빅데이터 처리 (화요일 (1:3교시))

6주차 강의

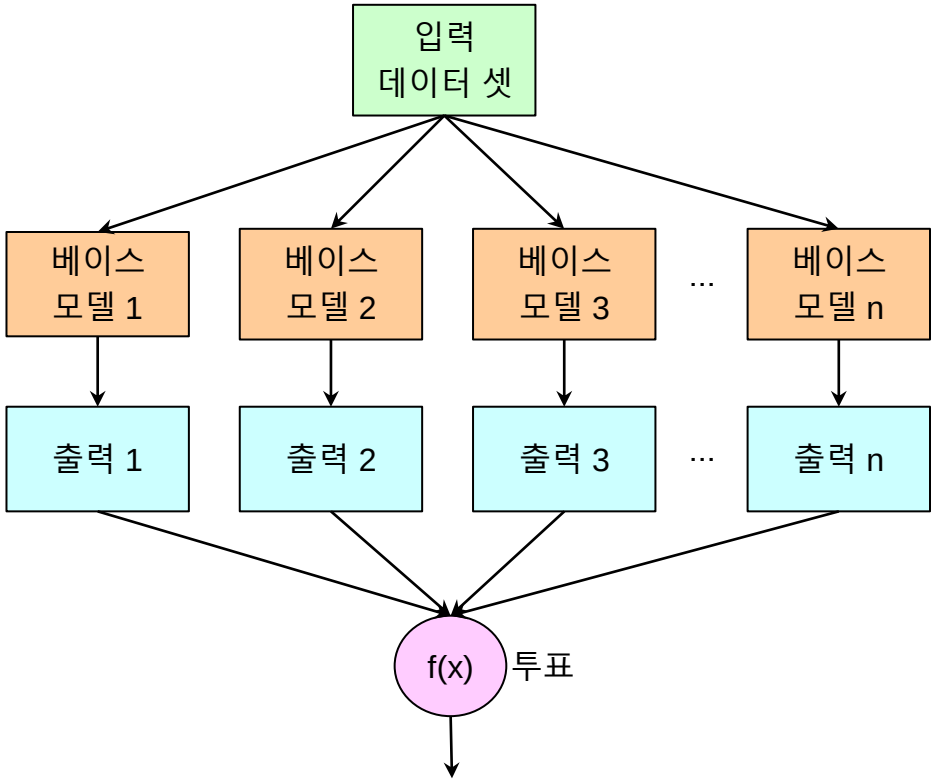
분류
앙상블학습법

2022.04.12

Institutor ; JS LEE

앙상블학습법 -개요

- 앙상블(ensemble) 모형은 여러 개의 분류모형에 의한 결과를 종합하여 분류의 정확도를 높이는 방법임
- 동일 데이터 셋을 이용하여 개별적으로 결과를 예측하는 모델들의 결과들을 결합하는 방법임 (집단지성)
- 보통은 투표 또는 가중투표를 통해 앙상블 결과를 출력함
- 대표적인 방법으로는 배깅(bagging), 부스팅(boosting)이 존재하며, 랜덤포레스트(random forest)는 배깅의 개념과 속성(또는 변수)의 임의적으로 선택하는 방법을 결합한 방법임
- 앙상블 기법은 다양한 Weak Learner를 통해 Strong Learner를 만들어가는 과정
  - ✓ 약학습기(약분류기, Weak Learner) : 무작위 선정이 아닌 성공확률이 높은, 즉 오차율이 일정 이하(50% 이하)인 학습 규칙
  - ✓ 강학습기(강분류기, Strong Learner) Weak Learner로부터 만들어내는 강력한 학습 규칙



[앙상블 모델]

앙상블학습법 -개요

Label	Learner1	Learner2	Learner3	voting
1	1	0	0	0
0	1	0	0	0
1	0	1	1	1
1	0	0	0	0
1	1	1	1	1
0	1	1	1	1
0	0	0	1	0
0	0	1	0	0
정확도	0.5	0.5	0.5	0.625

각각의 base learner (Learner1, Learner2, Learner3)는 0.5의 정확도(최소 정확도)를 가짐

이 base learner의 결과에서 다수로 나온 결과를 voting(투표)했을 때, 정확도는 0.625로 상승함.

앙상블은 각각 base leaner의 결과를 결합하여 더 좋은 성능을 내는 Machine Learning 기법임

앙상블학습법 -개요

앙상블 모델링의 장점

- 평균을 취함으로써 편의(편향)를 최소화:
  : 치우침이 있는 여러 모형의 평균을 취하면, 어느 쪽에도 치우치지 않는 결과(평균)를 얻게 됨

- 분산 감소
  : 한 개 모형으로부터의 도출 된 결과보다 여러 모형의 결과를 결합하면 변동이 작아짐

- 과적합 방지:
  : 여러 모형으로부터 예측을 결합하면 과적합의 여지가 줄어듦

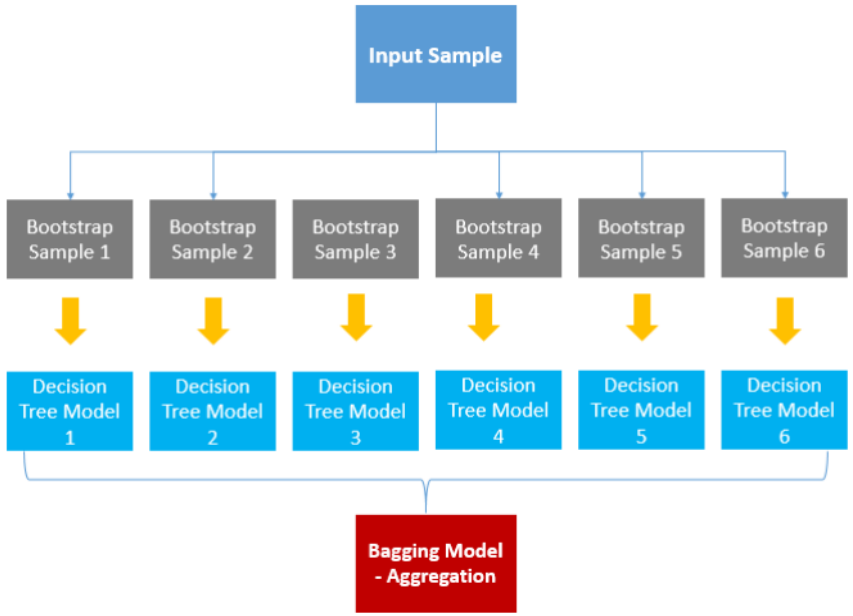
앙상블학습법 -Bagging

Bagging => bootstrap aggregating 의 준말

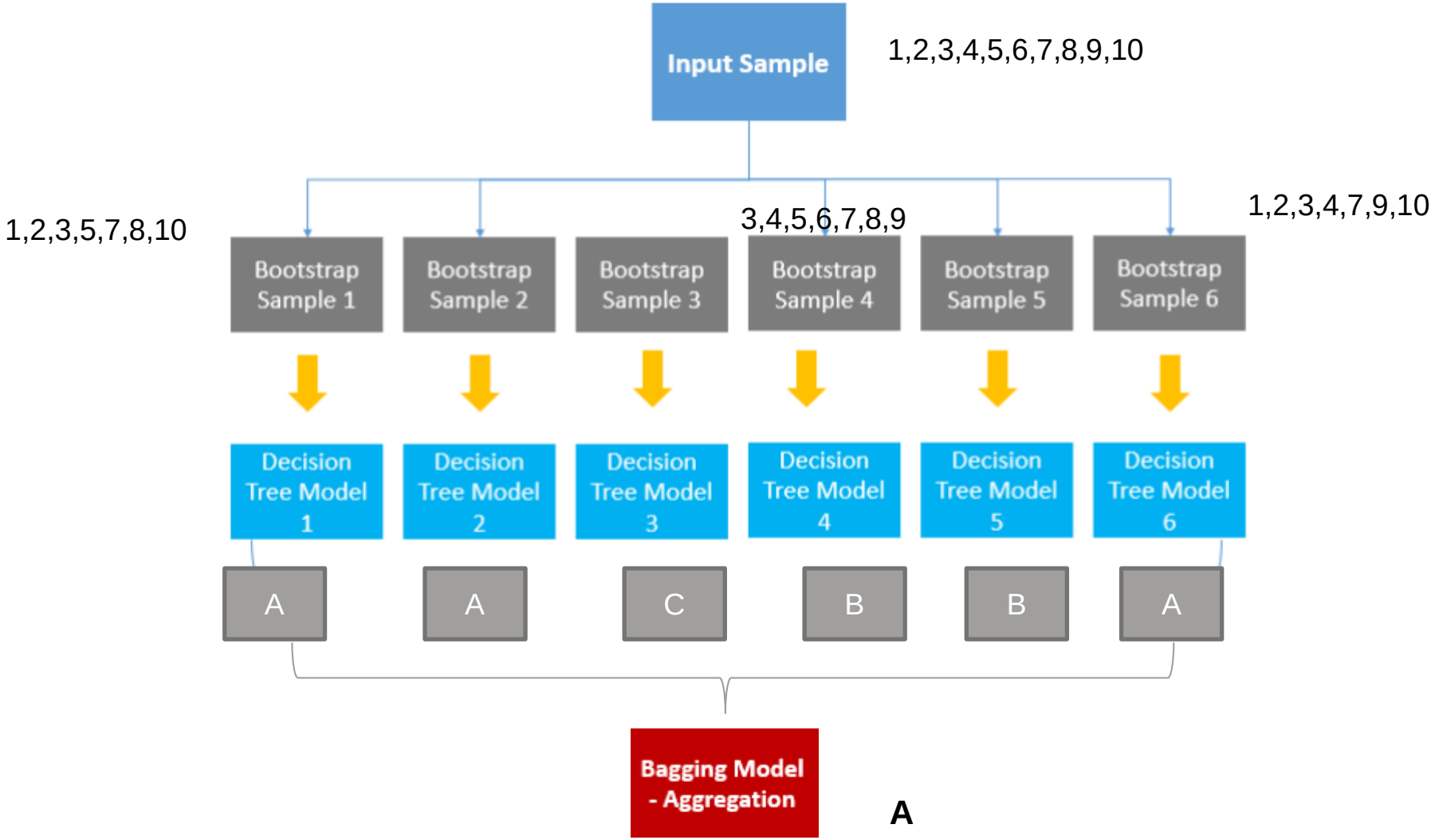
- Bagging은 샘플을 여러 번 뽑아 (Bootstrap) 각 모델을 학습시켜 결과물을 집계(Aggregation)하는 방법으로. Bootstrap Aggregation의 축약어 임

  * Bootstrap Sampling : 전체 데이터에서 N개의 sample을 복원추출

- N개의 bootstrap 샘플링된 표본에서 병렬로 학습하고. N개의 학습자의 결과를 투표(voting) 방식으로 예측 값을 결정함



앙상블학습법 -Bagging



앙상블학습법 -Bagging

<실습> Iris 데이터에 bagging 알고리즘을 적용해보자

- 필요 패키지 : adabag / 사용함수: bagging()
- bagging(formaula, data=train_data, mfial= number)
  * mfinal= 반복 수 또는 트리의 수(디폴트=100)

```
install.packages("adabag")
library(adabag)
iris.bagging <- bagging(Species~., data=iris, mfinal=10)
```

10번 반복복원추출하여 10개 트리 생성

```
# 분류 시 변수별 중요도 , 모델결과$importance
iris.bagging$importance
```

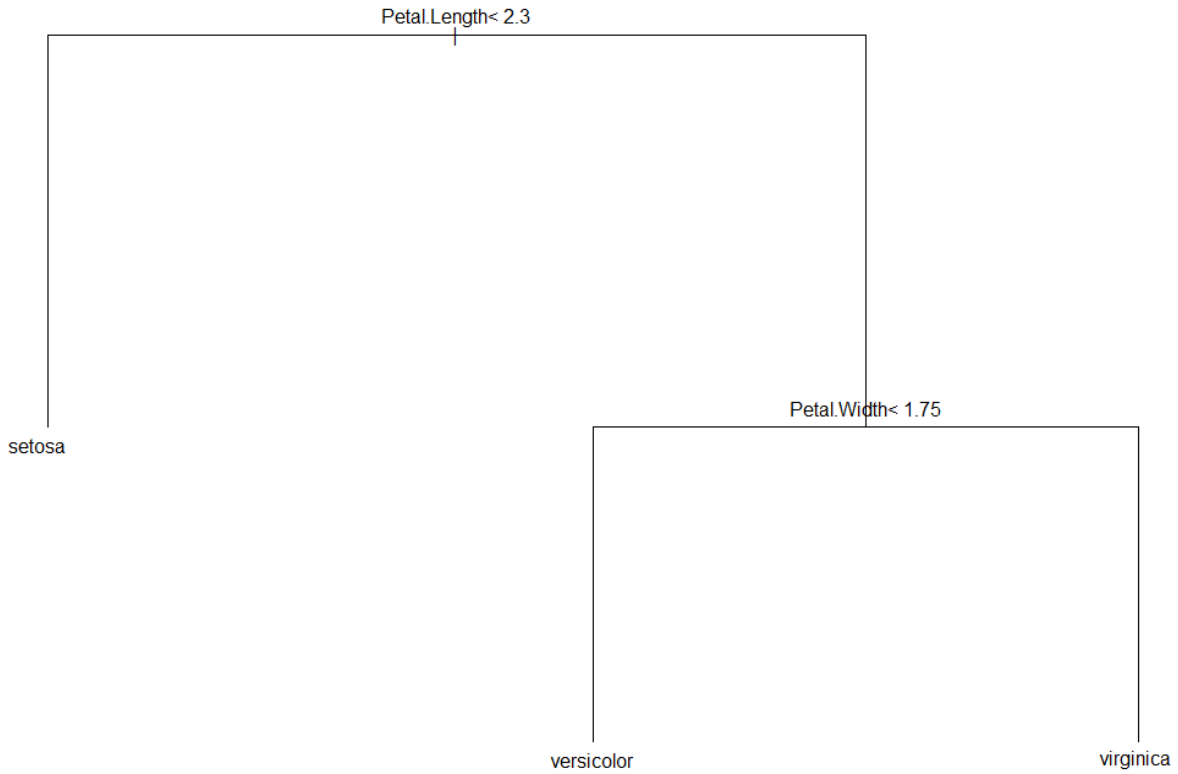
```
Petal.Length  Petal.Width Sepal.Length  Sepal.Width
    74.17114     25.82886      0.00000      0.00000
```

- 변수의 중요도는 각 트리에서 변수에 의해 주어지는 지니지수의 이익(gain)(또는 불확실성의 감소량)을 고려한 측도임.

앙상블학습법 -Bagging

〈실습〉 Iris 데이터에 bagging 알고리즘을 적용해보자

```
# 도식화

plot(iris.bagging$trees[[10]])

text(iris.bagging$trees[[10]])
```



Petal.Length	Petal.Width	Sepal.Length	Sepal.Width
74.17114	25.82886	0.00000	0.00000

앙상블학습법 -Bagging

〈실습〉 Iris 데이터에 bagging 알고리즘을 적용해보자

모형평가를 위해 ConfusionMatrix를 사용한다.

```
expect2 <- predict(iris.bagging, iris_test, type="response")
 # -> Species를 예측하여 분류한 변수를 팩터화 한다.
expect2$class<-as.factor(expect2$class)
confusionMatrix(expect2$class,iris_test$Species,mode="everything")
```

```
Confusion Matrix and Statistics

            Reference
Prediction   setosa versicolor virginica
  setosa         10          0         0
  versicolor      0         10         1
  virginica       0          0         9

Overall Statistics

               Accuracy : 0.9667
                 95% CI : (0.8278, 0.9992)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 2.963e-13

                  Kappa : 0.95

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            1.0000           0.9000
Specificity                 1.0000            0.9500           1.0000
Pos Pred Value              1.0000            0.9091           1.0000
Neg Pred Value              1.0000            1.0000           0.9524
Precision                   1.0000            0.9091           1.0000
Recall                      1.0000            1.0000           0.9000
F1                          1.0000            0.9524           0.9474
Prevalence                  0.3333            0.3333           0.3333
Detection Rate              0.3333            0.3333           0.3000
Detection Prevalence        0.3333            0.3667           0.3000
Balanced Accuracy           1.0000            0.9750           0.9500
> |
```

plot 생성시 아래와 같은 에러 발생시

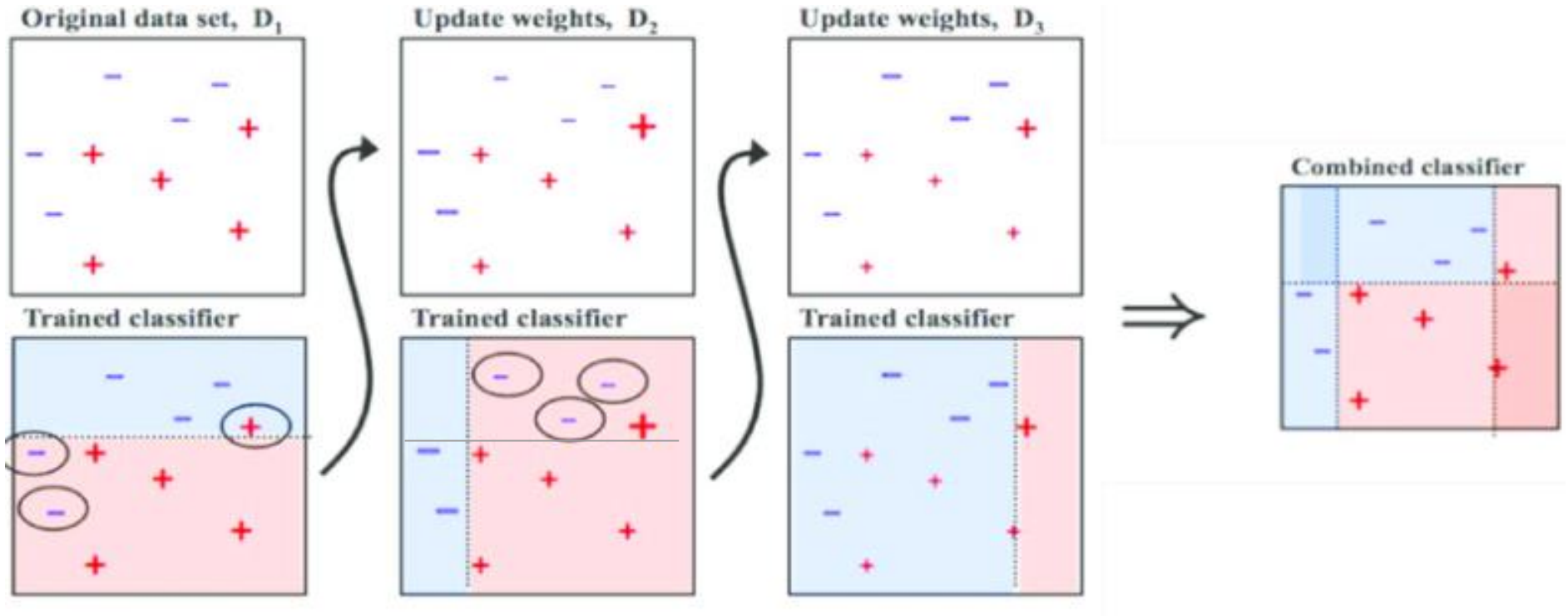
Error in plot.new() : figure margins too large

# 프롯의 마진을 최대로 해준다.
par("mar")
par(mar=c(1,1,1,1))

앙상블학습법 – Boosting

- Boosting의 기본 아이디어는 강력한 학습자를 만들기 위해 일련의 약한 학습자를 순차적으로 학습시키는 것임.
- 병렬하게 훈련되고 예측은 학습자에 대한 선호 없이 집계되는 Bagging의 경우와 달리 Boosting은 순차적으로 훈련하고, 오분류된 분류에 더 많은 가중치를 부과함
- 정분류된 데이터는 추출될 확률을 줄이고, 오분류된 데이터는 추출될 확률을 높여서 모형이 오분류된 데이터를 더 강하게 학습할 수 있도록 도와주는 방법임
- 모형결합시에도 정확도가 높은 모형에 가중치를 더 주는 방식으로 결합함
- Bagging은 학습자 간의 독립성을 활용하여 분산을 줄이기 위해 병렬로 학습하는 반면, Boosting은 학습자 간의 의존성을 이용하여 편향 및 분산을 줄이기 위해 순차적으로 학습함
- 아다부스팅(AdaBoosting: adaptive boosting)은 가장 많이 사용되는 부스팅 알고리즘임

앙상블학습법 -Boosing



앙상블학습법 -Boosting

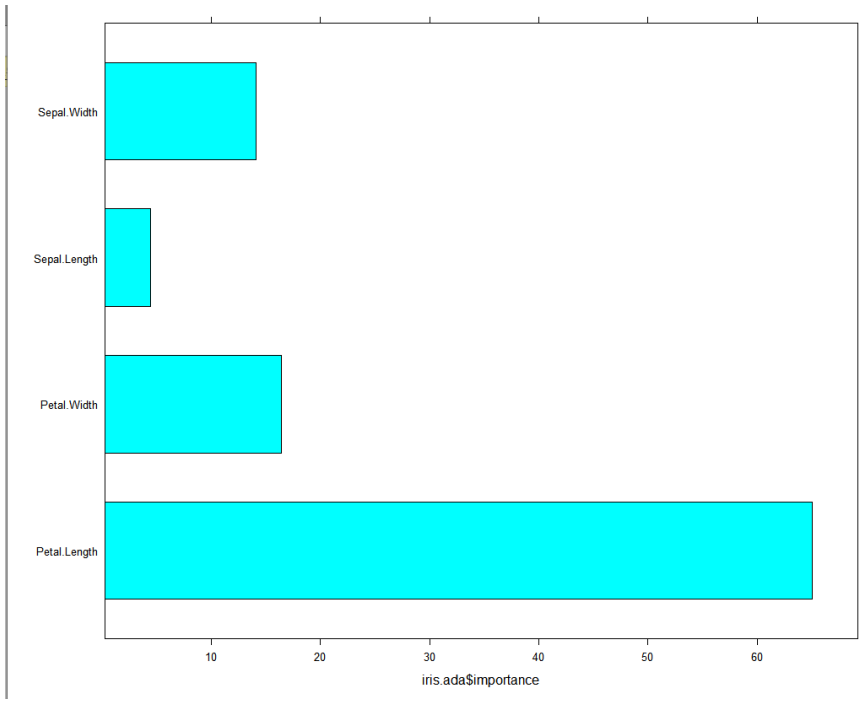
<실습> Iris 데이터에 boosting알고리즘을 적용해보자

- 필요 패키지 : adabag / 사용함수: boosting()
- boosting(formaula, data=train_data, mfial= number)
  * mfinal= 반복 수 또는 트리의 수(디폴트=100)

```
iris.ada <- boosting(Species~., data=iris_train, mfinal=10)

iris.ada$importance    # 분류 영향정도
```

```
Petal.Length  Petal.Width Sepal.Length  Sepal.Width
   65.027405    16.432576     4.458815    14.081204
```

```
barchart(iris.ada$importance)   # 도식화하여 변수의 중요도
```

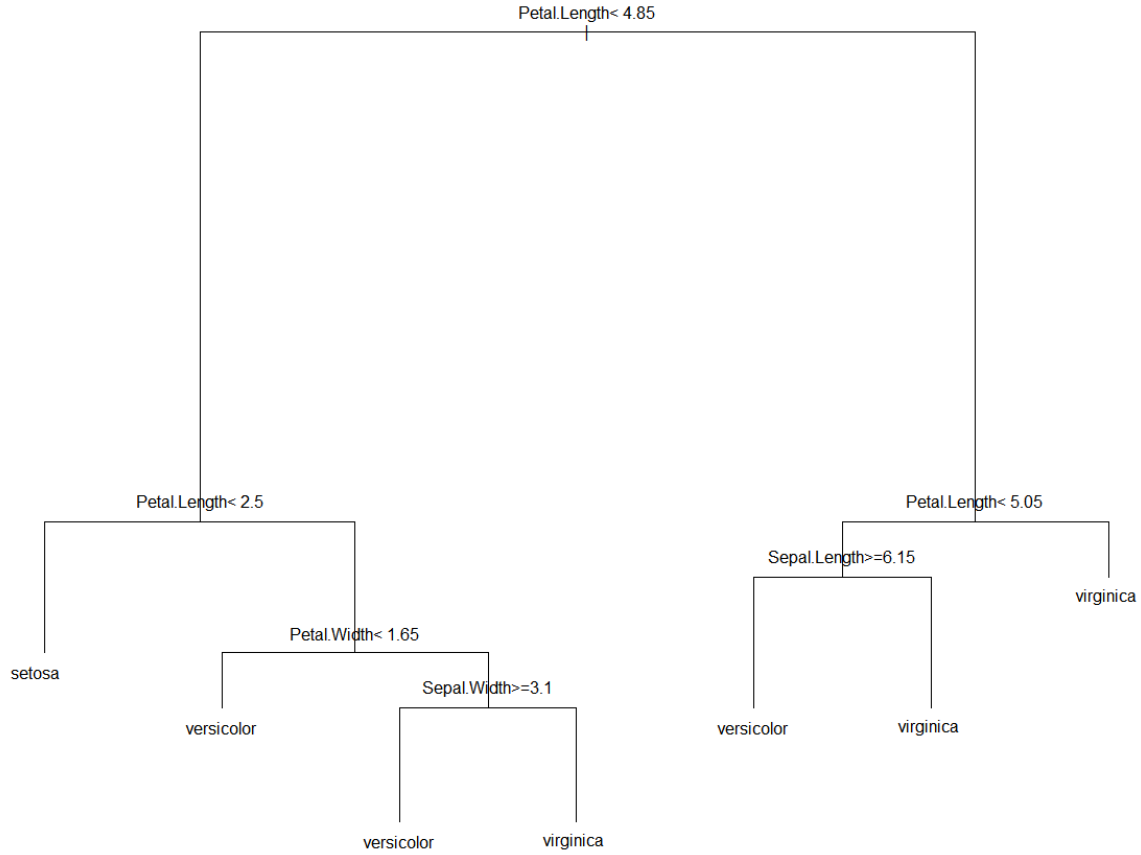


앙상블학습법 -Boosting

〈실습〉 Iris 데이터에 boosting알고리즘을 적용해보자

```
plot(iris.ada$trees[[10]])   #  tree 그림 그리기
text(iris.ada$trees[[10]])
```

AdaBoosting의 Tree는 보다 더 leaf-wise하고, depth 또한 더 깊은 것을 확인할 수 있



앙상블학습법 -Boosting

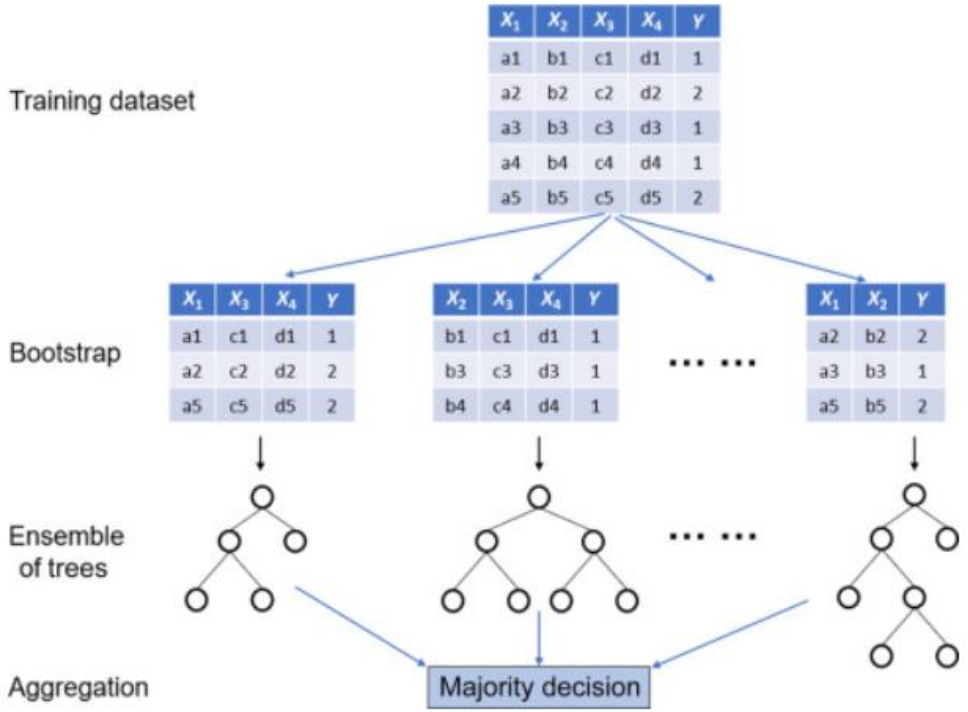
〈실습〉 Iris 데이터에 boosting알고리즘을 적용해보자

```
expect3 <- predict(iris.ada, iris_test, type="response")

expect3$class<-as.factor(expect3$class)

confusionMatrix(expect3$class,iris_test$Species,mode
="everything")
```

```
Confusion Matrix and Statistics

            Reference
Prediction   setosa versicolor virginica
  setosa         10          0         0
  versicolor      0          9         0
  virginica       0          1        10

Overall Statistics

               Accuracy : 0.9667
                 95% CI : (0.8278, 0.9992)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 2.963e-13

                  Kappa : 0.95

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            0.9000           1.0000
Specificity                 1.0000            1.0000           0.9500
Pos Pred Value              1.0000            1.0000           0.9091
Neg Pred Value              1.0000            0.9524           1.0000
Precision                   1.0000            1.0000           0.9091
Recall                      1.0000            0.9000           1.0000
F1                          1.0000            0.9474           0.9524
Prevalence                  0.3333            0.3333           0.3333
Detection Rate              0.3333            0.3000           0.3333
Detection Prevalence        0.3333            0.3000           0.3667
Balanced Accuracy           1.0000            0.9500           0.9750
```

앙상블학습법 -RandomForest

- 랜덤포리스트(random forest)는 배깅에 랜덤 과정을 추가한 방법임
- 원 자료로부터 붓스트랩 샘플을 추출하고, 각 붓스트랩 샘플에 대해 트리를 형성해 나가는 과정은 배깅과 유사하나, 예측변수들을 임의로 추출하고, 추출된 변수 내에서 최적의 분할을 만들어 나가는 방법을 사 용함
- 새로운 자료에 대한 예측은 분류(classification)의 경우는 다수결(majority votes)로, 회귀 (regression)의 경우에는 평균을 취하는 방법을 사용함 (다른 앙상블모형과 유사)



refer: Machine Learning for Subsurface Characterization

앙상블학습법 -RandomForest

<실습>146명의 전립선 암 환자의 자료(stagec)을 이용하여 Randomforest 알고리즘 실습

7개의 예측변수를 이용하 여 범주형의 반응변수(ploidy)를 예측(또는 분류)한다

데이터 stagec는 rpart 패키지에 존재함

```
library(rpart)
data(stagec)
```

Data셋 소개

- pgtime : Time to progression or last follow-up (years)
- pgstat : 1 = progression observed, 0 = censored
- age : age in years
- eet : early endocrine therapy, 1 = no, 2 = yes
- g2 : percent of cells in G2 phase, as found by flow cytometry
- grade: grade of the tumor, Farrow system
- gleason : grade of the tumor, Gleason system
- ploidy (종속변수) the ploidy status of the tumor, from flow cytometry. Values are **diploid, tetraploid**, and **aneuploid**

앙상블학습법 -RandomForest

1. 데이터 셋을 살펴보기

View(stagec)

NA 들이 존재함을 알 수 있음

2. NA 제거하기 위해, 어떠한 변수에 NA 가 있는지 알아본다.

```
colSums(is.na(stagec))
```

```
pgtime  pgstat     age     eet      g2   grade gleason  ploidy
     0       0       0       2       7       0       3       0
```

⇒ age, eet, grade에 총 12개 존재함을 알 수 있음

3. 분석질을 높이기 위해 NA 가 존재하는 행은 제외한다

```
stagec1 <- na.omit(stagec)
```

Environment 창에서 확인하면, stagec:146건, stagec1: 134건

	pgtime	pgstat	age	eet	g2	grade	gleason	ploidy
1	6.1	0	64	2	10.26	2	4	diploid
2	9.4	0	62	1	NA	3	8	aneuploid
3	5.2	1	59	2	9.99	3	7	diploid
4	3.2	1	62	2	3.57	2	4	diploid
5	1.9	1	64	2	22.56	4	8	tetraploid
6	4.8	0	69	1	6.14	3	7	diploid
7	5.8	0	75	2	13.69	2	NA	tetraploid
8	7.3	0	71	2	NA	3	7	aneuploid
9	3.7	1	73	2	11.77	3	6	diploid
10	15.9	0	64	2	27.27	3	7	tetraploid
11	6.3	0	65	2	19.34	3	7	tetraploid
12	2.9	1	58	2	14.82	4	8	tetraploid
13	1.5	1	70	2	10.22	3	8	diploid
14	14.5	0	67	2	15.66	2	6	tetraploid
15	4.2	1	66	2	17.79	3	7	tetraploid
16	1.7	1	74	2	11.11	3	8	diploid
17	5.0	0	70	2	11.44	2	5	diploid
18	13.2	0	57	2	14.78	2	4	tetraploid
19	10.9	0	63	2	54.93	3	8	tetraploid
20	13.0	0	65	2	24.58	3	7	tetraploid
21	11.4	0	62	2	27.79	2	5	tetraploid
22	2.6	1	72	2	14.86	3	6	tetraploid

앙상블학습법 -RandomForest

5. 결측값이 제거된 134개의 자료를 이용하여 모형구축을 위한 훈련용 자료 (training data)와 모형의 성능을 검증하기위한 검증용 자료(test data)를 70%와 30%로 구성한다.

랜덤샘플링을 활용하여 구분

set.seed(1234) => 초기값 설정

Random Sampling을 통해 70%와 30%로 구분한다

ind <- sample(2, nrow(stagec3), replace=TRUE, prob=c(0.7, 0.3))

70%는 trainData, 30% testData

trainData <- stagec1[ind==1,] # n=102개
testData <- stagec1[ind==2,] # n=32

앙상블학습법 -RandomForest

6. randomForest()함수를 이용하여 모델 구축

```
# randomForest 설치 및 로딩
  install.packages("randomForest")
 library(randomForest)


# 알고리즘 적용
randomForest(Formula ., data=trainData, ntree= number)
# ntree ; tree 생성 개수 ( default =500)


 rf <- randomForest(ploidy ~ ., data=trainData, ntree=100)
```

7. 변수의 중요도를 알아보기

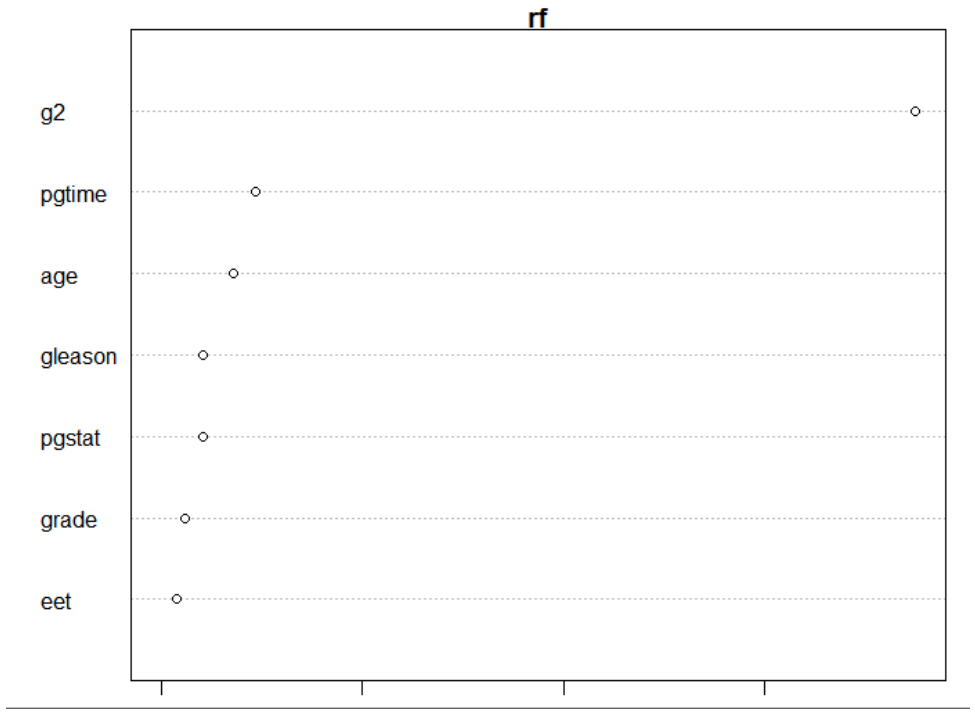
importance(), varImpPlot() 활용

앙상블학습법 -RandomForest

importance(rf)

```
          MeanDecreaseGini
pgtime          4.6800225
pgstat          2.0635061
age             3.5726107
eet             0.7875501
g2             37.5032896
grade           1.2084410
gleason         2.0820408
```

varImpPlot(rf)



g2가 분류에 가장 큰 영향을 미침을 알 수 있음

각 변수의 중요도를 나타내는 그림으로, 해당 변수로부터 분할이 일어날 때 불순도 (impurity)의 감소가 얼마나 일어나는지를 나타내는 값임(불순도의 감소가 클수록 순수도가 증가함).

지니 지수(Gini index)는 노드의 불순도를 나타내는 값임

앙상블학습법 -RandomForest

8. 오류율 살펴보기

- plot() 함수를 이용하여 트리 수에 따른 종속변수의 범주별 오분류를 볼 수 있음
- 검은색은 전체 오 분류율을 나타냄
- 오분류율이 1로 나타난 범주는 aneuploid 범주로 개체수가 매우 작은 범주에서 발생된 결과이다.

```
plot(rf)
 legend("topright",colnames(rf$err.rate),cex=0.8,fill=1:4) #범례 정리
* “top” : 범례의 위치
    colnames : 범주명
    cex : 범주의 크기
    fill : 범주 색 나타내기
```

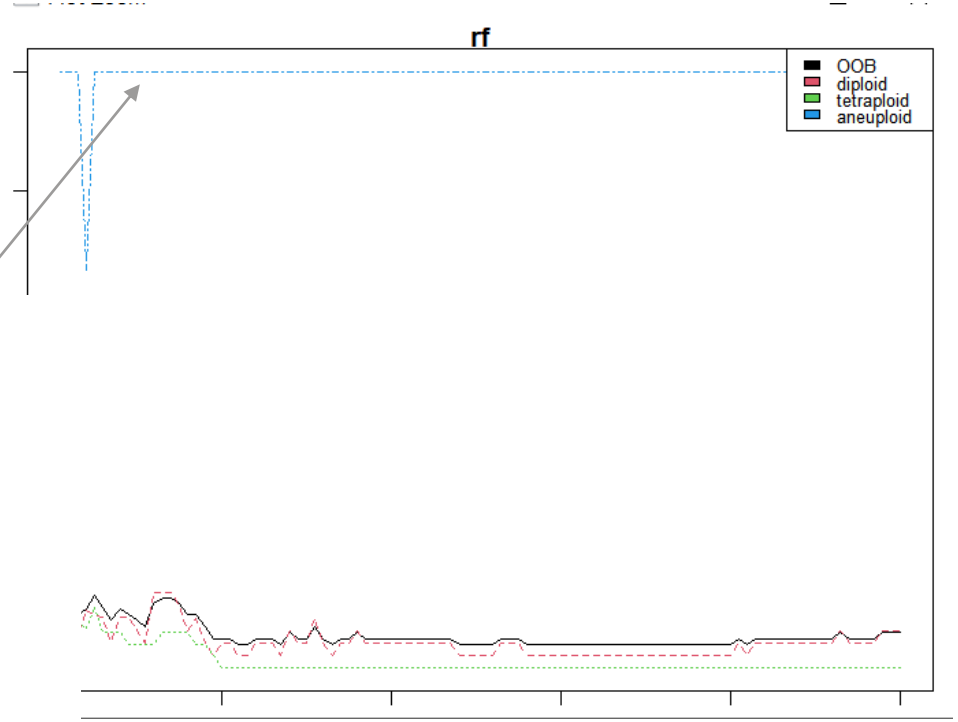
앙상블학습법 -RandomForest

8. 오류율 살펴보기

randomforest 결과를 보기 위헤서는

rf를 입력 후 실행함

```
Call:
 randomForest(formula = ploidy ~ ., data = trainData, ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 5.88%
Confusion matrix:
           diploid tetraploid aneuploid class.error
diploid         45          1         2      0.0625
tetraploid       0         51         0      0.0000
aneuploid        3          0         0      1.0000
> |
```



앙상블학습법 -RandomForest

9. 모형평가하기

```
library(caret)

predict <- predict(rf, testData)

confusionMatrix(predict,
testData$ploidy ,method="everything")
```

```
          Reference
Prediction   diploid tetraploid aneuploid
  diploid         17          0         1
  tetraploid       0         13         1
  aneuploid        0          0         0

Overall Statistics

               Accuracy : 0.9375
                 95% CI : (0.7919, 0.9923)
    No Information Rate : 0.5312
    P-Value [Acc > NIR] : 6.73e-07

                  Kappa : 0.8806

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: diploid Class: tetraploid Class: aneuploid
Sensitivity                  1.0000            1.0000           0.0000
Specificity                  0.9333            0.9474           1.0000
Pos Pred Value               0.9444            0.9286              NaN
Neg Pred Value               1.0000            1.0000           0.9375
Prevalence                   0.5312            0.4062           0.0625
Detection Rate               0.5312            0.4062           0.0000
Detection Prevalence         0.5625            0.4375           0.0000
Balanced Accuracy            0.9667            0.9737           0.5000
> |
```