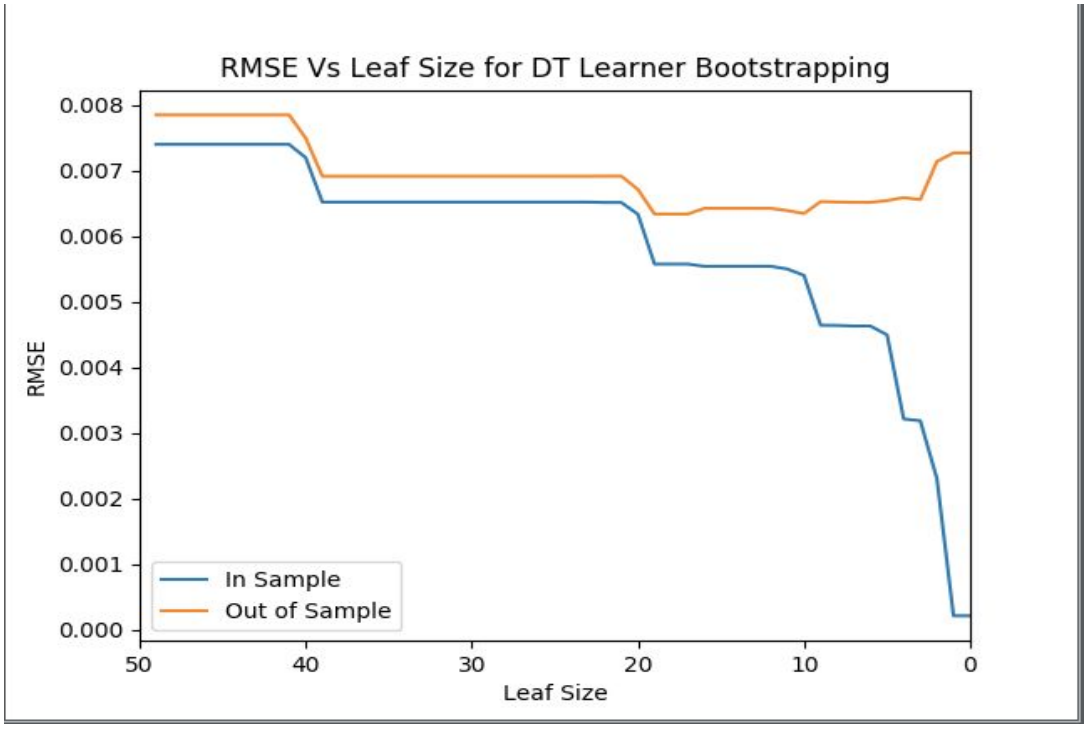
Instructions to run testlearner.py to generate the graphs :

Inside assess_learners directory execute the command :
PYTHONPATH=../:. python testlearner.py
No need to pass file name . It is already hardcoded for file name as "Istanbul.csv"

Question 1: Does overfitting occur with respect to leaf_size? Consider the dataset istanbul.csv with DTLearner. For which values of leaf_size does overfitting occur? Use RMSE as your metric for assessing overfitting. Support your assertion with graphs/charts. (Don't use bagging)

Answer: test learner generated : "Fig1"



Yes overfitting is occurring with respect to leaf size here .
In overfitting, model does really well on the training data but does poorly on the testing data.
According to the definition of overfitting explained in lecture - Region where in sample error is decreasing and out sample error is increasing is the region of overfitting.
In the graph generated , it is clearly visible that when leaf size is further decreased after 10 , In sample error is decreasing and Out of sample error is increasing ---this is the region of overfitting.So overfitting is occurring for leaf sizes **lower than 10.**

Experiment:
In testlearner.py ,after shuffling the data ,I ran DTlearner 50 times for varying leaf sizes from 1 to 51 .
For each value of leaf size -trained the tree model with training data. Collected the rmse values after querying the tree from training data(In sample) and testing data(Out of sample) separately in ndarrays.
Plotted the graph with RMSE Vs Leaf Size with two ndarrays of RMSE(training and testing).

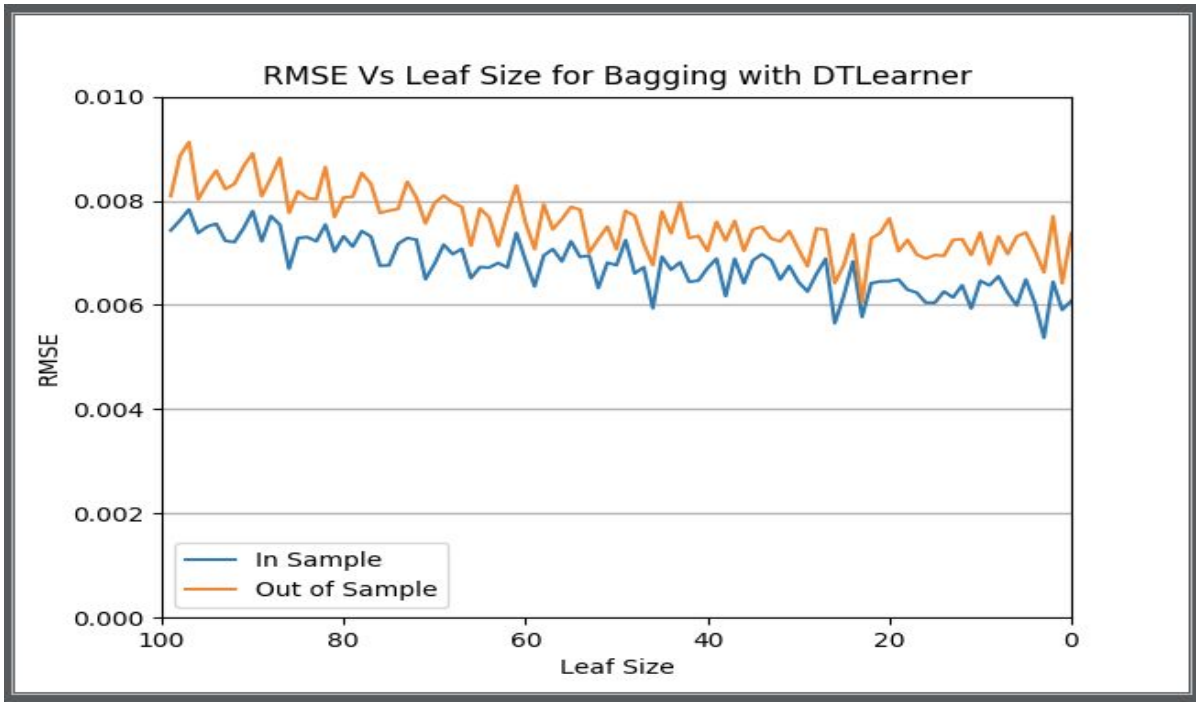
Reasoning:
Decision tree model is prone to overfitting since it makes the decision among a subset of all the features(columns), so when it reaches a final decision, it is a complicated and long decision chain. Only if a data point satisfies all the rules along this chain, the final decision can be made. This kind of specific rules on

training dataset make it very specific for the training set, on the other hand, cannot generalize well for new data points that it has never seen ,hence overfitting
If tree is too deep OR too many splits are required that signifies more overfitting.
If the leaf size is too small then it will take many splits to reach that leaf size.Too deep a tree means overfitting!

Question 2: Can bagging reduce or eliminate overfitting with respect to leaf_size? Again consider the dataset istanbul.csv with DTLearner. To investigate this choose a fixed number of bags to use and vary leaf_size to evaluate. Provide charts to validate your conclusions. Use RMSE as your metric.

Answer : Yes bagging can reduce the overfitting to the great extent with respect to leaf size.
As below graphs displays that overfitting is almost eliminated after bagging. There is no region of overfitting. Since test data is behaving decently well, we are not overfitted here.

testlearner generated : "Fig2"



Experiment : Executed BagLearner 100 times for varying leaf sizes from 1 to 101 with the number of bags as 20. For each leaf size, I have created 20 bags and trained the model.
Collected the rmse values after querying the bagging model from training data(In sample) and testing data(Out of sample) separately in ndarrays.
Plotted the graph with RMSE Vs Leaf Size with two ndarrays of RMSE(training and testing).

Reasoning:
Here we are aggregating the Independent decision trees since every time we are sampling the data with replacement.Replacing the data point every time we sample, allows us to maintain the distribution of our data, and reduces the variance due to the fact that each training set is statistically representative of the full dataset.Since high variance is the cause of overfitting .Due to the reduction in variance across different leaf size trees, overfitting is also reduced considerably. The final decision of ensemble is decided based on the average of outcome of all independent decision trees.

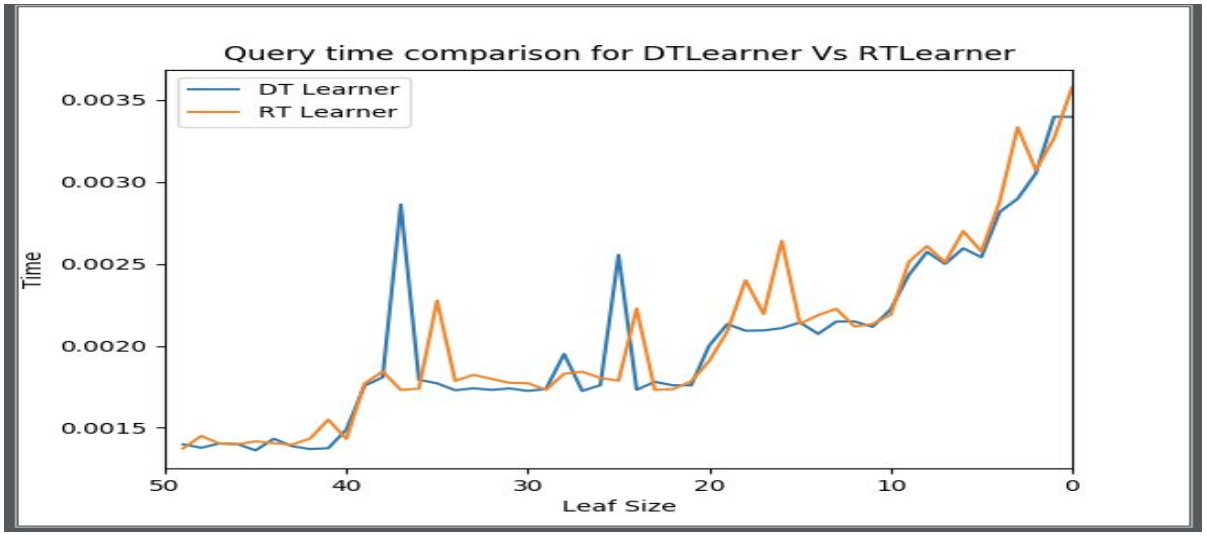
Question 3: Quantitatively compare "classic" decision trees (DTLearner) versus random trees (RTLearner). In which ways is one method better than the other? Provide at least two quantitative measures. Note that for this part of the report you must conduct new experiments, don't use the results of the experiments above for this.

Answer:

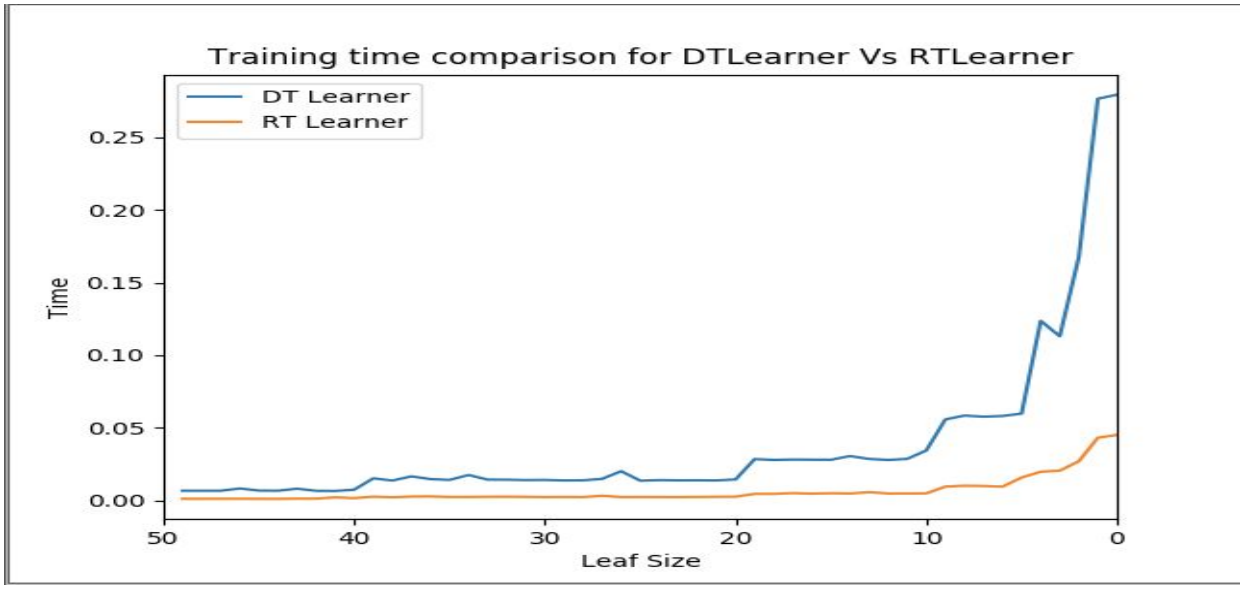
a). Time Comparison Experiment : I have taken below time measures for comparison between DTLearner and RTLearner -

1. Time taken to **query** the tree for particular leaf size
2. Time taken to **train** the model for each leaf size

testlearner generated : "Fig3"



test learner generated : "Fig4"



Experiment :

Executed DTLearner and RTLearner 50 times for leaf size varying from 1 to 51 and calculated time to train , time to query the model for each tree of different leaf size . Stored it into ndarrays of corresponding learners .

Query Time Calculation:

```
startTimeToQuery = time.time()
learner.query(testX)
endTimeToQuery = time.time()
Elapsed time = endTimeToQuery - startTimeToQuery
```

```
Training Time Calculation:
startTimeToTrain = time.time()
learner.addEvidence(trainX,trainY)
endTimeToTrain = time.time()
Elapsed time = endTimeToTrain - startTimeToTrain
```

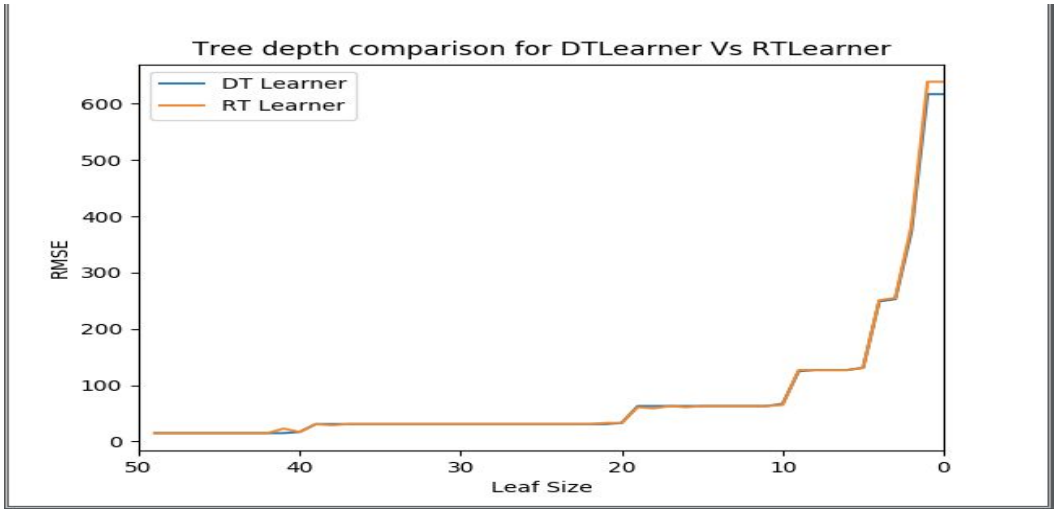
Reasoning:

Query time metrics: It is observed that Query time does not show any significant differences between the two learners and apart from some peaks , more or less both the graphs overlap. Hence it is not possible here to chose better performing model based on query time.

Training time metrics: However training time model clearly shows that DT Learner takes more time always than RT to train the model.Hence time performance of RT is better than DT. One more observation can be made by looking at first graph that after leaf size is further reduced to 10 , the time taken by DT learner is increasing at a very high rate. This can explained as- **If the leaf size is too small then it will take many splits to reach that leaf size.Many splits means more time taken .**

b).Tree Depth Comparison Experiment --Here I am comparing the no. of levels in DTLearner Tree and RT Learner Tree for a given leaf size in the below figure:

Testlearner.py generated: "Fig6"

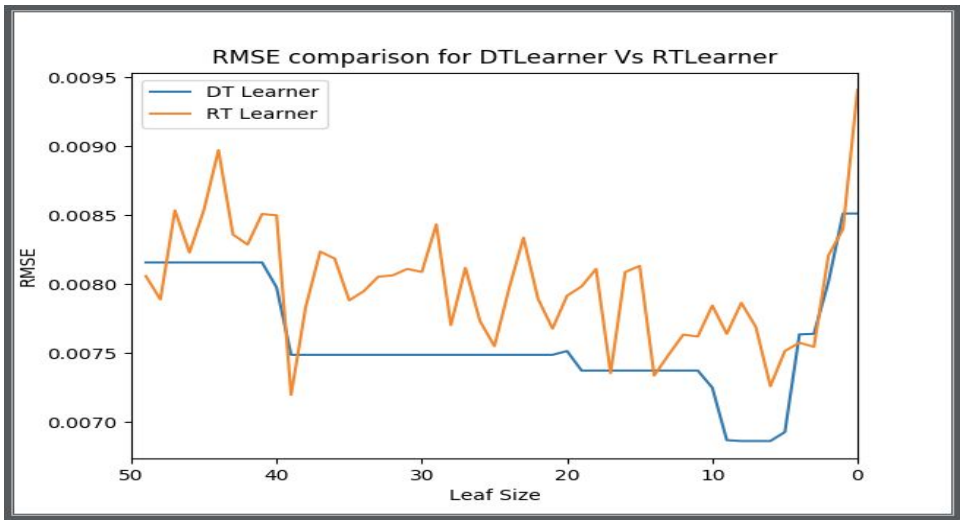


Both the RT learner and DT Learner graphs are overlapping. That means no.of levels in tree are varying with varying values of leaf size , but they are exactly same for DT Learner and RT Learner .

Hence that concludes - the depth of tree is independent of the way we select the best feature value .It only depends on leaf size. Also for leaf size smaller than 10 the tree depth is growing and very faster rate . that explains overfiiting in this region.

If we have look at RMSE comparison between DT and RT learner with respect to leaf size then the below graph shows that RT learner has always greater RMSE values as compared to DT learner.

Testlearner generated graph : "Fig5"



Following can be concluded from above experiments in question 3:
1). RT Learner is faster than DT learner to train the model because of randomized feature selection. And due to same random feature selection , it is more erroneous too(RMSE values are higher in RT learner than DT Learner)

2). However , RT learner does not show any **time** improvement over DT Learner while we query the data points.

3). The reasoning in point 2 is supported by tree depth experiment where the depth of tree is same for both the learners for a given leaf size.If tree level is same for the given leaf size in both the learners then it will take same amount of time to query the learner for test data.