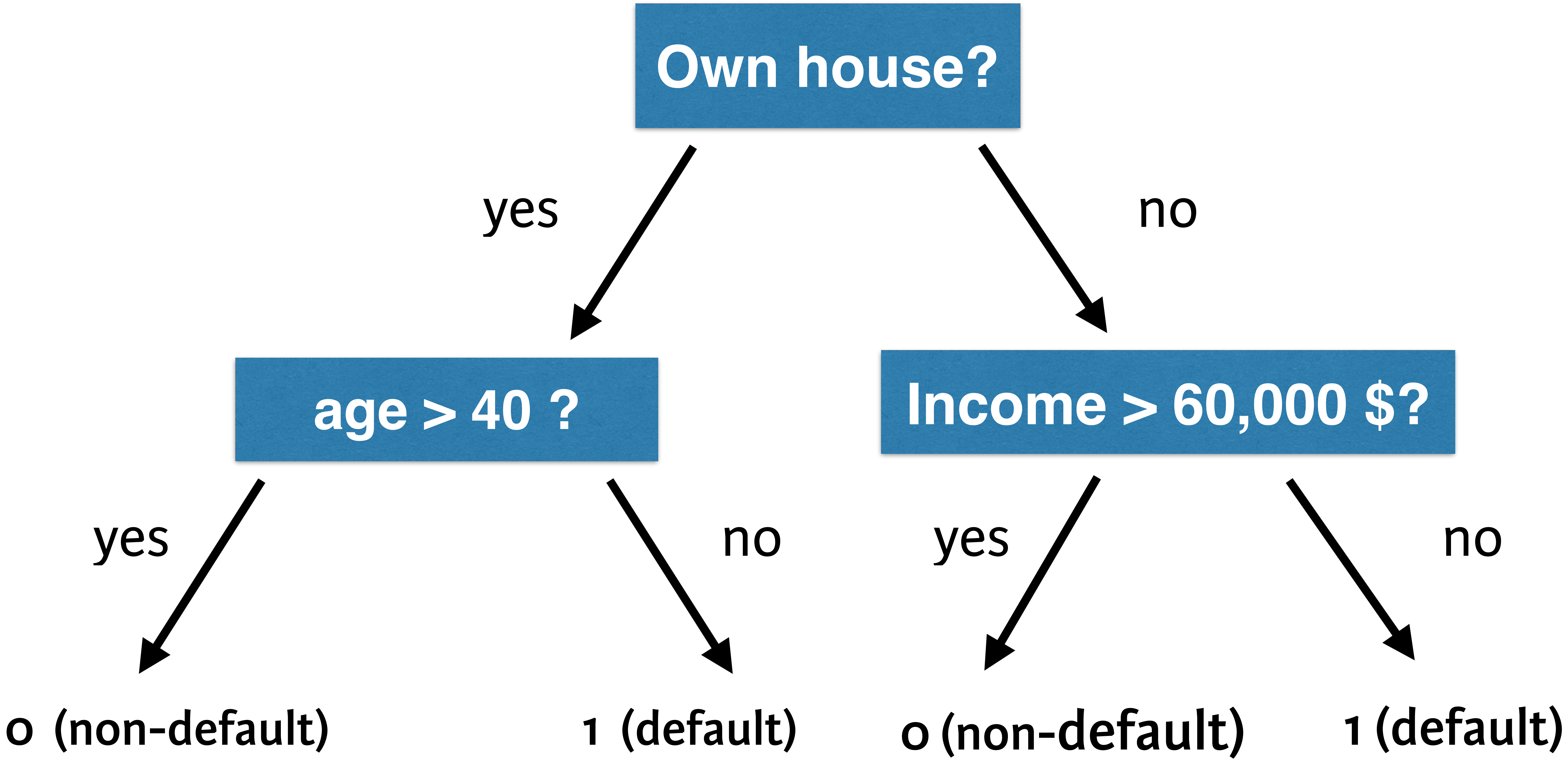
CREDIT RISK MODELING IN R

What is a decision tree?

Decision tree example



Own house?

yes

no

age > 40 ?

Income > 60,000 $?

yes

no

yes

no

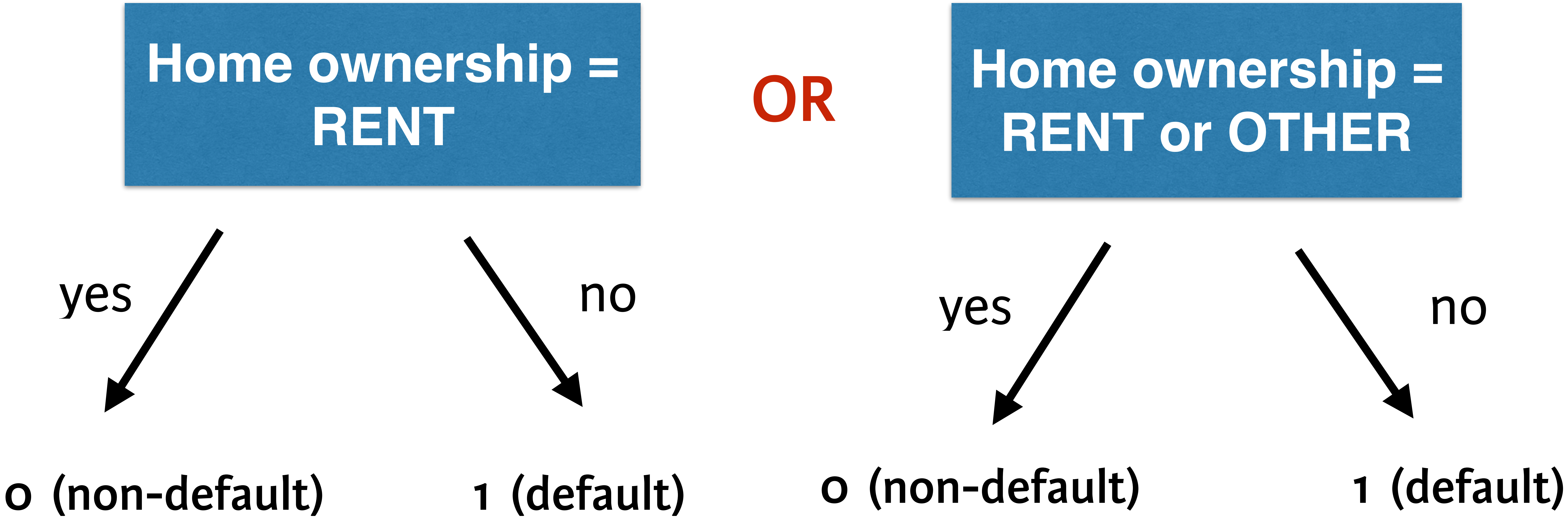
0 (non-default)

1 (default)

0 (non-default)

1 (default)

How to make splitting decision?



Home ownership = RENT

yes

no

0 (non-default)

1 (default)

OR

Home ownership = RENT or OTHER

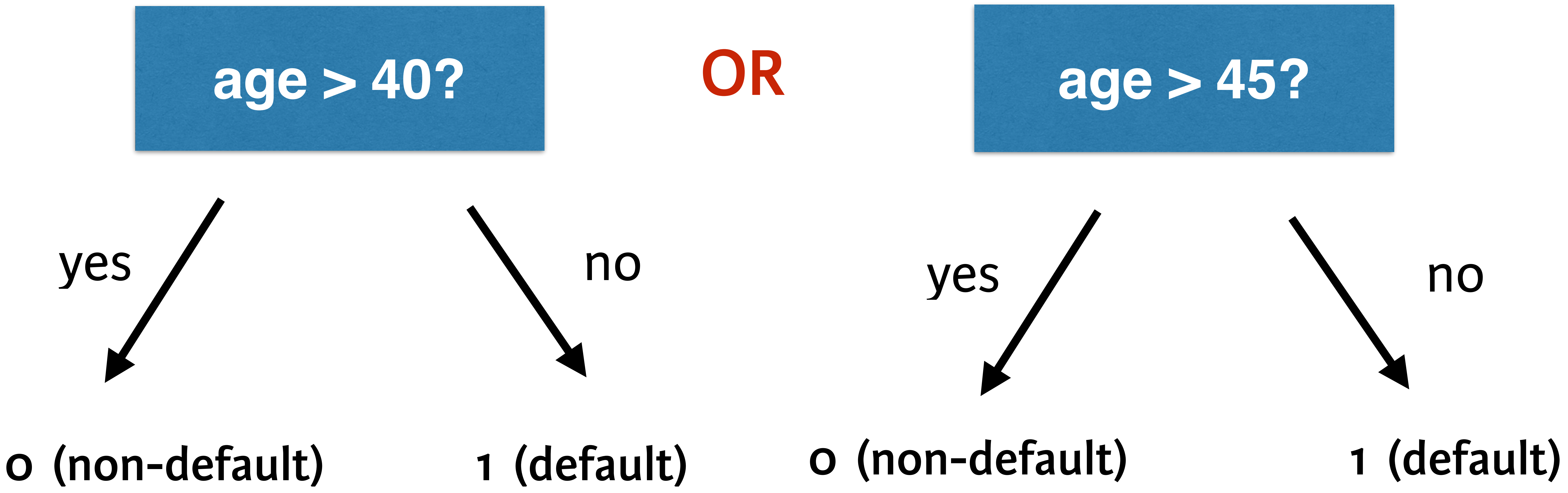
yes

no

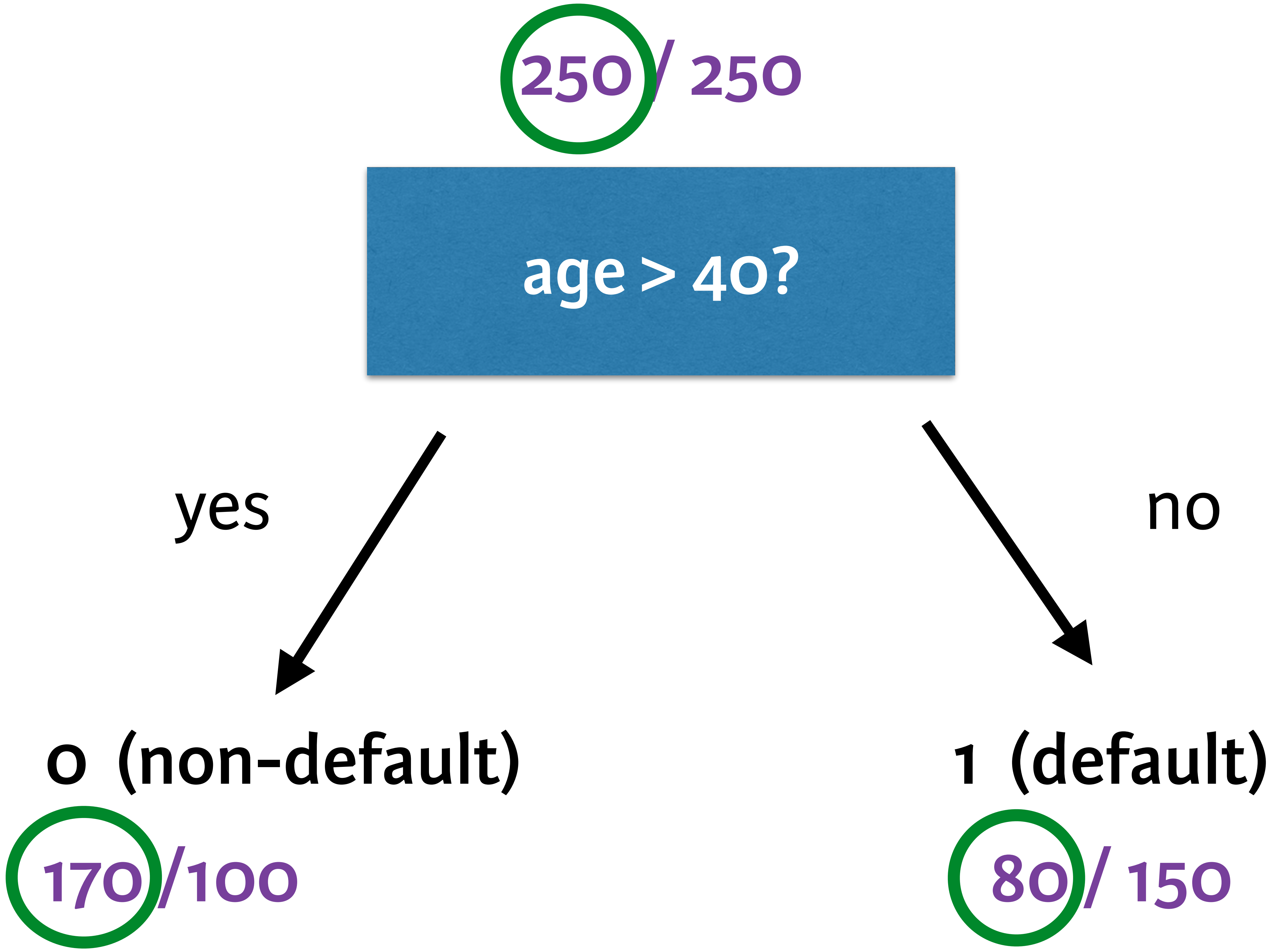
0 (non-default)

1 (default)

How to make splitting decision?

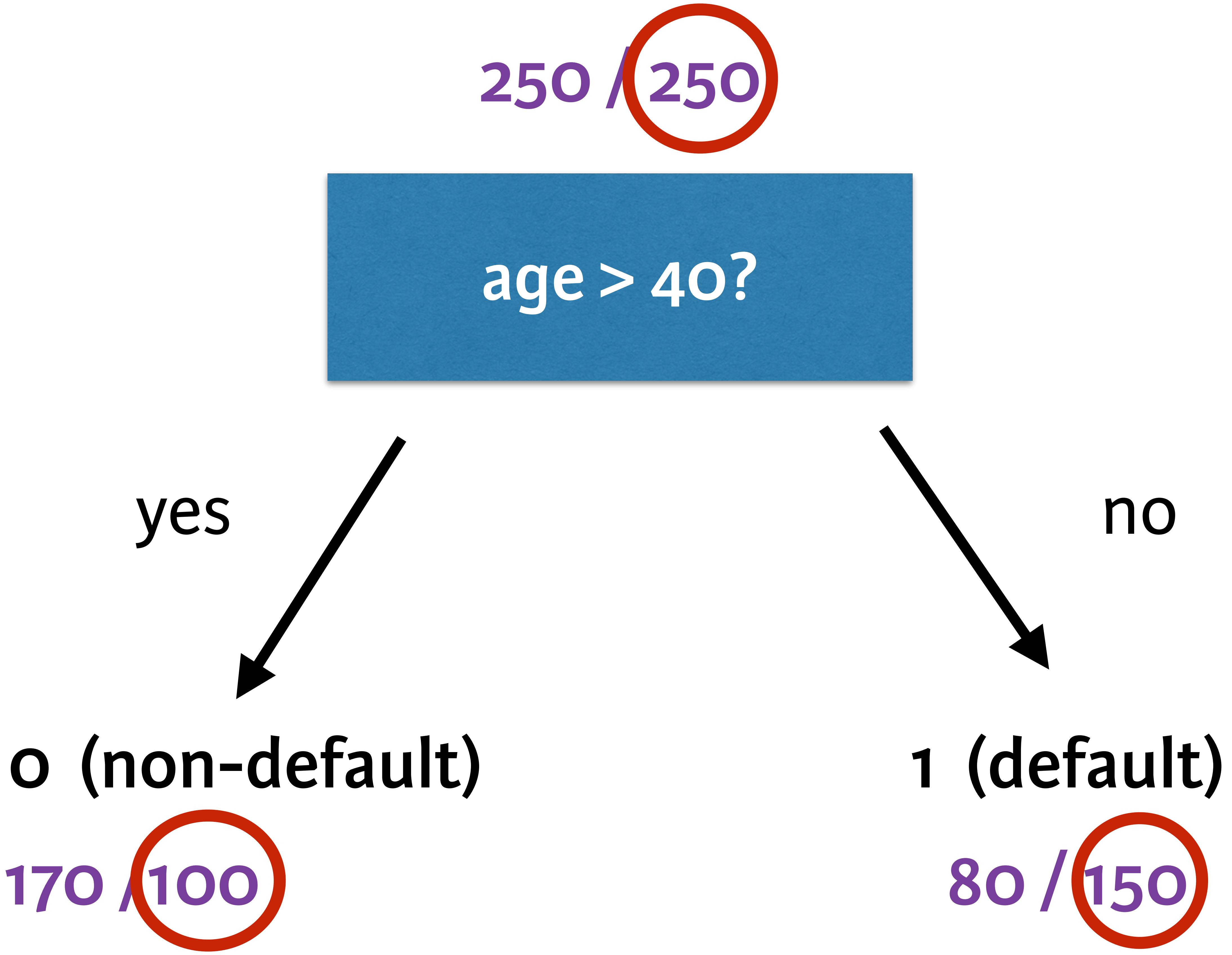


Example



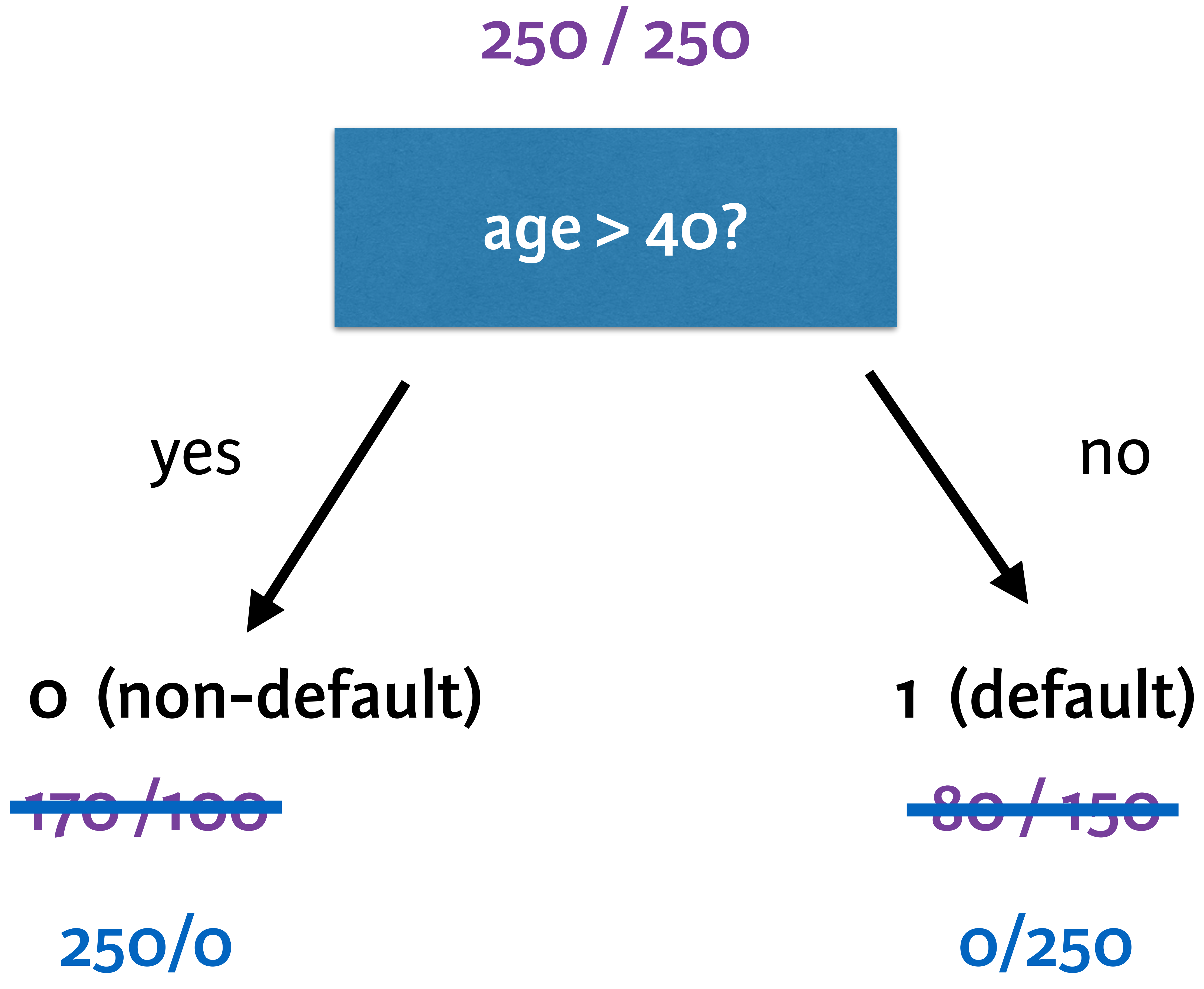
Actual **non-defaults** in this node using this split

Example



Actual **defaults** in this node using this split

Example



250 / 250

age > 40?

yes

no

0 (non-default)

~~170/100~~

250/0

1 (default)

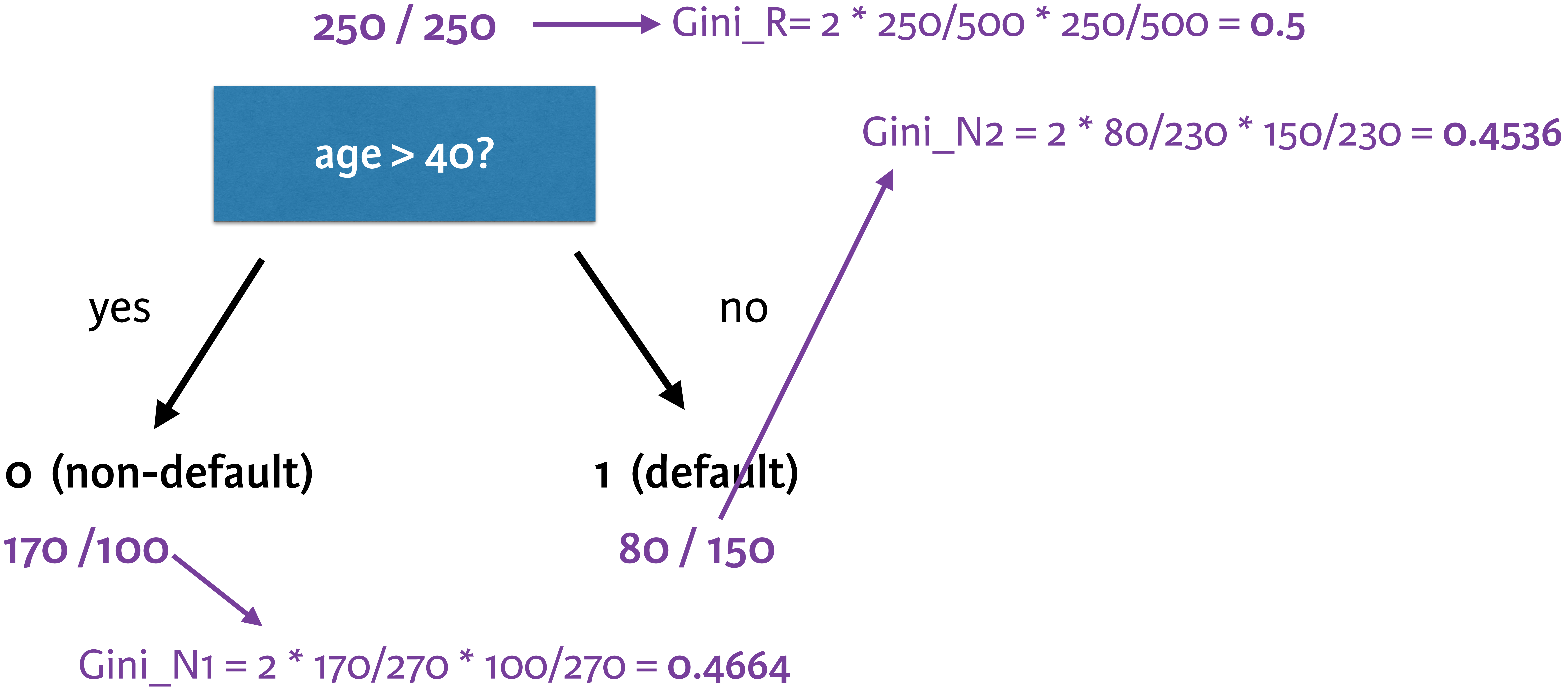
~~80/150~~

0/250

= IDEAL SCENARIO

Example

Gini = 2*prop(default)*prop(non-default)



250 / 250

Gini_R= 2 * 250/500 * 250/500 = **0.5**

age > 40?

Gini_N2 = 2 * 80/230 * 150/230 = **0.4536**

yes

no

0 (non-default)

1 (default)

170 /100

80 / 150

Gini_N1 = 2 * 170/270 * 100/270 = **0.4664**

Example

Gain = Gini_R - prop(cases in N1) * **Gini_N1**

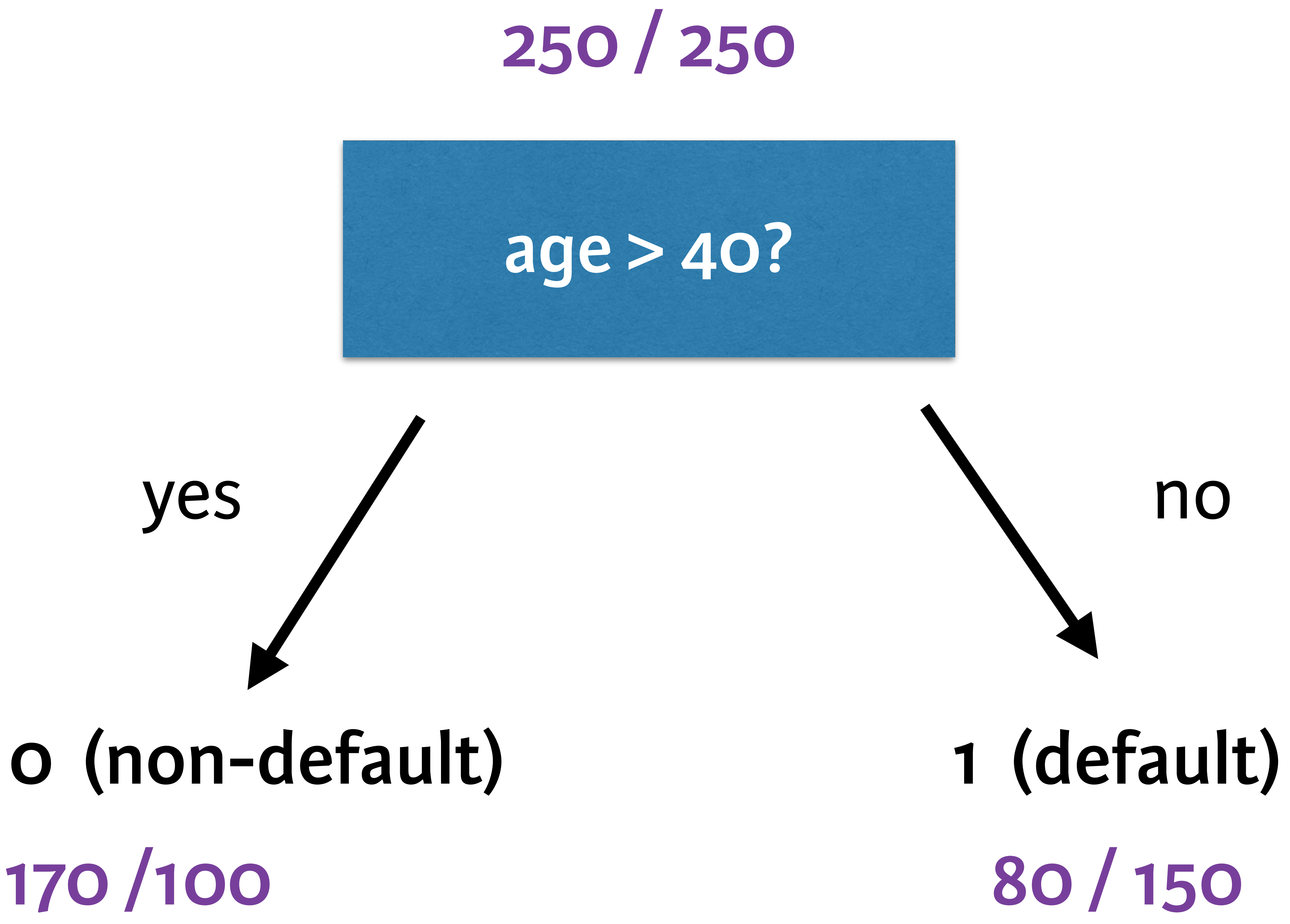
- prop(cases in N2) * **Gini_N2**

MAXIMIZE GAIN

= 0.5 - 270/500 * 0.4664

- 230/500*0.4536

= 0.039488



250 / 250

age > 40?

yes

no

0 (non-default)

170 /100

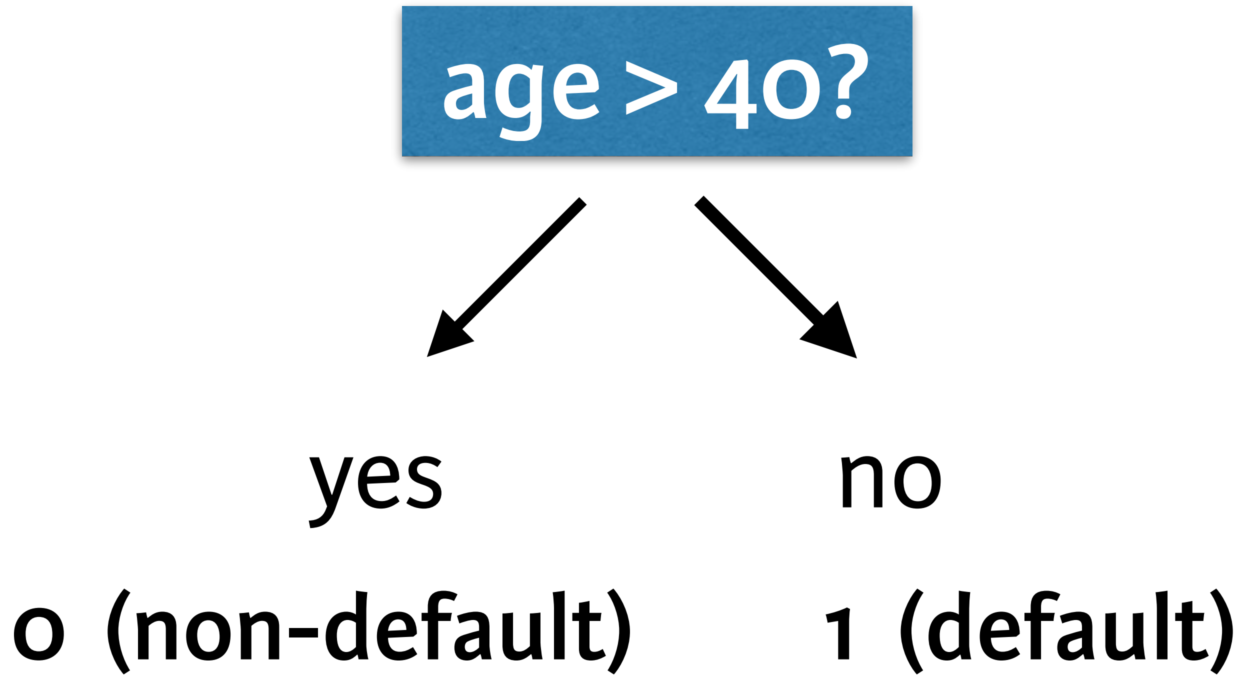
1 (default)

80 / 150

Let’s practice!

Building decision trees using the rpart()-package

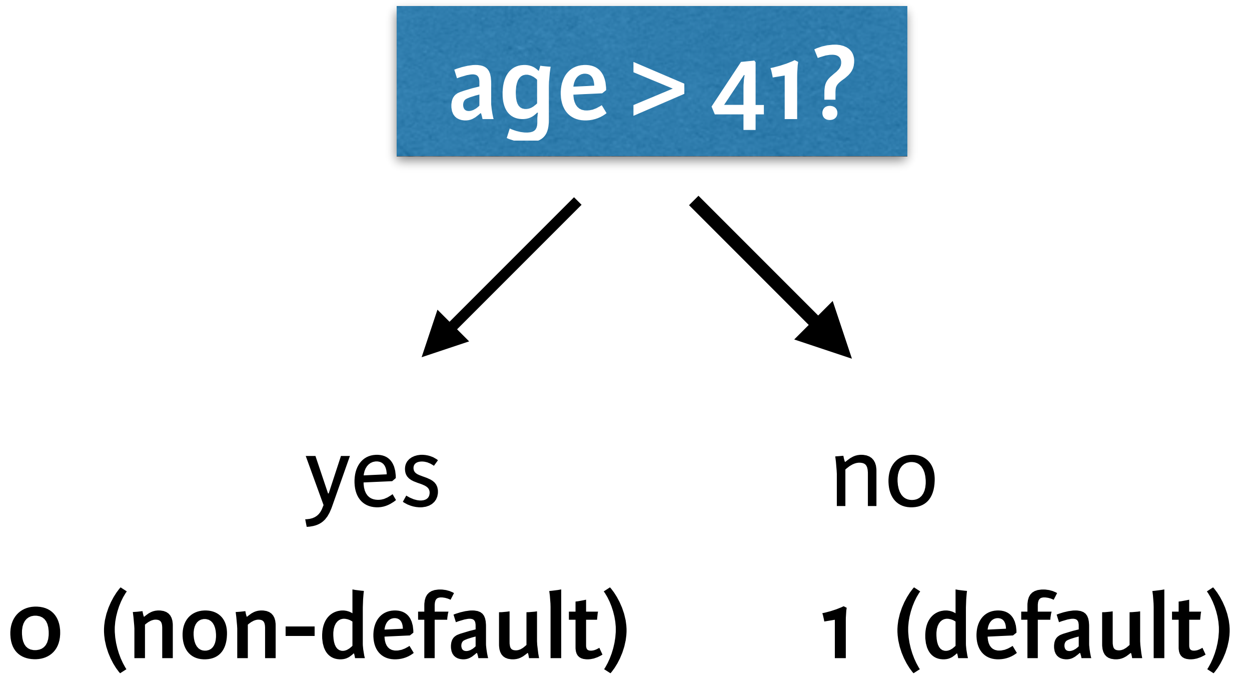
Imagine…



age > 40?

yes | no

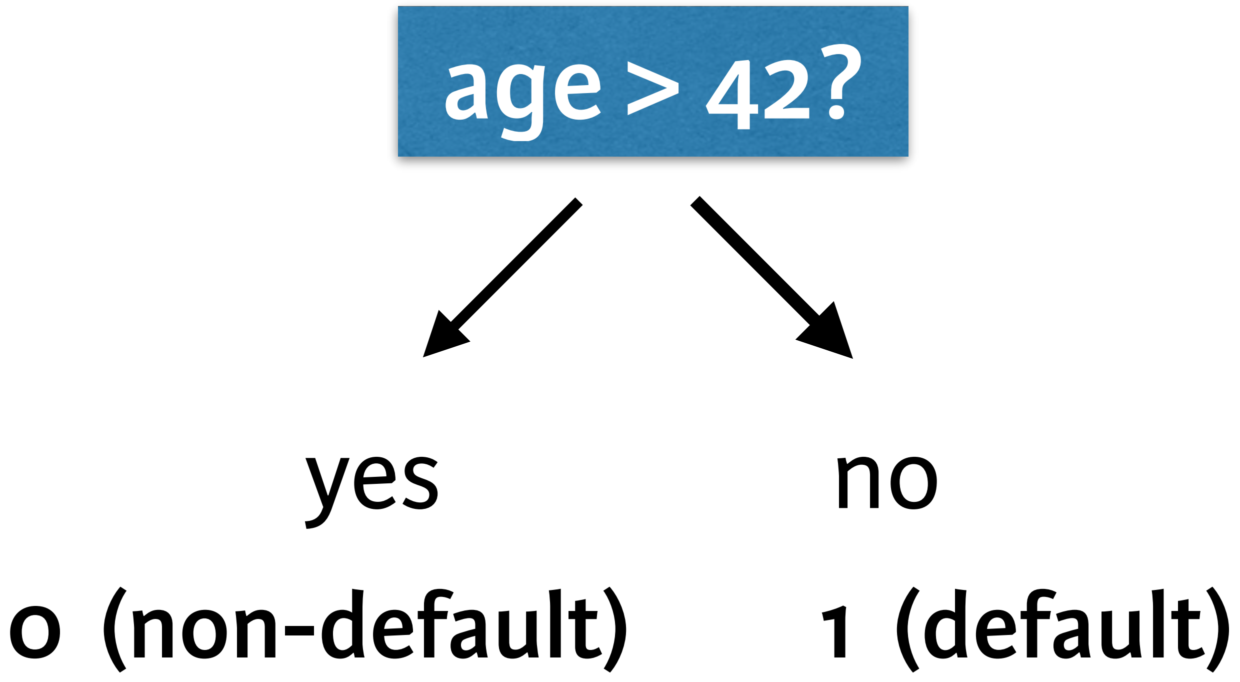
0 (non-default) | 1 (default)



age > 41?

yes | no

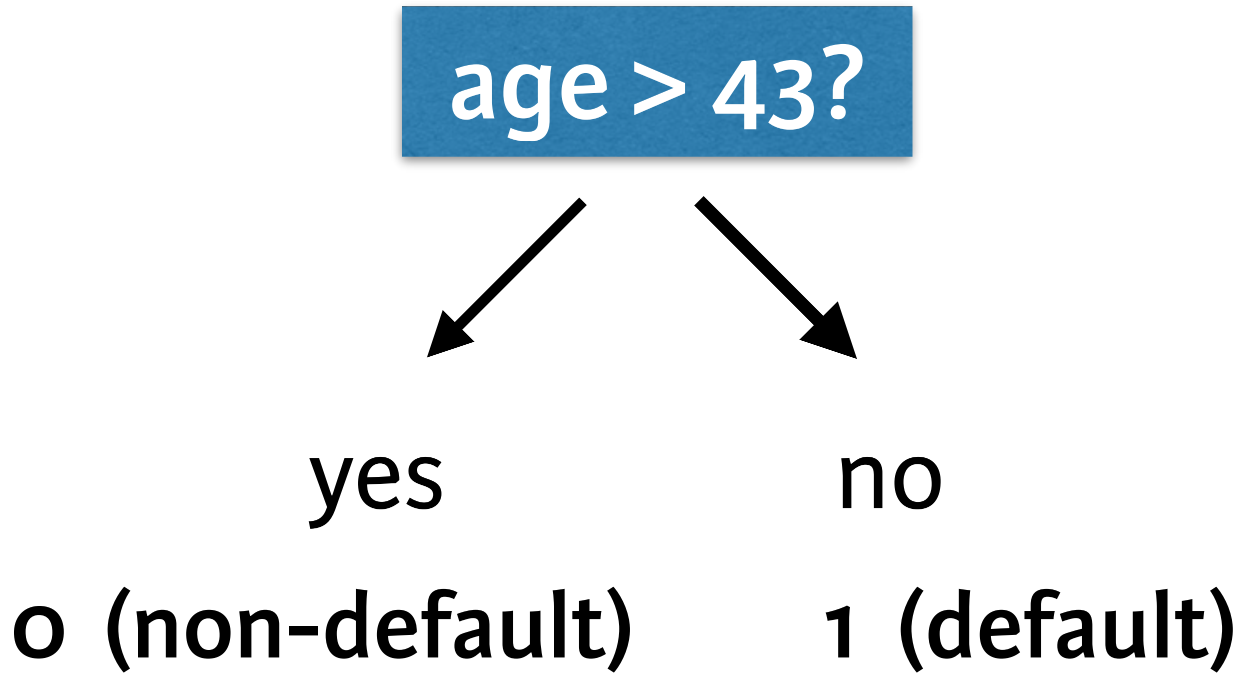
0 (non-default) | 1 (default)



age > 42?

yes | no

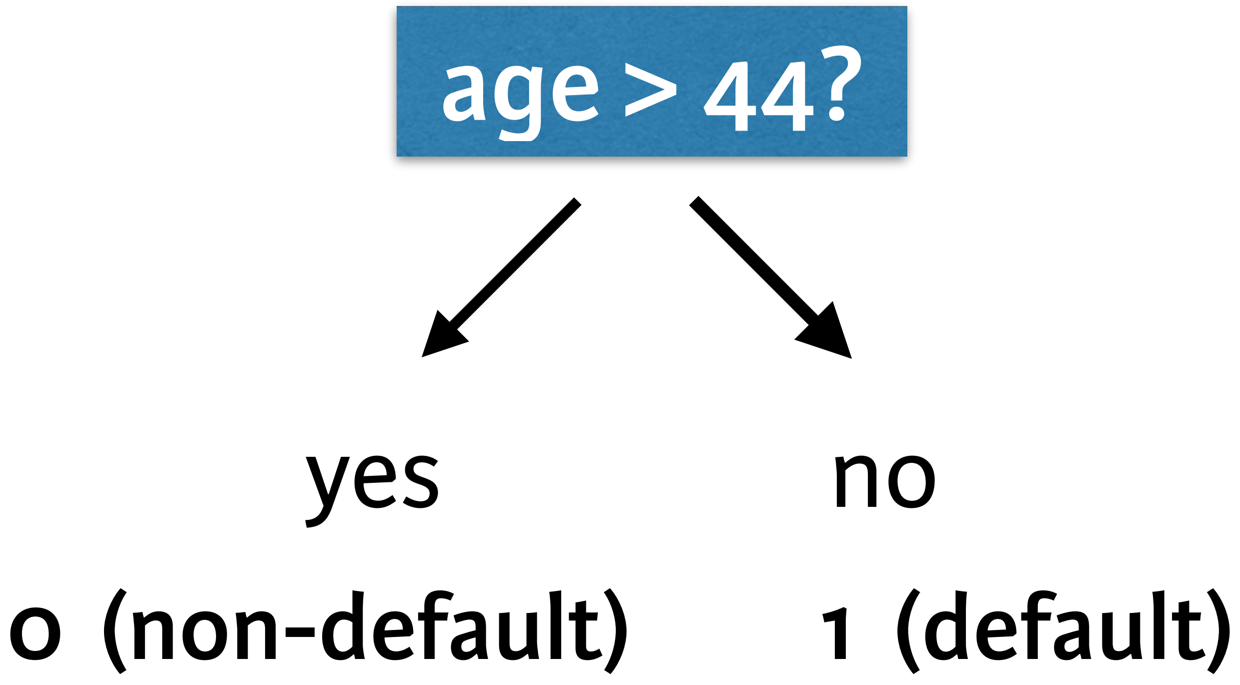
0 (non-default) | 1 (default)



age > 43?

yes | no

0 (non-default) | 1 (default)



age > 44?

yes | no

0 (non-default) | 1 (default)

. . .

rpart() package! But...

- hard building nice decision tree for credit risk data
- main reason: unbalanced data

```
> fit_default <- rpart(loan_status ~ ., method = “class",
 data = training_set)

> plot(fit_default)
Error in plot.rpart(fit_default) : fit is not a tree, just a root
```

Three techniques to overcome unbalance

- Undersampling or oversampling
    - Accuracy issue will disappear
    - Only training set
- Changing the prior probabilities
- Including a loss matrix

Validate model to see what is best!

Let’s practice!

CREDIT RISK MODELING IN R

Pruning the decision tree

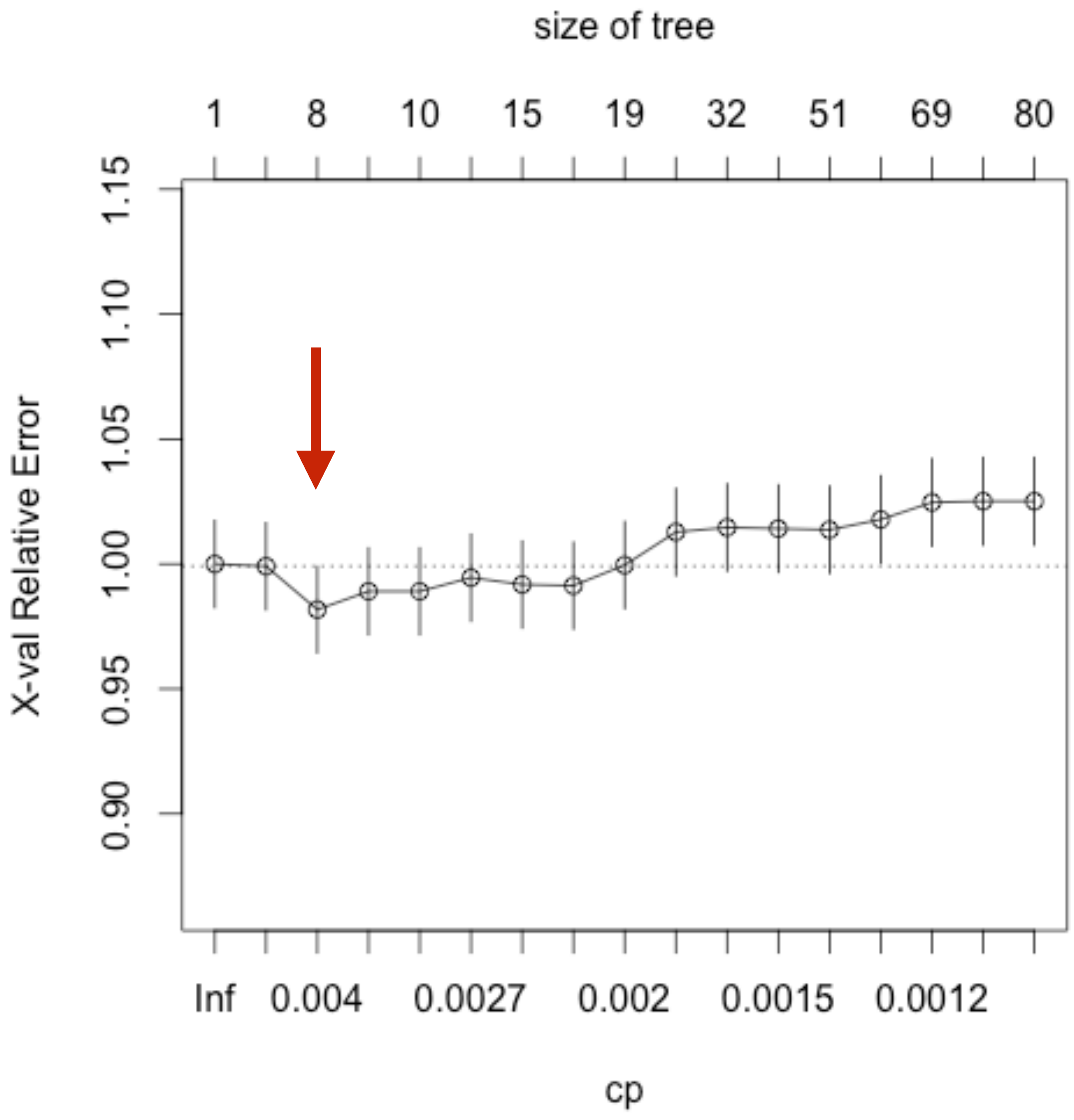
Problems with large decision trees

- Too complex: not clear anymore
- **Overfitting when applying to test set**
- Solution: use printcp(), plotcp() for pruning purposes

Printcp and tree_undersample

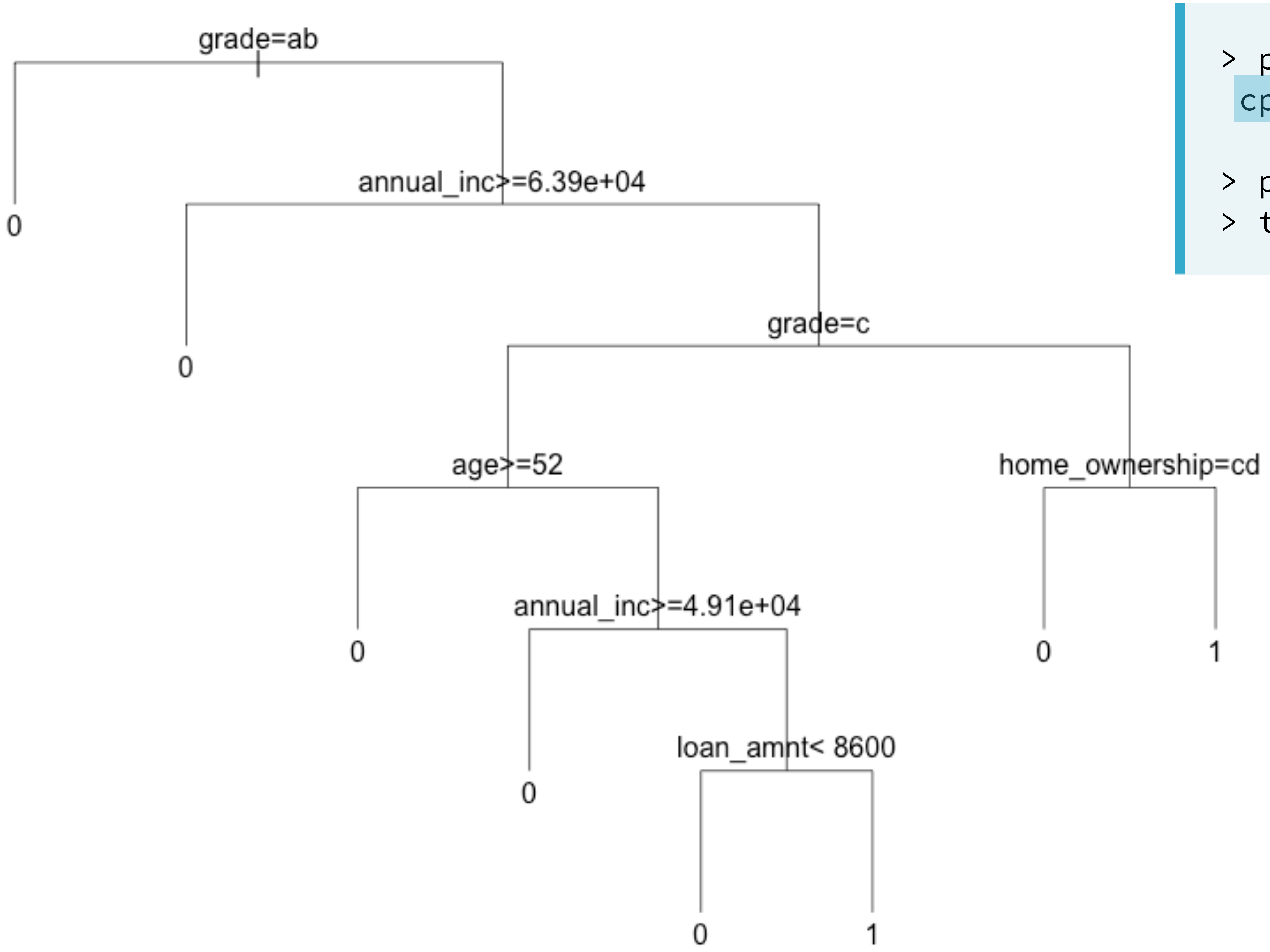
```
> printcp(tree_undersample)

Classification tree:
rpart(formula = loan_status ~ ., data = undersampled_training_set, method = "class",
 control = rpart.control(cp = 0.001))

Variables actually used in tree construction:
[1] age     annual_inc     emp_cat     grade    home_ownership   ir_cat     loan_amnt

Root node error: 2190/6570 = 0.33333

n= 6570

         CP     nsplit  rel error   xerror      xstd
1   0.0059361      0     1.00000   1.00000   0.017447
2   0.0044140      4     0.97443   0.99909   0.017443
3   0.0036530      7     0.96119   0.98174   0.017366
4   0.0031963      8     0.95753   0.98904   0.017399
                           …
16  0.0010654     76     0.84247   1.02511   0.017554
17  0.0010000     79     0.83927   1.02511   0.017554
```

Plotcp and tree_undersample



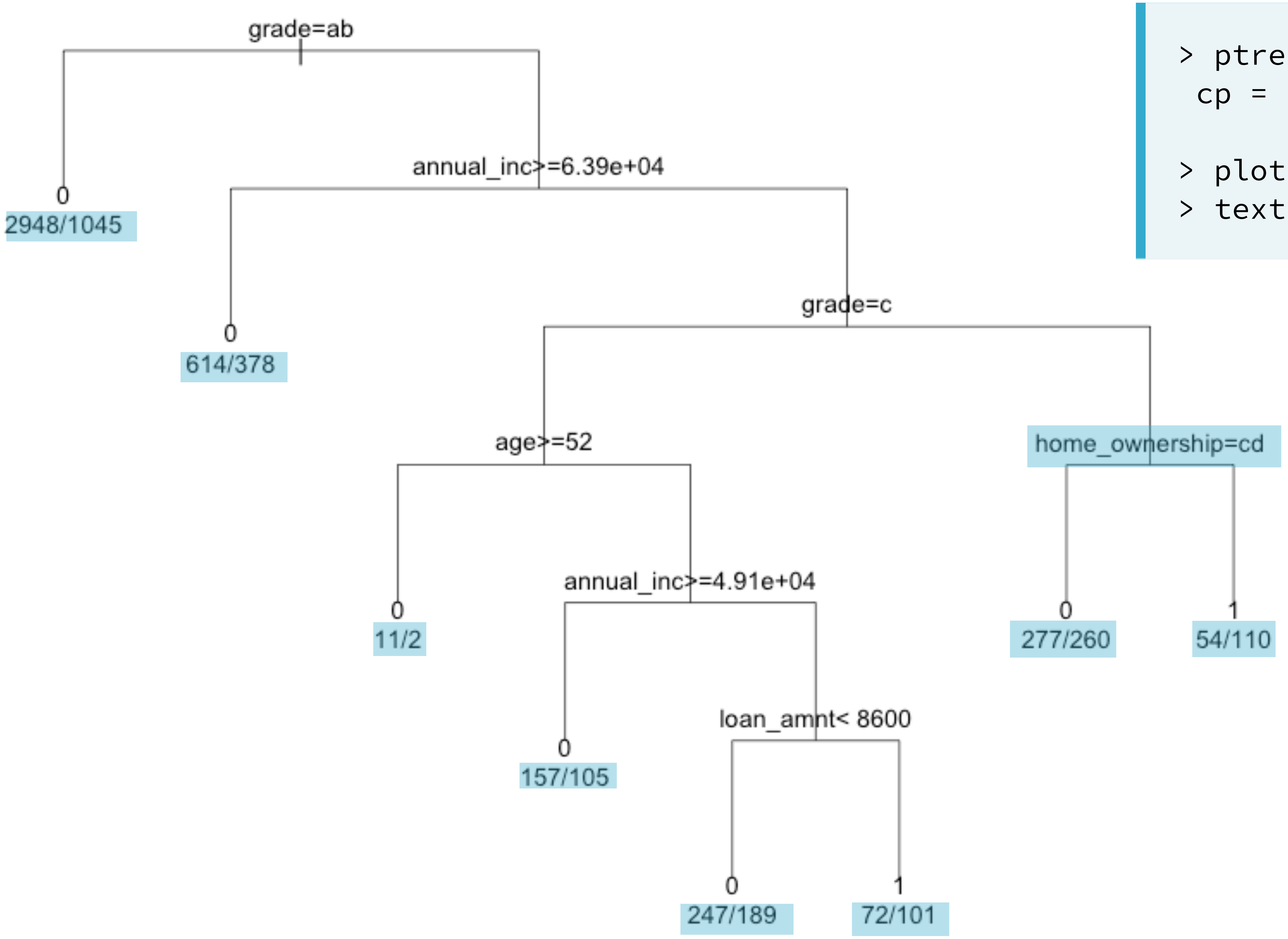
cp= 0.003653

plot the pruned tree



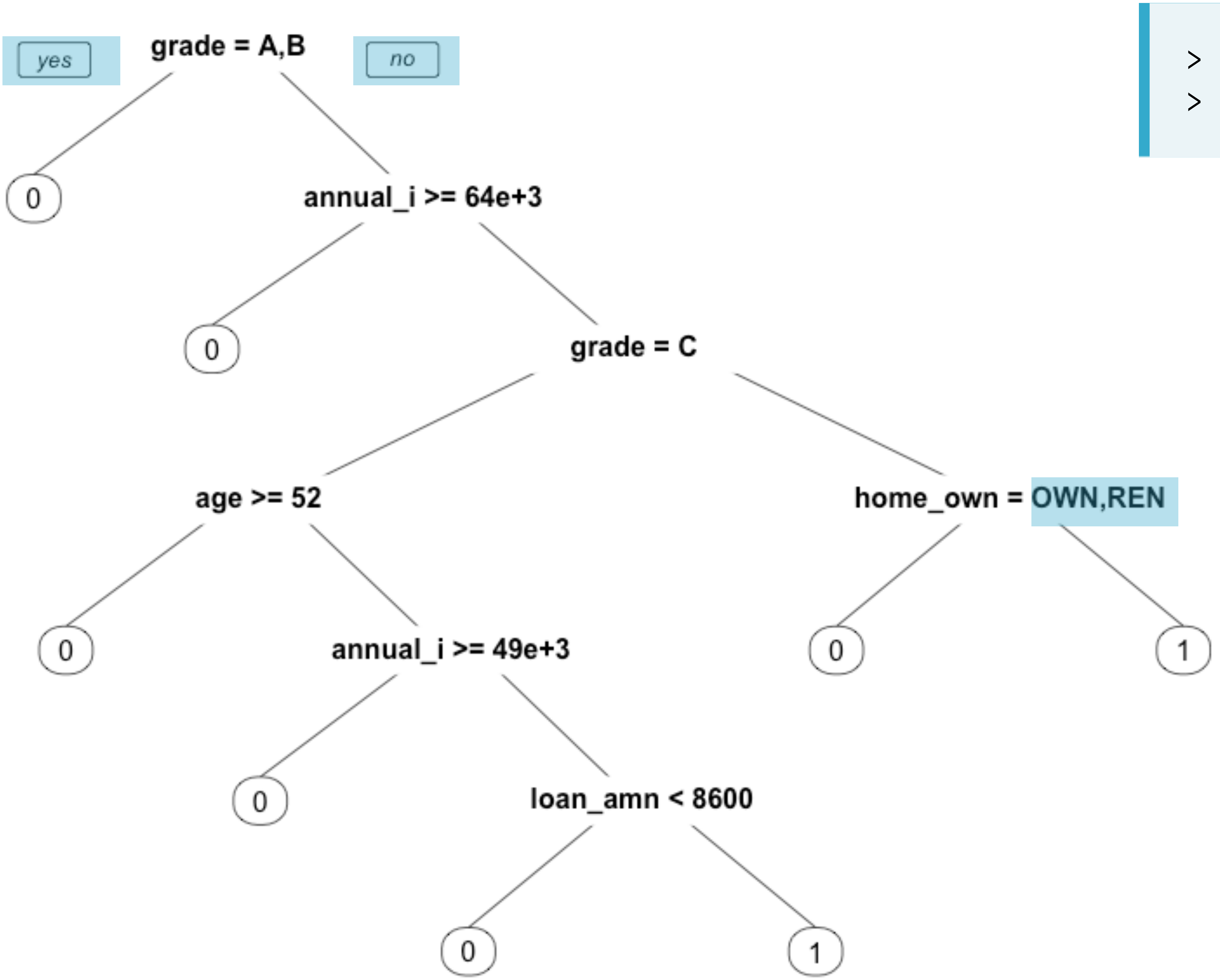
```
> ptree_undersample=prune(tree_undersample,
 cp = 0.003653)

> plot(ptree_undersample, uniform=TRUE)
> text(ptree_undersample)
```

plot the pruned tree



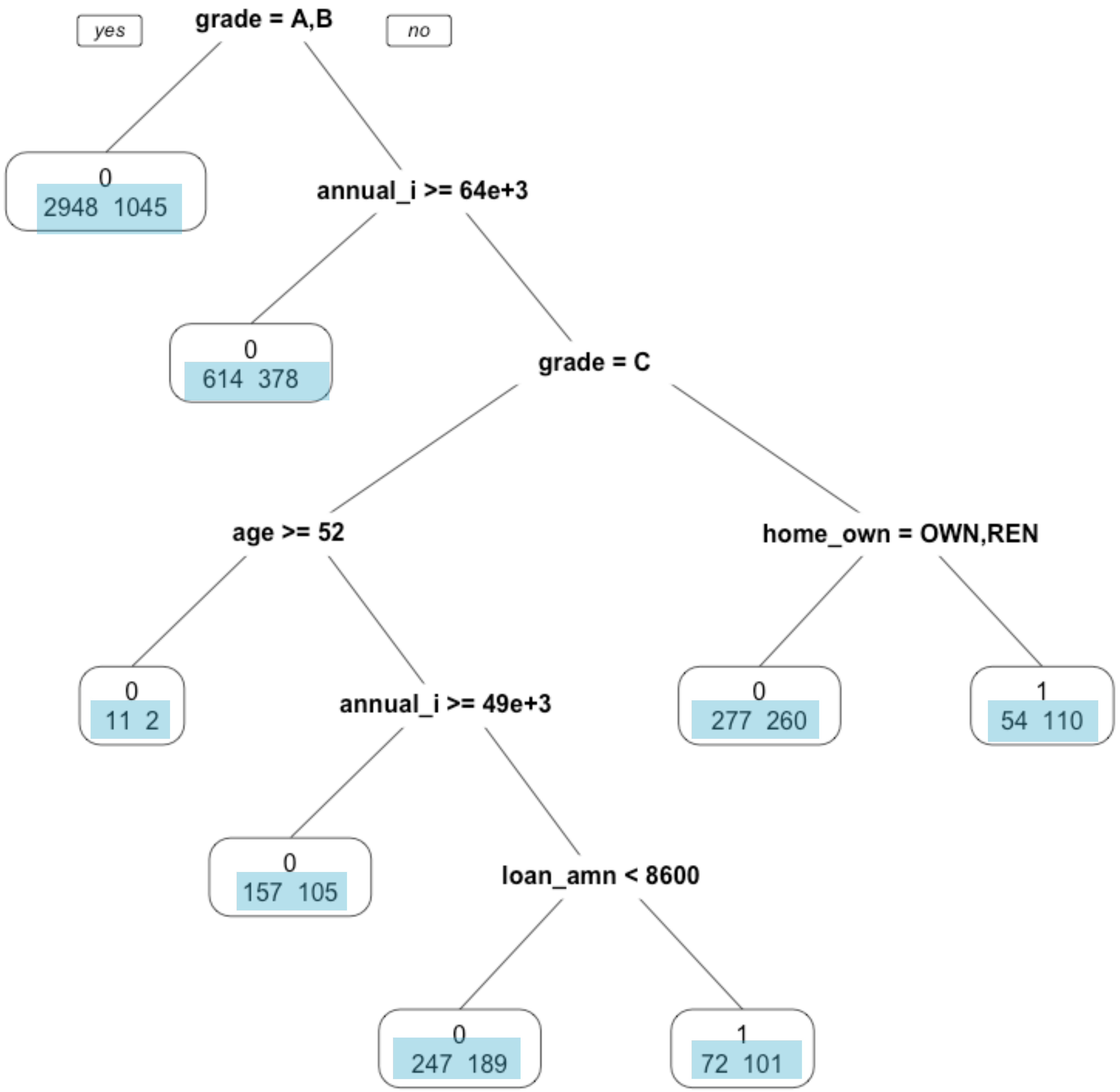
```
> ptree_undersample=prune(tree_undersample,
 cp = 0.003653)

> plot(ptree_undersample, uniform=TRUE)
> text(ptree_undersample, use.n=TRUE)
```

prp() in the rpart.plot-package



```
> library(rpart.plot)
> prp(ptree_undersample)
```

prp() in the part.plot-package



```
> library(rpart.plot)
> prp(ptree_undersample, extra = 1)
```

Let’s practice!

CREDIT RISK MODELING IN R

Other tree options and the construction of confusion matrices.

Other interesting rpart()-arguments

...in rpart()

- `weights:` include case weights

...in the control argument of rpart (rpart.control)

- `minsplit:` minimum number of observations for split attempt
- `minbucket:` minimum number of observations in leaf node

Making predictions using the decision tree

```
> pred_undersample_class = predict(ptree_undersample, newdata = test_set,
type =“class”)

1      2      3     …    29073 29079 29084 29090 29091
0      0      0     …      1     0     0     0     0
```

OR

```
> pred_undersample = predict(ptree_undersample, newdata = test_set)

          0         1
1      0.7382920 0.2617080
2      0.5665138 0.4334862
3      0.5992366 0.4007634
          …          …
29073 0.4161850 0.5838150
29079 0.6189516 0.3810484
29084 0.7382920 0.2617080
29090 0.7382920 0.2617080
29091 0.7382920 0.2617080
```

Constructing a confusion matrix

```
> table(test_set$loan_status, pred_undersample_class)

   pred_undersample_class
       0    1
  0 8314  346
  1  964   73
```

CREDIT RISK MODELING IN R

Let’s practice!