Seminar Data Science for Economics

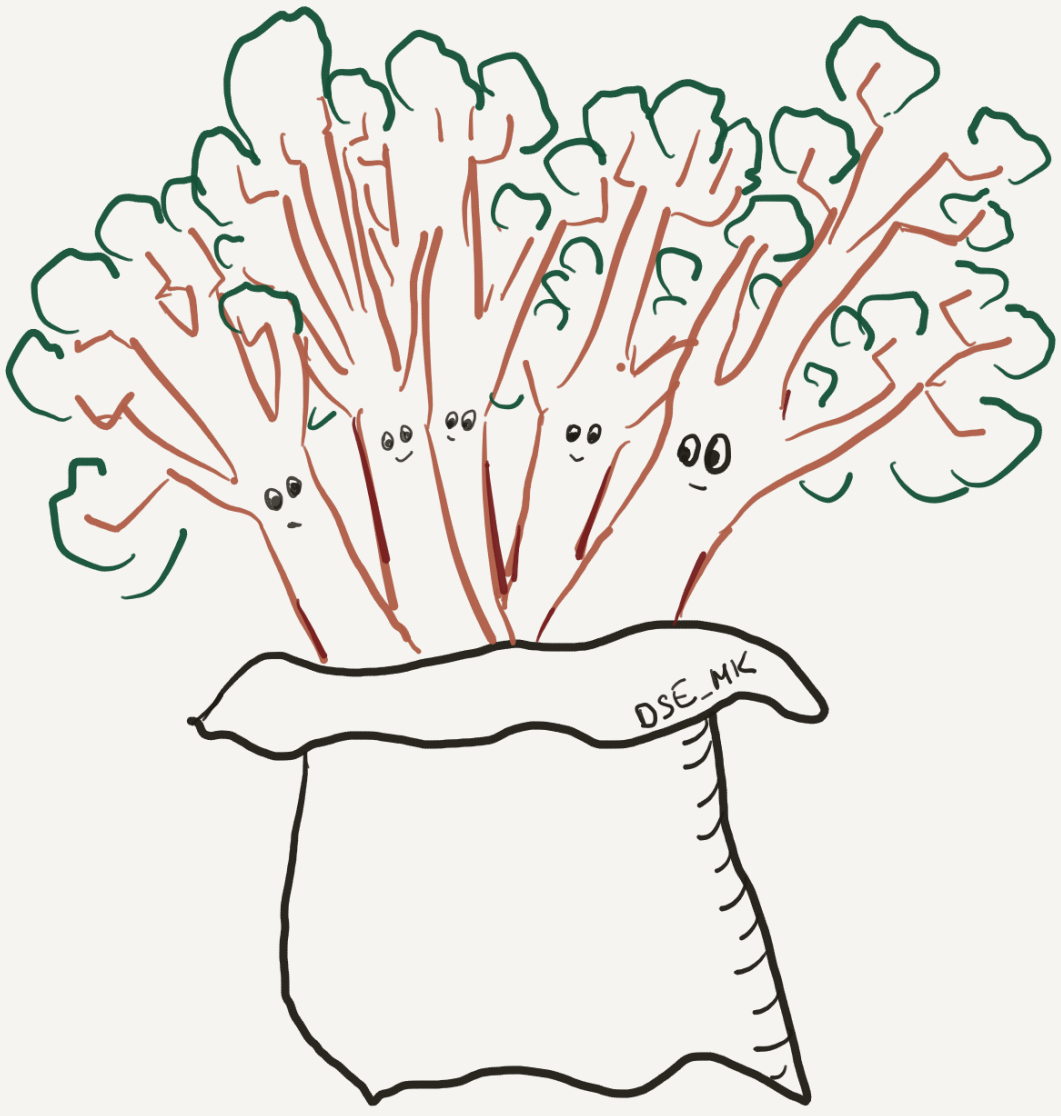
MSc. Economics program

Madina Kurmangaliyeva
m.kurmangaliyeva@uvt.nl

Spring 2020

Tilburg University

Bagging


DSE_MK

Bagging and Random Forest

Decision Tree: Low bias but High variance $\Longrightarrow$ Low prediction accuracy

Idea of bagging:

- Draw B (e.g., 1000) random subsamples from the training set (i.e., bootstrap – you saw this technique with Jan)
- Grow a decision tree for each subsample, in total B decision trees
- For any (new) obs. X, a bagged prediction is simply
  - the **average** of all B predictions for regression trees:
    $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B}\hat{y}^i(X)$
  - the **majority vote** by all B predictions for classification trees:
    $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$

Bagging for accuracy

By averaging over many low bias but high variance models, we reduce the overall variance $\Longrightarrow$ more accurate model.

Bagging is also called **bootstrap aggregation**, and can be applied to improve other models, e.g., OLS.

Note, however, we no longer can represent the resulting model graphically as a tree, i.e., the bagged model is less interpretable.

Random Forest

Same boosting, but now:

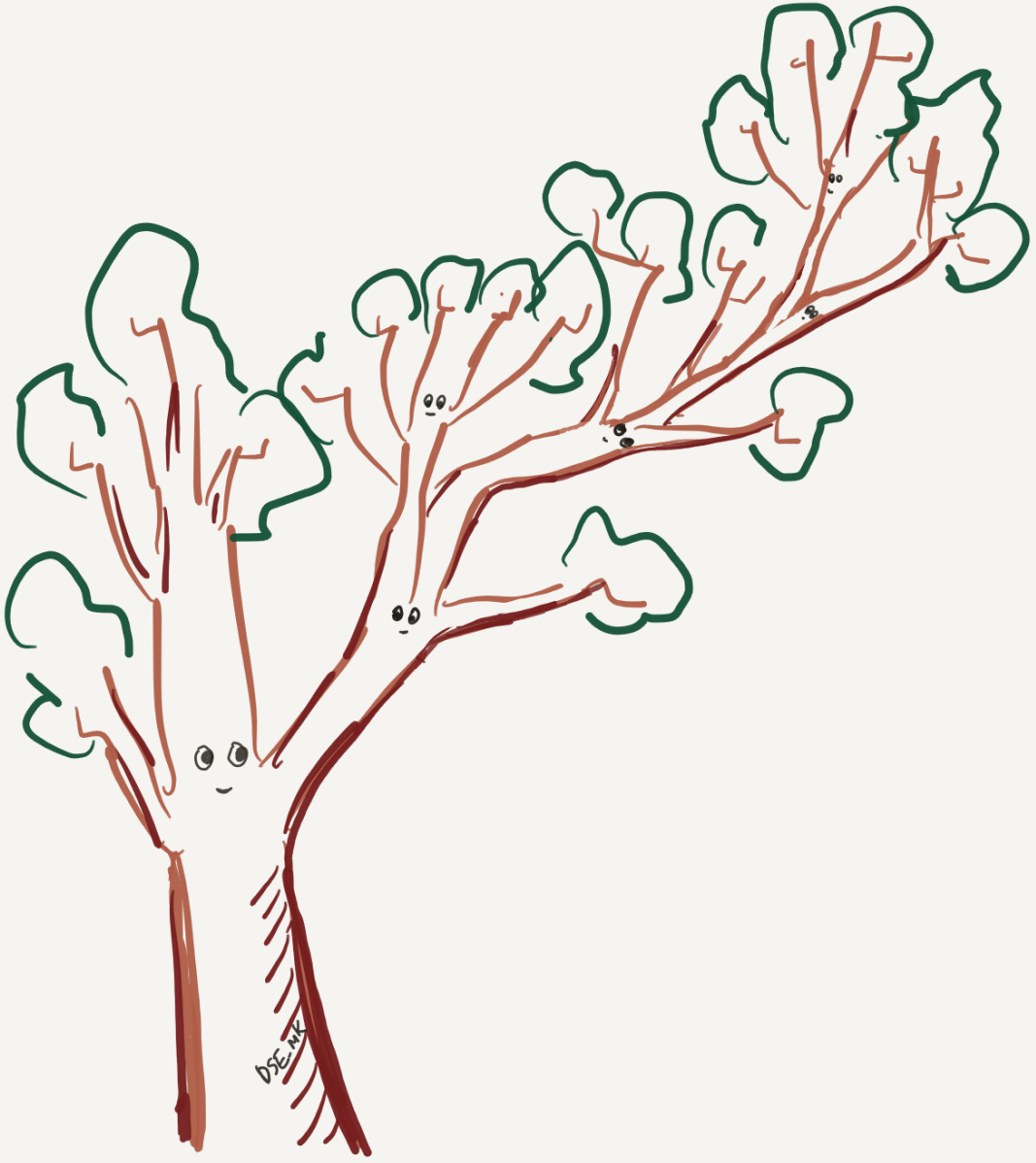
- At each split, only a **random sample of m predictors** is considered for splitting the node.
- Typically, m is set to $\sqrt{p}$. For example, if you have 100 predictors, then you would allow the random forest to consider only 10 random predictors at each split.

Why Random Forest?

Random forest is especially helpful when there is one strong predictor or a set of predictors that are highly correlated with each other.

By randomly removing some predictors at each split, we reduce the correlation between different trees across bootstrapped samples $\Longrightarrow$ reduce variance $\Longrightarrow$ more accurate model.

Boosting


DSE_MK

Boosting

Boosting will grow decision trees sequentially (iteratively):

- Each new decision tree will **"attack" the residuals** (unexplained errors) of the previous tree
- Each tree is usually **shallow**
- But there are **many of them** and each has a chance to improve over the previous models in its own turn
- Hence, boosting is a **slow learning** procedure, which accumulates the "wisdom" of many trees

Think about all those trees as little ants: each is small and powerless, but their strength is in numbers and joint attack on one goal!

Boosting for regression trees (Algorithm 8.2 from ISLR)

1. Set predictions to zero, $\hat{y}(X) = 0$, and residuals to y, $r_i = y_i$ for all i in training set.
2. For $b = 1, 2, \ldots, B$ repeat:
   a) Fit a tree $\hat{y}^b$ with d splits (i.e., $d + 1$ terminal nodes) to the training data (X, r) [Note: you do not pass y, you pass residuals]
   b) Update predictions over all domain of x by adding a shrunken prediction of the new tree:

$$\hat{y}^{new} = \hat{y}^{old} + \lambda \hat{y}^b(x) \tag{1}$$

   c) **Update the residuals** by deducting a shrunken prediction of the new tree:

$$r_i^{new} = r_i^{old} - \lambda \hat{y}^b(x_i) \tag{2}$$

3. Output the **boosted model**

$$\hat{y}(x) = \sum_{b=1}^{B} \lambda \hat{y}^b(x) \tag{3}$$

Free parameters

In the algorithm for boosting there are three free parameters:

1. The **number of trees** B. Choosing too high B may lead to overfitting (unlike in bagging or RF) $\Longrightarrow$ need to cross-validate this parameter
2. The **shrinkage parameter** λ, usually between 0.001 and 0.01. It controls the speed of learning. Smaller λ requires higher B.
3. The **number of splits** d, controls interaction depth. Usually, $d = 1$ works very well.