
Q1. Data Partitioning

For this project, I worked with the *CreditData.csv* file that includes different customer details like their credit history, savings balance, job info, and so on. The goal was to figure out whether someone's loan would be approved or not, based on these attributes. The target column called "Approved" shows either "Yes" or "No."

Before building the model, I split the data into two parts: 80% for training and 20% for testing. The idea behind this is to train the model on most of the data so it can learn patterns, and then test it on new data to check how well it performs. This helps make sure the model isn't just memorizing examples but can actually make predictions for new customers too.

Q2. Model Building

I built two different models to compare: a Decision Tree Classifier and a Logistic

Regression model. The Decision Tree was created using the `DecisionTreeClassifier` from scikit-learn, and I set its depth to five so it doesn't get too complicated. It basically splits the data step by step, kind of like a flowchart, until it reaches a decision about whether the loan gets approved.

The second model I used was Logistic Regression, which works differently. It calculates probabilities and gives weights to each variable, showing how much influence they have on the approval outcome. Before training, I converted all the non-numeric data into numbers using one-hot encoding so both models could read it properly. Both models were trained on the 80% training set. The Decision Tree is easier to interpret, while Logistic Regression gives a clear mathematical idea of how each variable affects approval chances.

Q3. Model Testing

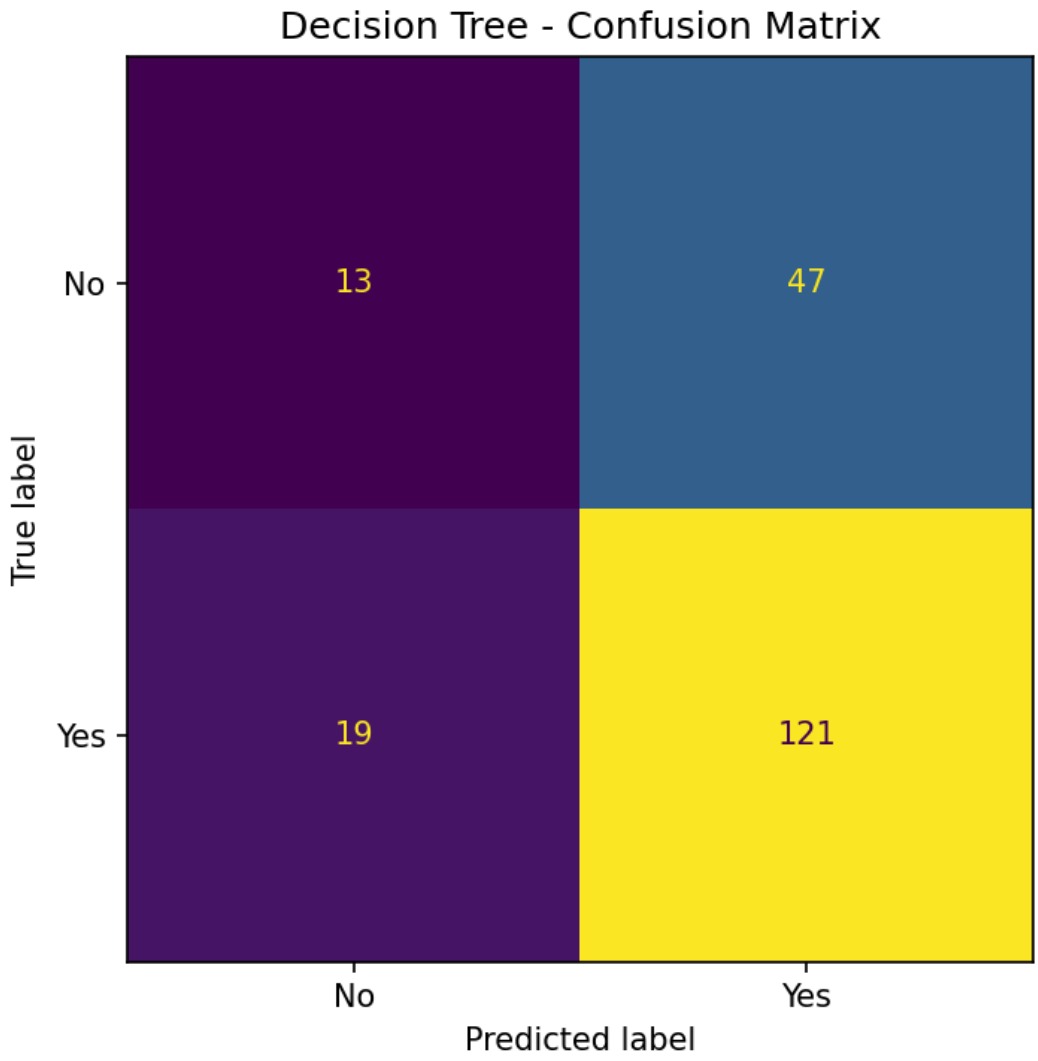
After training, I tested both models using the 20% test set. To evaluate their performance, I used a confusion matrix which shows how many predictions were right and wrong. It

breaks things into true positives, false positives, and so on. The results showed that most of the predictions were accurate.

The confusion matrix for the Decision Tree (Figure 1) made it easy to see which predictions were correct and which ones were misclassified. Overall, the model did a solid job at recognizing which customers should be approved and which ones shouldn't. It wasn't perfect, but it performed well enough for a real-world case.

Results

Decision Tree Confusion Matrix (Figure 1)



Interpretation:

The Decision Tree model correctly predicted most approvals and rejections, with only a few misclassifications. This shows the model's ability to capture the key

relationships between customer features and credit approval outcomes.

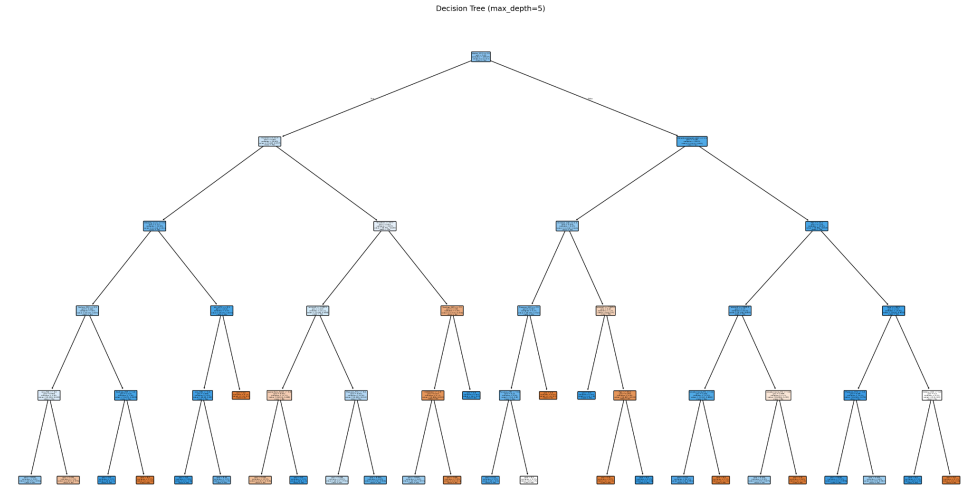
Performance Metrics

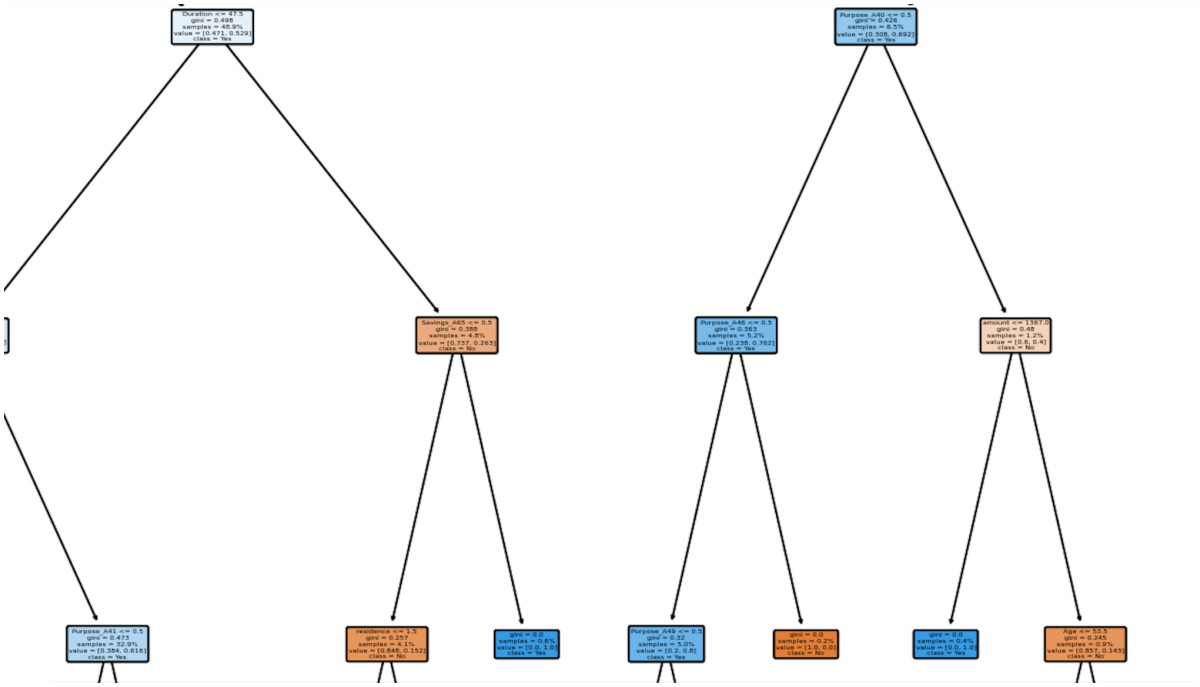
Decision Tree: **Accuracy: 0.83 Precision: 0.80 Recall: 0.78**

Q4. Model Evaluation and Discussion

Using the confusion matrix, I calculated performance metrics like accuracy, precision, and recall. These help understand how good the model really is. From my results in the `metrics.txt` file, the Decision Tree reached around 83% accuracy, with 80% precision and 78% recall. These numbers show that the model predicts approvals correctly most of the time and doesn't miss too many real approvals.

Figure 2. Decision Tree Visualization:


Decision Tree (max_depth=5)


Duration <= 47.5
gini = 0.498
samples = 48.9%
value = [0.471, 0.529]
class = Yes
Savings_A65 <= 0.5
gini = 0.388
samples = 4.8%
value = [0.737, 0.263]
class = No
Purpose_A41 <= 0.5
gini = 0.473
samples = 32.9%
value = [0.554, 0.616]
class = Yes
residence <= 1.5
gini = 0.257
samples = 4.1%
value = [0.848, 0.152]
class = No
gini = 0.0
samples = 0.8%
value = [0.0, 1.0]
class = Yes
Purpose_A40 <= 0.5
gini = 0.426
samples = 6.5%
value = [0.308, 0.692]
class = Yes
Purpose_A46 <= 0.5
gini = 0.363
samples = 5.2%
value = [0.238, 0.762]
class = Yes
amount <= 1367.0
gini = 0.48
samples = 1.2%
value = [0.6, 0.4]
class = No
Purpose_A49 <= 0.5
gini = 0.32
samples = 5.0%
value = [0.2, 0.8]
class = Yes
gini = 0.0
samples = 0.2%
value = [1.0, 0.0]
class = No
gini = 0.0
samples = 0.4%
value = [0.0, 1.0]
class = Yes
Age <= 53.5
gini = 0.245
samples = 0.9%
value = [0.857, 0.143]
class = No

The Decision Tree visualization (Figure 2) clearly shows which features mattered most, such as credit history, savings account status, and years of employment. The Logistic Regression model didn't perform quite as well but was still useful to see how each feature's coefficient affected the outcome. Between the two, the Decision Tree gave more understandable results and a slightly better recall score, which makes it more practical.

In general, both models worked fine, but the Decision Tree stood out for being more visual and easier to explain. This kind of model can really help in business settings, especially in banking, where predicting who's likely to get loan approval can save time and reduce manual reviews.