Overview

This report implements and evaluates a decision tree classifier for indoor room localisation from WiFi signal strengths, follwing the structure of the problem sheet given in the 70050 class. The classifier supports continuous attributes and multiclass labels, and is evaluated with 10-fold cross-validation on both the clean and noisy datasets. A post-training pruning scheme based on validation error is also applied.

All code to reproduce the experiments is available at github.com/felixbaastadberg/introml.

Tree Visualization

For each feature i, we sort their their feature-value with the corresponding label $\{x_k[i], y_k\}$. We iterate through all the features i and all the possible splitpoints k, and choose the combination that maximizes information gain - defined as the following:

$$\text{IG} = H(S) - \left(\frac{|S_{\text{left}}|}{|S|} H(S_{\text{left}}) + \frac{|S_{\text{right}}|}{|S|} H(S_{\text{right}}) \right)$$

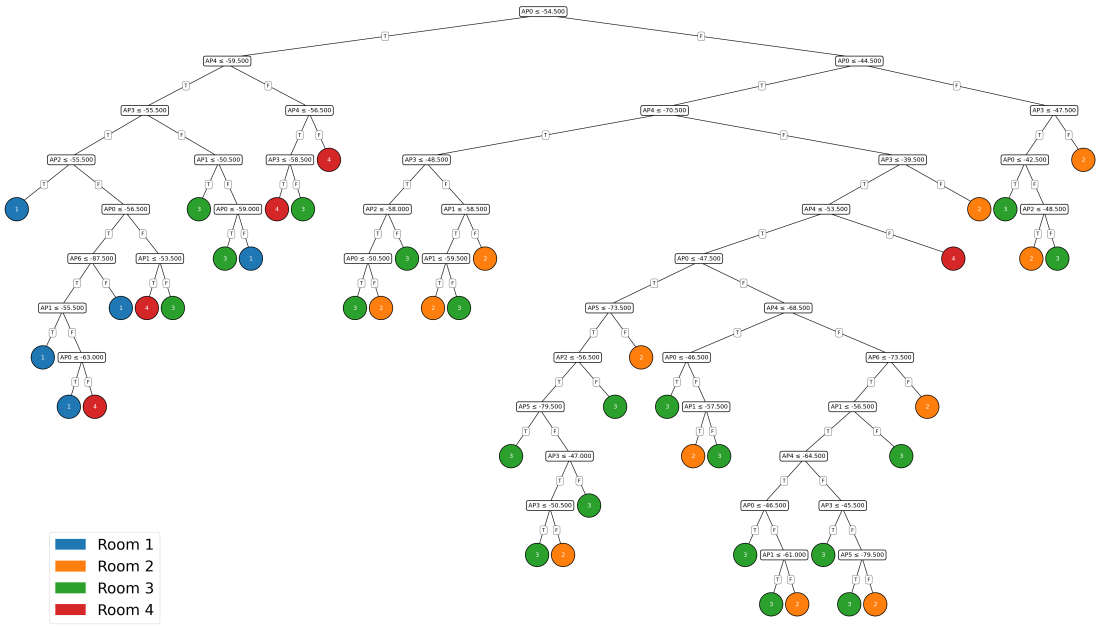

Figure 1: Decision tree visualisation trained on the full clean dataset.

As shown in Figure 1, the first split occurs when the WIFI strength of the first signal is less or equal to -54,50. This split adds a lot of information from a human standpoint as

well. We notice that only the left split contains signals from Room 1, while only the right split contains signals from Room 2. This seems like a very reasonable first split from an information gain standpoint.

Evaluation Before Pruning

All metrics are averaged (summed confusion matrices, accuracy, per-class precision/recall/F1) over 10 folds, as required.

Clean dataset (before pruning)

The following metrics in Table 1 and 2 are from testing the tree visualized in Figure 1.

	pred=1	pred=2	pred=3	pred=4
true=1	496	0	2	2
true=2	0	481	19	0
true=3	1	19	478	2
true=4	4	0	1	495

Table 1: Clean data: 10-fold summed confusion matrix before pruning. Accuracy = 0.9750.

Class	Precision	Recall	F1
1	0.9900	0.9920	0.9910
2	0.9620	0.9620	0.9620
3	0.9560	0.9560	0.9560
4	0.9920	0.9900	0.9910

Table 2: Clean data: per-class metrics before pruning. Average max depth = 12.80.

Noisy dataset (before pruning)

The following metrics in Table 3 and 4 are from testing the tree visualized in Figure 2.

	pred=1	pred=2	pred=3	pred=4
true=1	391	27	33	39
true=2	30	402	40	25
true=3	27	37	415	36
true=4	43	26	33	396

Table 3: Noisy data: 10-fold summed confusion matrix before pruning. Accuracy = 0.8020.

Result analysis

On the clean set, the tree almost perfectly recognises all rooms. However, the main confusion is between classes 2 and 3 (19 and 19 misclassifications), with tiny spillover between rooms 1 and 4. On the noisy set, errors are higher and more spread across all rooms, with confusion ranging between 25 and 43. Confusion slightly higher between room 2 and 3, and rooms 1 and 4. Overall, noise degrades class separability.

Class	Precision	Recall	F1
1	0.7963	0.7980	0.7971
2	0.8171	0.8089	0.8129
3	0.7966	0.8058	0.8012
4	0.7984	0.7952	0.7968

Table 4: Noisy data: per-class metrics before pruning. Average max depth = 18.50.

Dataset differences

Performance drops from 97.5% (clean) to 80.2% (noisy), and average depth grows (12.8 → 18.5), indicating the model fits more complex, noisy decision boundaries. Precision/recall remain reasonably balanced per class, but all are lower with noise. This behaviour is expected: noisy features blur thresholds so deeper trees chase spurious patterns and generalise worse.

Pruning (and evaluation again)

Pruning merges sibling leaves when it does not increase validation error (iterated until convergence).

Clean dataset (after pruning)

The following metrics in Table 5 and 6 are from testing the tree visualized in Figure 3.

	pred=1	pred=2	pred=3	pred=4
true=1	494	0	1	5
true=2	0	480	20	0
true=3	1	20	477	2
true=4	5	0	1	494

Table 5: Clean data: 10-fold summed confusion matrix after pruning. Accuracy = 0.9725.

Class	Precision	Recall	F1
1	0.9880	0.9880	0.9880
2	0.9600	0.9600	0.9600
3	0.9559	0.9540	0.9550
4	0.9860	0.9880	0.9870

Table 6: Clean data: per-class metrics after pruning. Average max depth = 11.50.

Noisy dataset (after pruning)

The following metrics in Table 7 and 8 the tree visualized in Figure 4.

	pred=1	pred=2	pred=3	pred=4
true=1	399	25	30	36
true=2	31	404	35	27
true=3	24	31	433	27
true=4	35	27	35	401

Table 7: Noisy data: 10-fold summed confusion matrix after pruning. Accuracy = 0.8185.

Class	Precision	Recall	F1
1	0.8160	0.8143	0.8151
2	0.8296	0.8129	0.8211
3	0.8124	0.8408	0.8263
4	0.8167	0.8052	0.8109

Table 8: Noisy data: per-class metrics after pruning. Average max depth = 17.7.

Result analysis after pruning

On the clean dataset, pruning slightly reduces accuracy (97.50% → 97.25%) while reducing depth (12.80 → 11.50). However, the drop in accuracy is likely not statistically significant, as changing the random seed results in accuracy variations of a similar magnitude. On the noisy dataset, pruning increases accuracy (80.20% → 81.85%) and reduces depth (18.50 → 17.70), showing that simplified subtrees generalise better with noise. The model overfit the noisy data, while there is no noise to overfit on the clean data.

Depth analysis

Trees on noisy data are deeper (18.50 levels) than on clean (12.80), reflecting added complexity from noise. Pruning shortens both decision trees a little bit. The noisy model benefits less in depth reduction but still gains accuracy, showing pruning's stabilising effect on noise. Shallower trees tend to capture more general trends, reducing variance and increasing bias.

Summary

This assignment implemented a decision tree classifier for indoor room localisation using WiFi signal strengths. The model achieved high performance on the clean dataset and moderate performance on the noisy dataset. Pruning did not change generalization performance much for the clean dataset, but it raised the noisy dataset, while reducing average tree depth. Deep decision trees indicates overfitting training data, introducing more variance to the test-set. Results confirm that noise increases model complexity and lowers separability, and that pruning mitigates overfitting by simplifying decision boundaries without sacrificing accuracy.

Appendix A - Commands Used

```
python train.py --data "wifi_db/clean_dataset.txt" --visualize
python train.py --data "wifi_db/clean_dataset.txt" --prune --visualize
python train.py --data "wifi_db/noisy_dataset.txt" --visualize
python train.py --data "wifi_db/noisy_dataset.txt" --prune --visualize
```

Appendix B - Visualization of All Trees

Figure 2: Noisy dataset without pruning

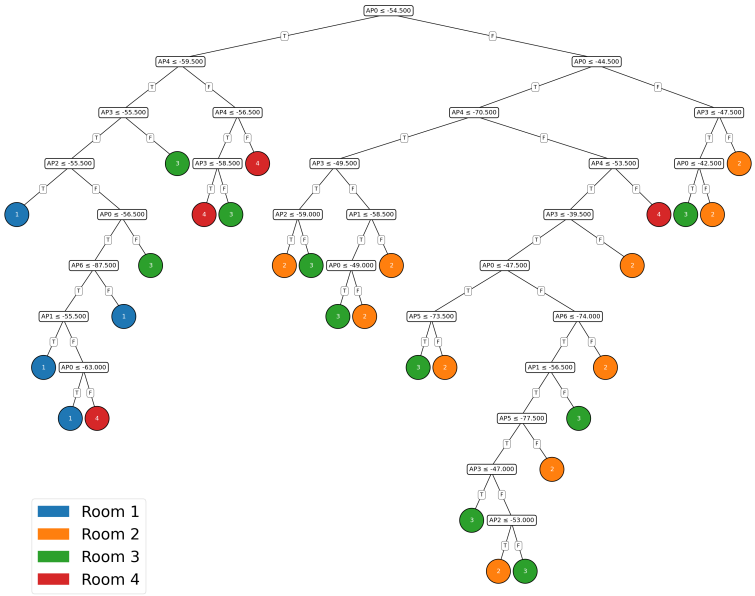

Figure 3: Clean dataset with pruning

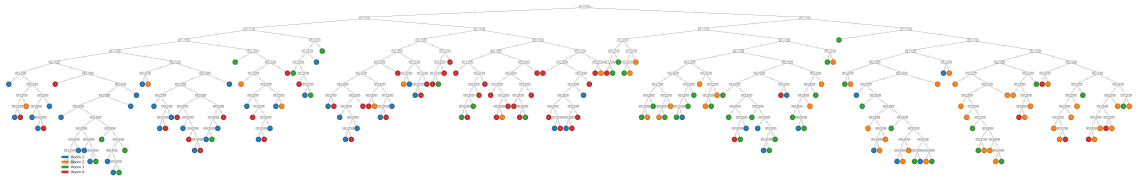

Figure 4: Noisy dataset with pruning