SCC.461 Final Assignment

36770071

2025-01-09

1. Abstract

This report compared a custom implementation of a Decision Tree Classifier against a SciKit-learn decision tree, and evaluated their performance against computational (training time, prediction time and memory usage) and machine learning aspects (accuracy, precision, recall and f1 score). The aim of this investigation was to determine which decision tree had the best performance metric results and thus was the most efficient. Both implementations were used using three well known datasets: “Iris”, “Wine” and “Breast Cancer Wisconsin (Diagnostic)”. Overall, the Sklearn decision tree performed better showing higher metric scores on computational and machine learning aspects compared to the custom decision tree. However, when comparing accuracy against the hyper-parameter of maximum depth of the tree, the custom decision tree showed greater accuracy scores than the Sklearn decision tree.

2. Introduction

2.1 Decision Trees

Decision trees are one of the most widely used and most implemented non-parametric supervised algorithms in machine learning and are used in many other aspects such as for decision analysis and for artificial intelligence. Decision trees, although can be used for both classification and regression tasks, this project will only focus on decision tree classifiers.

Decision trees aim to predict a target variable from input variables and attributes. To understand how this works, we first need to know how it is structured. A decision tree has a hierarchical structure with “nodes”. Nodes represent the decisions based on attributes which will split off into branches leading to “decision nodes” or “internal nodes”, this process can be repeated multiple times, where the “internal nodes” or “decision nodes” will become the new “root node” thus forming a sub-tree. This process will continue until either meeting stopping criteria or until the subset data is pure. The last nodes are “leaf nodes” representing the actual output.

Gini Impurity was used as a metric to determine the best splitting point at each node; it measures how pure a dataset is using this formula.

$$ = 1 - _{i=1}^{n} p_i^2

$$

Where p_i is the proportion of data points in class i and n is the total number of classes,

Although Entropy can be another metric to determine the best split that works quite similar to Gini Index, Gini index was more ideal as it is less intensive and is much faster to calculate. However, due to entropy having more complex calculation through logarithmic functions, it is more robust and the results obtained from it are better than Gini(GeeksforGeeks, 2024; Kaushik, 2023).

2.2 Scikit-learn

Scikit-learn is a widely used open-sourced machine learning library in Python, providing efficient tools for not only regression, forests but classification, which helps to provide machine learning implementations such as Decision Tree Classifier used in this project to compare against a custom implementation. One of the features of the Scikit-learn Decision Tree Classifier is that it defaults to using Gini Index to determine the best split, however there is the option to change it do Entropy (1.10. Decision Trees, n.d.; What Is Sklearn? | Domino Data Lab, n.d.).

2.3 Datasets

The data sets chosen were Iris, Wine and Breast Cancer Wisconsin (Diagnostic) picked out from the IC Irvin Machine Learning Repository (UCI Machine Learning Repository, n.d.). These datasets were mainly picked out due to the size of the database, with varying number of features.

The Iris dataset (Fisher, 1936) is much smaller than the others with four attributes depicting the features of the plant (sepal length, sepal width, petal width and petal length). From those four attributes it will determine whether it would be classified into one of three classes of Iris plant: Iris Setosa, Iris Versicolour and Iris Virginica.

The Wine dataset has 13 attributes and 3 classes of different wine cultivars, a bigger data set than Iris but not having the problem of having too many attributes thus is a good data set when first testing new decision tree classifiers due to not having challenging class structure. Finally, to challenge the classifier with a bigger dataset, Breast Cancer Wisconsin (Diagnostic) was used, with 30 attributes and 2 classes of diagnosis (malignant or benign).

2.4 Decision Tree Performance Measures

The aim of this project was to evaluate the machine learning and computational aspects of a custom and sklearn decision tree implementation.

Machine learning metrics such as accuracy, precision, recall and f1-scores were measured, where f1 score is the balance of recall and precision scores. Higher scores indicate better decision tree performance.

Computational aspects such as memory usage, training and prediction time were also measured to explore the efficiency of the models and compare them against each other. Lower scores on computational measures indicate better decision tree performance in these aspects.

3. Methods

This section provides an overview of the methods that went into the implementation process for both the custom decision tree and sklearn DecisionTreeClassifier. Focusing on how the tree was configurated, the training and prediction of the tree.

3.1 Tree Configurations

When configurating the decision tree, hyper-parameters were set to control for over-fitting and under-fitting the data and making it as accurate as possible.

The hyper-parameter “max_depth” refers to the maximum depth of the tree, it was used to control for the overall complexity of the tree. However this would allow for over-fitting if not defined as the nodes will keep splitting until all leaf nodes are pure (Importance of Decision Tree Hyperparameters on Generalization — Scikit-learn Course, n.d.). This project varied the tree depth from shallow (1) to very deep (10) to explore explore how this will affect the performance metrics (specifically accuracy) as the deeper the tree grows the more complex it becomes (Decision Tree Sklearn -Depth of Tree and Accuracy, n.d.).

Another hyper-parameter, “sample_min_split”, which refers to the minimum samples required to split decision nodes, was used. This means that if the data in the decision node is less than the minimum samples required, the node will become a leaf node. This hyper-parameter ensures the samples decide which split is considered, where large numbers prevent the tree from learning the data and small numbers mean the tree is an risk for overfitting (Importance of Decision Tree Hyperparameters on Generalization — Scikit-learn Course, n.d.). Thus, this project also varied the minimum sample size from a range of 2 to 5 to evaluate its effect on the accuracy of classifying.

These hyper-parameters are in place to ensure the tree performs well, as a shallow tree depth and fewer leaf nodes have been shown to be indicators of an efficient decision tree (Aaboub et al., 2023).

Test size was also varied when evaluating the decision trees. Test size refers to the portion of the dataset that is being tested, this ranged from 10%, 30%, 50% and 70%, to explore if the proportion of the dataset being tested affects the machine learning metrics, specifically accuracy results, of both decision tree implementations. For loops were used to test the combination of each of the hyper-parameters.

3.2 Fitting and Testing the Tree

Custom Decision Tree

Initially, when training the custom decision tree, the best split had to be determined. This was done through recursively splitting the data at each node by finding the best attribute and threshold to split at. As mentioned previously, Gini Index was used to determine how pure the dataset was, finding the best attribute and threshold that minimised the Gini value would indicate higher purity of the sample and thus would be the best split.

Based on the split, the data was split recursively into subsets. This process would continue until meeting the stopping criteria of the “max_depth” and “sample_min_split” or until the data in the node is completely pure. Once, reaching this criteria, the last data samples will become leaf nodes. The predicted class at these leaf node, if the decision tree met the stopping criteria, is the majority class in the leaf node samples.

Sklearn DecisionTreeClassifier

A library Decision Tree implementation was used for the comparison against the custom implementation. The “DecisionTreeClassifier” was taken from the Sci-kit learn website and was imported into the Pycharm to be trained and tested. Similar to the custom decision tree, it was initialised with “max_depth”, “min_sample_split” (“sample_min_split”- in the custom decision tree) hyper-parameters.

In order to train the tree, the .fit() function was used.

Once, importing the datasets, the same testing sizes were used to maintain consistent testing of performance measurements across both decision tree implementations. Sklearn.metric imported the performance metrics used for measuring accuracy, precision, f1 score and recall.

Computational Metrics

As mentioned previously, prediction time, training time and memory usage were measured and compared across both decision trees.

Training time was measured using the time.time() function in python which allowed to measure the prediction time through subtracting the start time from the final time, which was the time taken to finish the operation recorded in seconds.

To determine memory usage for both decision trees, memory_usage from memory_profiler was imported. However, once trialed, memory_usage was not recorded in the results. Thus, an alternative approach was used, as a fail-safe for when memory_profiler does not work. Python’s tracemalloc() would take snapshots of before and after performing an operation, to compare how much memory it used.

4. Results

4.1 Average Results

This section shows the average machine learning (accuracy, precision, recall and f1 score) and computational (training time, prediction time and memory usage) performance of the custom and sklearn decision trees for the three datasets that they were tested on.

Machine Learning

Small Dataset- Iris

Table 1 shows average scores (mean and SD) showing high machine learning metrics scores when tested in the Iris dataset. However, slightly higher scores from the sklearn decision tree showed better performance in machine learning aspects compared to the custom implementation.

Table 1. Iris Data :Mean and SD for Machine Learning Metrics

metric	mean_custom	sd_custom	mean_sklearn	sd_sklearn

metric	mean_custom	sd_custom	mean_sklearn	sd_sklearn
acc	0.8542698	0.1045152	0.8640476	0.1028300
recall	0.8542698	0.1045152	0.8640476	0.1028300
precision	0.8322598	0.1699279	0.8357903	0.1711329
f1	0.8330596	0.1468174	0.8413891	0.1462875

Medium Dataset- Wine

The results for the medium data (Table 2) set showed similar results to the small Iris dataset. Although both decision tree implementations showed high results again with the medium-size Wine dataset, the average machine learning metrics from the custom implementation was found to be smaller than those of the sklearn decision tree. It was also noted the scores were lower than they were when tested against the small dataset.

Table 2. Wine Data: Mean and SD for Machine Learning Metrics

metric	mean_custom	sd_custom	mean_sklearn	sd_sklearn
acc	0.8392340	0.1164320	0.8532617	0.1234958
recall	0.8337468	0.1368146	0.8438540	0.1418007
precision	0.8158915	0.1848358	0.8305449	0.1926624
f1	0.8176285	0.1694169	0.8291875	0.1754445

Large Dataset- Breast Cancer

The results have differed slightly compared to the two datasets, as seen in Table 3. Contrasting to the previous datasets, the machine learning performance metrics for the custom decision tree are shown to be greater than those of the sklearn decision tree. Although there was not a big difference, there is a slight difference in results, showing the custom decision tree performed better in terms of accuracy, precision, recall and f1 score.

Table 3. Breast Cancer Data: Mean and SD for Machine Learning Metrics

metric	mean_custom	sd_custom	mean_sklearn	sd_sklearn
acc	0.9186257	0.0143876	0.9133772	0.0139483
recall	0.9024108	0.0209657	0.8942568	0.0179399
precision	0.9256656	0.0154958	0.9227324	0.0174806
f1	0.9105978	0.0167902	0.9043992	0.0157584

Computational Aspects

Train time

Small dataset (Iris)

When determining the mean train time (in seconds) for both the custom and sklearn decision trees tested in the breast cancer dataset. Overall the findings showed a faster training time (in seconds) for the sklearn decision tree (M = 0.0014, SD = 0.00047) than the custom decision tree (M = 0.025, SD = 0.0095).

Medium dataset (Wine)

After determining the mean train time for the custom and sklearn implementations after being tested on the wine dataset, the results showed similar results to those in the other datasets, sklearn had a faster training time (M = 0.014, SD = 0.0005) compared to the custom decision tree (M = 0.215, SD = 0.114).

Large dataset (Breast Cancer)

The findings showed that train time was higher for the custom decision tree (M = 4.19, SD = 2.96) than the sklearn decision tree (M = 0.004, SD = 0.008). Therefore, showing that the sklearn decision tree was much faster to train.

Prediction Time

Small dataset (Iris)

As a small dataset was used to test for prediction time (in seconds) to compare both implementations of decision trees, the average prediction times for both are small. However, the results still showed a slightly faster prediction time for the sklearn implementation (M = 0.0000623, SD = 0.000244) compared to the custom implementation (M = 0.0000653, SD = 0.000243).

Medium dataset (Wine)

After determining the mean preduction time for the custom and sklearn implementations after being tested on the wine dataset, the results showed similar results to those in the other datasets, sklearn had a faster training time (M = 0.000025, SD = 0.00016) compared to the custom decision tree (M = 0.000087, SD = 0.00028).

Large dataset (Breast Cancer)

When measuring the average prediction time for both decision tree implementations, even though the prediction time for both were low, the sklearn decision tree had a lower prediction time (M = 0.0001, SD = 0.003) and was therefore faster at predicting class labels than the custom decision tree (M = 0.0003, SD = 0.0005).

Memory Usage

Small dataset (Iris)

To determine which decision tree was more efficient, the average memory usage was measured to compare how much memory was used up (in MegaBytes) between each of the decision trees when testing the smallest data (Iris). Overall the findings showed a lower memory usage for the sklearn decision tree (M = 0.00116, SD = 0.00123) than the custom decision tree (M = 0.00311, SD = 0.014).

Medium dataset (Wine)

Similar to the other datasets tested, when tested on the medium dataset (Wine), the sklearn decision tree used up less memory (M = 0.0011, SD = 0.00063) than the custom decision tree (M = 0.00214, SD = 0.00212).

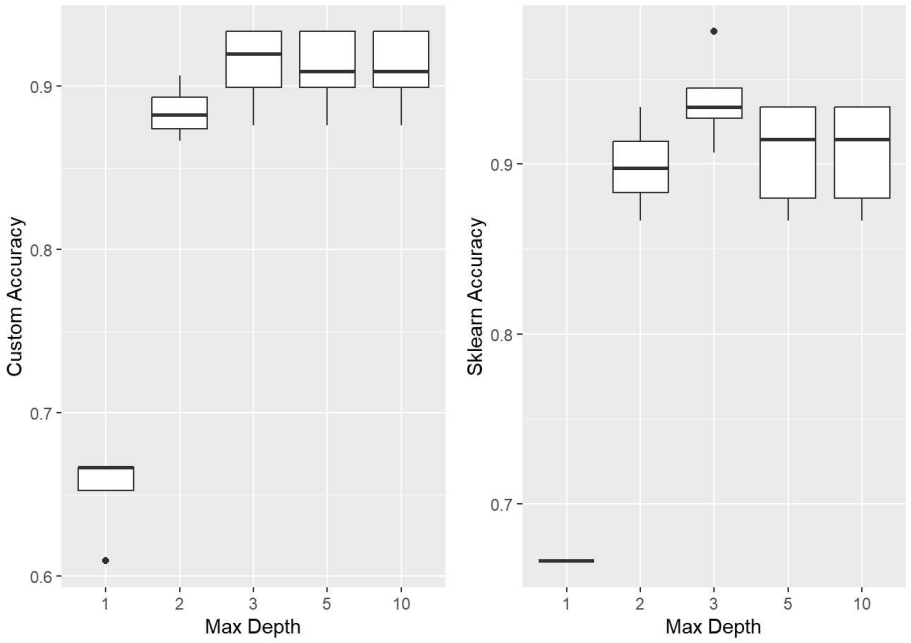
Large dataset (Breast Cancer)

Overall, the results showed the sklearn decision tree used up less memory (M = 0.0009, SD = 0.00000147) when classifying than the custom decision tree (M = 0.004, SD = 0.011) tested using the large datset.

4.2 Graphs of Results

4.2.1 Accuracy and Max_Depth- Small Dataset

Visualisations for both decision tree implementations showed how Accuracy varies with Max_Depth in the Iris dataset

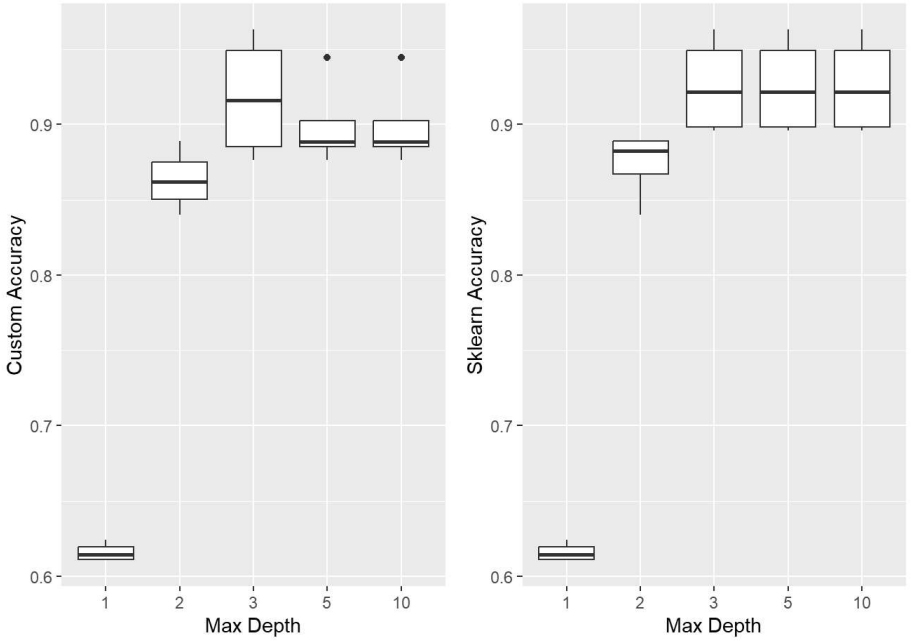


Iris: Boxplot for Accuracy and Max_Depth

This figure shows how accuracy differs depending on varied maximum levels of tree depth. Custom decision tree implementation shows distribution of greater accuracy levels at max_depth of 3 similarly to the sklearn decision tree, and showed the lowest accuracy level at max_depth of 1.

4.2.2 Accuracy and Max_Depth- Medium Dataset

Visualisation for accuracy and the hyper-parameter of max_depth in the Wine dataset.

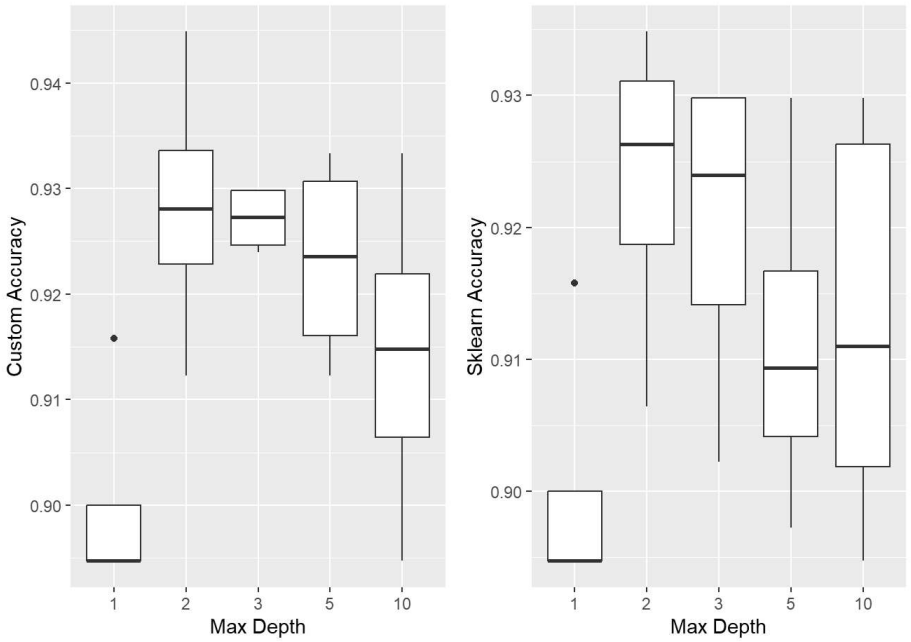


Wine: Boxplot for Accuracy and Max_Depth

This figure shows overall higher distribution of sklearn accuracy scores across all levels of max_depth compared to custom decision tree custom score. Excluding maximum depth of 1 where both implementations had similar distribution of low accuracy scores.

4.2.3 Accuracy and Max_Depth- Large Dataset

Visualisation for accuracy and the hyper-parameter of max_depth in the Breast Cancer dataset.



Breast Cancer: Boxplot for Accuracy and Max_Depth

This figure shows bigger distribution of high accuracy scores for the sklearn decision tree at maximum depth of 2 and 3. However, there is a bigger distribution for lower accuracy scores at maximum depth of 5 when compared against the custom decision tree.

Therefore these results show the effect of tree depth on accuracy for both decision trees suggesting that a decision tree's maximum depth should not be too high or too low.

4.3 Statistical Analyses

Paired T-tests were conducted to compare machine learning and computational aspects of both decision tree implementations.

Computational T-tests

A paired t-test comparing training times between the two decision tree implementations showed there was a statistically significant difference ($t(79)= 16.79$, $p < 2.2e-16$) showing the custom decision tree was 0.21 seconds longer.

When comparing memory usage between the two trees, there was a statistically significant difference ($t(79)= 4.61$, $p = 0.000015$) where there was a mean difference of 0.0011 MB.

However, when comparing the prediction times, there no statistical significant difference showing similar prediction times between the two implementations.

Machine Learning T-tests

Two-Way Analyses of variance (ANOVAs) were conducted to compare machine learning aspects against the hyper-parameters of “max_depth” and “sample_min_split”. The results are seen in Table 4 for the Iris dataset.

term <chr>	df <dbl>	sumsq <dbl>	meansq <dbl>	statistic <dbl>	p.value <dbl>	metric <chr>
max_depth	1	2.460969e-01	2.460969e-01	30.325134990	4.767269e-07	Custom Accuracy
sample_min_split	1	4.444444e-05	4.444444e-05	0.005476638	9.412014e-01	Custom Accuracy
max_depth:sample_min_split	1	4.763780e-05	4.763780e-05	0.005870136	9.391297e-01	Custom Accuracy
Residuals	76	6.167612e-01	8.115279e-03	*NA*	*NA*	Custom Accuracy
max_depth	1	2.460969e-01	2.460969e-01	30.325134990	4.767269e-07	Custom Recall
sample_min_split	1	4.444444e-05	4.444444e-05	0.005476638	9.412014e-01	Custom Recall
max_depth:sample_min_split	1	4.763780e-05	4.763780e-05	0.005870136	9.391297e-01	Custom Recall
Residuals	76	6.167612e-01	8.115279e-03	*NA*	*NA*	Custom Recall
max_depth	1	6.423336e-01	6.423336e-01	29.789164063	5.810676e-07	Custom Precision
sample_min_split	1	3.245022e-05	3.245022e-05	0.001504927	9.691568e-01	Custom Precision

1-10 of 28 rows Previous **1** 2 3 Next

Table 4. Anova Table for hyperparameters with machine learning aspects with Iris data

metric	p.value	term	df	statistic
Custom Accuracy	4.76726901753722e-07	max_depth	1	30.3251349897015
Custom Accuracy	0.94120139291457	sample_min_split	1	0.00547663771679197
Custom Accuracy	0.939129721315644	max_depth:sample_min_split	1	0.00587013629290406
Custom Accuracy	-	Residuals	76	-
Custom Recall	4.76726901753723e-07	max_depth	1	30.3251349897015
Custom Recall	0.94120139291457	sample_min_split	1	0.00547663771679197
Custom Recall	0.939129721315644	max_depth:sample_min_split	1	0.00587013629290401
Custom Recall	-	Residuals	76	-
Custom Precision	5.81067596455541e-07	max_depth	1	29.7891640633917
Custom Precision	0.969156843353103	sample_min_split	1	0.00150492666600375
Custom Precision	0.968068598718604	max_depth:sample_min_split	1	0.00161305623944566
Custom Precision	-	Residuals	76	-
Custom f1 score	6.53627897850854e-07	max_depth	1	29.4719237960728
Custom f1 score	0.957688082346089	sample_min_split	1	0.00283346538779667
Custom f1 score	0.956195888973668	max_depth:sample_min_split	1	0.00303705099144656
Custom f1 score	-	Residuals	76	-
Sklearn Accuracy	1.61252298091706e-05	max_depth	1	21.2268712770414
Sklearn Accuracy	0.680188061408266	sample_min_split	1	0.171228108417044
Sklearn Accuracy	0.669567951262206	max_depth:sample_min_split	1	0.183530915419456
Sklearn Accuracy	-	Residuals	76	-
Sklearn Recall	1.61252298091709e-05	max_depth	1	21.2268712770414
Sklearn Recall	0.680188061408265	sample_min_split	1	0.171228108417046
Sklearn Recall	0.669567951262206	max_depth:sample_min_split	1	0.183530915419456
Sklearn Recall	-	Residuals	76	-
Sklearn f1 score	8.51567029345283e-06	max_depth	1	22.8086979108081
Sklearn f1 score	0.765863644448874	sample_min_split	1	0.0893164335133211
Sklearn f1 score	0.757857228006935	max_depth:sample_min_split	1	0.0957338544252044
Sklearn f1 score	-	Residuals	76	-

These results showed “max_depth” having a statistically significant effect on machine learning aspects in the Iris dataset.

Table 5. Anova Table for hyperparameters with machine learning aspects with Wine data

metric	p.value	term	df	statistic
Custom Accuracy	9.08052654866357e-07	max_depth	1	28.5911306724276
Custom Accuracy	0.999999999999997	sample_min_split	1	1.6928021563632e-29
Custom Accuracy	0.999999999999997	max_depth:sample_min_split	1	1.58182957055717e-29
Custom Accuracy	-	Residuals	76	-
Custom Recall	1.14256675145377e-06	max_depth	1	27.9804947586353
Custom Recall	0.999999999999997	sample_min_split	1	1.75511421823546e-29
Custom Recall	0.999999999999998	max_depth:sample_min_split	1	9.15476243462323e-30
Custom Recall	-	Residuals	76	-
Custom Precision	1.72963570098187e-06	max_depth	1	26.8883908015506
Custom Precision	0.999999999999998	sample_min_split	1	4.22892612607914e-30
Custom Precision	0.999999999999999	max_depth:sample_min_split	1	1.87952272270184e-30
Custom Precision	-	Residuals	76	-
Custom f1 score	1.18170226302714e-06	max_depth	1	27.8913080096695
Custom f1 score	0.999999999999995	sample_min_split	1	4.3238971211085e-29
Custom f1 score	0.999999999999996	max_depth:sample_min_split	1	3.1767407420389e-29
Custom f1 score	-	Residuals	76	-
Sklearn Accuracy	7.12834000065832e-08	max_depth	1	35.6245001804683
Sklearn Accuracy	0.999999999999998	sample_min_split	1	4.56781868814906e-30
Sklearn Accuracy	0.999999999999999	max_depth:sample_min_split	1	1.78430417505823e-30
Sklearn Accuracy	-	Residuals	76	-
Sklearn Precision	2.50348701751026e-07	max_depth	1	32.0900194251478
Sklearn Precision	0.999999999999998	sample_min_split	1	5.56571742109015e-30
Sklearn Precision	0.999999999999998	max_depth:sample_min_split	1	5.56571742109015e-30
Sklearn Precision	-	Residuals	76	-
Sklearn Recall	2.24514495978803e-07	max_depth	1	32.3916086024792
Sklearn Recall	0.999999999999997	sample_min_split	1	1.8976659354444e-29
Sklearn Recall	0.999999999999997	max_depth:sample_min_split	1	1.51918408682391e-29
Sklearn Recall	-	Residuals	76	-
Sklearn f1 score	2.63039668704069e-07	max_depth	1	31.9533917227924
Sklearn f1 score	0.999999999999996	sample_min_split	1	2.31194143102298e-29
Sklearn f1 score	0.999999999999997	max_depth:sample_min_split	1	1.65529534410521e-29
Sklearn f1 score	-	Residuals	76	-

These results show that the hyper-parameter of maximum tree depth had an statistically significant effect on accuracy, recall, precision and f1 score for both decision tree implementations.

Table 6. Anova Table for hyperparameters with machine learning aspects with Breast Cancer data

metric	p.value	term	df	statistic
Custom Accuracy	0.763142722762242	max_depth	1	0.0914700552430133
Custom Accuracy	0.93660972909519	sample_min_split	1	0.00636730533009973
Custom Accuracy	0.884939371545208	max_depth:sample_min_split	1	0.021082298356726
Custom Accuracy	-	Residuals	76	-
Custom Recall	0.00587497529566127	max_depth	1	8.03490661330951
Custom Recall	0.916090219904528	sample_min_split	1	0.0111748025008875
Custom Recall	0.847978033572366	max_depth:sample_min_split	1	0.0370000350521791
Custom Recall	-	Residuals	76	-

metric	p.value	term	df	statistic
Custom Precision	0.000147896276938253	max_depth	1	15.9662560154907
Custom Precision	0.757588151537816	sample_min_split	1	0.0959537664372811
Custom Precision	0.574649447312379	max_depth:sample_min_split	1	0.3177051872982
Custom Precision	-	Residuals	76	-
Custom f1 score	0.306188117723315	max_depth	1	1.06129200810606
Custom f1 score	0.973873119628181	sample_min_split	1	0.00107971829023554
Custom f1 score	0.952478978429433	max_depth:sample_min_split	1	0.00357497276411941
Custom f1 score	-	Residuals	76	-
Sklearn Accuracy	0.910325465034171	max_depth	1	0.0127698708429999
Sklearn Accuracy	0.696739278574244	sample_min_split	1	0.153042471690027
Sklearn Accuracy	0.478738509452486	max_depth:sample_min_split	1	0.506727238942114
Sklearn Accuracy	-	Residuals	76	-
Sklearn Precision	0.00351847113856878	max_depth	1	9.07486066669898
Sklearn Precision	0.889623754573288	sample_min_split	1	0.0193895687970644
Sklearn Precision	0.800661941849564	max_depth:sample_min_split	1	0.0641993203083493
Sklearn Precision	-	Residuals	76	-
Sklearn Recall	0.0160126064419861	max_depth	1	6.06973520297034
Sklearn Recall	0.600067449019186	sample_min_split	1	0.277212393524074
Sklearn Recall	0.341076201272471	max_depth:sample_min_split	1	0.917856783282289
Sklearn Recall	-	Residuals	76	-
Sklearn f1 score	0.453757491457894	max_depth	1	0.567056057308648
Sklearn f1 score	0.671105430128446	sample_min_split	1	0.181718590704854
Sklearn f1 score	0.440346894349734	max_depth:sample_min_split	1	0.601674546388996
Sklearn f1 score	-	Residuals	76	-

However, in the breast cancer dataset “max_depth” hyperparameter only had a statistically significant effect on recall and precision not accuracy and f1 score for both decision tree implementations.

5. Discussion

5.1 Machine Learning Performance

The Sklearn Decision Implementation worked well through showing higher results of Machine Learning Performance scores; showing higher accuracy, precision, recall and f1 when compared against the custom decision tree and tested across the same performance measures.

5.2 Computational Performance

As previously mentioned, the sklearn implementation performed well showing faster testing times compared to the custom implementation as well as using up less memory. However, after a paired t-test comparing predicted time there was no statistically significant difference of prediction time, therefore suggesting both decision tree implementations were close in prediction time.

6. Conclusion

In conclusion, this project aimed to evaluate the machine learning and computational aspects of a sklearn library decision tree and a custom implemented decision tree to investigate their effectiveness.

The analysis showed that overall the sklearn decision implementation was more efficient scoring highly in machine learning (accuracy, precision, recall and f1 score) as well as computational (training time, prediction time and memory usage) aspects. However, it was shown that although there were a lot of statistically significant results, the sklearn prediction time was closely similar to the custom prediction time, therefore was deemed statistically insignificant.

Hyper-parameters were also measured to see their effect on machine learning aspects for both implementations. Although it was shown that there was no statistical effect with minimum sample split and its interaction with maximum depth, when just looking at maximum tree depth alone there was a significant effect on the machine learning aspects for both decision tree implementations.

7. Acknowledgements

I had some help from peers such as friends who take computer science and take a module on Artificial Intelligence, NumPy was used to help with coding the custom decision tree, Pandas was also used for converting results to .csv() files. For the R code the libraries used were: patchworks, broom, tidyverse, and ggplot2.

References

1.10. Decision Trees. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/tree.html#classification (https://scikit-learn.org/stable/modules/tree.html#classification)

Aaboub, F., Chamlal, H., & Ouaderhman, T. (2023). Statistical analysis of various splitting criteria for decision trees. Journal of Algorithms & Computational Technology, 17. https://doi.org/10.1177/17483026231198181 (https://doi.org/10.1177/17483026231198181)

Decision Tree Sklearn -Depth Of tree and accuracy. (n.d.). Stack Overflow. https://stackoverflow.com/questions/49289187/decision-tree-sklearn-depth-of-tree-and-accuracy (https://stackoverflow.com/questions/49289187/decision-tree-sklearn-depth-of-tree-and-accuracy)

GeeksforGeeks. (2024, October 11). ML | Gini Impurity and Entropy in Decision Tree. GeeksforGeeks. https://www.geeksforgeeks.org/gini-impurity-and-entropy-in-decision-tree-ml/ (https://www.geeksforgeeks.org/gini-impurity-and-entropy-in-decision-tree-ml/)

Importance of decision tree hyperparameters on generalization — Scikit-learn course. (n.d.). https://inria.github.io/scikit-learn-mooc/python_scripts/trees_hyperparameters.html# (https://inria.github.io/scikit-learn-mooc/python_scripts/trees_hyperparameters.html#):~:text=The%20hyperparameter%20max_depth%20controls%20the,the%20impact%20of%20the%20para

Kaushik, A. (2023, June 18). Gini Impurity and entropy for decision tree - Arpita Kaushik - medium. Medium. https://medium.com/@arpita.k20/gini-impurity-and-entropy-for-decision-tree-68eb139274d1 (https://medium.com/@arpita.k20/gini-impurity-and-entropy-for-decision-tree-68eb139274d1)

UCI Machine Learning Repository. (n.d.). https://archive.ics.uci.edu/ (https://archive.ics.uci.edu/)

What is sklearn? | Domino Data Lab. (n.d.). Domino Data Lab. https://domino.ai/data-science-dictionary/sklearn (https://domino.ai/data-science-dictionary/sklearn)