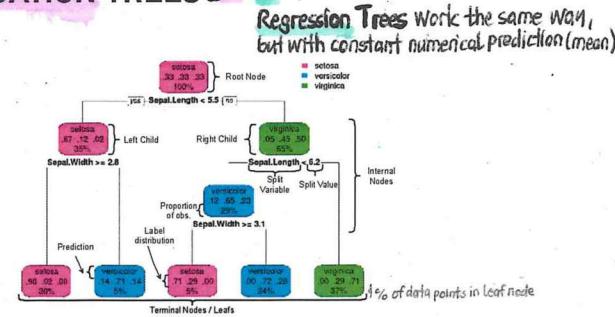


CLASSIFICATION TREES [CART]



- Classification trees use the structure of a binary tree
- Binary splits are constructed top-down in a *data optimal way*
- Each split is a threshold decision for a single feature
- Each node contains the training points which follow its path
- Each leaf contains a constant prediction
Instead of hard Label can also predict class probabilities (visible in picture)
- Predicting new data:** Use learned split points & pass observation through tree
Each (new) obs. lands in exactly one leaf

Sidenote: CART invariant to transf. of numerical features
Structure of tree identical, only split point values change

CART great with feature interaction as those are inherently modeled

CATEGORICAL FEATURES

- A split on a categorical feature partitions the feature levels:
 $x_j \in \{a, b, c\} \leftarrow N \rightarrow x_j \in \{d, e\}$
- For a feature with m levels, there are about 2^m different possible partitions of the m values into two groups ($2^{m-1} - 1$ because of symmetry and empty groups).
- Searching over all these becomes prohibitive for large values of m .
- For regression with L2 loss and for binary classification, we can define clever shortcuts.

For 0 - 1 responses, in each node:

- Calculate the proportion of 1-outcomes for each category of the feature in N .
- Sort the categories according to these proportions.
- The feature can then be treated as if it was ordinal, so we only have to investigate at most $m - 1$ splits.

(this may be super easy to separate)
Split evaluation only possible for observations where used feature has no NA \Rightarrow can bias splits towards features with many NAs
but bigger problem is NAs during prediction time (new obs.)

SURROGATE SPLITS

- created during training as *alternative split* using other features of that node, mimicking outcome of optimal X_j -split
really only ONE split
e.g. by learning $(x_j < c = \text{TRUE or FALSE})$ as target column
- typically create several surrogates (like 5) for every inner node \Rightarrow order of surrogates to use when better surrogate-feature also NA

TREE AS AN ADDITIVE MODEL

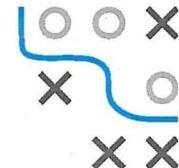
Trees divide the feature space \mathcal{X} into rectangular regions:

$$f(\mathbf{x}) = \sum_{m=1}^M c_m \mathbb{I}(\mathbf{x} \in Q_m),$$

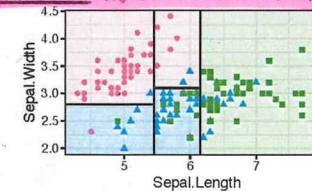
↳ defines Hypothesis Space

where a tree with M leaf nodes defines M "rectangles" Q_m .

c_m is the predicted numerical response, class label or class distribution in the respective leaf node.



Classif. Trees: Not great if linear decision boundary is best



↳ flexible with non-linearities, more stable vs. outliers than normal LM
Regression Trees with L2 loss:
piecewise constant LM (some as kNN) (bagging improves this)
cart model: sloped curve or smooth fct. in general
↳ linear dependences can only be modelled over several splits
constant pred. at edges!
⇒ terrible extrapolation to feature value outside train range and often bad performance on test data

↳ Very first split can later turn out to be globally suboptimal for final tree!
CART greedy search: deterministic tree, conditionally optimal based on previous split(s) only for current split
No joint global optimiz. of full sequence of possible splits, don't know if possible additional later splits are "worth" upfront (Horizon Effect)
⇒ Even for train data, CART often doesn't find best decision tree (lowest sum of Remp across all leaves) from Ht, doubtless: its NP-hard

↳ CART trees are very sensitive to small changes in Dtrain and tend to generalize worse to new data than other Learners bc. of overfitting
Boosting methods (#Random Forests) improve predictive performance by averaging over many trees
RF Trees more stable
These branches become new leaf nodes! leaf nodes
CART complexity pruning (CCP) Grow large tree (without stop criteria) \Rightarrow then, repeatedly, remove branches with least informative
Greedy backward search ends when regularized risk no longer sinks $R_{\text{reg}}(T) = \sum_{m=1}^M \sum_{i \in \text{leaf } m} L_i(y_i, c_m) + \alpha T_1$ T_1 is subtree after removal
⇒ then we found optimal (sub)tree size given To
as penalizes number of leaf nodes $|T|$
To: starting tree

CART FINDING THE BEST SPLIT

Empirical risk

$$t \in \{\text{midpoint of two unique } (x_j) \text{ vals}\}$$

- Splitting feature x_j at split point t divides a parent node N into two child nodes:

$$N_1 = \{(x, y) \in N : x_j \leq t\} \text{ and } N_2 = \{(x, y) \in N : x_j > t\}$$

- Compute empirical risks in child nodes and minimize their sum to find best split (impurity reduction):

$$\text{Automatic feature selection} \quad \arg \min_{j,t} \mathcal{R}(N, j, t) = \arg \min_{j,t} \mathcal{R}(N_1) + \mathcal{R}(N_2) \quad |N| = \text{number of data points in Node}$$

Note: If \mathcal{R} is the average instead of the sum of loss functions, we need to reweight: $\frac{|N_1|}{|N|} \mathcal{R}(N_1)$

[0-1 loss also an option, but less sensitive to Δt producing less pure nodes]

↳ g-way classification: $\hat{\pi}_k(N) = \frac{1}{|N|} \sum_{(x,y) \in N} I_{\{y=k\}}$ for both class. loss functions as optimal pred. prob. in child node split homogeneously
Brier score \rightarrow Gini impurity

$$\text{Brier Score: } \sum_{k=1}^M \frac{1}{|N|} \sum_{i=1}^{|N|} (\hat{\pi}_{k(i)} - \pi_{k(i)})^2$$

$$\text{Log Loss: } - \sum_{k=1}^M \hat{\pi}_k(N) \log(\hat{\pi}_k(N))$$

\hookrightarrow insert $\hat{\pi}_k(N)$... for impurity

Bernoulli loss \rightarrow entropy impurity

$$\mathcal{R}(N) = \sum_{(x,y) \in N} \sum_{k=1}^M \hat{\pi}_k(N) (1 - \hat{\pi}_k(N)) \quad \mathcal{R}(N) = - \sum_{(x,y) \in N} \sum_{k=1}^M \hat{\pi}_k(N) \log \hat{\pi}_k(N) \quad |N|: \text{hard label pred. by highest freq.}$$

Regression (quadratic loss): $\mathcal{R}(N) = \sum_{(x,y) \in N} (y - c)^2$ with $c = \frac{1}{|N|} \sum_{(x,y) \in N} y$ \rightarrow min. variance impurity

Optimization \hookrightarrow in i -th child node: $c_{Ni} = \arg \min_c \sum_{(x,y) \in N_i} L_i(y, c)$ optimal const. pred., L doesn't have to be different. In loop-based tree-opt.

- Exhaustive search over all split candidates, choice of risk-minimal split ("greedy" search)
- In practice: reduce number of split candidates (e.g., using quantiles instead of all observed values)

Exercise: CART Trees and Bagging/Random Forests

Regression Trees

We derive optimal const. predictor for node N (mean) by minimizing Remp under L2 Loss

$$\begin{aligned} \underset{c \in \mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{|N|} (\hat{y}_i - c)^2 &= \sum_{i=1}^{|N|} (\hat{y}_i - c)^2 = \sum_i (\hat{y}_i)^2 - 2c \sum_i \hat{y}_i + c^2 = |N|c^2 - 2c \sum_i \hat{y}_i + \sum_i \hat{y}_i^2 \\ \Rightarrow \frac{\partial}{\partial c} \left(\sum_{i=1}^{|N|} (\hat{y}_i - c)^2 \right) &= 2|N|c - 2 \sum_i \hat{y}_i = 0 \Leftrightarrow |N|c = \sum_i \hat{y}_i \Leftrightarrow \hat{c} = \frac{1}{|N|} \sum_{i=1}^{|N|} \hat{y}_i = \bar{y}_{\text{Node}} \end{aligned}$$

Same result for average risk $\frac{1}{|N|} \text{Remp}$

\bar{y} minimizes summed L2-Loss in Node. Plugging in \bar{y} for c we get for average risk: $\frac{1}{|N|} \sum_{i=1}^{|N|} (\hat{y}_i - \bar{y})^2$

This is identical the (biased) sample variance \Rightarrow Predicting sample mean minimizes both L2 risk and variance impurity

Quite an intuitive look at Prediction variance via L2 minimization, also connects to intercept LM trained with L2 loss within Node (piecewise for entire Tree)

That nicely explains why regression trees (without Early Stopping) grow so deep: Variance of Subsets cannot possibly be worse than variance of parent node
It's impossible to make the Regr. Tree worse by introducing a further split. Most likely it gets a lot better w.r.t. train error!

(Classification Trees) Develop intuition for Brier Score / Log Loss and their resulting Node Evaluation Metric (Gini / Entropy Impurity) for Splitting

Average: Cross-entropy loss
 a) Multidass. Log Loss in Node: $\text{PLL} = \left(\frac{1}{|N|}\right) \sum_{i=1}^{|N|} \sum_{k=1}^g I[\hat{y}_i = k] \log(\hat{\pi}_{ik}) = -\sum_{k=1}^g \left(\frac{1}{|N|} \sum_{i=1}^{|N|} I[\hat{y}_i = k]\right) \log(\hat{\pi}_{ik})$ (CART: $\hat{\pi}_{ik}$ indep. of i , const. node pred.)

Inserting $\hat{\pi}_{ik} = \frac{1}{|N|} \sum_{i=1}^{|N|} I[\hat{y}_i = k]$ we get $-\sum_{k=1}^g \hat{\pi}_{ik} \log(\hat{\pi}_{ik}) \triangleq$ Entropy impurity that we use to quantify empirical risk of single obs./pred. in Node.

Average:
 Multiclass Brier Score in Node: $\text{PBS} = \left(\frac{1}{|N|}\right) \sum_{i=1}^{|N|} \sum_{k=1}^g (I[\hat{y}_i = k] - \hat{\pi}_{ik})^2 = \sum_{k=1}^g \frac{1}{|N|} \sum_{i=1}^{|N|} (I[\hat{y}_i = k] - \hat{\pi}_{ik})^2$

Inserting $\hat{\pi}_{ik} = \frac{1}{|N|} \sum_{i=1}^{|N|} I[\hat{y}_i = k]$ yields that we can write the inner part as the variance of the binomial random variable $I[\hat{y}_i = k] \in \{0, 1\}$
 success probability $\hat{\pi}_{ik}$ "estimated" by $\hat{\pi}_{ik}$

(Biased) empirical variance of $I[\hat{y}_i = k]$: $\hat{\pi}_{ik} (1 - \hat{\pi}_{ik})$

$$\Rightarrow \text{Overall we get the Gini impurity } \sum_{k=1}^g \hat{\pi}_{ik} (1 - \hat{\pi}_{ik}) = \sum_{k=1}^g \hat{\pi}_{ik} - \sum_{k=1}^g (\hat{\pi}_{ik})^2 = 1 - \sum_{k=1}^g (\hat{\pi}_{ik})^2$$

b) Fractions of the classes $k = 1, \dots, g$ in Node N of decision tree are $\hat{\pi}_{ik}^{(N)} = \frac{1}{|N|} \sum_{(x_i^{(N)}, y_i^{(N)}) \in N} I[y_i^{(N)} = k]$

Deterministic prediction rule for hard label: $\hat{y}|N = \arg \max_k \hat{\pi}_{ik}^{(N)}$ pick class with highest empirical frequency

Randomizing rule $\hat{k} \sim \text{Cat}(\hat{\pi}_{1}^{(N)}, \dots, \hat{\pi}_{g}^{(N)})$ draw from categorical distribution parameterized by these same empirical frequencies

arbitrary most frequent class is drawn with prob. $\max_k \hat{\pi}_{ik}^{(N)}$, so in expectation we predict max class most often
 We assume train data for target $y^{(i)}$ are i.i.d. drawn from this cat. dist.: $P(y^{(i)} = k|N) = \hat{\pi}_{ik}^{(N)} \rightarrow$ stochastic pred. rule $P(\hat{y}^{(i)} = k|N) = \hat{\pi}_{ik}^{(N)}$ by design
 $y^{(i)}$ randomness obvious here, $\hat{y}^{(i)}$ randomness from training sample!

Define MCE for node with $n = |N|$ when using randomizing pred. rule: $\text{PMCE}(N) = \frac{1}{n} \sum_{i=1}^n I[\hat{y}_i^{(N)} \neq y_i^{(N)}]$

We will show that the expectation of the Misclassification Error in this setting is exactly the Gini impurity \triangleq expected freq. with which training sample (obs.)

both random, so we need joint distr.
 $\mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}}[\text{PMCE}(N)] = \mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}} \left(\frac{1}{n} \sum_{i=1}^n I[\hat{y}_i^{(N)} \neq y_i^{(N)}] \right) = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}}[I[\hat{y}_i^{(N)} \neq y_i^{(N)}]] =$
 $= \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}} \left(\sum_k \hat{\pi}_{ik}^{(N)} \cdot I[\hat{y}_i^{(N)} \neq k] \right) = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}} \left(1 - \hat{\pi}_{ik}^{(N)} \right) = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^g \hat{\pi}_{ik}^{(N)} \cdot (1 - \hat{\pi}_{ik}^{(N)})$
 $\hat{\pi}_{ik}^{(N)} = \frac{1}{n} \sum_{(x_j^{(N)}, y_j^{(N)}) \in N} I[y_j^{(N)} = k] \rightarrow \sum_{k=1}^g \hat{\pi}_{ik}^{(N)} = 1, \mathbb{E}_{y^{(1)}, y^{(2)}, \dots, y^{(n)}}[\hat{\pi}_{ik}^{(N)}] = \sum_k \hat{\pi}_{ik}^{(N)} P(y^{(i)} = k) = \sum_k \hat{\pi}_{ik}^{(N)} \hat{\pi}_{ik}^{(N)} = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^g \hat{\pi}_{ik}^{(N)} \cdot (1 - \hat{\pi}_{ik}^{(N)})$

Random Forests

Let's show that the prob. for an observation to be OOB in arbitrary bootstrap sample converges to $\frac{1}{e} \approx 0.37$

Prob of not being drawn = $1 - \frac{1}{n}$ \Rightarrow sample with replacement: $P(i \in \text{OOB}) = \left(1 - \frac{1}{n}\right)^n \lim_{n \rightarrow \infty} \left(1 - \frac{1}{n}\right)^n = e^{-1} = \frac{1}{e} \approx 0.37$

When we use OOB error* for GE-estimation then we do kind of a similar thing as a simple holdout split with $10\% = 0.37$; just more advanced with multiple component
 *Sufficiently Large n , $P(i \in \text{OOB}) < 0.37$ for small datasets!

& averaging effect