***Decision trees***

Applied to both regression and classification

Uses splitting rules on each level - </> X and branches on each side. Can grow with many branches. At the end – reach LEAVES (terminal nodes), internal points are called INTERNAL NODES.

The predicted response for each group is the average of all values that are part of it.

The higher the NODE in a tree -> the higher the importance and the information gain.

Greedy method – because when splitting, it is taken into consideration only the current best option, without caring for possible better and more optimal options considering some further splits.

The process continues until a stopping criterion is reached; for instance, we may continue until no region contains more than five observations.

In order to avoid overfitting, we fit a VERY LARGE TREE and then PRUNE IT, in order to obtain a subtree.

For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs.

We use Gini Index – shows the node PURITY – The lower the value the better (more observations from a single class)

Linear Regression vs Dec. Trees – depends on the real relationship between the response and predictors, if linear – regression is better, if not – trees may be better. Also Trees are chosen for easier INTERPRETABILITY AND VISUALISATION.

By aggregating many decision trees, using methods like bagging, random forests, and boosting, the predictive performance of trees can be substantially improved.

***BAGGING***

Bootstrapping method for Decision trees – to reduce variance (in general averaging a set of observations reduces variance). We take many different samples from the training set and train the model on each of them (construct a different tree for each) and AVERAGE all the predictions. The trees ARE NOT PRUNED => with high variance, but low bias, and after averaging – the variance is reduced.

For Classification problems – TAKE MAJORITY VOTE – the most commonly occurring class among the N predictions.

The number of trees B is not a critical parameter with bagging; using a very large value of B will not lead to overfitting

there is a very straightforward way to estimate the test error of a bagged model, without the need to perform cross-validation or the validation set approach. One can show that on average, each bagged tree makes use of around two-thirds of the observations. The remaining one-third of the observations not used to fit a given bagged tree are referred to as the out-of-bag (OOB) observations.

We predict the response for i for each of the trees, in which the observation was OOB, afterwards for a single prediction for observation *i* we average the values./majority vote for classification

Bagging improves prediction accuracy at the expense of interpretability.

SUMMARIZED - bagging involves creating multiple copies of the original training data set using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in order to create a single predictive model

**RANDOM FORESTS**

Improvement over bagging – because in bagging each time the tree is split using the same amount of m predictors. This way if there is one very strong predictor, it affects the trees and every time it starts with it on the top split and therefore will look all the same at the end.

Random Forests – improve this by taking a **sample s of all the predictors** every time a tree is created on the bootstrapped sample.

The number of predictors considered at each split is approximately equal to the square root of the total number of predictors

***BOOSTING***

Boosting works in a similar way as bagging, except that the trees are grown sequentially: each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set.

Boosting – LEARNS SLOWLY – in general slow methods perform better.

We fit a tree using the residuals from the previous model. Then we add the new dec. tree to the fitted function and update the residuals. Then fit again and etc.

Most significant difference with bagging – THE CONSTRUCTION OF EACH TREE DEPENDS STRONGLY ON THE TREES THAT HAVE ALREADY BEEN GROWN.

Number of splits d in each tree works well to be = 1 (each additional tree is a stump). So each additional tree involves at most d=1 variables.

Unlike bagging and random forests, boosting CAN overfit, if number of trees is too large.