

# < Programming Assignment #2 >

**Submission: LMS** (learning.hanyang.ac.kr)

- **Until May 8 (Thursday), 23:59 PM.**

## 1. Environment

- OS: Windows, Mac OS, or Linux
- Language: Python (any version is fine, but python3 is recommended)

**2. Goal:** Build a **decision tree**, and then classify the test dataset using the tree.

## 3. Requirements

The program must meet the following requirements:

- File name: studentID.py **(not ipynb!)**
  - Only put your student ID. Do NOT put any other information.
  - Example 1: **2008011666.py**
- I will execute your code with **three** arguments: **training dataset file name, test dataset file name, classification result file name**
  - Example: `python 2008011666.py dt_train.txt dt_test.txt dt_result.txt`
    - This means that Training file name='dt\_train.txt', test file name='dt\_test.txt', classification result file name='dt\_result.txt'
    - I will test your assignment using several different train/test datasets.
- Dataset
  - I will provide you with 2 datasets on LMS
    - Buy\_computer: dt\_train.txt, dt\_test.txt
    - Car\_evaluation: dt\_train1.txt, dt\_test1.txt
  - But please be aware that you need to make your program that can deal with **any** datasets
  - We will evaluate your program with other datasets. (format will be the same)
- File format for a training set

```
[attribute_name_1]\t[attribute_name_2]\t ... [attribute_name_n]\n
[attribute_1]\t[attribute_2]\t ... [attribute_n]\n
[attribute_1]\t[attribute_2]\t ... [attribute_n]\n
[attribute_1]\t[attribute_2]\t ... [attribute_n]\n
```

  - [attribute\_name\_1] ~ [attribute\_name\_n]: *n* attribute names
  - [attribute\_1] ~ [attribute\_n-1]
    - *n-1* attribute values of the corresponding tuple
    - All the attributes are **categorical** (not continuous-valued)
  - [attribute\_n]: a **class label** that the corresponding tuple belongs to
  - Example 1 (data\_train.txt):

age	income	student	credit_rating	Class:buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes

Figure 1. An example of the first training set.

■ Example 2 (data\_train1.txt):

buying	maint	doors	persons	lug_boot	safety	car_evaluation
high	high	3	4	big	low	unacc
med	high	2	2	small	med	unacc
low	med	5more	2	big	high	unacc
low	high	2	4	med	low	unacc
med	vhigh	4	2	med	med	unacc

Figure 2. An example of the second training set.

- Data name: car evaluation database
- Attribute values
  - Buying: vhigh, high, med, low
  - Maint: vhigh, high, med, low
  - Doors: 2, 3, 4, 5more
  - Persons: 2, 4, more
  - Lug\_boot: small, med, big
  - Safety: low, med, high
- **Class labels:** unacc, acc, good, vgood
- Number of instances: training set - 1,382; test set - 346

● Attribute selection measure: [gain ratio](#)

● File format for a test set

[attribute\_name\_1]\t[attribute\_name\_2]\t... [attribute\_name\_n-1]\n

[attribute\_1]\t[attribute\_2]\t... [attribute\_n-1]\n

[attribute\_1]\t[attribute\_2]\t... [attribute\_n-1]\n

[attribute\_1]\t[attribute\_2]\t... [attribute\_n-1]\n

■ Example 1 (dt\_test.txt):

age	income	student	credit_rating
<=30	low	no	fair
<=30	medium	yes	fair
31...40	low	no	fair

Figure 3. An example of the first test set.

■ Example 2 (dt\_test1.txt):

buying	maint	doors	persons	lug_boot	safety
med	vhigh	2	4	med	med
low	high	4	4	small	low
high	vhigh	4	4	med	med
high	vhigh	4	more	big	low
low	high	3	more	med	low

Figure 4. An example of the second test set.

- **The test set does NOT** have `[attribute_name_n]` (class label). You need to predict the class labels with the trained decision tree, and record the classification results on the result file (see below).

- Classification result (output) file format

`[attribute_name_1]\t[attribute_name_2]\t ... [attribute_name_n]\n`

`[attribute_1]\t[attribute_2]\t ... [attribute_n]\n`

`[attribute_1]\t[attribute_2]\t ... [attribute_n]\n`

`[attribute_1]\t[attribute_2]\t ... [attribute_n]\n`

- Output file name: **dt\_result.txt (for 1th dataset), or, dt\_result1.txt (for 2<sup>nd</sup> dataset)**

- You must print the following values:

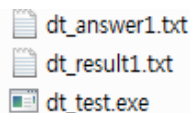
- `[attribute_1] ~ [attribute_n-1]`: given attribute values in the test set
- `[attribute_n]`: a class label predicted by your model for the corresponding tuple

- Please **DO NOT CHANGE** the **order of the tuples** in each test set when you print your outputs

- Please be sure to use `\t` to identify your attributes.

#### 4. Testing program

- Please put the following files in a same directory: Testing program, your output files (dt\_result.txt, dt\_result1.txt), an attached answer file (dt\_answer.txt, dt\_answer1.txt)



- Execute the testing program with two arguments (answer file name and your output file name)

```
DM_assignment2>dt_test.exe dt_answer1.txt dt_result1.txt
```

- Check your score for the input file

For example, **312 / 346**

- the number of your correct prediction / the number of correct answers
- Your tree may not be perfect. It is almost impossible to perfectly classify all the data in the test set. So, a slight number of misclassification is natural.
- The test program was build with program 'mono'. So, even if you are using mac or linux instead of window, you can run dt\_test.exe using C# mono.

#### 5. Important Notes:

- Your score will be measured based on your tree's classification accuracy.
- You don't need to apply the "pruning" strategy.

#### 6. Submission

- Please make a **single, runnable .py file** and upload it on LMS.

#### 7. Penalty

- **0 score:** Late submission, Library violation
  - Some libraries include decision tree implementations, but these libraries are **strictly prohibited!** Only basic libraries like numpy, pandas, should be used, and **the libraries which are directly related to and have implementations of the assignments must NOT be used.**
- **F grade:** Plagiarism