MACHINE LEARNING CS60050
ASSIGNMENT 1 REPORT

- **GROUP 61:**
  G Sai Chaitanya, 18CS30018.
  G Maneesh Kumar, 18CS10020.

- **Dataset given:** 1A, AggregatedCountriesCOVIDStats.csv

- **Decision Tree construction:**
  - The dataset given to us has the following input attributes or features:
    - Date
    - Country
    - Confirmed
    - Recovered
  - The target attribute is 'Deaths'.
  - The target attribute has been changed to 'label' to make it compatible with other datasets.
  - The attribute 'Date' is discarded and is not considered for tree construction.
  - The given dataset has many duplicates rows. These duplicate rows are replaced by one instance of these duplicates to avoid unnecessary computation(as date is not being considered).
  - In **Part1**, the dataset is divided into 10 random [60:20:20] splits, 60 is used as train set(train_set), 20 as validation set(val_set), 20 for testing final accuracy(test_set).
  - The tree that is constructed is a regression tree, as the input attributes and the target attribute are continuous(except 'Country'). the attributes are classified as either 'categorical' or 'continuous'.
  - At each step, we try to find out best possible split. The condition with best_overall_metric is chosen as node and it has two branches, one which satisfies the condition(yes_answer) and the one which doesnot satisfies the condition(no_answer).The data at the node is the divided to its children and the process the continues recursively (rectangular splits).
  - Then, the average accuracy over 10 splits is calculated on the 'test_set', the best accurate tree and splits are considered. The split for which the best accuracy is obtained is used in the subsequent parts.
  - 'r2_score' is used as a measure of 'accuracy' to find the best accurate tree over 10 random splits in part1.
  - In **Part2**, we tried to plot accuracy vs depth using r2_score as accuracy measure and found the best depth limit for the given dataset. The split obtained in part 1 is used here (We found out the r2_score till depth 15 over val_set, as depth more than this cannot be possible for the given dataset

,considering the worst case where No.of leafnodes = 27000,i.e., $\log_2(27K) \approx 14.77$, where 27K is size of dataset after removing duplicate rows.)

- The tree obtained in part2 is used for post-pruning process.
- In **Part3**, the tree is pruned in order avoid overfitting of data. We used Reduced-Error Pruning (a post-pruning method) using 'mse'(mean squared error) as statistical measure. The algorithm traverses through the entire regression tree to find whether, the 'mse' of the subtree is greater when compared to the case when that particular node is a leaf node. If subtree has more error, then we replace the subtree rooted at this node with a leaf node, which is the mean over all the data reaching this node. The algorithm does this in a recursive manner.
- In **Part4**,the tree is visualized using pprint().

- **Results(Run on Google Colab):**

The dataset given to us is very large(43K size), it took more than 1hr for the entire process to complete for input depth = 3.

Part 1(Best Accurate Tree over 10 random splits):

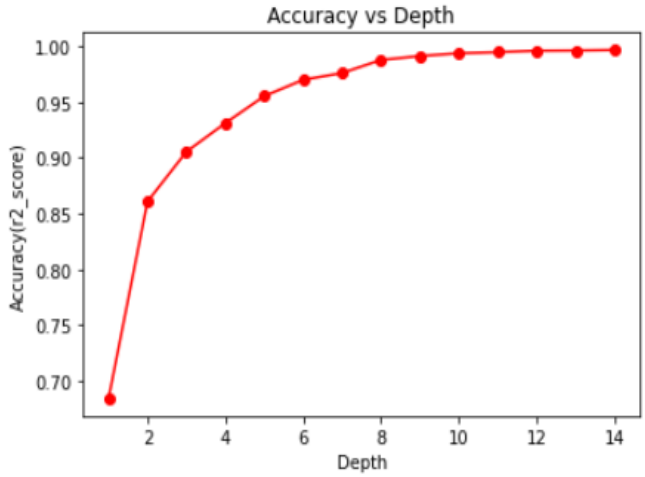
```
what is the depth of the tree to be constructed?(Choose something less than 5 for faster result)
3
*****************************PART 1****************************************
depth: 3 ; split_no: 1 ; r2_score: 0.9050381431777437
depth: 3 ; split_no: 2 ; r2_score: 0.902606603602282
depth: 3 ; split_no: 3 ; r2_score: 0.9055834007832997
depth: 3 ; split_no: 4 ; r2_score: 0.9016864616903346
depth: 3 ; split_no: 5 ; r2_score: 0.886376023488103
depth: 3 ; split_no: 6 ; r2_score: 0.877731391945939
depth: 3 ; split_no: 7 ; r2_score: 0.8884584143536164
depth: 3 ; split_no: 8 ; r2_score: 0.9049296320392489
depth: 3 ; split_no: 9 ; r2_score: 0.8522848265520039
depth: 3 ; split_no: 10 ; r2_score: 0.8634107257772397


Maximum r_sq score= 0.9055834007832997  for split number= 3


The average accuracy over 10 [60:20:20] splits is  0.888810562340981
```

Part 2(Best Depth limit and Accuracy vs Depth Plot):

```
*****************************PART 2****************************************
depth: 1 ;r2_score: 0.6840768416582284
depth: 2 ;r2_score: 0.8608640805012867
depth: 3 ;r2_score: 0.9055834007832997
depth: 4 ;r2_score: 0.9312701473684042
depth: 5 ;r2_score: 0.955493372372088
depth: 6 ;r2_score: 0.9701455647886598
depth: 7 ;r2_score: 0.9762247786526366
depth: 8 ;r2_score: 0.988011749524608
depth: 9 ;r2_score: 0.9914478770509064
depth: 10 ;r2_score: 0.9939675801465593
depth: 11 ;r2_score: 0.9949299425955409
depth: 12 ;r2_score: 0.9962341482428994
depth: 13 ;r2_score: 0.9964350801430228
depth: 14 ;r2_score: 0.9969107362021313
```




```
The best possible depth limit is: 14
```

- The tree obtained from part2 has depth ‘14’.

Part 3(Reduced-Error Pruning):

```
**********************************PART 3****************************************
MSE error on val_set before and after pruning:
MSE of Tree:        474,392
MSE of pruned Tree: 449,160

MSE error on test_set before and after pruning:
MSE of Tree:        489,628
MSE of pruned Tree: 547,914

Plot on test set
```