**Non-linear methods: decision trees**

> evaluation

> als html speichern

Basic idea:

* Aim: final node where all samples belong to one category (= max purity)

The goal of the algorithm is to minimize the weighted sum of dissimilarity measures. At each node, the selected predictor is the one with the largest reduction of dissimilarity. Hence the tree is hierarchical, and produces a ranking of predictor importance

* Procedure: **split data** into subsets **based** on features providing highest information gain (increasing the purity) **to make** classification decision
* Root node, child node, end node
* Non-linear
  + capture **complex relationships** that involve **interactions and dependencies** between features
  + not a single equation like in regression, but **hierarchical structure of conditional statements** that leads to a final decision or prediction
  + while regression estimates the association based on given linear model, decision trees find the underlying pattern of the data (=structure of the tree) & thereby explain the association
* feature importance
  + **splitting features** relevant for making predictions
  + more **frequently used** features (in splitting) 🡪 more influential in making predictions
  + how much it **reduces impurity** (Gini impurity or entropy) or increases information gain
* multicollinearity & omitted variables
  + less of an issue bc:
    - make splits based on the values of **individual features independently**; without consideration of the other features
    - highly correlated features may **individually contribute to reducing impurity**, leading the algorithm to select either feature for splitting based on its own merit.

Questions:

* Two kind of tree: regression & classification
* Which criteria do we apply for splitting the tree?
  + Purity (gini; entropy): increase nr of samples belonging to one class in a node
* Why should we do pruning?
  + Reduce overfitting resulting from overly complex tree
* Why do we use ensemble learning methods (random forest, bagging, boosting)?
  + Trees can have high variance; prone to overfitting
  + Combining indiv learners improves generalizing ability

Exercise:

*REGRESSION TREE*

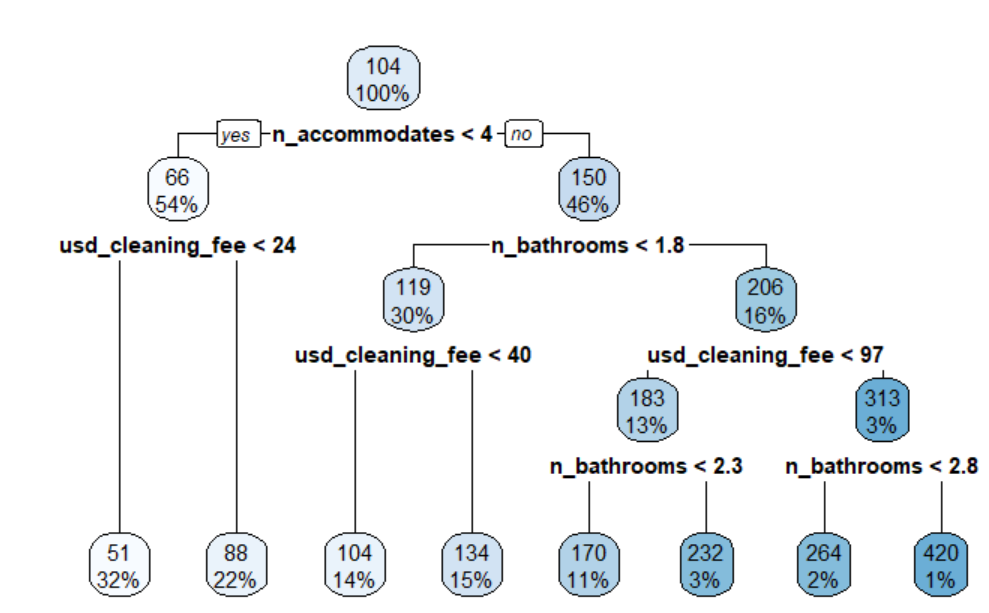
* n= **20634**

1) root **20634** 127935500 **104.27070 (= mean price) 🡪 root node**

2) n\_accommodates< 3.5 **11203 (=N = 54% of pop)** 15470560 **66.03240 (avg price)** **🡪 child node**

4) usd\_cleaning\_fee< 23.5 6585 3663398 50.57828 \* **🡪 end node**

5) usd\_cleaning\_fee>=23.5 4618 7991901 88.06908 \*



🡪 n\_accommodates most important factor in determining price

🡪 rpart automatically performs 10-fold cv to prune the tree; find optimal cost factor (which is increasing w/ number of terminal nodes)

* adding this option: control = list(cp = 0, xval = 10) shows uncut tree

*CLASSIFICATION TREE*

n= 14444

1) root 14444 6947 **0** (0.5190390 **0.4809610**) 🡪 **48% high rating**, thus = 0

2) p\_host\_response\_rate< 99.5 4469 1593 0 (0.6435444 0.3564556)

4) n\_number\_of\_reviews>=1.5 3856 1226 0 (0.6820539 0.3179461) \*

5) n\_number\_of\_reviews< 1.5 613 246 1 (0.4013051 0.5986949) \*

3) p\_host\_response\_rate>=99.5 9975 4621 1 (0.4632581 0.5367419)

6) n\_number\_of\_reviews>=4.5 6944 3419 0 (0.5076325 0.4923675)

12) d\_smokedetector< 0.5 914 318 0 (0.6520788 0.3479212) \*

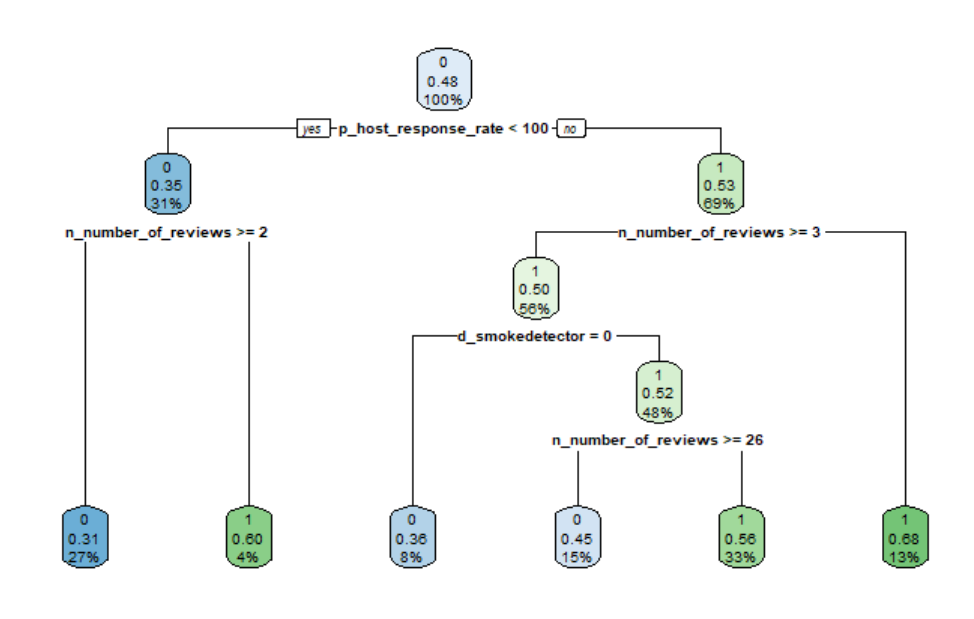
13) d\_smokedetector>=0.5 6030 2929 1 (0.4857380 0.5142620)

26) n\_number\_of\_reviews>=25.5 2250 1016 0 (0.5484444 0.4515556) \*

27) n\_number\_of\_reviews< 25.5 3780 1695 1 (0.4484127 0.5515873) \*

7) n\_number\_of\_reviews< 4.5 3031 1096 1 (0.3615968 0.6384032) \*

🡪 **6 end nodes**



* root node:
  + 48% of full data (100%) have high rating 🡪 <50% 🡪 low rating node
  + left = yes; right = no
  + blue = no, green = yes

🡪 of those having a response rate <100 & having more than 2 reviews, 31% have a high rating; making up of 27% of the entire population

*EVALUATION*

on **training** data:

* accuracy: 61.25
* precision (of all pos pred, how many correct): 63.53%
* recall (of all pos, how many pred): 59.49%

on **test** data:

* accuracy: 61.39
* precision (of all pos pred, how many correct): 64.86%
* recall (of all pos, how many pred): 58.88%

🡪 almost equally good

compared to **log reg** (on test data):

* accuracy: 53
* precision (of all pos pred, how many correct): 68.18%
* recall (of all pos, how many pred): 0.5%

🡪 very high number of false negatives

🡪 decision tree performs better

