Decision Tree Learning from Scratch

Stefano Fochesatto

MATH 692 Mathematics for Machine Learning

2/10/2022

Presentation Outline

- What is a Decision Tree?
- Motivations and Prevalence.
- Training the Tree.
- Advantages and Pitfalls.
- Code Demo.
- Application in Ensemble Models.

What is Decision Tree Learning?

- Supervised Learning.
- A flowchart where internal nodes represent a test for the data.
- Leaf nodes apply classification label or regression.
- Derived from recursive partitioning.
- We will discuss classification mainly,

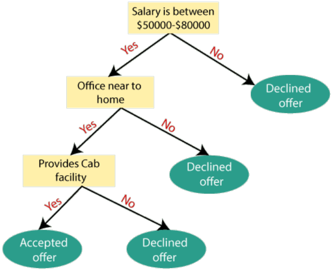

Figure: Decision Tree Example

What is Decision Tree Learning?

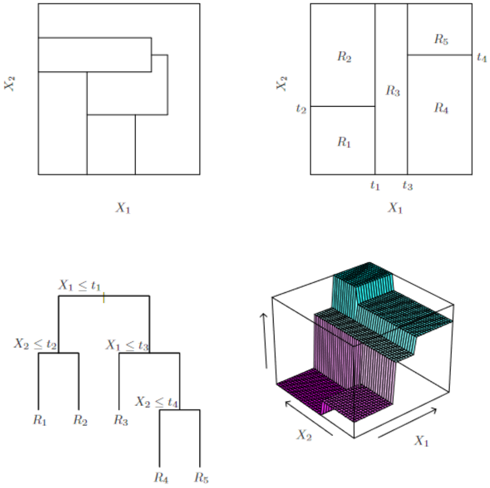

Figure: E.S.L. Friedman, Hastie, Tibshirani

Training the Tree.

- Methods don't guarantee the optimal solution (Greedy).
- Top-Down Induction of Decision Trees (R.Quinlan)
- There are several algorithms for training.
  - CART (L.Breimann)
  - ID3 and C4.5 (R.Quinlan)
  - C5.0 (R.Quinlan)

Information Theory

- Consider the following splits,

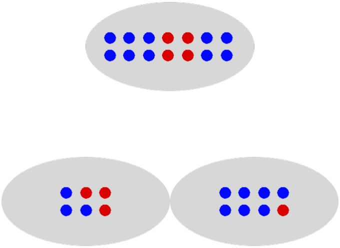

Figure: Example Split # 1

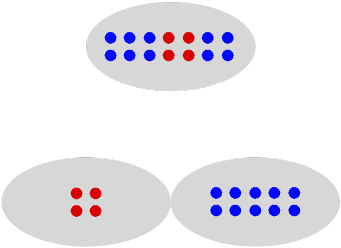

Figure: Example Split # 2

- To optimize our tree we need to be able to quantify the quality of a split (or generalized state)?

Information Theory

- Let X(observation) be a discrete random variable with n(classes) possible outcomes and pmf p(x)(prob distribution at node).

- In information theory, the unit of information ascribed to an outcome $x \in n$ is a log measure of 1/p(x),

$$I = \log\left(\frac{1}{p(x)}\right) = -\log(p(x)).$$

- Events that are rare have more information, events that are common have less information.

Information Theory

- We want to know the expected information at a node over all outcomes (classification classes),

$$\mathbb{E}(I) = \sum_{i=1}^{n} p(i) \log_2 \left(\frac{1}{p(i)} \right) = -\sum_{i=1}^{n} p(i) \log_2 (p(i)).$$

- We want to find the split which maximizes ΔI or information gain,

$$\Delta I = \mathbb{E}(I)_{Parent} - \sum w(i) \mathbb{E}(I)_{Children}.$$

- Where $w(i)$ is the size of the child node relative to the parent node.
- Sometimes we only care about the difference between splits.

$$\Delta I = 1 - \sum w(i) \mathbb{E}(I)_{Children}.$$

Training the Tree

- For the CART algorithm the Gini Impurity is used to evaluate the quality of a node,

$$Gini = 1 - \sum_{i=1}^{n} p(i)^2.$$

- Again evaluating a split we take a weighted sum,

$$Gini_{split} = \sum Gini_{children}.$$

- Both methods are largely the same, Gini is preferred for predictive performance and computational complexity.

Training the Tree

- BuildTree(node);
  - if statement for stopping criteria;
  - Best_gini = gini(node.observations)
  - Best_feature
  - Best_threshold
  - loop through features;
    - sort all observations by current feature;
    - loop through observations;
    - left_gini = gini(node.observations[1:i])
    - right_gini = gini(node.observations[i+1:n])
    - split_gini = (i/n)left_gini + ((i+1 - n)/n)right_gini
    - if split_gini < Best_gini; Best_Gini = split_gini; Best_feature = j; Best_threshold = i
  - BuildTree(node.left)
  - BuildTree(node.right)