A Comprehensive Review of Decision Tree Classification for Custom Dataset

Table of Contents

[Overview 2](#_Toc184188889)

[Introduction 2](#_Toc184188890)

[DT model Splitting principle 2](#_Toc184188891)

[Gini Impurity 2](#_Toc184188892)

[Information Gain 2](#_Toc184188893)

[Preprocessing & Dataset Selection 3](#_Toc184188894)

[DT Model Training 5](#_Toc184188895)

[Evaluation of the Model 6](#_Toc184188896)

[Model Hyperparameter tuning 8](#_Toc184188897)

[Conclusion and Findings 11](#_Toc184188898)

[References 11](#_Toc184188899)

# Overview

For this tutorial a Decision Tree (DT) is used which is a very effective and visually appealing machine learning (ML) algorithm. A major strength of the DT model is its ability to deal with both numerical and categorical data (Charbuty & Abdulazeez, 2021). Also, they are very simple to analyze, represent and comprehend that is why they are applied in decision making. This tutorial will demonstrate a detailed explanation of how to develop a Decision Tree model and its functionality and uses.

* **Introduction:** An introduction to the DT classifier and choice of data set.
* **Data Pre-processing**: Make data clean and eliminate any possible errors and wrong entries on the data.
* **Model Training:** Implementations of the DT model on the dataset
* **Model Evaluation:** Compare the models according to the several evaluation measures.
* **Hyperparameter Tuning:** Improving the generality and performance of the basic model via fine tuning of DT model was also done.

# Introduction

A Decision Tree is one of the supervised learning algorithms that partition the data into subsets according to the values of predictor variables to generate relatively pure subsets. It is a fairly basic model which is very commonly used in both classification and regression problems. A decision node in the tree corresponds to a feature while a branch from the decision node encapsitates a decision rule based on that feature (Dabiri et al., 2022).

In this tutorial, we will only consider Decision Trees for binary classification problem. The bank Loan Dataset from Kaggle is used for the analysis in this task (Amin, 2023). The objective is to classify a customer as likely or not likely to be approved for a personal loan given the following attributes: age, income, credit card average score, and so on. This is a typical binary classification task where the target variable (Personal Loan) has two possible outcomes: 1 = Loan granted and 0 = No loan.

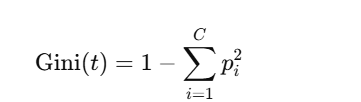
In binary classification (the process of predicting output as Yes or No such as “Loan applied” or “Not Loan applied”), Decision Trees splits the data at each node in a way that will give the maximum possible information gain or minimum impurity by trying to make each branch as pure as possible.

## DT model Splitting principle

Two commonly used splitting criteria in DT classifier are Gini Impurity and Information Gain (Ma et al., 2024):

### Gini Impurity

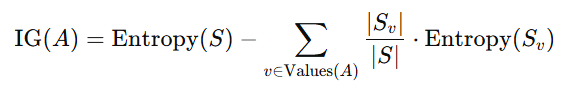
It is defined as how often a randomly picked element from the dataset could be misclassified if it was labelled by the label distribution in that node. The formula is:



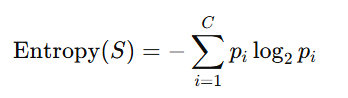
* ‘t’ is the dataset
* ‘C’ shows the number of classes
* ‘Pi’ is the probability of the node 𝑡 being in class 𝑖
* Gini impurity is between 0 (pure) and 0.5 (impure)

### Information Gain

It is a measure of how much our entropy decreases when we split our dataset according to the given feature. To select the feature that best reduces entropy (or equally best splits the data into as homogeneous a result), Information Gain is used. It is calculated as:



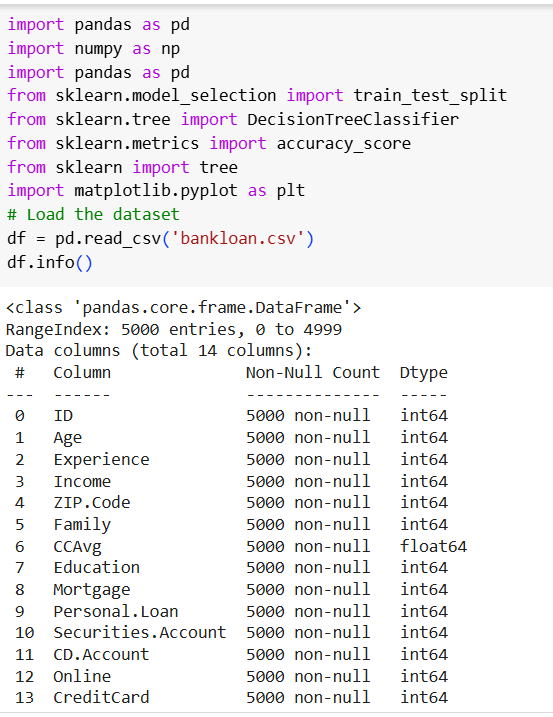
* Entropy (S) is for the dataset S which is calculated as below in which pi is the probability of the class ‘i’ in dataset S:



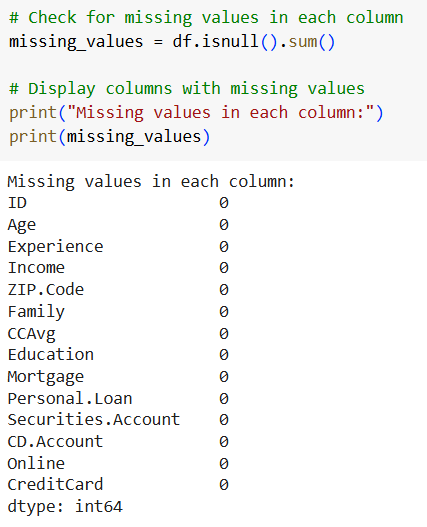
* ‘Sv’is the subset of the data where v is the values of feature A

# Preprocessing & Dataset Selection

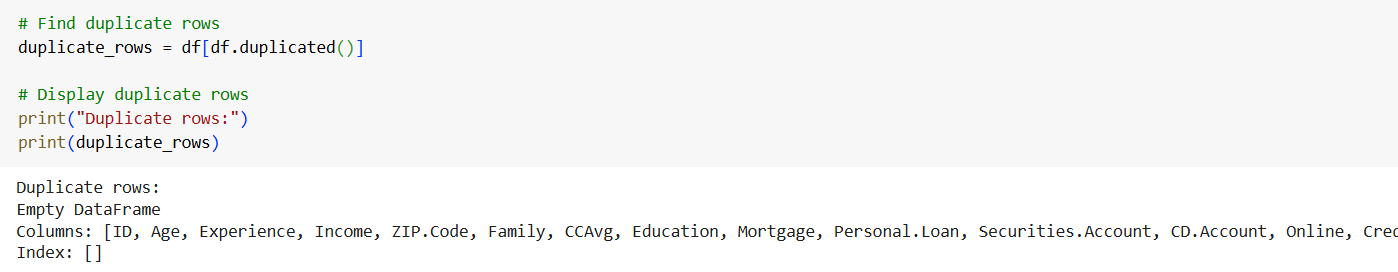
The DT model is applied to a bank loan dataset, where the target variable is whether a customer has been granted a personal loan (binary classification: 1 indicates loan granted and 0 indicates no loan). There are 14 features in the dataset which includes age, family size, credit card average score, income etc. The dataset is used to predict the target variable. The python language is applied for data preprocessing and further analysis. The details of the dataset is shown below.



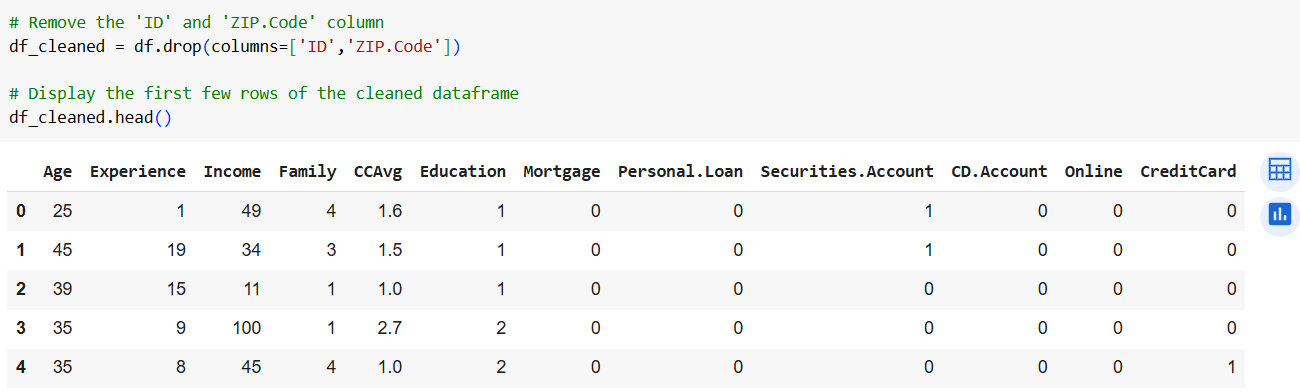
Then data preprocessing is performed and finds the missing values from the dataset. There are no missing values which can be seen below.



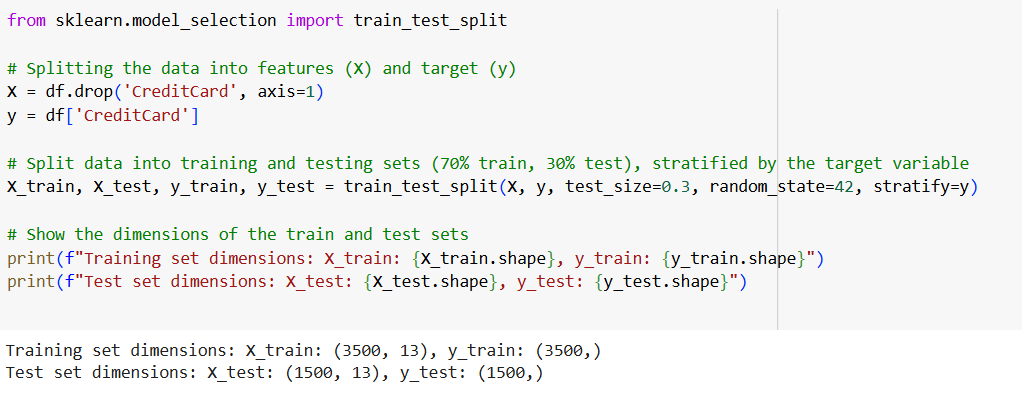
The duplication of the dataset is also determined and finds that there no duplication in the dataset.



The ‘ID’ and ‘Zip.Code’ column is extra and no needs for the further analysis so we discarded it.

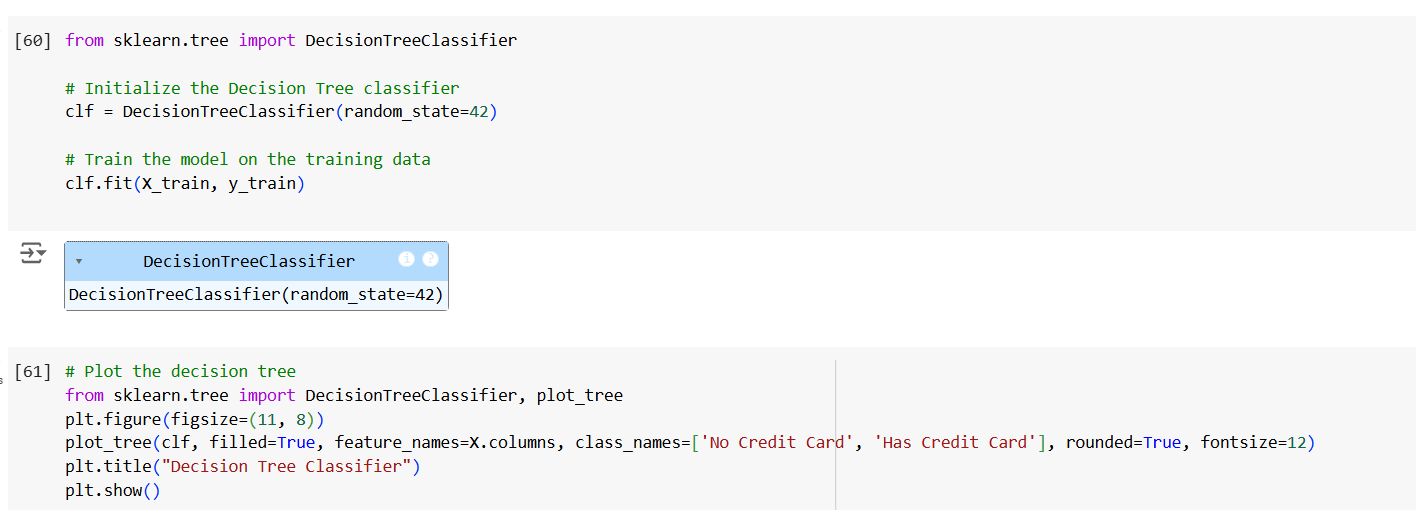


After that the dataset split for training and testing the DT classifier and 70% dataset is used for raining and 30% for testing. The stratify split is used for the better reults.

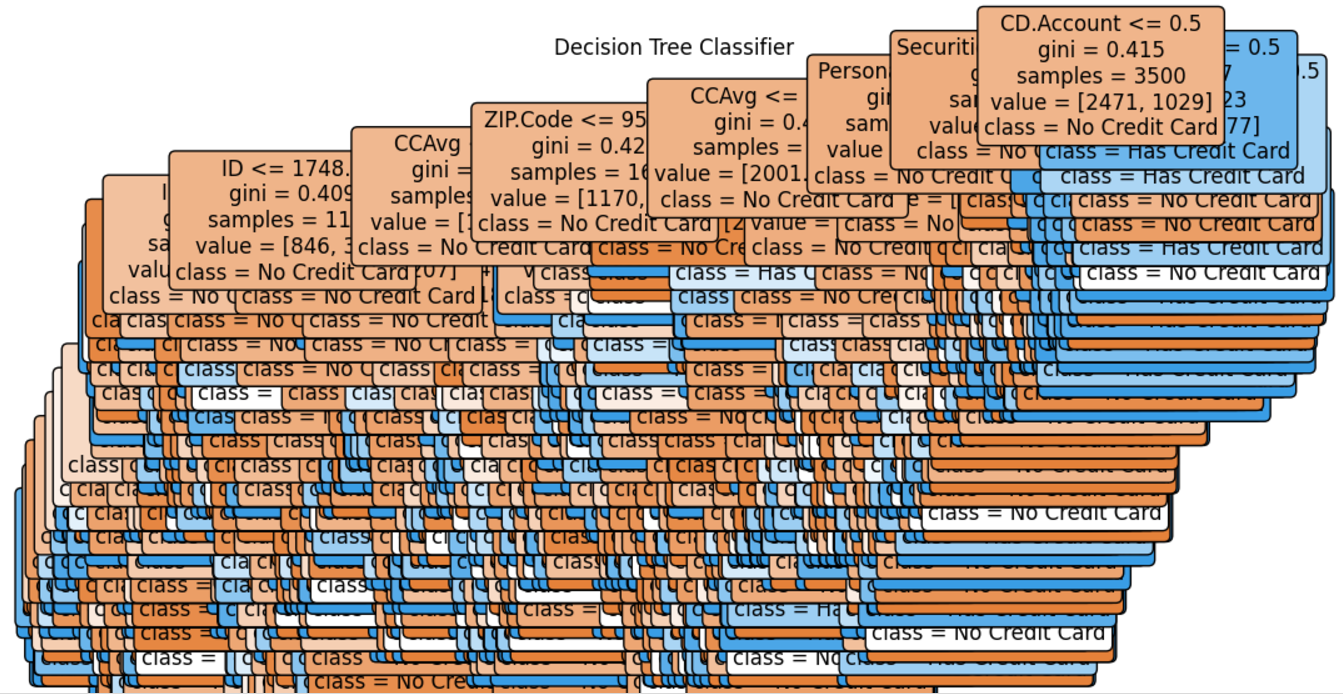


# DT Model Training

Having prepared the data and preprocessed it, we proceed to the building of the Decision Tree model using the default hyperparameter (Sarang, 2023). We will also use all other hyperparameters of the on default settings.

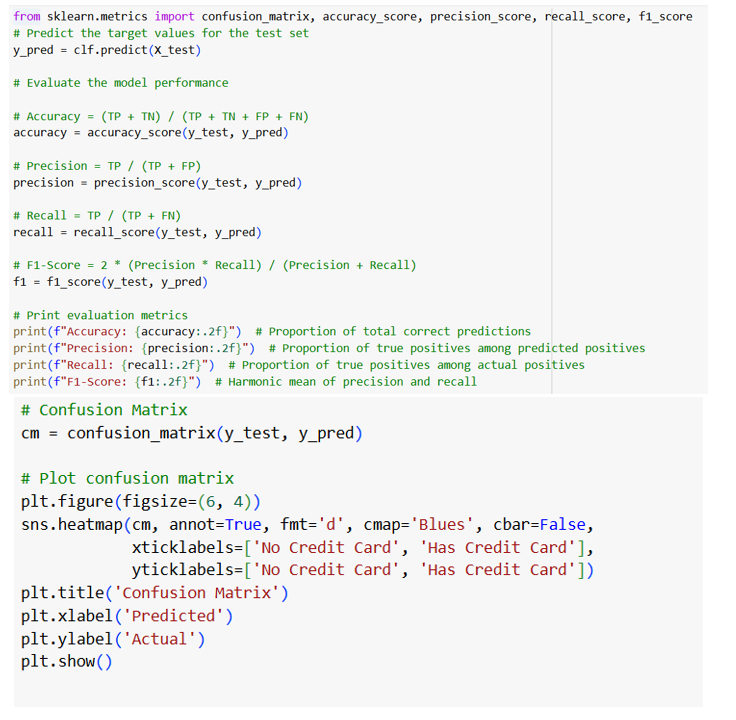


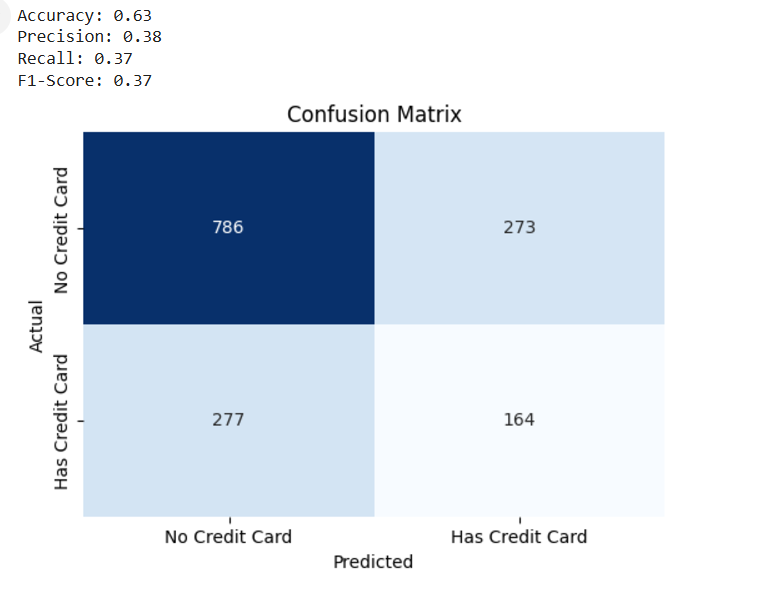
The plot for the trained DT classifier is also show below. The decision tree allows the visual representation of how the data is being divided according to the features to determine if a customer has a credit card. Gini impurity (by default) is applied at each node to decide on which attribute provides the most satisfactory partitioning of the data into class members. Outcome see that the DT model is overfitted much and all the nodes are not good which can be seen here for the reference. To tackle this issue we need to tune to model on the best parameters to get the best results.



# Evaluation of the Model

In this work, after training the model, we assess the model’s performance using metrics including accuracy, precision, recall and F1- score. Accuracy is the measure of how many times the model is right in general while precision is how many of the predicted positives are actually positive (Powers, 2020). Recall shows the percentage of correct positive cases that are correctly predicted by a model and F1 score is the average of precision and recall and is very helpful in the case of imbalance data. These metrics combined present a complete evaluation of the model’s efficiency.





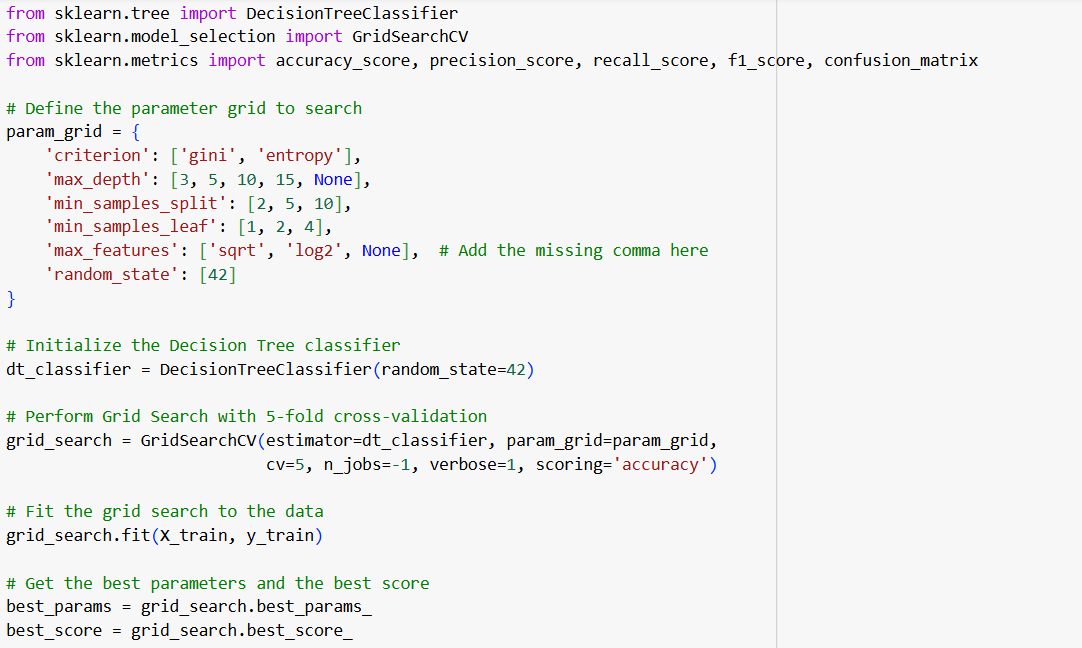
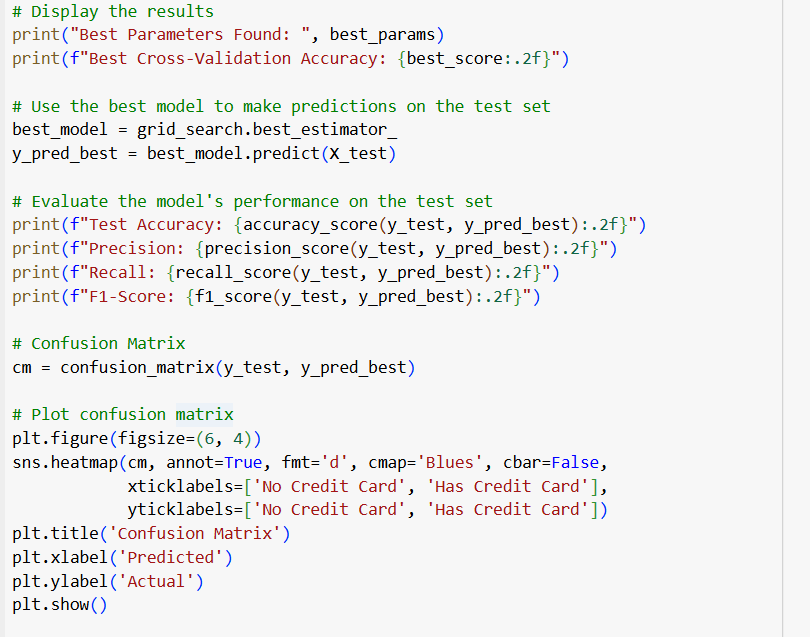
A look at the confusion matrix tells that the model makes “786 true negatives (TN)” predictions, rightly predicting that they did not possess a credit card. It also 277 false positive (FP)'; it classifies customers without a credit card as having one. Furthermore, in "273 false negatives (FN), a customer who does have a credit card is incorrectly predicted not to have one". Last, the model accurately classifies 164 true positives (TP)', correctly anticipating that customers have a credit card.

The model’s performance metrics actually show a ‘0.63 accuracy’, but this is meaningless (except that it is extremely poor) because the dataset is extremely skewed. Since there are many true negatives (1,013), the accuracy looks good (as overall), but the model’s performance of identifying the credit card holders (the minority class) shows very poor performance. The 0.38 "precision" suggests that 38% of the time when the model says a customer has a credit card, it actually does, which is pretty strong for the positive class. Recall is very low: 0.37 meaning this model get’s it right of the time it misses actual credit card holders. In doing so, I illustrate a flaw of the model not being able to accurately perceive the minority class.

F1 score of 0.37 is the tradeoff between Precision and Recall because it combines both. Because this F1 score is so low, we can tell that the model is having trouble finding a balance between correctly flagging credit card holders and minimizing false positives. While the precision of the model is bad, it has a poor recall, which results in missing all the customers who do have a credit card, which is a serious issue when we want to find those who do have a credit card. The model could also be helped by techniques such as hyperparameter tuning, class balancing, such as either oversampling the minority class or reweighting of the class weights to be more sensitive on credit card holders in order to increase recall and improve performance.

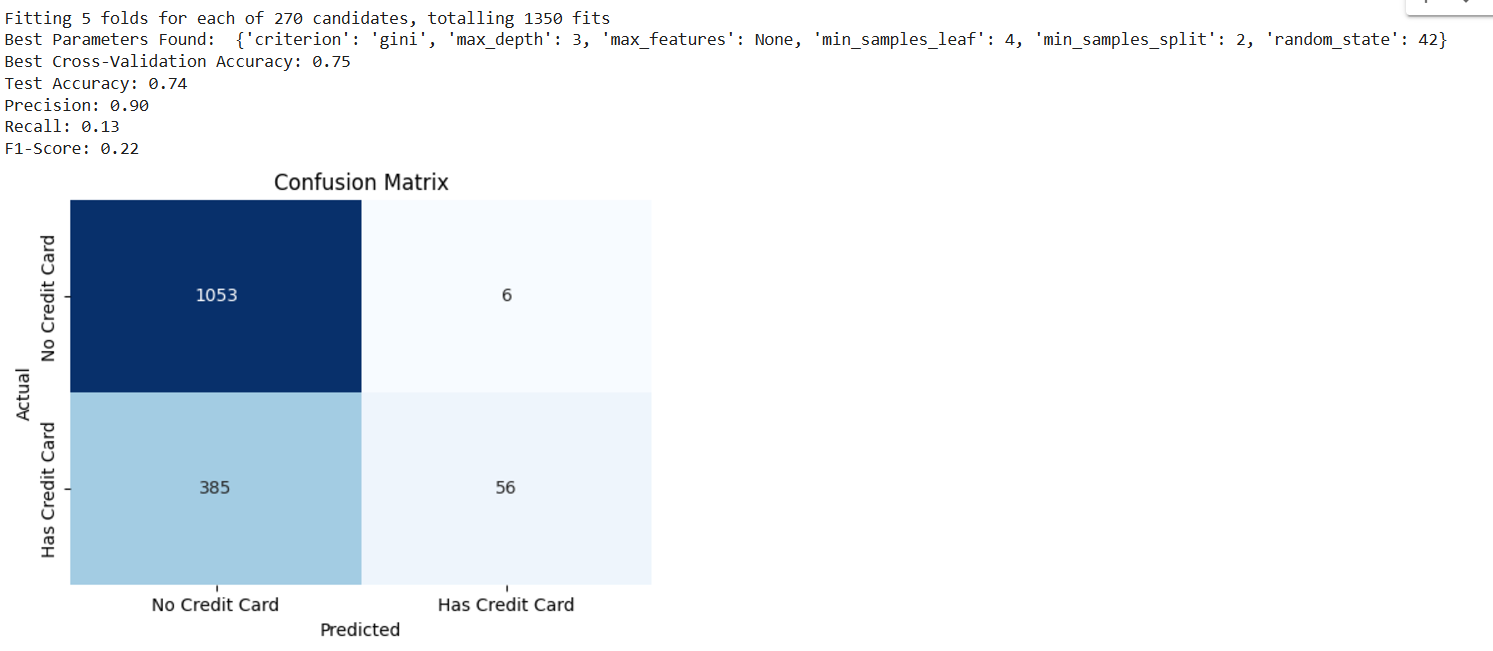
# Model Hyperparameter tuning

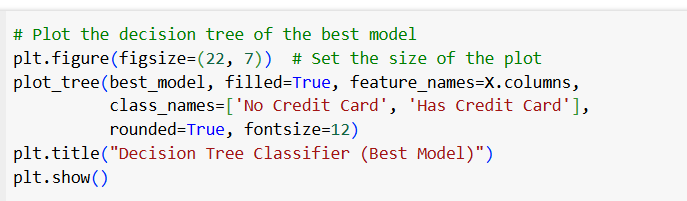
An important step when optimizing a ML model for performance is called fine tuning or optimization. In a DT classifier, this can be done including tuning of the hyperparameters — splitting criterion (Gini Impurity or Information Gain), maximum tree depth, minimum samples required for a split, minimum samples in leaf nodes, etc (Mantovani et al., 2018). These parameters determine how complex the tree itself will be, and how complex the tree patterns that will be captured (i.e. shall not overfit if possible). Instead, 'GridSearchCV' takes a score taken from performing cross-validation on multiple combinations of hyperparameters, and selects one of them based on some kind of 'performance metric' we choose (e.g. accuracy) (Shekar & Dagnew, 2019). Apart from enhancing an improved predictive ability for a model, this process improves the robustness, resulting in more reliable models for real world applications.

Decision Tree classifier results indicate that model has high precision (0.90), but very low recall (0.13)." Although the model is quite good at predicting correctly, when it does predict positively (high precision), the model does not identify a majority of the actual positive cases (low recall). Most notably, the model correctly identifies only 13% of customers who are supposed to receive loans, meaning that many customers that should have been predicted as qualified for given loans were missed. As a result, the false negative rate is high, which can be bothersome, especially when it is required at all times to account for as many possible loan applicants as possible.

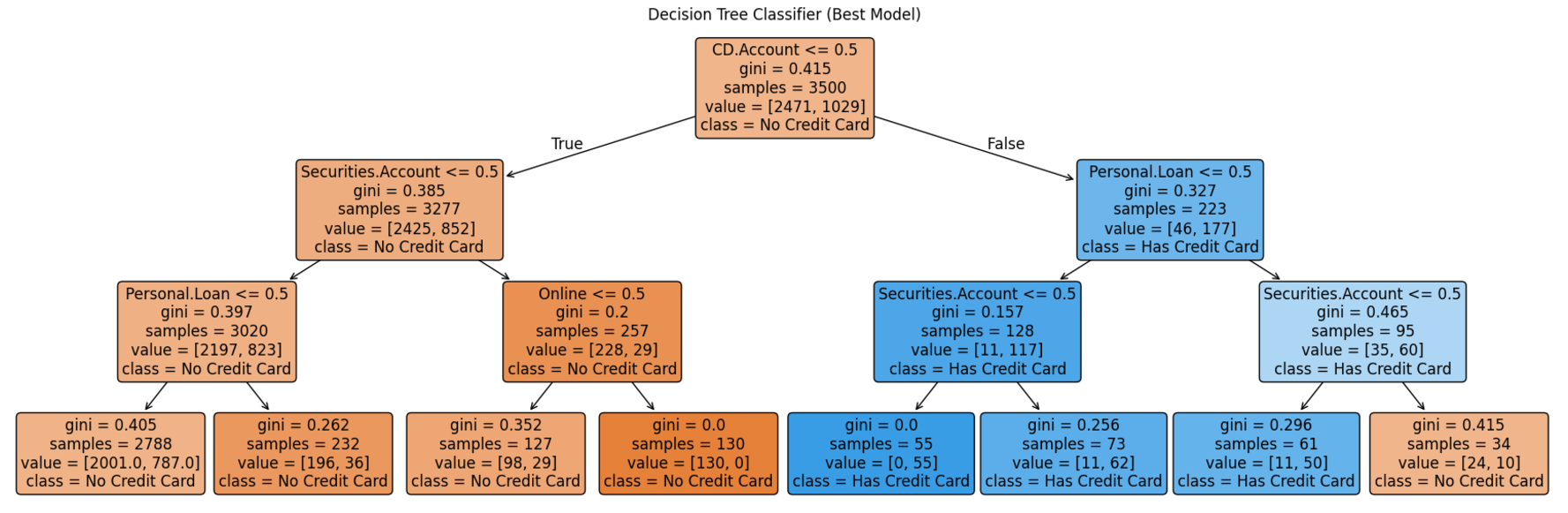
Furthermore, its F1 score (0.22) is also very low, as we are not accounting for the tradeoff between precision (0.12) and recall (0.48), which may be resulting in entire poor performance on the data set. The test accuracy of around 74% alone is best as compared to the previous model. The low recall also indicates that the model is biased toward the majority negative class (or not eligible for loans) and this model needs further tuning to be able to detect the positive cases. For this case, we could think to balance the dataset through oversampling the minority class, setting a higher decision threshold for greater recall, or exploring other models and hyperparameters for higher balance and better overall model results."



This is the confusion matrix which tells you that the model really scores on accurately predicting "No Credit Card" for most of these customers because there are a huge number of True Negatives (1,053). The number of True Positives (56) also shows, though, that the model has a hard time predicting "Has Credit Card." False Positives (385), meaning a lot of customers that don’t have a credit card were incorrectly predicted to have one, and False Negatives (6), meaning we missed a relatively small number of customers that do have credit cards. These issues are further corroborated by evaluation metrics, in that the model is 74% accurate, it correctly predicts 74% of the cases. 

The fine tunned model plotted here and shown here. Based on CD accounts, securities accounts, personal loans and/or online accounts, the decision tree model projects whether a person has a credit card. The most influential factor is the presence of a CD account at the root of the tree. 2,471 of the 3,500 samples are predicted to belong to the "No Credit Card" class if a person does not have a CD account (CD Account ≤ 0.5). If they don’t have a CD account (CD Account > 0.5), however, they are also more likely to have a card. This first split demonstrates how important CD account ownership is to predicting whether someone owns a credit card.

It then further branches into more detailed splits on other financial products. The probability of owning a credit card increases for those who do not have a securities account and personal loan, and slightly decreases for those who do not have a CD account without a securities account and personal loan, but do have an online account. In contrast, those with a CD account are more likely to have hold a credit card, but only if they also hold a personal loan or securities account. The Gini index at each node measures the purity of the classification and lower index value means more confident groups and more homogeneous. In this context, this model shows how having different financial products affects the probability of owning credit card.



# Conclusion and Findings

The Bank Loan dataset was applied to predict the customer probability of being granted a personal loan in this study using the Decision Tree (DT) classifier. Initially, the model was trained with the default settings which lead to overfitting, captured from the view that the tree is highly complex and the number of misclassifications are high. However, the model was able People to get reasonable accuracy on overall predictions but did not work well with minority class (people having the credit card, as demonstrated by low values of recall and F1 score).

Model evaluation through evaluation of model’s performance was observed to be about 74% and the same had been skewed by the imbalance of dataset with a significant number of True Negatives (representing non-deposit of credit cards). While the model had a relatively high precision (0.90) it had very low recall (0.13), meaning it was missing many actual credit card holders. Its F1 score was low as well (0.22) due to the model’s inability to capture the positive class. It also shows that the Decision Tree classifier, as used, was biased towards the majority class.

Futhermore, upon further tuning the Decision Tree model became more reliable and showed the following: financial features, such as the presence of a CD account, securities account, and personal loan were influential in determining if the customer would have a crrdit card. We also fine tuned the model using customers without CD account to be predicted as 'No Credit Card' and with CD account to be predicted as 'Yes Credit Card' refining the model to understand what financial products are important in 'Yes Credit Card' and 'No Credit Card' in loan approval prediction.

This study concludes that Decision Trees are a good choice for classification problems, but they are very sensitive to class imbalance and model overfit. They (proper evaluation metrics: precision recall, f1 score) are also useful to add if you want to design a not so specific model that can handle to give more accurate predictions for real world applications and that is where hyperparameter tuning comes in.

# References

Amin, V. (2023). *Bank Loan Approval - LR, DT, RF and AUC: Classification Problem*. <https://www.kaggle.com/datasets/vikramamin/bank-loan-approval-lr-dt-rf-and-auc>

Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, *2*(01), 20-28.

Dabiri, H., Farhangi, V., Moradi, M. J., Zadehmohamad, M., & Karakouzian, M. (2022). Applications of decision tree and random forest as tree-based machine learning techniques for analyzing the ultimate strain of spliced and non-spliced reinforcement bars. *Applied Sciences*, *12*(10), 4851.

Ma, Y., Zhang, H., Cai, Y., & Yang, H. (2024). Decision tree for locally private estimation with public data. *Advances in neural information processing systems*, *36*.

Mantovani, R. G., Horváth, T., Cerri, R., Junior, S. B., Vanschoren, J., & de Carvalho, A. d. L. (2018). An empirical study on hyperparameter tuning of decision trees. *arXiv preprint arXiv:1812.02207*.

Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

Sarang, P. (2023). Decision Tree: A Supervised Learning Algorithm for Classification. In *Thinking Data Science: A Data Science Practitioner’s Guide* (pp. 75-96). Springer.

Shekar, B., & Dagnew, G. (2019). Grid search-based hyperparameter tuning and classification of microarray cancer data. 2019 second international conference on advanced computational and communication paradigms (ICACCP),