Group

- Dubai CW PG Thursday Group 6

Group Members

- Pratibha Yadubanshi | Mohamed Aman | Faizan Watare | Fardeen Khan

F21DL Coursework Part 3 – Decision Trees

In this report, we present the results and findings from Part 3 of the coursework. We focused on experimenting with decision tree parameters to optimize the model's performance. Key steps include the use of the provided test dataset, the application of decision trees, parameter tuning, assessment of overfitting, and an exploration of the Random Forest algorithm.

Using Decision Tree classifier on the training dataset

We have used two datasets initially to test the decision tree algorithm. First one is the original dataset and then the top 10 correlated features dataset from previous part of the coursework. We tested both on decision tree algorithm using different train test split and 10-Fold cross validation.

		Train Test Split (Test Size = 0.2)	Train Test Split (Test Size = 0.3)	10-Fold Cross Validation (Mean)
Original dataset	Accuracy	82.51%	81.91%	82.92%
	Precision	82.75%	81.95%	83.09%
	Recall	82.51%	81.91%	82.92%
	F1 Score	82.58%	81.89%	82.93%
Top 10 features correlated dataset	Accuracy	81.17%	80.43%	81.91%
	Precision	81.14%	80.49%	82.02%
	Recall	81.17%	80.43%	81.91%
	F1 Score	81.08%	80.42%	81.89%

This initial evaluation aimed to assess the Decision Tree Classifier's ability to learn from the training data and generalize to unseen instances. The decision tree exhibited reasonably high accuracy and other metrics.

Using Decision Tree classifier on the testing set

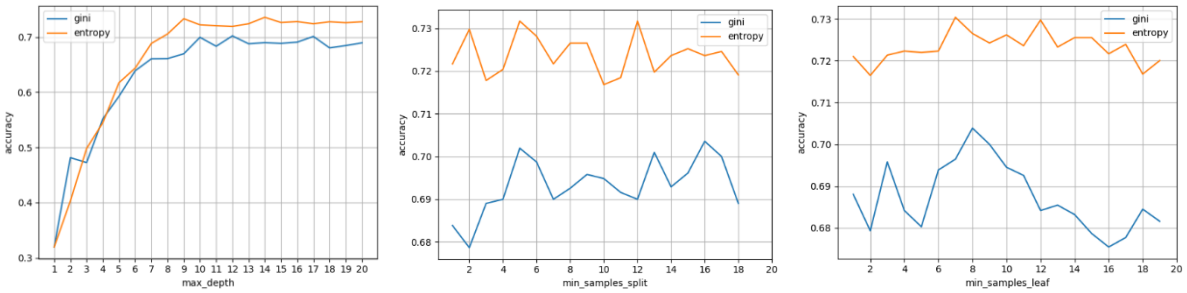
We applied the decision tree on the given testing dataset and calculated the metrices.

- Accuracy: 68.54%
- Precision: 68.56%
- Recall: 68.54%
- F1-Score: 68.43%

Decision tree exhibited a decrease in performance when applied to the testing dataset compared to the training set. This drop in performance suggests that it did not generalize well to new, unseen data. The metrics, including accuracy, precision, recall, and F1-score, are lower when evaluated on the testing dataset. The drop in performance indicates that the decision tree might have been overfitting.

Experiment with various decision tree parameters

We experimented with various decision tree parameters and hyperparameter tuning on depth of the tree, Splitting criteria, minimum sample split, and minimum sample leaf. We initially tried using GridsearchCV to find the best hyperparameter tuning; unfortunately, it was time-consuming to execute and we couldn't achieve any result. Hence, we did the manual tuning. We adjusted various parameters and plot graphs for the different hyperparameter values.



Based on the results and visualization, criterion entropy works well for our dataset than the gini index. The maximum depth converges after 10. Minimum sample is higher around 5, and minimum sample leaf gave higher accuracy when it is at 7 and 8. Meanwhile, we used RandomsearchCV with 5-fold cross validation to find the best tuning parameters.

	RandomsearchCV	Manual Tuning
Criterion	Entropy	Entropy
Max_depth	12	10
Min_saples_split	15	4
Min_samples_leaf	5	7
Accuracy	69.46%	73.17%

In comparing RandomSearchCV and manual tuning, we found a consistent preference for entropy as the criterion, and identified a more optimal set of hyperparameters through manual tuning. The resulting model achieved an accuracy of 73.17%, demonstrating the effectiveness of a more targeted tuning approach.

Moving 30% and 60% of instances from training data to testing data

We moved 30% and 60% of our instances from training dataset to testing dataset to observe the overfitting and generalisation issues. Different performance metrics were calculated, and the results are as follows.

30% Training Data in Testing Set:

- Accuracy: 73.65%
- Precision: 73.72%
- Recall: 73.65%
- F1-Score: 73.56%

60% Training Data in Testing Set:

- Accuracy: 71.72%
- Precision: 71.67%
- Recall: 71.72%
- F1-Score: 71.64%

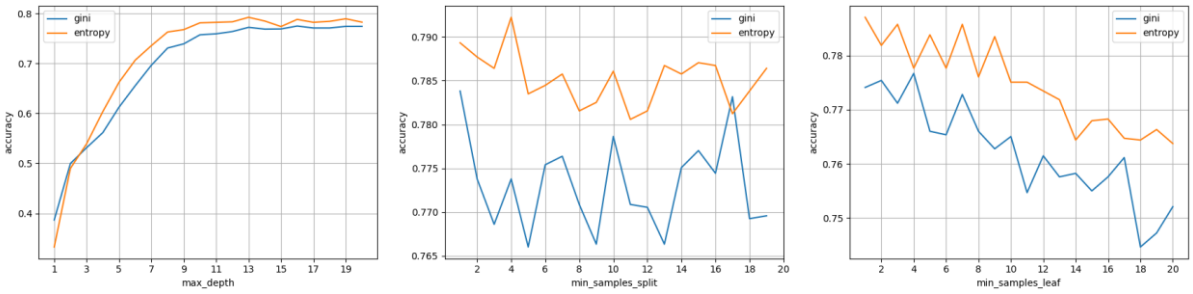
The observed decrease in performance metrics as a larger proportion of the training data moves into the testing set suggests a potential overfitting issue. The decline in accuracy, precision, recall, and F1-score as more training data is included in the testing set indicated that the model is becoming less effective at generalizing to new instances.

Random Forest

As extra research steps, we used random forest and calculated major metrices. It performed exceptionally well on the unseen data, resulting an outstanding performance. When using 10-Fold cross validation, it gave the following results.

- Mean Accuracy: 95.47%
- Mean Precision: 95.52%
- Mean Recall: 95.47%
- Mean F1 Score: 95.44%

Also, we tried tuning different hyperparameters on the Random Forest classifier.



Again, the entropy performed well than the gini index in all scenarios. We used randomizedsearchCV to check the best parameter and manual tuning as well.

	RandomizedsearchCV	Manual Tuning
Criterion	Entropy	Entropy
Max_depth	11	13
Min_saples_split	5	5
Min_samples_leaf	7	7
Accuracy	75.85%	77.90%

For Random Forest, our revised hyperparameter tuning results showed improved performance, with a manual tuning accuracy of 77.90%, further emphasizing the effectiveness of targeted adjustments in enhancing model capabilities.

Conclusion

The initial phase of this coursework showcased outstanding performance and accuracy during training. However, the change to a new testing dataset exposed a notable drop in performance, raising questions about its generalization capabilities. This highlighted the need for effective tuning to ensure optimal performance. The manual tuning process became critical, fine-tuning hyperparameters to elevate the model's accuracy. Random Forests offered more promising results. Hyperparameter tuning in Random Forests resulted in a significant increase in accuracy to 77.90%.

Use of Decision Trees and Random Forest algorithm showed the complicated balance between overfitting and generalization. It showed that the iterative enhancement and thoughtful parameter tuning are required to get a good performance, especially from an unbalanced dataset where it is prone to overfitting.

- Github Link: https://github.com/amannuhman/F21DL-CW

Thank You