Math 345 - Notes
Decision Trees
November 5, 2018

Decison Trees

In this section we discuss a supervised classification algorithm (ID3) for constructing a decision tree. The idea is to break up big decisions into a series of questions. The input data are d-dimensional vectors, and the output is a category (class or label). The decision **nodes** are questions or attributes and the **leaves** are decisions or labels/classes.

Entropy $H(X)$ and information gain $IG(X, A)$

To decide which tree structure to use, we try to minimize entropy at every step and hence maximize information gain. Consider the data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, with y_k $(1 \leq k \leq N)$ in the set of labels/classes $\{c_1, \cdots, c_L\}$.
Suppose X is a subset of $\{y_1, \cdots, y_N\}$. Intuitively, the entropy of the set X measures how mixed up a set is, with larger $H(X)$ meaning the set is better mixed. Formally, if we let

$$p_k = \text{proportion of data in } X \text{ labeled as class } c_k,$$

we define the entropy of the set X

$$H(X) = -p_1 \log_2(p_1) - p_2 \log_2(p_2) - \cdots - p_L \log_2(p_L) = -\sum_{k=1}^{L} p_k \log_2(p_k).$$

Since $\lim_{t \to 0} t \log_2(t) = 0$, we use the convention that $0 \log_2(0) = 0$.

The information gained $IG(X, A)$ measures how much we learn about X once we know the attribute A. Formally,

$$IG(X, A) = H(X) - H(X|A),$$

that is, the information gained is the difference between the *parent entropy* and the weighted sum of *entropy from children*. Here,

$$H(X|A) = \sum_{\text{all } a} P(A = a) H(X|A = a) = -\sum_{\text{all } a} P(A = a) \left[\sum_{k=1}^{L} P(X = c_k | A = a) \log_2(P(X = c_k | A = a)) \right]$$

That is, $H(X|A)$ measures how mixed the labels are once we know the value of the attribute A.

ID3 Algorithm

The algorithm will be built so that at every step we maximize the information gained, or in other words we minimize the weighted entropy from the children. In other words, we want to split the data into branches where the labels are not very mixed, hence the subsets are more and more similar.

ID3 Algorithm:

1. At $t = 0$, initialize X at the root node with $X = \mathcal{D}$
2. For $t \geq 0$, for all attributes A not in an ancestor node,
   - compute $H(X|A)$ and/or $IG(X, A)$
   - pick the attribute A with lowest $H(X|A)$ (or largest $IG(X, A)$)
   - split set X according to attribute A
   - continue recursively by letting X be the set of labels in each subtree, until all attributes have been considered.
3. Stop on a subset when:
   - all elements in X belong to the same class $\Rightarrow$ leaf labeled with *common class*
   - there are no more attributes to consider $\Rightarrow$ leaf labeled with *most common class in* X
   - the set is empty $\Rightarrow$ leaf labeled with *most common class of parent*

Remarks:

- this is a greedy algorithm
- the solution is not necessarily optimal, so you may need to run it a few times to get a good outcome
- it can easily overfit the data, so a few work arounds are:
  - the tree may need to be "pruned"
  - one can work with *random forests*
  - build multiple trees and let them "vote" on how to classify inputs
  - build trees on different training data to get different trees
  - at every node, randomly select a subset of attributes for splitting the set