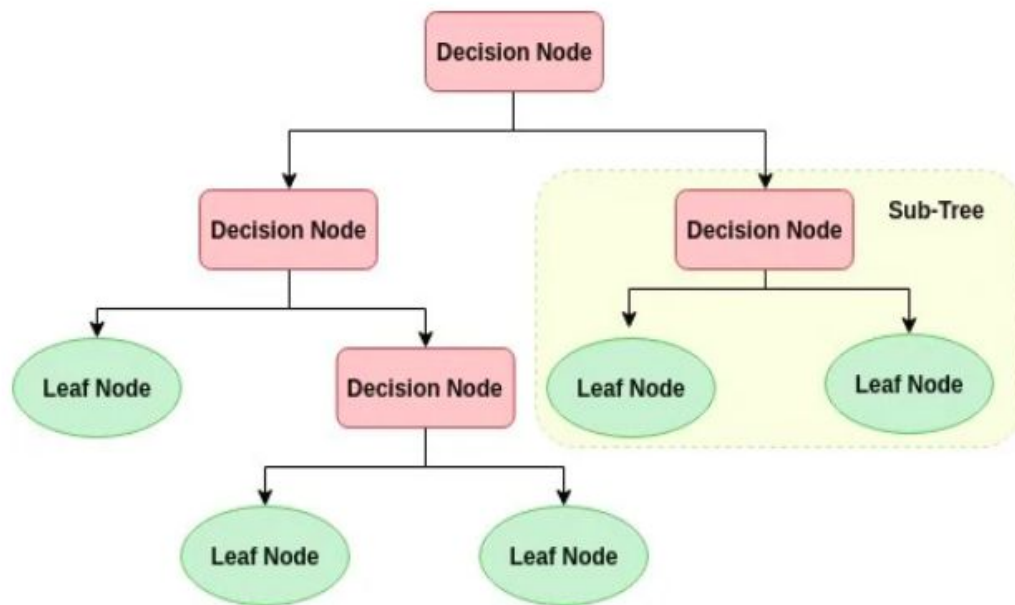
Suppose P_1 is the point, for which the label needs to predict. First, you find the k closest point to P_1 and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction. For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance. KNN has the following basic steps:

- Calculate distance
- Find closest neighbors
- Vote for labels



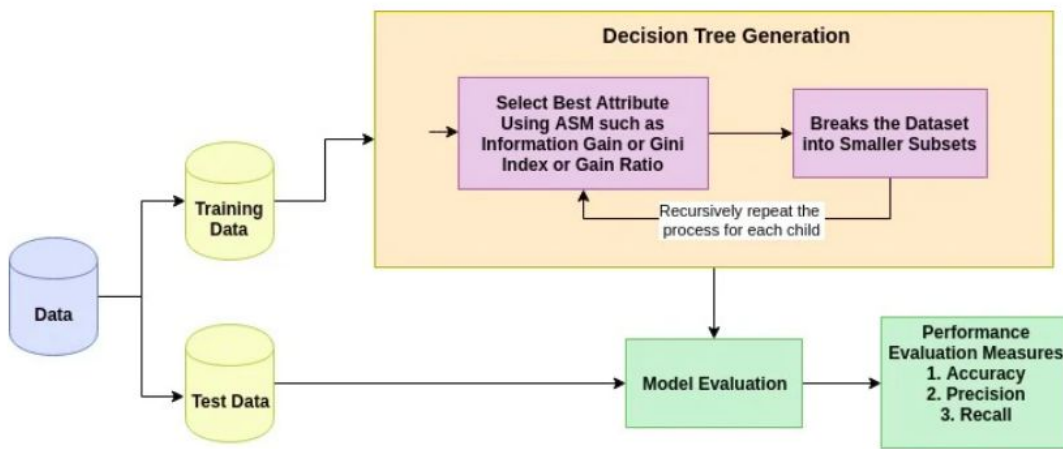
2. Decision Tree Algorithm

A decision tree is a flowchart-like tree structure where an internal node represents a feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in a recursive manner called as recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.



The basic idea behind any decision tree algorithm is as follows:

- a. Select the best attribute using Attribute Selection Measures(ASM) to split the records.
- b. Make that attribute a decision node and break the dataset into smaller subsets.
- c. Starts tree building by repeating this process recursively for each child until one of the condition will match:
 - All the tuples belong to the same attribute value.
 - There are no more remaining attributes.
 - There are no more instances.



IV. LIBRARIES USED

1. Pandas

Pandas is an open source library in Python. It provides ready to use high-performance data structures and data analysis tools. Pandas module runs on top of NumPy and it is popularly used for data science and data analytics. It allows us to store and manipulate tabular data as a 2-D data structure.

2. Scikit

Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machines, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy.

3. Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

V. DATA PREPROCESSING

In the original dataset the first few rows had to be removed as it contained information about the table which was not required . That was preprocessed and then the following other preprocessing, cleaning and analysis of the dataset was performed :

We created a dataframe by reading the .csv file (dataset). We calculated the sum of all NULL values present in each feature /column of the dataset. The following results were noticed :

Sl. No.	0
Year	0

Month	18
Date	57
Mw	2504
Lat (N)	0
Long (E)	0
Depth (km)	2178
Reference	1582

1. On analyzing the above values, we noticed that there were NULL and 0 values for **"MW"** for some entries. Since the data set is very large compared to the number of null values, we decided to remove those rows.
2. Secondly, we noticed that for the feature **"MONTH"**, there were only 8 NULL values and hence it seemed to be an important feature. So we removed the rows where NULL Values occurred as this feature could probably help in predicting better models.
3. We also noticed the same case with the **"DATE"** feature. So we removed the NULL Values.
4. In the **"MONTH"** feature, the values were in string instead of integers 1-12 (For example, the months were given strings like 0,00 ,05 etc. which is not the right format of it.) So **DATA CLEANING** had to be done in the following manner :
 - We first checked how the Month values looked like.
 - We checked for each month how many entries were present so we noticed there were months namely "0, 00 and null" which were just garbage values in the dataset. These three values held less number of entries (around 35). Thus, we could remove them.
 - Then we converted string to float to match issues like 1 and 01 .
 - Decimal points were removed by converting the float values to integer values.

After the above steps, the dataset had the following sum of NULL values:

Sl. No.	0
Year	0
Month	0
Date	0
Mw	0
Lat (N)	0
Long (E)	0
Depth (km)	1251
Reference	1582

5. There are around 2000 values (which after the above modifications were left with around 1251 null values) entries which are null in the '**DEPTH**' feature, these need to be filled with some value (**Feature Engineering Method of missing values**). Thus we used describe() to analyse the values for the Depth Feature. Following were the results :

count	49185.000000
mean	45.677553
std	56.108936
min	0.000000
25%	10.000000
50%	33.000000
75%	55.300000
max	831.200000

Since the data has a variance of about 56, which is very high, filling it with the mean or median values will not be the right method so it's better to just drop them.

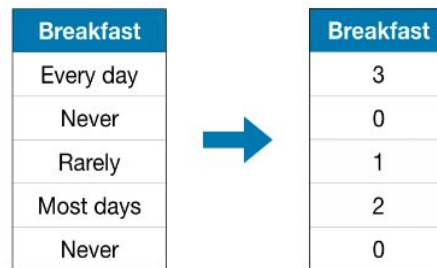
6. For Latitude and Longitude, we just removed the symbols and converted to float.
7. We used the **feature engineering concept of label encoding (for categorical variables)** for the '**REFERENCE**' feature as the reference values were strings/object type. Thus, for training and predicting models, these

entries could be label encoded as the distinct entries for this feature contained 21 features.

Label encoding can be explained in the following manner :

2) Label Encoding

Label encoding assigns each unique value to a different integer.



The diagram illustrates the process of label encoding. It consists of two tables connected by a blue arrow pointing from left to right. The left table, titled 'Breakfast', has five rows with the following categorical values: 'Every day', 'Never', 'Rarely', 'Most days', and 'Never'. The right table, also titled 'Breakfast', has five rows with the corresponding numerical values: 3, 0, 1, 2, and 0. This shows how each unique category is mapped to a specific integer.

Breakfast
Every day
Never
Rarely
Most days
Never

Breakfast
3
0
1
2
0

This approach assumes an ordering of the categories: "Never" (0) < "Rarely" (1) < "Most days" (2) < "Every day" (3).

This assumption makes sense in this example, because there is an indisputable ranking to the categories. Not all categorical variables have a clear ordering in the values, but we refer to those that do as **ordinal variables**. For tree-based models (like decision trees and random forests), you can expect label encoding to work well with ordinal variables.

8. Other features/columns which had data type as "Object" were also converted to their equivalent label encoded numerical type.
9. But later, we realised that the '**REFERENCE**' feature can be dropped from the dataset as occurring of an earthquake is not much related to the Organisation that reports its occurrence (We tried to train the model with and without the Reference Feature and the accuracy was almost similar. Thus, it was a good approach to drop this particular feature while training the model.)
10. We had also dropped the '**DATE**' feature because on getting the correlation matrix, it gave a lower correlation value with MW (target label). So we decided to remove that feature for the final model.

11. Threshold Chosen : 4.5

Reasons :

- a. 4.5 is the median of the values of MW. Thus, median is the middle entry in the entire data entries (sorted manner). Hence, it was ideally the correct value to be chosen for the threshold.
- b. We tried lower values of threshold such as 4. But we observed that though we achieved better accuracy it was only because of a huge difference (there were around 10 times more 1s than 0s) in the number of 1s and 0s. So, when we used 4.5 as the value of threshold, we got an even/equal distribution of 0's and 1's about the threshold i.e. a more balanced division of 0s and 1s. This ensured that it wasn't biased towards 0 or 1 and hence helped avoiding overfitting of the model. Other values of threshold showed a bias towards 0 or 1.

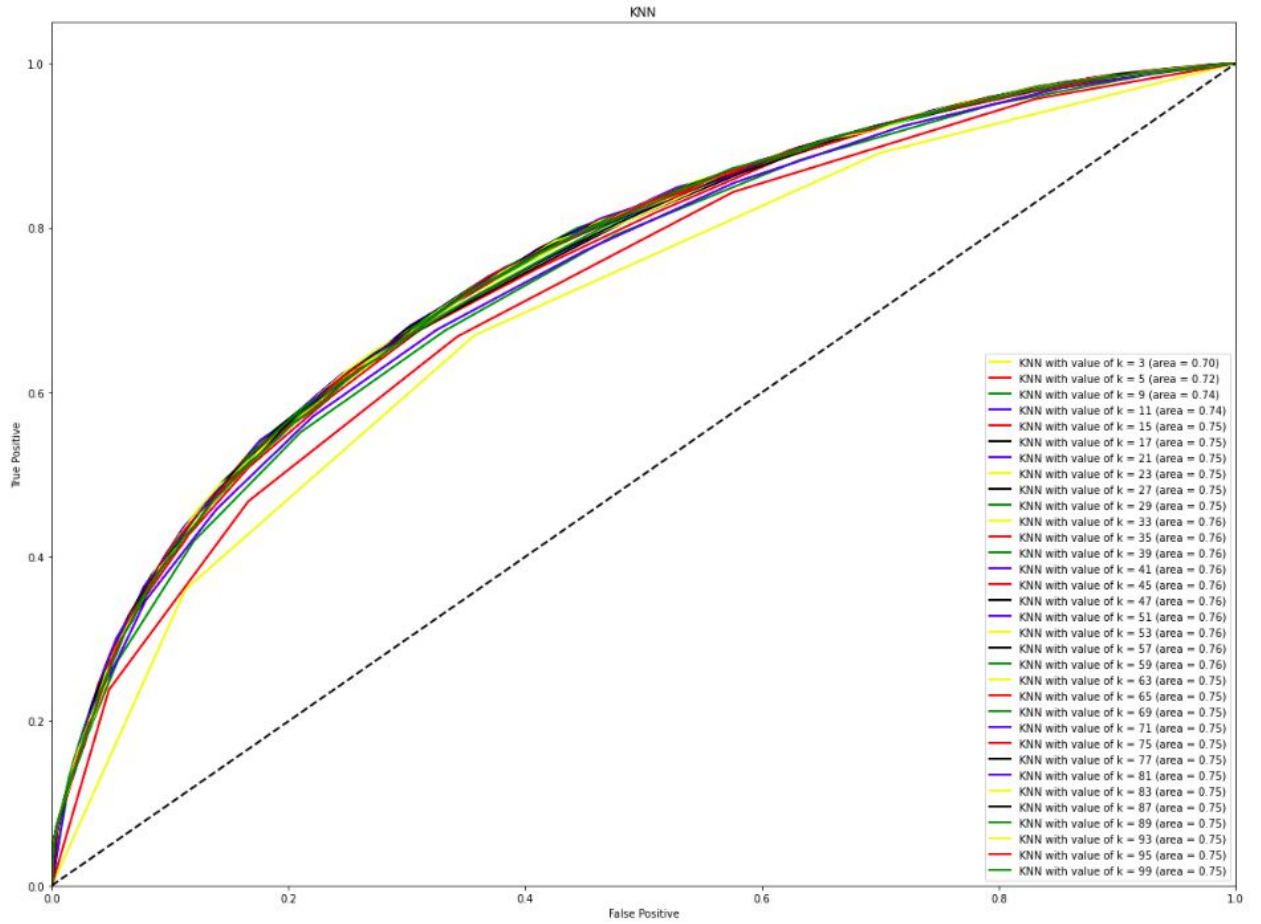
12. Training and Test Data split : **0.75 & 0.25**

VI. TASKS

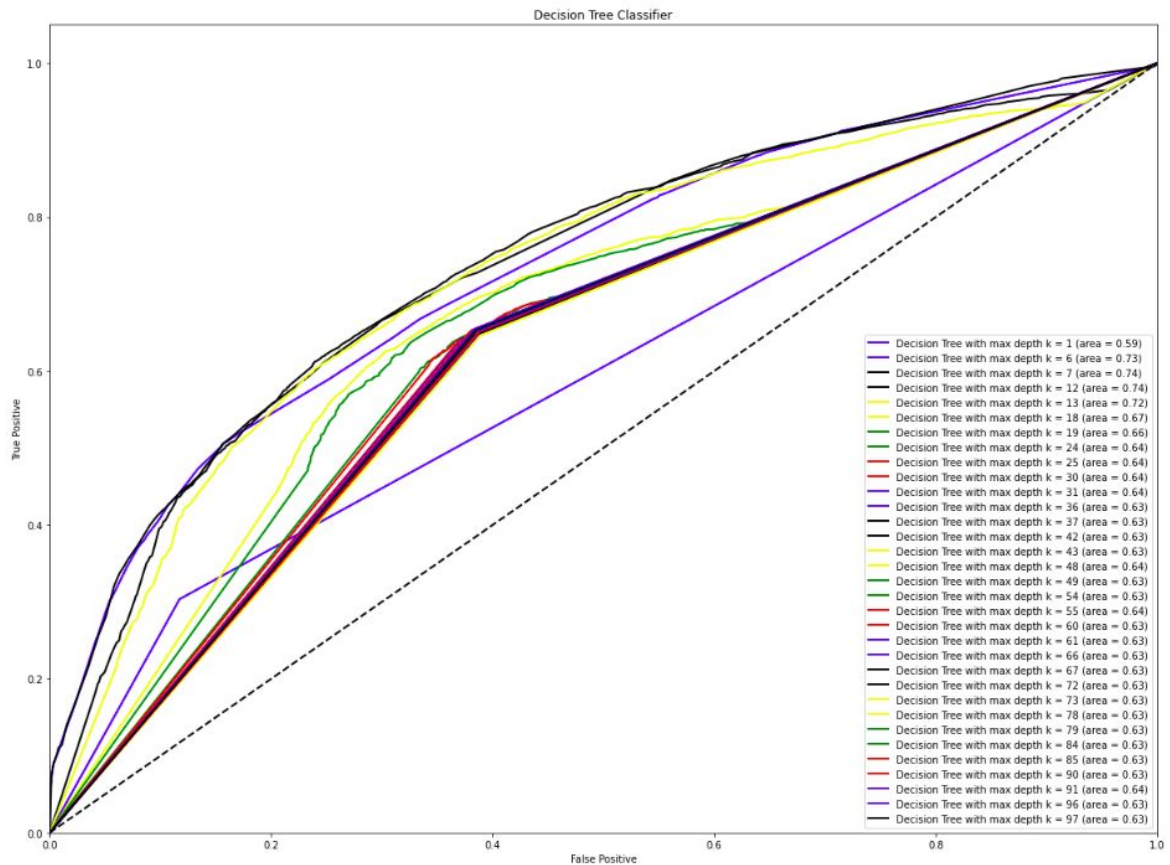
1. Plot ROC for both these classifiers for K as parameter in KNN and pre-prune depth as a parameter in the Decision Tree.

Answer :

1. ROC for KNN for different multiple values of K



2. ROC for Decision Tree with different multiple Maximum Depth Values



2. Which is the better classifier for this data amongst the two? Give Reasoning.

Answer :

Though AUC (Area Under Curve) values were very similar for both the models the metrics for KNN were a little more promising than the Decision Tree . The reasons for this we think are :

1. It would keep data points with similar features relatively close.

For Example : Instances with the **same year** or **location** (latitude & longitude) will be relatively closer which of course is a good thing in the given scenario.

2. In a scenario of a natural disaster like an earthquake, the **recall value** is the preferred metric and recall value for KNN was better than Decision Tree.

Decision Tree					
[[4161 2777]					
[2667 4676]]					
0.6187942020866886					
	precision	recall	f1-score	support	
0	0.61	0.60	0.60	6938	
1	0.63	0.64	0.63	7343	
accuracy			0.62	14281	
macro avg	0.62	0.62	0.62	14281	
weighted avg	0.62	0.62	0.62	14281	

KNN					
[[4812 2126]					
[2407 4936]]					
0.682585253133534					
	precision	recall	f1-score	support	
0	0.67	0.69	0.68	6938	
1	0.70	0.67	0.69	7343	
accuracy			0.68	14281	
macro avg	0.68	0.68	0.68	14281	
weighted avg	0.68	0.68	0.68	14281	

Thus, KNN Model is a better classifier as compared to Decision Tree Model. This can be seen in the metrics calculated above.

3. What could be the best possible values of the parameters for each classifier based on the ROC curves? Give Reasoning.

Answer :

For KNN :

K = 37

Reason : Maximum area **(0.76)** under ROC curve & relatively higher metric values (classification report shown above)

For Decision Tree :

Maximum Depth (represented by K in the graph) = 8

Reason : Maximum area **(0.74)** under ROC curve & relatively higher metric values (classification report shown above)

4. If you have to choose only a subset of two features to predict earthquakes, which ones would it be? Give Reasoning. [Hint: You may use nodes of estimated Decision Tree or other techniques]

Answer :

Initially we expected a combination like latitude and longitude or depth along with latitude. But to our surprise after trying all possible combinations it turns out the best results are given by features **year and longitude (KNN)**. They even had higher correlation with the target variable and area under the curve was maximum for these 2 features .

5. Consider test results of the best model from above analysis. Report the input features that were used to achieve this. Try to improvise the test results by applying feature processing (You may come up with additional features by processing original ones). Report the new set of features that was used and also report the improvements in test results that were achieved. Please use appropriate metrics to report the results.

Answer :

KNN Model with $K = 37$ with features **{YEAR,LAT,LONG,MONTH,DEPTH}** gave the best results. We tried several modifications and preprocessing the features like :

1. Merging 2 features like :
 - multiplying Latitude(N) and Longitude(E) values
 - combining year month and date as total days
2. Simple scaling of values like adding latitude and year to create a new feature .
3. Random experiments like multiplying columns with different constants but there was no significant improvement . So in the end we chose to stick to our model without any special treatment to features.

VII. CONCLUSION

Finally, the input chosen for the model that performed best were the following :

Final Features included : YEAR, MONTH, DEPTH (km), LAT (N), LONG (E)
