Decision Trees

COMPSCI 371D — Machine Learning
Outline

1 Motivation
2 Recursive Splits and Trees
3 Prediction
4 Purity
5 How to Split
6 When to Stop Splitting
Motivation

Linear Predictors $\rightarrow$ Trees $\rightarrow$ Forests

- **Linear predictors:**
  - Few parameters $\rightarrow$ Good generalization, efficient training
  - Convex risk $\rightarrow$ Unique minimum risk, easy optimization
  - Score-based $\rightarrow$ Measure of confidence
  - Few parameters $\rightarrow$ Limited expressiveness:
    - Regressor is an affine function
    - Classifier is a set of convex regions in $X$

- **Decision trees:**
  - Score based (in a sophisticated way)
  - Arbitrarily expressive: Flexible, but generalizes poorly

- **Random decision forests:**
  - Ensembles of trees that vote on an answer
  - Expressive (somewhat less than trees), generalize well
Splitting $X$ Recursively

Recursive Splits and Trees

COMPSCI 371D — Machine Learning
Decision Trees

4 / 17
A Decision Tree

Choose splits to maximize purity
What’s in a Node

- **Internal:**
  - Split parameters: Dimension \( j \in \{1, \ldots, d\} \), threshold \( t \in \mathbb{R} \)
  - Pointers to children, corresponding to subsets of \( T \):
    \[
    L \overset{\text{def}}{=} \{(x, y) \in S \mid x_j \leq t\} \\
    R \overset{\text{def}}{=} \{(x, y) \in S \mid x_j > t\}
    \]
- **Leaf:** Distribution of training values \( y \) in this subset of \( X \):
  - \( p \), discrete for classification, histogram for regression
- At inference time, return a *summary* of \( p \) as the value for the leaf
  - Mode (majority) for a classifier
  - Mean or median for a regressor
  - (Remember \( k \)-NN?)
Why Store $p$?

- Can’t we just store $\text{summary}(p)$ at the leaves?
- With $p$, we can compute a confidence value
- (More important) We need $p$ at every node during training to evaluate purity
Prediction

function y ← predict(x, τ, summary)
    if leaf?(τ) then
        return summary(τ.p)
    else
        return predict(x, split(x, τ), summary)
    end if
end function

function τ ← split(x, τ)
    if x_{τ,j} ≤ τ.t then
        return τ.L
    else
        return τ.R
    end if
end function
Design Decisions for Training

- How to define (im)purity
- How to find optimal split parameters $j$ and $t$
- When to stop splitting
Purity: The Gini Index

- $S$: subset of $T$ that reaches the given node
- $i(S) = 1 - \sum_{y \in Y} p^2(y|S)$ where $p(y|S) = \frac{1}{|S|} \sum_{(x_i,y_i) \in S} \mathbb{I}(y_i \approx y)$
- Measures the empirical risk for the stochastic predictor $y = h_{\text{Gini}}(x) = y$ with probability $p(y|S(x))$
- True answer $y \rightarrow$ error probability $\approx 1 - p(y|S)$, so $L_S(h_{\text{Gini}}) = \sum_{y \in Y} p(y|S)(1 - p(y|S)) = 1 - \sum_{y \in Y} p^2(y|S) = i(S)$
- Not the only option. For classifiers, a simpler option is $i(S) = \overline{\text{err}}(S) = 1 - \max_y p(y|S)$
How to Split

• Split at training time:
  If training subset $S$ made it to the current node,
  put all samples in $S$ into either $L$ or $R$ by the split rule
• Split at inference time: Send $x$ either to $\tau.L$ or to $\tau.R$
• Either way:
  • Choose a dimension $j$ in $\{1, \ldots, d\}$
  • Choose a threshold $t$
  • Any data point for which $x_j \leq t$ goes to $\tau.L$
  • All other points go to $\tau.R$
• How to pick $j$ and $t$?
How to Pick $j$ and $t$ at Each Node?

- **Try all possibilities and pick the best**
- "Best:" Maximizes the decrease in impurity:
  \[
  \Delta i(S, L, R) = i(S) - \left( \frac{|L|}{|S|} i(L) + \frac{|R|}{|S|} i(R) \right)
  \]
- "All possibilities:" Choices are finite in number
  - Sorted unique values in $x_j$ across $T$: $x_j^{(0)}, \ldots, x_j^{(u_j)}$
  - Possible thresholds: $t = t_j^{(1)}, \ldots, t_j^{(u_j)}$
    where $t_j^{(\ell)} = \frac{x_j^{(\ell-1)} + x_j^{(\ell)}}{2}$ for $\ell = 1, \ldots, u_j$
  - Nested loop: for $j = 1, \ldots, d$
    for $t = t_j^{(1)}, \ldots, t_j^{(u_j)}$
  - Efficiency hacks are possible
Stopping too Soon is Dangerous

- Temptation: Stop when impurity does not decrease
When to Stop Splitting

- Possible stopping criteria
  - Impurity is zero
  - Too few samples would result in either $L$ or $R$
  - Maximum depth reached
- Overgrow the tree, then prune it
- There is no optimal pruning method
  (Finding the optimal tree is NP-hard)
  (Reduction from set cover problem, Hyafil and Rivest)
- Better option: *Random Decision Forests*
Summary: Training a Decision Tree

- Use exhaustive search at every node to find the dimension \( j \) and threshold \( t \) that splits \( T \) with the biggest decrease in impurity.
- Store \( j \) and \( t \) at the root of the tree.
- Make new children with \( L \) and \( R \).
- Repeat until some criterion is met.
Summary: Predicting with a Decision Tree

- Use \( j \) and \( t \) at the root \( \tau \) to see if \( x \) belongs in \( \tau.L \) or \( \tau.R \).
- Go to the appropriate child.
- Repeat until a leaf is reached.
- Return \( \text{summary}(p) \).
- \( \text{summary} \) is majority for a classifier, mean or median for a regressor.
When to Stop Splitting

From Trees to Forests

- Trees are flexible $\rightarrow$ good expressiveness
- Trees are flexible $\rightarrow$ poor generalization
- Pruning is an option, but messy
- *Random Decision Forests* let several trees vote
- Use the bootstrap to give different trees different views of the data
- Randomize split rules to make trees even more independent