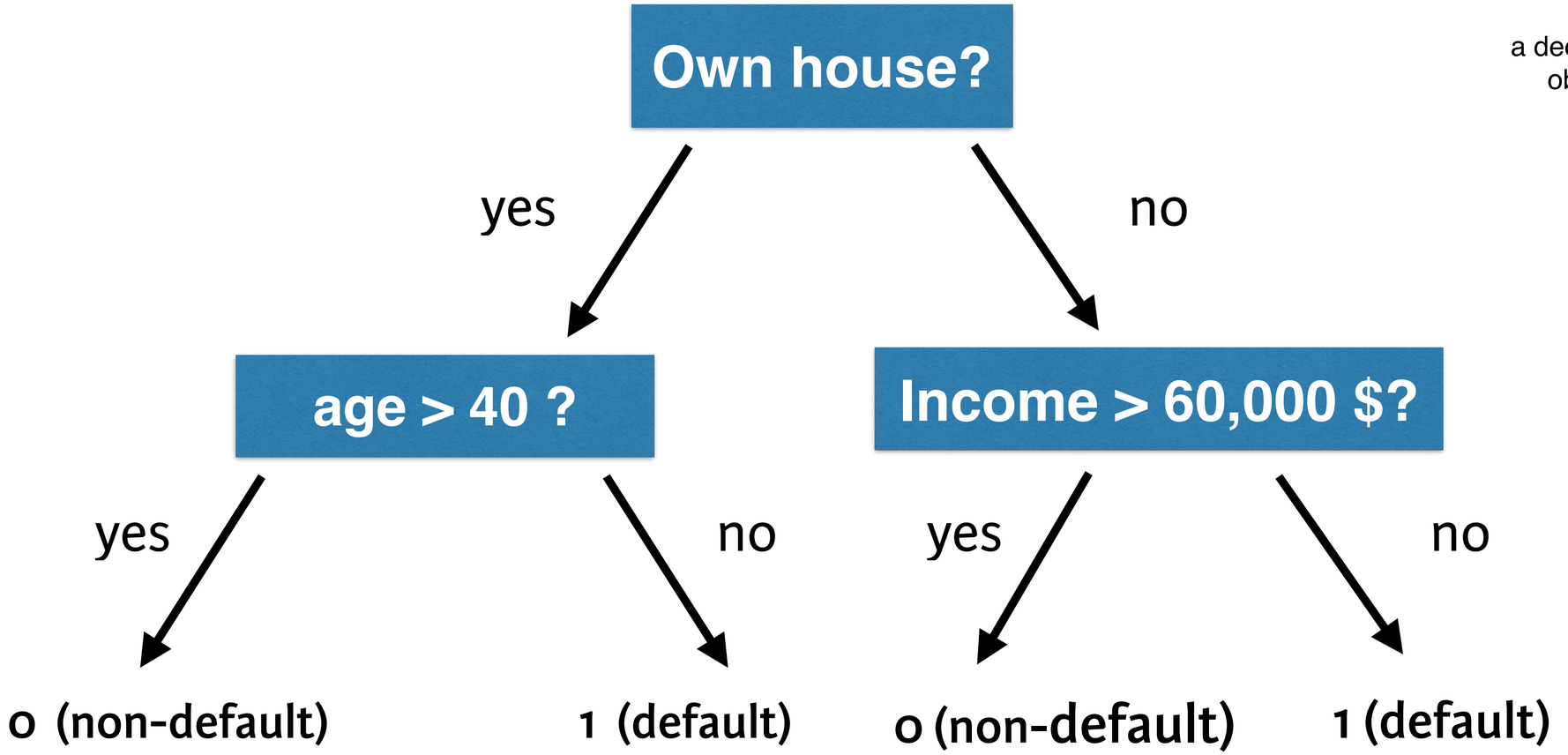
CREDIT RISK MODELING IN R

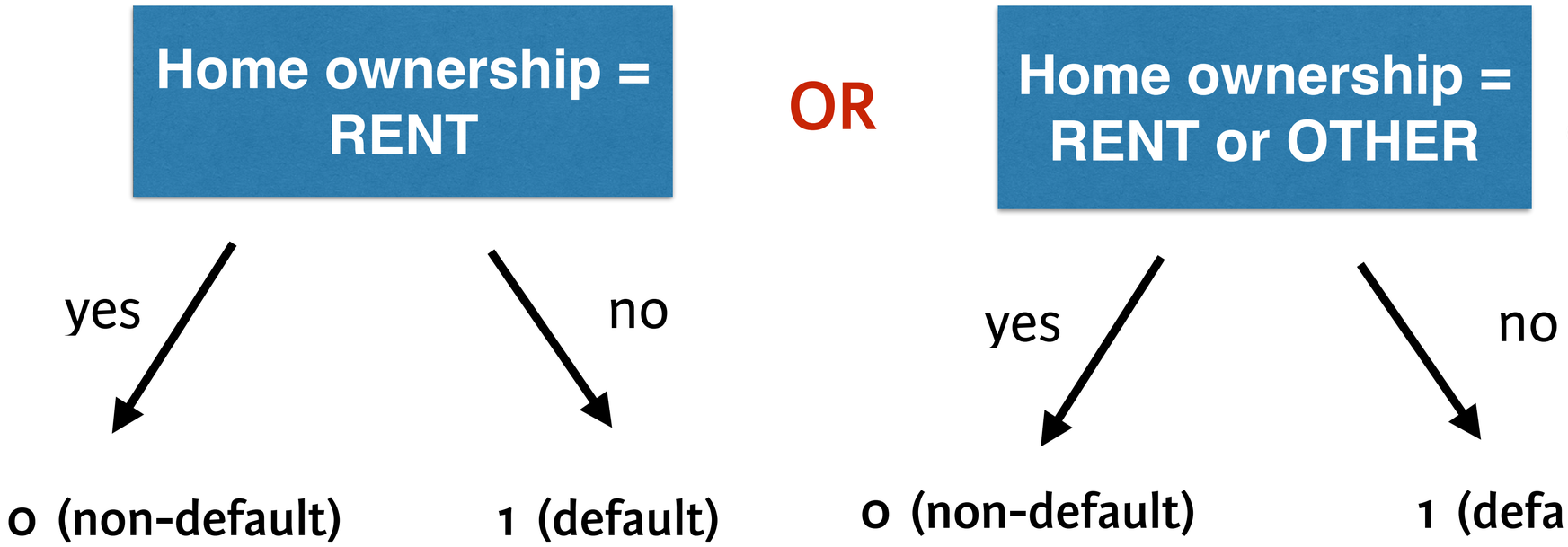
What is a decision tree?

Decision tree example

Answer the questions from the top or the Root to the bottom or leaves nodes,
a decision is made in wether if classify and observation as default or non-default



How to make splitting decision?



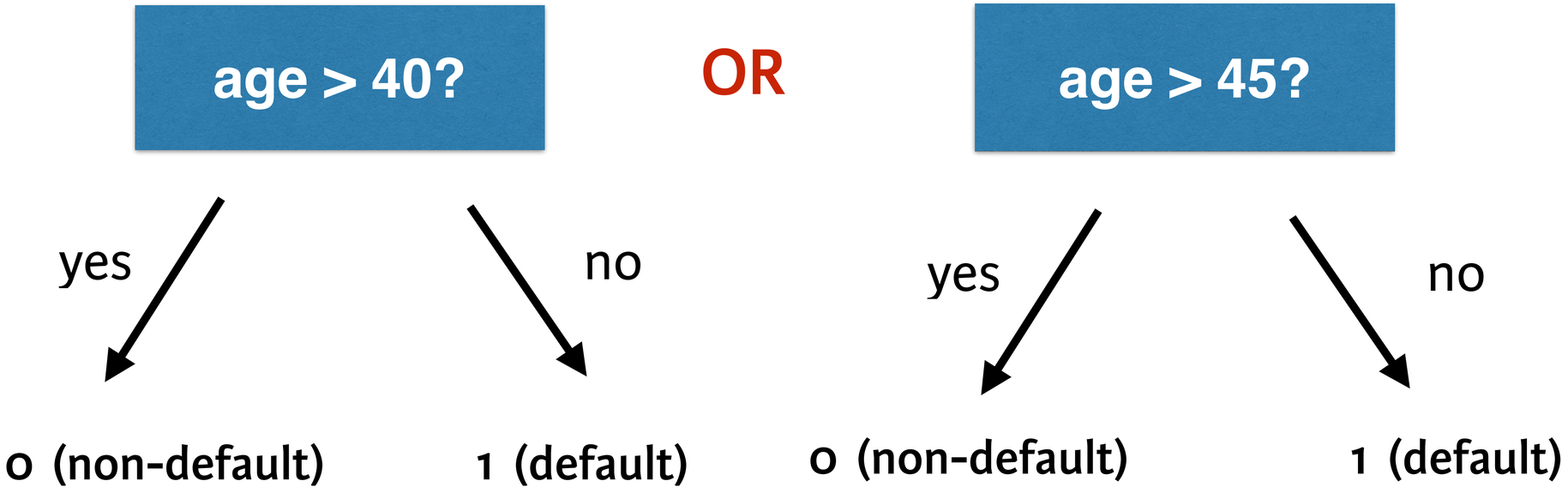
Defining a measurement for impurity!

We would like to minimize impurity for each node of the tree.

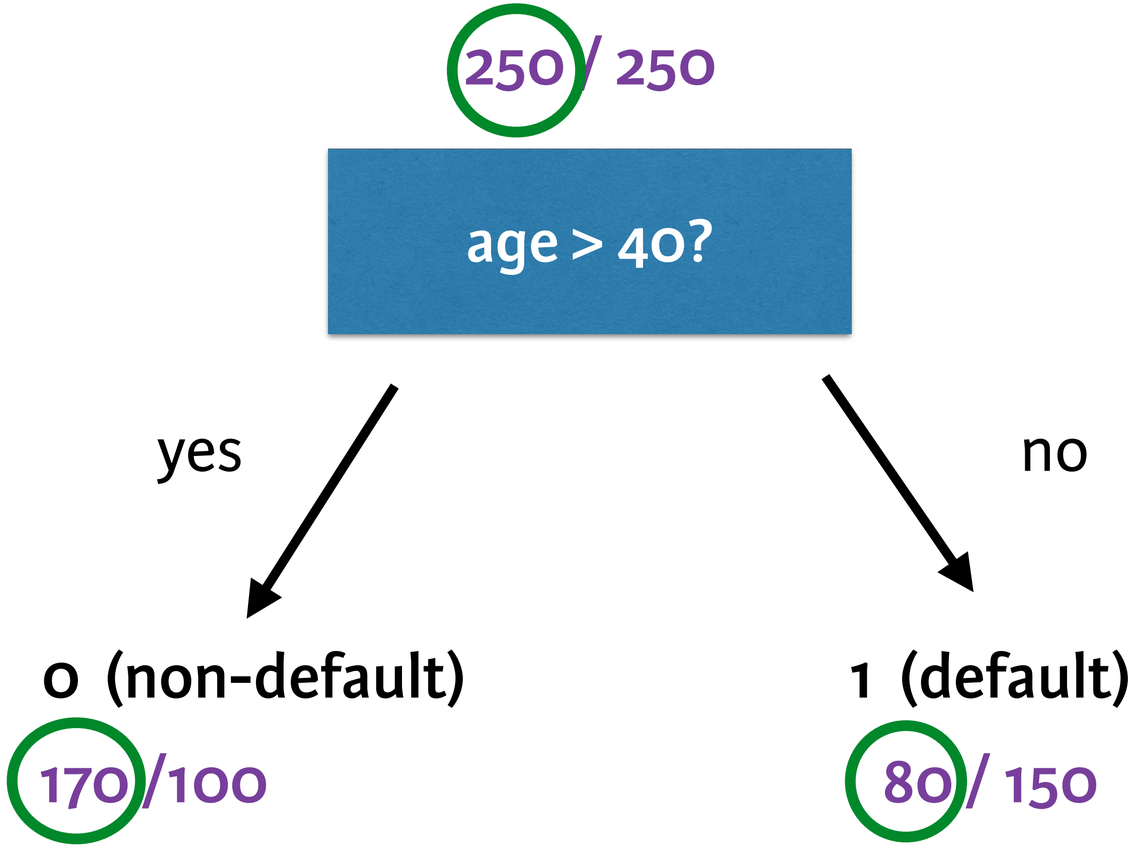
For instance, GINI measure.
Gini-measure is used to create the perfect split for a tree.

The ideal scenario is to perfectly split default and non-default.

How to make splitting decision?

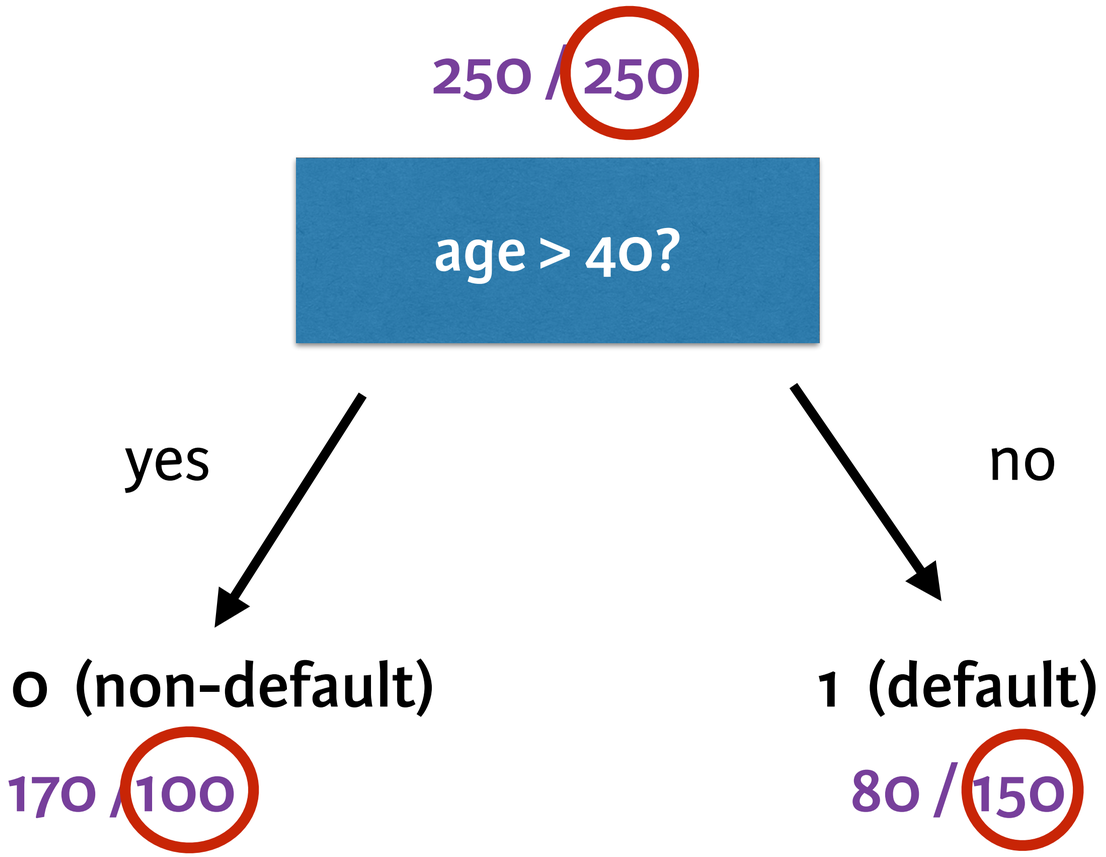


Example



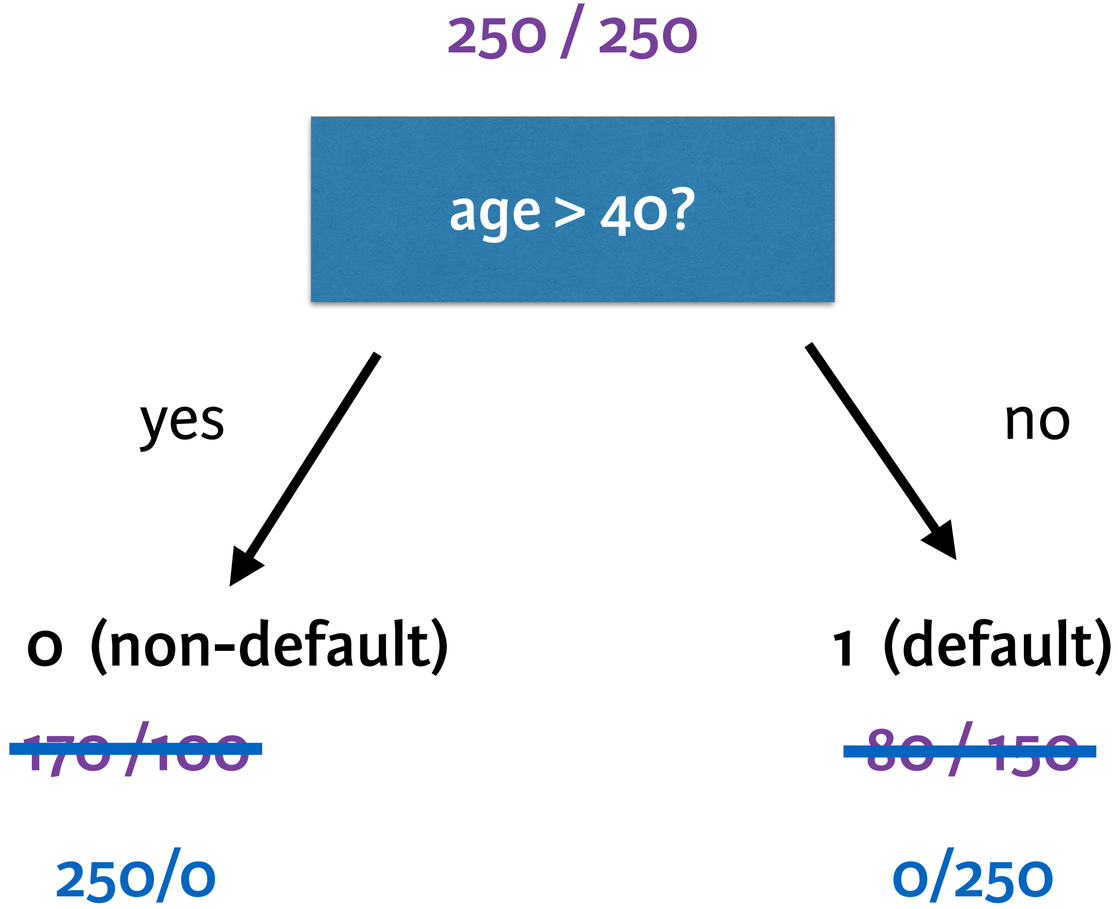
Actual **non-defaults** in this node using this split

Example



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

Example

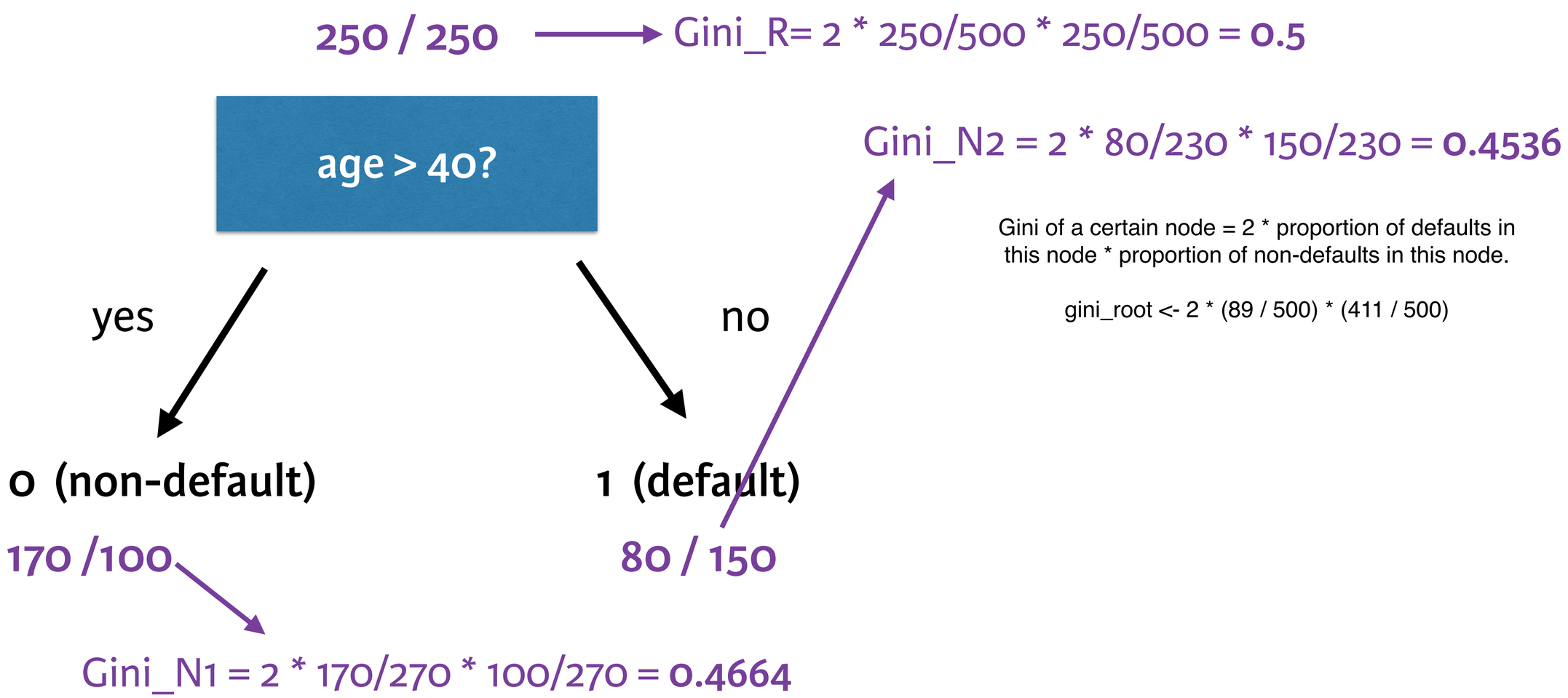


= IDEAL SCENARIO

Example

Computing the impurity in a node:
where, prop = proportion

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



Gini of a certain node = 2 * proportion of defaults in this node * proportion of non-defaults in this node.

```
gini_root <- 2 * (89 / 500) * (411 / 500)
```

Example

INFORMATION GAIN:
Calculate the GAIN per VAR and select the highest GAIN as the split.

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

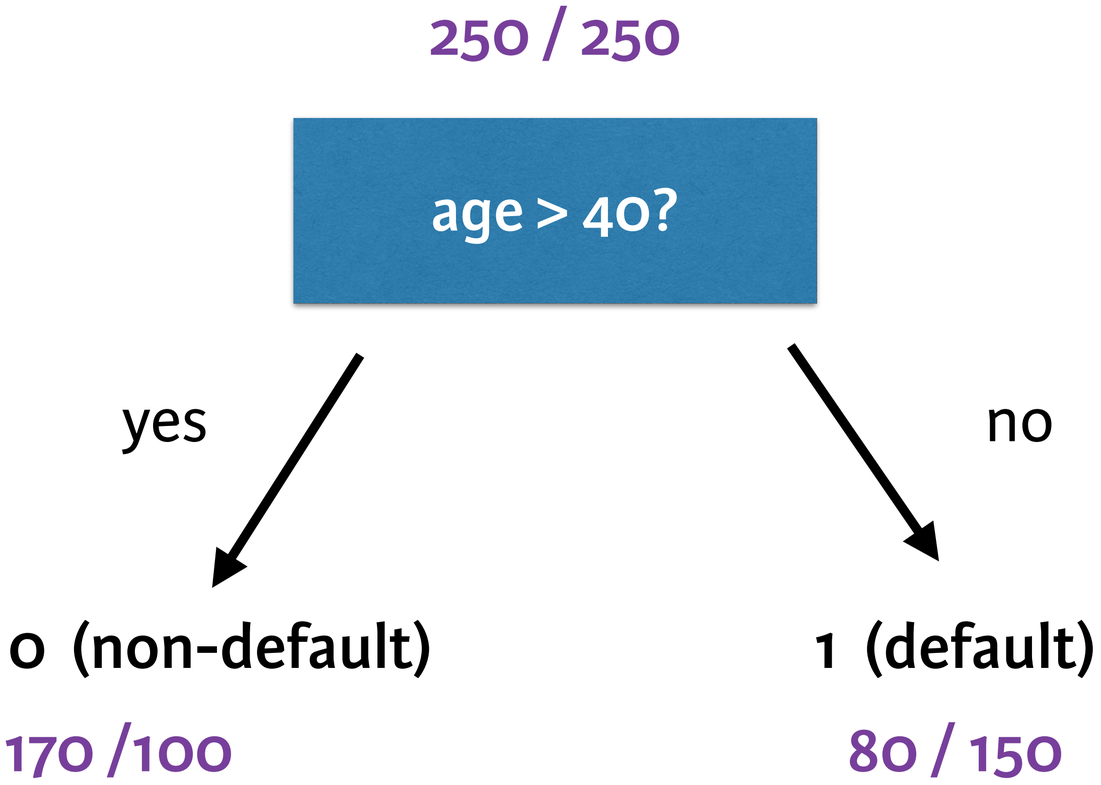
- prop(cases in N2) * **Gini_N1**

MAXIMIZE GAIN

= 0.5 - 270/500 * 0.4664

- 230/500*0.4536

= 0.039488



INFORMATION GAIN of a node:
Gain of the leaf nodes with respect to the root node.

Gain = gini_root - (prop(cases left leaf) * gini_left) - (prop(cases right leaf * gini_right))

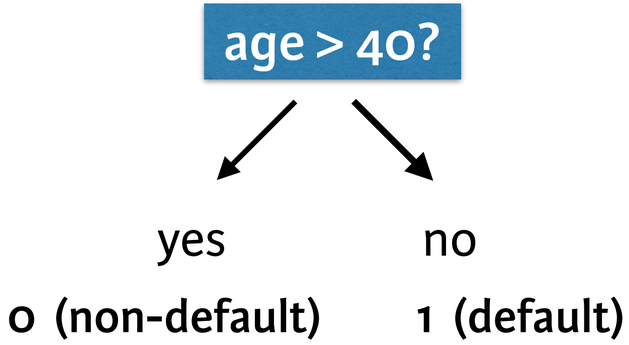
CREDIT RISK MODELING IN R

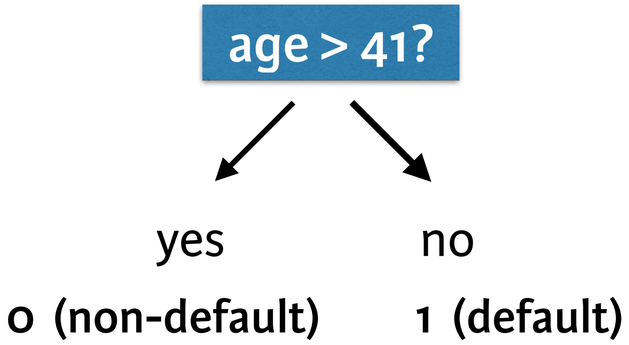
Let’s practice!

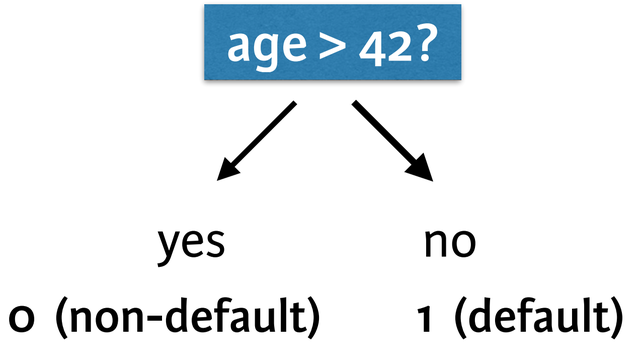
CREDIT RISK MODELING IN R

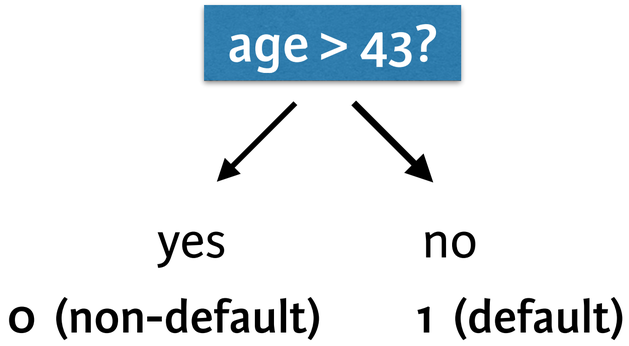
Building decision trees using the rpart()-package

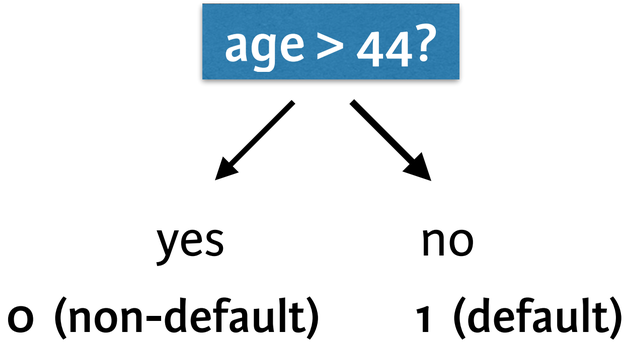
Imagine...


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!

Balancing the Training Set (Not the test set)

Oversampling under represented group = default
OR
Undersampling the over represented group = NON-default

EX: The training set has been undersampled for you, such that 1/3 of the training set consists of defaults, and 2/3 of non-defaults. The new dataset contains less observations (6570 instead of 19394)

Changing the prior probabilities for default bigger, attaching more importance to default.
This is an indirect way of adjusting the importance of misclassifications for each class.

Including a loss matrix: different cost can be associated with a missclasification of default as a non-default (increased, more attention to this) vs. a missclasification of non-default as default.

loss matrix, changing the relative importance of misclassifying a default as non-default versus a non-default as a default. You want to stress that misclassifying a default as a non-default should be penalized more heavily.

CREDIT RISK MODELING IN R

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

CP, which is the Complexity Parameter, is the threshold value for a decrease in overall lack of fit for any split. If cp is not met, further splits will no longer be pursued. cp's default value is 0.01, but for complex problems, it is advised to relax cp.

Print cp:
Get an overview of how the tree grows using more splits (nplit)
Xerror: cross validated error of the decision tree, we want the cp that minimizes this value (need to set a seed for reproducible results).

Pruning a tree is necessary to avoid overfitting.
And pruning a previously constructed tree with the changed prior probabilities.

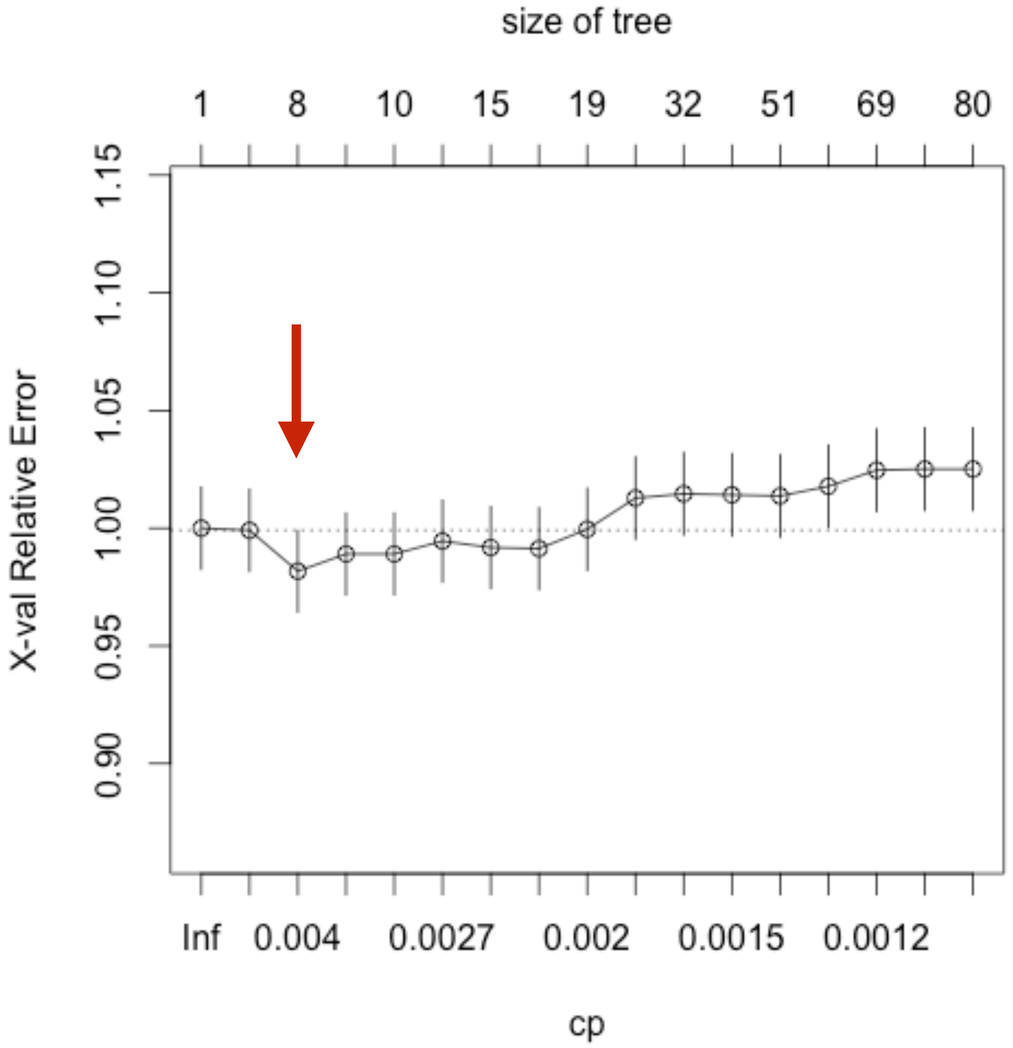
You will first set a seed to make sure the results are reproducible, because you will be examining cross-validated error results. Results involve randomness and could differ slightly upon running the function again with a different seed.

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
```

CP = 0.003653, generates the minimum Crossvalidation error

Plotcp and tree_undersample

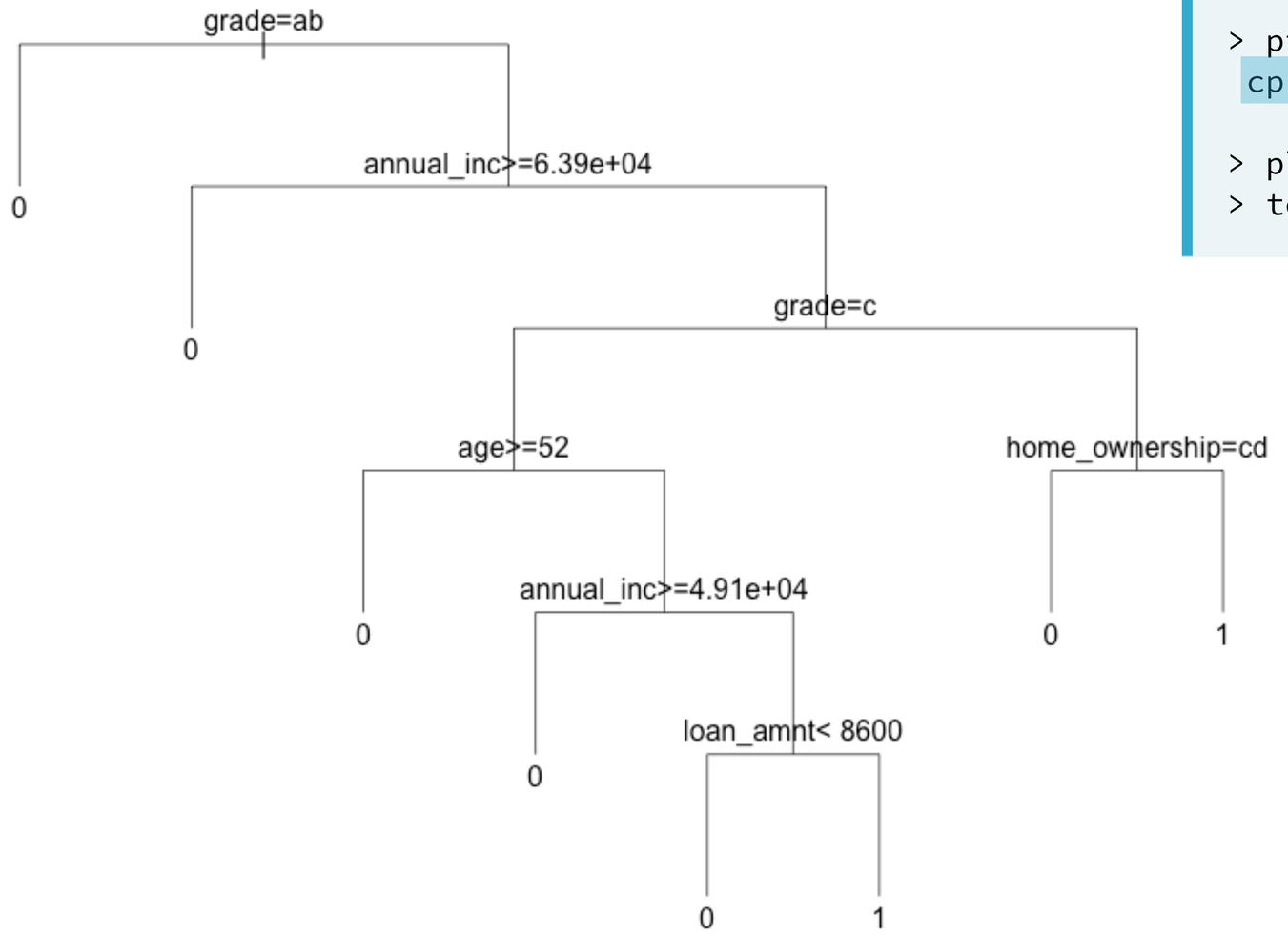


Plot the crossvalidation Error (xerror):

CP = 0.003653, generates the minimum Crossvalidation error

cp= 0.003653

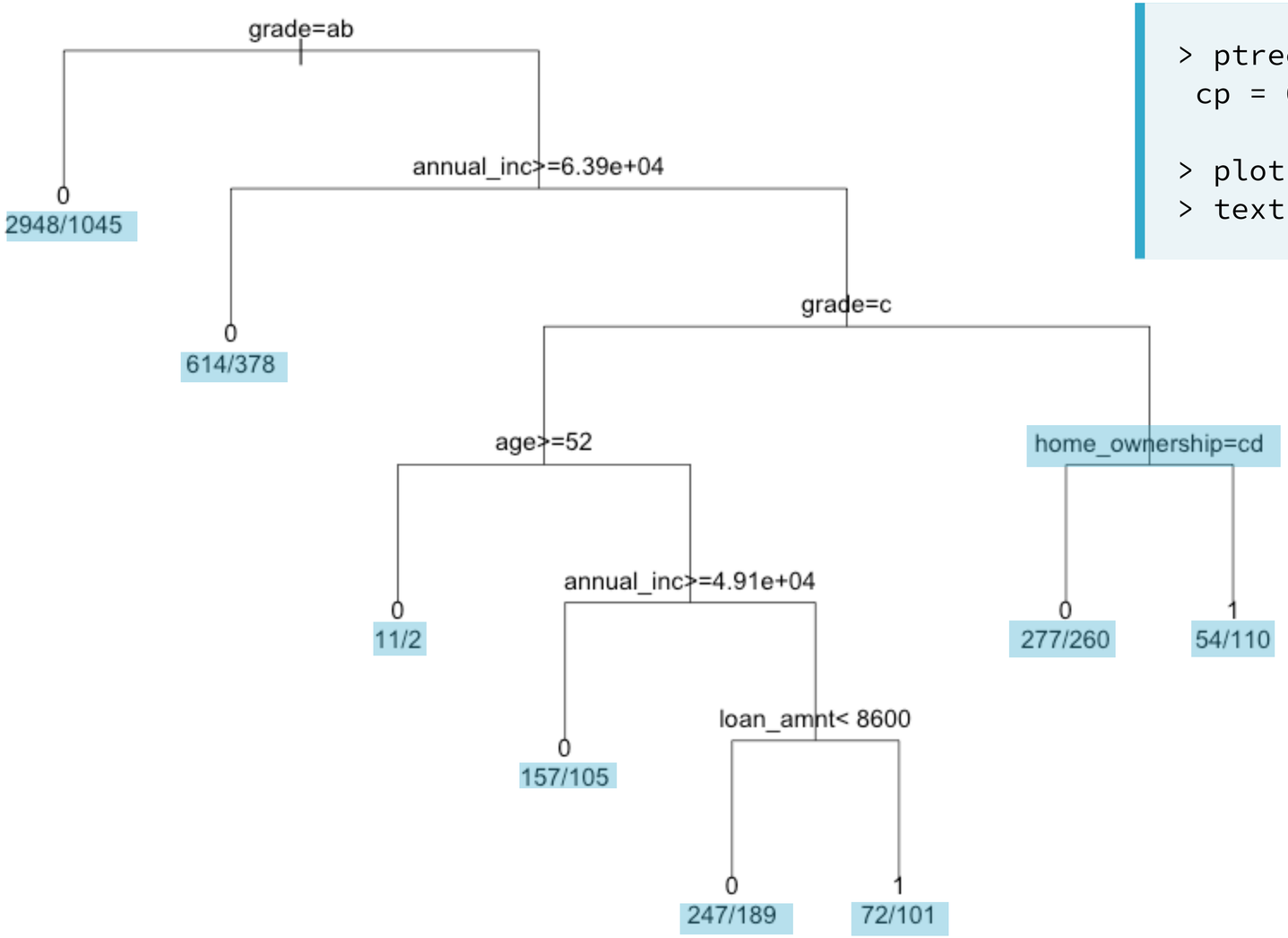
plot the pruned tree




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

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

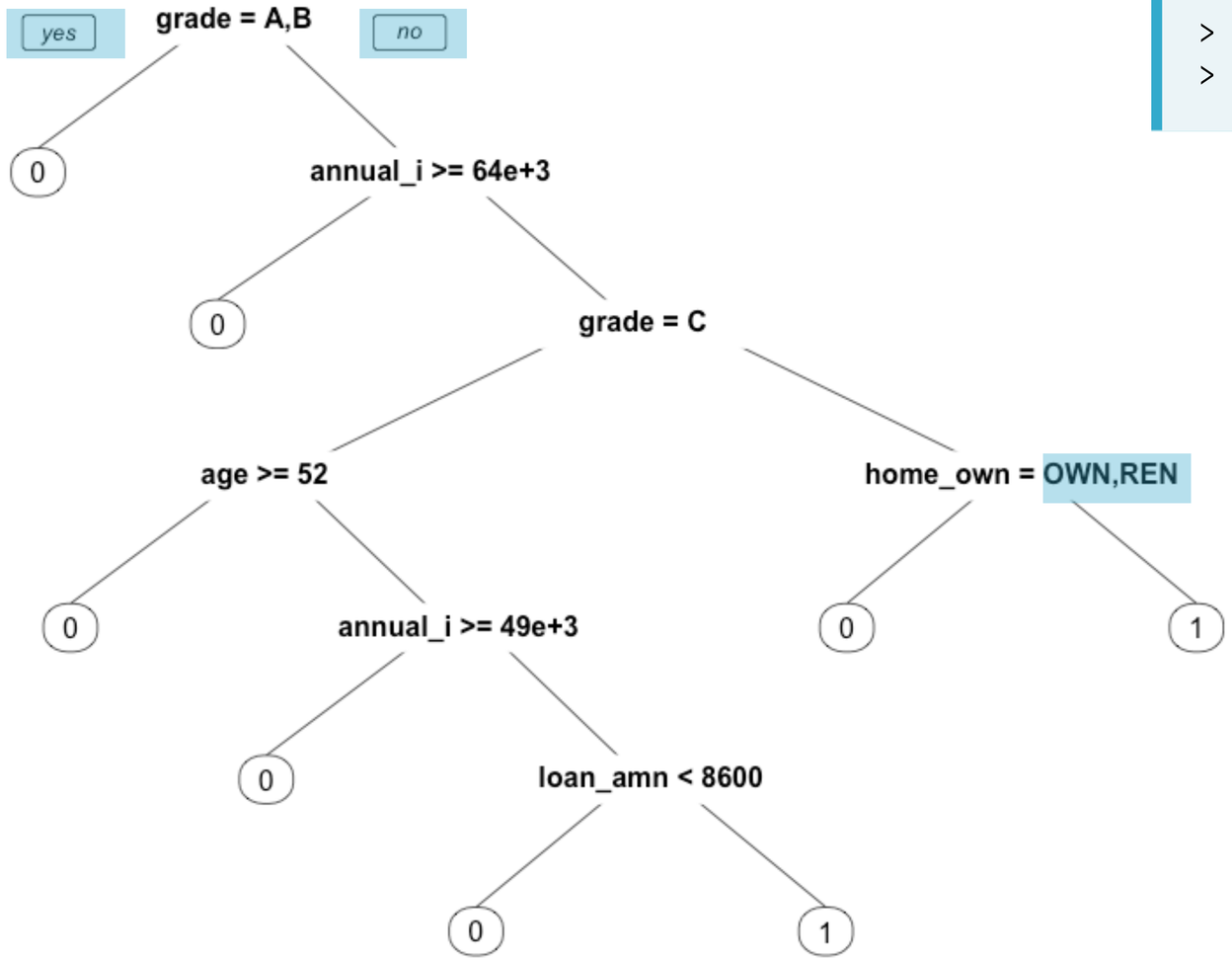
CP = 0.003653, generates the minimum Crossvalidation error

plot the pruned tree



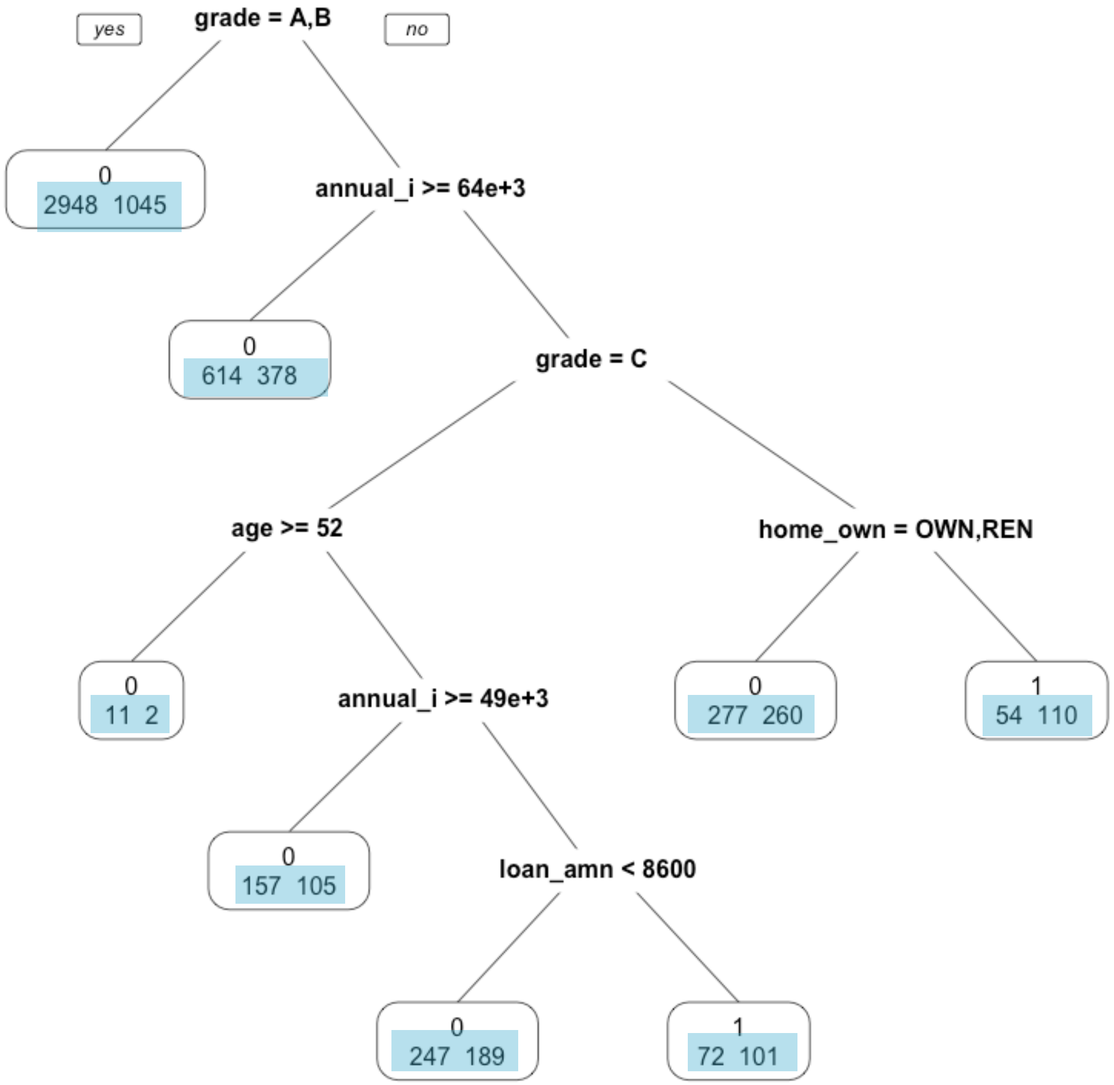

```
> ptree_undersample=prune(tree_undersample,
 cp = 0.003653)

> 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)
```

Extra = 1, return number of default and non-default cases

CREDIT RISK MODELING IN R

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

  Increasing the weight for the default cases would be another strategy to counter the unbalance in the decision tree

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

- `minsplit:` minimum number of observations for split attempt

  Min. number of obs in a node before splitting, default value is 20, for unbalance data could be lowered.

- `minbucket:` minimum number of observations in leaf node

  Default value is 1/3 of minsplit. Lowering this value to much could lead to overfitting

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!