TREE-BASED METHODS

Data Analytics and Machine Learning

Introduction to tree-based methods

What are tree-based methods?

Tree-based methods represent a family of supervised Machine Learning methods which performs classification and regression tasks.

Why are they called trees?

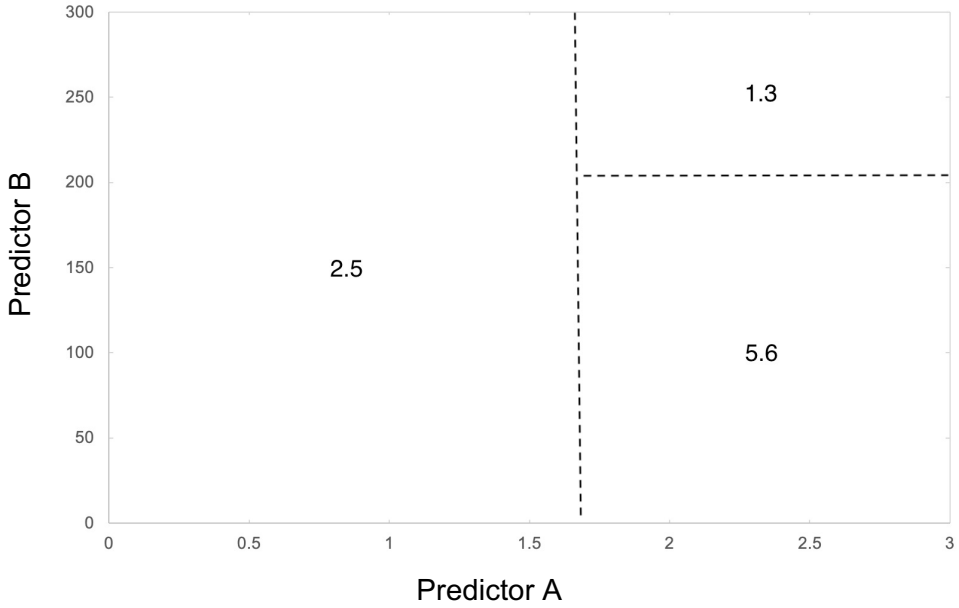
They are called tree-based methods because a tree-like structure (a decision tool called Decision Tree) is used for predicting the target variable class or value according to the features.

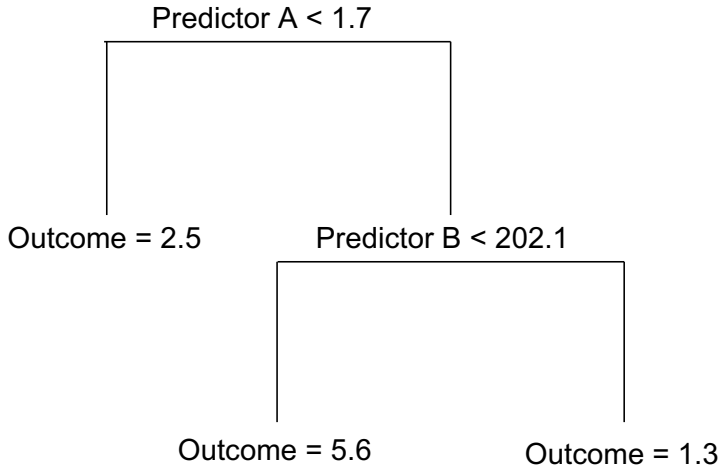
Introduction to tree-based methods

What are tree-based methods?

Tree-based models consists of one or more nested if-then statements for the predictors that partition the data. Within these partitions, a model is used to predict the outcome.
To obtain a prediction for a new sample, we would follow the if-then statements defined by the tree until we get to a terminal node, where a formula would then be used to generate the prediction.

Introduction to tree-based methods





In the example above, a two-dimensional predictor space is cut in three regions, and in each of them the outcome is predicted with a single number.

In the terminology of tree models, we can say that there are two splits of the data into three terminal nodes (also called leaves).

Building a Regression Tree: INTRO

Roughly speaking there are two main steps:

- We divide the Predictor Space (i.e. the set of possible values for X) in distinct and non overlapping regions. In theory the regions could have any shape, but for ease of interpretation, we choose to divide the predictor space in high-dimensional rectangles
- For every observation that falls into a specific region, we make the same prediction, which is the mean of the response values for the training observations in that region.

The goal is to find the region that minimizes the Residual Sum of Squares (RSS)

Building a Regression Tree: MINIMIZING RSS

When building a regression tree the goal is to find the region that minimizes the Residual Sum of Squares (RSS). As it would be unfeasible to consider all possible partitions of the prediction space, we proceed by employing a recursive binary splitting approach.

This approach is defined:

- top-down, as it begins at the top of the tree and then successively splits the predictor space
- Greedy, because at each step the best split is made at that particular step, rather than looking ahead and picking a split that could lead to a better tree in a future step. In practical terms, we consider all predictors X1…Xj and all possible values of the cut-point s for each of the predictors and then we choose the Xj – s pair that minimizes the following equation:

$$\sum_{i:\, x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\, x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

Next, we repeat the process in order to split the data further and minimize the RSS within each of the resulting regions. This goes on until a stopping criterion is met (e.g. The region does not contain more than five observations.

Building a Regression Tree: TREE PRUNING

The greedy approach, especially because of the excessive size of the resulting tree, may overfit the data, leading to poor set performance. A smaller tree with fewer splits could lead to lower variance and better interpretation. To get to a smaller tree we could:

- Build the tree only so long as the decrease in the RSS due to each split exceeds a pre-set threshold. This approach could be too short-sighted, as a seemingly worthless split earlier on in the tree may be followed by a great RSS reduction later on.
- Build a large tree and prune it back.
    - We could choose, among all possible subtrees, the one that would result in the lowest test error rate, that we could estimate with cross-validation or validation set approach. However, estimating the CV error for every possible subtree would not be feasible
    - We could choose the subtree that would result in the lowest test error rate out of only a small set of possible subtrees. **Cost Complexity Pruning** is a way to do just this.

Building a Classification Tree: INTRO

Roughly speaking there are two main steps:

- We divide the Predictor Space (i.e. the set of possible values for X) in distinct and non overlapping regions. In theory the regions could have any shape, but for ease of interpretation, we choose to divide the predictor space in high-dimensional rectangles
- For every observation that falls into a specific region, we make the same prediction, which is equal to the most commonly occurring class of training observations (mode) in that region.

The goal is to find the region that minimizes the Classification Error Rate, which correspond to the fraction of the observation in the region that do not belong to the most common class:

$$E = 1 - \max_k(\hat{p}_{mk})$$

Building a Classification Tree: SPLITS AND PRUNING

When building a Classification tree, besides the Classification Error Rate, the quality of a particular split can be evaluated also by using:

- The Gini Index:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

- The Entropy:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

However, given the higher sensitivity to node purity, the last two approaches are considered to be more appropriate.

When pruning a tree, on the other hand, although all approaches could be employed, the Classification Error Rate would be preferable to achieve the highest prediction accuracy.

Rule-based models

A rule is a set of if-then statements that have been collapsed into independent conditions (e.g. if Predictor A >= 1.7 and Predictor B >= 202.1 then Outcome = 1.3).

If-then statements generated by a tree define a unique route to one terminal node for any sample. However, rules can be simplified (or pruned) in such a way that samples are covered by multiple rules. This approach can offer several advantages over simple tree-based models and represents the base ground for rule-based models.

PROs and CONs of Tree-based models

PROs:

- Easy to explain (some people think that trees, more than other models, closely mirror human decision making)
- Can be displayed graphically and intuitively understood by non-experts
- Can handle qualitative predictors without the need to create dummy variables

CONs:

- Lower levels of predictive accuracy
- Not very robust models: a small change in the data can cause a big change in the final estimated tree

Aggregated tree-based models

By aggregating many decision trees the predictive performance can be drastically improved. The techniques used for the aggregation of the trees are:

- Bagging
- Random Forest
- Boosting

Bibliography

[1] Nabipour, Mojtaba & Nayyeri, Pooyan & Jabani, H. & Mosavi, Amir & Salwana, Ely & Band, Shahab. (2020). Deep Learning for Stock Market Prediction. Entropy. 22. 840. 10.3390/e22080840.

[2] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (1st ed.) [PDF]. Springer.

[3] Song YY, Lu Y. Decision tree methods: applications for classification and prediction. Shanghai Arch Psychiatry. 2015;27(2):130-135. doi:10.11919/j.issn.1002-0829.215044