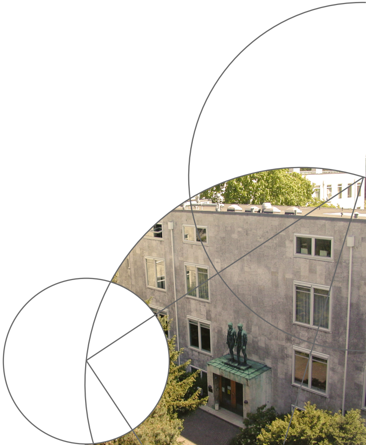
Faculty of Science

Trees and Forests

Data Science

Christian Igel
Department of Computer Science



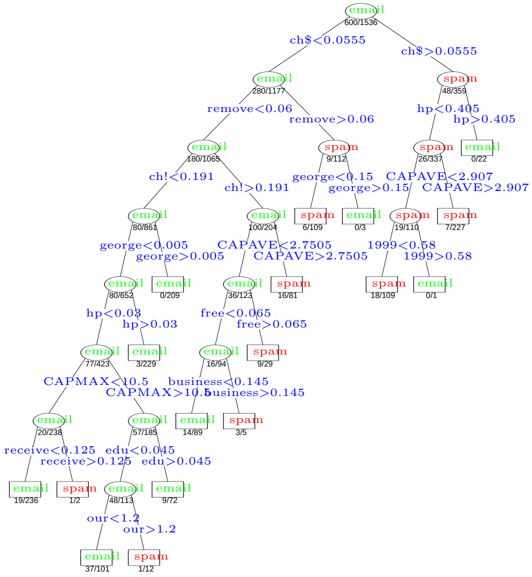
Outline

Outline

Tree-based methods

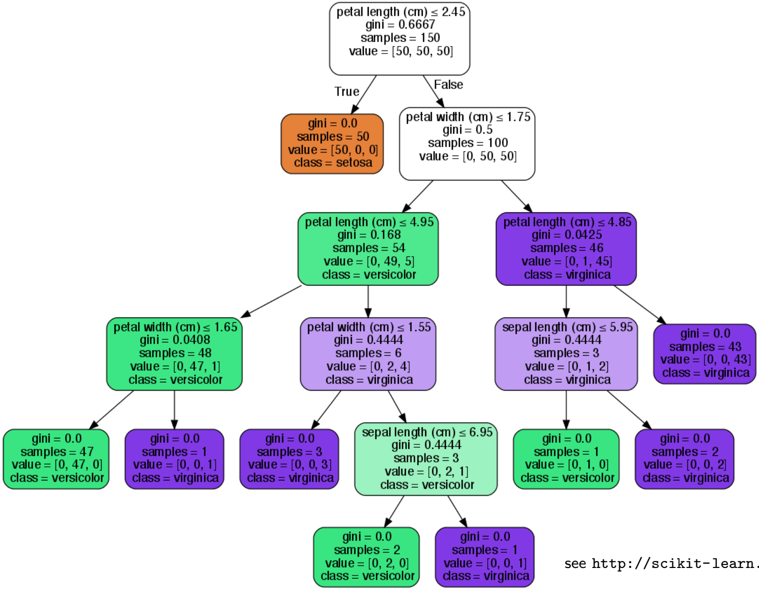
- Tree-based models are simple but powerful and are human interpretable.
- Tree-based methods partition the feature space (in the case of the $\mathbb{R}^D$ into rectangular regions) and assign a simple model – usually a constant value/label – to each region $\mathcal{R}_\tau$.
- Several tree-based methods exist (C4.5, ID3, ...), we will focus on CART (Classification and Regression Trees). CART trees are binary trees.

Example: Spam detection



Hastie, Tibshirani, & Friedman. *The Elements of Statistical Learning*. Springer, 2009

Example: Iris data



see `http://scikit-learn.org`

Tree evaluation, basic idea

To evaluate a tree given an input $\boldsymbol{x}$:

- Start at the root node
- Each inner node corresponds to some if-then rule assigning $\boldsymbol{x}$ to one of its children, e.g.:

  `if` $x_d < \theta$ `then goto left child node,`
  `else goto the right child node.`
- When a leaf node is reached, $\boldsymbol{x}$ is assigned to the value or label (or distribution over labels) associated with that leaf node.

Let the leaf nodes be indexed by $\tau = 1, \dots, |T|$. They define the regions. If $\boldsymbol{x}$ reaches leaf node τ, we have $\boldsymbol{x} \in \mathcal{R}_\tau$.

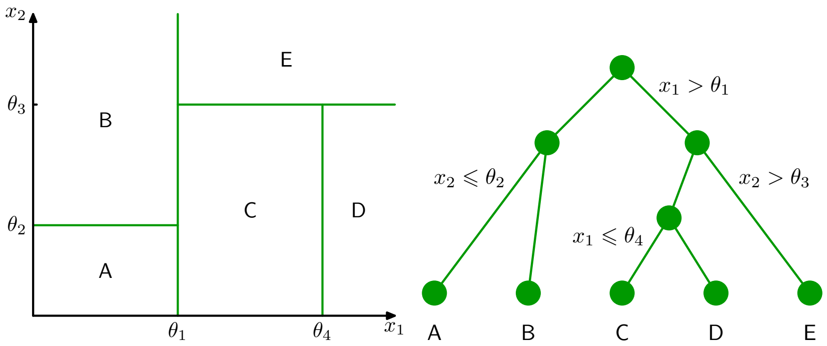
CART rules

- Every inner node is associated with one coordinate $d \in \{1, \dots, D\}$ and a threshold θ.
- At each inner node, the training data $S = \{(\boldsymbol{x}_1, y_1), \dots\}$ at that node is split into

  $$L_{d,\theta} = \{(\boldsymbol{x}, y) \in S \,|\, x_d < \theta\} \text{ and } R_{d,\theta} = \{(\boldsymbol{x}, y) \in S \,|\, x_d \geq \theta\} \ ,$$

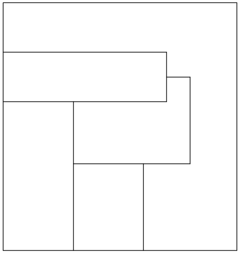
  passed to the left and right daughter node, respectively.
- The optimal tree cannot be found efficiently. Therefore trees are built using a heuristic.

Example



Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

Not all splits are possible with a binary tree



Hastie, Tibshirani, & Friedman. *The Elements of Statistical Learning*. Springer, 2009

Building a CART tree, basic idea

- Every inner node is associated with one coordinate $d \in \{1, \dots, D\}$ and a threshold θ.
- At each inner node, the training data $S = \{(\boldsymbol{x}_1, y_1), \dots\}$ at that node is split into $L_{d,\theta}$ and $R_{d,\theta}$ such that the information gain

$$G_{d,\theta}(S) = Q(S) - \frac{|L_{d,\theta}|}{|S|} Q(L_{d,\theta}) - \frac{|R_{d,\theta}|}{|S|} Q(R_{d,\theta})$$

  is maximized, where Q is some impurity measure.
- If the number of datapoints $|S|$ at that node is smaller than a threshold $s_{\text{threshold}}$ or the data is pure (i.e., all elements have the same label), the node becomes a leaf.
- After growing the tree, it is pruned to reduce its complexity.

Growing a tree recursively

Procedure `GrowTreeRecursively(`S`)`

```
Input: $S = \{(\boldsymbol{x}_1, y_1), \dots$, maximum number $m$ of variables
       considerd per split
1 if $|S| < s_{\mathit{threshold}}$ then return terminal node with labels
  $\{y_1, \dots, y_{|S|}\}$
2 if $\forall(\boldsymbol{x}_i, y_i), (\boldsymbol{x}_j, y_j) \in S : y_i = y_j$ then return terminal node
  with labels $\{y_1, \dots, y_{|S|}\}$
3 find $(d, \theta)$ maximizing $G_{d,\theta}(S)$
4 left child = GrowTreeRecursively $(L_{d,\theta}(S))$
5 right child = GrowTreeRecursively $(R_{d,\theta}(S))$
6 return inner node with split $(d, \theta)$
```

Regression trees

- Training data $\mathcal{S} = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\} \in (\mathbb{R}^D \times \mathbb{R})^N$
- Each leaf node $\tau = 1, \dots, |T|$ of a tree T returns a constant, i.e., the output $\hat{y} = T(\boldsymbol{x})$ given an input $\boldsymbol{x}$ is

$$T(\boldsymbol{x}) = \sum_{\tau=1}^{|T|} c_\tau \mathbb{I}\{\boldsymbol{x} \in \mathcal{R}_\tau\} \ .$$

- We consider the squared loss $(\hat{y} - y)^2$. Then the choice for the constants minimizing the empirical risk is

$$c_\tau = \frac{1}{N_\tau} \sum_{\boldsymbol{x}_n \in \mathcal{R}_\tau} y_n$$

with $(\boldsymbol{x}_n, y_n) \in S$ and $N_\tau = |\{(\boldsymbol{x}_n, y_n) \in S \,|\, \boldsymbol{x}_n \in \mathcal{R}_\tau\}|$.

How to find the optimal split?

- If the impurity measure is the squared loss, we have to find at every node the split (d, θ) solving

$$\min_{d,\theta}\left[\min_{c_L}\sum_{(\boldsymbol{x},y)\in L_{d,\theta}}(y-c_L)^2+\min_{c_R}\sum_{(\boldsymbol{x},y)\in R_{d,\theta}}(y-c_R)^2\right]$$

  given the training data at the node.
- The inner minimizations can be solved . . . by the averages $c_L = \frac{1}{|L_{d,\theta}|}\sum_{(\boldsymbol{x},y)\in L_{d,\theta}} y$ and $c_R = \frac{1}{|R_{d,\theta}|}\sum_{(\boldsymbol{x},y)\in R_{d,\theta}} y$.
- For every d, after sorting the training data according to the dth component, only thresholds corresponding to means of dth components of two successive data points need to be tested.

Pruning criterion for regression trees

- For a (sub)tree T, we define the impurity measure

$$Q_\tau(T) = \frac{1}{N_\tau} \sum_{(\boldsymbol{x}_n, y_n) \in S \wedge \boldsymbol{x}_n \in \mathcal{R}_\tau} \{y_n - c_\tau\}^2 \ ,$$

  where $N_\tau = |\{(\boldsymbol{x}_n, y_n) \,|\, (\boldsymbol{x}_n, y_n) \in S \wedge \boldsymbol{x}_n \in \mathcal{R}_\tau\}|$.
- Pruning criterion for $\alpha \geq 0$:

$$C_\alpha(T) = \sum_{\tau=1}^{|T|} N_\tau Q_\tau(T) + \alpha |T|$$

Pruning regression trees

- Problem: For given α, find subtree T_α minimizing $C_\alpha(T)$
- Solution: Starting from the full-grown tree T_0, create sequence of subtrees by collapsing (i.e., replacing by fusing the children) in every step the inner node such that the increase in $\sum_\tau N_\tau Q_\tau(T)$ is minimal
- This finite sequence contains all T_α.
- Proper α is selected using hold-out data.

Classification trees

- Consider classification into K classes
- Let $p_{\tau k}$ be the fraction of training data points in region $\mathcal{R}_\tau$ belonging to class k.
- The tree T assigns

$$T(\boldsymbol{x}) = \operatorname{argmax}_k p_{\tau k} \quad \text{if } \boldsymbol{x} \in \mathcal{R}_\tau \ .$$

Note that $p_{\tau k}$ gives a probability distribution over the classes for a given $\boldsymbol{x} \in \mathcal{R}_\tau$.

Impurity measures for classification trees

Classification error:

$$Q_\tau(T) = \frac{1}{N_\tau} \sum_{(\boldsymbol{x}_n, y_n) \in S \wedge \boldsymbol{x}_n \in \mathcal{R}_\tau} \mathbb{I}\{y_n \neq T(\boldsymbol{x}_n)\}$$

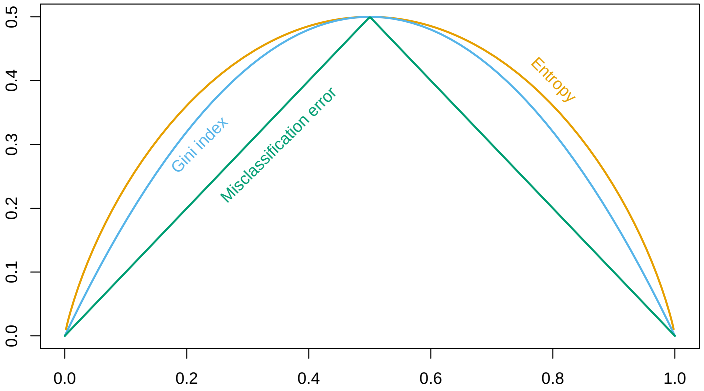
Entropy:

$$Q_\tau(T) = -\sum_{k=1}^{K} p_{\tau k} \ln p_{\tau k}$$

Gini index:

$$Q_\tau(T) = \sum_{k=1}^{K} p_{\tau k}(1 - p_{\tau k})$$

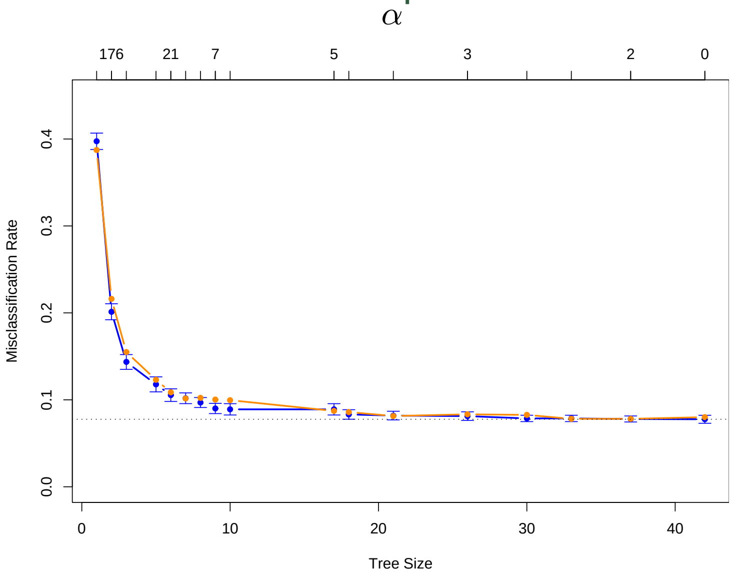
Comparison of impurity measures



binary task, x-axis shows fraction of first class
entropy rescaled to 0.5

Hastie, Tibshirani, & Friedman. *The Elements of Statistical Learning*. Springer, 2009

Spam classification example: Tree size



Hastie, Tibshirani, & Friedman. *The Elements of Statistical Learning*. Springer, 2009

Outline

Bias-variance decomposition, noiseless case

Let h_S be the hypothesis learnt on training data S and $(x, y) \sim p$. The expected risk under the squared loss is:

$$\begin{aligned}
&\mathbb{E}_S \underbrace{\mathbb{E}_p\{(y - h_S(x))^2\}}_{\text{risk under squared loss}} \\
&= \mathbb{E}_S\mathbb{E}_p\{(y - \mathbb{E}_{S'}\{h_{S'}(x)\} + \mathbb{E}_{S'}\{h_{S'}(x)\} - h_S(x))^2\} \\
&= \mathbb{E}_S\mathbb{E}_p\{(y - E_{S'}\{h_{S'}(x)\})^2\} \\
&+2\mathbb{E}_S\mathbb{E}_p\{(y - E_{S'}\{h_{S'}(x)\})(\mathbb{E}_{S'}\{h_{S'}(x)\} - h_S(x))\} \\
&+\mathbb{E}_S\mathbb{E}_p\{(\mathbb{E}_{S'}\{h_{S'}(x)\} - h_S(x))^2\} \\
&= \underbrace{\mathbb{E}_p\{(y - \mathbb{E}_{S'}\{h_{S'}(x)\})^2\}}_{\text{bias}^2} + \underbrace{\mathbb{E}_S\mathbb{E}_p\{(\mathbb{E}_{S'}\{h_{S'}(x)\} - h_S(x))^2\}}_{\text{variance}}
\end{aligned}$$

Typically: Large bias indicates that hypothesis class is too restricted, large variance indicates overfitting and occurs if hypothesis class is too complex.

Outline

Limitations of trees

Single decision trees

1. are instable, in the sense that changing the data set slightly may lead to a very different tree,
2. lack smoothness,
3. are not well understood in terms of statistical learning theory.

Random forests address the first two issues by averaging over many trees.

- For training, B trees are grown, using different subsets of the data and different splitting variables.
- The outputs of the trees are combined to give the final prediction.

A random forest is an ensemble classifier. Averaging models is commonly used for variance reduction.

Growing a random forest tree recursively

```
Procedure GrowRFTreeRecursively($S$, $m$)
Input: $S = \{(\boldsymbol{x}_1, y_1), \dots\}$
1 if $|S| < s_{threshold}$ then return terminal node with labels $\{y_1, \dots, y_{|S|}\}$
2 if $\forall(\boldsymbol{x}_i, y_i), (\boldsymbol{x}_j, y_j) \in S : y_i = y_j$ then return terminal node with labels $\{y_1, \dots, y_{|S|}\}$
3 randomly select variables $d_1, \dots, d_m$ from $\{1, \dots, D\}$ find $(d, \theta) \in \{d_1, \dots, d_m\} \times \mathbb{R}$ maximizing $G_{d,\theta}(S)$
4 left child = GrowRFTreeRecursively $(L_{d,\theta}(S))$
5 right child = GrowRFTreeRecursively $(R_{d,\theta}(S))$
6 return inner node with split $(d, \theta)$
```

Growing and evaluating a random forest

```
Algorithm 1: Random Forest
Input: $S = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, number of trees $B$,
       number of variables $m$
Output: trees $T_1, \dots, T_B$
1 for $b = 1, \dots, B$ do
2     draw a bootstrap sample $S'$ by drawing $N$ elements (with
      replacement) from $S$
3     $T_b$ = GrowRFTreeRecursively ($S'$, $m$)
```

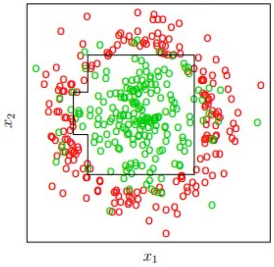
Regression:

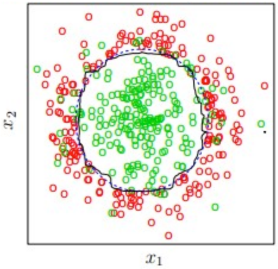
$$f_{\mathsf{RF}}(\boldsymbol{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\boldsymbol{x})$$

Classification:

$$f_{\mathsf{RF}}(\boldsymbol{x}) = \text{majority vote of } T_1, \dots, T_B$$

Example of decision boundary

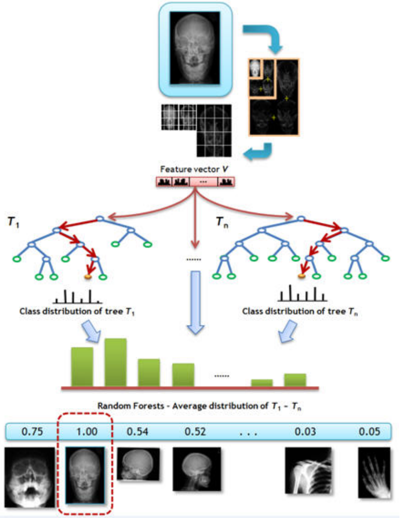




tree (left) vs. random forest (right)

from `http://www.rapidsnail.com`

X-ray classification example


Feature vector V
T_1
T_n
Class distribution of tree T_1
Class distribution of tree T_n
Random Forests - Average distribution of T_1 - T_n
0.75
1.00
0.54
0.52
. . .
0.03
0.05

- Number of trees denoted by n instead of B.
- Fusion of the probabilistic output instead of voting. Final results are not normalized.

Ko, Kim, & Nam. X-ray Image Classification using Random Forests with Local Wavelet-Based CS-Local Binary Patterns. *Journal of Digital Imaging* 24(6):1141-1151, 2011

Random forests details

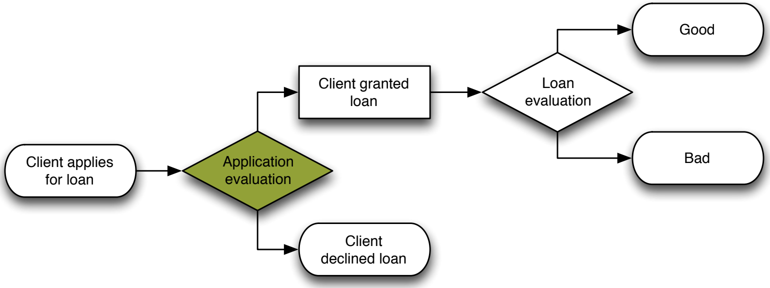
- Pruning is not used.
- Defaults for choosing m:
  - For classification $m = \lfloor \sqrt{D} \rfloor$
  - For regression $m = \lfloor D/3 \rfloor$
- Choosing B: In general, the bigger the better. $B = 100$ may serve as a starting point.

Out-of-bag samples

- Out-of-bag (OOB) samples can be used to evaluated generalization performance of random forests: For each observation $\boldsymbol{x}_i$ in the training set a random forest predictor is built by averaging over all trees constructed not using $\boldsymbol{x}_i$.
- Number of trees can be determined by increasing forest until OOB sample error converges.
- OOB sample error can be used to determine m instead of using cross-validation.

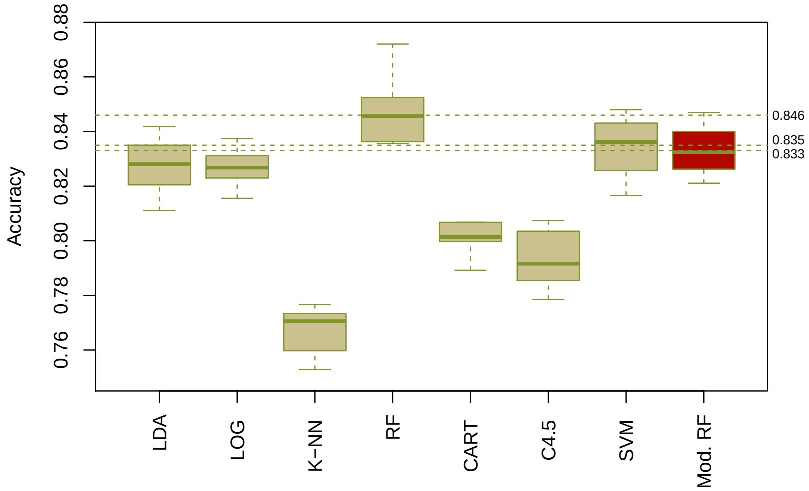
Business example: Credit scoring

A credit score measures the creditworthiness of a client.



figures in this section provided by Kasper Nybo Hansen

Results from MSc thesis



CART: Pros and cons

- ⊕ Good interpretability
- ⊕ Optional probabilistic output
- ⊕ Applicable to both numerical and categorical data
- ⊕ Applicable to large data sets
- ⊖ Suffer from instability and high variance
- ⊖ Non-smooth decision boundaries
- ⊖ Comparatively little theoretical understanding in terms of statistical learning theory

Random forests: Pros and cons

- ⊕ Good performance without much tuning
- ⊕ Applicable to both numerical and categorical data
- ⊕ Simple parallelization, applicable to large data sets
- ⊕ Optional probabilistic output
- ⊖ Comparatively little theoretical understanding in terms of statistical learning theory