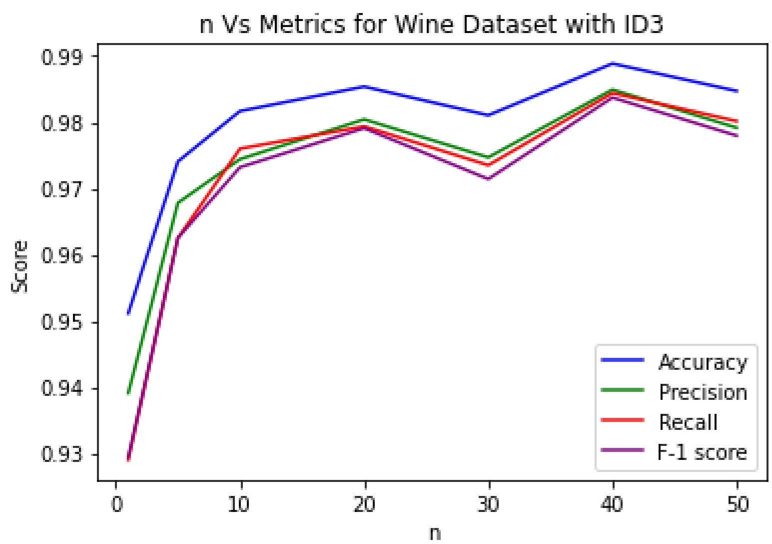
Implementation Details for Algorithms:

- I used the recommended fold value of k = 10
- I used the max depth of the algorithm to 10.
- For the min size before split ie. remaining n values after I don't split, is set to 5.
- Minimal Gain is set to 0.01 being close to 0 but not 0.
- The bootstrap ratio has been set to 0.1 so 10% of the data got resampled.
- Stopping criterion like minimal size for split criterion, maximal depth, minimal_gain

For the Wine Dataset using ID3:

1 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.951
Precision: 0.939
Recall: 0.929
F-Score (beta=1): 0.929

5 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.974
Precision: 0.968
Recall: 0.962
F-Score(beta=1): 0.962



10 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.982
Precision: 0.974
Recall: 0.976
F-Score(beta=1): 0.973

20 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.985
Precision: 0.98
Recall: 0.979
F-Score(beta=1): 0.979

30 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.981
Precision: 0.975
Recall: 0.973
F-Score(beta=1): 0.971

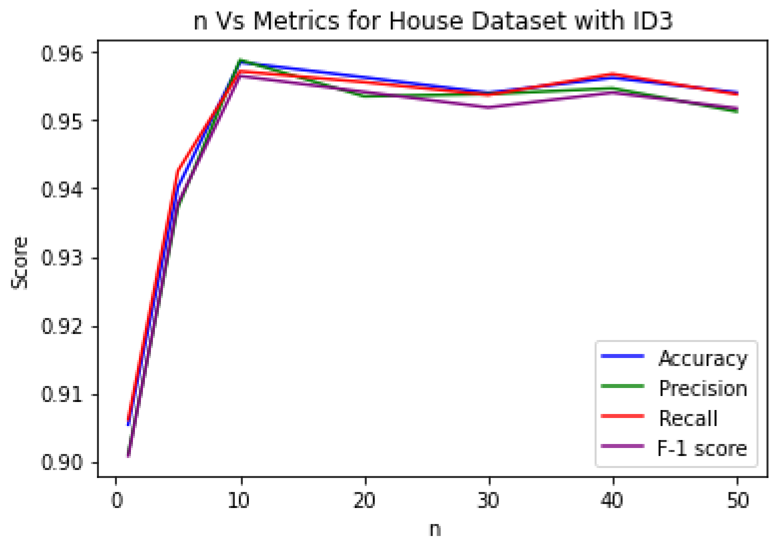
40 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.989
Precision: 0.985
Recall: 0.984
F-Score(beta=1): 0.984

50 Trees Random Forest of Wine Dataset with ID3:
Accuracy: 0.985
Precision: 0.979
Recall: 0.98
F-Score(beta=1): 0.978

For the House Dataset using ID3:

1 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.915
Precision: 0.914
Recall: 0.91
F-Score(beta=1): 0.909

5 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.94
Precision: 0.94
Recall: 0.937
F-Score(beta=1): 0.937



10 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.956
Precision: 0.957
Recall: 0.953
F-Score(beta=1): 0.954

20 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.954
Precision: 0.951
Recall: 0.954
F-Score(beta=1): 0.952

30 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.961
Precision: 0.958
Recall: 0.961
F-Score(beta=1): 0.959

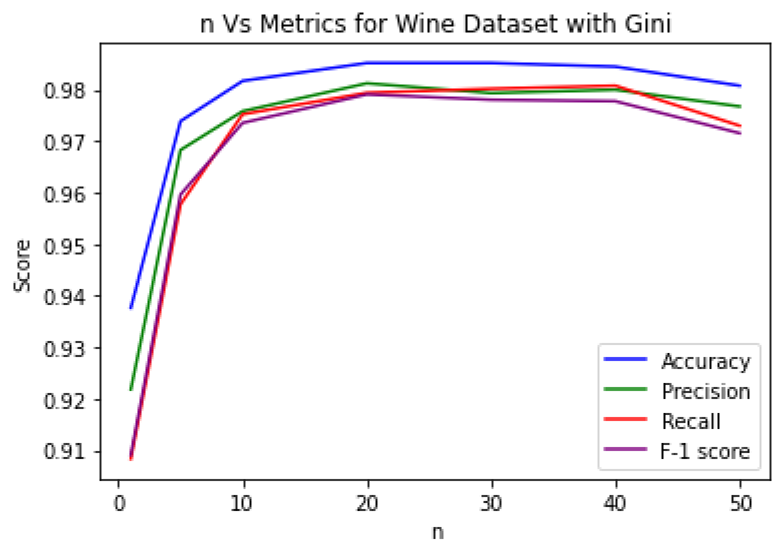
40 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.956
Precision: 0.953
Recall: 0.957
F-Score(beta=1): 0.954

50 Trees Random Forest of House Dataset with ID3:
Accuracy: 0.961
Precision: 0.959
Recall: 0.961
F-Score(beta=1): 0.959

For the Wine Dataset using Gini:

1 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.938
Precision: 0.922
Recall: 0.908
F-Score(beta=1): 0.909

5 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.974
Precision: 0.968
Recall: 0.958
F-Score(beta=1): 0.96



10 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.982
Precision: 0.976
Recall: 0.975
F-Score(beta=1): 0.974

20 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.985
Precision: 0.981
Recall: 0.979
F-Score(beta=1): 0.979

30 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.985
Precision: 0.979
Recall: 0.98
F-Score(beta=1): 0.978

40 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.984
Precision: 0.98
Recall: 0.981
F-Score(beta=1): 0.978

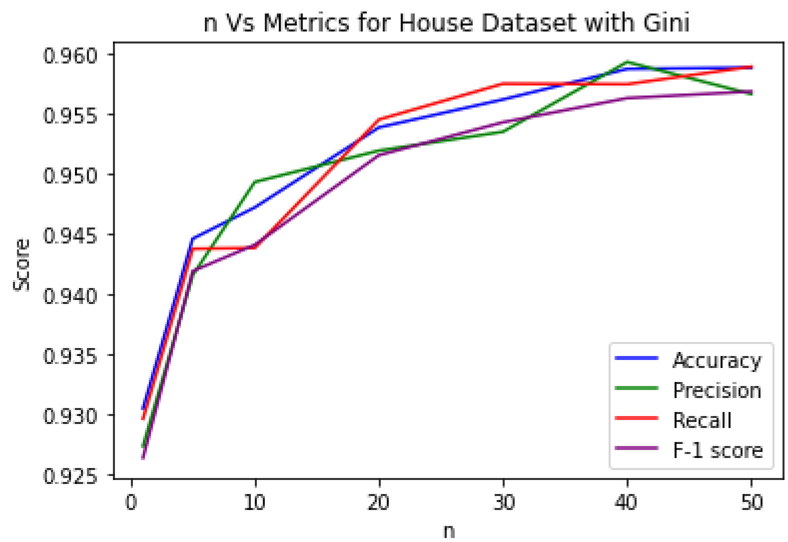
50 Trees Random Forest of Wine Dataset with Gini:
Accuracy: 0.981
Precision: 0.977
Recall: 0.973
F-Score(beta=1): 0.972

For the House Dataset using Gini:

1 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.931
Precision: 0.927
Recall: 0.93
F-Score(beta=1): 0.926

5 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.945
Precision: 0.942
Recall: 0.944
F-Score(beta=1): 0.942

10 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.947
Precision: 0.949
Recall: 0.944
F-Score(beta=1): 0.944



20 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.954
Precision: 0.952
Recall: 0.955
F-Score(beta=1): 0.952

30 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.956
Precision: 0.953
Recall: 0.957
F-Score(beta=1): 0.954

40 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.959
Precision: 0.959
Recall: 0.957
F-Score(beta=1): 0.956

50 Trees Random Forest of House Dataset with Gini:
Accuracy: 0.959
Precision: 0.957
Recall: 0.959
F-Score(beta=1): 0.957

4) *For each metric being evaluated (and for each dataset), discuss which value of ntree you would select if you were to deploy this classifier in real life. Explain your reasoning.*

a) For the Wine Dataset:
   i) Accuracy: We observe that while using Entropy, the value ntree = 20 returns the most optimal results. In the case of Cart, the value ntree = 40 returns the most optimal results. Hence, backed by our findings, I would choose the value of ntree to be 40 if I were to deploy this classifier in real life for the most optimal accuracy results.
   ii) Precision: We observe that while using Entropy, the value ntree = 40 returns the most optimal results. In case of Cart, the range of ntree value from 20 - 40, almost remains constant showing a next to none loss in precision value with a standard deviation of ±0.001. Hence, backed by our findings, I would choose the value of ntree to be 40 if I were to deploy this classifier in real life for the most optimal precision results.
   iii) Recall: We observe that while using Entropy, the value ntree = 40 returns the most optimal results. In the case of Cart, the range of ntree value from 20 - 40, almost remains constant showing a next to none loss in recall value with a standard deviation of ±0.001. Hence, backed by our findings, I would choose the value of ntree to be 40 if I were to deploy this classifier in real life for the most optimal recall results.
   iv) F-1: We observe that while using Entropy, the value ntree = 40 returns the most optimal results. In the case of Cart, the range of ntree value from 20 - 40, almost remains constant showing a next to none loss in F-1 value. Hence, backed by our findings, I would choose the value of ntree to be 40 if I were to deploy this classifier in real life for the most optimal F-1 results.

   Hence, overall I would pick ntree value to be 40 for the most optimal results across all metrics if I were to deploy the model in real life.

b) For the House Votes Dataset:
   i) Accuracy: We observe that while using Entropy, the value ntree = 30/50 returns the most optimal results. In the case of Cart, the value ntree = 50 returns the most optimal results.
   ii) Precision: We observe that while using Entropy, the value ntree = 30/50 returns the most optimal results. In the case of Cart, the value ntree = 40 or 50 showing a very little deviation and returns the most optimal results.
   iii) Recall: We observe that while using Entropy, the value ntree = 30/50 returns the most optimal results. In the case of Cart, the value ntree = 50 returns the most optimal results.
   iv) F-1: We observe that while using Entropy, the value ntree = 30/50 returns the most optimal results. In the case of Cart, the value ntree = 50 returns the most optimal results.

   Hence, overall I would pick ntree value to be either 30 or 50 in case of Entropy, and would pick ntree value as 50 in case of Cart for the most optimal results across all metrics. Hence, if I were to deploy the model in real life, I would choose ntree value as 50 to obtain the most optimal results.

5) *Discuss (on a high level) which metrics were more directly affected by changing the value of ntree and, more generally, how such changes affected the performance of your algorithm. For instance: was the accuracy of the random forest particularly sensitive to increasing ntree past a given value? Was the F1 score a "harder" metric to optimize, possibly requiring a significant number of trees in the ensemble? Is there a point beyond which adding more trees does not improve performance—or makes the performance worse?*

In case of Entropy(ID3), all the metrics were equally sensitive to the value of ntree. This makes sense as accuracy and f1-score are somewhat related to the values of precision and recall. In general, as the value of ntree increased, the value of all the metrics also increased showing an upward trend and slowly stabilizing, as should be the case. Hence the performance of the classifier increased as the value ntree increased.
In the case of the Wine dataset, we observe all the values increasing until ntree = 40 after which they start to dip little as ntree value reaches 50 which could be due to overfitting of the model as the number of trees increases.. However, in the case of House Votes dataset, we observe the values peaking early around ntree = 30 and stabilizing after all the way to ntree = 50 indicating that increasing the number of trees beyond that point may not provide significant improvements in performance.

In the case of Cart(Gini), all the metrics were equally sensitive to the value of ntree. This makes sense as accuracy and f1-score are somewhat related to the values of precision and recall. In general, as the value of ntree increased, the value of all the metrics also increased showing an upward trend and slowly stabilizing, as should be the case. Hence the performance of the classifier increased as the value ntree increased.
In the case of the Wine dataset, we observe all the values increasing until ntree = 40 after which they start to dip little as ntree value reaches 50. However, in the case of House Votes dataset, we observe the values constantly increasing until the value of ntree reaches 50 where we, in our test case, received the most optimal results. We know that As the number of trees increases, the variance of the model decreases, which leads to better generalization performance.
The difference in the behavior of the Wine and House Votes datasets could be explained due to the complexity of the datasets and the size of the dataset. The Wine dataset is less complex and smaller than the House Votes dataset, which is why the performance of the model starts to dip after ntree = 40.

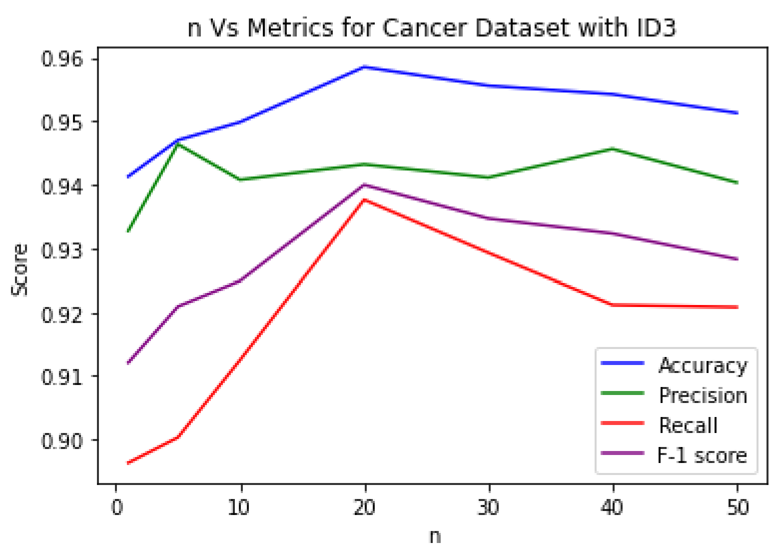
(Extra Points #2: 8 Points) Analyze a third dataset: the Breast Cancer Dataset. The goal, here, is to classify whether tissue removed via a biopsy indicates whether a person may or may not have breast cancer. There are 699 instances in this dataset. Each instance is described by 9 numerical 4 attributes, and there are 2 classes. You should present the same analyses and graphs as discussed above. This dataset can be found in the same zip file as the two main datasets.

For the Cancer Dataset using ID3:

1 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.941
Precision: 0.933
Recall: 0.896
F-Score(beta=1): 0.912

5 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.947
Precision: 0.946
Recall: 0.9
F-Score(beta=1): 0.921

10 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.95
Precision: 0.941
Recall: 0.912
F-Score(beta=1): 0.925



20 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.959
Precision: 0.943
Recall: 0.938
F-Score(beta=1): 0.94

30 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.956
Precision: 0.941
Recall: 0.929
F-Score(beta=1): 0.935

40 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.954
Precision: 0.946
Recall: 0.921
F-Score(beta=1): 0.932

50 Trees Random Forest of Cancer Dataset with ID3:
Accuracy: 0.951
Precision: 0.94
Recall: 0.921

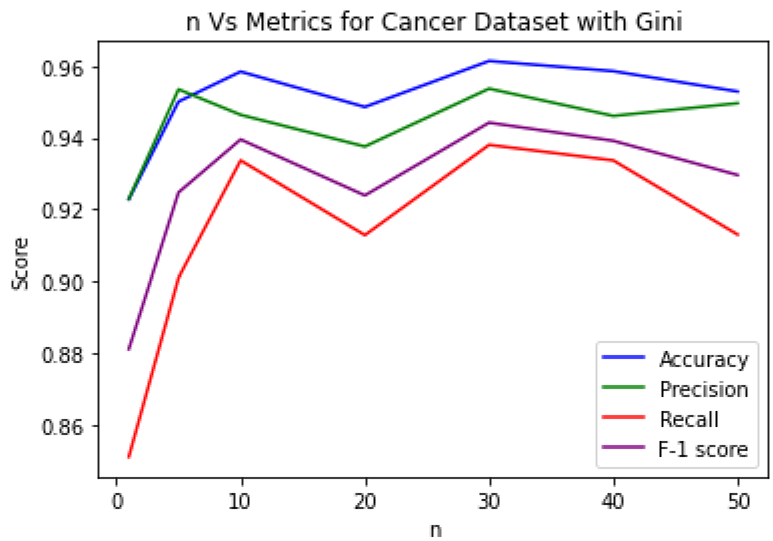
F-Score(beta=1): 0.928

For the Cancer Dataset using Gini:

1 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.923
Precision: 0.923
Recall: 0.851
F-Score(beta=1): 0.881

5 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.95
Precision: 0.953
Recall: 0.901
F-Score(beta=1): 0.925

10 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.958
Precision: 0.946
Recall: 0.934
F-Score(beta=1): 0.94



20 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.949
Precision: 0.938
Recall: 0.913
F-Score(beta=1): 0.924

30 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.961
Precision: 0.954
Recall: 0.938
F-Score(beta=1): 0.944

40 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.959
Precision: 0.946
Recall: 0.934
F-Score(beta=1): 0.939

50 Trees Random Forest of Cancer Dataset with Gini:
Accuracy: 0.953
Precision: 0.95
Recall: 0.913
F-Score(beta=1): 0.93

4)

For the Cancer Dataset:

Accuracy: We observe that while using Entropy, the value ntree = 20 returns the most optimal results. In the case of Cart, the value ntree = 30 returns the most optimal results.

Precision: We observe that while using Entropy, the value ntree = 5/40 (with ntree = 20 showing a very tiny variance of 0.003) returns the most optimal results. In the case of Cart, the value ntree = 5/30 showing a very little deviation and returns the most optimal results.

Recall: We observe that while using Entropy, the value ntree = 20 returns the most optimal results. In the case of Cart, the value ntree = 30 returns the most optimal results.

F-1: We observe that while using Entropy, the value ntree = 20 returns the most optimal results. In the case of Cart, the value ntree = 30 returns the most optimal results.

Hence, overall I would pick ntree value to be 20 in case of Entropy, and would pick ntree value as 30 in case of Cart for the most optimal results across all metrics. Hence, if I were to deploy the model in real life, I would choose ntree value between 20 to 30 to obtain the most optimal results.

5)

In both cases ie. Entropy and Cart, we observe that the clayes increase rapidly to ntree value of 20/30 and then start diverging both ways but in sort of a stabilizing manner until starting to decrease after reaching ntree = 50. I believe this trend is due to the complexity of the dataset and increasing the value of ntree leads to overfitting after a certain point and doesn't lead to much informational gain (around 20-30).

File Descriptions:

1) Hw3.py: Runs all the code
2) decisionTree.py: implementation of decision tree forest
3) randomForest.py. Implementation of random forest
4) evaluation.py: Metrics and results