CMPE 480 HOMEWORK 4 REPORT

2020400051 - Bilge Kaan Güneyli

Information about Dataset

In the project I used the Loan Approval Classification Dataset from Kaggle. The dataset includes 13 features and 1 target variable. The target is a binary field, 0 stands for rejection of the loan and 1 stands for approval. Features include information about the person who made the loan offer such as age, gender, income, home ownership, credit score. Some of these fields are continuous fields however the model I implemented can handle only categorical data. Therefore, I had to encode these fields to 5 classes using the *qcut* function numpy in order to be able to run the model on the data.

Usage of 5-fold Cross-Validation

In order to apply hyperparameter tuning with cross-validation, I had to add parameters to the class. I tried to choose my parameters in a way that will reduce fallouts and added *max_depth* (limits the depth of the tree) and *min_samples_split* (minimum number of samples for a node to be allowed to get split). These parameters have default values which makes them uninfluential on the model. In implementation, I tried to keep my model as similar to the decision tree classifier of scikit-learn as possible so I named everything with the same names as the library.

I implemented cross-validation as a method under the *DecisionTree* class. The method accepts only *max_depth* and *min_samples_split* as parameters, else it raises error. If one or two of these parameters are not specified by the user it assigns them with the default value.

Then the method operates as follows:

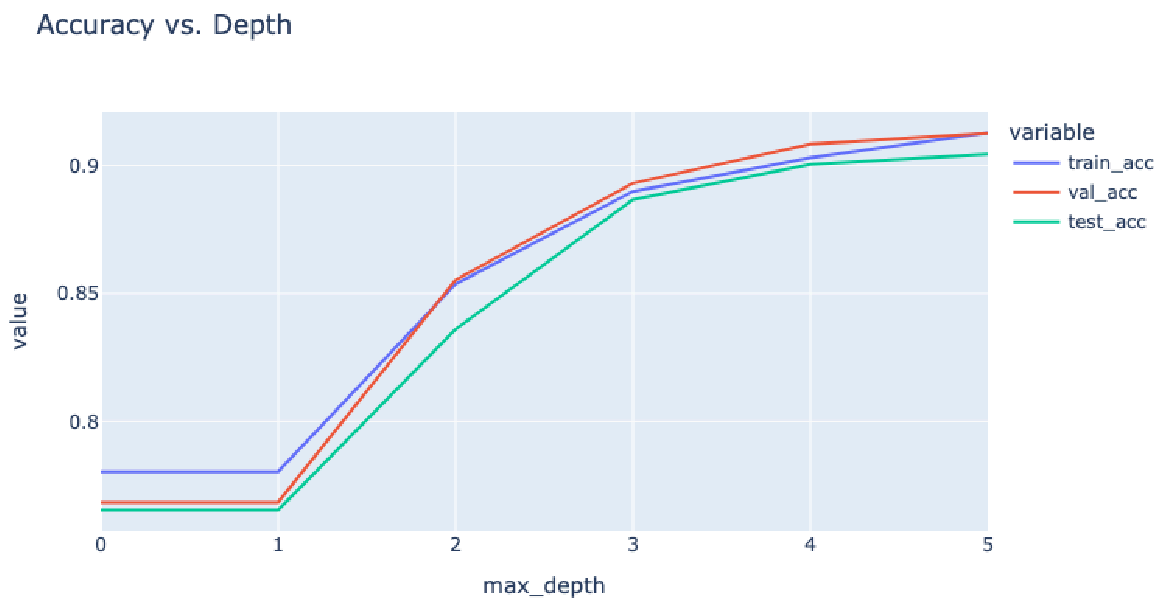
- One combination of *max_depth* and *min_samples_split* is chosen
- The data is split to 5 pieces of equal size.
- 1 of the pieces is named as the validation set and union of other 4 pieces is named the training set.
- Accuracy on the validation set is calculated.
- Step 3 and 4 are applied to other 4 pieces too.
- Average accuracy is calculated. If it is higher than the previous best *max_depth* and *min_samples_split* kept as the best parameters.
- All the steps above are applied to all parameter combinations.
- The method returns the best parameters

With the default values of the model, the accuracy on the test data was approximately 0.869. After applying hyperparameter tuning with cross-validation I reached the accuracy of 0.905 which is a valuable increase in these levels of accuracies and also I had a simpler model with maximum depth of 5 and minimum 60 samples for nodes which are split.

Note: Since the algorithm doesn't work fast with cross-validation, I firstly created a parameter grid with large differences between values then narrowed it down depending on the output. Therefore, in the source code you may see a narrow range of values.

Error Plots

In the optimum model, maximum depth is 5. Therefore, to see the improvement I created this plot using increasing values for *max_depth* until reaching 5



Best Decision Tree

```
Split on feature: previous_loan_defaults_on_file
   Value = No:
   Split on feature: loan_percent_income
       Value = (-0.001, 0.06]:
       Split on feature: loan_int_rate
           Value = (5.419, 7.88]:
           Split on feature: person_income
               Value = (7999.999, 43081.8]:
               Leaf: predicts 0
               Value = (43081.8, 59456.0]:
               Split on feature: loan_intent
                   Value = DEBTCONSOLIDATION:
                   Leaf: predicts 0
                   Value = EDUCATION:
                   Leaf: predicts 0
                   Value = HOMEIMPROVEMENT:
```

```
        Leaf: predicts 0
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 0
        Value = VENTURE:
        Leaf: predicts 0
    Value = (59456.0, 76436.8]:
    Split on feature: cb_person_cred_hist_length
        Value = (1.999, 3.0]:
        Leaf: predicts 0
        Value = (3.0, 4.0]:
        Leaf: predicts 0
        Value = (4.0, 6.0]:
        Leaf: predicts 0
        Value = (6.0, 9.0]:
        Leaf: predicts 0
        Value = (9.0, 30.0]:
        Leaf: predicts 0
    Value = (76436.8, 103841.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
        Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 0
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 0
        Value = VENTURE:
        Leaf: predicts 0
    Value = (103841.0, 7200766.0]:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Leaf: predicts 0
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Leaf: predicts 0
Value = (7.88, 10.58]:
Split on feature: loan_intent
    Value = DEBTCONSOLIDATION:
    Split on feature: person_income
```

```
        Value = (7999.999, 43081.8]:
        Leaf: predicts 0
        Value = (43081.8, 59456.0]:
        Leaf: predicts 0
        Value = (59456.0, 76436.8]:
        Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = EDUCATION:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 0
        Value = (43081.8, 59456.0]:
        Leaf: predicts 0
        Value = (59456.0, 76436.8]:
        Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = HOMEIMPROVEMENT:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 0
        Value = (59456.0, 76436.8]:
        Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = MEDICAL:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 0
        Value = (4400.0, 6600.0]:
        Leaf: predicts 0
        Value = (6600.0, 10000.0]:
        Leaf: predicts 0
        Value = (10000.0, 14520.2]:
        Leaf: predicts 0
        Value = (14520.2, 35000.0]:
```

```
            Leaf: predicts 1
        Value = PERSONAL:
        Split on feature: loan_amnt
            Value = (499.999, 4400.0]:
            Leaf: predicts 0
            Value = (4400.0, 6600.0]:
            Leaf: predicts 0
            Value = (6600.0, 10000.0]:
            Leaf: predicts 0
            Value = (10000.0, 14520.2]:
            Leaf: predicts 0
            Value = (14520.2, 35000.0]:
            Leaf: predicts 1
        Value = VENTURE:
        Leaf: predicts 0
    Value = (10.58, 11.49]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 1
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
        Value = EDUCATION:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Split on feature: person_age
            Value = (19.999, 23.0]:
            Leaf: predicts 1
```

```
            Value = (23.0, 25.0]:
            Leaf: predicts 0
            Value = (25.0, 27.0]:
            Leaf: predicts 0
            Value = (27.0, 31.0]:
            Leaf: predicts 0
            Value = (31.0, 144.0]:
            Leaf: predicts 0
        Value = MEDICAL:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 0
            Value = (43081.8, 59456.0]:
            Leaf: predicts 0
            Value = (59456.0, 76436.8]:
            Leaf: predicts 1
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
        Value = PERSONAL:
        Split on feature: person_education
            Value = Associate:
            Leaf: predicts 0
            Value = Bachelor:
            Leaf: predicts 0
            Value = Doctorate:
            Leaf: predicts 1
            Value = High School:
            Leaf: predicts 0
            Value = Master:
            Leaf: predicts 0
        Value = VENTURE:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
    Value = (11.49, 13.49]:
```

```
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 1
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
            Value = EDUCATION:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 0
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Split on feature: person_income
                Value = (7999.999, 43081.8]:
                Leaf: predicts 1
                Value = (43081.8, 59456.0]:
                Leaf: predicts 1
                Value = (59456.0, 76436.8]:
                Leaf: predicts 0
                Value = (76436.8, 103841.0]:
                Leaf: predicts 0
                Value = (103841.0, 7200766.0]:
                Leaf: predicts 0
            Value = MEDICAL:
            Split on feature: loan_amnt
                Value = (499.999, 4400.0]:
                Leaf: predicts 0
                Value = (4400.0, 6600.0]:
                Leaf: predicts 0
                Value = (6600.0, 10000.0]:
                Leaf: predicts 0
```

```
            Value = (10000.0, 14520.2]:
            Leaf: predicts 0
            Value = (14520.2, 35000.0]:
            Leaf: predicts 1
        Value = PERSONAL:
        Split on feature: person_age
            Value = (19.999, 23.0]:
            Leaf: predicts 0
            Value = (23.0, 25.0]:
            Leaf: predicts 0
            Value = (25.0, 27.0]:
            Leaf: predicts 0
            Value = (27.0, 31.0]:
            Leaf: predicts 0
            Value = (31.0, 144.0]:
            Leaf: predicts 0
        Value = VENTURE:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
    Value = (13.49, 20.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 1
            Value = (591.0, 626.0]:
            Leaf: predicts 1
            Value = (626.0, 652.0]:
            Leaf: predicts 1
            Value = (652.0, 676.0]:
            Leaf: predicts 1
            Value = (676.0, 850.0]:
            Leaf: predicts 1
        Value = EDUCATION:
        Split on feature: person_home_ownership
            Value = MORTGAGE:
```

```
        Leaf: predicts 0
        Value = OTHER:
        Leaf: predicts 0
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Leaf: predicts 1
    Value = HOMEIMPROVEMENT:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 1
        Value = (59456.0, 76436.8]:
        Leaf: predicts 1
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = MEDICAL:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 1
        Value = (59456.0, 76436.8]:
        Leaf: predicts 1
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 1
    Value = PERSONAL:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Leaf: predicts 0
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Leaf: predicts 1
    Value = VENTURE:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 1
```

```
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
    Value = (0.06, 0.1]:
    Split on feature: loan_int_rate
        Value = (5.419, 7.88]:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 1
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 1
                Value = VENTURE:
                Leaf: predicts 0
            Value = (43081.8, 59456.0]:
            Split on feature: person_age
                Value = (19.999, 23.0]:
                Leaf: predicts 0
                Value = (23.0, 25.0]:
                Leaf: predicts 0
                Value = (25.0, 27.0]:
                Leaf: predicts 0
                Value = (27.0, 31.0]:
                Leaf: predicts 0
                Value = (31.0, 144.0]:
                Leaf: predicts 0
            Value = (59456.0, 76436.8]:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 0
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
```

```
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 0
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 0
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
        Value = (7.88, 10.58]:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 1
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
```

```
        Value = (43081.8, 59456.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (59456.0, 76436.8]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Split on feature: person_home_ownership
            Value = MORTGAGE:
            Leaf: predicts 0
            Value = OWN:
            Leaf: predicts 0
            Value = RENT:
            Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
```

```
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
    Value = (10.58, 11.49]:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = OTHER:
        Leaf: predicts 0
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Split on feature: person_age
            Value = (19.999, 23.0]:
            Leaf: predicts 0
            Value = (23.0, 25.0]:
            Leaf: predicts 0
            Value = (25.0, 27.0]:
            Leaf: predicts 0
            Value = (27.0, 31.0]:
            Leaf: predicts 0
            Value = (31.0, 144.0]:
            Leaf: predicts 0
    Value = (11.49, 13.49]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 1
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
```

```
        Value = (652.0, 676.0]:
        Leaf: predicts 0
        Value = (676.0, 850.0]:
        Leaf: predicts 0
    Value = EDUCATION:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 0
        Value = (43081.8, 59456.0]:
        Leaf: predicts 0
        Value = (59456.0, 76436.8]:
        Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = HOMEIMPROVEMENT:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 1
        Value = (59456.0, 76436.8]:
        Leaf: predicts 0
        Value = (76436.8, 103841.0]:
        Leaf: predicts 0
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 0
    Value = MEDICAL:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Leaf: predicts 0
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Leaf: predicts 0
    Value = PERSONAL:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 0
        Value = (4400.0, 6600.0]:
        Leaf: predicts 0
        Value = (6600.0, 10000.0]:
        Leaf: predicts 0
        Value = (10000.0, 14520.2]:
```

```
            Leaf: predicts 0
            Value = (14520.2, 35000.0]:
            Leaf: predicts 0
        Value = VENTURE:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 0
            Value = (43081.8, 59456.0]:
            Leaf: predicts 0
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
    Value = (13.49, 20.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Split on feature: cb_person_cred_hist_length
            Value = (1.999, 3.0]:
            Leaf: predicts 1
            Value = (3.0, 4.0]:
            Leaf: predicts 1
            Value = (4.0, 6.0]:
            Leaf: predicts 1
            Value = (6.0, 9.0]:
            Leaf: predicts 1
            Value = (9.0, 30.0]:
            Leaf: predicts 1
        Value = EDUCATION:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 0
            Value = (43081.8, 59456.0]:
            Leaf: predicts 1
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 1
```

```
            Value = (43081.8, 59456.0]:
            Leaf: predicts 1
            Value = (59456.0, 76436.8]:
            Leaf: predicts 1
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
        Value = MEDICAL:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 1
            Value = (591.0, 626.0]:
            Leaf: predicts 1
            Value = (626.0, 652.0]:
            Leaf: predicts 1
            Value = (652.0, 676.0]:
            Leaf: predicts 1
            Value = (676.0, 850.0]:
            Leaf: predicts 1
        Value = PERSONAL:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 1
            Value = (43081.8, 59456.0]:
            Leaf: predicts 1
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 1
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
        Value = VENTURE:
        Split on feature: person_home_ownership
            Value = MORTGAGE:
            Leaf: predicts 0
            Value = OTHER:
            Leaf: predicts 0
            Value = OWN:
            Leaf: predicts 0
            Value = RENT:
            Leaf: predicts 1
Value = (0.1, 0.14]:
Split on feature: loan_int_rate
    Value = (5.419, 7.88]:
```

```
        Split on feature: loan_amnt
            Value = (499.999, 4400.0]:
            Leaf: predicts 0
            Value = (4400.0, 6600.0]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 1
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
            Value = (6600.0, 10000.0]:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 0
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
            Value = (10000.0, 14520.2]:
            Split on feature: person_home_ownership
                Value = MORTGAGE:
                Leaf: predicts 0
                Value = OWN:
                Leaf: predicts 0
                Value = RENT:
                Leaf: predicts 0
            Value = (14520.2, 35000.0]:
            Leaf: predicts 0
        Value = (7.88, 10.58]:
        Split on feature: loan_amnt
            Value = (499.999, 4400.0]:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 1
```

```
    Value = (591.0, 626.0]:
    Leaf: predicts 1
    Value = (626.0, 652.0]:
    Leaf: predicts 1
    Value = (652.0, 676.0]:
    Leaf: predicts 1
    Value = (676.0, 850.0]:
    Leaf: predicts 0
Value = (4400.0, 6600.0]:
Split on feature: loan_intent
    Value = DEBTCONSOLIDATION:
    Leaf: predicts 0
    Value = EDUCATION:
    Leaf: predicts 0
    Value = HOMEIMPROVEMENT:
    Leaf: predicts 1
    Value = MEDICAL:
    Leaf: predicts 0
    Value = PERSONAL:
    Leaf: predicts 0
    Value = VENTURE:
    Leaf: predicts 0
Value = (6600.0, 10000.0]:
Split on feature: loan_intent
    Value = DEBTCONSOLIDATION:
    Leaf: predicts 0
    Value = EDUCATION:
    Leaf: predicts 0
    Value = HOMEIMPROVEMENT:
    Leaf: predicts 0
    Value = MEDICAL:
    Leaf: predicts 0
    Value = PERSONAL:
    Leaf: predicts 0
    Value = VENTURE:
    Leaf: predicts 0
Value = (10000.0, 14520.2]:
Split on feature: person_gender
    Value = female:
    Leaf: predicts 0
    Value = male:
    Leaf: predicts 0
Value = (14520.2, 35000.0]:
Split on feature: person_age
    Value = (19.999, 23.0]:
```

```
            Leaf: predicts 0
            Value = (23.0, 25.0]:
            Leaf: predicts 0
            Value = (25.0, 27.0]:
            Leaf: predicts 0
            Value = (27.0, 31.0]:
            Leaf: predicts 0
            Value = (31.0, 144.0]:
            Leaf: predicts 0
    Value = (10.58, 11.49]:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Split on feature: cb_person_cred_hist_length
            Value = (1.999, 3.0]:
            Leaf: predicts 1
            Value = (3.0, 4.0]:
            Leaf: predicts 0
            Value = (4.0, 6.0]:
            Leaf: predicts 0
            Value = (6.0, 9.0]:
            Leaf: predicts 1
            Value = (9.0, 30.0]:
            Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (6600.0, 10000.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
```

```
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (10000.0, 14520.2]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (14520.2, 35000.0]:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
    Value = (11.49, 13.49]:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
```

```
        Leaf: predicts 1
        Value = VENTURE:
        Leaf: predicts 1
    Value = (4400.0, 6600.0]:
    Split on feature: cb_person_cred_hist_length
        Value = (1.999, 3.0]:
        Leaf: predicts 0
        Value = (3.0, 4.0]:
        Leaf: predicts 0
        Value = (4.0, 6.0]:
        Leaf: predicts 0
        Value = (6.0, 9.0]:
        Leaf: predicts 0
        Value = (9.0, 30.0]:
        Leaf: predicts 0
    Value = (6600.0, 10000.0]:
    Split on feature: credit_score
        Value = (389.999, 591.0]:
        Leaf: predicts 0
        Value = (591.0, 626.0]:
        Leaf: predicts 0
        Value = (626.0, 652.0]:
        Leaf: predicts 0
        Value = (652.0, 676.0]:
        Leaf: predicts 0
        Value = (676.0, 850.0]:
        Leaf: predicts 0
    Value = (10000.0, 14520.2]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
        Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 0
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 0
        Value = VENTURE:
        Leaf: predicts 0
    Value = (14520.2, 35000.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
```

```
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
    Value = (13.49, 20.0]:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 1
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = OTHER:
        Leaf: predicts 1
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 1
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 1
            Value = PERSONAL:
            Leaf: predicts 1
            Value = VENTURE:
            Leaf: predicts 1
Value = (0.14, 0.21]:
```

```
    Split on feature: loan_int_rate
        Value = (5.419, 7.88]:
        Split on feature: loan_amnt
            Value = (499.999, 4400.0]:
            Leaf: predicts 1
            Value = (4400.0, 6600.0]:
            Split on feature: credit_score
                Value = (389.999, 591.0]:
                Leaf: predicts 0
                Value = (591.0, 626.0]:
                Leaf: predicts 0
                Value = (626.0, 652.0]:
                Leaf: predicts 0
                Value = (652.0, 676.0]:
                Leaf: predicts 0
                Value = (676.0, 850.0]:
                Leaf: predicts 0
            Value = (6600.0, 10000.0]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 0
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
            Value = (10000.0, 14520.2]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 0
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 0
                Value = MEDICAL:
                Leaf: predicts 0
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
```

```
    Value = (14520.2, 35000.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
        Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 0
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 0
        Value = VENTURE:
        Leaf: predicts 0
Value = (7.88, 10.58]:
Split on feature: loan_amnt
    Value = (499.999, 4400.0]:
    Split on feature: cb_person_cred_hist_length
        Value = (1.999, 3.0]:
        Leaf: predicts 1
        Value = (3.0, 4.0]:
        Leaf: predicts 1
        Value = (4.0, 6.0]:
        Leaf: predicts 1
        Value = (6.0, 9.0]:
        Leaf: predicts 1
        Value = (9.0, 30.0]:
        Leaf: predicts 1
    Value = (4400.0, 6600.0]:
    Split on feature: credit_score
        Value = (389.999, 591.0]:
        Leaf: predicts 1
        Value = (591.0, 626.0]:
        Leaf: predicts 0
        Value = (626.0, 652.0]:
        Leaf: predicts 0
        Value = (652.0, 676.0]:
        Leaf: predicts 0
        Value = (676.0, 850.0]:
        Leaf: predicts 0
    Value = (6600.0, 10000.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
```

```
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (10000.0, 14520.2]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (14520.2, 35000.0]:
        Split on feature: person_income
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
    Value = (10.58, 11.49]:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 0
```

```
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (6600.0, 10000.0]:
        Split on feature: person_home_ownership
            Value = MORTGAGE:
            Leaf: predicts 0
            Value = OWN:
            Leaf: predicts 0
            Value = RENT:
            Leaf: predicts 0
        Value = (10000.0, 14520.2]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (14520.2, 35000.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 0
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
    Value = (11.49, 13.49]:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
```

```
        Leaf: predicts 1
        Value = EDUCATION:
        Leaf: predicts 1
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 1
        Value = MEDICAL:
        Leaf: predicts 1
        Value = PERSONAL:
        Leaf: predicts 1
        Value = VENTURE:
        Leaf: predicts 1
    Value = (4400.0, 6600.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 1
        Value = EDUCATION:
        Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 1
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 1
        Value = VENTURE:
        Leaf: predicts 0
    Value = (6600.0, 10000.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
        Leaf: predicts 0
        Value = HOMEIMPROVEMENT:
        Leaf: predicts 1
        Value = MEDICAL:
        Leaf: predicts 0
        Value = PERSONAL:
        Leaf: predicts 0
        Value = VENTURE:
        Leaf: predicts 0
    Value = (10000.0, 14520.2]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Leaf: predicts 0
        Value = EDUCATION:
        Leaf: predicts 0
```

```
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 0
            Value = MEDICAL:
            Leaf: predicts 0
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 0
        Value = (14520.2, 35000.0]:
        Split on feature: credit_score
            Value = (389.999, 591.0]:
            Leaf: predicts 0
            Value = (591.0, 626.0]:
            Leaf: predicts 0
            Value = (626.0, 652.0]:
            Leaf: predicts 0
            Value = (652.0, 676.0]:
            Leaf: predicts 0
            Value = (676.0, 850.0]:
            Leaf: predicts 0
    Value = (13.49, 20.0]:
    Split on feature: loan_intent
        Value = DEBTCONSOLIDATION:
        Split on feature: person_age
            Value = (19.999, 23.0]:
            Leaf: predicts 1
            Value = (23.0, 25.0]:
            Leaf: predicts 1
            Value = (25.0, 27.0]:
            Leaf: predicts 1
            Value = (27.0, 31.0]:
            Leaf: predicts 1
            Value = (31.0, 144.0]:
            Leaf: predicts 1
        Value = EDUCATION:
        Split on feature: loan_amnt
            Value = (499.999, 4400.0]:
            Leaf: predicts 1
            Value = (4400.0, 6600.0]:
            Leaf: predicts 1
            Value = (6600.0, 10000.0]:
            Leaf: predicts 0
            Value = (10000.0, 14520.2]:
            Leaf: predicts 0
            Value = (14520.2, 35000.0]:
```

```
        Leaf: predicts 0
    Value = HOMEIMPROVEMENT:
    Split on feature: person_income
        Value = (7999.999, 43081.8]:
        Leaf: predicts 1
        Value = (43081.8, 59456.0]:
        Leaf: predicts 1
        Value = (59456.0, 76436.8]:
        Leaf: predicts 1
        Value = (76436.8, 103841.0]:
        Leaf: predicts 1
        Value = (103841.0, 7200766.0]:
        Leaf: predicts 1
    Value = MEDICAL:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Leaf: predicts 1
        Value = OTHER:
        Leaf: predicts 1
        Value = OWN:
        Leaf: predicts 0
        Value = RENT:
        Leaf: predicts 1
    Value = PERSONAL:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Leaf: predicts 1
        Value = (6600.0, 10000.0]:
        Leaf: predicts 1
        Value = (10000.0, 14520.2]:
        Leaf: predicts 0
        Value = (14520.2, 35000.0]:
        Leaf: predicts 0
    Value = VENTURE:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Leaf: predicts 1
        Value = (6600.0, 10000.0]:
        Leaf: predicts 1
        Value = (10000.0, 14520.2]:
        Leaf: predicts 0
```

```
            Value = (14520.2, 35000.0]:
            Leaf: predicts 1
    Value = (0.21, 0.66]:
    Split on feature: person_home_ownership
        Value = MORTGAGE:
        Split on feature: loan_int_rate
            Value = (5.419, 7.88]:
            Split on feature: loan_amnt
                Value = (4400.0, 6600.0]:
                Leaf: predicts 1
                Value = (6600.0, 10000.0]:
                Leaf: predicts 0
                Value = (10000.0, 14520.2]:
                Leaf: predicts 0
                Value = (14520.2, 35000.0]:
                Leaf: predicts 0
            Value = (7.88, 10.58]:
            Split on feature: person_income
                Value = (7999.999, 43081.8]:
                Leaf: predicts 0
                Value = (43081.8, 59456.0]:
                Leaf: predicts 0
                Value = (59456.0, 76436.8]:
                Leaf: predicts 0
                Value = (76436.8, 103841.0]:
                Leaf: predicts 0
                Value = (103841.0, 7200766.0]:
                Leaf: predicts 0
            Value = (10.58, 11.49]:
            Split on feature: loan_intent
                Value = DEBTCONSOLIDATION:
                Leaf: predicts 1
                Value = EDUCATION:
                Leaf: predicts 0
                Value = HOMEIMPROVEMENT:
                Leaf: predicts 0
                Value = MEDICAL:
                Leaf: predicts 1
                Value = PERSONAL:
                Leaf: predicts 0
                Value = VENTURE:
                Leaf: predicts 0
            Value = (11.49, 13.49]:
            Split on feature: person_income
                Value = (7999.999, 43081.8]:
```

```
            Leaf: predicts 1
            Value = (43081.8, 59456.0]:
            Leaf: predicts 0
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
            Value = (76436.8, 103841.0]:
            Leaf: predicts 0
            Value = (103841.0, 7200766.0]:
            Leaf: predicts 0
        Value = (13.49, 20.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 0
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 1
            Value = PERSONAL:
            Leaf: predicts 0
            Value = VENTURE:
            Leaf: predicts 1
    Value = OTHER:
    Leaf: predicts 1
    Value = OWN:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Leaf: predicts 1
        Value = (6600.0, 10000.0]:
        Split on feature: loan_int_rate
            Value = (5.419, 7.88]:
            Leaf: predicts 0
            Value = (7.88, 10.58]:
            Leaf: predicts 0
            Value = (10.58, 11.49]:
            Leaf: predicts 0
            Value = (11.49, 13.49]:
            Leaf: predicts 0
            Value = (13.49, 20.0]:
            Leaf: predicts 1
        Value = (10000.0, 14520.2]:
        Leaf: predicts 0
```

```
        Value = (14520.2, 35000.0]:
        Split on feature: loan_int_rate
            Value = (5.419, 7.88]:
            Leaf: predicts 0
            Value = (7.88, 10.58]:
            Leaf: predicts 0
            Value = (10.58, 11.49]:
            Leaf: predicts 0
            Value = (11.49, 13.49]:
            Leaf: predicts 0
            Value = (13.49, 20.0]:
            Leaf: predicts 0
    Value = RENT:
    Split on feature: loan_amnt
        Value = (499.999, 4400.0]:
        Leaf: predicts 1
        Value = (4400.0, 6600.0]:
        Split on feature: loan_intent
            Value = DEBTCONSOLIDATION:
            Leaf: predicts 1
            Value = EDUCATION:
            Leaf: predicts 1
            Value = HOMEIMPROVEMENT:
            Leaf: predicts 1
            Value = MEDICAL:
            Leaf: predicts 1
            Value = PERSONAL:
            Leaf: predicts 1
            Value = VENTURE:
            Leaf: predicts 1
        Value = (6600.0, 10000.0]:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 1
            Value = (43081.8, 59456.0]:
            Leaf: predicts 0
        Value = (10000.0, 14520.2]:
        Split on feature: person_income
            Value = (7999.999, 43081.8]:
            Leaf: predicts 1
            Value = (43081.8, 59456.0]:
            Leaf: predicts 1
            Value = (59456.0, 76436.8]:
            Leaf: predicts 0
        Value = (14520.2, 35000.0]:
```

```
               Split on feature: person_income
                   Value = (7999.999, 43081.8]:
                   Leaf: predicts 1
                   Value = (43081.8, 59456.0]:
                   Leaf: predicts 1
                   Value = (59456.0, 76436.8]:
                   Leaf: predicts 1
                   Value = (76436.8, 103841.0]:
                   Leaf: predicts 1
                   Value = (103841.0, 7200766.0]:
                   Leaf: predicts 1
   Value = Yes:
   Leaf: predicts 0
```
```

### Source Code

```python
import numpy as np
import pandas as pd
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

class DecisionTree():
   def __init__(self, max_depth: int = None, min_samples_split: int = 2):
       self.max_depth = max_depth if max_depth is not None else float('inf')   #
maximum depth of the tree
       self.min_samples_split = min_samples_split  # minimum number of samples
required to split an internal node

       # a recursive dictionary that represents the decision tree
       # has 4 keys: 'leaf', 'label', 'split_feature', 'children'
       # these 4 keys represent only the first node of the tree
       # other nodes are stored in the 'children' key recursively
       self._tree = None

   def __str__(self):
       if self._tree is None:
           return "Decision tree has not been fitted yet."

       return self._tree_to_str(self._tree, indent="")


   def _tree_to_str(self, node, indent=""):
       if node['leaf']:
```
```

```
            return f"{indent}Leaf: predicts {node['label']}\n"    # return the label
of the leaf node

        # if the node is not a leaf node, print the split feature and recursively
print the children
        # also create a tree-like structure with indentations
        s = f"{indent}Split on feature: {node['split_feature']}\n"
        for val, child_node in node['children'].items():
            s += f"{indent}     Value = {val}:\n"
            s += self._tree_to_str(child_node, indent + "         ")
        return s


    def fit(self, X, y):
        self._tree = self._fit(X, y, depth=0)


    def _fit(self, X, y, depth):
        # return a leaf node if the stopping criteria are met (base case)

        # base case 1: if all the labels are the same, return a leaf node
        if len(np.unique(y)) == 1:
            return {
                'leaf': True,
                'label': y.iloc[0],
                'depth': depth
            }

        # base case 2: if there are no features left to split on, return a leaf node
        if len(X.columns) == 0:
            return {
                'leaf': True,
                'label': y.value_counts().idxmax(),
                'depth': depth
            }

        # base case 3: if the maximum depth is reached, return a leaf node
        if depth >= self.max_depth:
            return {
                'leaf': True,
                'label': y.value_counts().idxmax(),
                'depth': depth
            }
```

```
        # base case 4: if the number of samples is less than the minimum samples required to split, return a leaf node
        if len(y) < self.min_samples_split:
            return {
                'leaf': True,
                'label': y.value_counts().idxmax(),
                'depth': depth
            }

        current_entropy = self._entropy(y)   # calculate the entropy of the current node

        best_info_gain = 0
        best_feature = None
        best_X_list = []
        best_y_list = []

        # iterate over all the features and find the one that gives the best information gain
        for feature in X.columns:
            X_list, y_list = self._split_feature(X, y, feature)
            info_gain = self._information_gain(current_entropy, y, y_list)
            if info_gain > best_info_gain:
                best_info_gain = info_gain
                best_feature = feature
                best_X_list = X_list
                best_y_list = y_list

        # base case 5: if the best information gain is 0 (no improvement on the model), return a leaf node
        if best_info_gain == 0:
            return {
                'leaf': True,
                'label': y.value_counts().idxmax(),
                'depth': depth
            }

        depth += 1  # increment the depth of the tree, since we are going to split on a feature

        # create a node with the best feature to split on
        node = {
            'leaf': False,
            'split_feature': best_feature,
            'children': {},
```

```
            'label': y.value_counts().idxmax(),  # non-leaf nodes have labels as
well (most common label) as a fallout plan
            'depth': depth
        }

        # recursively create the children of the node
        unique_values = np.unique(X[best_feature])
        for i, val in enumerate(unique_values):
            child_subtree = self._fit(best_X_list[i], best_y_list[i], depth)
            node['children'][val] = child_subtree

        return node


    def predict(self, X):
        # apply the _predict_one function to each row of the dataframe
        predictions = []
        for _, row in X.iterrows():
            predictions.append(self._predict_one(row))
        return pd.Series(predictions, index=X.index)


    def _predict_one(self, x):
        # traverse the tree until a leaf node is reached
        # while traversing go to the child node that corresponds to the value of the
split feature

        node = self._tree
        while not node['leaf']:
            split_feature = node['split_feature']
            val = x[split_feature]
            try:
                node = node['children'][val]

            # occurs when there is a combination of feature values that the model
has not seen before
            # in this case, return the most common label of the current node
            # label field of the non-leaf are used here
            except KeyError:
                return node['label']

        return node['label']


    def cross_validation(self, X, y, params, cross_val_splits=5):
```

```
        # assign defaults to parameters if they are not provided
        if not params['max_depth']:
            params['max_depth'] = [None]
        if not params['min_samples_split']:
            params['min_samples_split'] = [2]

        # check if the parameters are valid
        for key in params:
            if key not in ['max_depth', 'min_samples_split']:
                raise ValueError(f"Invalid parameter: {key}")

        best_accuracy = 0
        len_val = len(X) // cross_val_splits

        # iterate over all the hyperparameters
        for max_depth in params['max_depth']:
            for min_samples_split in params['min_samples_split']:
                sum_accuracy = 0
                self.max_depth = max_depth if max_depth is not None else
float('inf')
                self.min_samples_split = min_samples_split

                # iterate over all the cross validation splits
                for i in range(cross_val_splits):
                    # create the training and validation sets
                    X_train_cv = pd.concat([X.iloc[:i*len_val, :].copy(),
X.iloc[(i+1)*len_val:, :].copy()])
                    X_val = X.iloc[i * len_val: (i + 1) * len_val]
                    y_train_cv = pd.concat([y_train.iloc[:i*len_val].copy(),
y_train.iloc[(i+1)*len_val:].copy()])
                    y_val = y.iloc[i * len(y) // cross_val_splits: (i + 1) * len(y)
// cross_val_splits]

                    # fit the model and get the accuracy on the validation set
                    self.fit(X_train_cv, y_train_cv)
                    y_pred_cv = self.predict(X_val)
                    sum_accuracy += accuracy_score(y_val, y_pred_cv)

                print(f"max_depth: {max_depth}, min_samples_split:
{min_samples_split}, accuracy: {sum_accuracy / cross_val_splits}")

                # update the best hyperparameters if the current model is better
                if sum_accuracy / cross_val_splits > best_accuracy:
                    best_accuracy = sum_accuracy / cross_val_splits
                    best_params = (max_depth, min_samples_split)
```

```
        print(f"Best accuracy: {best_accuracy}, Best params: {best_params}")
        return best_params


    def _entropy(self, y):
        # calculate the entropy of a node for a given variable

        classes = np.unique(y)
        entropy = 0
        for class_ in classes:
            nominator = (y == class_).sum()
            denominator = len(y)
            item = nominator / denominator
            entropy -= item * np.log2(item)
        return entropy


    def _information_gain(self, current_entropy, old_y, new_y_list = []):
        # calculate the information gain for a given entropy, variable and list of
new splits

        for y in new_y_list:
            current_entropy -= (len(y) / len(old_y)) * self._entropy(y)
        return current_entropy


    def _split_feature(self, X, y, feature):
        # split the dataset based on the unique values of a feature
        # return the new X's and y's as lists
        X_list = []
        y_list = []
        for class_ in np.unique(X[feature]):
            mask = X[feature] == class_
            X_list.append(X[mask].drop(columns=[feature]))  # drop the feature
column to avoid using it again
            y_list.append(y[mask])
        return X_list, y_list


if __name__ == "__main__":
    df = pd.read_csv('loan_data.csv')

    # convert the numerical values to categorical values
```

```python
    # apply this to the test data as well, so that the test data has the same
categories as the training data
    numerical_classes = [
        'person_age',
        'person_income',
        'person_emp_exp',
        'loan_amnt',
        'loan_int_rate',
        'loan_percent_income',
        'cb_person_cred_hist_length',
        'credit_score',
    ]
    for class_ in numerical_classes:
        df[class_] = pd.qcut(df[class_], 5, duplicates='drop')

    X = df.drop('loan_status', axis=1)
    y = df['loan_status']

    # create a train test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1,
random_state=42)

    # apply cross validation to find the best hyperparameters
    dt = DecisionTree()
    params = {
        'max_depth': [2, 5, 7, 10, 12, None],
        'min_samples_split': [200, 100, 50, 20, 10, 5, 2]
    }
    best_params = dt.cross_validation(X_train, y_train, params)

    # create one more instance of the model with the best hyperparameters
    max_depth, min_samples_split = best_params
    dt = DecisionTree(max_depth=max_depth, min_samples_split=min_samples_split)

    # write the tree to a file to visualize it
    with open('tree.txt', 'w') as f:
        f.write(str(dt))

    # fit the model and make predictions
    dt.fit(X_train, y_train)
    y_pred = dt.predict(X_test)

    # calculate the accuracy
    print(accuracy_score(y_test, y_pred))
```