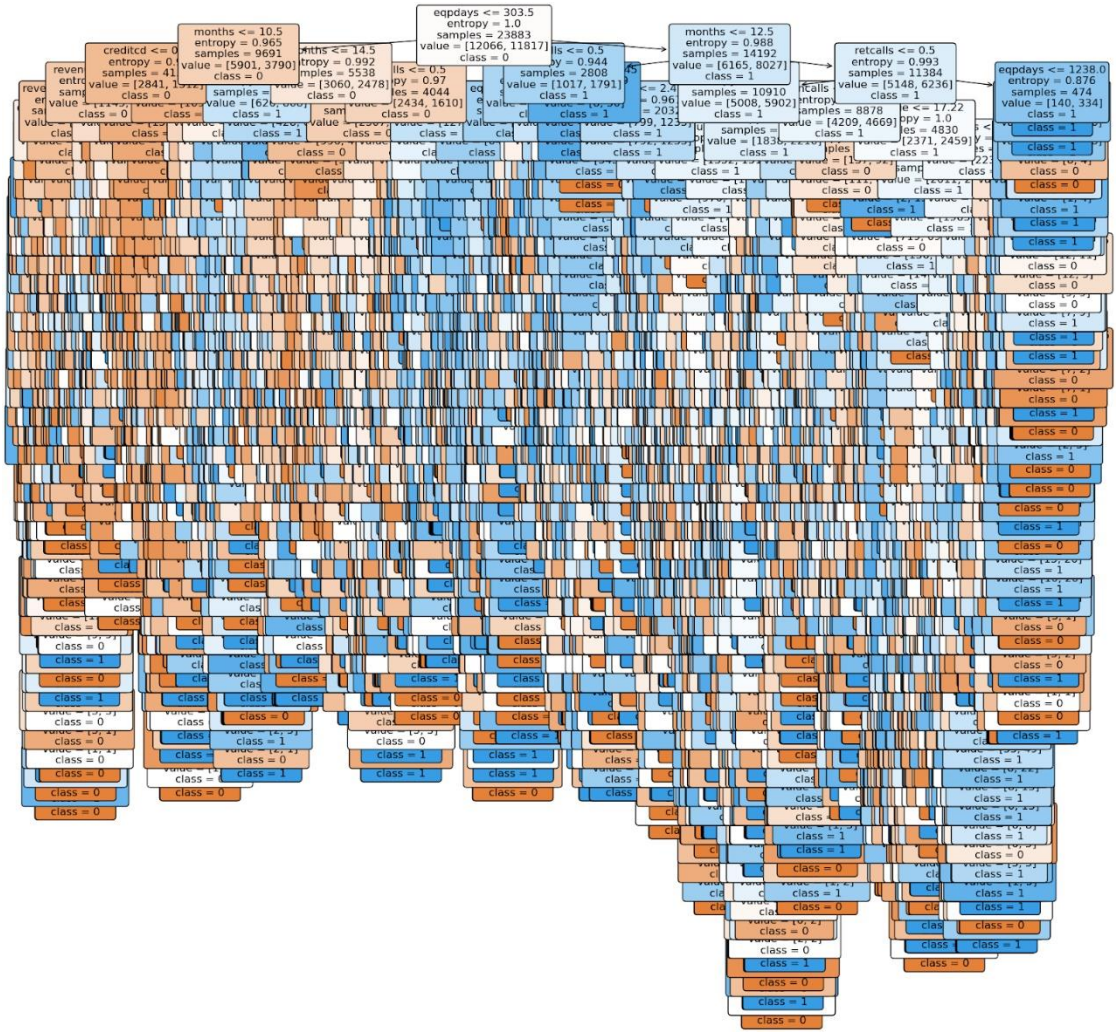
Part_1 Python

A. <u>Build a decision tree model that predicts whether a consumer will terminate his/her contract. In particular, I would like for you to create a decision tree using entropy with no max depth. Some possible issues / hints to think about: using training vs. test datasets.</u>

If we leveraged entropy as the criterion for branching and imposed no constraints on max depth of the tree, the resultant decision tree, with its intricate structure, reached a staggering number of layers, as evinced by the accompanying diagrams:



As we see, the decision tree creates hundreds of layers. Regardless of the helpfulness or not, the layers can be misleading and thus increase our workload while trying to analyze it. The profound depth of our tree, though possibly encompassing detailed data variations, poses several challenges:

1. **Overfitting Susceptibility**: Although an intricate decision tree may accurately represent the training dataset, it may not perform well when introduced to new data. This is because the tree may be capturing noise present in the training data as patterns, which can lead to poor generalization. In other words, the tree may perform exceptionally well on the training dataset but fail to accurately predict outcomes for new, unseen data.
2. **Interpretability Dilution:** One of the merits of decision trees is their innate interpretability. This advantage diminishes when confronted with an overwhelmingly dense structure. A significant advantage of decision trees is their inherent interpretability. However, with hundreds of layers, it becomes cumbersome and nearly impossible to derive meaningful insights manually.
3. **Generalization Concerns:** The deeper the tree, the finer the distinctions it makes among the input samples. This could lead to poor generalization to new data, as the tree might be too tailored to the training dataset.
4. **Computational Overhead:** Processing time and computational resources can significantly increase with tree depth, especially with large datasets.
5. **Analytical Challenge:** Analyzing and drawing insights from a vast tree becomes a formidable task. Instead of simplifying data understanding, it could further obfuscate it.

Additionally, the use of entropy ensures that the tree's splits are designed to maximize information gain. However, it is important to note that alternative criteria may lead to the formation of different tree structures. Therefore, it is recommended that these other criteria be taken into consideration in the future analysis.

B. Explore how well the decision trees perform for several different parameter values (e.g., for different splitting criteria)

To evaluate the performance of the decision tree model, we varied three parameters:

1. **Test Size (training data and testing data proportion) :**
   We experimented with different splitting proportions: from 0.2, 0.3, 0.4, to 0.5 to see the impact on model's performance.
2. **Max Depth:**
   We experimented with tree depths from 2 to 10 to determine the optimal depth that yields the best results without leading to an overly complex tree structure.

3. **Criterion Parameter:**

   We used both ‘entropy’ and ‘gini’ impurity to see which gives better classification.

   Entropy works by choosing splits that offer the most significant reduction in uncertainty.

   Gini Impurity operates by reducing the chance of misclassification.

After apply these parameter in the model, we checked the following performance metrics to measure the model performance:

- Accuracy: It tells how often the tree was right, providing a general overview of the model's performance across all predictions.
- Precision: From the predicted positives, how many were truly positive. It is vital when the cost of a false positive is high. For instance, mislabeling a non-churning customer as one likely to churn could lead to unnecessary retention offers and costs, precision becomes crucial.
- Recall: From the real positive cases, how many our tree got right. It becomes the metric of choice when missing out on a positive case has high consequences.
- F1-Score: A mix of precision and recall, gives a balanced perspective, considering both precision and recall. It's particularly useful when there's no significant preference between false positives and false negatives. In this case, we consider F1- Score as our performance metric since it provides the most balanced view.

As we take F1-Score as our primary performance metric, we sorted the result based on the F1-Score.

```
# Rank the results by F1-Score in descending order
ranked_results = results_depth_df.sort_values(by='F1-Score', ascending=False)

ranked_results[['Test Size', 'Max Depth', 'Accuracy', 'Recall', 'Precision', 'F1-Score']]
```

The result of the performance metrics shows in the following table:

	Test Size	Max Depth	Accuracy	Criterion	Recall	Precision	F1-Score
6	0.2	4	0.596640	entropy	0.756363	0.577351	0.654843
9	0.2	5	0.598524	gini	0.751707	0.579564	0.654506
7	0.2	4	0.596169	gini	0.755121	0.577087	0.654208
8	0.2	5	0.597896	entropy	0.749224	0.579314	0.653404
4	0.2	3	0.593971	entropy	0.754811	0.575213	0.652886
5	0.2	3	0.593971	gini	0.754811	0.575213	0.652886
28	0.3	5	0.596085	entropy	0.751400	0.576703	0.652561
29	0.3	5	0.596190	gini	0.750985	0.576843	0.652495
27	0.3	4	0.595876	gini	0.751192	0.576544	0.652381
26	0.3	4	0.595876	entropy	0.751192	0.576544	0.652381
11	0.2	6	0.598367	gini	0.738982	0.581015	0.650546

The initial two rows of our results table highlight the F1-Score values attained under distinct parameter settings. Upon comparison, we identified the parameters in the second row as yielding the superior predictive performance.

Furthermore, an interesting observation from the table is the consistency in the accuracy metric. Regardless of the specific parameters chosen, the accuracy metric stays notably stable, always hovering close to the 60% mark. Such consistency in accuracy stands as a testament to the model's resilience and reliability. It suggests that our model is not overly sensitive to parameter tweaks and generally holds its ground in terms of prediction accuracy across a broad spectrum of conditions.

C. Discuss the model (decision tree) that provides the best predictive performance from experimenting with different parameter values in question (b)

The combination that achieves the best performance (highest F1-Score) is:

Test Size: 20%

Max Depth: 5

Accuracy: 59.66%

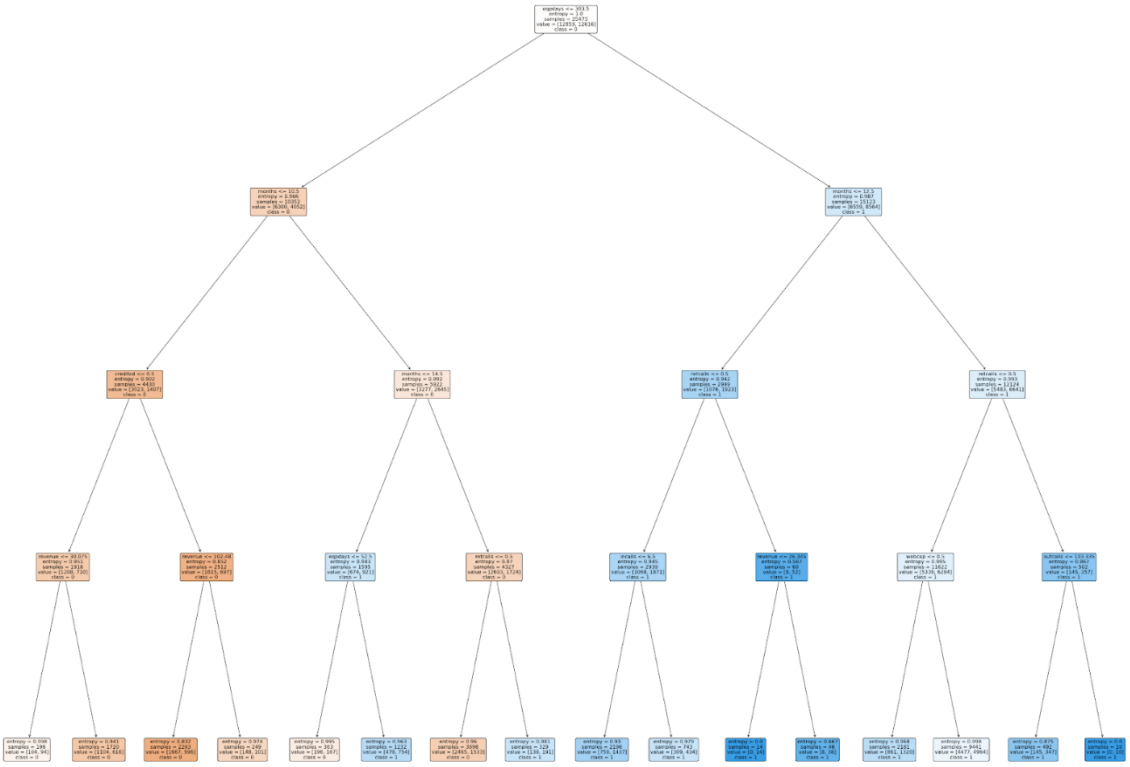
Recall: 75.15%

Precision: 57.95%

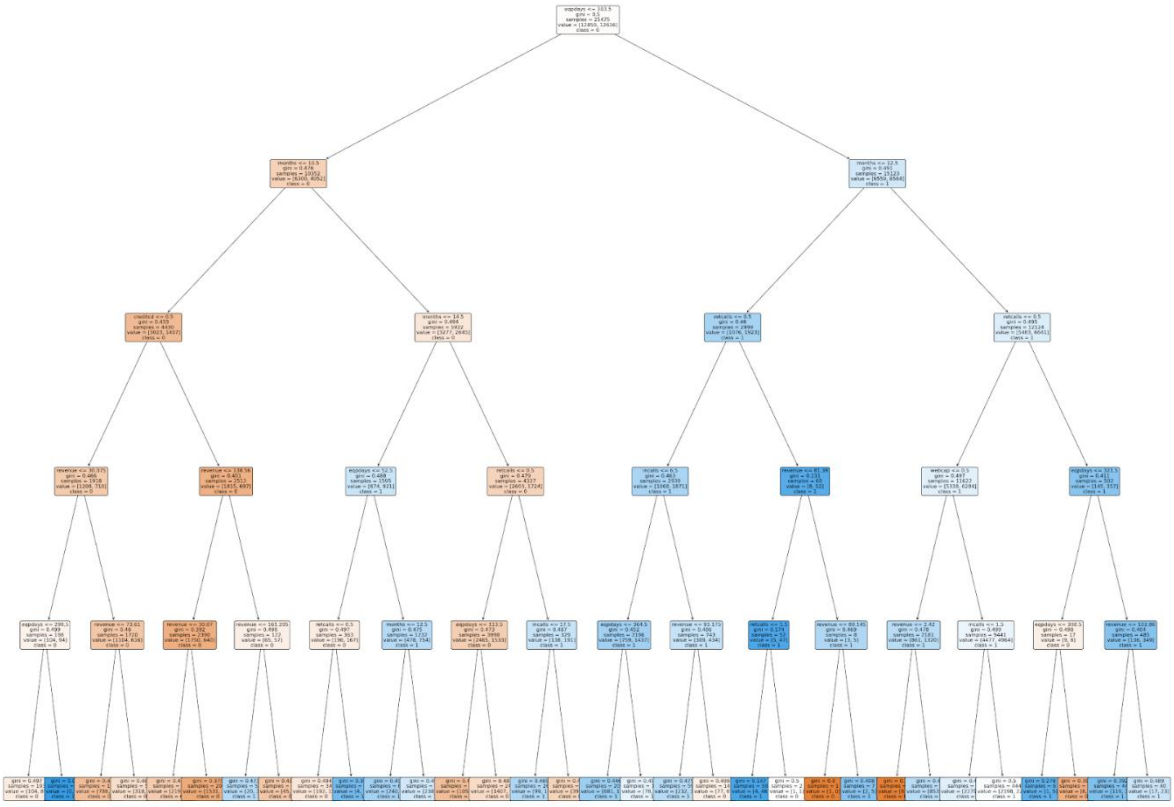
F1-Score: 65.45%

Criterion : Gini

Entropy graph:



Gini graph:



After carefully considering our options, we determined that utilizing the F1-Score as the primary performance metric would be the most reasonable course of action. The F1-Score strikes the balance between precision and recall, making it the ideal comprehensive metric for our goal.

A high recall indicates that we effectively identified a significant portion of the customers who are likely to churn. This is essential for the business, as it allows the company to take proactive actions to retain these customers.

A high precision means that among all the customers the model predicts will churn, a significant proportion actually do. This ensures that the company doesn't expend resources unnecessarily on customers who are not truly at risk of churning.

Based on our analysis, the best F1-score is achieved when the test size is set to 20%. This indicates that using 20% of the data as the test set and limiting the decision tree to a max_depth of 5 provides a balanced performance in terms of precision and recall, as reflected by the F1-Score.

It's worth noting that a max_depth of 5 also means that the decision tree will be relatively easy to interpret as it won't be too complex. This can be an advantage when we need to explain the model's decisions to stakeholders.

D. Present a brief overview of your predictive modeling process, explorations, and discuss your results. That is, you need to lay out the steps you have taken in order to build and evaluate the decision tree model. For instance, how did you explore the data set before you built the model? Write this report in a way that the upper level management of the team would understand what you are doing. Why is the decision tree an appropriate model for this problem? How can we evaluate the predictive ability of the decision tree? If you build decision trees with different splitting criteria, which decision tree would you prefer to use in practice? Make sure you present information about the model "goodness" (please report the confusion matrix, predictive accuracy, classification error, precision, recall, f-measure).

To build the decision tree, we perform four steps to fulfuill our goal:
1) Explore the dataset
2) Build the decision tree based on the dataset
3) Evaluate the model
4) Output the result and make a reasonable conclusion helping the stakeholders to plan business strategies.

The details of each step shows as follows:

1) Explore the dataset

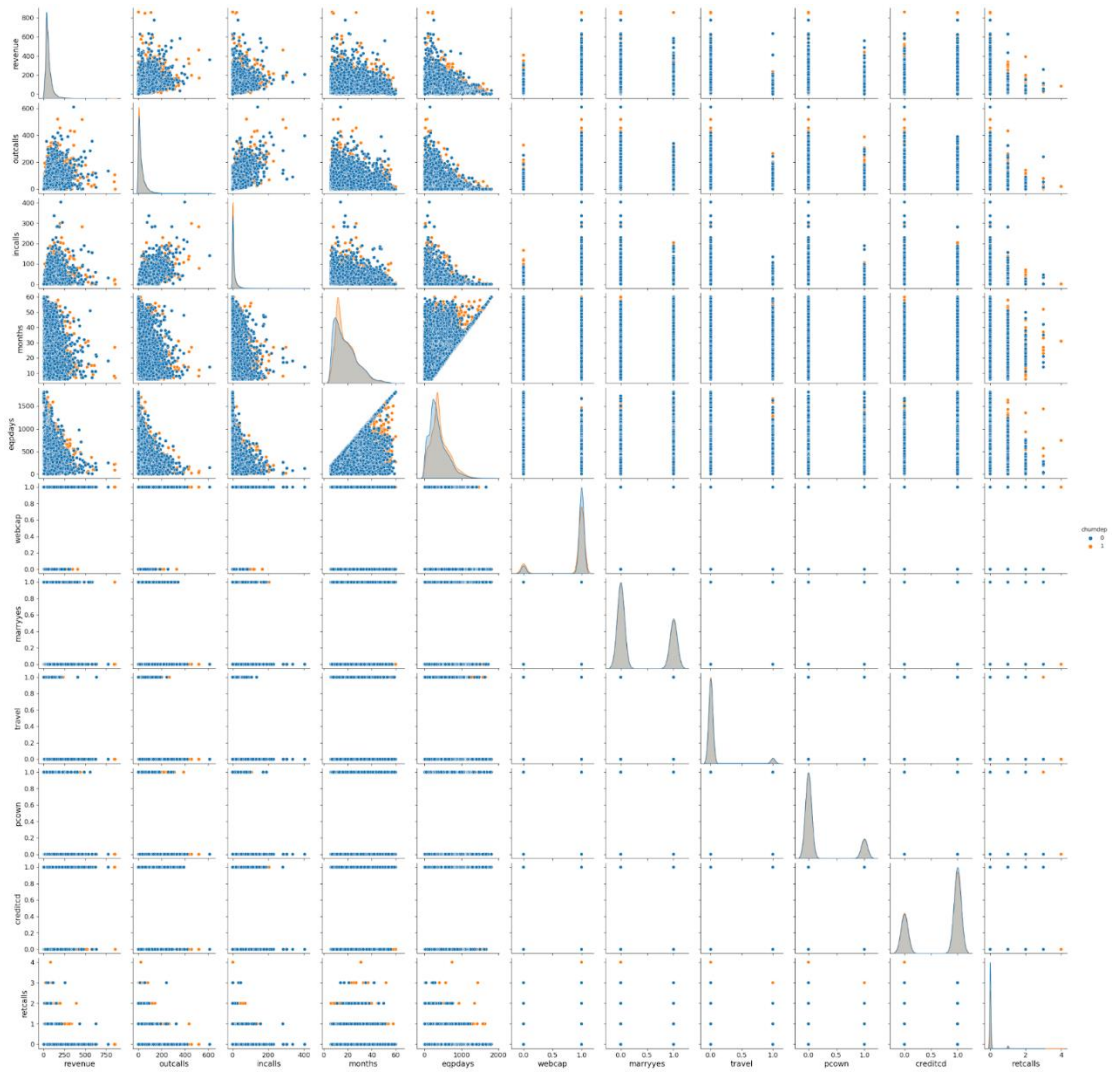
Initially, we identified the attribute values and the target value using both the data description and the dataset itself. This preliminary step was important to gain an overview of the dataset's structure and potential predictors. In the following outcome, we can realize that there is a binary result in the dataset. (0: customers will not churn; 1: customers will churn)

```
The frequency of instances per class is:  {0: 16006, 1: 15838}
The names of the three distinct classes are:  [0, 1]
```

From there, we extracted these attributes and target values, ensuring that the most relevant data was isolated for further analysis.

Second, to effectively utilize the data, we began by filtering out null values and any unreasonable entries. For example, negative revenue values could significantly skew our results, acting as potential interference. This data cleaning process is fundamental to model accuracy. The presence of such values, if unattended, can lead to spurious correlations or misleading patterns.

Upon investigating the target variable, we discovered that it had two distinct values: 0 and 1. This binary nature of the target variable suggested the suitability of classification algorithms for our analysis. To better understand the dataset and its underlying patterns, we visualized it by plotting each attribute. Using the target variable as the hue provided a color-coded distinction, with 'churndep = 0' represented in blue, ‘churndep = 1’ represented in orange. These plots, more than just visually appealing, are instrumental in initial exploratory data analysis. They reveal potential trends, correlations, or anomalies, aiding in the selection of relevant features and the subsequent modeling process. By examining the plots, we were able to discern the relationship between each attribute and the target variable 'churndep', setting the stage for more advanced analytical techniques.



2) Build the decision tree based on the dataset

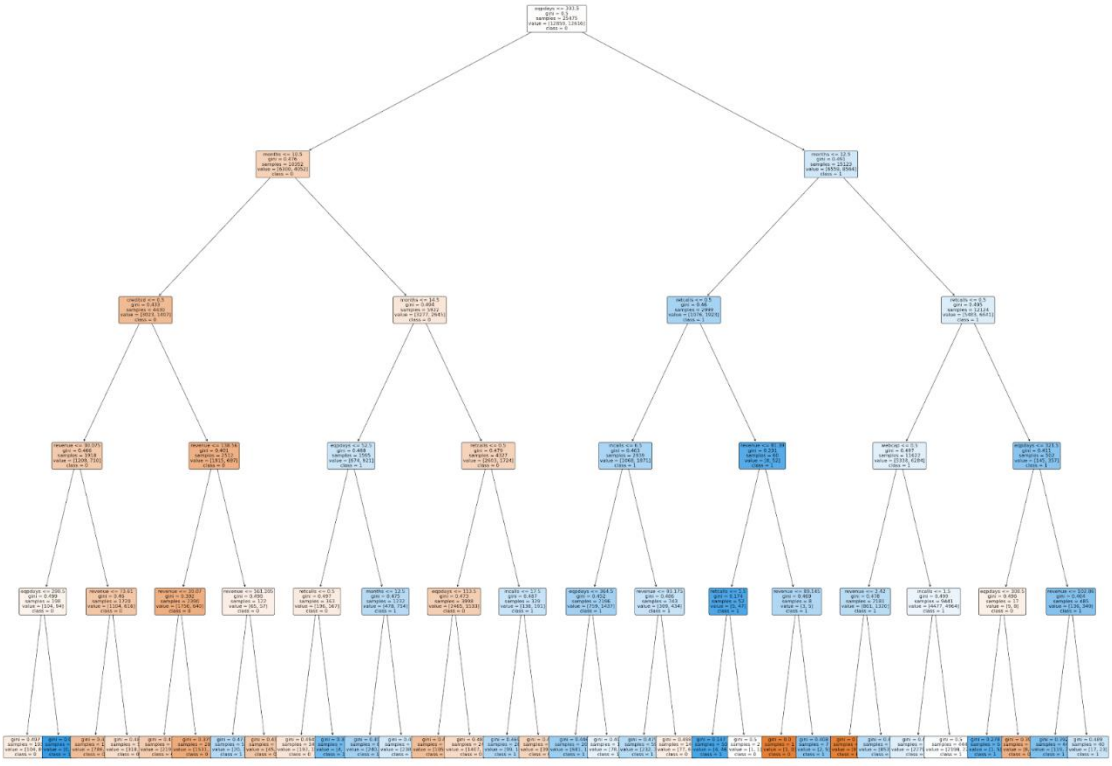
When it comes to predicting customer behavior, we aimed for a model that's both powerful and easily interpretable. Here's why decision trees fit the most:

- Simplicity & Visualization: decision tree works like a flowchart, making decisions based on asking a series of straightforward questions. This simplicity allows us to visually chart our decision-making process. This visual representation is invaluable for stakeholders who aren't data scientists. They can see and understand the flow of decisions without getting buried in numbers.
- Flexibility with Data Types: Many models are strict about the kind of data they can process. Some are tailored for numbers, others for categories. However, decision trees can comfortably process both numerical and

categorical data. Considering our target variable 'churndep' is categorical, decision trees offer us the flexibility we need without compromising on accuracy.

- Minimal Data Preparation: A significant portion of data analysis goes into preparing data to fit for the model. Decision trees, however, are more forgiving once we have our data divided into training and testing segments. This can speed up our modeling process, ensuring we deliver insights faster.

Starting from the top, 'eqpdays' sits as our root node. This position indicates its pivotal role in influencing decisions. Every branch thereafter represents a series of conditions leading to an eventual prediction. This tree can act as a quick reference to understand which factors most heavily influence customer decisions.



3) Evaluate the model

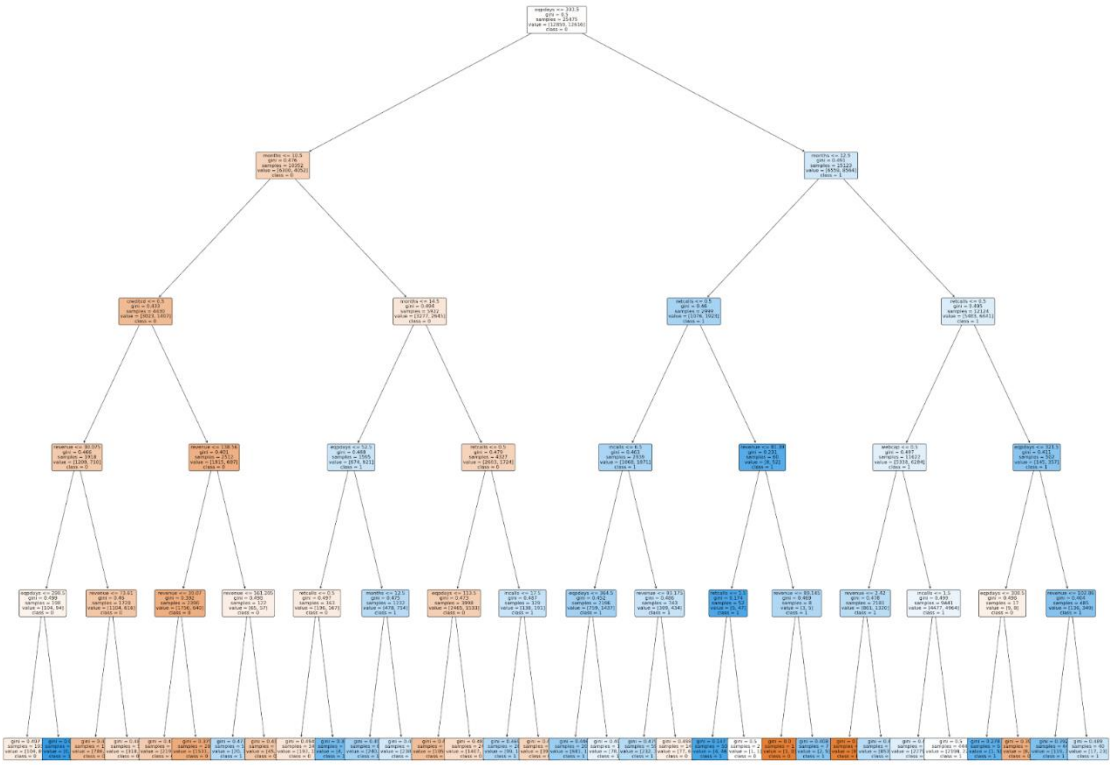
To evaluate the model, we apply several distinct parameters to test our model as we mentioned in question C.

	Test Size	Max Depth	Accuracy	Criterion	Recall	Precision	F1-Score
6	0.2	4	0.596640	entropy	0.756363	0.577351	0.654843
9	0.2	5	0.598524	gini	0.751707	0.579564	0.654506
7	0.2	4	0.596169	gini	0.755121	0.577087	0.654208
8	0.2	5	0.597896	entropy	0.749224	0.579314	0.653404
4	0.2	3	0.593971	entropy	0.754811	0.575213	0.652886
5	0.2	3	0.593971	gini	0.754811	0.575213	0.652886
28	0.3	5	0.596085	entropy	0.751400	0.576703	0.652561
29	0.3	5	0.596190	gini	0.750985	0.576843	0.652495
27	0.3	4	0.595876	gini	0.751192	0.576544	0.652381
26	0.3	4	0.595876	entropy	0.751192	0.576544	0.652381
11	0.2	6	0.598367	gini	0.738982	0.581015	0.650546

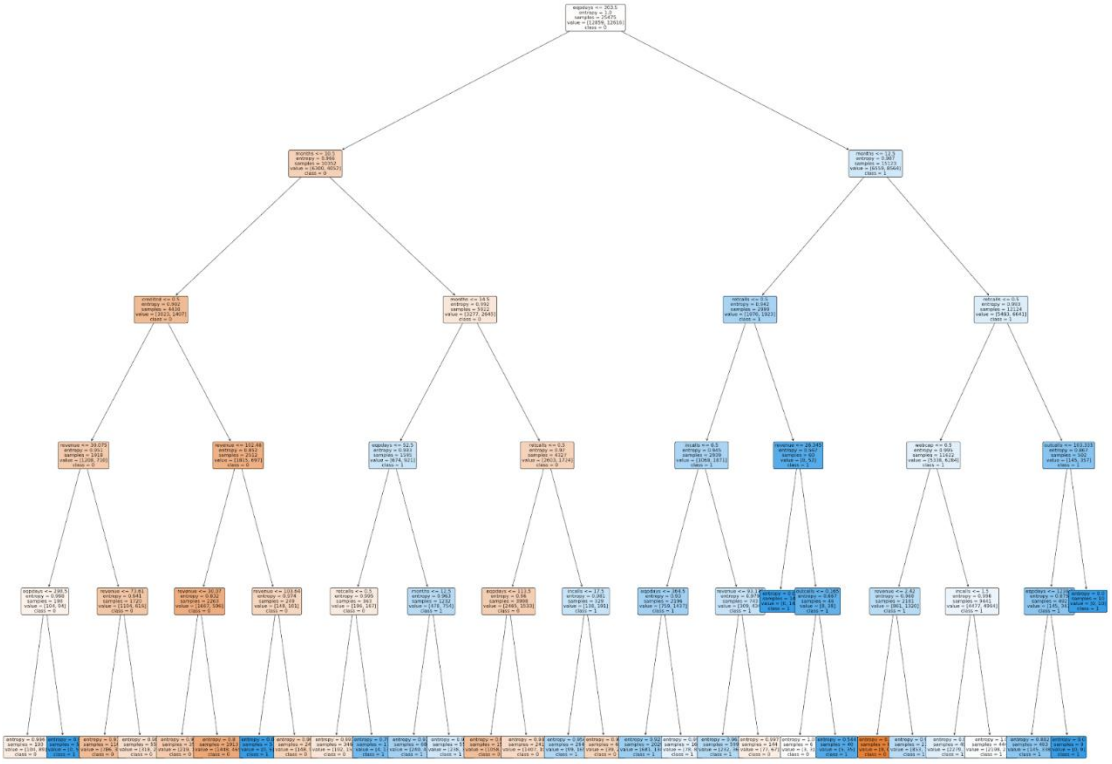
4) Output the result and make a reasonable conclusion helping the stakeholders making decision.

Here are the plot based on different criteria ‘Gini’ and ‘Entropy’

Gini



Entropy - overall too big so we used gini instead



Root Node: This is the top-most node and represents the entire dataset. When building a decision tree, the algorithm determines the best feature to split on at the root node to achieve the best separation based on the chosen criterion (entropy, Gini impurity, etc.). Examining the root node can give you insight into which feature is the most important or informative for your entire dataset.

Leaf Nodes: These are the nodes at the bottom of the tree where no further splitting occurs. They represent the final decisions or outcomes. The leaf nodes provide the predicted class or value (in the case of regression trees) for data instances that follow the path from the root to that leaf. Examining leaf nodes helps you understand the final decisions made by the tree

Confusion Matrix Evaluation:

To straightforwardly evaluate our classification model's performance, we generated a confusion matrix using parameters: a test split size of 0.2, criterion set to 'gini', and max depth set at 5. The elements of the confusion matrix have the following interpretations in the context of predicting customer churn:

- True Positive (TP = 2422): This represent customers who the model predicted would churn, and they indeed did. In essence, the model successfully flagged 2,422 customers who ended up churning.
- True Negative (TN = 1390): These are customers who the model predicted would not churn, and they indeed did not. Our model rightly anticipated that 1,390 customers would continue their subscriptions or services.
- False Positive (FP = 1757): This category involves customers whom the model erroneously predicted would churn, but they did not. Specifically, 1,757 customers were inaccurately identified by the model as potential churners, but they chose to stay.
- False Negative (FN = 800): These are customers where the model predicted that customers would stay, but they eventually churned. Thus, our model missed 800 customers who decided to leave.

From the normalized matrix:

- True Negative Rate (Specificity) = 0.44: 44% of the actual non-churning customers were correctly identified. This means that less than half of the loyal customers were correctly identified by the model.
- False Positive Rate = 0.56: 56% of the customers who didn't churn were incorrectly flagged by the model as at risk of churning.
- False Negative Rate = 0.25: 25% of the customers who churned were missed by the model. This means one in every four customers who churned was not flagged by the model.
- True Positive Rate (Sensitivity/Recall) = 0.75: 75% of the churning customers were correctly identified. This is a strong point, indicating that the model is quite good at identifying those customers who are at risk of churning.

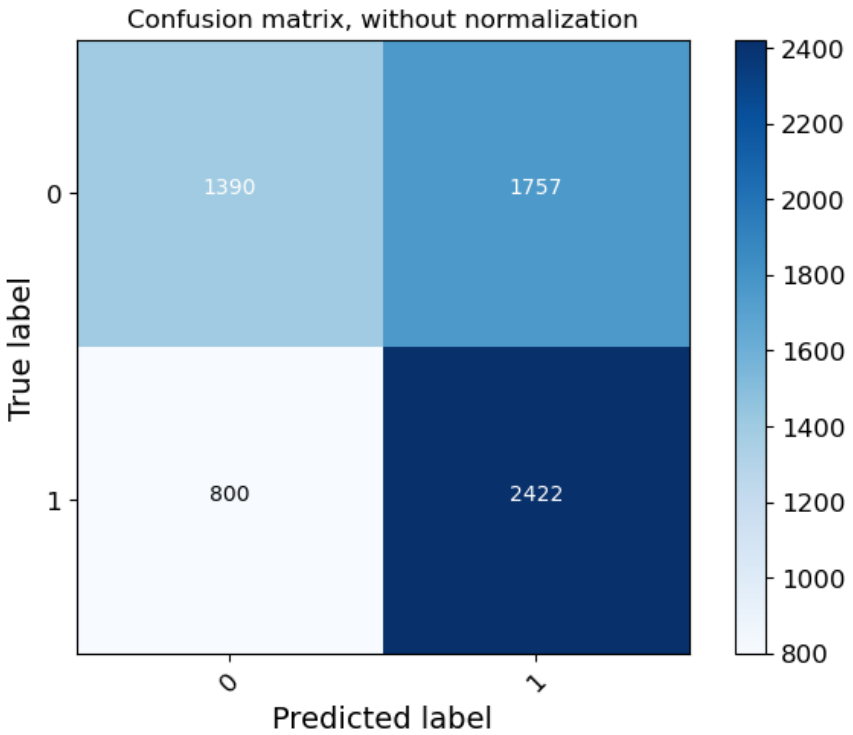
In terms of customer churn, our model is relatively eddective in identifying customers who are likely to churn, with a recall rate of 75%. This is crucial because identifying such customers allows businesses to take proactive measures to retain them.

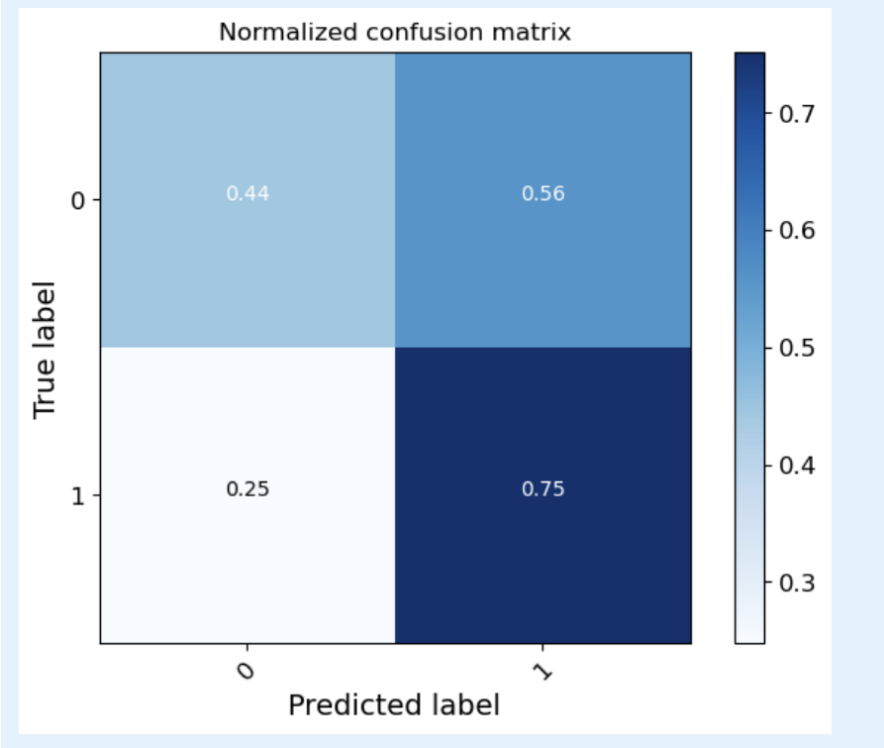
However, the model faces a challenge in correctly identifying loyal customers. The model tends to misclassify a significant number of loyal customers as potential churners. This might lead to unnecessary retention efforts or offers extended to customers who wouldn't leave.

The high false positive rate means that the model might be a bit conservative and tends to predict churn even when the customer is likely to stay.

This information can assist businesses in refining their retention strategies and allocating resources effectively. They may decide, for instance, that it is worthwhile to falsely identify some non-churners if it means catching more actual churners. On the other hand, businesses might aim to further refine the model to reduce the number of false positives.

```
Confusion matrix, without normalization
[[1390 1757]
 [ 800 2422]]
Normalized confusion matrix
[[0.44 0.56]
 [0.25 0.75]]
```





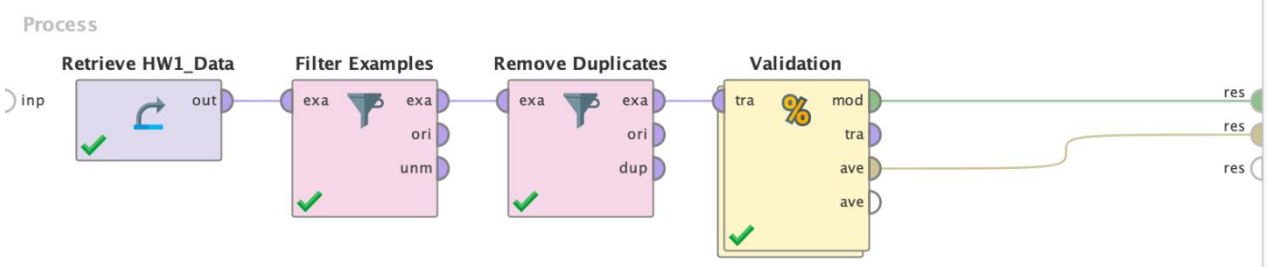
Part_2 RapidMiner

Processes:

The process diagram demonstrates the general steps to derive the decision tree in RapidMiner, including filter, remove duplicates, and validation consisting of decision tree function.

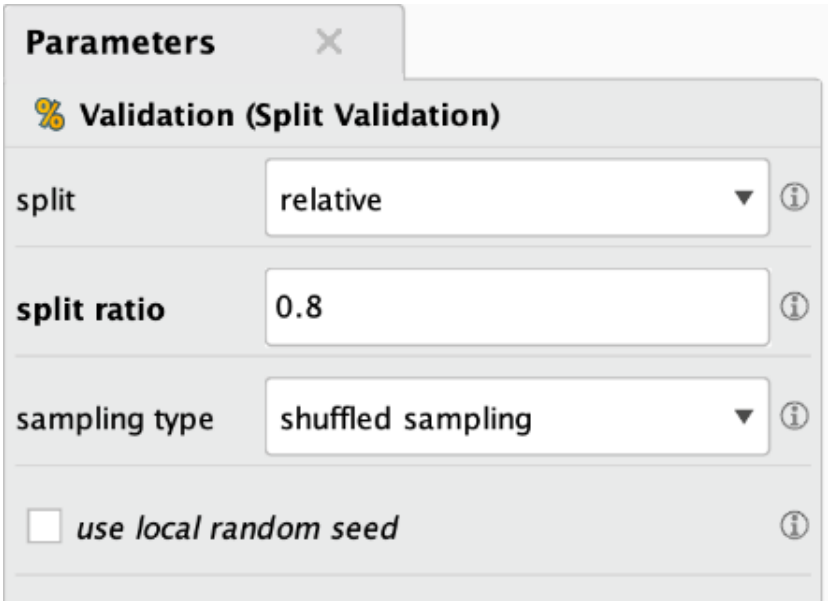
We filtered out any ‘eqpday’ that are negative to perform an initial data cleaning work and obtain a cleaner datasets ready to partition.

Next, we further remove duplicates to ensure no duplicate observation involved that will distort our decision tree model.

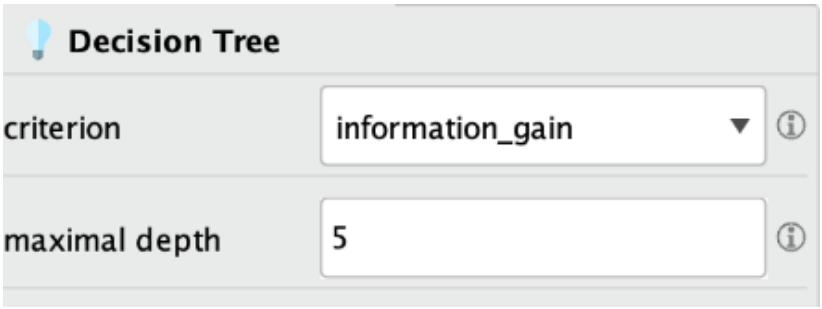


Parameters:

We split the dataset by 0.8 to the training model and 0.2 to the testing model, according to our performance test results generated in previous steps in Part_1.



We also set the maximal depth of the decision at 5 based on the same performance test results generated before in Part_1.



Decision Tree:

Results:

The metrics that we use to make our Performance Evaluations as follows:

Accuracy: 59.85%

Precision: 64.92%

Recall: 42.16%

F1-Score: 51.12%

	true 1	true 0	class precision
pred. 1	2473	1833	57.43%
pred. 0	722	1336	64.92%
class recall	77.40%	42.16%	

The 60 % accuracy rate indicates that we have a moderately good model, but there is certainly room for improvement. This means our model correctly predicts whether a consumer will terminate their contract approximately 60% of the time.

However, it's essential to look other index to understand the model's nuances:

Precision of 64.92%: This indicates that when our model predicts that a consumer will terminate their contract, it's correct about 64.92% of the time. So, roughly two-thirds of our positive predictions are accurate, while the other third are false alarms.

Recall of 42.16%: This metric reveals that our model can identify only 42.16% of all actual contract terminations. This means that there's a significant number of consumers who terminate their contracts that our model fails to identify.

F1-Score of 51.12%: The F1-Score gives a balanced measure that considers both precision and recall. An F1-Score of 51.12% suggests that our model's balance between precision and recall is average. In scenarios where both false positives and false negatives have considerable implications, this score becomes particularly crucial.