Decision and Regression Trees

Andrew Nobel

November, 2021

Review: Histogram Classification Rules

- Data set $D_n = (x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \{0, 1\}$
- Partition $\gamma = \{A_1, \ldots, A_m\}$ of feature space $\mathcal{X}$ into disjoint cells
- Membership function $\gamma(x) =$ cell A_j of γ containing x

Definition: Histogram rule $\hat{\phi}_n^\gamma(x) =$ majority vote $\{y_i : \gamma(x_i) = \gamma(x)\}$

Fact: Histogram rule $\hat{\phi}_n^\gamma$ minimizes the empirical risk (training error)

$$R_n(\phi) = n^{-1} \sum_{i=1}^{n} \mathbb{I}(\phi(x_i) \neq y_i)$$

among all decision rules ϕ that are constant on cells of partition γ

Histogram Regression Estimates

- Data set $D_n = (x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathbb{R}$
- Partition $\gamma = \{A_1, \ldots, A_m\}$ of feature space $\mathcal{X}$ into disjoint cells
- Membership function $\gamma(x) =$ cell A_j of γ containing x

Definition: Histogram rule $\hat{\varphi}_n^\gamma(x) =$ average $\{y_i : \gamma(x_i) = \gamma(x)\}$

Fact: Histogram rule $\hat{\varphi}_n^\gamma$ minimizes the empirical risk (training error)

$$R_n(\varphi) = n^{-1} \sum_{i=1}^{n} (\varphi(x_i) - y_i)^2$$

among all functions φ that are constant on cells of partition γ

Decision Trees in a Nutshell

1. A decision tree is a histogram classification rule based on a *data dependent partition*

2. The partition is characterized by a binary tree T that describes a recursive splitting of the feature space

3. Each split in the partition is simple, involving a single variable and an associated threshold

Binary Trees

Definition: A *binary tree* T is a (directed, acyclic) graph such that

1. T has a distinguished node t_0 called the **root** with no parent

2. Every other node $t \in T$ has one parent and zero or two children

   - Nodes with two children are called **internal**, denoted by T^o
   - Nodes with no children are called **leaves**, denoted by ∂T

Note: Tree usually drawn upside-down, with root node at the top. If T has more than one node, the root node is internal

Labeled Binary Tree

Suppose $\mathcal{X} = \mathbb{R}^p$. A *labeled binary tree* is described by

1. A binary tree T
   - root node t_0
   - interior nodes T^o
   - leaves ∂T
2. For each internal node $t \in T^o$
   - a splitting feature/variable $j_t \in \{1, \ldots, p\}$
   - a splitting threshold $\tau_t \in \mathbb{R}$

Tree-Structured Partition

Each node t of a labeled binary tree corresponds to a region $A(t) \subseteq \mathbb{R}^p$

Regions are recursively defined, starting at the root

- $A(t_0) = \mathbb{R}^p$.
- For $t \in T^o$ region $A(t)$ split on component j_t at threshold τ_t yielding

$$A(t_l) = A(t) \cap \{x : x_{j_t} \leq \tau_t\} \text{ and } A(t_r) = A(t) \cap \{x : x_{j_t} > \tau_t\}$$

Fact: The terminal regions $\{A(t) : t \in \partial T\}$ associated with the leaves of T partition $\mathbb{R}^p$. This is called the partition generated by T

Example: Regression Tree (ESL)

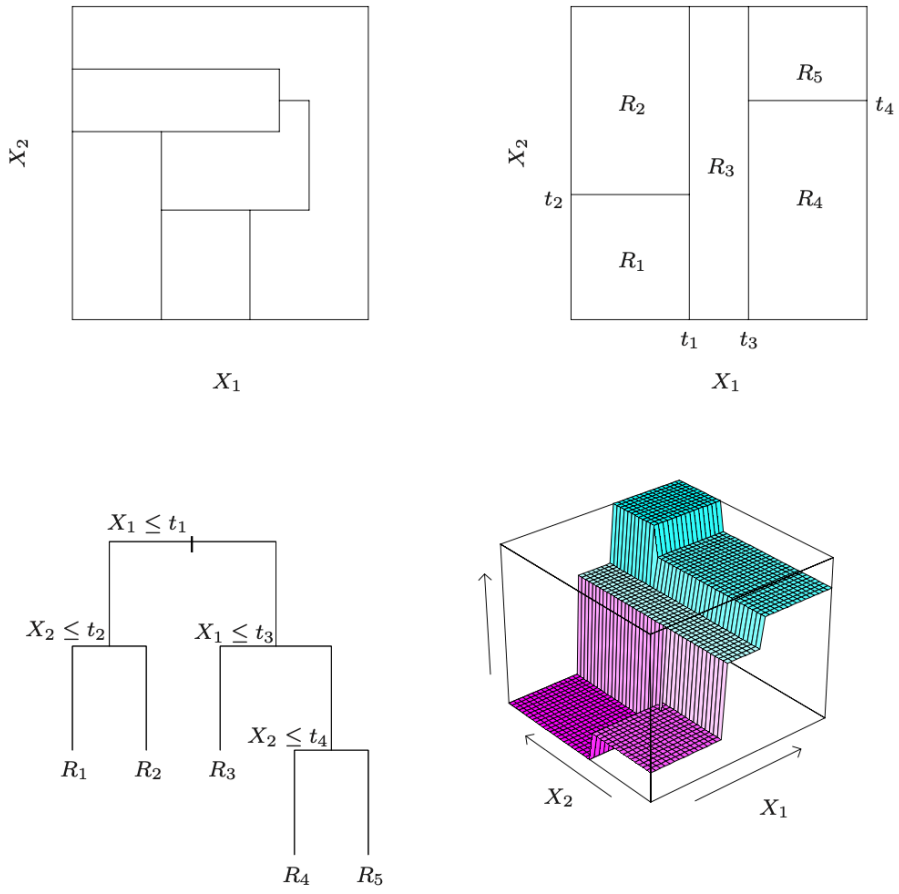

FIGURE 9.2. *Partitions and CART. Top right panel shows a partition of a two-dimensional feature space by recursive binary splitting, as used in CART, applied to some fake data. Top left panel shows a general partition that cannot be obtained from recursive binary splitting. Bottom left panel shows the tree corresponding to the partition in the top right panel, and a perspective plot of the prediction surface appears in the bottom right panel.*

Decision Trees

Definition: A decision tree is a histogram classification rule that is based on the partition of a labeled binary tree T. Formally,

$$\phi_T(x) \;=\; \sum_{t \in \partial T} c(t)\, \mathbb{I}(x \in A(t))$$

where $c(t) \in \{0, 1\}$ is the class associated with terminal region $A(t)$.

Outline: How to obtain a decision tree from data D_n?

- Use D_n to grow a large labeled tree T_0 in a greedy fashion
- Prune large tree T_0 to get a "right-sized" tree T
- Assign each terminal region of T to its majority class

Impurity Measures for Regions

Given: Data $(x_1, y_1), \ldots, (x_n, y_n)$ and region $A \subseteq \mathcal{X}$ define

- $|A| =$ number of $x_i \in A$
- $\hat{p}(A) = |A|^{-1} \sum_{x_i \in A} \mathbb{I}(y_i = 1) =$ fraction of points in A with $y_i = 1$

Impurity Measures: Let $\hat{p} = \hat{p}(A)$. Standard measures include

1. Misclassification rate $M(A) = \min(\hat{p}, 1 - \hat{p})$
2. Gini index $G(A) = \hat{p}\,(1 - \hat{p})$
3. Entropy $H(A) = -\hat{p} \log \hat{p} - (1 - \hat{p}) \log(1 - \hat{p})$

Idea: Impurity measures quantify extent to which A contains points from one class. Small when $\hat{p}$ close to 0 or 1, maximized at $p = 1/2$.

Impurity Reduction from Region Splitting

Definition: If region A split into A_l and A_r the *reduction* in misclassification rate is given by

$$\Delta_M \;=\; M(A) - \left[\frac{|A_l|}{|A|}M(A_l) + \frac{|A_r|}{|A|}M(A_r)\right]$$

Changes Δ_G and Δ_H for Gini and entropy measures are defined similarly

Fact: The quantities Δ_M, Δ_G, and Δ_H are non-negative. Better splits are associated with larger values of Δ_*

Tree Growing from Data

Fix in advance

- Data set $D_n = (x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^p \times \{-1, 1\}$
- Impurity measure M, G, or H with associated reduction measure Δ
- Count threshold n_0: only split regions with at least n_0 points
- Reduction threshold Δ_0: only implement splits with $\Delta > \Delta_0$

Tree Growing from Data

Initialize: Let $T := \{t_0\}$ with $A(t_0) = \mathbb{R}^p$.

Iterate: For each leaf $t \in \partial T$ with $|A(t)| \geq n_0$ do

1. For each $j \in \{1, \ldots, p\}$ and each $\tau \in \mathbb{R}$ find change Δ for the split

$$A(t) \cap \{x : x_j \leq \tau\} \text{ and } A(t) \cap \{x : x_j > \tau\}$$

2. Identify optimal variable $j(t)$ and threshold $\tau(t)$

3. If optimal $\Delta \leq \Delta_0$ stop. Otherwise add leaves t_l and t_r to node t
   - assign $A(t_l) = A(t) \cap \{x : x_{j(t)} \leq \tau(t)\}$
   - assign $A(t_r) = A(t) \cap \{x : x_{j(t)} > \tau(t)\}$

Output: Baseline tree T_0

Pruning to Balance Fit and Complexity

Let $T \leq T_0$ be a binary subtree of T_0 with the same root

- Leaf regions $\{A(t) : t \in \partial T\}$ of T partition feature space $\mathcal{X}$
- Majority voting in terminal regions of T yields decision tree ϕ_T
- Measure *fit* of ϕ_T by its empirical risk $\hat{R}_n(T)$
- Measure *complexity* of ϕ_T by $|T| =$ number of nodes in T

Definition: For each $\lambda \geq 0$ define subtree

$$T_\lambda \;=\; \underset{T \leq T_0}{\operatorname{argmin}} \left\{ \hat{R}_n(T) + \lambda |T| \right\}$$

Tree T_λ offers optimal balance of fit and complexity with weight λ

Cost-Complexity Pruning

Focus: Subtrees $T_\lambda = \text{argmin}_{T \leq T_0} \left\{ \hat{R}_n(T) + \lambda |T| \right\}$ for $\lambda \geq 0$

- If $T_a \leq T_b$ then $|T_a| \leq |T_b|$, while $\hat{R}_n(T_a) \geq \hat{R}_n(T_b)$
- $T_0 =$ full tree T_0. For λ is sufficiently large, $T_\lambda = \{\text{root}\}$

Fact: The trees T_λ are nested: $\lambda_1 \leq \lambda_2$ implies $T_{\lambda_2} \leq T_{\lambda_1}$. Full sequence $\{T_\lambda : \lambda \geq 0\}$ can be found by successively removing nodes with small Δ_M

Upshot: Given large initial tree T_0

- Choose value $\tilde{\lambda}$ of penalty λ using cross validation
- Final rule is decision tree $\hat{\phi}^{\text{tree}}$ associated with $T_{\tilde{\lambda}}$

Example: Spam Data (ESL)

Data: 4061 messages classified as “email” or “spam”

Features: Each message has 57 predictors

- percentage of words matching those on a list (48)
- percentage of characters matching characters on a list (6)
- counting sequences of capital letters (3)

Decision Tree

- grow with entropy based impurity, training set 3065 samples
- penalty parameter $\hat{\lambda}$ chosen by 10-fold CV
- use test set to evaluate node impurity, overall performance

Example: Pruned Tree for Spam Detection (ESL)

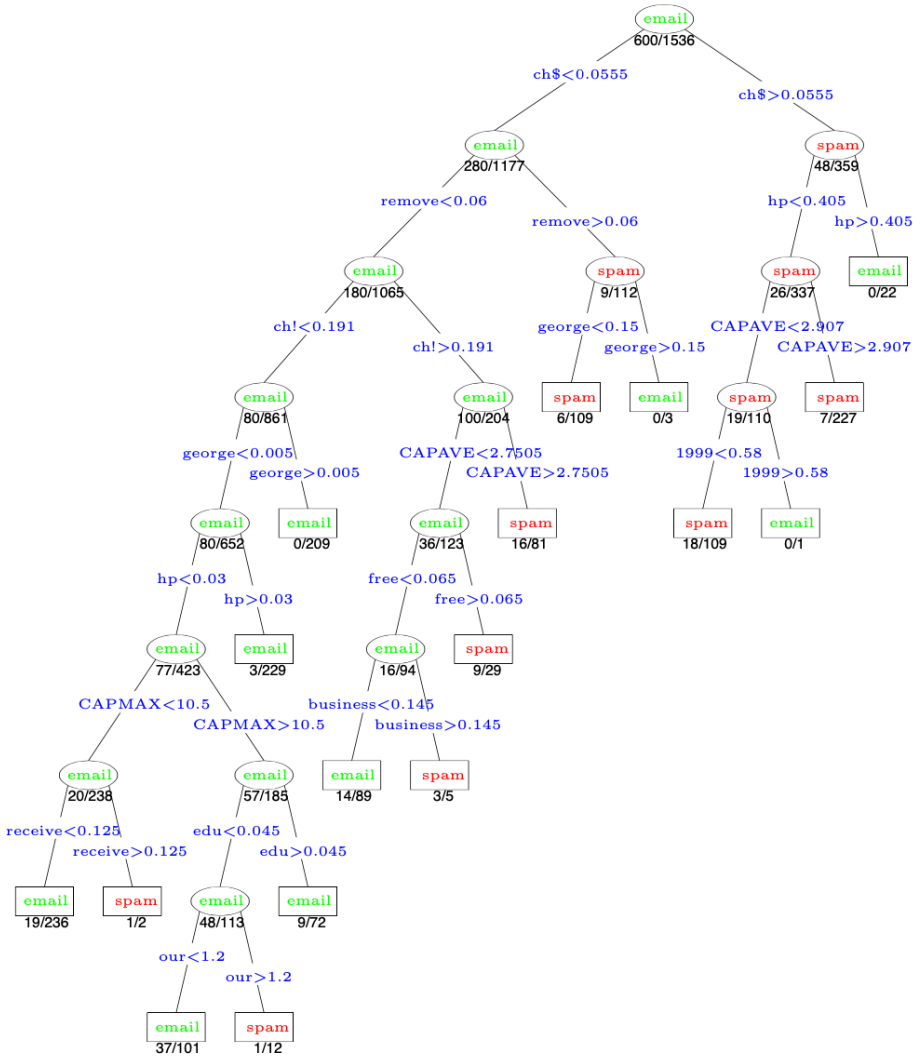

FIGURE 9.5. *The pruned tree for the* `spam` *example. The split variables are shown in blue on the branches, and the classification is shown in every node. The numbers under the terminal nodes indicate misclassification rates on the test data.*

Estimating Conditional Probabilities

Idea: Given data $D_n = (X_1, Y_1), \ldots, (X_n, Y_n)$ and decision tree T with terminal regions

$$\gamma = \{A(t) : t \in \partial T\}$$

Rather than poll class labels in each terminal region, we may estimate $\eta(x) = \mathbb{P}(Y = 1 \,|\, X = x)$ by averaging the responses, that is

$$\hat{\eta}(x) \;=\; \text{average}\{y_i : \mathbb{I}(\gamma(x_i) = \gamma(x))\}$$

Note that corresponding decision tree rule is $\hat{\phi}^{\text{tree}}(x) = \mathbb{I}(\hat{\eta}(x) \geq 1/2)$

Regression Trees

1. Growing. Grow large initial tree T_0 from data using impurity measure

$$U(A) = \sum_{x_i \in A} (y_i - \overline{y}_A)^2 \quad \text{with} \quad \overline{y}_A = |A|^{-1} \sum_{x_i \in A} y_i$$

2. Pruning. Consider optimal cost-complexity subtrees

$$T_\lambda = \operatorname*{argmin}_{T \leq T_0} \{R_n(T) + \lambda |T|\} \quad \lambda \geq 0$$

where $R_n(T)$ is empirical risk of histogram regression rule based on T

3. Output. Use cross validation to select penalty $\hat{\lambda}$ and associated tree $T_{\hat{\lambda}}$. Final rule is regression tree $\hat{\varphi}^{\text{tree}}$ based on $T_{\hat{\lambda}}$

Classification and Regression Trees

Pluses

- Growing and pruning are computationally efficient
- Trees are readily interpretable
- Can accommodate ordinal and categorical features

Minuses

- Unstable: a small change in the data can lead to a very different rule
- Histogram rules are piecewise constant, not smooth
- Some simple rules not well-captured by tree-structured partitions