Classification trees, bagging, random forests

Žiga Trojer - 63200440

Overview of functions and parameters

I implemented classification trees, bagging and random forests. Those are implemented as classes, that provides a method build, which returns the model as an object. Method predict returns the predicted target class of a given input samples. Tree is a flexible classification tree with the following attributes:

- rand random generator for reproducibility,
- get_candidate_column a function, that returns a list of column indices considered for a split,
- min_samples the minimum number of samples, where a node is still split.

Bagging with attributes

- rand random generator,
- tree_builder an instance of Tree used internally,
- n number of bootstrap samples.

RandomForest with attributes

- rand random generator,
- n number of bootstrap samples,
- min_samples the minimum number of samples, where a node is still split.

In the above classes, Gini index was used for selecting the optimal split.

Results

Classification tree

When building a tree with function hw_tree_full and parameter min_samples = 2, we get misclassification rates:

- train set: 0.00
- test set: 0.15

It is obvious the misclassification rate on the train set is 0, because the tree perfectly splits a set until there is a leaf with only one data point in it.

Classification tree and cross validation

Function hw_cv_min_samples with 5-fold cross-validation on training data, gives us the best value of min_samples = 10 (we tested min_samples $\in \{2, 3, \ldots, 50\}$) A lot of misclassification rates were the same, especially for min_samples above 35 . Misclassification rates are:

- train set: 0.045
- test set: 0.16

It is very interesting that we got higher misclassification for the "optimal choice of min_samples". The reason is probably that we had luck with the tree with min_samples = 2. Plot 1 shows misclassification rates for train and test data for different choices of min_samples. We averaged rates from CV for each choice of min_samples. Then we searched for the lowest rate and got the result above. The minimum of test rate is nicely seen from the graph.

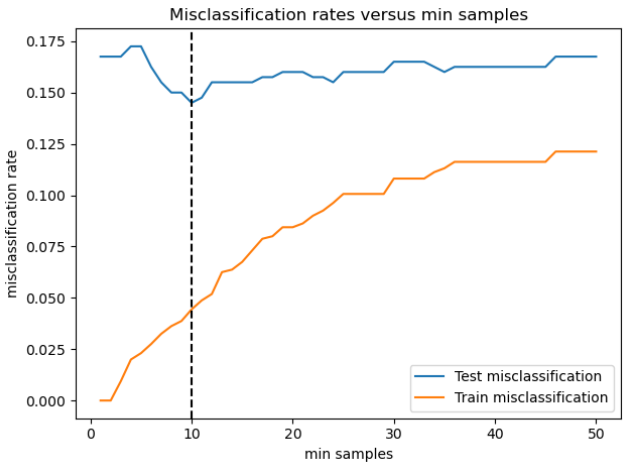

Figure 1. Train and test misclassification rate versus min_samples parameter. Vertical line represents optimal choice of min_samples. Rates starts to stagnate after choice of min_samples above 35.

Bagging

We used parameters n = 50 and min_samples = 2. Misclassification rates are:

- train set: 0.0025
- test set: 0.1

Plot 2 shows test misclassification for different number of trees used for bagging. Also, we used 3 different random

seeds. It is interesting that the choice of a random seed influences the optimal choice of number of trees. Because we don't have a large dataset, choice of large number of trees is not optimal. Misclassification rate stagnates above some choice of number of random trees.

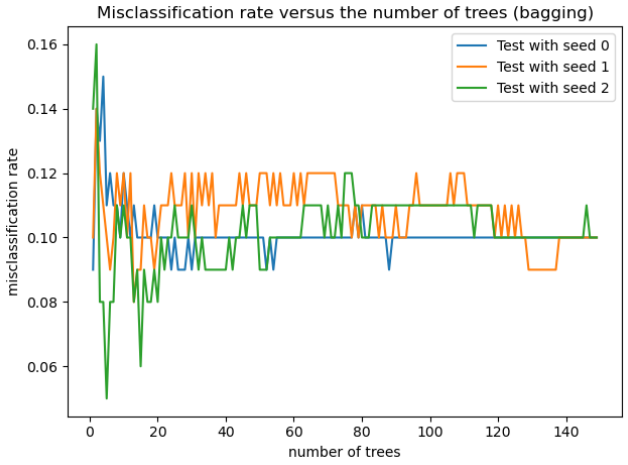

Figure 2. Test misclassification rate versus number of tree parameter for bagging, min_samples = 2 was used. Each line represents rate for different choice of random seed. The lowest rate is achieved with the third random seed and with less that 10 used trees - accuracy on test is around 95 %. When increasing number of trees, misclassification rate stagnates (with some jumps).

Random forest

We used parameters n = 50 and min_samples = 2. Misclassification rates are:

- train set: 0.0
- test set: 0.11

Plot 3 shows test misclassification for different number of trees used for random forest. Here too, different random seeds were used. Test rate with seed 0 stagnates from the number of trees = 35 on. This is probably due to the fact, that our dataset is not big and after some time, the choice of larger number of trees does not affect the rate.

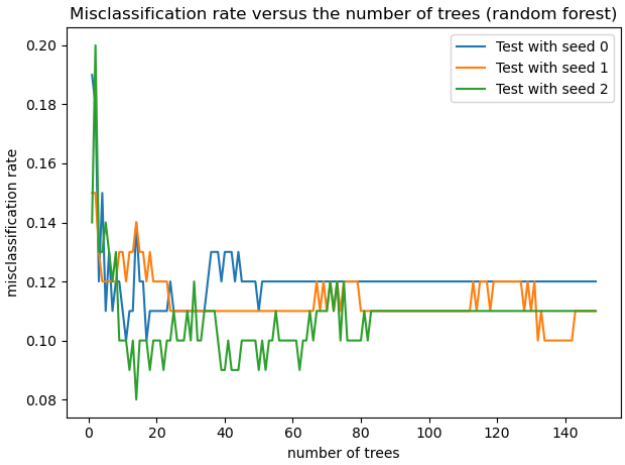

Figure 3. Test misclassification rate versus number of tree parameter for random forest, min_samples = 2 was used. Each line represents rate for different choice of random seed. The lowest rate is achieved with the third random seed and with around 15 used trees - accuracy on test is around 92 %. When increasing number of trees, misclassification rate stagnates (with some minor jumps).