北京航空航天大學
BEIHANG UNIVERSITY

Decision Tree Learning for Classification

Experiment Report

School	School of Automatic Science
Major	Pattern Recognition and Intelligent System
Student	Tianxiang Lan
Student ID	15231087
Teacher	Zengchang Qin

May 7,2018

1 Introduction

Decision tree induction is one of the simplest and yet most successful learning algorithms. A decision tree (DT) consists of internal and external nodes and the interconnections between nodes are called branches of the tree. An internal node is a decision-making unit to decide which child nodes to visit next depending on different possible values of associated variables. In contrast, an external node also known as a leaf node, is the terminated node of a branch. It has no child nodes and is associated with a class label that describes the given data. A decision tree is a set of rules in a tree structure, each branch of which can be interpreted as a decision rule associated with nodes visited along this branch.

2 Principle and Theory

Decision trees classify instances by sorting them down the tree from root to leaf nodes.This tree-structured classifier partitions the input space of the data set recursively into mutually exclusive spaces. Following this structure, each training data is identified as belonging to a certain subspace, which is assigned a label, a value, or an action to characterize its data points. The decision tree mechanism has good transparency in that we can follow a tree structure easily in order to explain how a decision is made.Thus interpretability is enhanced when we clarify the conditional rules characterizing the tree.

Entropy of a random variable is the average amount of information generated by observing its value. Consider the random experiment of tossing a coin with probability of heads equal to 0.9, so that P(Head) = 0.9 and P(Tail) = 0.1. This provides more information than the case where P(Head) = 0.5 and P(Tail) = 0.5.

Entropy is used to evaluate randomness in physics, where a large entropy value indicates that the process is very random. The decision tree is guided heuristically according to the information content of each attribute. Entropy is used to evaluate the information of each attribute; as a means of classification. Suppose we have *m* classes, for a particular attribute, we denoted it by pi by the proportion of data which belongs to class C_i where *i = 1, 2, … m.*

The entropy of this attribute is then:

$$Entropy = \sum_{i=1}^{m} -p_i \cdot \log_2 p_i$$

We can also say that entropy is a measurement of the impurity in a collection of training examples: larger the entropy, the more impure the data is. Based on entropy, Information Gain (IG) is used to measure the effectiveness of an attribute as a means of discriminating between classes.

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

where all examples *S* is divided into several groups (i.e. S_v for $v \in Values(A)$) according to the value of *A*. It is simply the expected reduction of entropy caused by partitioning the examples according to this attribute.

3 Objective

The goals of the experiment are as follows:

(1) To understand why we use entropy-based measure for constructing a decision tree.

(2)To understand how Information Gain is used to select attributes in the process of building a decision tree.

(3) To understand the equivalence of a decision tree to a set of rules.

(4) To understand why we need to prune the tree sometimes and how can we prune? Based on what measure we prune a decision tree.

(5) To understand the concept of Soft Decision Trees and why they are important extensions to classical decision trees.

4 Contents and Procedure

Stage 1:

(1) According to the above principle and theory in section 3.2, implement the code to calculate the information entropy of each attribute.

(2) Select the most informative attribute from a given classification problem (e.g., we will be given the Iris Dataset from the UCI Machine Learning Repository)

(3) Find an appropriate data structure to represent a decision tree. Building the tree from the root to leaves based on the principal discussed in section 3.2 by using Information Gain guided heuritics.

Stage 2:

(1) Now consider the case of with continuous attributes or mixed attributes (with both continuous and discrete attributes), how can we deal with the decision trees? Can you propose some approaches to do discretization?

(2) Is there a trade off between the size of the tree and the model accuracy?

Is there existing an optimal tree in both compactness and performance?

(3)For one data element, the classical decision tree gives a hard boundary to decide which branch to follow, can you propose a "soft approach" to increase the robustness of the decision tree?

(4) Compare to the Naïve Bayes, what are the advantages and disadvantages of the decision tree learning?

Stage 3:

Explore the questions in the previous section and design experiments to answer these questions. Complete and submit an experiment report about all experiment results with comparative analysis and a summary of experiences about this experiment study.

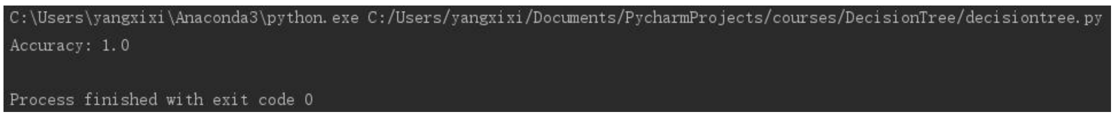
5 Results and Analyzation

Stage 1:

I choose the Iris Dataset from the UCI Machine Learning Repository to do experiment.After code written,I finished all the content of stage1.I think binary tree is an appropriate data structure.

The results are following:

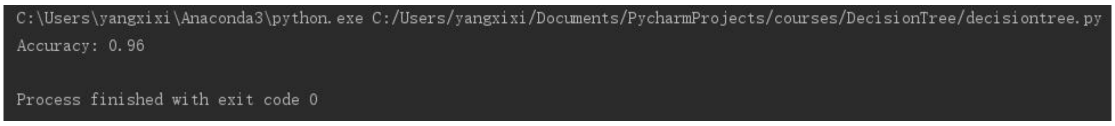
The accuracy of the Decision Tree is 1.0 because of over-fitting.



```
C:\Users\yangxixi\Anaconda3\python.exe C:/Users/yangxixi/Documents/PycharmProjects/courses/DecisionTree/decisiontree.py
Accuracy: 1.0

Process finished with exit code 0
```

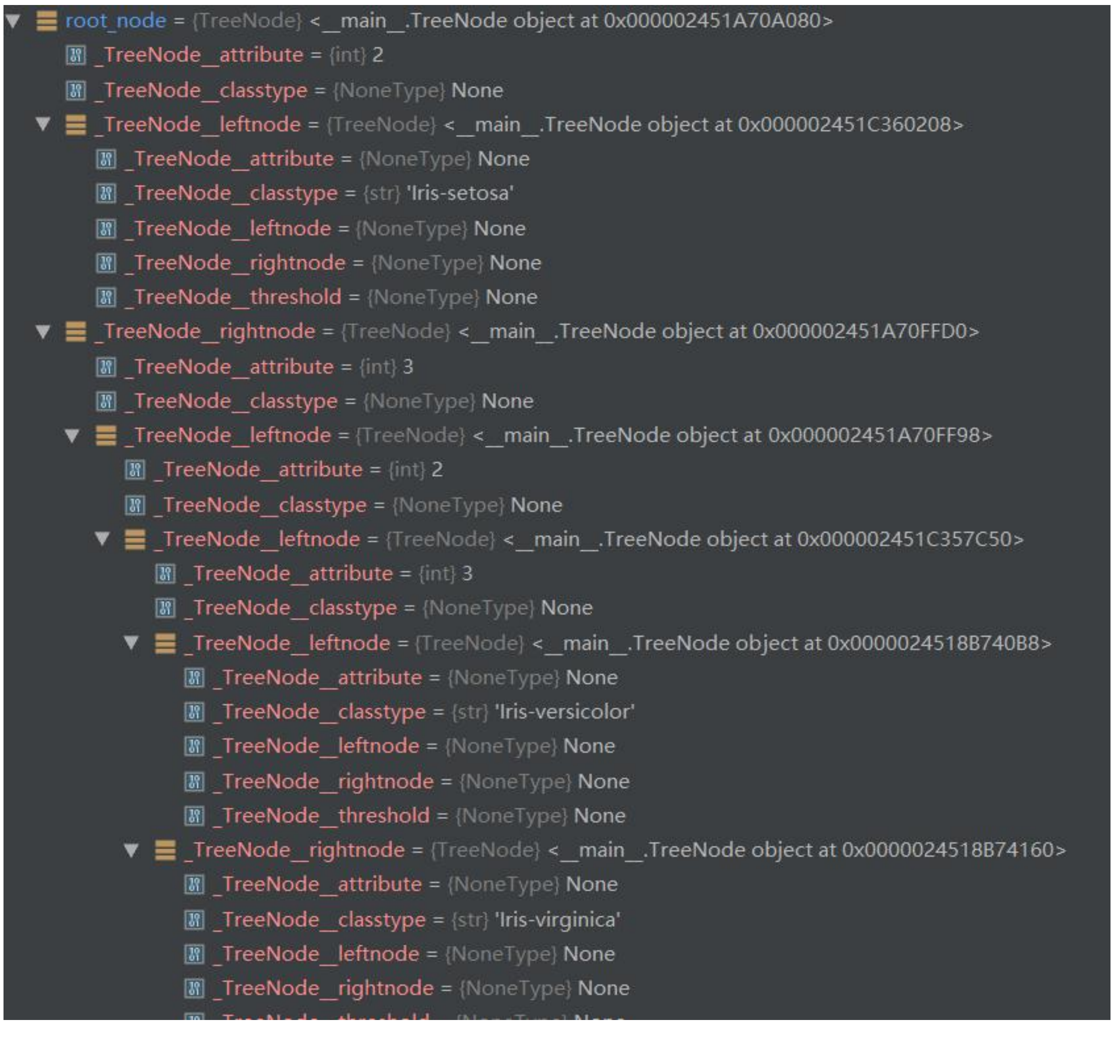
Now I consider weakening it.In my code,I assume a perameter $\alpha \in [0,1]$ .If the largest sample number of every class is greater than $\alpha \times sumof$number ,the node is set as leaf node, rather than equaling $sumof$number .For example,when I set $\alpha = 0.9$,then the accuracy is 0.96.As you can see,the over-fitting is weakened.



```
C:\Users\yangxixi\Anaconda3\python.exe C:/Users/yangxixi/Documents/PycharmProjects/courses/DecisionTree/decisiontree.py
Accuracy: 0.96

Process finished with exit code 0
```

You also can see a part of learned tree structure here.If you want to see the whole tree,please visit my github ,download my code and run them.You can check it when debugging.My github url is placed at the end of this experiment report.



```
▼ root_node = {TreeNode} <__main__.TreeNode object at 0x000002451A70A080>
    _TreeNode__attribute = {int} 2
    _TreeNode__classtype = {NoneType} None
  ▼ _TreeNode__leftnode = {TreeNode} <__main__.TreeNode object at 0x000002451C360208>
      _TreeNode__attribute = {NoneType} None
      _TreeNode__classtype = {str} 'Iris-setosa'
      _TreeNode__leftnode = {NoneType} None
      _TreeNode__rightnode = {NoneType} None
      _TreeNode__threshold = {NoneType} None
  ▼ _TreeNode__rightnode = {TreeNode} <__main__.TreeNode object at 0x000002451A70FFD0>
      _TreeNode__attribute = {int} 3
      _TreeNode__classtype = {NoneType} None
    ▼ _TreeNode__leftnode = {TreeNode} <__main__.TreeNode object at 0x000002451A70FF98>
        _TreeNode__attribute = {int} 2
        _TreeNode__classtype = {NoneType} None
      ▼ _TreeNode__leftnode = {TreeNode} <__main__.TreeNode object at 0x000002451C357C50>
          _TreeNode__attribute = {int} 3
          _TreeNode__classtype = {NoneType} None
        ▼ _TreeNode__leftnode = {TreeNode} <__main__.TreeNode object at 0x0000024518B740B8>
            _TreeNode__attribute = {NoneType} None
            _TreeNode__classtype = {str} 'Iris-versicolor'
            _TreeNode__leftnode = {NoneType} None
            _TreeNode__rightnode = {NoneType} None
            _TreeNode__threshold = {NoneType} None
        ▼ _TreeNode__rightnode = {TreeNode} <__main__.TreeNode object at 0x0000024518B74160>
            _TreeNode__attribute = {NoneType} None
            _TreeNode__classtype = {str} 'Iris-virginica'
            _TreeNode__leftnode = {NoneType} None
            _TreeNode__rightnode = {NoneType} None
```

Stage 2:

(1) Now consider the case of with continuous attributes or mixed attributes, how can we deal with the decision trees?

We can use the method named bi-partition to do discretization .Assume there are n number different value of attribute.Sort them from small to large,forming a set like $\{a^1,a^2,\ldots,a^n\}$ .We find a threshold T^i to split them using formula $T^i = \frac{a^i + a^{i+1}}{2}$ .Then T^i forms a set like $\{T^1,T^2,\ldots,T^{n-1}\}$ .We consider every T^i to find the largest Information gain.And you can see my code to find how I deal with it.

(2)Is there a trade off between the size of the tree and the model accuracy?

Yes.One method is to pruning the Decision Tree.We can pruning the Decision Tree,if the performance decline not too much after pruning.Then we can get a balance of both compactness and performance.The other is that,if the largest sample number of every class is greater than $\alpha \times sumof$number ,then the node is set as leaf node, rather than equal $sumof$number ,which I mentioned at stage1.

(3)Propose a "soft approach" to increase the robustness of the decision tree?

If distance between a sample and boundary is very short,it means that the probability that the sample is classified wrongly is higher.So we need to regard it as "unknown".We should consider it in other dimension or attribute.Therefore, we need to change bi-split to tri-split.That is to say we should divide data into three parts that are left,middle and right rather than two parts that are left and right.

(4)Compare to the Naïve Bayes, what are the advantages and disadvantages of the decision tree learning?

Compare to the Naïve Bayes,there are many advantages in using the decision tree model in the classification problem.For examples,the decision tree is easy to use and efficient.The rules can be easily constructed according to the decision tree, and the rules are usually easy to explain and understand.The decision tree can be well extended to the large database, while its size is independent of the size of the database.The decision tree is very small. Another advantage of the model is that it can construct decision trees for datasets with many attributes. There are some shortcomings in the decision tree model, such as the difficulty of dealing with missing data, the emergence of overfitting problems, and the neglect of the correlation between the attributes of the data set.

All the code and instruction files are on my github.You can run them if you want:
https://github.com/Deep-Lan/Decision-Tree.git