The main parameters to adjust when using these methods is n\_estimators and max\_features. The former is the number of trees in the forest. The larger the better, but also the longer it will take to compute. In addition, note that results will stop getting significantly better beyond a critical number of trees. The latter is the size of the random subsets of features to consider when splitting a node. The lower the greater the reduction of variance, but also the greater the increase in bias. Empirical good default values are max\_features=1.0 or equivalently max\_features=None (always considering all features instead of a random subset) for regression problems, and max\_features="sqrt" (using a random subset of size sqrt(n\_features)) for classification tasks (where n\_features is the number of features in the data). The default value of max\_features=1.0 is equivalent to bagged trees and more randomness can be achieved by setting smaller values (e.g. 0.3 is a typical default in the literature). Good results are often achieved when setting max\_depth=None in combination with min\_samples\_split=2 (i.e., when fully developing the trees). Bear in mind though that these values are usually not optimal, and might result in models that consume a lot of RAM. The best parameter values should always be cross-validated. In addition, note that in random forests, bootstrap samples are used by default (bootstrap=True) while the default strategy for extra-trees is to use the whole dataset (bootstrap=False). When using bootstrap sampling the generalization error can be estimated on the left out or out-of-bag samples. This can be enabled by setting oob\_score=True.