ITE4005 Data Science course

Assignment 2

2016026080 한다진

* **Experiment environments**

OS Mac OS

Language python3

Tools Jupyter notebook

Pure python script

* **Assignment\_1**

**I rearranged the folder system. There is no modification of the content and only the file system has been changed. Please refer to it. Thank you.**

* **Decision Tree**

Most important part is Partitioning part.

Partitioning operates in a similar order with prefix order in binary tree. Each node try to find a branch condition with attribute, and midpoint making smallest impurity.

Make DecisionTree object. This would be a root node of entire DecisionTree.

while not (every terminal nodes cannot make a branch anymore)

for each attribute in dataset of current node

sort dataset by current attribute.

Find the value of current attribute that divide current dataset into two dataset with smallest impurity and save it to temporary\_branch.

Compare all temporary branch of each attributes. And find the best condition with smallest impurity.

Branch current node. Do the same for two child nodes..

**Quick Start**

* + Install Libraries

pip install pandas / conda install pandas

* + Option #1 – save result to result.txt

스크린샷이(가) 표시된 사진

자동 생성된 설명

Python dt.py *filename\_train filename\_test filename\_result*

* + Option #2 – save result to result.txt and get scores

스크린샷이(가) 표시된 사진

자동 생성된 설명

Python dt.py *filename\_train filename\_test filename\_result filename\_answer*

* **Code explanation**
  + Import Libraries

Import *pandas* libraries to use DataFrame, and *math* to use log function

* + 텍스트이(가) 표시된 사진

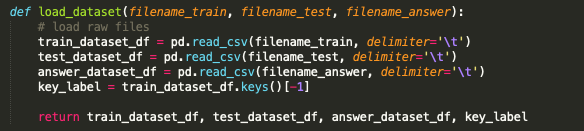
    자동 생성된 설명텍스트이(가) 표시된 사진

    자동 생성된 설명**스크린샷이(가) 표시된 사진

    자동 생성된 설명Decision Tree Class**

Make DecisionTree class and use the internal function in decision class. The structure of DecisionTree class is recursive. The internal function and variables work with recursive pattern, like subtree. Detail of each function is below.

* + Load input file



Load train.txt and test.txt. set the last value of DataFrame.keys() to our target attribute.

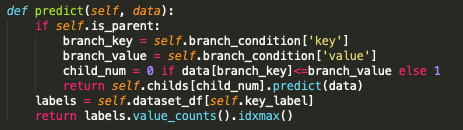
* + Save output and get score

스크린샷이(가) 표시된 사진

자동 생성된 설명

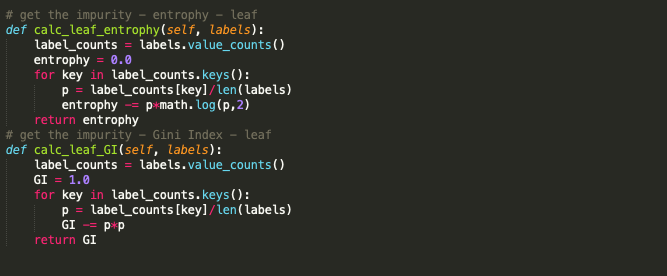
Get the prediction of my DecisionTree class. And compare it with answer.txt when filename\_answer is entered in commands.

* + Predict



When current node is terminal node, check the value of target attribute with the most number. Else, check the branch condition of current node and predict the answer in child’s node recursively.

* + **Calculate the impurity of leaf node**



Calculate the impurity of leaf node. There are two measure methods, entrophy and Gini-Index. Each function calculate the impurity of node. Input parameter labels is the target attributes of selective dataset. I made a Decision tree with recursive structure. So every node has different selective dataset. Data of terminal nodes are not duplicated.

Key of label\_counts is the value of target attribute and the value of label\_counts is the counts of target\_attribute. I manage the impurity just in current subtree. So probability P is calculated with len(current\_dataset) not len(total\_dataset).

* + **Calculate the impurity of leaf node on the assumption that a child node is created**



This function is used to compare the impurity between ‘leaf node version’ and ‘parent node version on the assumption that a child node is created with new attribute branch condition’. So when *calc\_info\_gain* is smaller than *calc\_leaf,* then the impurity of current node and structure is changed.

Calculate impurity of current node using impurity of childs. The Formula is announced above.

* + **Partitioning**

스크린샷이(가) 표시된 사진

자동 생성된 설명

Partitioning the Decision tree unitl it would be a full tree. With zero impurity. Current node check the branch condition that make a smallest impurity. Consider every key and every midpoint. When find the best branch condition, make two child nodes. Two child nodes are root node of each subtree. But a terminal node in current node. Make two child nodes, and repeat this work in both two child nodes until it is impossible to make a new child node.

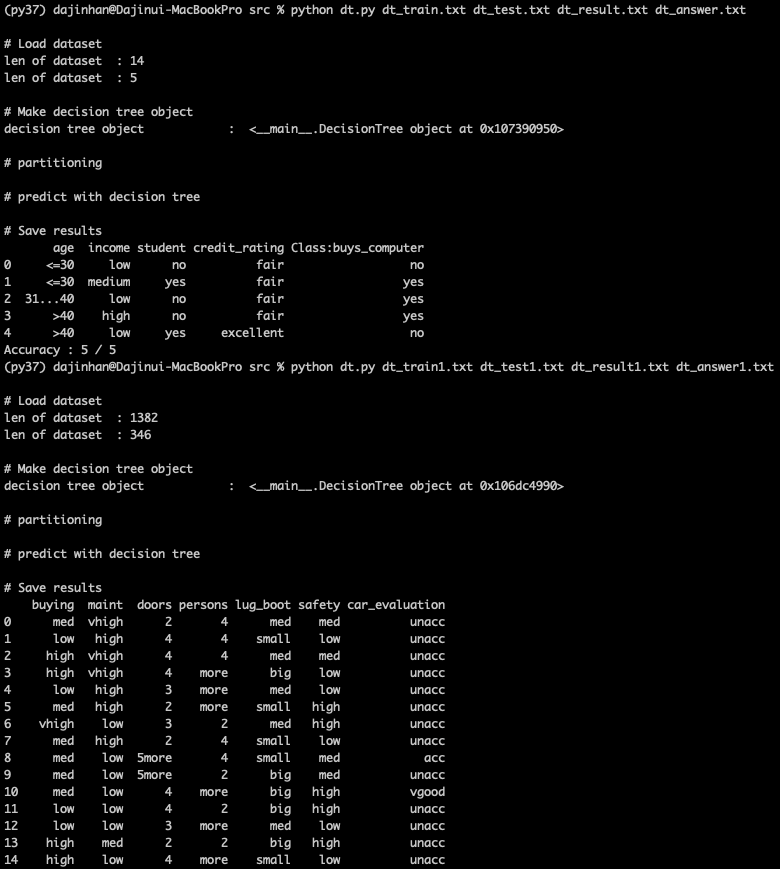
* + **Pruning**

스크린샷이(가) 표시된 사진

자동 생성된 설명

Pruning the decision tree to make it compatible with general case, not only on training\_samples. Make the Decision tree with smallest cost function. Current condition of if phrase is for entrophy version. The annotated part is for Gini-Index version. Pruning is works from the terminal nodes to root node. if cost function of subtree is decrease when convert current node to terminal node, then erase sub branches from current node.

**Experiment**



측정기이(가) 표시된 사진

자동 생성된 설명

This is the result screen when executing the commands “python dt.py dt\_train.txt dt\_test.txt dt\_result.txt dt\_answer.txt” and “python dt.py dt\_train1.txt dt\_test1.txt dt\_result1.txt dt\_answer1.txt” each.

Impurity : entrophy

Pruning : don’t use it.

Accuracy : 341 / 346

1. When I use entrophy as a impurity, and don’t pruning the Decision Tree, accuracy is highest.
2. I assumed that if I use Gini\_Index instead of entrophy accuracy would become higher, but it was not. Accuracy decrease to 339 / 346. Actually there is no significant difference.
3. When I pruned the decision tree, accuracy decreased to 317.