Report.md

Classification

Aim

- The aim of the this assignment is not just for understanding and implementing classification models, but to analyse how do each of them work and make significant observations based on your analysis.

Libraries Used

1. Pandas
2. Scikit-learn
3. Matplotlib
4. Numpy

Meta-Data

1. `Sl. No.` : Serial Number
2. `Year, Month, Day` : Date of a particular earthquake as per UTC (Coordi-nated 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 earth-quake`
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

Explain the problem you are trying to solve

- Given a dataset about occurrences of earthquakes in a geographical region collected by the Earthquake Engineering Research Centre, IIIT Hyderabad. We have data in which we have input feature `Mw` which when compared with a threshold (set by us), gives:
    - For Mw < T, label becomes 0 (no earthquake) and for Mw ≥ T becomes 1 (earthquake).
- So by using other features except Mw we are trying to predict if it belongs to 0 class (no earthquake) or it belongs to 1 class (earthquake).
- We will be using Decision Trees classifier, KNN classifier and Ensemble Learning classifier to predict labels as Magnitude(Mw).
- The threshold that we are using is 4.5 which is the median of Mw. (Reason explained under data cleaning).

How did you perform cleaning on the data?

- The initial csv file consisted of a lot improper data. The csv was initially un-loadable, because of some data at top of csv file. Manually the title, etc was removed from the csv.
- After that the data was loaded (52989 rows × 20 columns).
- Many of the features consisting of NULL values.
- Inconsistent data type in some columns.
- Useless columns such Sl. No.,etc with no relation to our output were removed.
- The metadata such as year, month have unnecessary space before the comma , the space was removed from csv manually. Due to this `data['YEAR']` can be used instead of `data['YEAR ']` (Notice the space).

```
drop_features = [
    'Sl. No.', 'ORIGIN TIME (UTC)', '(IST)',
    'MAGNITUDE (Mw)', 'Mb', 'Mb.1', 'Ms', 'ML',
    'INTENSITY (MM)', 'MMI', 'MME', 'LOCATION', 'REFERENCE'
]
```

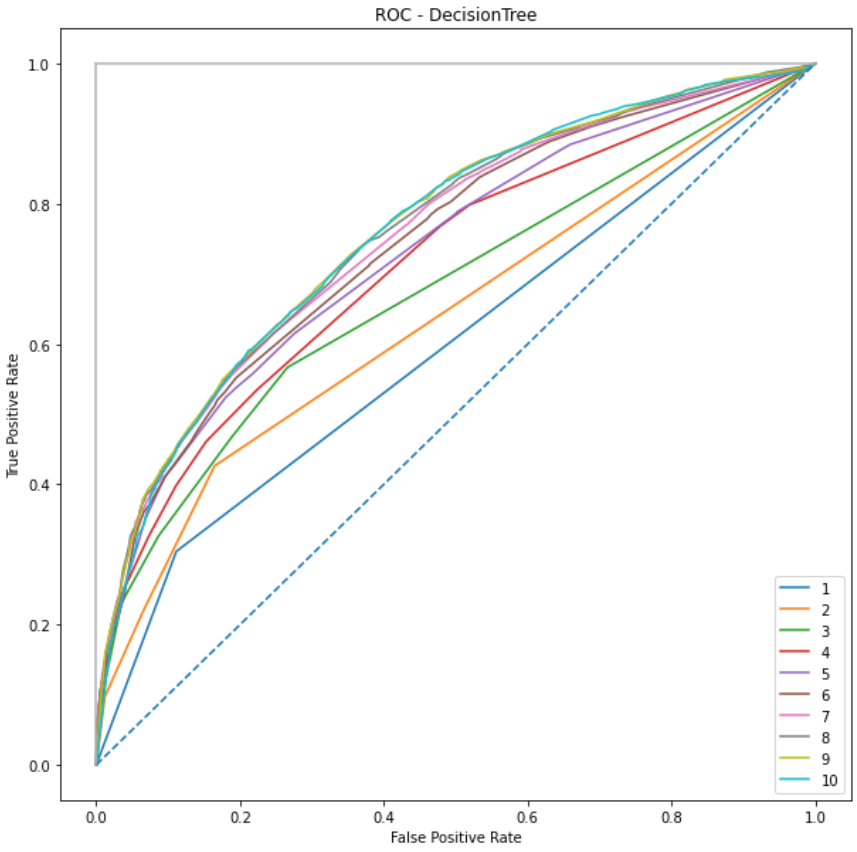
- Wrong values such as 0 value in date, month, etc. are handled. For month, cells containing 0 are converted into NaN.
- For each feature we performed separate cleaning.
  - YEAR: 1700 and more year are only considered.
  - MONTH: Convert to numeric & then 0 value cells converted to NaN
  - Mw: It's dtype was object. Converted that to float.
  - Latitude: Object to float & removing the chars, symbols in some cells.
  - Longitude: same as latitude
  - Depth:
    - The NaN values were not replaced by mean because when we calculated the variance of the data it turned out to be `56.108936` , which is a lot. Therefore we just removed the rows with depth=0.
    - ROC-AUC does not work well under severe imbalance in the dataset. Hence high variance of depth => NULL values removed instead of mean.
  - Finally we fill all the NaN data with mean of the feature in each cell.
  - The Reference feature was initially feature engineered with use of concept of label encoding (for categorical variables). This was also same as DATE feature and was removed.

Tasks

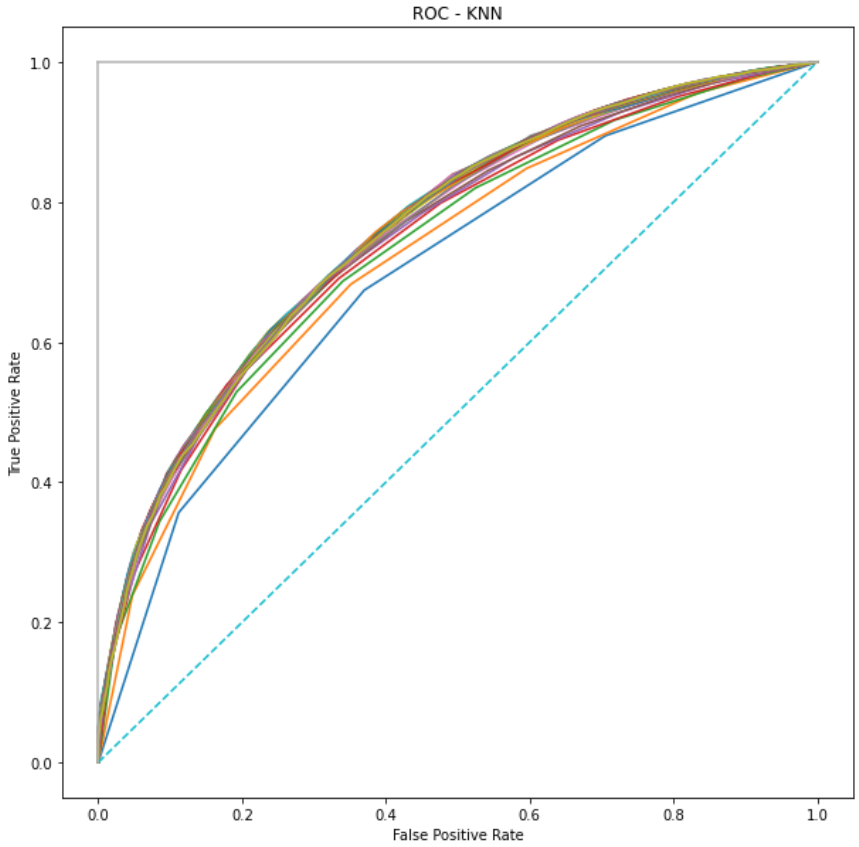
Task 1

1. Plot ROC for both these classifiers for K as parameter in KNN, pre-prune depth as a parameter in Decision Tree and number of estimators as parameter in ensemble learning.

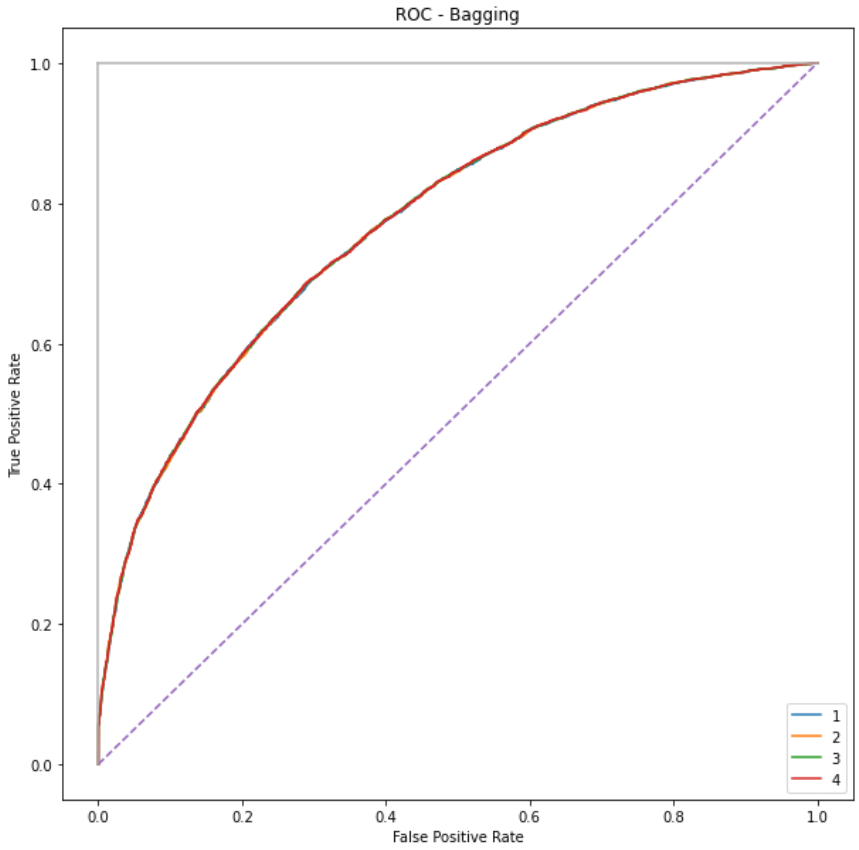
The labels denote the pre-prune depth:



The labels here are removed because they were a lot in number:



The labels denote the `no. of estimators/500` in ensemble learning:



Ensemble learning was taking a lot of time and also didn't produce that much different results, therefore only 4 iterations were done on it.

Task 2

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

Ensemble learning outperformed all of them as expected.

```
# Scores for DecisionTree

print('roc_auc_score: ', dec_max)
print('recall score: ', recall_score(y_test, dec_predict))
print('precision score: ', precision_score(y_test, dec_predict))
print('f1 score: ', f1_score(y_test, dec_predict))
```

```
roc_auc_score:  0.7622229084579758
recall score:  0.6741472172351886
precision score:  0.7130875138471278
f1 score:  0.6930708298085058
```

```
# Scores for KNN

print('roc_auc_score: ', knn_max)
print('recall score: ', recall_score(y_test, knn_predict))
print('precision score: ', precision_score(y_test, knn_predict))
print('f1 score: ', f1_score(y_test, knn_predict))
```

```
roc_auc_score:  0.7611081197206732
recall score:  0.6926989826451226
precision score:  0.7069781646052833
f1 score:  0.699765737172221
```

```
# Scores for Ensemble

print('roc_auc_score: ', bag_max)
print('recall score: ', recall_score(y_test, bag_predict))
print('precision score: ', precision_score(y_test, bag_predict))
print('f1 score: ', f1_score(y_test, bag_predict))
```

```
roc_auc_score:  0.7737081300764435
recall score:  0.6970377019748654
precision score:  0.718094944512947
f1 score:  0.707409656847859
```

In all these scores in almost all of them, bagging(ensemble) outperforms knn and decision tree.

In a scenario of a natural disaster like an earthquake, recall value is the preferred metric and recall value for bagging was better than others. `Recall value is what proportion of actual positives was identified correctly`

Task 3

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

The reasoning can be seen in the scores shown above in the image as well as in the code. For Decision Tree:

- Depth = 9 Reason: Max. area under ROC curve (0.762) & relatively higher metric values.
- k = 43 Reason: Max. area under ROC curve (0.761) & relatively higher metric values.
- No. of estimators = 3*500 = 1500 Reason: Max. area under ROC curve (0.773) & relatively higher metric values.

Task 4

4. If you have to choose only a subset of two features to predict earthquake, which ones would it be? Give Reasoning.

Both {YEAR,LATITUDE}, {YEAR,LONGITUDE} are good.

- {YEAR,LATITUDE} overall seems better in terms on all scores.
- {YEAR,LONGITUDE} has better recall score.

```
YEAR MONTH
{
    roc_auc_score for Logistic Regression:  0.7005825346852319
    f1 score for DecisionTree:  0.637319405819711
    precision score for DecisionTree:  0.691899852724595
    recall score for DecisionTree:  0.5907204828366655
    Accuracy: 0.6489857546116983
```

```
}
YEAR LAT (N)
{
    roc_auc_score for Logistic Regression:  0.7503815772747309
    f1 score for DecisionTree:  0.6977506143280195
    precision score for DecisionTree:  0.6992927506946198
    recall score for DecisionTree:  0.696215264679995
       Accuracy: 0.6850915774962253
}

YEAR LONG (E)
{
    roc_auc_score for Logistic Regression:  0.747714517501862
    f1 score for DecisionTree:  0.6977607324013361
    precision score for DecisionTree:  0.6867161816632168
    recall score for DecisionTree:  0.7091663523198793
       Accuracy: 0.6792489988840018
}

YEAR DEPTH (km)
{
    roc_auc_score for Logistic Regression:  0.7251339946360693
    f1 score for DecisionTree:  0.6738554672720393
    precision score for DecisionTree:  0.6759013282732448
    recall score for DecisionTree:  0.6718219539796303
       Accuracy: 0.6604739709840478
}

MONTH YEAR
{
    roc_auc_score for Logistic Regression:  0.7010073173714253
    f1 score for DecisionTree:  0.6350910573525415
    precision score for DecisionTree:  0.6909655478338016
    recall score for DecisionTree:  0.587577014962907
       Accuracy: 0.6474758747456181
}

MONTH LAT (N)
{
    roc_auc_score for Logistic Regression:  0.6505674650384193
    f1 score for DecisionTree:  0.6611999769013108
    precision score for DecisionTree:  0.6113840239214011
    recall score for DecisionTree:  0.7198541430906576
       Accuracy: 0.6148493402481455
}

MONTH LONG (E)
{
    roc_auc_score for Logistic Regression:  0.6519235346258168
    f1 score for DecisionTree:  0.6523109868538457
    precision score for DecisionTree:  0.618988484985324
    recall score for DecisionTree:  0.689425374072677
       Accuracy: 0.6162935731635265
}

MONTH DEPTH (km)
{
    roc_auc_score for Logistic Regression:  0.6136631435645958
    f1 score for DecisionTree:  0.6311494520867335
    precision score for DecisionTree:  0.5882864283385852
    recall score for DecisionTree:  0.680749402741104
       Accuracy: 0.5845860959758419
}

LAT (N) YEAR
{
    roc_auc_score for Logistic Regression:  0.7505847800194273
    f1 score for DecisionTree:  0.6962160800856261
    precision score for DecisionTree:  0.6972257250945776
    recall score for DecisionTree:  0.6952093549603923
       Accuracy: 0.6832534628766493
}
```

```
LAT (N) MONTH
{
    roc_auc_score for Logistic Regression:  0.6504194975149332
    f1 score for DecisionTree:  0.6616523952788707
    precision score for DecisionTree:  0.6127960561568964
    recall score for DecisionTree:  0.7189739720860053
       Accuracy: 0.6160966323114291
}

LAT (N) LONG (E)
{
    roc_auc_score for Logistic Regression:  0.6617608791623315
    f1 score for DecisionTree:  0.6487955080601341
    precision score for DecisionTree:  0.6240418118466899
    recall score for DecisionTree:  0.6755941154281403
       Accuracy: 0.6181316877831025
}

LAT (N) DEPTH (km)
{
    roc_auc_score for Logistic Regression:  0.671103308862645
    f1 score for DecisionTree:  0.6565007042684794
    precision score for DecisionTree:  0.6399235912129895
    recall score for DecisionTree:  0.673959512133786
       Accuracy: 0.6317862535285236
}

LONG (E) YEAR
{
    roc_auc_score for Logistic Regression:  0.747653591222056
    f1 score for DecisionTree:  0.6967654153008843
    precision score for DecisionTree:  0.6856134371957157
    recall score for DecisionTree:  0.7082861813152269
       Accuracy: 0.6781330007221165
}

LONG (E) MONTH
{
    roc_auc_score for Logistic Regression:  0.6514328600168849
    f1 score for DecisionTree:  0.6539765629647255
    precision score for DecisionTree:  0.6205689771957552
    recall score for DecisionTree:  0.6911857160819816
       Accuracy: 0.6181316877831025
}

LONG (E) LAT (N)
{
    roc_auc_score for Logistic Regression:  0.6612806367215082
    f1 score for DecisionTree:  0.649831243972999
    precision score for DecisionTree:  0.6240305590924875
    recall score for DecisionTree:  0.6778574122972463
       Accuracy: 0.6185912164379964
}

LONG (E) DEPTH (km)
{
    roc_auc_score for Logistic Regression:  0.6690404512499948
    f1 score for DecisionTree:  0.6553535841536324
    precision score for DecisionTree:  0.6304903555658843
    recall score for DecisionTree:  0.682258267320508
       Accuracy: 0.6253528523600079
}

DEPTH (km) YEAR
{
    roc_auc_score for Logistic Regression:  0.7249104284373995
    f1 score for DecisionTree:  0.6797944145148307
    precision score for DecisionTree:  0.669716935090288
    recall score for DecisionTree:  0.690179806362379
       Accuracy: 0.6605396179347469
}

DEPTH (km) MONTH
```

```
{
    roc_auc_score for Logistic Regression:  0.6142311613006635
    f1 score for DecisionTree:  0.6288460411416514
    precision score for DecisionTree:  0.5889132821075741
    recall score for DecisionTree:  0.6745882057085376
       Accuracy: 0.5842578612223462
}

DEPTH (km) LAT (N)
{
    roc_auc_score for Logistic Regression:  0.6708396289049816
    f1 score for DecisionTree:  0.6549563935634443
    precision score for DecisionTree:  0.6401728899027495
    recall score for DecisionTree:  0.6704388281151766
       Accuracy: 0.6311954309722313
}

DEPTH (km) LONG (E)
{
    roc_auc_score for Logistic Regression:  0.6686678466761454
    f1 score for DecisionTree:  0.6588305847076462
    precision score for DecisionTree:  0.6297867461591378
    recall score for DecisionTree:  0.6906827612221803
    Accuracy: 0.6265344974725924
}
```

Task 5

5. Consider test results of the best model from above analysis. Report the input features that was 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 was achieved. Please use appropriate metrics to report the results.

- Input features used are: `{YEAR,MONTH,LAT (N), LONG (E), DEPTH (km)}` .
- Bagging with 1500 number of estimators gave the best result.
- Initially the `NULL` value cells in the `DEPTH` feature was replaced by mean of the column. But after seeing the variance of the column, which came out too much, we removed the rows of cells consisting the `NULL` data. This did not make us lose a lot of data because not many rows were removed. This significantly helped in increasing the scores of the classifiers.
- We also did kind of feature engineering on YEAR, MONTH, i.e, the NULL values were replaced by mean values (float) instead of int. This basically means we have made new features from YEAR and MONTH. This also resulted in good scores.
- I also tried multiple feature pre-processing but none of them gave significant results so they were not added in implementation:
    - I tried multiplying constants with some columns such as MONTH, as I thought MONTHS is an important feature because in some months earthquakes are more frequent. Although this didn't result in better scores.
    - Adding together combinations of the YEAR,MONTH,DATE to form new feature such as total no of days, etc.
    - Weighted combination of latitudes and longitudes to form a new feature also didn't give results as expected.

Data Preprocessing

- Although a lot are mentioned in data cleaning itself.
- The training:test data = 75:25 ratio.
- Threshold was chosen as 4.5 which is median of Mw.
- I tried lower values of threshold such as 4. But I observed that though we achieved better accuracy it was only because of a huge difference (there were around 7-8 times more 1s than 0s) in the number of 1s and 0s .
- So, when I used 4.5 as the value of threshold, I 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 to avoid overfitting of the model. Other values of threshold showed a bias towards 0 or 1.

Conclusion

We performed different classifiers on same data and compared different scores. We got to know that some classifiers work best for some cases and most of the times ensemble is the superior although it takes more time.