데이터과학을 위한 R프로그래밍

11주차. 의사결정나무와 랜덤포레스트

이혜선 교수
포항공과대학교 산업경영공학과

목차

11주차. 의사결정나무와 랜덤포레스트

11주차

1차시

의사결정나무 I

POSTECH

의사결정나무(Decision Tree)

의사결정나무(Decision Tree)

- 기계학습 중 하나, 의사결정 규칙을 나무 형태로 분류해 나가는 분석 기법

- 분석 과정이 직관적이고 이해하기 쉬움
- 연속형/범주형 변수 모두 사용할 수 있음
- 분지 규칙은 불순도를 최소화시킴

불순도 → 범주들이 섞여있는 정도

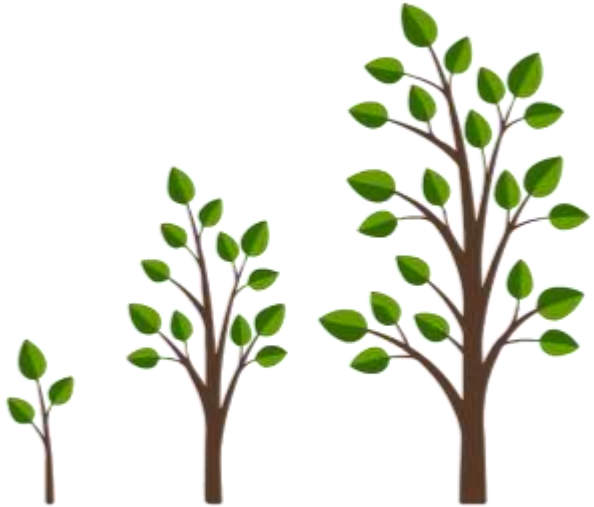
POSTECH

의사결정나무(Decision Tree)

Step1 tree 형성(Growing tree)

Step1 tree 가지치기(Pruning tree)

Step1 최적 tree로 분류(Classification)



POSTECH

의사결정나무(Decision Tree)

- 의사결정나무 실행 패키지 : tree(그 외 rpart, party 패키지)

```
# lec11_1_tree.R
# Decision tree
# use package "tree"

#decision tree packages download
install.packages("tree")
library(tree)

#package for confusion matrix
#install.packages("caret")
library(caret)
```

tree : 의사결정나무 수행을 위한 패키지

caret : confusionMatrix 사용을 위한 패키지

POSTECH

의사결정나무(Decision Tree)

iris 데이터(iris.csv)

input변수(독립변수)

output변수(종속변수, 타겟변수)

A	B	C	D	E
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa

타겟변수(y) : setosa, versicolor, virginica

Iris setosa Iris versicolor Iris virginica

POSTECH

의사결정나무(Decision Tree)

Iris 데이터(학습데이터와 검증데이터의 분할)

```
# set working directory
setwd("D:/tempstore/moocr")

# read csv file
iris<-read.csv("iris.csv", stringsAsFactors = TRUE)
attach(iris)

# training (n=100)/ test data(n=50)
set.seed(1000)
N<-nrow(iris)
tr.idx<-sample(1:N, size=N*2/3, replace=FALSE)
# split train data and test data
train<-iris[tr.idx,]
test<-iris[-tr.idx,]
#dim(train)
#dim(test)
```

데이터분할
(학습데이터 2/3, 검증데이터 1/3)

train (100개의 데이터)
test (50개의 데이터)

POSTECH

의사결정나무(Decision Tree)

☑ tree패키지에 있는 tree함수

```
help("tree")
```

tree {tree} R Documentation

Fit a Classification or Regression Tree

Description

A tree is grown by binary recursive partitioning using the response in the specified formula and choosing splits from the terms of the right-hand-side.

Usage

```
tree(formula, data, weights, subset,
     na.action = na.pass, control = tree.control(nobs, ...),
     method = "recursive.partition",
     split = c("deviance", "gini"),
     model = FALSE, x = FALSE, y = TRUE, wts = TRUE, ...)
```

Arguments

formula	A formula expression. The left-hand-side (response) should be either a numerical vector when a regression tree will be fitted or a factor, when a classification tree is produced. The right-hand-side should be a series of numeric or factor variables separated by +; there should be no interaction terms. Both . and - are allowed: regression trees can have offset terms.
data	A data frame in which to preferentially interpret formula, weights and subset.
weights	Vector of non-negative observational weights; fractional weights are allowed.
subset	An expression specifying the subset of cases to be used.

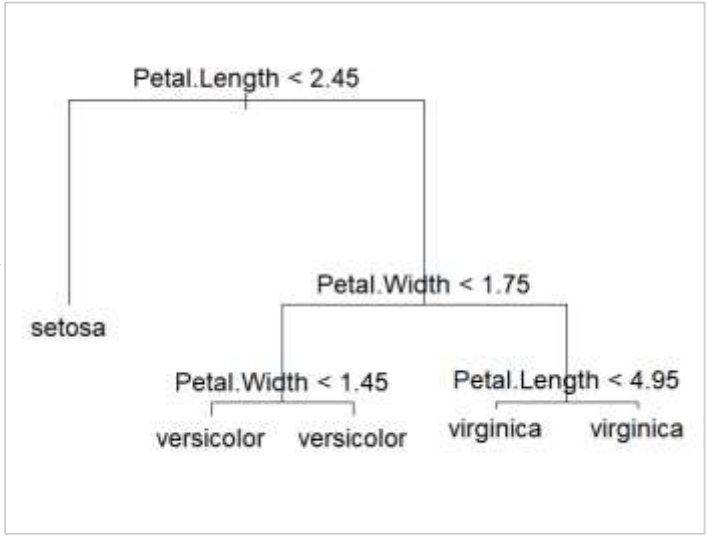
POSTECH

의사결정나무(Decision Tree)

☑ 의사결정나무 함수 : tree(종속변수~x1+x2+x3+x4, data=)

Step1

```
# step1 : growing tree
treemod<-tree(Species~., data=train)
treemod
plot(treemod)
text(treemod,cex=1.5)
```



- plot(treemod) : 의사결정나무 분지를 그림으로 표현
- cex : 폰트 사이즈

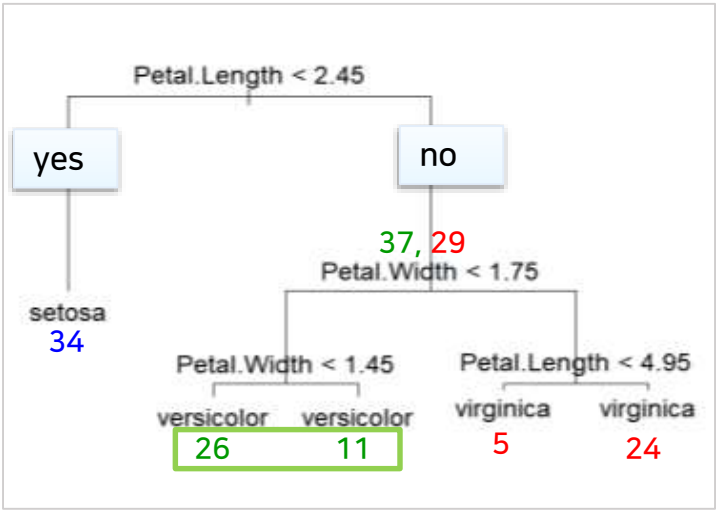
POSTECH

의사결정나무(Decision Tree)

- 학습데이터의 tree결과
- tree의 결과(*는 터미널노드) : 마디 6에서는 더 이상 분지할 필요 없음

```
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 100 219.500 versicolor ( 0.34000 0.35000 0.31000 )
   2) Petal.Length < 2.45 34   0.000 setosa ( 1.00000 0.00000 0.00000 ) *
   3) Petal.Length > 2.45 66  91.250 versicolor ( 0.00000 0.53030 0.46970 )
     6) Petal.Width < 1.75 37  20.820 versicolor ( 0.00000 0.91892 0.08108 )
      12) Petal.Width < 1.45 26   0.000 versicolor ( 0.00000 1.00000 0.00000 ) *
      13) Petal.Width > 1.45 11  12.890 versicolor ( 0.00000 0.72727 0.27273 ) *
     7) Petal.Width > 1.75 29   8.700 virginica ( 0.00000 0.03448 0.96552 )
      14) Petal.Length < 4.95 5   5.004 virginica ( 0.00000 0.20000 0.80000 ) *
      15) Petal.Length > 4.95 24   0.000 virginica ( 0.00000 0.00000 1.00000 ) *
```



POSTECH

의사결정나무(Decision Tree)

Step2

- 최적tree모형을 위한 가지치기(pruning) : cv.tree(tree모형결과, FUN=)
- 아래 결과에서 교차검증오차(deviance)의 값이 최소가 되는 노드 수를 선택

```
# step2 : pruning using cross-validation
cv.trees<-cv.tree(treemod, FUN=prune.misclass)
cv.trees
plot(cv.trees)
```

```
> cv.trees
$size
[1] 5 3 2 1

$dev
[1]  5  5 43 78

$k
[1] -Inf    0   27   34
```

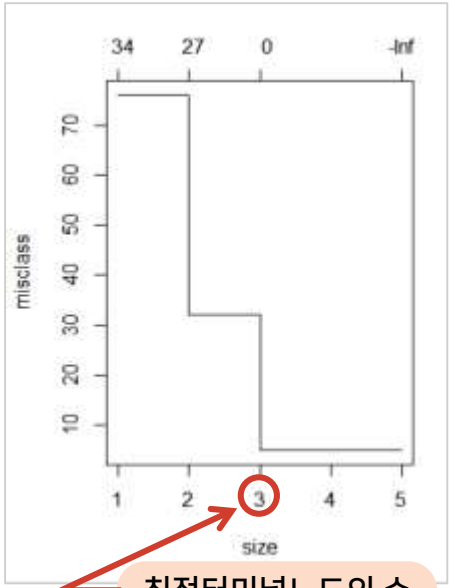
최적터미널노드의 수

dev는 교차검증오차(deviance) FUN= prune.misclass인 경우 misclass를 의미

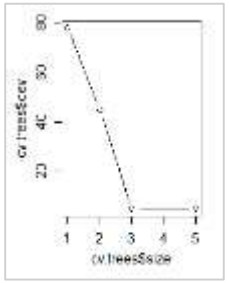
k는 복잡도계수(complexity parameter)

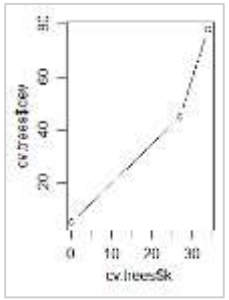
```
par(mfrow=c(1,2))
plot(cv.trees$size, cv.trees$dev, type="b")
plot(cv.trees$k, cv.trees$dev, type="b")
```

dev변화에 따른 k와 size의 변화 시각화



최적터미널노드의 수





POSTECH

의사결정나무(Decision Tree)

Step2

pruning(가지치기) : cv.tree함수를 이용하여 최적 터미널노드를 탐색

```
help(prune.misclass)
```

prune.tree {tree} R Documentation

Cost-complexity Pruning of Tree Object

Description

Determines a nested sequence of subtrees of the supplied tree by recursively "snipping" off the least important splits.

Usage

```
prune.tree(tree, k = NULL, best = NULL, newdata, nwts,
           method = c("deviance", "misclass"), loss, eps = 1e-3)

prune.misclass(tree, k = NULL, best = NULL, newdata,
               nwts, loss, eps = 1e-3)
```

Arguments

`tree`	fitted model object of class `tree`. This is assumed to be the result of some function that produces an object with the same named components as that returned by the `tree()` function.
`k`	cost-complexity parameter defining either a specific subtree of `tree` (`k` a scalar) or the (optional) sequence of subtrees minimizing the cost-complexity measure (`k` a vector). If missing, `k` is determined algorithmically.
`best`	integer requesting the size (i.e. number of terminal nodes) of a specific subtree in the cost-complexity sequence to be returned. This is an alternative way to select a subtree than by supplying a scalar cost-complexity parameter `k`. If there is no tree in the sequence of the requested size, the next largest is returned.

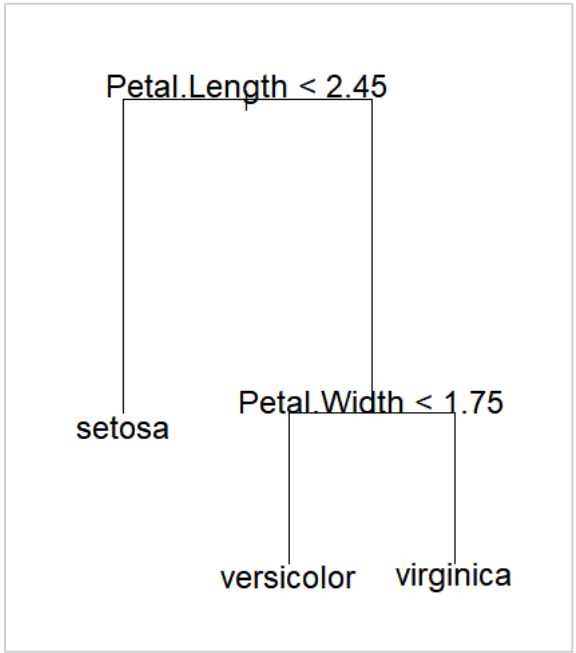
POSTECH

의사결정나무(Decision Tree)

최종 tree모형(iris data)

iris data는 best=3

```
# final tree model with the optimal node
prune.trees<-prune.misclass(treemod, best=3)
plot(prune.trees)
text(prune.trees,pretty=0, cex=1.5)
```



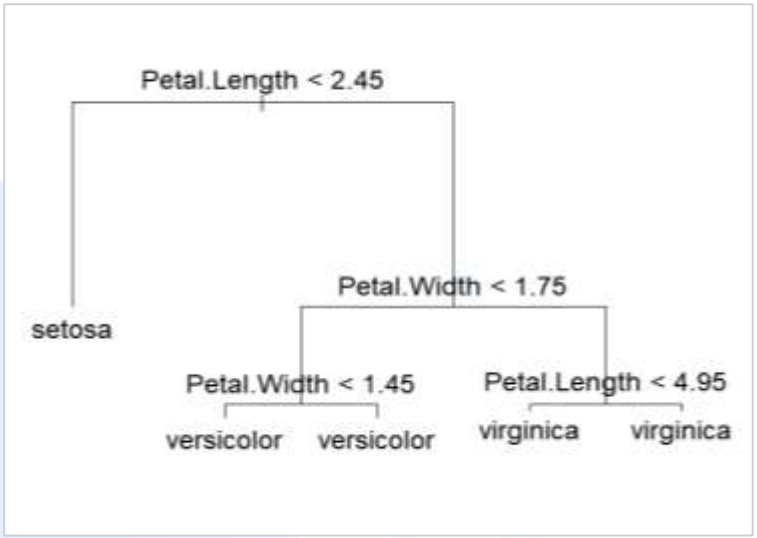
prune.trees은 최종모형의 이름
(최종터미널노드=3)

POSTECH

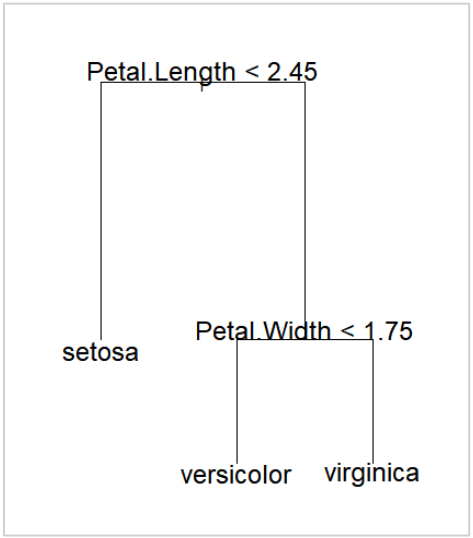
의사결정나무(Decision Tree)

- pruning(가지치기) 전과 후의 tree

원래 tree



가지치기 후 tree



POSTECH

의사결정나무(Decision Tree)

☑ 의사결정나무결과 정확도 : test data에 대한 정확도

Step3

```
# step 3: classify test data
treepred<-predict(prune.trees,test,type='class')
# classification accuracy
confusionMatrix(treepred,test$Species)
```

```
Confusion Matrix and Statistics

                 Reference
Prediction    setosa versicolor virginica
  setosa          16          0         0
  versicolor       0         15         2
  virginica        0          0        17

Overall Statistics

               Accuracy : 0.96
```

실제범주

예측범주