# 09 - Decision Trees and Forests

Ethan Graham

20th June 2023

## General Idea

we make a binary tree wherein the nodes are weak learners.

We use a few different types of weak learners: - **axis aligned** (vertical line / aligned with coorinates). We split along a single feature - **oriented line** Considers combinations of features. More expressive but also more expensive - **conic section** Can capture non-linear relations in features. More expensive and more prone to overfitting

## Entropy and Gini Index

We have a few metrics for measuring uncertainty/entropy. **Gini Index:** $Q(S) = \sum_{k \in K} p^k(1 - p^k)$

**Entropy:** $Q(S) = -\sum_{k \in K} p^k \, ln(p^k)$

Both of these are maximal when $p = \frac{1}{2}$ and minimal ($= 0$) when $p = 1$

$\rightarrow$ minimizing these favors leaves in which a large fraction of samples belong to the same class.

**At each node, pick the learner that maximizes the information gain** This strategy works well for decision trees, but not for AdaBoost. Here's why

**Decision Trees:** Decision trees use metrics like information gain to choose the *best split* at every node. The goal is to maximize the homogeneity of both of the resulting child nodes by splitting them in two. **This works because we recursively partition based on individual features.**

**AdaBoost:** Adaboost, instead of recursively partitioning into subclasses using weak learners, combines them all together as an ensemble. **If we were to choose a weak learner that maximized information gain, we would very quickly overfit our problem**. Instead, we focus on minimizing the exponential loss function.

## Using multiple trees

Using multiple trees can increase robustness of the model. We then call it a **Decision Forest**.

We combine them in the following way: $p(c|\mathbf{v}) = f\left(p_1(c|\mathbf{v}) + ... + p_T(c|\mathbf{v})\right)$ Where $T$ is the number of trees in the forest.

We train the different trees on **different random subsets** of the training set. We call this *bagging*

This ensures that the trees won't all react the same way.

Assuming that all the trees are independant (because all training-subsets are disjoint) We can state that $p(c|v) \propto p_1(c|v) \cdot ... \cdot p_T(c|v)$

Our loss function becomes $L(c, v) = \frac{1}{T} \sum_t -log(p_t(c|v))$

## Relationship to Boosting

Boosting is essentially just a decision tree, but with one long cascading branch. This is because we always apply the same weak learners for every new iteration. Random Forests are a lot less deep and more balanced.