Supervised Learning - Trees, Bagging, Boosting

Jan-Philipp Kolb

05 Mai, 2019

What is supervised learning?

Supervised learning includes tasks for “labeled” data (i.e. you have a target variable).

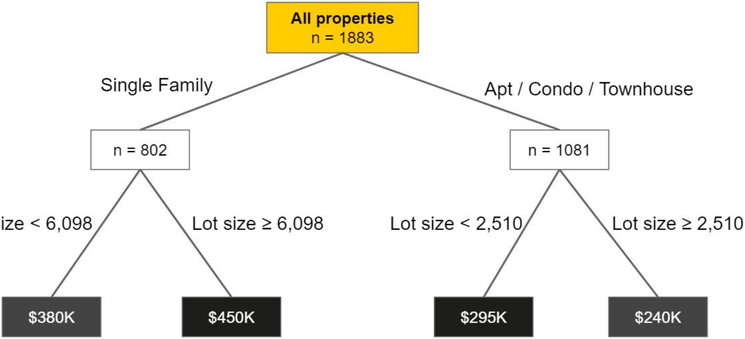
- In practice, it’s often used as an advanced form of predictive modeling.
- Each observation must be labeled with a “correct answer.”
- Only then can you build a predictive model because you must tell the algorithm what’s “correct” while training it (hence, “supervising” it).
- Regression is the task for modeling continuous target variables.
- Classification is the task for modeling categorical (a.k.a. “class”) target variables.

Tree-Based Models

- Trees are good for interpretation because they are simple
- Tree based methods involve stratifying or segmenting the predictor space into a number of simple regions. (Hastie and Tibshirani)
- These methods do not deliver the best results concerning prediction accuracy.

Explanation: decision tree

Decision trees model data as a "tree" of hierarchical branches. They make branches until they reach "leaves" that represent predictions.



All properties
n = 1883

Single Family

Apt / Condo / Townhouse

n = 802

n = 1081

Lot size < 6,098

Lot size ≥ 6,098

Lot size < 2,510

Lot size ≥ 2,510

$380K

$450K

$295K

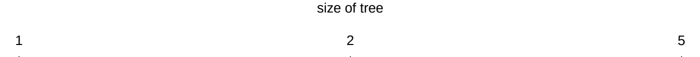
$240K

Figure 1: Decission tree on ElitedataScience

Decision trees used in data mining are of two main types:

- Classification tree
- Regression tree

```
##
## Classification tree:
## rpart(formula = Kyphosis ~ Age + Number + Start, data =
##     method = "class")
##
## Variables actually used in tree construction:
## [1] Age   Start
##
## Root node error: 17/81 = 0.20988
##
## n= 81
##
##          CP nsplit rel error  xerror    xstd
## 1 0.176471      0   1.00000 1.00000 0.21559
## 2 0.019608      1   0.82353 0.88235 0.20565
## 3 0.010000      4   0.76471 0.88235 0.20565
```



Conditional inference tree

- performs recursively univariate split recursively
- **Vignette** package `party`

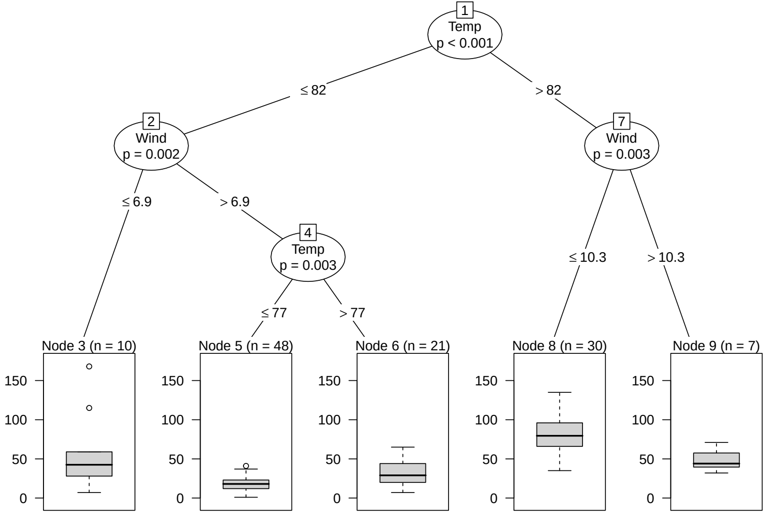
ctree example

The data behind

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   57.00   71.00   79.00   77.87   85.00   97.00
```

A first model

The plot for `ctree`

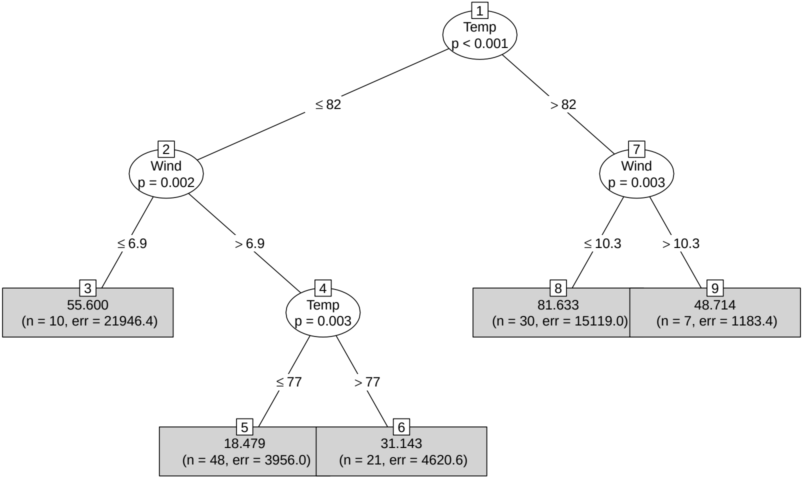


Recursive partitioning algorithms are special cases of a

simple two-stage algorithm

- First partition the observations by univariate splits in a recursive way and
- second fit a constant model in each cell of the resulting partition.

ctree - Regression



Decision Trees

Regression tree vs. classification tree

Grow a tree

```
##
## Classification tree:
## rpart(formula = Kyphosis ~ Age + Number + Start, data =
##     method = "class")
##
## Variables actually used in tree construction:
## [1] Age   Start
##
## Root node error: 17/81 = 0.20988
##
## n= 81
##
##         CP nsplit rel error xerror    xstd
```

Ensembling

Ensembles are machine learning methods for combining predictions from multiple separate models.

Bagging

attempts to reduce the chance overfitting complex models.

- It trains a large number of “strong” learners in parallel.
- A strong learner is a model that’s relatively unconstrained.
- Bagging then combines all the strong learners together in order to “smooth out” their predictions.

Boosting

attempts to improve the predictive flexibility of simple models.

- It trains a large number of “weak” learners in sequence.
- A weak learner is a constrained model (i.e. you could limit the max depth of each decision tree).
- Each one in the sequence focuses on learning from the mistakes of the one before it.

Random Forest

Random forest aims to reduce the previously mentioned correlation issue by choosing only a subsample of the feature space at each split. Essentially, it aims to make the trees de-correlated and prune the trees by setting a stopping criteria for node splits, which I will cover in more detail later.

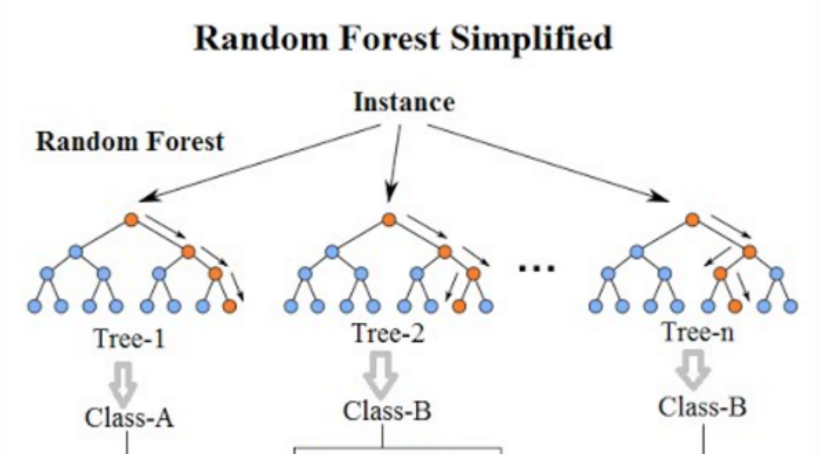
What are the advantages and disadvantages of decision trees?

Advantages: Decision trees are easy to interpret, nonparametric (which means they are robust to outliers), and there are relatively few parameters to tune.

Disadvantages: Decision trees are prone to be overfit. However, this can be addressed by ensemble methods like random forests or boosted trees.

Random forest

- Ensemble learning method - multitude of decision trees
- Random forests correct for decision trees' habit of overfitting to their training set.



Bagging

- Bagging is also known as bootstrap aggregation.
- Bagging is a method for combining predictions from different regression or classification models and was developed by Leo Breiman.
- The results of the models are then averaged in the simplest case.
- The result of each model prediction is included in the prediction with the same weight.
- The weights could depend on the quality of the model prediction, i.e. “good” models are more important than “bad” models.
- Bagging leads to significantly improved predictions in the case of unstable models.

Gradient boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

The idea of gradient boosting originated in the observation by Leo Breiman that boosting can be interpreted as an optimization algorithm on a suitable cost function.

Breiman, L. (1997). “Arcing The Edge”. Technical Report 486. Statistics Department, University of California, Berkeley.

Explicit algorithms

Explicit regression gradient boosting algorithms were subsequently developed by Jerome H. Friedman,[2][3] simultaneously with the more general functional gradient boosting perspective of Llew Mason, Jonathan Baxter, Peter Bartlett and Marcus Frean.[4][5]

The latter two papers introduced the view of boosting algorithms as iterative functional gradient descent algorithms. That is, algorithms that optimize a cost function over function space by iteratively choosing a function (weak hypothesis) that points in the negative gradient direction. This functional gradient view of boosting has led to the development of boosting algorithms in many areas of machine learning and statistics beyond regression and classification.

Advantages of gradient boosting

- Often provides predictive accuracy that cannot be beat.
- Lots of flexibility - can optimize on different loss functions and provides several hyperparameter tuning options that make the function fit very flexible.
- No data pre-processing required - often works great with categorical and numerical values as is.
- Handles missing data - imputation not required.

Disadvantages of gradient boosting

- GBMs will continue improving to minimize all errors. This can overemphasize outliers and cause overfitting. Must use cross-validation to neutralize.
- Computationally expensive - GBMs often require many trees ($>$1000) which can be time and memory exhaustive.
- The high flexibility results in many parameters that interact and influence heavily the behavior of the approach (number of iterations, tree depth, regularization parameters, etc.). This requires a large grid search during tuning.
- Less interpretable although this is easily addressed with various tools (variable importance, partial dependence plots, LIME, etc.).

Two types of errors for tree methods

Bias related errors

- Adaptive boosting
- Gradient boosting

Variance related errors

- Bagging
- Random forest

Gradient Boosting for Linear Regression - why does it not work?

While learning about Gradient Boosting, I haven't heard about any constraints regarding the properties of a “weak classifier” that the method uses to build and ensemble model. However, I could not imagine an application of a GB that uses linear regression, and in fact when I've performed some tests - it doesn't work. I was testing the most standard approach with a gradient of sum of squared residuals and adding the subsequent models together.

The obvious problem is that the residuals from the first model are populated in such manner that there is really no regression line to fit anymore. My another observation is that a sum of subsequent linear regression models can be represented as a single regression model as well (adding all intercepts and corresponding coefficients) so I cannot imagine how that could ever improve the model. The last observation is that a linear regression (the most typical approach) is using sum of squared residuals as a loss function - the

Links - Boosting

- **Gradient Boosting Machines**
- How to Visualize Gradient Boosting Decision Trees With XGBoost in Python

Links

- **Vignette** for package `partykit`
- Conditional Inference Trees
- Conditional inference trees vs traditional decision trees
- Video on tree based methods