# Lecture Note - 06: Tree, CART, Random Forest, Gradient Boosting Machine

Dihui Lai

March 29, 2020

## Contents

**1 Classification and Regression Tree (CART) 1**

1.1 Impurity Measurement . . . . . 1

1.2 Model . . . . . 2

1.3 Model: How to Grow a Tree? . . . . . 2

1.4 Gain at a Split Node . . . . . 3

**2 Random Forest 3**

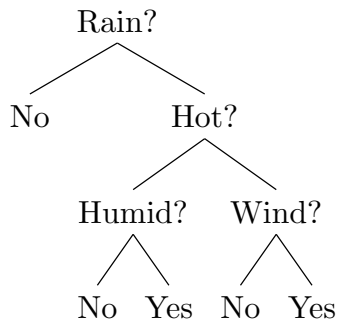
**3 GBM 4**

# 1 Classification and Regression Tree (CART)

## 1.1 Impurity Measurement

At node $t$, the fraction of class $i$ is denoted as $p(i|t)$

Entropy: $H(t) = -\sum_{i=1}^{C} p(i|t) \log(p(i|t))$




Gini-Index: $Gini(t) = \sum_{i=1}^{C} [p(i|t)]^2$

Classification Error: $Error(t) = 1 - \max_i p(i|t)$

Considering the entropy of a data set that contains two types of outcomes: 0, 1. Running 10 experiments, we get the following out comes (0, 0, 1, 0, 1, 0, 1, 1, 0, 1). The entropy is

$$H(t) = -\sum_{i=1}^{2} \frac{1}{2} \log_2(\frac{1}{2}) = 1$$

What if we have out come of (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)? The entropy is

$$H(t) = 1 \log_2(1) + 0 \log_2(0) = 0$$

The less variance in the data, the smaller the entropy is. For binary outcome, a probability of 1/2 leads to the largest uncertainty/entropy.

## 1.2 Model

John likes playing tennis on Saturday, The liklihood of John playing tennis depends on the weather. John has never played on a rainy. He may or may not play on a windy day. Given the weather, can you predict if John will play tennis this weekend?

## 1.3 Model: How to Grow a Tree?

- At each node, the tree algorithm will check every variable for a possible split.
- A variable is selected for the split if it maximally reduces the impurity in the child nodes (e.g. the largest reduction of entroy, the largest reduction of variance etc.)
- Pruning: a tree can grow till each leave only contains one data point. Pruning the tree is needed to avoid overfitting.

Stopping criterion

- The number of training sample after split is less than certain number 'min_ samples_ split'.
- When the depth of the tree reaches to certain threshold 'max_ depth'.

### 1.4 Gain at a Split Node

Assuming we collect John's tennis activities for 20 weeks. Out of the 20 weekends, we have 8 sunny day and John plays tennis on all of them. Out of the rest 12 days, John played tennis on 7 days. Before any split, we have the entropy calculated as

$$H(root) = -\frac{15}{20}\log(\frac{15}{20}) - \frac{5}{20}\log(\frac{5}{20}) = 0.562$$

Split based on the weather, we have on sunny days $H(sunny) = 0$ and on other days $H(!sunny) = -\frac{7}{12}\log(\frac{7}{12}) - \frac{5}{12}\log(\frac{5}{12}) = 0.679$ Therefore the child nodes have average entropy of

$$H(childs) = \frac{8}{20}\cdot 0 + \frac{12}{20}\cdot 0.679 = 0.407$$

We have entropy reduced by $\Delta H = 0.562 - 0.390 = 0.154$

Instead of splitting by forcast, we look at the wind speed. Out of the 20 weekends, 10 days are windy and 10 days are not windy. Out of the windy day, John played 6 times and 9 times for the non-windy days.

Split based on the wind condition, we have on windy days $H(!windy) = -\frac{6}{10}\log(\frac{6}{10}) - \frac{4}{10}\log(\frac{4}{10}) = 0.673$ and on other days $H(windy) = \frac{9}{10}\log(\frac{9}{10}) + \frac{1}{10}\log(\frac{1}{10}) = 0.325$

Therefore the child nodes have average entropy of

$$H(childs) = \frac{1}{2}0.673 + \frac{1}{2}0.325 = 0.499$$

We have entropy reduced by $\Delta H = 0.562 - 0.5 = 0.063$

## 2 Random Forest

It terms out that multiple trees together can produce better model than a single tree.

In order to build a random forest, do the following to grow each individual tree.

- use a random sub-sample with replacement, to grow each individual tree.

- At each node, $M << m$ columns are randomly selected, out of all m predictors. Usually, the default choice of $M = m/3$ for regression tree and $M = \sqrt{m}$ for classification tree.
- Each tree is grown to the largest extent possible (one sample in the leaves)

# 3 GBM

The Gradient boosting tree is a collection of trees, where each individual tree is constructed to predict the 'residuals'. $\hat{y}(x_i) = \sum_{k=1}^{K} \gamma_k h_k(x_i)$, where $h_k(x)$ is a tree.

The $k^{th}$ tree is built to fit the 'residuals' from the precedent $k-1$ trees

The algorithm in a nutshell

Step 1:

$$\hat{y}_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y^i, \gamma)$$

Step 2: Fit a tree model $h_k$ by training it on the dataset $[\vec{x}^i, r_k^i]$

Here,

$$r_k^i = -\left[\frac{\partial L(y^i, y(x^i))}{\partial y(x^i)}\right]_{y(x)=\hat{y}_{k-1}(x)} \quad \text{for } i = 1, \ldots, n.$$

Step 3: Update the estimator $\hat{y}_k(x) = \hat{y}_{k-1}(x) + \gamma_k h_k(x)$

Here, $\gamma_k = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y^i, \hat{y}_{k-1}(x^i) + \gamma h_k(x^i)\right)$

Reference:

Xgboost