[Statistical Modeling & Machine Learning HW 3]

2017311974 통계학과 진수정

0. Import Dataset

```
> train$X3[train$X3 %in% c(5,6)] = 0
```

: 변수의 값이 해당 변수의 범위에 맞지 않는 경우, 별도의 하나의 범주(unknown)로 통합

1. Imputation

```
> miss_var_summary(train)
```

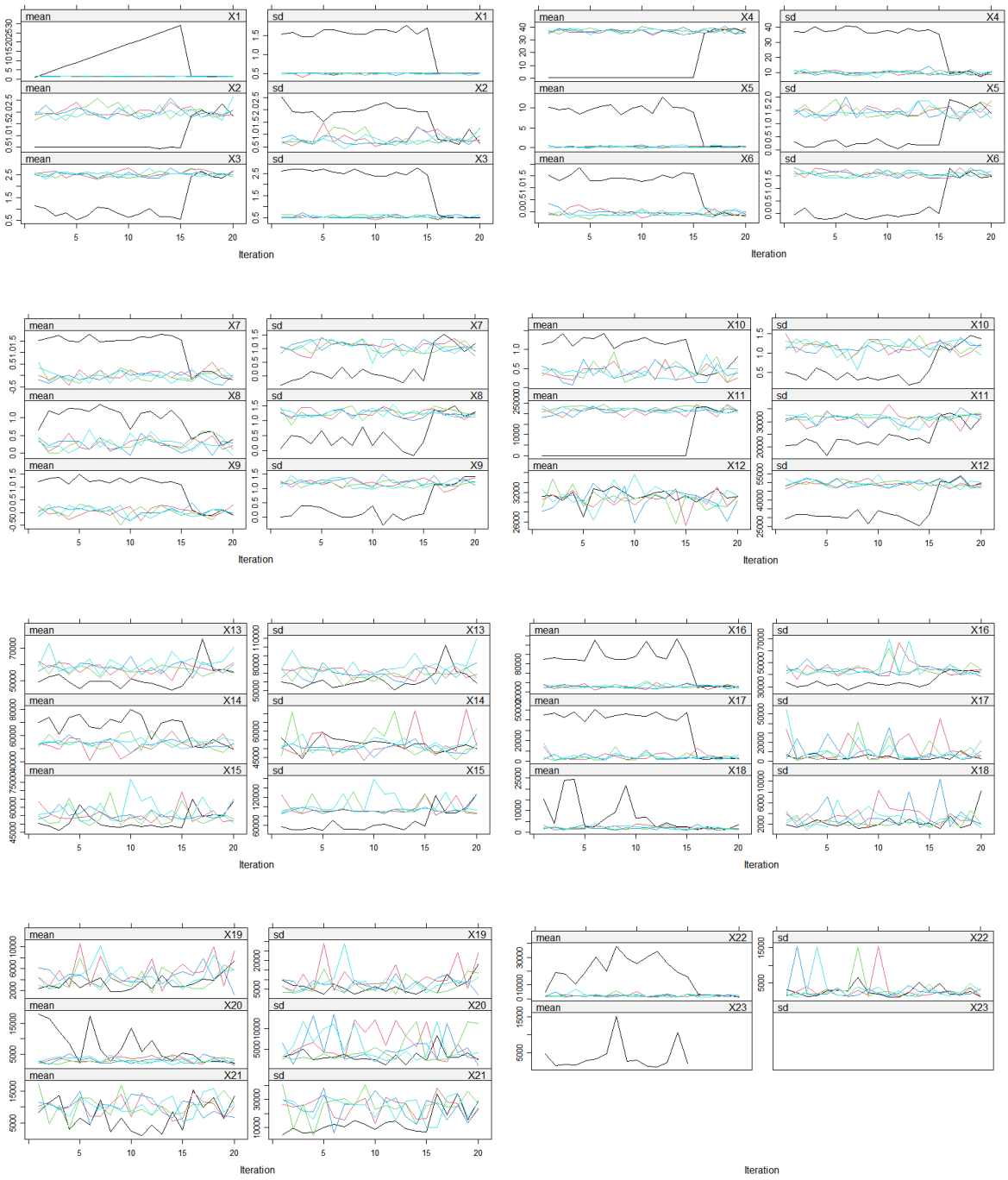
```
# A tibble: 24 x 3
   variable n_miss pct_miss
   <chr>     <int>    <dbl>
 1 X22          27    1.35
 2 X5           23    1.15
 3 X13          23    1.15
 4 X18          23    1.15
 5 X21          23    1.15
 6 X12          20    1
 7 X16          19    0.95
 8 X7           18    0.900
 9 X9           18    0.900
10 X10          18    0.900
# ... with 14 more rows
```

```
> md.pattern(train)
```

```
X1 Y X8 X15 X17 X14 X20 X4 X6 X11 X2 X3 X19 X7 X9 X10 X23 X16 X12 X5 X13 X18 X21 X22
1635  1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   1  1   1   0
21    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   1  1   0   1
19    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   1  0   1   1
1     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   1  0   0   2
19    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   0  1   1   1
19    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   0  1   1   1
1     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  1   0  1   1   2
17    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  0   1  1   1   1
2     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   1  0   0  1   1   2
17    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   1   0  1   1  1   1   1
16    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   0   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   0   1  1   1  1   1   2
1     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   1   0   1  0   1  1   1   2
16    1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   0   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  1  1   1  1  1   1   0   1   1  1   0  1   1   2
17    1 1  1   1   1   1  1  1   1  1  1   1  1  1   0   1   1   1  1   1  1   1   1
15    1 1  1   1   1   1  1  1   1  1  1   1  1  1   0   1   1   1  1   1  1   1   1
2     1 1  1   1   1   1  1  1   1  1  1   1  1  0   1   1   1   1  1   1  0   1   2
13    1 1  1   1   1   1  1  1   1  1  1   1  0  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  1  1   1  0  1   1   1   1   1  1   1  0   1   2
1     1 1  1   1   1   1  1  1   1  1  1   1  0  1   1   1   1   1  1   0  1   1   2
1     1 1  1   1   1   1  1  1   1  1  1   1  0  1   0   1   1   1  1   1  1   1   2
12    1 1  1   1   1   1  1  1   1  1  1   0  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  1  1   0  1  1   1   1   1   1  1   1  1   0   2
1     1 1  1   1   1   1  1  1   1  1  1   0  1  1   1   1   0   1  1   1  1   1   2
15    1 1  1   1   1   1  1  1   1  1  0   1  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  1  0   1  1  0   1   1   1   1  1   1  1   1   2
1     1 1  1   1   1   1  1  1   1  1  0   0  1  1   1   1   1   1  1   1  1   1   2
12    1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   1  0   1   2
1     1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   0  1   1   2
1     1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  0   1  1   1   2
1     1 1  1   1   1   1  1  1   1  0  1   0  1  1   1   1   1   1  1   1  1   1   2
15    1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   1  0   1   2
15    1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  1   1  0  1   1  1  1   1   1   1   0  1   1  1   1   2
14    1 1  1   1   1   1  1  0   1  1  1   1  1  1   1   1   1   1  1   1  1   1   1
1     1 1  1   1   1   1  1  0   1  1  1   1  1  1   1   1   1   1  1   1  1   0   2
1     1 1  1   1   1   1  1  0   1  1  1   1  0  1   1   1   1   0  1   1  1   1   3
 [ reached getOption("max.print") -- omitted 14 rows ]
```

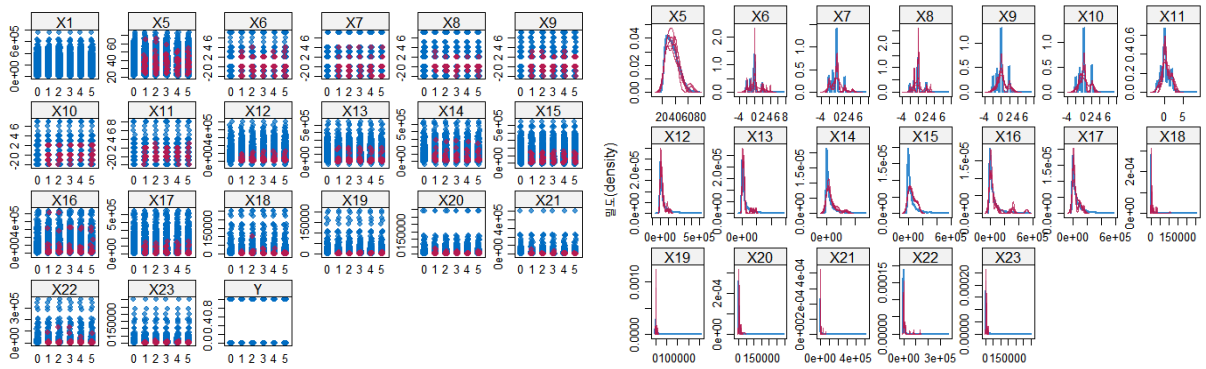
: missing 비율이 높은 변수나, missing 비율이 높은 obs(고객)은 존재하지 않는다. 주로 변수 하나씩에 대해서만 missing이 된 경우가 많았다. 여러 변수에 대해서 missing이 되어 있으므로, MICE를 이용해 imputation을 진행하였다. 이 때 연속형 변수에 대해서는 pmm을, 2개의 범주를 갖는 변수에 대해서는 logreg를, 3개 이상의 범주를 갖는 변수에 대해서는 polyreg 방법을 이용하여 총 5개의 데이터셋을 완성하였다.

```
> set.seed(42)
> imp = mice(train, m = 5, method =
c('','logreg','polyreg','polyreg','pmm',rep('pmm',18),''),maxit = 20)
> plot(imp,paste0('X',seq(1,23)))
```



⇨ Iteration을 20으로 늘린 결과, MICE의 수렴성을 확인할 수 있었다.

```
> stripplot(imp,pch = 20,cex = 1.2)
> densityplot(imp,scales = list(relation = 'free'))
```

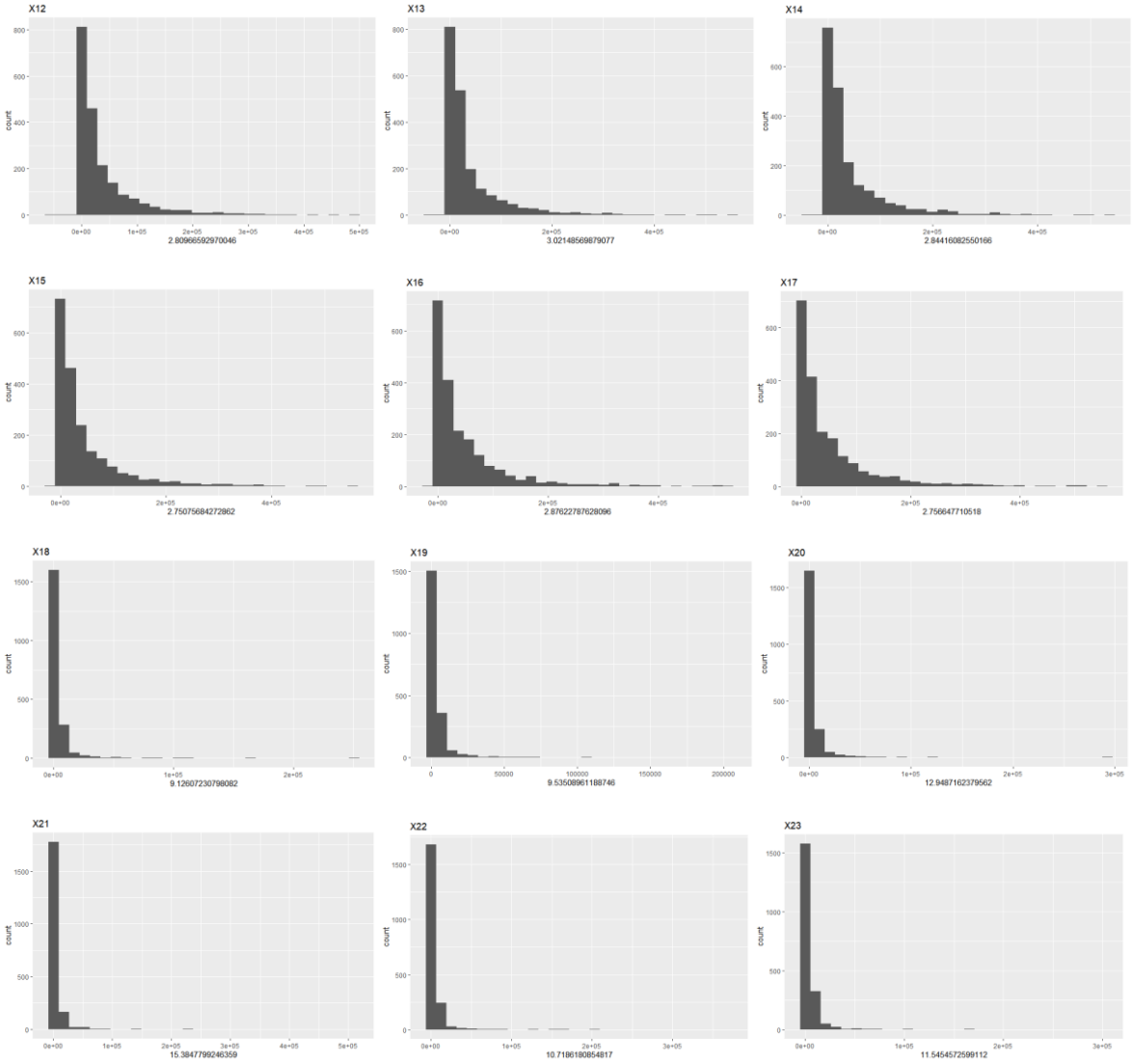


⇨ 원래의 데이터셋과 비슷한 값으로 대체가 되었음을 확인할 수 있다.

2. Feature Engineering & Data Transformation

: 월별로 (bill amount – payment amount) / (credit amount)를 계산한 변수를 추가하였다(X24 – X29). X24 – X29 값이 클수록 defaulter일 가능성이 커질 것이다.

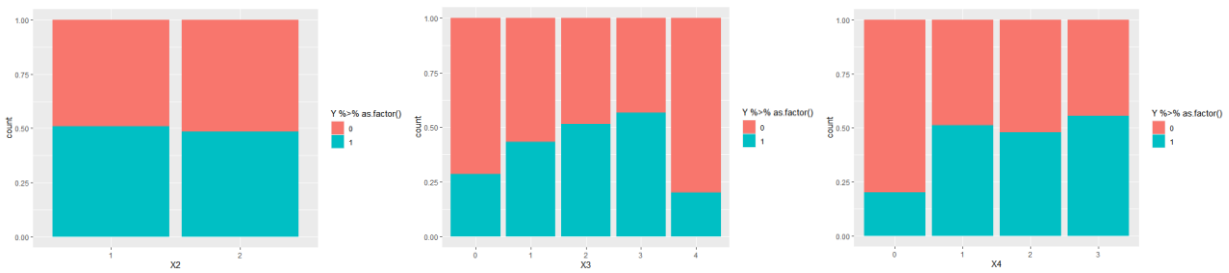
```
> for (m in 1:M) {
+   imp.dat[[m]] = complete(imp,m)
+   imp.dat[[m]] = imp.dat[[m]] %>% mutate(X24 = (X12 - X18) / X1,
+                                          X25 = (X13 - X19) / X1,
+                                          X26 = (X14 - X20) / X1,
+                                          X27 = (X15 - X21) / X1,
+                                          X28 = (X16 - X22) / X1,
+                                          X29 = (X17 - X23) / X1)
+ }
```



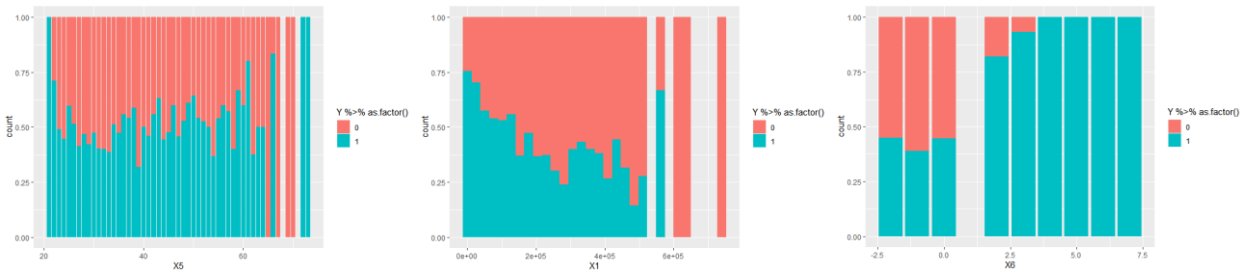
⇨ Skewness가 2 이상인 변수들에 대해서 Yeo-Johnson Transformation 시켜주었다.

3. EDA & Modeling

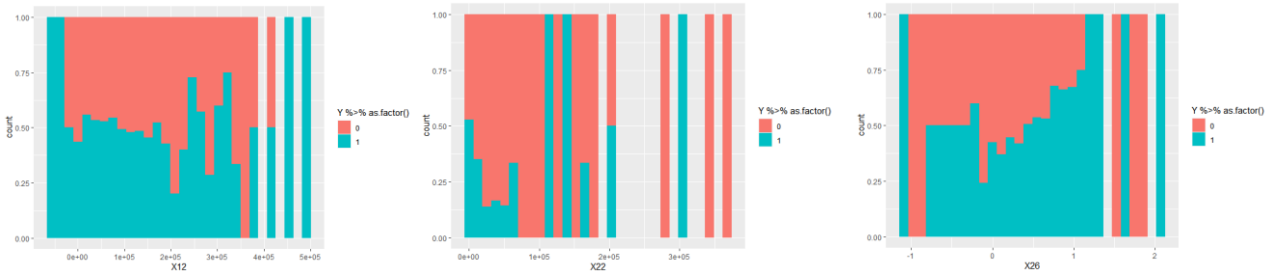
(1) Visualization



: 성별이나 결혼 상태에 따른 defaulter의 분포에는 뚜렷한 패턴이 없었으며, 교육수준이 높아짐에 따라 defaulter의 비율은 감소하였다.



: 나이에 따른 defaulter의 비율은 미세하게 감소하다가 증가하였다. 주어진 credit의 값(X1)이 클수록 defaulter의 비율은 감소하였으며, (X6 – X11) payment가 delay 될수록 defaulter의 비율이 높았다.



: bill에 따른 defaulter의 분포에서는 bill의 값이 작거나 클 때 defaulter의 비율이 좀 더 높았다. payment에 따른 defaulter의 분포에서는 뚜렷한 패턴을 찾을 수 없었고, 파생변수 (X24 – X29)의 값이 클수록 defaulter의 비율은 높았다.

(2) Modeling

: Lasso 패널티를 적용한 Logistic Regression, Random Forest, Xgboost 모델의 성능을 비교한 결과 Random Forest의 성능이 가장 좋았다.

- Random Forest: 변수를 선택했을 때의 성능이 더 좋지 않았기 때문에 최종 모델은 변수 선택을 하지 않은 모델로 선택하였다.
- 우선 imputed된 5개의 데이터셋 각각에 대해 파라미터 튜닝(3 fold cv & Grid Search)을 진행한 후 prediction값(0/1)을 구한다. Prediction값 5개에 대해 majority vote rule을 적용하여 최종 prediction값을 구한 후에 test error rate을 계산하였다.

– Grid Search : mtry, ntree 파라미터 튜닝

```
> param_df = data.frame(
+   mtry = rep(c(2,3,4,5),4),
+   ntree = rep(c(200,400,600,800),each = 4),
+   accuracy = rep(NA,16)
+ )
> set.seed(1234)
> # 3-fold Grid Search
> cv = createFolds(transformed_imp[[1]]$Y, k = 3)
>
> pb = progress_bar$new(total = nrow(param_df))
>
> for (i in 1:nrow(param_df)) {
+   temp_acc = NULL
+
+   for (j in 1:3) {
+     valid_idx = cv[[j]]
+     cv_test = transformed_imp[[1]][valid_idx,]
+     cv_train = transformed_imp[[1]][-valid_idx,]
+
+     set.seed(1234)
+     temp_fit = randomForest(Y ~., cv_train, mtry = param_df[i,'mtry'],ntree =
param_df[i,'ntree'])
+     temp_pred = predict(temp_fit,newdata = cv_test)
+     temp_class = ifelse(temp_pred >= 0.5,1,0)
+     temp_acc[j] = mean(temp_class == cv_test$Y)
+   }
```

```
+   param_df[i,'accuracy'] = mean(temp_acc)
+   pb$tick()
+   Sys.sleep(0.01)
+ }
```

- Optimal한 파라미터로 모델링

```
> optim_param = lapply(grid_result,function(x) x[which.max(x$accuracy),c('mtry','ntree')])
> optim_rf = function(train,m) {
+   fit = randomForest(Y ~., train[[m]], mtry = optim_param[[m]]$mtry,ntree =
optim_param[[m]]$ntree)
+   phat = predict(fit,newdata = transformed_test)
+   yhat = ifelse(phat >= 0.5,1,0)
+
+   return (yhat)
+ }
> yhat.all = vector(mode = 'list',length = M)
```

- Pooling (imputed dataset이 5개이므로, prediction 결과 5개의 mean이 0.5 이상(=1이 0보다 더 많음)이면) 최종적으로 1로 예측하고, 0.5 미만(=0이 1보다 많음)이면 최종적으로 0으로 예측하였다.

```
> for (m in 1:M) yhat.all[[m]] = optim_rf(transformed_imp,m)
> yhat = matrix(unlist(yhat.all),nrow(transformed_test),M)
> yhat = rowSums(yhat) / 5
> yhat = ifelse(yhat > 0.5,1,0)
> mean(yhat == transformed_test$Y)  # 0.704
[1] 0.704
```

- Test misclassification rate = 0.704