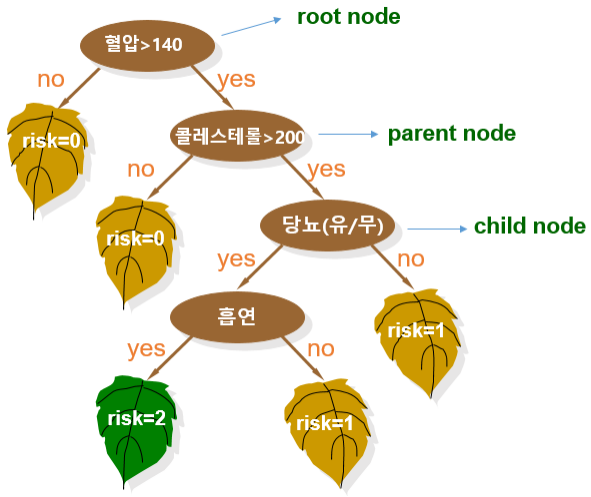
	단위별 학습내용 (Week12)
wk12-1	의사결정나무 (Decision Tree) I
wk12-2	의사결정나무 (Decision Tree) II
wk12-3	랜덤포레스트 (Random Forest)

Wk12-1 : 의사결정나무 (Decision Tree) I

1. 의사결정나무 (Decision Tree)

•의사결정나무 (Decision Tree)

기계학습 중 하나로 의사결정 규칙을 나무 형태로 분류해나가는 분석 기법



- 분석 과정이 직관적이고 이해하기 쉬움
- 연속형/범주형 변수를 모두 사용할 수 있음
- 분지규칙은 불순도를 최소화시킴

⇩

범주들이 섞여있는 정도

1. 의사결정나무 (Decision Tree)

step 1: tree 형성 (Growing tree)

step 2: tree 가지치기 (pruning tree)

step 3: 최적 tree로 분류 (classification)

1. 의사결정나무 (Decision Tree)

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

```
# lec12_1_tree.R
# Decision tree
# use package "tree"

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

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

의사결정나무 수행을 위한 패키지 설치 (tree)
라이브러리 설정

caret 라이브러리 설정 :
오분류율 교차표(confusion matrix) 생성을 위한 패키지

1. 의사결정나무 (Decision Tree)

- iris 데이터 (iris.csv)

input변수(독립변수) output변수(종속변수, 타겟변수)

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

1. 의사결정나무 (Decision Tree)

• iris 데이터 (학습데이터와 검증데이터의 분할)

```
# set working directory
setwd("D:/tempstore/moocr/wk12")

# read csv file
iris<-read.csv("iris.csv")
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)

1. 의사결정나무 (Decision Tree)

• tree패키지에 있는 tree함수

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

tree {tree} R Documentatio

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, ...)
```

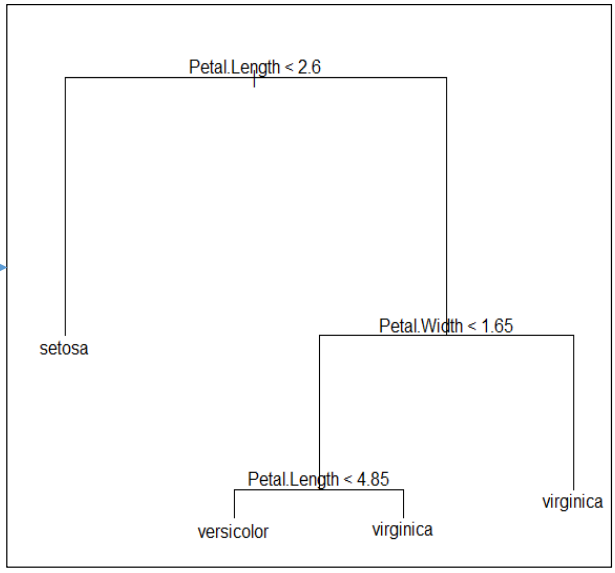
Arguments

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.

1. 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)
```



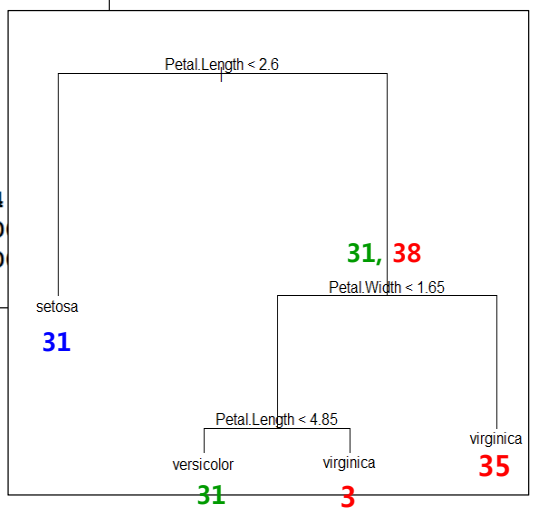
treemod는 iris데이터의 범주를 분리해주는 분지결과를 저장
plot(treemod) – 의사결정나무 분지를 그림으로 표현
cex-폰트 사이즈 (1)

1. 의사결정나무 (Decision Tree)

• 학습데이터의 tree결과

• tree의 결과 (*는 터미널노드) : 마디 6에서는 더이상 분지할 필요 없음

```
> treemod<-tree(Species~., data=train)
> treemod
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 100 218.80 virginica ( 0.31000 0.31000 0.38000 )
   2) Petal.Length < 2.6 31   0.00 setosa ( 1.00000 0.00000 0.00000 ) *
   3) Petal.Length > 2.6 69  94.94 virginica ( 0.00000 0.44928 0.55072 )
     6) Petal.Width < 1.65 34  20.29 versicolor ( 0.00000 0.91176 0.08824
      12) Petal.Length < 4.85 29   0.00 versicolor ( 0.00000 1.00000 0.00
      13) Petal.Length > 4.85 5   6.73 virginica ( 0.00000 0.40000 0.6000
     7) Petal.Width > 1.65 35   0.00 virginica ( 0.00000 0.00000 1.00000
```



1. 의사결정나무 (Decision Tree)

- 최적tree모형을 위한 가지치기(pruning) : cv.tree(tree모형결과, FUN=)
- 아래 결과에서 복잡도계수(cost complexity parameter)의 값이 최소가 되는 노드수를 선택

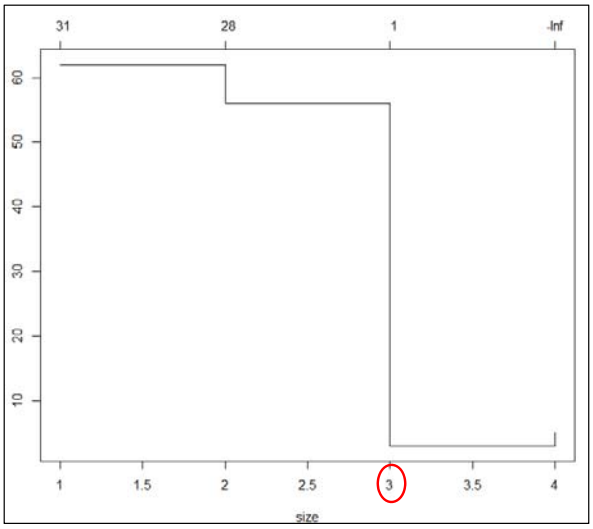
Step2

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

```
> cv.tr<-cv.tree(treemod, FUN=prune.misclass)
> cv.tr
$size
[1] 4 3 2 1

$dev
[1]  8  6 56 69

$k
[1] -Inf    1   28   31

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

최적터미널노드의 수

k는 복잡도계수(complexity parameter)



최적터미널노드의 수

1. 의사결정나무 (Decision Tree)

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

Step2

```
help(cv.tree)
```

cv.tree함수에서 가지치기함수는 prune.tree 혹은 prun.misclass를 사용할 수 있다.

cv.tree {tree} R Documentation

Cross-validation for Choosing Tree Complexity

Description

Runs a K-fold cross-validation experiment to find the deviance or number of misclassifications as a function of the cost-complexity parameter k.

Usage

```
cv.tree(object, rand, FUN = prune.tree, K = 10, ...)
```

Arguments

object	An object of class "tree".
rand	Optionally an integer vector of the length the number of cases used to create object, assigning the cases to different groups for cross-validation.
FUN	The function to do the pruning.
K	The number of folds of the cross-validation.
...	Additional arguments to FUN.

1. 의사결정나무 (Decision Tree)

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

Step2

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

최적 터미널노드의 수

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.

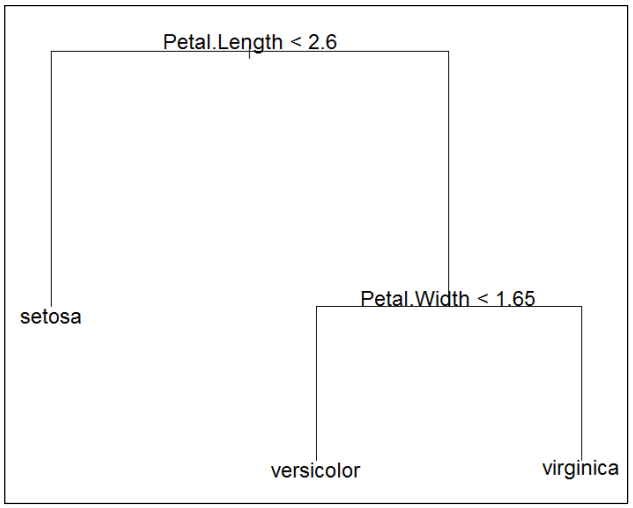
1. 의사결정나무 (Decision Tree)

- 최종 tree모형 (iris data)

iris data는 best=3

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

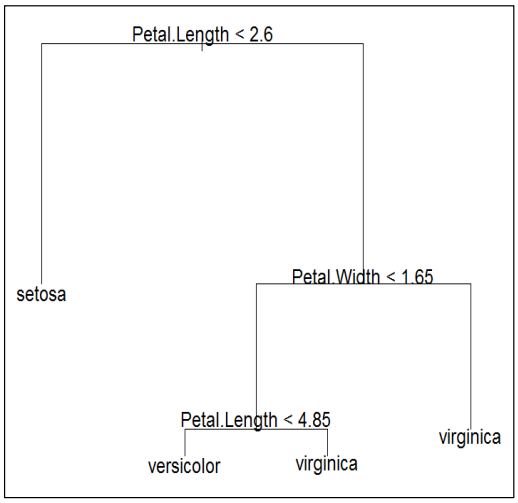
prune.tr은 최종모형의 이름(최종터미널노드=3)



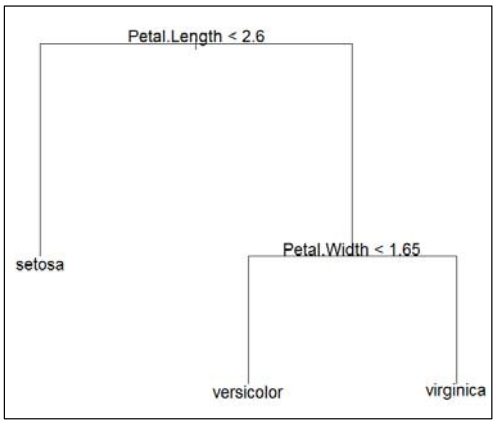
1. 의사결정나무 (Decision Tree)

- pruning(가지치기) 전과 후의 tree

원래 tree



가지치기 후 tree



1. 의사결정나무 (Decision Tree)

- 의사결정나무결과 정확도 : test data에 대한 정확도

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

Step3

```
Confusion Matrix and Statistics

          Reference
Prediction    setosa versicolor virginica
  setosa          19          0         0
  versicolor       0         17         1
  virginica        0          2        11

Overall Statistics

               Accuracy : 0.94
```

예측범주 / 실제범주