Machine Learning Report:

Decision Trees

Overview

To put context to this assignment, a noisy and clean data set of how a person standing with a mobile phone in 4 different rooms perceived 7 different signals were given. Based on the WIFI signal strengths collected, this coursework required the implementation of a decision tree algorithm capable of determining which room the person was standing in. The tree had to be trained using both clean and noisy data sets. Furthermore, k-fold cross validation and nested k-fold cross-evaluation were used to evaluate the performance of the generated trees giving insight on performance metrics such as: accuracy, precision, recall and F-measure.

Following 4 main steps, this decision tree algorithm was implemented:

- Step 1: Loading data
- Step 2: Creating Decision Trees
- Step 3: Evaluation of Trees using a 10-fold cross validation
- Step 4: Prune the Trees and Evaluate them using nested 10-fold cross-validation

2. Creating Decision Trees:

The following figures are visualizations of the decision tree algorithm after being trained with the clean and noisy data set:

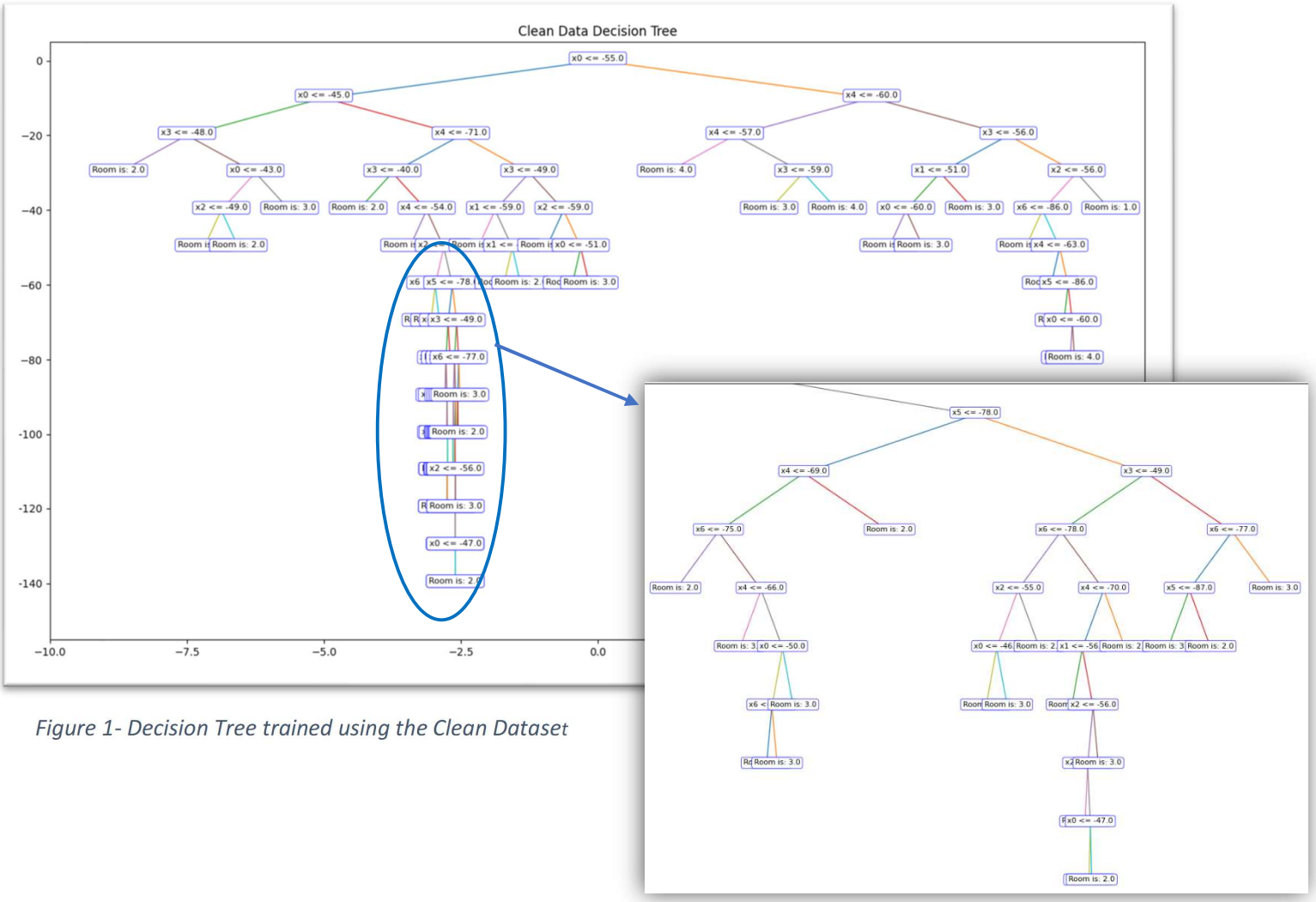


Figure 1- Decision Tree trained using the Clean Dataset

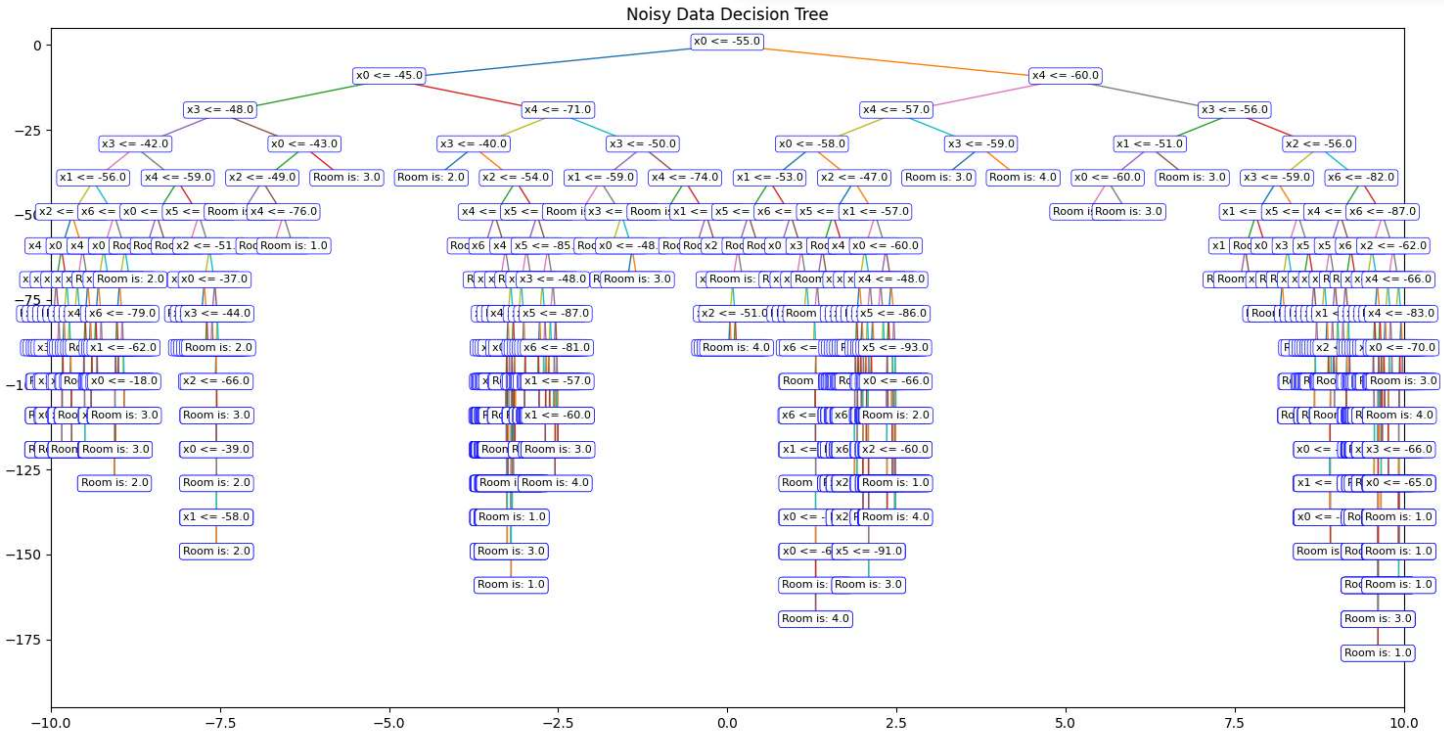


Figure 2- Decision Tree trained using Noisy Dataset

3. Evaluation:

To evaluate the performance of the trees on both datasets, a 10-fold cross validation is carried out. The confusion matrices of each test fold are then obtained, normalized, and averaged out to produce the average confusion matrix over all test folds. From that, the precision, recall and F1-measures are derived. We obtain the following performance metrics:

Clean Dataset Performance Metrics

Average confusion matrix:

```
Confusion matrix:
[[0.98593476 0.         0.00388386 0.0104772 ]
 [0.         0.96448466 0.03573596 0.        ]
 [0.00425725 0.03928933 0.95697826 0.00227273]
 [0.00375    0.         0.00647773 0.98909352]]
```

Average accuracy: 0.9734814578531178

	Room 1	Room 2	Room 3	Room 4
Precision	0.9919439	0.9608583	0.9540438	0.9872735
Recall	0.9856431	0.9642719	0.9543085	0.9897653
F1	0.9887835	0.9625621	0.9541761	0.9885178

Noisy Dataset Performance Metrics

Average confusion matrix:

```
Confusion matrix:
[[0.7881178  0.05538873 0.06363992 0.08837569]
 [0.06207434 0.80404581 0.07448704 0.05104951]
 [0.07897868 0.09779396 0.77596437 0.062747  ]
 [0.07573683 0.08351321 0.0751795  0.77590076]]
```

Average accuracy: 0.7834622742897301

	Room 1	Room 2	Room 3	Room 4
Precision	0.78426888	0.772569989	0.78438011	0.79329537
Recall	0.79166275	0.810810644	0.76413253	0.76796742
F1	0.78794847	0.791228537	0.77412395	0.78042595

Result analysis: Comment for both datasets which rooms are recognized with high/low accuracy, and which rooms are confused.

As a general observation, in the clean dataset all rooms are, on average, highly accurately recognized (95%+ accuracy); whereas in the noisy set all rooms have much lower accuracy. When observing the clean dataset's table, for a given room, it's precision and recall values are highly similar (having less than 1% difference between them). This means that all rooms are mostly identified with a high probability of being correctly predicted. For the noisy dataset, the rooms have a less stable difference between the precision and recall values which implies that some rooms may be correctly recognized but not correctly predicted and vice versa. More precisely, Room 1 and 2 have lower precision values meaning that they are the most likely to be confused.

Dataset differences: Is there any difference in the performance when using the clean and noisy datasets? If yes/no explain why.

Across all metrics, the clean dataset tree performance is far better (approx. 20% **higher**) than that of the noisy dataset. For the noisy dataset, the decision tree will learn the noise pattern alongside the actual data hence overfitting. Noise in the dataset increases the complexity (greater branching as seen in the visualization above) and variance of the generated model. This reduces its ability to perform well on the unseen test data and potentially misclassify a class label. This is supported by the evaluation data displayed above (drop in average accuracy from 0.97348 to 0.78346 between clean and noisy trees).

4. Pruning (and evaluation again):

Following the training and evaluation of a tree, to improve the performance, data can be compressed such as some branches are shortened and the tree is generalized. This process is called pruning. To evaluate the performance of a pruned tree, a method involving nested 10-fold cross validation (referred as ”option 2” in the lectures) was implemented. It performs an evaluation across 90 different trees, averaging confusion matrices, such as the following data presents an understanding to the effect of pruning on a decision tree trained with a clean data set or a noisy dataset.

Clean dataset pruned

Average Confusion matrix:

```
Confusion matrix:
[[0.98356804 0.         0.0088055  0.00798067]
 [0.         0.95984799 0.04203496 0.        ]
 [0.00599245 0.04826346 0.94136013 0.00584401]
 [0.00355123 0.         0.01072165 0.9856322 ]]
```

Accuracy: 0.9667314809086851

	Room 1	Room 2	Room 3	Room 4
Precision	0.9903901	0.9521249	0.9386173	0.98616781
Recall	0.9832198	0.9580440	0.9399877	0.98572577
F1	0.9867919	0.9550753	0.9393020	0.98594674

Noisy dataset pruned

Average Confusion matrix:

```
Confusion matrix:
[[0.83460665 0.05103346 0.05134014 0.06957016]
 [0.04528757 0.83944682 0.07588338 0.04107232]
 [0.05041381 0.07784494 0.83243233 0.05169486]
 [0.06220957 0.04204938 0.05311916 0.84743977]]
```

Accuracy: 0.8303850718491583

	Room 1	Room 2	Room 3	Room 4
Precision	0.8408986	0.83082732	0.8219321	0.83923448
Recall	0.8291752	0.83803048	0.8222480	0.84337648
F1	0.8349957	0.83441335	0.8220900	0.84130038

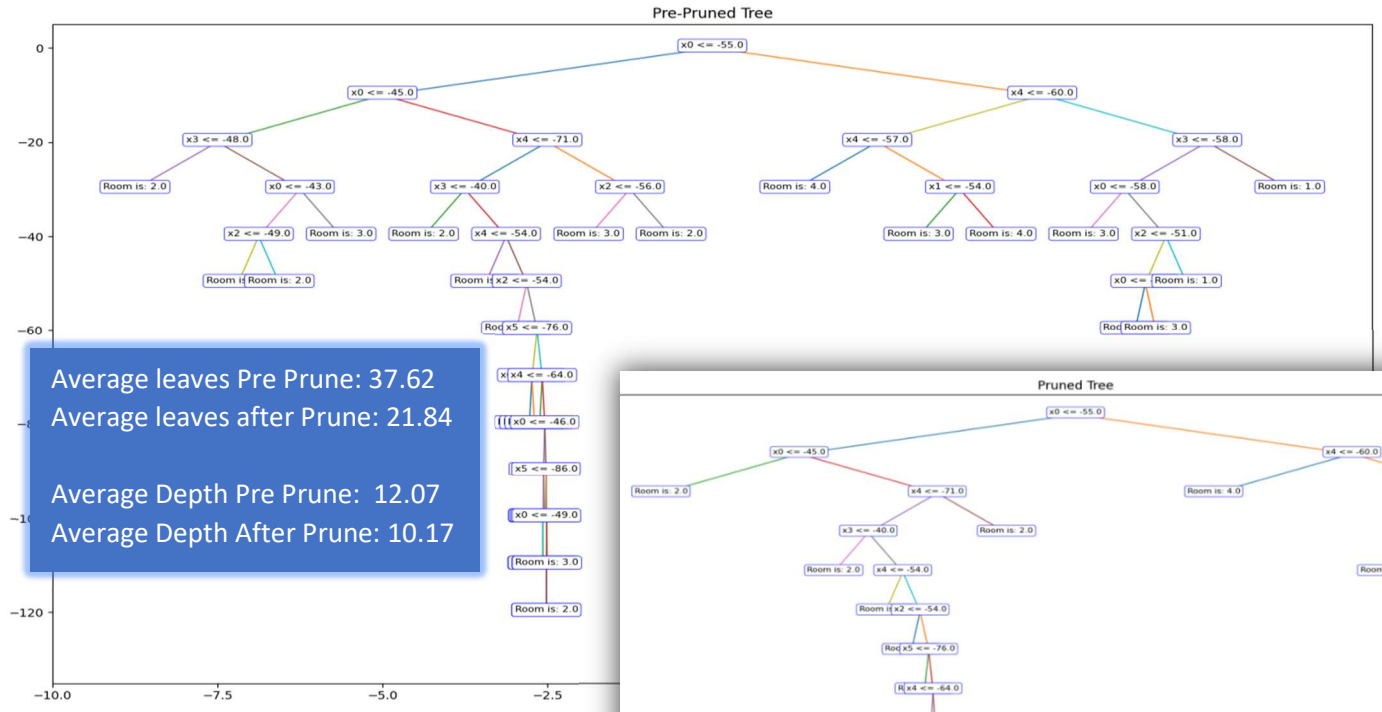
Result analysis after pruning**:** Comment the difference of performance before and after pruning for both datasets. Briefly explain these performance differences.

After multiple runs, a consistent increase in accuracy after pruning the noisy dataset tree was seen; whereas the clean dataset tree had varying changes with the performance usually being decreased. As the clean dataset has no noise, every leaf and node in the clean trained tree only increases performance in classifying the class labels correctly. Therefore, pruning the clean tree would cause for a loss in information hence a decrease in performance. The noisy tree however is overfitted to the data due to noise. Pruning the noisy trained tree leads to filtering the noise data from the training set hence the consistent increase in performance after pruning for the noisy dataset.

Depth analysis**:** Comment on the average depth of the trees that you generated for both datasets, before and after pruning. What can you tell about the relationship between maximal depth and prediction accuracy?

As seen above, the noisy dataset tree had a larger maximum depth but a lower overall performance in comparison to the clean tree. This means maximal depth is too high as it has led to the tree overfitting to the training data (high variance). Pruning this noisy tree led to a decrease in average depth but also an increase in performance. The converse also applies in which the depth must not be too low; otherwise, we would underfit the training data (assume too low bias) causing an increase in errors (e.g. pruning the clean dataset tree leading to a decrease in performance). These observations show that there’s an optimal point between high and low maximal depth that will lead to the best prediction accuracy for the tree.

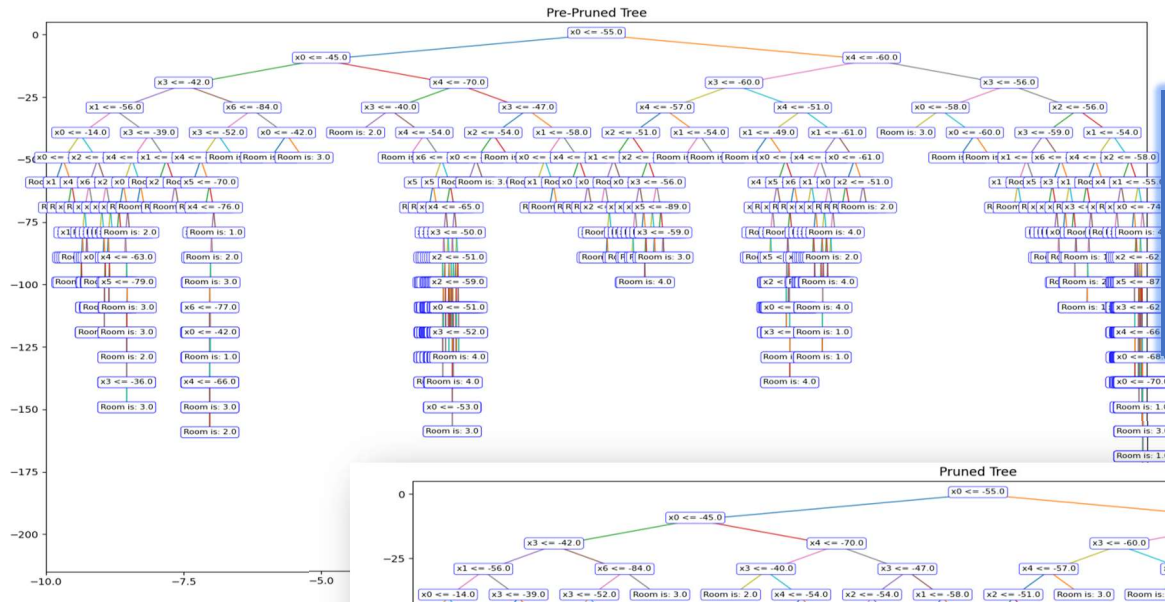
Example Clean Dataset Before and After Pruning



Average leaves Pre Prune: 37.62
Average leaves after Prune: 21.84

Average Depth Pre Prune: 12.07
Average Depth After Prune: 10.17

Example Noisy Dataset Before and After Pruning



Average leaves Pre Prune: 265.51
Average leaves after Prune: 172.59

Average Depth Pre Prune: 18.07
Average Depth After Prune: 17.38