Summary of Decision Tree Algorithms

Yifei Fang,Detao Zhao

December 22, 2019

1 Introduction of descision tree

A decision tree is a hierarchical model for supervised learning, whereby local regions are decided by a few steps of recursive splitting.
A decision tree is a tree structure similar to a flowchart: each node represents a test on an attribute, each branch represents an attribute output, and each leaf node represents a class or class distribution. The top level of the tree is the root node. And there are three algorithms to solve the decision tree model:

1. ID3 Algorithm
2. C4.5 Algorithm
3. CART Algorithm

And in the following part, we will give the details about these three algorithm.

2 ID3 Algorithm

2.1 Introduction to the ID3 algorithm

J. Ross Quinlan and others proposed the ID3 algorithm in 1986. The core is "information entropy". In the process of creating a decision tree, each attribute in the sample set is sequentially consulted, and the attribute with the largest information gain value is selected. This attribute is used as the test attribute and the division criterion. This standard divides the original data set into multiple more pure subsets, and repeats this process in each subset until all the samples in the branch subset cannot continue to be divided, that is, the sample attributes belong to the same category. The tree is created.

2.2 ID3 Algorithm Principle

ID3 algorithm principle includes information entropy and information gain in information theory. Information entropy is used as a measure of the impureness of the attribute category. The higher the entropy value, the lower the purity of the attribute, and the higher the contrary. Information gain is obtained by information entropy subtraction, which reflects the importance of this attribute characteristic in the overall data set.

Information gain:Known sample set T, training attribute B of decision tree. Suppose$\{b_1, b_2, b_3, ..., b_n\}$ is n different values of attribute B. Through these n different valus we could divide set T into n different subset $\{a_1, a_2, a_3, ..., a_n\}$.Then each subset corresponds to a branch generated by a node. The mathematical expectation of attribute B for set T partition is:

$$E(B) = \sum_{j=1}^{n} \frac{a_{1j}, a_{2j}, a_{3j}, ..., a_{nj}}{a} + H(a_{1j}, a_{2j}, a_{3j}, ..., a_{nj})$$

The weight of the corresponding subset can be obtained by dividing a from the sample of attribute B. That is $\frac{a_{1j},a_{2j},a_{3j},...,a_{nj}}{a}$, the final information gain is:

$$Gain(B) = H(a_1, a_2, a_3, ..., a_n) - E(B)$$

Information entropy:It is known that the sample set T, whose number of samples is a, has n different values (attributes), can be divided into n classes $M_i(i = 1, 2, ..., n)$. Assuming that the number of sample M_i is a_i, the information entropy evaluation formula corresponding to the classification of sample M_i is:

$$H(a_1, a_2, a_3, ..., a_n) = -\sum_{i=1}^{n} p_i log(p_i)$$

and p_i represents the probability of sample Mi,and $Mi = \frac{a_i}{a}$,log base reflects the information entropy unit.

2.3 ID3 Algorithm

1. Create node N
2. If S are all in the same class c,return N as a leaf node, recorded as class c
3. Select the attribute A with the largest information gain in D, and mark node N as A
4. For each A unknown value, grow a branch with condition A = value from node N
5. if Bvalue==NULL, then add a leaf node and mark it as the most common class in S,else add a node returned from Create _Tree (Bvalue, D – A)

3 C4.5 Algorithm

3.1 Introduction to the C4.5 algorithm

C4.5 algorithm is an extension and optimization of the ID3 algorithm. The C4.5 algorithm makes several improvements to the ID3 algorithm:

1. Selecting split attributes by information gain rate overcomes the disadvantage of ID3 algorithm that tends to select attributes with multiple attribute values as split attributes through information gain;
2. It can handle discrete and continuous attribute types, that is, discretize continuous attributes;
3. Pruning after constructing the decision tree;
4. It can process training data with missing attribute values.

3.2 C4.5 Algorithm Principle

Selection of splitting attributes-information gain rate
The criterion of split attribute selection is the fundamental difference between decision tree algorithms. Different from the ID3 algorithm, the split attribute is selected by the information gain, and the C4.5 algorithm selects the split attribute by the information gain rate. Split information for attribute A:

$$SplitInfor_A(S) = -\sum_{j=1}^{m} \frac{|S_j|}{|S|} log_2 \frac{|S_j|}{|S|}$$

The training data set S is divided into m sub-data sets by the attribute value of the attribute A, $|S_J|$ represents the number of samples in the j-th sub-dataset, and $|S|$ represents the total number of samples in the data set before the division. Information gain of the sample set after splitting by attribute A:

$$InfoGain(S, A) = E(S) - E_A(S)$$

Information gain rate of the sample set after splitting by attribute A:

$$InfoGainRation(S, A) = \frac{InfoGain(S, A)}{SplitInfo_A(S)}$$

When constructing a decision tree using the C4.5 algorithm, the attribute with the highest information gain rate is the splitting attribute of the current node. With recursive calculation, the information gain rate of the calculated attribute will become smaller and smaller, and in the later period, the relative The attribute of the relatively large information gain rate is used as the splitting attribute.
Discretization of continuous attributes
When the attribute type is discrete, the data does not need to be discretized; When the attribute type is continuous, the data needs to be discretized. The C4.5 algorithm aims at the discretization of continuous attributes. The core idea is to arrange the N attribute values of attribute A in ascending order; Divide all the attribute values of attribute A into two parts by dichotomy (there are N-1 division methods, the threshold value is the median value of two adjacent attribute values); Calculate the information gain corresponding to each

division method, and select the threshold value of the division method with the largest information gain as the threshold of attribute A dichotomy. The detailed process is as follows:

1. All data samples on the node Node are arranged according to the specific value of the continuous attribute A, and the sequence of the attribute value of the attribute A is obtained as$(x_1^A, ..., x_N^A)$;
2. The are N-1 dichotomies in the sequence$(x_1^A, ..., x_N^A)$,hat is, a total of N-1 separation thresholds are generated. For the i-th dichotomy method, the dichotomy threshold $\theta_i = \frac{x_i^A + x_{i+1}^A}{2}$. It divides the dataset on this node into 2 sub-datasets $(x_1^A, ..., x_i^A)(x_{i+1}^A, ..., x_N^A)$ and calculate the information gain under this dichotomy;
3. Calculate the information gain under N-1 binary results, select the binary result with the largest information gain as the division result of attribute A, and record the binary threshold at this time.

Then comes to the prune process.
Pruning-PEP
The C4.5 algorithm uses PEP (Pessimistic Error Pruning) pruning method. The PEP pruning method proposed by Quinlan is a top-down pruning method. It determines whether to perform subtree pruning based on the error rate before and after pruning, so no separate pruning data set is required: For a leaf node, it covers n samples with e errors, then the leaf node has an error rate of$\frac{(e+0.5)}{n}$, if we put the penalty factor as 0.5. For a subtree with L leaf nodes, the false positive rate of the subtree is:

$$ErrorRatio = \frac{\sum_{i=1}^{L} e_i + 0.5L}{\sum_{i=1}^{L} n_i}$$

Where e_i represents the number of samples misclassified by the i-th leaf node of the subtree, and n_i indicates the total number of samples in the i-th leaf node of the subtree. Suppose that a subtree incorrectly classifies a sample with a value of 1 and correctly classifies a sample with a value of 0, then the number of misjudgments of the subtree can be considered as a Bernoulli distribution, so the mean and Standard deviation:

$$ErrorMean = ErrorRation * \sum_{i=1}^{L} n_i$$

$$ErrorSTD = \sqrt{ErrorRatio * \sum_{i=1}^{L} n_i * (1 - ErrorRatio)}$$

After replacing the subtree with a leaf node, the false positive rate of the leaf node is:

$$ErrorRatio' = \frac{e' + 0.5}{n'}$$

Where $e^{'} = \sum_{i=1}^{L} e_i$,$n^{'} = \sum_{i=1}^{L} n_i$ At the same time, the number of misjudgments of the leaf node is also a Bernoulli distribution, so the average value of the number of misjudgments of the leaf node is:

$$ErrorMean^{'} = ErrorRatio^{'} * n^{'}$$

The conditions for pruning are:

$$ErrorMean + ErrorSTD \geq ErrorMean^{'}$$

When the pruning condition is satisfied, the obtained leaf node is replaced with the subtree, which is a pruning operation.
Treatment of missing attribute values
Sample sets with missing attributes typically cause three problems:

1. When constructing a decision tree, the selection of each split attribute is determined by the information yield of all attributes in the training sample set. And at this stage, if some samples in the training sample set lack some attributes, how to calculate the information yield rate of the attributes at this time;

2. When an attribute has been selected as the split attribute, the sample set should be branched according to the value of the attribute, but for those samples whose attribute value is unknown, which subtree should be branched to;

3. After the decision tree has been constructed, if some attribute values are missing in the sample to be classified, how does the classification process of the sample proceed.

And C4.5 algorithm can slove the problems by the following methods:

1. Facing the first problem, when calculating the information yield rate of each attribute, if the attribute value of some samples is unknown, we can deal with it as follows: When calculating the information yield rate of an attribute, ignore the sample missing this attribute; The attribute value that appears most frequently in the samples is assigned to the sample that lacks this attribute;

2. Facing the second problem, suppose that attribute A has been selected as a branch node in the decision tree. When branching the sample set, for those samples whose attribute A's value is unknown, you can send samples for processing: do not process those attributes A unknown , Simply ignore them; or assign values to unknown samples based on the values of other samples of attribute A;

3. Facing problem three, according to the generated decision tree model, when classifying a sample to be classified, if the value of the attribute A of the sample is unknown, it can be processed as follows: The classification

process can be ended, and the category to which this sample belongs is the category with the highest probability in the subtree of attribute A; or attribute A of the sample to be classified is assigned a most common value, and then the classification process is continued.

3.3 C4.5 Algorithm Description

And C4.5 algorithm can slove the problems by the following methods:

```
 1. Input:an attribute-valued dataset $D$
 2. if $D$ is "pure" or other stopping criteria met,then
 3. terminate
 4. end if
 5. for all attribute $a \in D$ do
 6. Compute information-theroetic criteria if we split on $a$
 7. end for
 8. $a_{best}$= Best attribute according to above computed criteria
 9. Tree = Creat a decision node that tests $a_{best}$ in the root
10. $D_v$=Induced sub-datasets from $D$ based on $a_{best}$
11. for all $D_v$ do
12. $Tree_v$=$C4.5(D_v)$
13. Attach $Tree_v$ to the corresponding branch of Tree
14. end for
15. return Tree
```

4 CART Algorithm

4.1 Introduction to CART Algorithm

The CART decision tree algorithm is a decision tree construction algorithm proposed by Breiman in 1984. It uses a binary segmentation method to cut the data into two parts at a time, and enters the left and right subtrees. Each non-leaf node has Two children, the tree thus built is a binary tree. The CART algorithm uses the Gini index to select the attributes to be divided. Each iteration of CART reduces the Gini coefficient. When the data contains more categories, the coefficient is larger. Only when the coefficient is smaller, the data contains different data. The fewer types, the better the features. When all the sample data in a node belong to a class, the Gini coefficient is 0.

4.2 CART Algorithm Principle

In the classification problem, suppose there are K categories, and the probability of the kth category is p_k, then the expression of the Gini coefficient is:

$$Gini(p) = \sum k = \sum_{k=1}^{K} p_k(1-p_k) = 1 - \sum_{k=1}^{K} p_k^2$$

If it is a binary classification problem, the calculation is even simpler. If the probability of output belonging to the first sample is p, then the expression of the Gini coefficient is:

$$Gini(p) = 2p(1-p)$$

For a given sample D, assuming that there are K categories and the number of the kth category is C_k, the expression of the Gini coefficient of sample D is:

$$Gini(D, A) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

and Gini(D) represents uncertainty of set D; Gini(A,D) represents the uncertainty of the set D after $A = a$ segmentation.

Comparing the expressions of the Gini coefficient and the expressions of the entropy model, the second operation is much simpler than the logarithm, especially the calculation of the second-class classification is even simpler. But simple to simple, compared with the measurement method of entropy model, how big is the error corresponding to Gini coefficient? For the two-class classification, the curve of half the Gini coefficient and entropy is as follows:

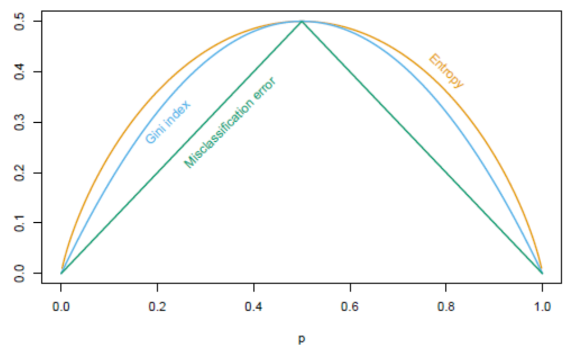

Figure 1: Gini and Entropy Model

As can be seen from the above figure, the curve of the Gini coefficient and half of the entropy is very close, and the error is only slightly larger around the 45-degree angle. Therefore, the Gini coefficient can be used as an approximate replacement for the entropy model. The CART classification tree algorithm uses the Gini coefficient to select the characteristics of the decision tree. At the same time, for further simplification, the CART classification tree algorithm only bisects the value of a certain feature instead of multiple points. In this way, the CART classification tree algorithm builds a binary tree instead of a multi-tree. In this way, the calculation of the Gini coefficient can be further simplified, and in the second, a more elegant binary tree model can be established.

4.3 CART Algorithm Description

The algorithm starts from the root node and uses the training set to recursively build a CART tree:

1. For the current node's data set as D, if the number of samples is less than the threshold or no features, the decision subtree is returned, and the current node stops recursion.

2. Calculate the Gini coefficient of sample set D. If the Gini coefficient is less than the threshold, the decision tree subtree is returned, and the current node stops recursion.

3. Calculate the Gini coefficient of each feature value pair data set D of each feature currently at the current node.

4. From the calculated Gini coefficients of each feature value of each feature to the data set D, the feature A with the smallest Gini coefficient and the corresponding feature value a are selected. According to this optimal feature and optimal eigenvalue, the data set is divided into two parts D1 and D2, and the left and right nodes of the current node are established at the same time. The data set D of the node is D1, and the data set D of the right node is D2.

5. Recursively call the left and right child nodes in steps 1-4 to generate a decision tree.

5 Reference

[1]https://blog.csdn.net/Amber_amber/article/details/47317173
[2]https://blog.csdn.net/fuqiuai/article/details/79456971
[3] https://blog.csdn.net/ch18328071580/article/details/99224997