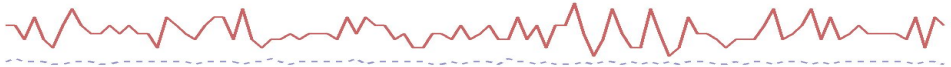
Supervised learning: Random Forest

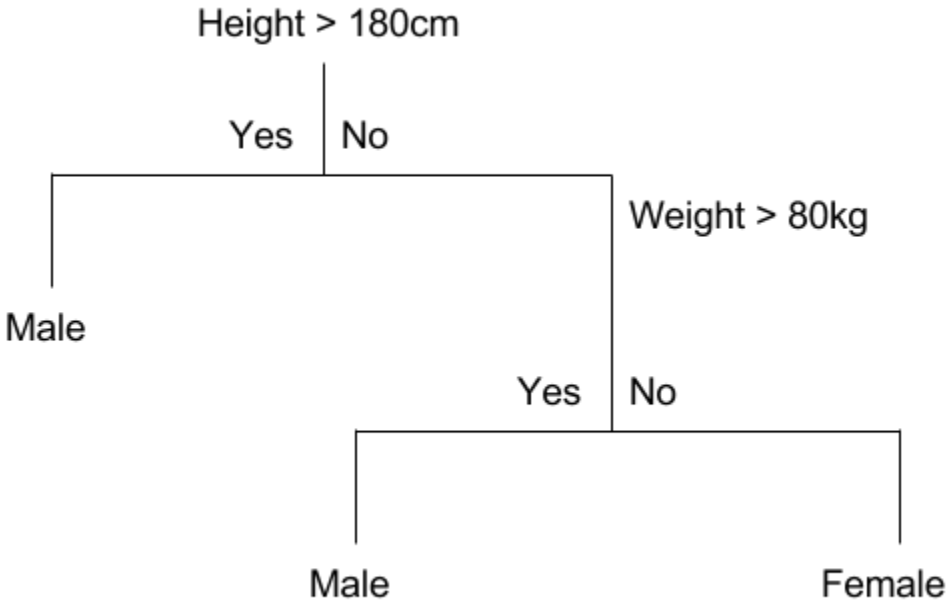
Filippo Biscarini (CNR, Milan, Italy)

filippo.biscarini@cnr.it

From decision trees to bagging



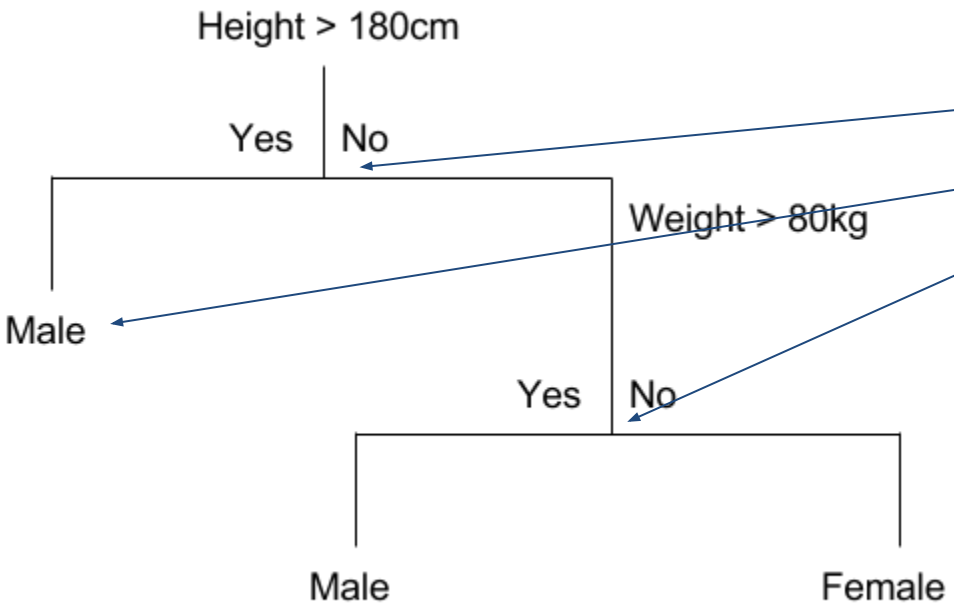
Decision trees



- tree-based methods are simple and useful for interpretation
- can be applied to classification and regression problems
- **high variance**, usually poor predictive performance
- combining a large number of trees can result in dramatic improvements in prediction accuracy
- **bagging**, **random forest** (these methods grow multiple trees which are then combined to yield a single consensus prediction)

From: https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/

Decision trees

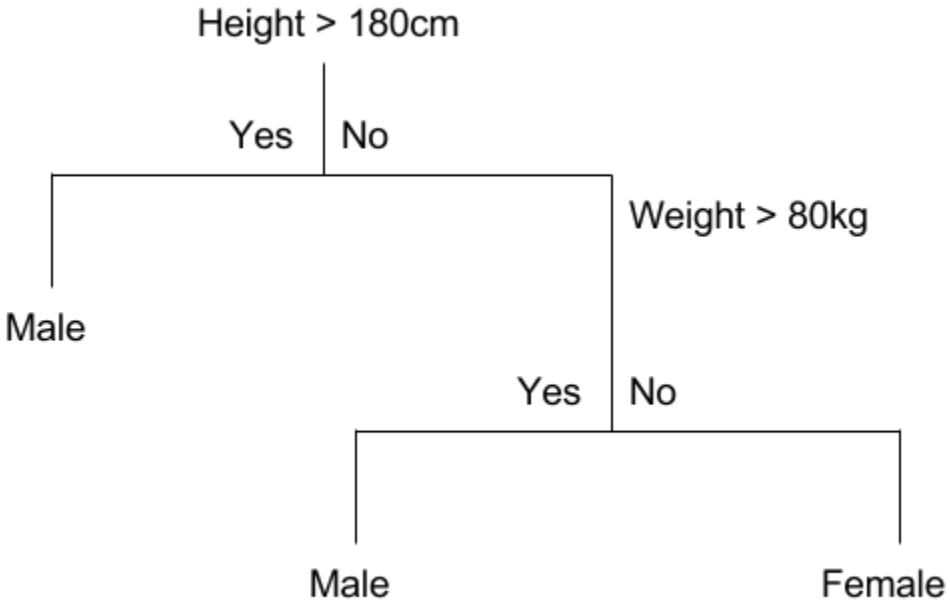


Terminology

- **branches**
- internal nodes
- **terminal nodes** (leaves)
- **splits**

From: https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/

Decision trees



How do we decide on splits?

- **min(RSS)** for regression trees
- **min(Gini index)** or **min(cross-entropy)** for classification trees

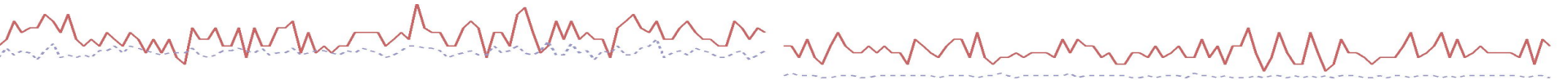
$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

- *K* classes
- $\mathbf{p_{mk}}$: proportion of obs from class *k* that go to branch *m*
- G is small if all of the p_{mk} are close to zero or one (node purity)

From: https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/

Decision trees

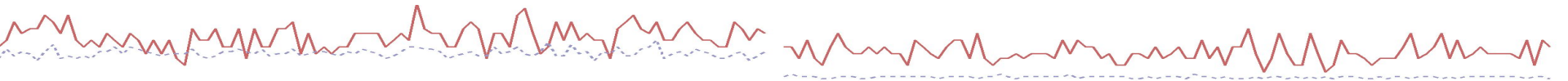
- recursive binary splitting: **top-down greedy algorithm** for tree-building
  - **top-down**: it begins at the top → first variable, first split
  - **greedy**: at each step the best split is made without considering next steps (splits) in the tree



Decision trees

- recursive binary splitting: **top-down greedy algorithm** for tree-building
  - **top-down**: it begins at the top → first variable, first split
  - **greedy**: at each step the best split is made without considering next steps (splits) in the tree

continue until stopping criterion is reached (e.g. n. of obs in terminal nodes, reduction in RSS / entropy < threshold etc.)



Decision trees - predictions

- once the tree is trained, **test observations** are run through it and **predicted** by using the **mean** (regression) or **majority vote** (classification) of **training observations** in **that region/node**

Decision trees - predictions

- once the tree is trained, **test observations** are run through it and **predicted** by using the **mean** (regression) or **majority vote** (classification) of **training observations** in **that region/node**
- trees may grow to be very complex → **overfitting!**

Decision trees - predictions

- once the tree is trained, **test observations** are run through it and **predicted** by using the **mean** (regression) or **majority vote** (classification) of **training observations** in **that region/node**
- trees may grow to be very complex → **overfitting!**

- high threshold as stopping criterion - risk of stopping too early ("short trees")
- **tree pruning**

From: https://www.canstockphoto.com/happy-family-in-garden-28321970.html

Decision trees - pruning

- grow a **very large tree**, **T0**
- then **prune it** to obtain a **subtree**
- many subtrees T ⊂ T0
- cost complexity pruning →

$$C = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{Rm})^2 + \alpha|T|$$

- |T|: n. terminal nodes in T
- x_i: i_{th} predictor
- R_m: subset of predictor space for m_{th} terminal node
- y_hat_{Rm}: mean of training obs in R_m
- choose T ⊂ T0 so that C is smallest

Decision trees - pruning

- grow a **very large tree**, **T0**
- then **prune it** to obtain a **subtree**
- many subtrees T ⊂ T0
- cost complexity pruning →

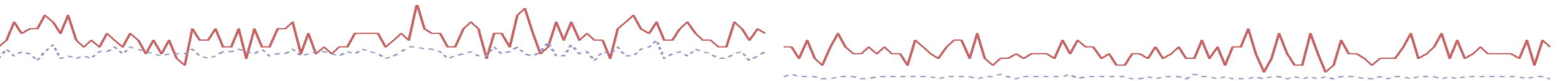
Tuning parameter that controls subtree complexity (trade-off with fit to the data)

$$C = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{Rm})^2 + \alpha |T|$$

- |T|: n. terminal nodes in T
- x_i: i_{th} predictor
- R_m: subset of predictor space for m_{th} terminal node
- y_hat_{Rm}: mean of training obs in R_m
- choose T ⊂ T0 so that C is smallest

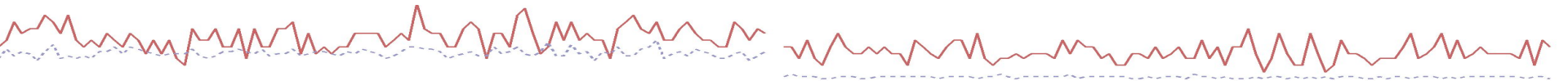
Decision trees - recap

1. recursive binary splitting to grow a large tree (training data)
2. cost-complexity for tree pruning
3. k-fold cross-validation to select α
   a. repeat steps 1 and 2 on the k-1 training folds
   b. evaluate prediction accuracy on the kth validation fold
   c. for each value of α, average over k validation folds
   d. select α that minimises the average error (max avg accuracy)
4. return T (from step 2) that corresponds to selected α

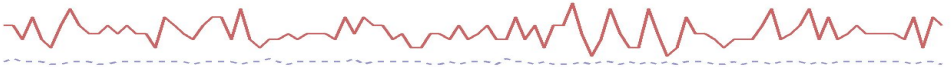


Pros and cons of decision trees

- easy to interpret and explain
- can be displayed graphically
- can easily handle qualitative predictors without the need to create dummy variables
- no need of feature normalization
- trees generally compare (very) poorly in terms of predictive accuracy (high variance)



A new resampling method

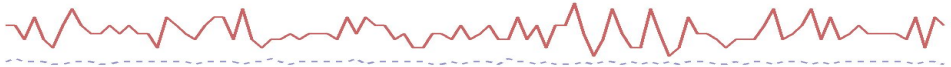


Bootstrapping

- A resampling method (**make do with what we have**)

“The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.”

[“The Surprising Adventures of Baron Munchausen” by Rudolph Erich Raspe]



Bootstrapping

- repeatedly sampling observations from the original data **with replacement**
- each “bootstrap set” comes from sampling with replacement and is the same size as the original data; some observations may appear more than once, while some not at all (**out-of-bag observations**)
- flexible and powerful technique to quantify the uncertainty associated with an estimator or machine learning method (e.g. estimate of the standard error of a coefficient, or a confidence interval around it)

Bootstrapping

obs	x	y
1	2.5	3.5
2	3.4	0.8
3	0.7	1.2
4	-1.8	-0.5

original data

B_1

obs	x	y
1	2.5	3.5
2	3.4	0.8
2	3.4	0.8
4	-1.8	-0.5

Θ_1

B_2

obs	x	y
4	-1.8	-0.5
2	3.4	0.8
1	2.5	3.5
4	-1.8	-0.5

Θ_2

B_3

obs	x	y
1	2.5	3.5
3	0.7	1.2
1	2.5	3.5
4	-1.8	-0.5

Θ_3

$$\Theta = \frac{1}{B} \sum_{b=1}^{B} \Theta_i$$

Bootstrapping

obs	x	y
1	2.5	3.5
2	3.4	0.8
3	0.7	1.2
4	-1.8	-0.5

original data

B_1

obs	x	y
1	2.5	3.5
2	3.4	0.8
2	3.4	0.8
4	-1.8	-0.5

Θ_1

B_2

obs	x	y
4	-1.8	-0.5
2	3.4	0.8
1	2.5	3.5
4	-1.8	-0.5

Θ_2

B_3

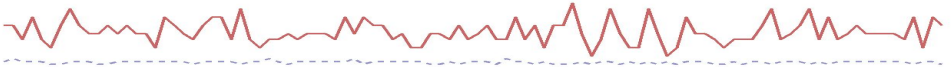
obs	x	y
1	2.5	3.5
3	0.7	1.2
1	2.5	3.5
4	-1.8	-0.5

Θ_3

$$\Theta = \frac{1}{B}\sum_{b=1}^{B}\Theta_i$$

You can also estimate the variability of Θ, and its distribution

Bagging and Random Forest

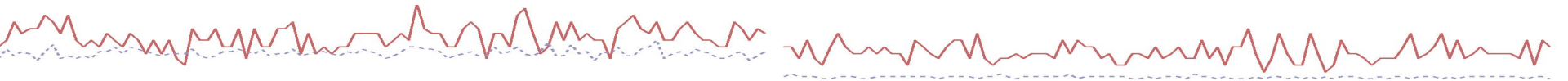


Bagging: bootstrap aggregation

- **bagging** is a general technique to **reduce the variance** of machine-learning methods

$$Var(\overline{x}) = \frac{\sigma^2}{n}$$

- averaging reduces the variance (the larger *n*, the larger the reduction)



Bagging: bootstrap aggregation

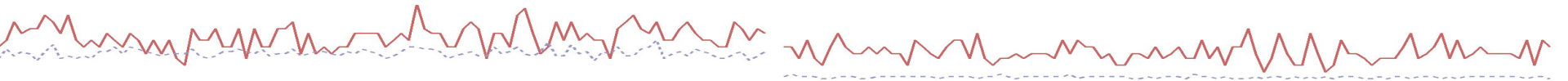
- we normally do not have multiple training sets, but we can use **bootstrapping** to generate them

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x)$$

This is bagging, folks!

Out-of-bag (OOB) observations

- In bootstrapping, typically ⅔ of the data are used in each bootstrap replicate → hence, ⅓ is left out → **out-of-bag observations**
- We can use OOB observations to **estimate the prediction error** (sort of built-in cross-validation)
- This is a valid estimate of the test error, since OOB observations are not used to train the decision tree

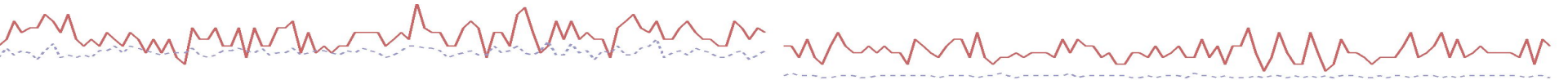


Random Forest

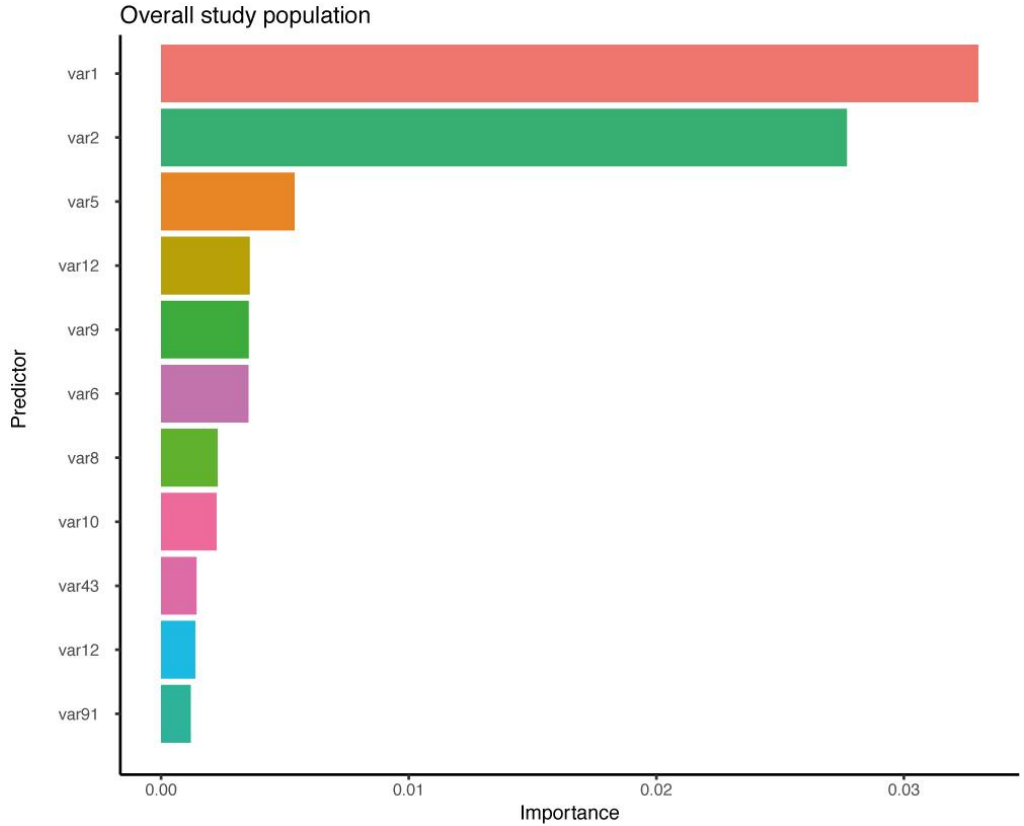
- **bagging** improves over decision trees, but the bootstrapped trees are correlated → overfitting
- **Random Forest** improves over bagging by decorrelating the bootstrapped trees
- this is done by selecting a **random subset of predictors** (variables) for each bootstrapped tree
- this way, trees in the forest differ in terms of **both observations** (sampling with replacement) **and variables** (random subset)
- typically, **~sqrt(p)** predictors are used for classification, **~p/3** predictors for regression → **tuning parameter**

Variable importance

- **bagging** and **RF** improve the predictive ability at the expense of model interpretation
- however, one can use the large number of trees to **extract variable importance**
- this is based on the amount that the RSS (or Gini index) is decreased due to splits over a given predictor, averaged over all B trees. A large value indicates an important predictor



Variable importance



source:https://stats.stackexchange.com/questions/329995/scale-of-variable-importance-in-randomforest-party-gbm

Variable importance

How do we calculate variable importance?

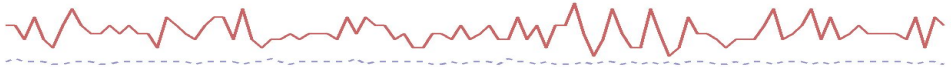
Simple answer:
In the case of RF regression, we can record the total amount that the RSS (residual sum of squares) is decreased due to splits over a given predictor, averaged over all B trees. A large value indicates an important predictor.

Detailed answer:
Once the RF has been trained (and trees with their nodes and splits have been constructed -using OOB observations or testing sets), we can measure the reduction of RSS: each variable is used at some nodes in each tree, and the RSS reduction over such nodes can be summed up and then averaged over the number of trees in the forest.

Alternative method:
Again, after trees have been constructed, OOB observations are passed through each tree twice: i) as they are; ii) after permuting randomly the value for each variable. In both cases the accuracy of prediction is measured. The reduction in the accuracy is then used as a measure of the variable importance (the larger the reduction, the more important the variable)

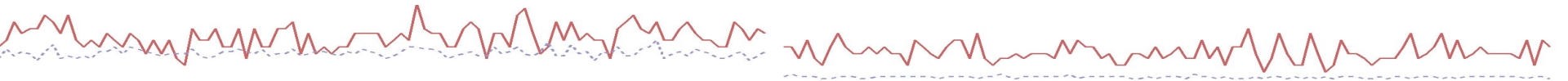
Variable importance - RSS reduction

- record reduction in RSS or node impurity over all splits involving any given variable
- default variable importance metric in scikitlearn
- average over thousands of trees in the “forest”
- usually rescaled [0,100]
- variables with large average RSS reduction are more important

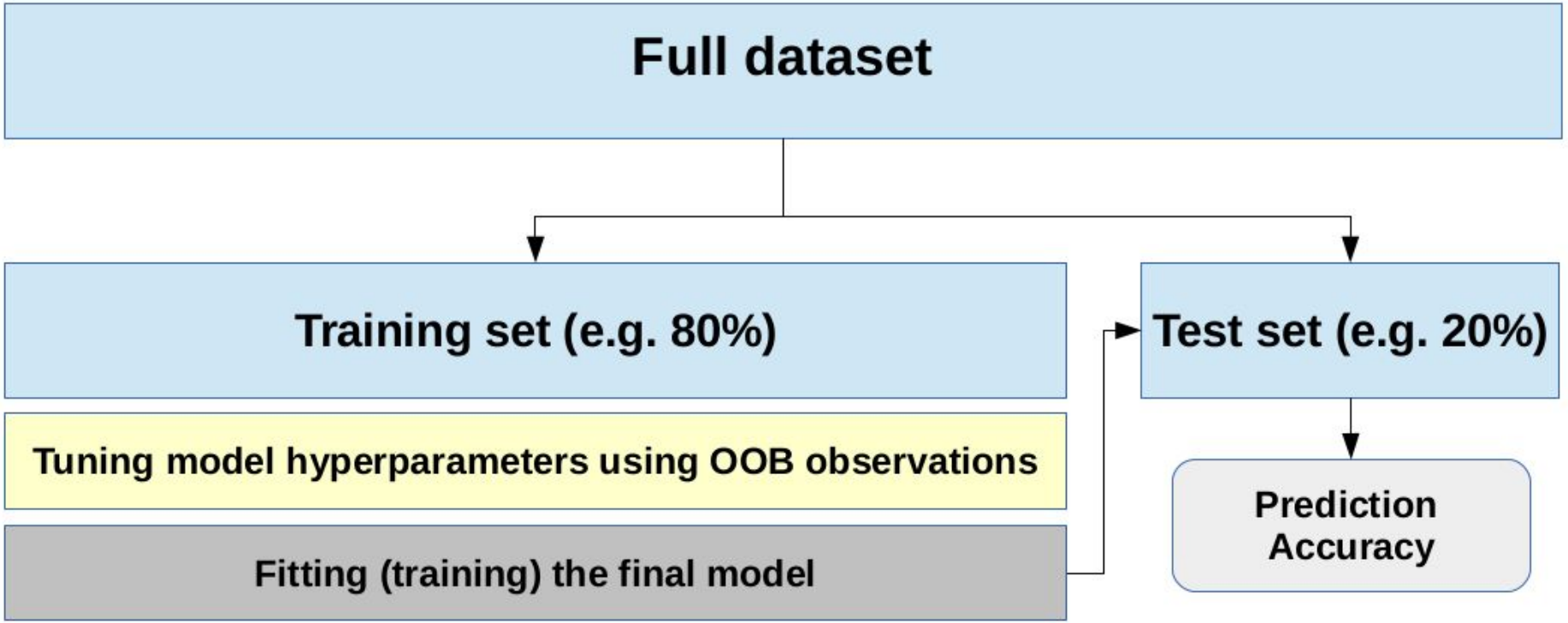


Variable importance - permutations

1. for **each tree** grown in a random forest, get number of **correct predictions** in **OOB data**
2. **random permutation** (**shuffling**) of a **predictor's values** in the **OOB data**, then check again the **number of correct predictions**
3. **subtract** the **number of correct predictions** in the **permuted data** from the **number of the correct predictions** in the **original OOB data**
4. the **average** of this number over **all trees** in the forest is the **variable importance** (can be normalized)
5. variables with **large differences between permuted and non-permuted data** are ranked as more important (a model without that variable gives worse predictions → the variable is important)



Model tuning



Random Forest

- demonstration 8.1
- Exercise 8.1

→ 8.random_forest.Rmd

→ 9.multiclass_random_forest.Rmd