Decision Tree - Overview

Huyen Do

May 11, 2010

1 Decision Tree - General framework

Decision tree learning is a method for approximating discrete valued target functions [it is also extended for continuous values], in which the learned function is represented by a decision tree. Learned trees can also be represented as sets of if-then rules (to improve human readability). Decision tree learning methods search a completely expressive hypothesis space and thus avoid the difficulties of restricted hypothesis spaces. The hypothesis spaces of DT is defined over feature set, which is all possibility to assign class to each partitions of a feature space, creating by feature values. In discrete case, the number of all partitions in case of having m features, each feature has f_i discrete values, $i = 1..m$, is: $\Pi_{i=1}^{m}(f_i + 1)$. Therefore if there are c classes, the hypothesis space contains $c\Pi_{i=1}^{m}(f_i + 1)$ different hypothesis. The continuous value can be reduced to discrete case using threshold. Obviously, discrete search in this complete space is a combinatorial optimization problem. Moreover with a small number of training instances, there are a lot of hypothesis which are consistent with the training set, and may be not consistent with the test set. Therefore Decision Tree adds some more inductive bias. DTs prefer hypothesis which optimize some criteria. Different criteria lead to different DT algorithms. The most popular optimizing criterion is: minimize the entropy of the instance sets.

The final goal is to minimize misclassification cost. One of the cost can be least squared mean error:

$$\min_{h \in \mathcal{H}} \sum_{i}^{n} (h(x) - y(x))^2$$

where $\mathcal{H}$ is hypothesis space, $y(x)$ is class label, n is number of instances.

Least squared error is proportional to Entropy. (proof?)

Still, optimizing the least squared error is combinatorial. Different greedy search methods have been developed to solve the problem. Greedy search bases on some criteria such as Information Gain or Information Gain Ratio,...

Decision Tree 'prefers' smaller tree (simpler theories) over larger ones. This bias, in fact, reflects the greedy search strategy while finding a (heuristic) optimal tree which maximizes entropy.

Because the partitions can be expressed by geometric regions (with support of box constraint), or set of rules (if-then), or in form of a partitioning tree, decision function of Decision Tree algorithms can be expressed as a decision

tree, or equivalent set of rules or as following functional form:

$$\hat{y} = T(\boldsymbol{x}) = \sum_{m=1}^{M} c_m I(\boldsymbol{x} \in \hat{R}_m)$$

where

- $\hat{R}_m$ is hyperrectangles in input space induced by tree cuts
- c_m estimated response within each rectangle. It can be class labels.
- $I(.)$ indicator function.

We can also convert DT decision function to a common form of decision boundary function: $f(x) = 0$. For example, in a simple case: $\boldsymbol{x} \in$ class 1 if $f(\boldsymbol{x}) < 0$ and $\boldsymbol{x} \in$ class 2 if $f(\boldsymbol{x}) > 0$. The box constraints (to describe regions or partitions) can be always converted to continuous function. For example. $-a < x < a$ and $-b < y < b$ is equivalent to $x^2 + y^2 = R^2$.

Therefore model structure of Decision Tree can be:

- sequence of rules
- function $\hat{y} = T(x)$
- regions
- function of decision boundary

and respectively the model parameter of Decision Tree can be:

- rules parameters
- c_m and $\hat{R}_m$
- box constraint for regions
- params of function of decision boundary

With a complete (big) hypothesis space, we mostly suffer from overfitting. The hypothesis which fit perfectly to the training set may be completely wrong with the test set. To overcome this overfitting problem, some pruning techniques have been developed. The pruning process is similar to the process of choosing hyperparameter C of SVM, using cross validation. However it is more complicated then cross validation, because at each validation step, we have to decide how to prune the tree.

One may raise a question: pruning tree should bring a bigger entropy. Our goal is to minimize the entropy, so why do we prune tree? A possible answer for this may be: the entropy we try to maximize is, in fact, only an empirical entropy, which is computed using only the training instances. It is not the real entropy in the population.

There are many selection measures such as Information Gain, Information Gain Ratio, ... [Mingers(1989)- An empirical comparison of selection measure for decision tree induction. Machine Learning.]: provides an experimental analysis of the relative effectiveness of several selection measures over a variety of problems . He reports significant differences in the sizes of the unpruned trees produces by the different selection measures.

In following subsections we describe entropy and information gain/information gain ration, to see why they are chosen as optimization criteria in decision tree.

1.1 Entropy

Entropy is a measure of how organized or disorganized a system is. The entropy of random variable X with values in C is defined by:

$$H(X) = -\sum_{x \in C} p(x) log_2 p(x)$$

.

Given a random variable Y with value in $C = positive, negative$. The entropy of Y is:

$$H(Y) = -p_{positive} log_2 p_{positive} - p_{negative} log_2 p_{negative}$$

Multiclass : c classes: $|C| = c$:

$$H(Y) = \sum_{i=1}^{c} -p_i long_2 p_i$$

where p_i is the probability $p(Y = C_i)$

Entropy is maximized if p is uniform.

$$H(X) \leq log(|S|)$$

with equality iff $p_i = 1/|S|, \forall i = 1..c$

Because we have only limited number of training instances, we compute p_i approximately by the proportion of class i in the training set.

For any multidimensional random variable $X = (X_1, X_2, ..., X_m)$ we have:

$$H(\boldsymbol{X}) \leq \sum_{i=1}^{m} H(X_i)$$

Equality holds if and only if the variables $X_1, X_2, ...X_m$ are mutually independent.

Conditional entropy:

$$H(X|Y) = \sum_{y} p_Y(y) H(X|y)$$

Mutual Information between random variables X and Y is defined as the reduce of entropy $H(X)$ when Y is known (how much knowing one of these variables reduces our uncertainty about the other).

$$I(X, Y) = \sum_{y} p_Y(y)(H(X) - H(X|y)) = H(X) - H(X|Y)$$

$$I(X, Y) = I(Y, X)$$

In context of Decision Tree, Mutual Information is called Information Gain.

Decision Tree algorithm tries to partition in order of mutual information of target variable Y and features (attributes) A

1.2 Information Gain

Information gain of an attribute A, relative to a set of samples S is defined as:

$$IG(S, A) = H(S) - H(S|A) = H(S) - \sum_a p_A(a)H(S|a)$$

where $p_A(a) = P(A = a)$ is approximated by propotion of instances which has value $A == a$ over the total number of instances: $p_A(a) = \frac{|S_a|}{|S|}$

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v)$$

In fact, it is an approximated mutual information between Y- class label and attribute A

$$I(Y, A) = H(Y) - H(Y|A) = H(Y) - \sum_{a \in ValuesA} p_A(a)H(Y|a)$$

If attributes $A_1, A_2, ...A_n$ are independent, we have following equality:

$$H(Y|A_1, A_2, A_n) = \sum_i H(Y|A_i)$$

where $Values(A)$ is the set of all possible values for attribute A, and S_v is the subset of S for which attribute A has value v (i.e: $S_v = s \in S|A(x) = v$). Note the the first term is just the entropy of the original set S, an the second term is the expected value of the entropy after S is partitioned using attribute S. $IG(S, A)$ is therefore the expected reduction in entropy caused by knowing the value of attribute A.

1.3 Information Gain Ratio

Split Information: measure how broadly and uniformly the attribute A splits the data

$$SplitInformation(S, A) = \sum_{i=1}^{c} \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|}$$

Note that $SplitInformation$ is actually the entropy of S with respect to the values of attribute A. The $GainRatio$ measure is defined in terms of the earlier Information Gain measure, as well as this $SplitInformation$, as follows:

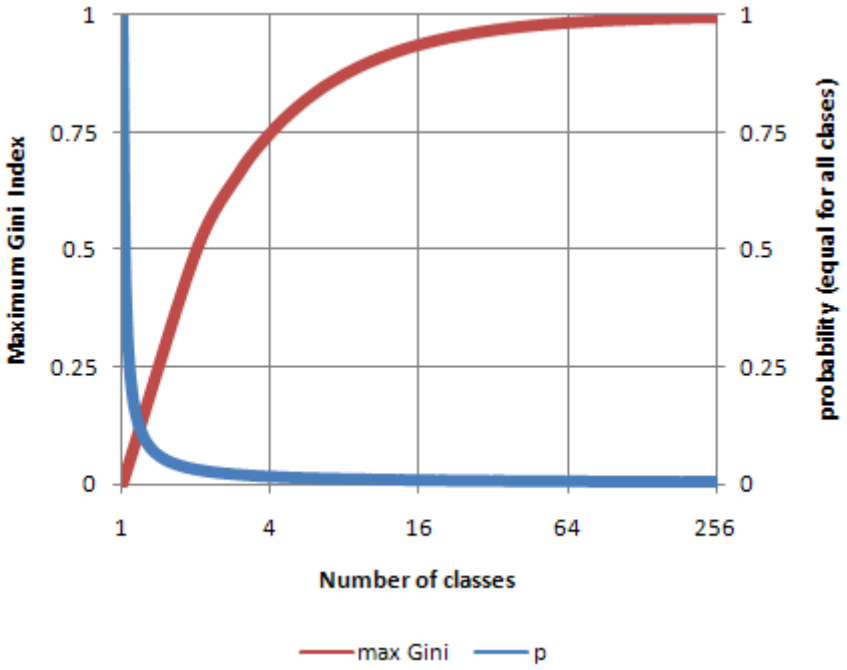
$$GainRatio(S, A) = \frac{IG(S, A)}{SplitInformation(S, A)}$$

Notice that the $SplitInformation$ term discourages the selection of attributes with many uniformly distributed values.

Information Gain Ratio is also called Normalized Information Gain.

1.4 Gini Index

Another way to measure impurity degree is using Gini index: $GiniIndex = 1 - \sum_j^c p_j^2$ Value of Gini index is always between 0 and 1, regardless the number of



classes.

2 Some examples of Decision Tree algorithms

2.1 ID3

- Split criterion: Information Gain or Information Gain Ratio
- Optimization problem: Optimize the Entropy of the training set
- Optimization strategy: greedy discrete search

2.2 C4.5

- Split criterion: Information Gain Ratio
- Optimization problem: Optimize the Entropy of the training set
- Optimization strategy: greedy discrete search
- Handle both continuous and discrete attributes. C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it.
- Handle training data with missing attribute values (missing attribute values are simply not used in gain and entropy calculations).
- Prune tree

2.3 CART

Classification and Regression Tree.

2.4 LTree

2.5 Random Forest

2.6 Other notes

Sequential covering can be considered as a special case of Decision Tree.

2.7 Advantages

Decision Tree is easy to interpret, especially for discrete value. It is proved to be efficient in many applications.

2.8 Disadvantages

1. High variance caused by greedy search strategy (local optima)
2. Use univariate feature selection/criteria → not good if the target depends on multi variables.
3. Linear function is the worst function for trees.

3 Tree pruning