Decision Trees

1 Hypothesis

Consider the domain $\mathcal{X} = \mathbb{R}^D$ and the codomain $\mathcal{Y} = [1, M]$. In Decision Tree based classification, our hypothesis $h_\theta : \mathcal{X} \to \mathcal{Y}$ is defined by a (almost always binary) tree which partitions the feature space into regions with consistent labels. Inner nodes of the tree consist of decisions, which partition the data based on some threshold of some feature. Each leaf represents a label with which the data point will be classified.

2 Decisions

For each subset of data, we choose the split that minimizes the "cost" of the two new subsets. To choose a split, we check all features and all thresholds for each (which are the values in the data points themselves). Thus, we have

$$(f^*, t^*) = \arg \min_{f \in [1,M]} \min_{t \in \mathcal{T}_f} \text{cost}(S_l) + \text{cost}(S_r)$$
$$\mathcal{T}_f = \{x_i^{(f)} : \forall \mathbf{x}_i \in S\}$$
$$S_l = \{(\mathbf{x}_i, y_i) : x_i^{(f)} \leq t\}$$
$$S_r = \{(\mathbf{x}_i, y_i) : x_i^{(f)} > t\}$$

In other words, $\mathcal{T}_f$ are all values of a feature f, S_l is the "left" set and S_r is the "right" set.

Similar to loss, the cost is some metric that measures how "ineffective" the subset is at providing useful information. By defining the probability that a data point belongs to a certain class c in a subset of data S_j with

$$p_c(S_j) = \frac{1}{|S_j|} \sum_{i \in S_j} \mathbb{I}(y_i = c)$$

(in other words, the ratio between data points of that class and all data points), we can define various cost metrics, some of which are detailed below.

- **Misclassification rate:** determines the proportion of data points which are not the majority class of the subset. Is given by

$$\text{cost}(S_j) = \frac{1}{|S_j|} \sum_{i \in S_j} \mathbb{I}(y_i \neq \hat{y}) = 1 - p_{\hat{y}}(S_j)$$

$$\hat{y} = \arg \max_c p_c(S_j)$$

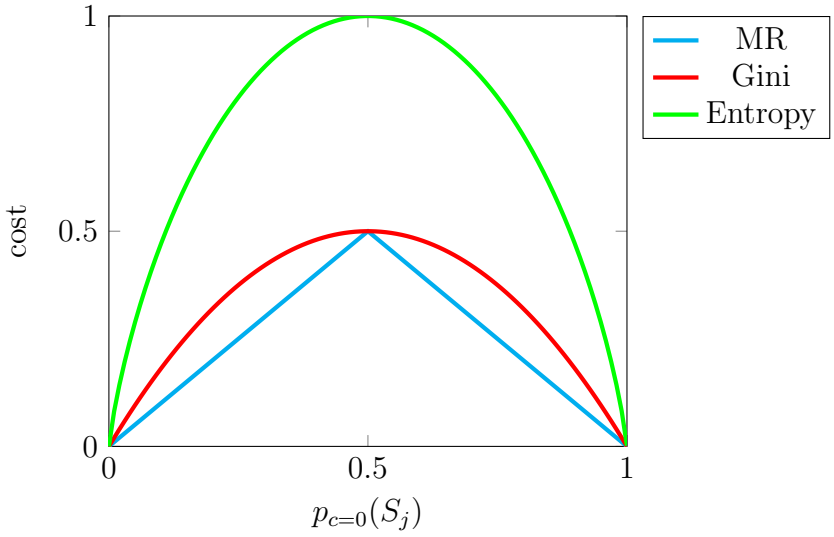
- **Entropy:** describes how much "uncertainty" there is in a subset (how non-homogeneous it is). Is given by

$$\text{cost}(S_j) = -\sum_{c \in \mathcal{Y}} p_c(S_j) \log p_c(S_j)$$

- **Gini index:** measures how often some element in the subset would be incorrectly labeled if it were randomly labeled, according to the distribution of labels. Is given by

$$\text{cost}(S_j) = \sum_{c \in \mathcal{Y}} p_c(S_j)(1 - p_c(S_j))$$

Below is a plot of each cost function for a subset S_j with two classes, 0 and 1.



3 Overfitting

Due to the complexity of the hypothesis class (there are 2^{2^D} possible Decision Trees with D Boolean features, for example), overfitting can become an issue. One approach is limiting the maximum depth of the trees, but that can cause underfitting instead. Better approaches are the use of bagging or boosting instead.

A bagging technique specific to Random Forests (the name given to the ensemble of Decision Trees created with either bagging or boosting) is to pick split points (i.e. make decisions) randomly instead of optimally. Thus, f^* and t^* are instead picked as such

$$j^* \sim U(\{1, \ldots, D\}) = p(j) = \frac{1}{D}$$

$$j^* \sim U(\mathcal{T}_f) = p(t) = \frac{1}{|\mathcal{T}_f|}$$