Dataset: Customer Churn Rate

Link: https://www.kaggle.com/datasets/divu2001/customer-churn-rate

NB: https://github.com/jlixander/ml_622/blob/main/DT_RF.ipynb

Article: The GOOD, The BAD & The UGLY of Using Decision Trees – DeciZone

Link: https://decizone.com/blog/the-good-the-bad-the-ugly-of-using-decision-trees

Decision Trees (DT) are an essential algorithm used by Data Scientists for both classification and regression tasks. Decision trees are highly valued for their interpretability, their capacity to handle non-linear relationships, and their effectiveness with non-parametric distributions. In addition, they provide the ability to identify key features through the hierarchical structure of the tree, where the top nodes indicate the greatest importance.

In this study, we will train two Decision Trees and a Random Forest model to analyze a public dataset shared on Kaggle. The dataset contains details about customers of a given bank. Features include demographics, income, and product engagement metrics. There are a total of 10,000 observations, spanning a total of 14 columns. The target variable is 'Exited', which identifies if a person is no longer a bank customer. Moreover, we will explore the behavior of randomizing feature node order for Decision Trees and evaluate how to tackle known weaknesses of the algorithm. We will gauge performance by evaluating classification scores.

Lastly, all three models were trained using cross-validation and by performing gridsearch(f1) to find the most optimal hyper-parameters.

1. Data Preprocessing

To preprocess the data, several transformations took place:

- Removing ID Columns: these columns are typically removed because they are non-informative; no predictive-power gains are observed.
- One-Hot-Encoding: two categorical features were found and encoded, 'Geography' and 'Gender'.

2. Checking For Class Imbalance

The dataset was checked for class imbalance and a 2037/7963 split between '1', and '0' was found, respectively. To address this issue, under-sampling of the majority class, and SMOTE was explored. It was found that the model performed approximately 10% better with SMOTE in terms of accuracy. However, the positive ('1') class underperformed yielding accuracy, recall, and precision scored in the low 50% range. Hence, the undersampled dataset was chosen because it displayed a balanced score between recall and precision.

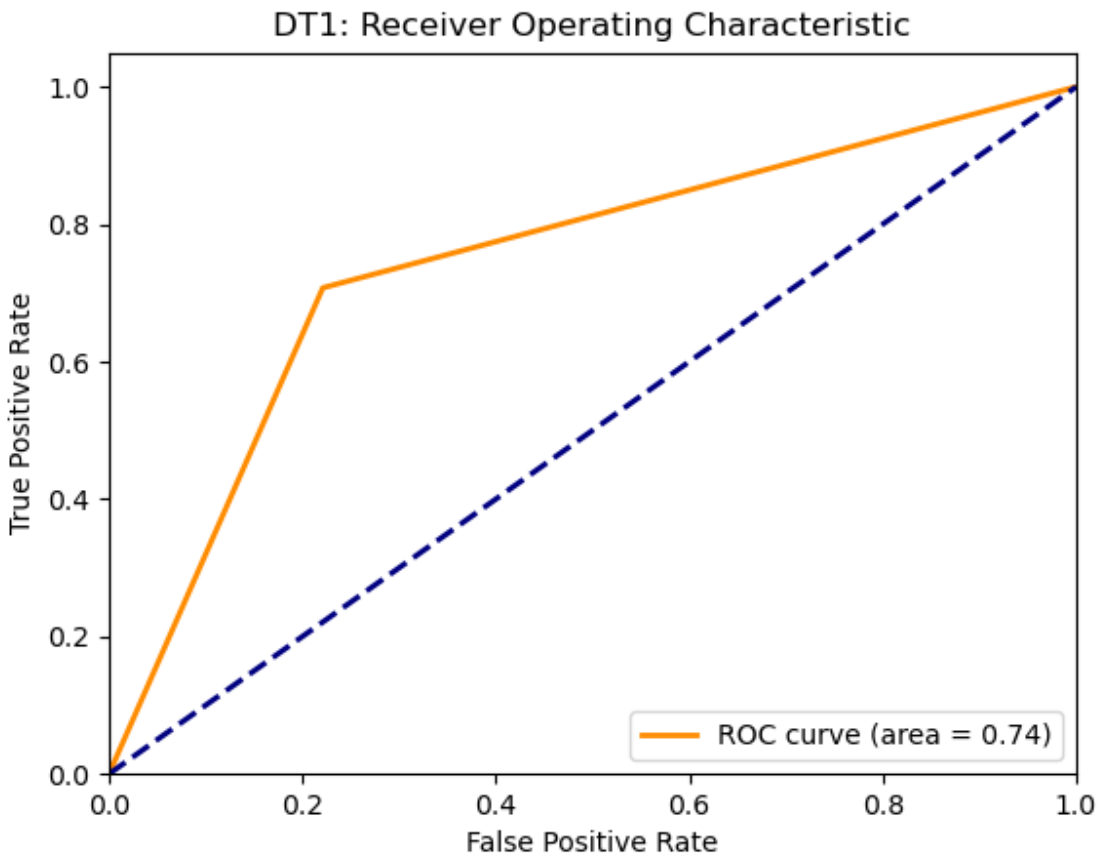
3. Decision Tree Model 1

This model was trained by using the 'best' splitter found by the algorithm. That is, the most important features were used in descending order of importance. The root node being 'Age <= 41.5'. The best parameters were found to be 'gini', max_depth=10, min_samples_leaf=2, min_samples_split=5.

Table 1: Decision Tree Model 1 Scores

	precision	recall	f1-score	support
0	0.717593	0.778894	0.746988	398.000000
1	0.770235	0.707434	0.737500	417.000000
accuracy	0.742331	0.742331	0.742331	0.742331
macro avg	0.743914	0.743164	0.742244	815.000000
weighted avg	0.744527	0.742331	0.742133	815.000000

Figure 1: Decision Tree Model 1 ROC



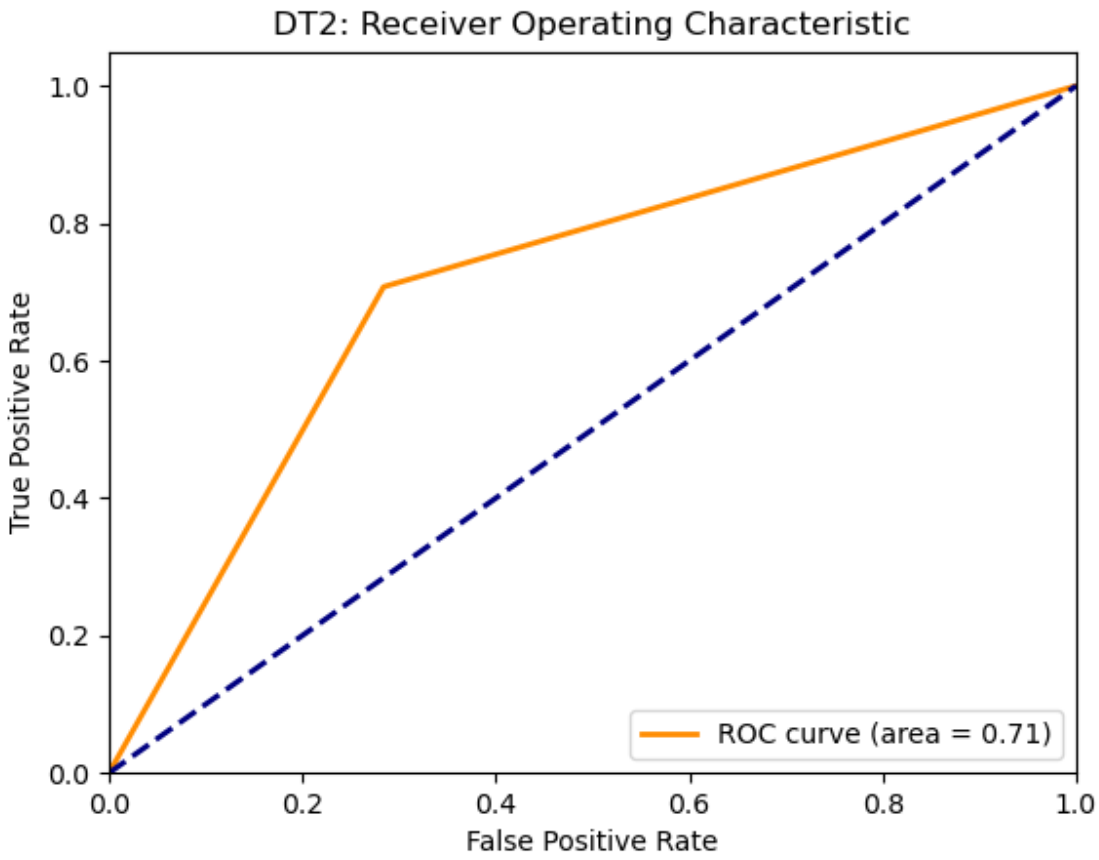
4. Decision Tree Model 2

This model was trained by using the 'random' splitter parameter provided by the Scikit-Learn. That is, no order of importance is used to arrange nodes. The root node became 'Age <= 46.53'. The best parameters were found to be 'entropy', max_depth=10, min_samples_leaf=1, min_samples_split=10.

Table 2: Decision Tree Model 2 Scores

	precision	recall	f1-score	support
0	0.700246	0.716080	0.708075	398.000000
1	0.723039	0.707434	0.715152	417.000000
accuracy	0.711656	0.711656	0.711656	0.711656
macro avg	0.711642	0.711757	0.711613	815.000000
weighted avg	0.711908	0.711656	0.711696	815.000000

Figure 2: Decision Tree Model 2 ROC



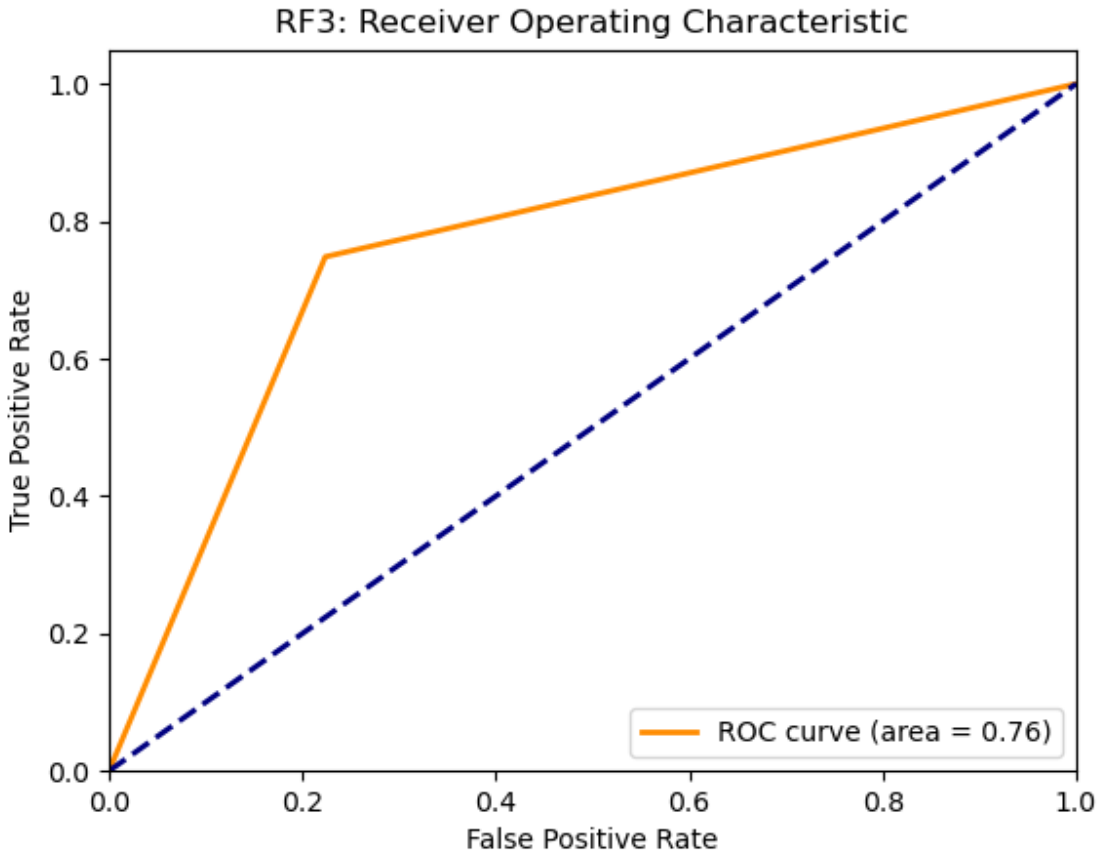
5. Random Forest Model

The Random Forest model performed best with max_depth=7, max_features=0.75, and n_estimators=100. It yielded an accuracy score of 76.19% with a balanced precision and recall score for both classes. Random Forest models are known to outperform Decision Trees. This could be due to the bootstrap aggregation technique employed by RF - essentially it reduces variance by taking the averages of many randomly generated trees.

Table 3: Random Forest Model 3 Scores

	precision	recall	f1-score	support
0	0.746377	0.776382	0.761084	398.000000
1	0.778055	0.748201	0.762836	417.000000
accuracy	0.761963	0.761963	0.761963	0.761963
macro avg	0.762216	0.762292	0.761960	815.000000
weighted avg	0.762585	0.761963	0.761980	815.000000

Figure 3: Random Forest Model 3 ROC



6. Discussion

Decision Trees are known to be prone to over-fit data easily, particularly when the data is too complex or noisy. Nevertheless, they remain an important tool in Data Science due to their simplicity and interpretability. The article published by Decizone.com highlights that evolvement or cumulative changes in a decision tree can be "BAD". However, this adaptability can actually be an advantage in data science,

especially when new, relevant features are added to the model. Such enhancements help in better generalizing the model, thereby increasing its predictive accuracy.

In response to concerns about decision complexity, which can be overwhelming with deep trees and numerous features, employing ensemble methods like Random Forests or gradient boosting offer a solution. These ensemble methods take advantage of multiple decision trees to create a more robust model that counters over-fitting by averaging the results, thereby reducing variance without increasing bias, see *figure 6,7, and 8*. It not only addresses the issue of complexity by distributing decision across simpler trees, but also enhance model stability and predictive accuracy.

Nonetheless, I agree that viewing deep decision trees can be daunting when the depth is too deep. Depending on how many features, and the scale of numerical features, the tree can grow dramatically making it very tedious to follow branches, see *figures 4 and 7* for reference.

Figure 4: Decision Tree Model 1 Tree Visualization

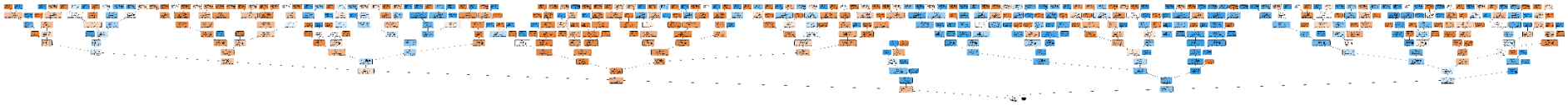


Figure 5: Decision Tree Model 2 Tree Visualization

Figure 6: Random Forest Tree 1

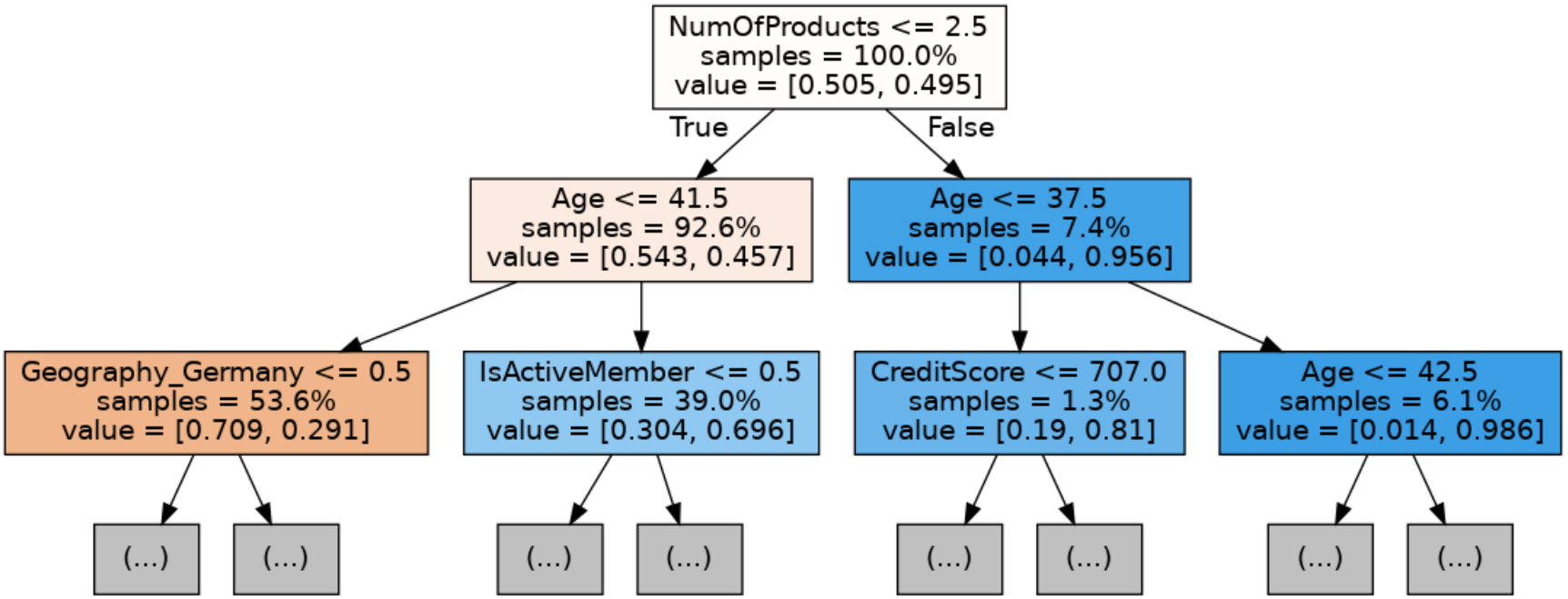


Figure 7: Random Forest Tree 2

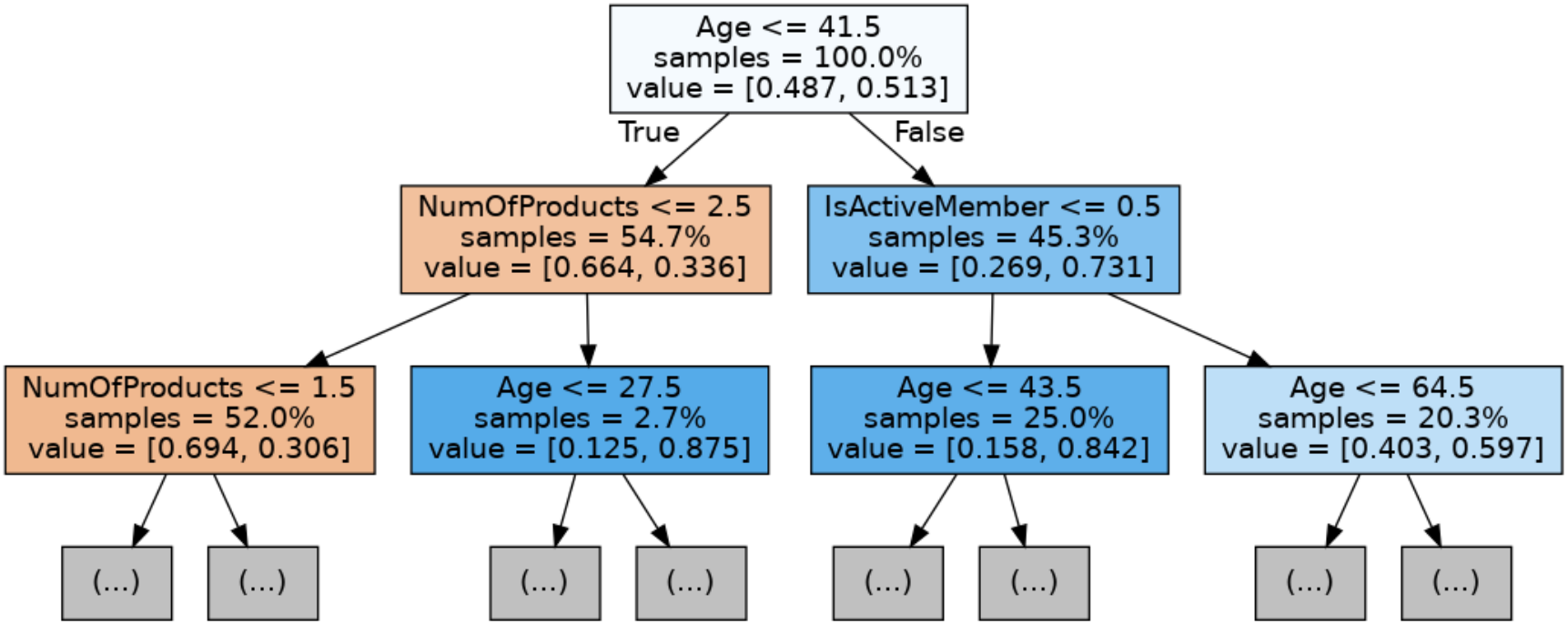


Figure 8: Random Forest Tree 3