Report

2016025196

김동규

Test 환경

Windows10

python3.9.2

Numpy 1.22.3

구조

main

-open_file : train data를 DB로 생성합니다.

-open_test : classify에 사용될 test data를 가져옵니다.

-output_write : classify가 완료된 test data를 output path에 저장합니다.

-construct_tree

-majority : majority voting으로 major를 생성합니다. 두개 이상일 경우 false를 리턴합니다.

-gain : information gain 기법을 사용하여 gain값을 리턴합니다.

-classify : 노드를 따라 내려가며 leaf node의 value를 리턴합니다.

Class node

-insert_child : 선택된 attribute의 값의 종류에 따라 딕셔너리 형태로 child를 저장합니다.

-select_attr : classify에 사용될 attribute를 저장합니다.

-leaf : 해당 노드를 leaf node로 만들고 value를 입력합니다.

-distribution : 노드에 사용된 DB의 distribution을 저장합니다.

알고리즘

dt-py 코드는 decision tree 알고리즘 중 information gain을 사용하는 모델입니다.

open_file함수를 이용하여 DB를 생성한 뒤, construct_tree 함수를 이용하여 tree형태로 저장되는 자료를 리턴합니다.

Tree 생성 알고리즘은 다음과 같습니다.

1. 현재 노드의 초기 gain값을 구합니다.
2. 분할에 사용되지 않은 attribute들을 순회하며 값에 따라 DB를 분할한 뒤, gain값을 구합니다. 이때, 분할 이후 0개의 길이를 갖는 partition이 생기면, 해당 노드는 leaf로 간주한 뒤, majority voting에 따라 value를 갖습니다.
3. 가장 큰 information gain을 갖게 하는 attribute를 선정하기 위해 분할된 gain값이 가장 작은 attribute를 선정하여 remain_attribute에서 제거합니다.
4. 초기 gain값과 분할된 gain값이 같을 경우, 더 이상 분할되지 않는 노드로 간주, majority voting으로 값을 정한 뒤 leaf node로 만듭니다.
5. 만약 4번에서 분할 가능한 노드일 경우, 선택된 attribute의 값 범위에서 순환하며 child node를 생성합니다. 이때, child중 하나라도 정상적인 값을 리턴하지 못하는 경우(majority가 2가지 이상인 경우) parent node도 majority voting을 통해 값을 정하고 leaf노드로 만듭니다. 이 과정은 major가 단 하나이거나, 모든 child가 정상 생성되는 parent node까지 연쇄적으로 진행됩니다.
6. Child node의 생성이 정상적으로 이뤄진 경우, classify에 필요한 node의 attribute의 값을 정해주고 return합니다.

Decision tree를 생성하며, 해당 노드나 child node에서 majority voting을 통해 2개 이상의 major가 생성되는 경우가 있습니다. 이 과제에서는 해당 경우에 대해 단 하나의 major가 정해지지 않는 brunch들은 모두 pruning하는 방법을 택했습니다.

이후 classfier에서는 root노드와 분류할 데이터가 주어집니다. 데이터는 노드에서 분할에 필요한 attribute를 보고 child node로 내려가며 leaf node일 경우 중단하고 value를 출력합니다.

실행 방법

dt.py [train data file] [test data file] [output file path] 의 명령어를 통해 실행합니다. 실행 시, result string을 출력합니다.

```
C:₩Users₩mg352₩OneDrive₩바탕 화면₩4-1₩데사₩assignment2>"dt.py" dt_train.txt dt_test.txt output.txt
train_data open
test_data open

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

```
C:₩Users₩mg352₩OneDrive₩바탕 화면₩4-1₩데사₩assignment2>"dt.py" dt_train1.txt dt_test1.txt output.txt
train_data open
test_data open

result write
buying  maint   doors   persons lug_boot        safety  car_evaluation
med     vhigh   2       4       med     med     acc
low     high    4       4       small   low     unacc
high    vhigh   4       4       med     med     acc
high    vhigh   4       more    big     low     unacc
low     high    3       more    med     low     unacc
med     high    2       more    small   high    acc
vhigh   low     3       2       med     high    unacc
med     high    2       4       small   low     unacc
med     low     5more   4       small   med     good
med     low     5more   2       big     med     unacc
med     low     4       more    big     high    vgood
low     low     4       2       big     high    unacc
low     low     3       more    med     low     unacc
high    med     2       2       big     high    unacc
high    low     4       more    small   low     unacc
med     vhigh   3       4       med     med     acc
```

실험 결과

실험 결과 dt_test에 대해서는 100% dt_test1에 대해서는 약 90%의 정확도를 보입니다.

```
C:₩Users₩mg352₩OneDrive₩바탕 화면₩4-1₩데사₩assignment2>dt_test.exe output.txt dt_answer1.txt
318 / 346
```