Team Project Part 1

May 17, 2020

Contents

1 background knowledge

1.1 Decision Tree

Decision tree has advantages:

- Simple Idea: IF...THEN...
- It can deal with high dimension data and winnow importan variables.
- The results are easy to understand.
- Quick calculation
- Ideal correctness

CART decision tree is called Classification and Regression tree. When the dataset is of continuous type, the tree can be a Regression Tree. We can predict the value by the expected value of leaf nodes. When dataset is of discrete type, we can regard it as a Classification Tree. The tree **is a binary tree**. One feature can be used many times. Every non-leaf node can only extend to two children.

1.2 Entropy

Definition: the degree of disorder or randomness in the system.
Suppose X is a discrete random variable, the pmf:

$$P(X = X_i) = p_i, i = 1, 2, \dots, n$$

then the entropy of RV X is:

$$H(X) = -\sum_{i=1}^{n} p_i log_2 p_i$$

The more the entropy is, the unceritainty the RV is.

1.3 Conditional Entropy

In the given condition of X, the conditional entropy of RV Y $H(Y|X)$ is defined as:

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = X_i)$$

In the equation, $p_i = P(X = X_i)$

1.4 Information Gain

Definition: Information gain is the reduction in entropy or surprise by transforming a dataset and is often used in training decision trees. Information gain is calculated by comparing the entropy of the dataset before and after a transformation.
The information gain that the feature A contribuites to dataset D is called

$$g(D, A) = H(D) - H(D|A)$$

For the dataset D, we need to calculate the information regarding to each feature and each feature value, and choose the largest one, which is the best.
Suppose a training dataset D, the capacity is |D|, has k categories C_k, $|C_k|$ is the sample number of C_k. Suppose one feature A has n values $a_1, a_2, \ldots, an$ We can divide D into n subsets $D_1, D_2, \ldots, D_n$, $|D_i|$ is the sample number of D_i. We denote D_{ik} as a subset of D_i which belong to C_k, $|D_{ik}|$ is the sample number of D_{ik}. We then calculate the information gain as follows:

1. calculate
$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} log_2 \frac{|C_k|}{|D|}$$
.
2. calculate the conditional entropy of feature A contributing to D
$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|Di|} log_2 \frac{|D_{ik}|}{|D_i|}$$
3. calculate information gain
$$g(D, A) = H(D) - H(D|A)$$

1.5 Information Gain Ratio

Sometimes we may choose improperly a feature that has too much values. Such situation makes no sense. We must correct it using information gain ratio.

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}$$

$$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} log_2 \frac{|D_i|}{|D|}$$

1.6 Gini Index

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

In the equation, p_i is the probability of class C_i in D
For a discrete variable, we need to calculate the weight sum of each zone's impurity, As the following:

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

For a continuous variable, we can set a dividing point to get the same goal.
Our goal is to make the weight sum as small as possible by choose the best feature and the best feature value.

2 Get Access to Data

Use open() function to extract data from csv files.

```
def get_data(file_name):
    f = open(file_name, 'r').readlines()
    data_name = f[0].split(',')[:-1]
    data = []
    for i in f[1:]:
        d = [float(j) for j in i.split(',')]
        d[-1] = d[-1] > 6 # Transform the quality figures into True or False
        data.append(d)
    return data_name, data
```

3 Build a Decition Tree

- Starting from root node, calculate the possible **information gain/information gain ratio/gini index** regarding each feature and value. Choose the best information gain/ratio/gini index. Construct different child nodes according to the feature and value.
- Use recursion to the child node and build the tree.
- Until all the labels are the same after selection.

3.1 Feature Choice

The most popular methods are:

- ID3: Depend on information gain
- CD4.5: Depend on information gain ratio
- CART: Depend on Gini Index when it is Classification Tree, on MSE when it is Regression Tree.

Here we use CART to construct the classification tree.

4 Prune the Tree

We can do pruning to optimize the tree size and reduce overfitting. Generally, there are two kinds of pruning: Pre-pruning and Post-pruning.

4.1 Pre-pruning

- Pre-pruning is also called early stoping.
- Pre-pruning stops the tree before it grows perfectly.
- People can set maximum depth of the tree to restrict the tree.

4.2 Post-pruning

4.2.1 REP(Reduced-Error Pruning)

REP is one of the simplest forms of Post-pruning.
First, calculate the error of a subtree $E_r(t)$. Then, calculate the error of each leaf node of this subtree $E_t(Tt)$. If

$$E_r(t) < \sum E_r(T_t)$$

Then, replace the subtree with a leaf node, whose label is determined by the majority label of the subtree. Repeat this process from the bottom of the tree to the top.

4.2.2 PEP(Pessimistic-Error Pruning)

For a leaf node with N samples and E errors, its error rate e is $\frac{E+0.5}{N}$. "0.5" is called penalty factor. For a subtree with L leaf nodes, its error rate is

$$p = \frac{\sum_{i=1}^{L} E_i + 0.5L}{\sum_{i=1}^{L} N_i}$$

Suppose that all the samples in a subtree is of binomial distribution $B(N, P)$, then the expectation and standard deviation of error before pruning are:

$$E_T = N * p = N * \frac{\sum_{i=1}^{L} E_i + 0.5L}{\sum_{i=1}^{L} N_i} = \sum_{i=1}^{L} E_i + 0.5L$$

$$\sigma = \sqrt{N * p * (1 - p)}$$

Expectation of error after pruning is:

$$E_t = N * e = N * \frac{E + 0.5}{N} = E + 0.5$$

If

$$E_t - E_T < \sigma$$

then prune the subtree. Repeat this process from the top of the tree to the bottom.

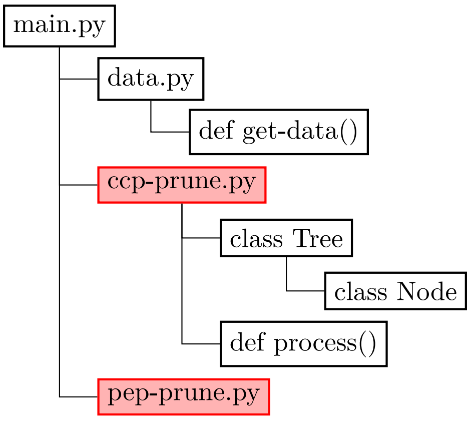
4.2.3 CCP(Cost-Complexity Pruning)

α is a real number called the complexity parameter, it's defined as follows:

$$\alpha = \frac{R(t) - R(T_t)}{|T_t| - 1},$$

where $R(t)$ is the error after pruning, $R(T_t)$ is the error of the original subtree and $|T_t|$ is the number of samples in the subtree.
Step1: Calculate the value of α for all the subtrees from the bottom to the top, each time prune the subtree with minimal α. Get a set $\{T_0, T_1, ..., T_M\}$, where T_0 is a complete tree and T_M is a root node.
Step2: Pick the best tree from $\{T_0, T_1, ..., T_M\}$, according to its performance on the testing sets.
*Note that if we use CCP to prune the tree, we should separate a testing set from the given training set before training begins.

5 Code structure



6 Result

We can attain 89.7916666667 percent accuracy @fptop 5pt

6.1 The Confusion Matrix

	True Bad Wine	True Good Wine
Predicted Bad Wine	395	28
Predicted Good Wine	21	36