```
[ ] package_version(R.version)
```

```
[1] '4.2.0'
```

```
[ ] system("gdown --id 1puITTv_Y_2g9rKg7jSaT4XN41HuoFywr")
```

```
[ ] system("gdown --id 1JCy0LLNnrauv8RVi0oZLfjRugEDe_Qc3")
```

```
[ ] hmeqtrain <- read.csv("hmeqN_train.csv", header = T)
    str(hmeqtrain)
```

```
'data.frame':   2000 obs. of  7 variables:
 $ ID     : int  4952 5546 938 277 5204 5762 2354 2896 76 4700 ...
 $ BAD    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ LOAN   : int  26100 35000 9100 5700 28100 44000 14200 16000 3900 24800 ...
 $ MORTDUE: num  73525 391000 17218 58400 61000 ...
 $ VALUE  : num  89870 505000 36721 75000 99000 ...
 $ REASON : chr  "DebtCon" "DebtCon" "DebtCon" "HomeImp" ...
 $ JOB    : chr  "Office" "ProfExe" "Other" "ProfExe" ...
```

```
[ ] hmeqtest <- read.csv("hmeqN_test.csv", header = T)
    str(hmeqtest)
```

```
'data.frame':   378 obs. of  7 variables:
 $ ID     : int  5632 675 3234 3537 3804 926 462 2229 4770 184 ...
 $ BAD    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ LOAN   : int  38700 8000 17300 18600 20000 9000 6900 13700 25000 5000 ...
 $ MORTDUE: num  119847 37871 73000 64248 60336 ...
 $ VALUE  : num  162365 89870 95000 82690 132430 ...
 $ REASON : chr  "HomeImp" "HomeImp" "DebtCon" "DebtCon" ...
 $ JOB    : chr  "ProfExe" "ProfExe" "Other" "Mgr" ...
```

결측값 관리

```
[ ] colSums(is.na(hmeqtrain)); colSums(is.na(hmeqtest));
```

```
ID:      0 BAD:      0 LOAN:      0 MORTDUE:      0 VALUE:      0 REASON:      0 JOB:      0
ID:      0 BAD:      0 LOAN:      0 MORTDUE:      0 VALUE:      0 REASON:      0 JOB:      0
```

```
[ ] install.packages("reshape")
    library(reshape)
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

also installing the dependencies 'Rcpp', 'plyr'
```

변환 하여 넣어주는 작업

```
[ ] hmeqtrain <- as.data.frame(hmeqtrain)
    hmeqtrain$BAD <- as.factor(hmeqtrain$BAD)
    hmeqtrain$LOAN <- as.numeric(hmeqtrain$LOAN)
    hmeqtrain$REASON <- as.factor(hmeqtrain$REASON)
    hmeqtrain$JOB <- as.factor(hmeqtrain$JOB)
    str(hmeqtrain)
```

```
'data.frame':   2000 obs. of  7 variables:
 $ ID     : int  4952 5546 938 277 5204 5762 2354 2896 76 4700 ...
 $ BAD    : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
 $ LOAN   : num  26100 35000 9100 5700 28100 44000 14200 16000 3900 24800 ...
 $ MORTDUE: num  73525 391000 17218 58400 61000 ...
 $ VALUE  : num  89870 505000 36721 75000 99000 ...
 $ REASON : Factor w/ 3 levels "DebtCon","HomeImp",..: 1 1 1 2 1 1 1 1 2 1 ...
 $ JOB    : Factor w/ 7 levels "Mgr","missing",..: 3 5 4 5 1 4 5 4 4 1 ...
```

```
[ ] hmeqtest <- as.data.frame(hmeqtest)
    hmeqtest$BAD <- as.factor(hmeqtest$BAD)
    hmeqtest$LOAN <- as.numeric(hmeqtest$LOAN)
    hmeqtest$REASON <- as.factor(hmeqtest$REASON)
    hmeqtest$JOB <- as.factor(hmeqtest$JOB)
    str(hmeqtest)
```

```
'data.frame':   378 obs. of  7 variables:
 $ ID     : int  5632 675 3234 3537 3804 926 462 2229 4770 184 ...
 $ BAD    : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
```

```
 $ LOAN   : num  38700 8000 17300 18600 20000 9000 6900 13700 25000 5000 ...
 $ MORTDUE: num  119847 37871 73000 64248 60336 ...
 $ VALUE  : num  162365 89870 95000 82690 132430 ...
 $ REASON : Factor w/ 3 levels "DebtCon","HomeImp",..: 2 2 1 1 1 1 2 1 1 1 ...
 $ JOB    : Factor w/ 7 levels "Mgr","missing",..: 5 5 4 1 4 4 6 5 6 4 ...
```

```
[ ] install.packages("ipred")
    library(ipred)
    library(rpart)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘listenv’, ‘parallelly’, ‘future’, ‘globals’, ‘future.apply’, ‘progressr’, ‘numDeriv’, ‘SQUAREM’, ‘lava’,
```

Bagging

```
[ ] bagg.hmeq <- bagging(BAD~.-ID, #Y변수 - BAD | X변수 - ID를 뺀 모두
                         data=hmeqtrain,
                         nbag=1000, #bagging 을 1000번 해라
                         control=rpart.control(minsplit=10), #tree split을 일으킬때 최소한 10개는 들어가야 함
                         coob=T)
    bagg.hmeq
```

```
Bagging classification trees with 1000 bootstrap replications

Call: bagging.data.frame(formula = BAD ~ . - ID, data = hmeqtrain,
    nbag = 1000, control = rpart.control(minsplit = 10), coob = T)

Out-of-bag estimate of misclassification error:  0.1565
```

Y가 multinomial 이면, out of bag estimate를 계산 못한다는 메시지가 뜨는데, 그러면 coob=T 없애고 다시 돌려야 한다

coob=T:bagging의 error rate -> 이걸 빼고 돌리면, out of bag error가 안나온다.

```
install.packages("caret")
library(caret)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘proxy’, ‘iterators’, ‘gower’, ‘hardhat’, ‘timeDate’, ‘e1071’, ‘foreach’, ‘ModelMetrics’, ‘pROC’, ‘recipes


Loading required package: ggplot2

Loading required package: lattice

Warning message in system("timedatectl", intern = TRUE):
“running command 'timedatectl' had status 1”
```

```
[ ] bagg.predict <- predict(bagg.hmeq, hmeqtest, type="class")
    confusionMatrix(bagg.predict, hmeqtest$BAD)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 137   5
         2  52 184

               Accuracy : 0.8492
                 95% CI : (0.8091, 0.8837)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6984

 Mcnemar's Test P-Value : 1.109e-09

            Sensitivity : 0.7249
            Specificity : 0.9735
         Pos Pred Value : 0.9648
         Neg Pred Value : 0.7797
             Prevalence : 0.5000
         Detection Rate : 0.3624
   Detection Prevalence : 0.3757
      Balanced Accuracy : 0.8492

       'Positive' Class : 1
```

correct classification rate = (137+184)/378 = 0.8492

들여쓴 블록

Radom Forest

```
[ ] install.packages("randomForest")
    require(randomForest)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Loading required package: randomForest

randomForest 4.7-1.1

Type rfNews() to see new features/changes/bug fixes.


Attaching package: ‘randomForest’


The following object is masked from ‘package:ggplot2’:

    margin
```

```
[ ] rf.hmeq <- randomForest(BAD~.-ID,
                        data=hmeqtrain,
                        importance=TRUE,
                        ntree=1000,
                        mtry=2)
    rf.hmeq
```

```
Call:
 randomForest(formula = BAD ~ . - ID, data = hmeqtrain, importance = TRUE,      ntree = 1000, mtry = 2)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 2

        OOB estimate of  error rate: 12.15%
Confusion matrix:
    1   2 class.error
1 802 198       0.198
2  45 955       0.045
```

입력변수(컬럼)가 5개(root 5 => mtry를 2로 쓴 것이다)

```
[ ] rf.predict <- predict(rf.hmeq, hmeqtest, type = "class")
    summary(rf.predict)
```

```
1:      178 2:      200
```

```
[ ] confusionMatrix(rf.predict, hmeqtest$BAD)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 159  19
         2  30 170

               Accuracy : 0.8704
                 95% CI : (0.8323, 0.9025)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7407

 Mcnemar's Test P-Value : 0.1531

            Sensitivity : 0.8413
            Specificity : 0.8995
         Pos Pred Value : 0.8933
         Neg Pred Value : 0.8500
             Prevalence : 0.5000
         Detection Rate : 0.4206
   Detection Prevalence : 0.4709
      Balanced Accuracy : 0.8704

       'Positive' Class : 1
```

```
[ ] importance(rf.hmeq)
```

```
[ ] importance(rf.hmeq)
```

A matrix: 5 × 4 of type dbl

	1	2	MeanDecreaseAccuracy	MeanDecreaseGini
LOAN	201.684073	295.22017	312.26298	556.18749
MORTDUE	19.744909	83.24400	93.12678	158.98536
VALUE	-11.782749	87.85284	80.72041	169.84688
REASON	9.476457	49.51785	47.09872	24.20300
JOB	23.855353	97.68437	100.68613	72.23654

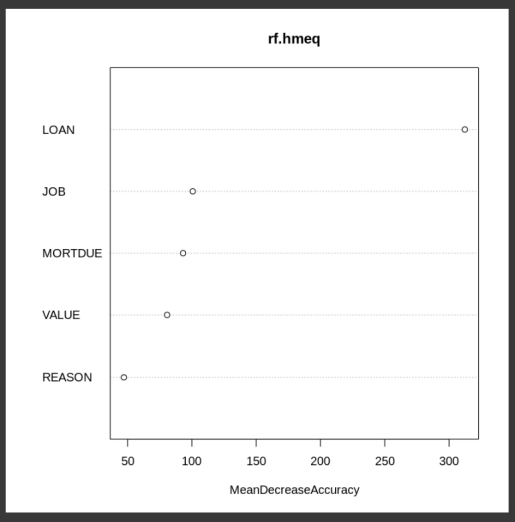
LOAN 은 중요한 변수 (312) REASON 은 중요하지 않은 변수 (100)

```
[ ] importance(rf.hmeq, type=1)
```

A matrix: 5 × 1 of type dbl

	MeanDecreaseAccuracy
LOAN	312.26298
MORTDUE	93.12678
VALUE	80.72041
REASON	47.09872
JOB	100.68613

```
[ ] varImpPlot(rf.hmeq, type=1)
```

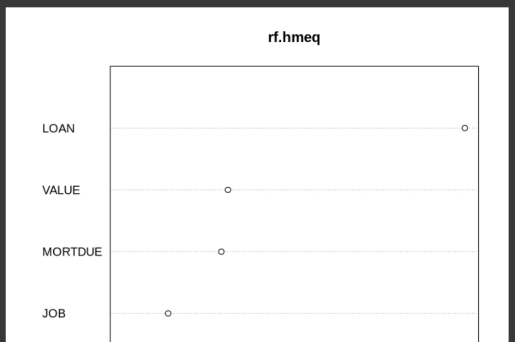


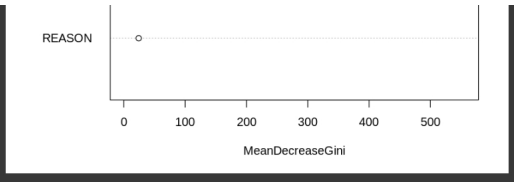

```
[ ] importance(rf.hmeq, type=2)
```

A matrix: 5 × 1 of type dbl

	MeanDecreaseGini
LOAN	556.18749
MORTDUE	158.98536
VALUE	169.84688
REASON	24.20300
JOB	72.23654

```
[ ] varImpPlot(rf.hmeq, type=2)
```





Boosting - gbm package

categorical, numeric 둘다 쓸 수 있는 알고리즘

```
[ ] install.packages("gbm")
    require(gbm)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Loading required package: gbm

Loaded gbm 2.1.8
```

```
[ ] boost.hmeq <- gbm(BAD~.,-ID,
                      data=hmeqtrain,
                      distribution="multinomial", # 안쓰면 오류남
                      n.trees=1000,
                      shrinkage=0.01,
                      interaction.depth = 4)
```

```
Warning message:
“Setting `distribution = "multinomial"` is ill-advised as it is currently broken. It exists only for backwards compatibility. Use at your c
```

```
[ ] boost.predict <- predict.gbm(object=boost.hmeq,
                                 newdata=hmeqtest,
                                 n.trees=1000,
                                 type="response")
```

```
[ ] print(boost.predict)
```

```
, , 1000

                1            2
  [1,] 0.99896035 0.0010396536
  [2,] 0.82277840 0.1772216027
  [3,] 0.99892768 0.0010723184
  [4,] 0.99894289 0.0010571094
  [5,] 0.99903882 0.0009611808
  [6,] 0.23868239 0.7613176052
  [7,] 0.81944621 0.1805537878
  [8,] 0.99879062 0.0012093841
  [9,] 0.99853801 0.0014619949
 [10,] 0.79487632 0.2051236788
 [11,] 0.99904089 0.0009591101
 [12,] 0.99901384 0.0009861571
 [13,] 0.69254755 0.3074524466
 [14,] 0.59890320 0.4010967970
 [15,] 0.99883453 0.0011654656
 [16,] 0.16383326 0.8361667428
 [17,] 0.25178244 0.7482175555
 [18,] 0.99912357 0.0008764276
 [19,] 0.99899695 0.0010030488
 [20,] 0.99855856 0.0014414408
 [21,] 0.99896249 0.0010375073
 [22,] 0.12394018 0.8760598154
 [23,] 0.99889786 0.0011021387
 [24,] 0.99897335 0.0010266484
 [25,] 0.37242247 0.6275775277
 [26,] 0.99895963 0.0010403723
 [27,] 0.99897996 0.0010200359
 [28,] 0.99899318 0.0010068163
 [29,] 0.99899271 0.0010072917
 [30,] 0.99901384 0.0009861571
 [31,] 0.79038312 0.2096168840
 [32,] 0.99899695 0.0010030488
 [33,] 0.63893614 0.3610638612
 [34,] 0.99899271 0.0010072917
 [35,] 0.06155102 0.9384489790
 [36,] 0.99899695 0.0010030488
 [37,] 0.99896944 0.0010305566
 [38,] 0.99840736 0.0015926358
 [39,] 0.99897730 0.0010226965
 [40,] 0.99890508 0.0010949153
 [41,] 0.86944124 0.1305587644
 [42,] 0.80029213 0.1997078680
 [43,] 0.99923734 0.0007626565
 [44,] 0.58405330 0.4159466995
```

```
 [45,] 0.99869587 0.0013041347
 [46,] 0.99895514 0.0010448596
 [47,] 0.99886135 0.0011386519
 [48,] 0.99880065 0.0011993484
 [49,] 0.51194836 0.4880516382
 [50,] 0.99896944 0.0010305566
 [51,] 0.98749078 0.0125092225
 [52,] 0.99917892 0.0008210791
 [53,] 0.99883332 0.0011666766
 [54,] 0.37232021 0.6276797905
 [55,] 0.99890022 0.0010997803
```

boost_predict 데이터의 변수 중 max 인 값의 column을 1,2,3... 순으로 써라. 여기서는 1 또는 2

```
[ ] value <- apply(boost.predict, 1, which.max)
    value
```

```
1·1·1·1·1·2·1·1·1·1·1·1·1·1·1·2·2·1·1·1·1·2·1·1·2·1·1·1·1·1·1·1·1·1·2·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·2·1·1·1·2·1·1·1·1·1·1·2·1·1·2·
1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·1·2·1·1·1·1·2·1·2·1·1·1·1·1·1·1·1·1·2·1·1·1·2·1·1·2·1·1·1·1·1·1·1·2·2·1·1·1·1·1·2·1·1·1·1·2·1·1·
2·1·1·2·1·1·1·1·1·1·1·2·1·1·1·1·1·1·1·1·1·1·1·2·1·1·2·1·1·1·1·2·2·1·2·1·1·1·1·2·1·1·1·1·1·1·1·1·1·1·1·1·1·2·2·2·2·2·2·2·2·2·2·2·2·2·
2·2·2·2·1·2·1·2·2·1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·1·2·2·1·2·2·2·2·2·2·2·2·1·2·1·2·2·2·2·2·2·2·2·2·
2·2·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·1·1·2·2·2·1·2·2·1·2·2·2·2·2·2·2·2·1·2·1·2·2·2·2·2·2·2·2·
2·2·2·2·2·2·2·2·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2
```

```
[ ] result = data.frame(hmeqtest$BAD, value) #실제값
    print(result)
```

```
   hmeqtest.BAD value
1             1     1
2             1     1
3             1     1
4             1     1
5             1     1
6             1     2
7             1     1
8             1     1
9             1     1
10            1     1
11            1     1
12            1     1
13            1     1
14            1     1
15            1     1
16            1     2
17            1     2
18            1     1
19            1     1
20            1     1
21            1     1
22            1     2
23            1     1
24            1     1
25            1     2
26            1     1
27            1     1
28            1     1
29            1     1
30            1     1
31            1     1
32            1     1
33            1     1
34            1     1
35            1     2
36            1     1
37            1     1
38            1     1
39            1     1
40            1     1
41            1     1
42            1     1
43            1     1
44            1     1
45            1     1
46            1     1
47            1     1
48            1     1
49            1     1
50            1     1
51            1     1
52            1     1
53            1     1
54            1     2
55            1     1
56            1     1
57            1     1
```

```
[ ] with(result, table(hmeqtest.BAD, value))
```

```
            value
hmeqtest.BAD   1   2
           1 160  29
           2  15 174
```

```
correct classification rate = (155+172)/378 = 0.8651
```

✓ 0초 오전 12:25에 완료됨