# Classification and Regression Trees
# Random Forest

WI4231

TUD

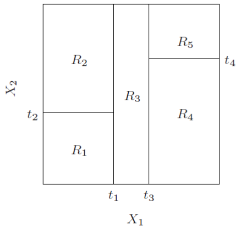
## Outline

- Classification and Regression Trees
- Ensemble (Aggregation) methods
  - Bagging
  - Random Forest
- Introduction to homework

## Tress

In regression and classification trees we are looking for a simple function $\hat{y}(\mathrm{x})$, where x are predictors/inputs and $y$ is the output.

The idea is to partition the input space onto disjoint regions and fit a simple model on these regions.



- Classification: Majority vote within the region.
- Regression: Mean of output of training data within the region.

## Partitions

How to find an "optimal" partition?

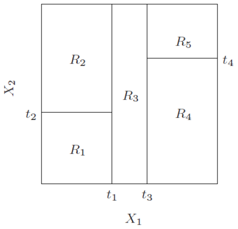
Even if we restrict our attention to simple regions (e.g. "boxes"), finding an optimal partition which minimizes the training error is computationally infeasible.

Instead, "greedy" approach is used: recursive binary splitting.

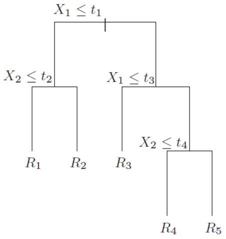
1. Select one input variable $x_j$ and a cut-point $s$. Partition the input space into two half-spaces,
$$\{\mathrm{x} : x_j < s\} \text{ and } \{\mathrm{x} : x_j \geq s\}.$$
2. Repeat this splitting for each region until some stopping criterion is met (e.g. no region contains more than 5 training data points).

## Regression Trees

Partition of input space



Tree representation

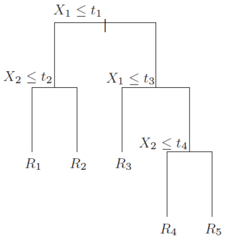


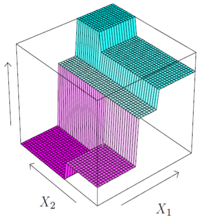
## Trees

Once the input space is partitioned into $M$ regions, $R_1, R_2, ..., R_M$ the prediction model is:

$$\hat{y}(x) = \sum_{m=1}^{M} \hat{y}_m \mathbb{1}_{\{x \in R_m\}}$$

where $\mathbb{1}_A$ denotes the indicator function of set $A$ and $\hat{y}_m$ is a constant prediction of the output in $m^{th}$ region.





The prediction surface is not smooth but can handle nonlinearities.

## How to find partitions?

For regression trees:

For any $j$ and $s$, define

$$R_1(j,s) = \{\mathrm{x} | x_j < s\} \text{ and } R_2(j,s) = \{\mathrm{x} | x_j \geq s\}.$$

We then seek $(j,s)$ that minimize

$$\sum_{i:\mathrm{x}_i \in R_1(j,s)} (y_i - \hat{y}_1(j,s))^2 + \sum_{i:\mathrm{x}_i \in R_2(j,s)} (y_i - \hat{y}_2(j,s))^2,$$

where

$$\hat{y}_1(j,s) = avarage\{y_i : \mathrm{x}_i \in R_1(j,s)\},$$
$$\hat{y}_2(j,s) = avarage\{y_i : \mathrm{x}_i \in R_2(j,s)\}.$$

This optimization problem is solved by evaluating all possible splits.

## How to find partitions?

For classification trees:

Compute the proportion of data points from class $k$ in node $m$ corresponding to region $R_m$ with $N_m$ points

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{i:x_i \in R_m} \mathbb{1}_{y_i = k}.$$

Pick instead of the squared loss used to construct the regression tree a measure suitable for categorical outputs:

- Misclassification error: $L(\hat{p}_m) = 1 - \max_k \hat{p}_{mk}$
- Entropy/deviance: $L(\hat{p}_m) = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$
- Gini index: $L(\hat{p}_m) = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$

Each split is chosen to minimize sum of losses over possible splits:

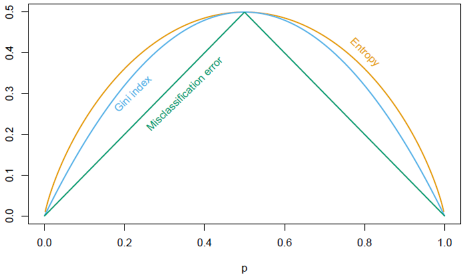
$$L(\hat{p}_1) + L(\hat{p}_2).$$

The prediction for each region is chosen by majority vote within the region, hence the class with largest $\hat{p}_{mk}$.

## Loss functions for $m = 2$

For two classes, if $p$ is the proportion in the second class, Misclassification error, Entropy and Gini Index measures are:

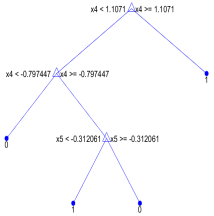
$$1 - \max(p, 1-p), \quad -[p\log p + (1-p)\log(1-p)], \quad 2p(1-p).$$

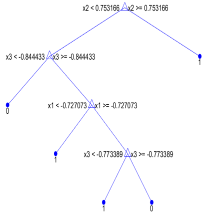


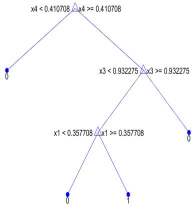
## CaRT

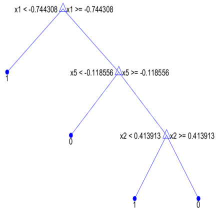
- To obtain a small bias the tree needs to be grown deep,
- this results in a high variance.

  $N = 30$, 5D feature vector with Gaussian distribution correlated 0.95,
  $P(y = 1|x_1 \leq 0.5) = 0.2$ and $P(y = 1|x_1 > 0.5) = 0.8$.









## CaRT

To improve the practical performance:

- Pruning – grow a deep tree (small bias) which is then pruned into a smaller one (reduce variance).
- Ensemble methods – average or combine multiple trees.

## How to Prune

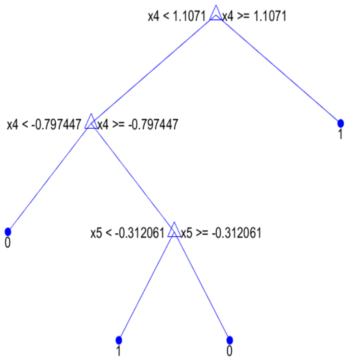
- Strategy 1:
  Split nodes only if the decrease in sum-of-squares due to the split exceeds some threshold. NO
- Strategy 2:
  - Grow a large tree $T_0$, stopping the splitting process only when some minimum node size (say 5) is reached.
  - Define a subtree $T \subset T_0$ to be any tree that can be obtained by pruning $T_0$. Index terminal nodes by $m$ (node $m$ representing region $R_m$). Let $|T|$ denote the number of terminal nodes in $T$.

$$C_\alpha(T) = \sum_{m=1}^{|T|} \left[ \sum_{i: x_i \in R_m} (y_i - \hat{y}_m)^2 \right] + \alpha |T|.$$

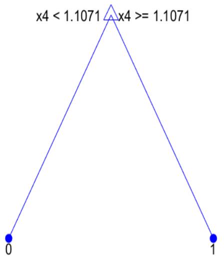
  Find, for each $\alpha$, the subtree $T_\alpha \subset T_0$ to minimize $C_\alpha(T)$ via cross validation. as LASSO

## Simulated example

Not Pruned

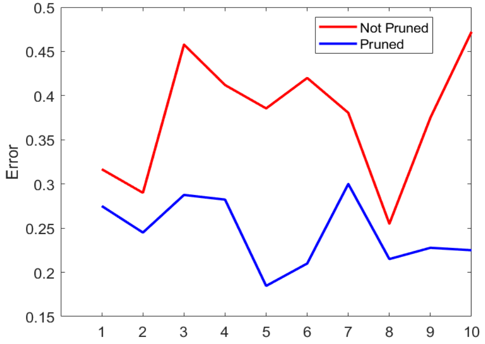


Pruned



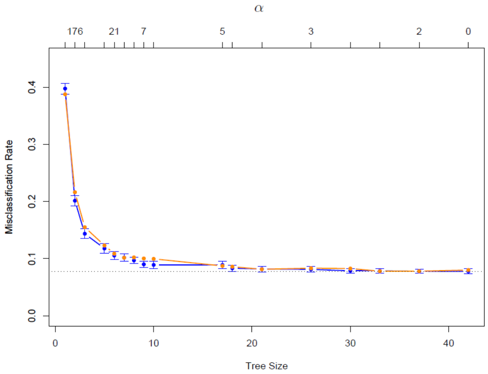
## Simulated example

In sample (training) error: $L_{NotPruned} = 0.0333$ and $L_{Pruned} = 0.1333$.
Out of sample (test) error:



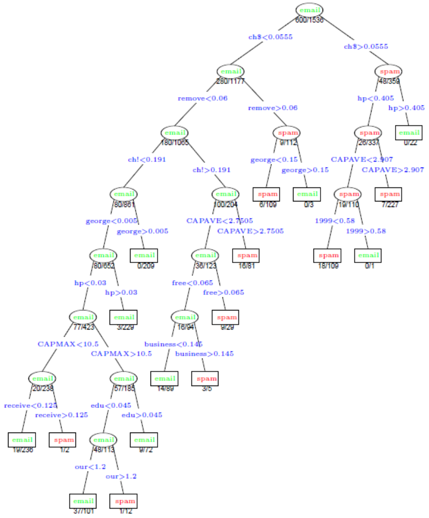
## SPAM Example

4601 e-mails - classify as spam/e-mail based on features.



blue- 10-fold cross- validation estimate of classification error; orange - test error

## SPAM Example



Test error: 0.093

## Interpretability, Variable Importance

- CaRT are very easily interpretable.
- Variables used to make split higher in the tree are more important.
- Measure of importance: Calculate the total amount that the SE or Gini index is decreased due to splits over a given predictor.

## Averaging

Let $z_b$, $b = 1, ..., B$ be identically distributed random variables with mean $E[z_b] = \mu$ and variance $Var[z_b] = \sigma^2$. Let $\rho$ be the correlation between distinct variables. Then,

$$\begin{aligned} E\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] &= \mu, \\ Var\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] &= \frac{1-\rho}{B}\sigma^2 + \rho\sigma^2 \end{aligned}$$

The variance is reduced for large $B$ and when $\rho < 1$. However, positive correlation dampens this reduction.

## Why averaging works - theoretically

- $D = \{(x_1, y_1), ..., (x_N, y_N)\}$ iid from $\mathcal{P}$.
- Train an algorithm on $D$ to get classifier $y_D(x)$.
  $D$ drawn from $\mathcal{P}^N$ hence
  we can consider $y_{ag}(x) = E_D(y_D(x))$.

Test error (of algorithm)

$$\begin{aligned} E_{x,y,D}\left[(y - y_D(x))^2\right] &= E_{x,y,D}\left[\{(y - y_{ag}(x)) + (y_{ag}(x) - y_D(x))\}^2\right] \\ &= E_{x,y}\left[(y - y_{ag}(x))^2\right] + E_{x,D}\left[(y_{ag}(x) - y_D(x))^2\right] \\ &\geq E_{x,y}\left[(y - y_{ag}(x))^2\right] \end{aligned}$$

True population aggregation never increases mean square error. Extra error on the right hand side comes from variance of $y_D(x)$.

## Bagging

- Get bootstrap samples $\mathcal{T}^1, ..., \mathcal{T}^B$ from the original training data.
- Train classifiers $y_{\mathcal{T}^b}(x), b = 1, ..., B$
- Form aggregate classifier

$$y_{ag}(x) = \frac{1}{B} \sum_{b=1}^{B} y_{\mathcal{T}^b}(x).$$

This process is called bootstrap aggregation or bagging.

True population aggregation never increases mean square error. This suggests that bagging—drawing samples from the training data will often decrease mean-squared error.

## Bagging classification trees

For classification trees ($K$ classes):
$B$ bootstrap sets $\mathcal{T}^1, ..., \mathcal{T}^B$ .

$$y_{ag}(\mathrm{x})$$

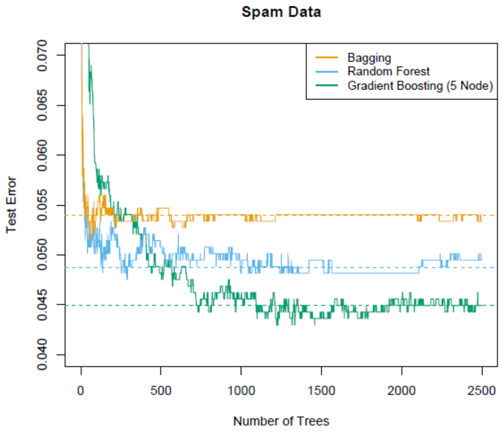
we get

$$(p_1(\mathrm{x}), p_2(\mathrm{x}), ..., p_K(\mathrm{x}))$$

where $p_k(\mathrm{x})$ is equal to proportion of trees (out of $B$) predicting class $k$, $k = 1, ..., K$ at x. The bagged classifier selects class with the most "votes", largest $p_k(\mathrm{x})$.

Theoretical justification for square loss does not hold for classification under 0-1 loss because no additivity of bias and variance. In this setting averaging good classifier might make it better but averaging bad classifier might make it worse.

## Bagging SPAM Example



## Bagging

- Bagging results in improved accuracy over prediction using a single tree.
  It can happen that grown trees are very similar which leads to correlation between trees and the reduction of variance due to averaging is diminished. This happens due to bootstrapping (specially in small data sets) and/or in case there are only few important predictors which appear in top splits of all trees.
- Unfortunately, difficult to interpret the obtained model.
  Bagging improves prediction accuracy at the expense of interpretability.
- Variable importance averaged over trees.

## Random Forest

Algorithm Random Forest for Regression or Classification.

```
1. For $b = 1$ to $B$:
    (a) Draw a bootstrap sample $Z^*$ of size $N$ from the training data.
    (b) Grow a random-forest tree $T_b$ to the bootstrapped data, by re- cursively
        repeating the following steps, until the minimum node size $N_{min}$ is reached.
        i. Select $m$ variables at random from the $p$ variables.
        ii. Pick the best variable/split-point among the $m$.
        iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_{b=1}^{B}$.
```

To make a prediction at a new point x:

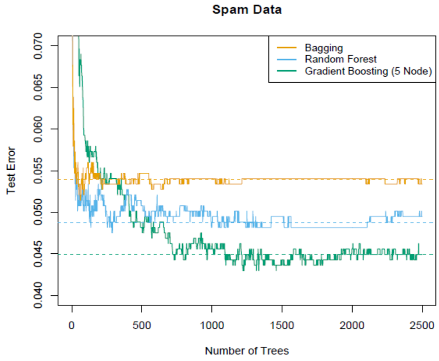
Regression:

$$\hat{y}_{rf}(\mathrm{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\mathrm{x})$$

Classification: Let $\hat{C}_b(\mathrm{x})$ be the class prediction of the $b^{th}$ random-forest tree.

$$\hat{C}_{rf}^{B}(\mathrm{x}) = \text{ majority vote } \{\hat{C}_b(\mathrm{x})\}_1^B.$$

## SPAM Example



## Homework

- Analyze and predict price of shoes (data set on BS)
- Split the data on training and test (70% /30%).
- Build a tree model, perform bagging and Random Forest
- Discuss choices of parameters of these models. Is it the case that the default choices lead to the best performing model? Present discussion of your findings.
- Compare performance of your models on training and test data.
- Analyze variable importance and give a discussion.
- Provide a discussion about fitted models from the perspective of simplicity, interpretability, easiness of fitting, accuracy etc.
- Submit your report and implementation via BS.
- Deadline 02.04.2021.