CS60050
MACHINE LEARNING - AUTUMN 2020
ASSIGNMENT 1

Submitted by :
Group No. 083
20CS60R60 - RAVI PRATAP SINGH
20CS60R64 - SAHIL JAIN

Part 1:

Build a decision tree by taking as input a maximum depth

Step 1: Imported the necessary libraries for the algorithm like:
Pandas(data analysis and manipulation tool)
Numpy(for manipulation of data using mathematical tools)
Matplotlib (for plotting)
Random (for random splits)
seaborn(for visualisation of data)
pprint(for printing tree)

Step 2: Read the data from CSV file (Converted the given excel data file into CSV because of facing some problem with excel file)

Step 3: Dropped the unnecessary attributes from the data like Sno, Time and labelled the target attribute(Confirmed) ad 'label'.

Step 4: Splitted the given data into 80-20 ratio , where 80% belongs totrain data set and 20% belongs to test data set.

Step 5: Created a bunch of utility functions required for building the decision tree .
-create_leaf for creating the leaf from the label attribute .
-split_data for splitting the node into two nodes .
-get_potential_split for making all potential splits ..
-calculate_mse :we calculated mean squared error because it tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the "errors") and squaring them. The squaring is necessary to remove any negative signs .
-calculate_overall_metric :for adjusting mse according to the above and below data points.
-determine_best_split : For evaluating all potential splits and choosing the best out of them.

Step 6: The problem resembles regression so implemented the decision tree algorithm according to the logistic regression methodology.

Step 7: To check with our regression model , we took a random element from the test data set and gave it to predict_example function to predict the label value corresponding to the test value.

Part 2:

(includes accuracy of part 1) Best possible depth limit to be used for the dataset and a Plot explaining the same .

We iterated over 10 random splits and chose the split which provides the best (lowest) root mean squared error(accuracy). We took random samples ranging in the test data set over a step of 5 .

After performing over 10 random we get accuracy as well as best possible depth for the corresponding data set.

The best possible depth limit to be used with our data set is 10 and the minimum number of samples required for it is 20.

Created a function for plotting the actual v/s predicted values for both training and test data set.

```
Progress: Iteration 1/10
Progress: Iteration 2/10
Progress: Iteration 3/10
Progress: Iteration 4/10
Progress: Iteration 5/10
Progress: Iteration 6/10
Progress: Iteration 7/10
Progress: Iteration 8/10
Progress: Iteration 9/10
Progress: Iteration 10/10
```

	max_depth	min_no_samples	r_squared_train	r_squared_test
93	10	20	0.881738	0.750472
92	10	15	0.884251	0.750259
83	9	20	0.878211	0.747491
82	9	15	0.880724	0.747278
94	10	25	0.876842	0.747143

Snapshot from program for 10 iterations over the 80-20 split

Plots of training and test data sets . ≫

Plot of actual and predicted value over the training set.

Plot of actual and predicted value over the test set .

Part 3:

We performed Post Pruning to avoid overfitting and to generalise our model so it will predict better with unseen data. First we let the tree to expand till max depth and then we will perform Pruning.
To perform post pruning we followed following steps:

Step 1 - Utility function required to perform post pruning -

- determine_leaf - Determine which node to be chosen as leaf node**.**
- Filter_df - to check nature of feature i.e discrete or continuous.
- Make_predictions - to make predictions after pruning nodes of the tree.
- Calculate_accuracy - to calculate accuracy after pruning.
- Pruning_result - to check whether to tree after pruning or before pruning.

Step 2 - Prune the node selected by determine_leaf function.

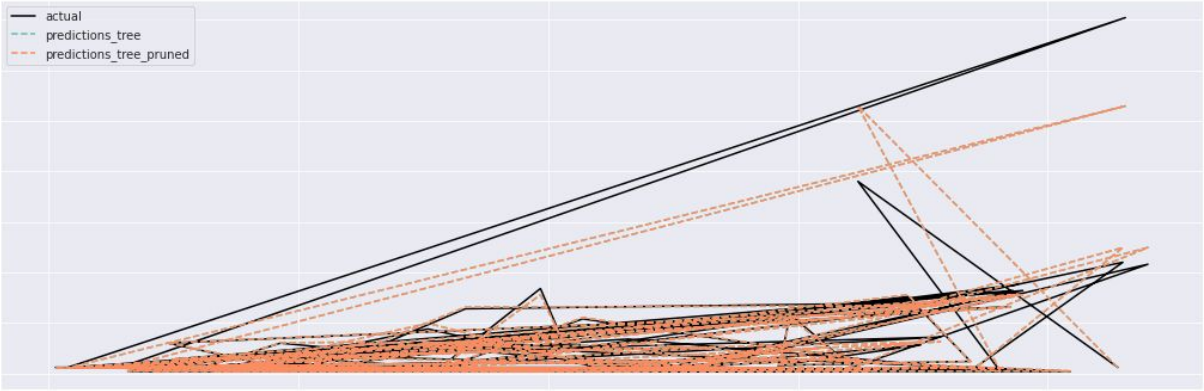
Step 3 - After pruning the node and making it a leaf node , we perform make_prediction on the training set.

Step 4 - Then we check the accuracy of the pruned tree using calculate_accuracy method and then return the accuracy.

Step 5 - Analysed the pruning result and decided whether to keep the tree before or after pruning.

Step 6 - If the accuracy of our tree increases we will keep the pruned tree and check for other nodes. Else we will discard this node for pruning and consider other nodes.

Step 7 - Plot the results and compare trees before and after pruning.



Part 4:

Printed the decision tree using the pprint library .Since the tree is stored as a dictionary pprint helps to print the tree in neat format .

Conclusion:

From the given data and our decision tree model we get the following statistics :
Best_Max_depth =10
Minimum No of Samples required for that depth = 20
Accuracy with Best_Max_depth chosen = 63.61%