빅콘데스트 챌린지리그 : 대출 연체,상환 예측하기

Code

장은아 (dmsdk2121@gmail.com (mailto:dmsdk2121@gmail.com))

2017.10.13

Introduction.

2017 빅콘데스트 챌린지리그 : 대출 연체,상환 예측하기

> 보험, 통신, 신용평가사 (개인정보 비식별)결합데이터를 활용한 대출상환 예측 알고리즘 개발

실제 기업 (한화생명/SKT/SCI) Data 기반의 데이터

- 한화생명 : 직업 / 소득 / 배우자직업 / 신용등급 / 한화생명 신용대출정보 / 보험 정보 등 36개
- SKT : 통화시간 / 멤버쉽 등급 / 가입년월 / 정지 일수 / 연체금액 등 15개
- SCI : 대출 건수 / 대출 총 금액 / 대출계좌 유지 기간 / 보증 건수 / 보증 금액 등 14개

전체변수 총 69 개 , 관측치 102,252개

TARGET의 반응비율(연체한 비율)이 4%대로 매우 낮음

평가지표 : 예측값의 F1 score

목차

3) 랜덤포레스트 모델을 사용해 중요한 변수 찾기

5. 랜덤포레스트 모델 사용하기

1) 모델 적합, 튜닝하기
2) 튜닝한 모델로 예측하기
3) 예측한 결과 분석하기

6. SVM 모델 사용하기

1) 모델 적용 전에 데이터 스케일링, 센터링 작업
2) 모델 적합, 튜닝하기
3) 튜닝한 모델로 예측하기
4) 예측한 결과 분석하기

7. 최종예측

Hide

```
# 패키지 로드
library(dplyr)
```

```


다음의 패키지를 부착합니다: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

Hide

```
library(ggplot2)
library(ISLR)
library(MASS)
```

```


다음의 패키지를 부착합니다: 'MASS'

The following object is masked from 'package:dplyr':

    select
```

Hide

```
library(glmnet)
```

```
필요한 패키지를 로딩중입니다: Matrix
필요한 패키지를 로딩중입니다: foreach
foreach: simple, scalable parallel programming from Revolution Analytics
Use Revolution R for scalability, fault tolerance and more.
http://www.revolutionanalytics.com
Loaded glmnet 2.0-13
```

Hide

```
library(randomForest)
```

```
randomForest 4.6-12
Type rfNews() to see new features/changes/bug fixes.

다음의 패키지를 부착합니다: 'randomForest'

The following object is masked from 'package:dplyr':

    combine

The following object is masked from 'package:ggplot2':

    margin
```

Hide

```
library(gbm)
```

```
필요한 패키지를 로딩중입니다: survival
필요한 패키지를 로딩중입니다: lattice
필요한 패키지를 로딩중입니다: splines
Loaded gbm 2.1.3
```

Hide

```
library(rpart)
library(boot)
```

```


다음의 패키지를 부착합니다: 'boot'

The following object is masked from 'package:lattice':

    melanoma

The following object is masked from 'package:survival':

    aml
```

Hide

```
library(rpart)
library(caret)
```

```
다음의 패키지를 부착합니다: 'caret'

The following object is masked from 'package:survival':

    cluster
```

1. 데이터 로드하기

Hide

```
setwd("/Users/jang-eun-a/challenge_data")
options("scipen" = 100)
train.data <- read.csv("Data_set.csv", header = TRUE)
test.data <- read.csv("Test_set.csv", header = TRUE)
full.data <- rbind(train.data, test.data)
dim(full.data)
```

```
[1] 102252     69
```

1) 데이터 구조 살펴보기

Hide

```
# glimpse(full.data)
# TARGET , CUST_ID , LAST_CHLD_AGE factor형으로 바꾸기
full.data$TARGET <- as.factor(full.data$TARGET)
full.data$CUST_ID <- as.factor(full.data$CUST_ID)
full.data$LAST_CHLD_AGE <- as.factor(full.data$LAST_CHLD_AGE)
glimpse(full.data)
```

```
Observations: 102,252
Variables: 69
$ CUST_ID           <fctr> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...
$ TARGET            <fctr> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, ...
$ BNK_LNIF_CNT      <int> 1, 1, 0, 0, 4, 1, 0, 1, 2, 0, 0, 0, 1, 2, 3...
$ CPT_LNIF_CNT      <int> 0, 0, 1, 2, 0, 0, 1, 0, 0, 1, 3, 1, 0, 0, 1...
$ SPART_LNIF_CNT    <int> 0, 0, 3, 4, 0, 1, 2, 0, 0, 1, 5, 2, 0, 0, 1...
$ ECT_LNIF_CNT      <int> 0, 0, 2, 2, 0, 1, 1, 0, 0, 0, 2, 1, 0, 0, 0...
$ TOT_LNIF_AMT      <int> 9001, 24001, 15001, 6001, 21001, 141001, 12...
$ TOT_CLIF_AMT      <int> 9001, 0, 9001, 3001, 15001, 27001, 3001, 30...
$ BNK_LNIF_AMT      <int> 9001, 24001, 0, 0, 21001, 111001, 0, 3001, ...
$ CPT_LNIF_AMT      <int> 0, 0, 3001, 3001, 0, 0, 9001, 0, 0, 9001, 1...
$ CRDT_OCCR_MDIF    <int> 1, 0, 1, 1, 1, 1, 121, 1, 37, 1, 1, 49, 13,...
$ SPTCT_OCCR_MDIF   <int> 0, 0, 25, 25, 0, 1, 121, 0, 0, 1, 25, 49, 0...
$ CRDT_CARD_CNT     <int> 2, 2, 4, 4, 1, 4, 2, 2, 5, 3, 4, 6, 0, 4, 3...
$ CTCD_OCCR_MDIF    <int> 13, 121, 121, 61, 97, 121, 121, 121, 121, 2...
$ CB_GUIF_CNT       <int> 3, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0...
$ CB_GUIF_AMT       <int> 420001, 0, 0, 0, 0, 0, 6001, 0, 0, 0, 0, 0,...
$ OCCP_NAME_G       <fctr> 공무원, 자영업, 주부, 학생, 공무원, 3차산업 종사자, 주부, 기업/단체...
$ CUST_JOB_INCM     <int> 5400, 5500, 0, 0, 4800, 4400, 0, 0, 0, 4700...
$ HSHD_INFR_INCM    <int> 7700, 8100, 4900, 10100, 4800, 7700, 7700, ...
$ ACTL_FMLY_NUM     <int> 4, 4, 4, 2, 4, 2, 5, 3, 4, 4, 3, 2, 3, 2, 3...
$ CUST_FMLY_NUM     <int> 1, 2, 1, 1, 1, 2, 3, 1, 1, 1, 1, 1, 2, 1, 1...
$ LAST_CHLD_AGE     <fctr> 24, 29, 34, 0, 14, 0, 19, 24, 9, 14, 24, 0...
$ MATE_OCCP_NAME_G  <fctr> 주부, 주부, 2차산업 종사자, NULL, 주부, 단순 사무직, 2차산업 종...
$ MATE_JOB_INCM     <int> 0, 0, 0, 0, 0, 3300, 4400, 5000, 5400, 7500...
$ CRDT_LOAN_CNT     <int> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0...
$ MIN_CNTT_DATE     <int> 0, 0, 0, 0, 0, 0, 200106, 0, 0, 0, 0, 0, 0,...
$ TOT_CRLN_AMT      <int> 0, 0, 0, 0, 0, 0, 4000000, 0, 0, 0, 0, 0, 0...
$ TOT_REPY_AMT      <int> 0, 0, 0, 0, 0, 0, 4000000, 0, 0, 0, 0, 0, 0...
$ CRLN_OVDU_RATE    <int> 0, 0, 0, 0, 0, 0, 81, 0, 0, 0, 0, 0, 0, 0, ...
$ CRLN_30OVDU_RATE  <int> 0, 0, 0, 0, 0, 0, 14, 0, 0, 0, 0, 0, 0, 0, ...
$ LT1Y_CLOD_RATE    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ STRT_CRDT_GRAD    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ LTST_CRDT_GRAD    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ PREM_OVDU_RATE    <int> 12, 13, 2, 4, 0, 25, 71, 4, 3, 15, 1, 0, 40...
$ LT1Y_PEOD_RATE    <fctr> 20미만, 0, 0, 0, 0, 0, 10미만, 0, 10미만, 10미만, ...
$ AVG_STLN_RATE     <int> 0, 0, 0, 0, 95, 0, 94, 0, 0, 99, 0, 0, 98, ...
$ STLN_REMN_AMT     <int> 0, 0, 0, 0, 2000000, 0, 3000000, 0, 0, 0, 0...
$ LT1Y_STLN_AMT     <int> 0, 0, 0, 0, 0, 0, 2000000, 0, 0, 3000000, 0...
$ LT1Y_SLOD_RATE    <int> 0, 0, 0, 0, 0, 0, 10, 0, 0, 30, 0, 0, 0, 0,...
$ GDINS_MON_PREM    <int> 190000, 0, 0, 0, 0, 100000, 0, 0, 0, 300000...
$ SVINS_MON_PREM    <int> 0, 0, 0, 0, 0, 0, 200000, 0, 60000, 0, 0, 0...
$ FMLY_GDINS_MNPREM <int> 190000, 110000, 0, 0, 0, 190000, 300000, 0,...
$ FMLY_SVINS_MNPREM <int> 0, 0, 0, 0, 0, 0, 200000, 0, 60000, 0, 0, 0...
$ MAX_MON_PREM      <int> 190000, 0, 100000, 0, 300000, 190000, 20000...
$ TOT_PREM          <int> 20000000, 7000000, 11000000, 4000000, 40000...
$ FMLY_TOT_PREM     <int> 20000000, 36000000, 11000000, 4000000, 4000...
$ CNTT_LAMT_CNT     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ LT1Y_CTLT_CNT     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ AUTR_FAIL_MCNT    <int> 10, 0, 0, 0, 0, 0, 1, 0, 0, 21, 2, 0, 31, 0...
$ FYCM_PAID_AMT     <int> 0, 300000, 0, 0, 500000, 300000, 800000, 0,...
$ FMLY_CLAM_CNT     <int> 0, 2, 0, 0, 2, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0...
$ FMLY_PLPY_CNT     <int> 0, 5, 2, 1, 1, 1, 0, 1, 0, 2, 0, 0, 0, 1, 0...
$ AGE               <fctr> 50, 50, 60, 35, 45, 45, 40, 60, 40, 40, 55...
$ SEX               <fctr> 1, 1, 2, 1, 1, 1, 2, 1, 2, 1, 2, 1, 2, 1, ...
$ AVG_CALL_TIME     <int> 450, 81, 139, 1118, 396, 268, 744, 309, 314...
```

```
$ AVG_CALL_FREQ     <int> 493, 22, 17, 0, 354, 179, 535, 221, 179, 0,...
$ TEL_MBSP_GRAD     <fctr> , , , , W, Q, W, R, R, , , R, W, W, Q, Q, ...
$ ARPU              <int> 30000, 30000, 30000, 30000, 50000, 60000, 5...
$ MON_TLFE_AMT      <int> 80000, 40000, 40000, 80000, 80000, 80000, 1...
$ CBPT_MBSP_YN      <fctr> N, N, Y, N, Y, N, N, Y, Y, N, N, N, Y, N, ...
$ MOBL_FATY_PRC     <int> 800000, 500000, 500000, 900000, 800000, 400...
$ TEL_CNTT_QTR      <int> 20111, 20143, 20103, 20144, 20131, 20154, 2...
$ NUM_DAY_SUSP      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ CRMM_OVDU_AMT     <int> 0, 0, 0, 540000, 130000, 0, 120000, 0, 0, 0...
$ TLFE_UNPD_CNT     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ LT1Y_MXOD_AMT     <int> 0, 0, 0, 630000, 90000, 0, 290000, 0, 0, 0,...
$ PAYM_METD         <fctr> O, O, O, G, G, O, G, O, O, O, R, O, O, K, ...
$ LINE_STUS         <fctr> U, U, U, S, U, U, U, U, U, U, U, U, U, U, ...
$ MOBL_PRIN         <int> 580000, 90000, 120000, 320000, 410000, 1700...
```

Hide

```
summary(full.data)
```

```
   CUST_ID         TARGET        BNK_LNIF_CNT     CPT_LNIF_CNT
1       :     1   0   :95946   Min.   :0.0000   Min.   :0.0000
2       :     1   1   : 4287   1st Qu.:0.0000   1st Qu.:0.0000
3       :     1   NA's: 2019   Median :1.0000   Median :0.0000
4       :     1                Mean   :0.8401   Mean   :0.5013
5       :     1                3rd Qu.:1.0000   3rd Qu.:1.0000
6       :     1                Max.   :5.0000   Max.   :5.0000
(Other):102246
SPART_LNIF_CNT    ECT_LNIF_CNT    TOT_LNIF_AMT     TOT_CLIF_AMT
Min.   :0.0000   Min.   :0.000   Min.   :     1   Min.   :     0
1st Qu.:0.0000   1st Qu.:0.000   1st Qu.: 12001   1st Qu.:     0
Median :1.0000   Median :0.000   Median : 39001   Median :  9001
Mean   :0.9533   Mean   :0.478   Mean   : 84028   Mean   : 33233
3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:105001   3rd Qu.: 27001
Max.   :7.0000   Max.   :6.000   Max.   :994001   Max.   :994001

 BNK_LNIF_AMT     CPT_LNIF_AMT    CRDT_OCCR_MDIF   SPTCT_OCCR_MDIF
Min.   :     0   Min.   :     0   Min.   :  0.00   Min.   :  0.00
1st Qu.:     0   1st Qu.:     0   1st Qu.:  0.00   1st Qu.:  0.00
Median :  9001   Median :     0   Median :  1.00   Median :  0.00
Mean   : 51609   Mean   :  4189   Mean   : 18.08   Mean   : 13.99
3rd Qu.: 63001   3rd Qu.:  3001   3rd Qu.: 25.00   3rd Qu.: 13.00
Max.   :944001   Max.   :301001   Max.   :121.00   Max.   :121.00

CRDT_CARD_CNT    CTCD_OCCR_MDIF    CB_GUIF_CNT        CB_GUIF_AMT
Min.   : 0.000   Min.   :  0.00   Min.   : 0.00000   Min.   :     0
1st Qu.: 2.000   1st Qu.: 61.00   1st Qu.: 0.00000   1st Qu.:     0
Median : 3.000   Median :121.00   Median : 0.00000   Median :     0
Mean   : 3.092   Mean   : 91.01   Mean   : 0.09626   Mean   :  9168
3rd Qu.: 4.000   3rd Qu.:121.00   3rd Qu.: 0.00000   3rd Qu.:     0
Max.   :11.000   Max.   :121.00   Max.   :10.00000   Max.   :980001

        OCCP_NAME_G    CUST_JOB_INCM   HSHD_INFR_INCM  ACTL_FMLY_NUM
주부           :28137   Min.   :    0   Min.   :    0   Min.   :1.000
사무직         :16924   1st Qu.:    0   1st Qu.: 4800   1st Qu.:2.000
2차산업 종사자: 9816   Median : 3600   Median : 6600   Median :3.000
자영업         : 9663   Mean   : 2788   Mean   : 6922   Mean   :2.758
3차산업 종사자: 8421   3rd Qu.: 4700   3rd Qu.: 9200   3rd Qu.:4.000
공무원         : 5184   Max.   :10000   Max.   :20000   Max.   :8.000
(Other)        :24107
CUST_FMLY_NUM   LAST_CHLD_AGE         MATE_OCCP_NAME_G MATE_JOB_INCM
Min.   :1.000   0      :51146   NULL           :46620   Min.   :    0
1st Qu.:1.000   24     :10992   주부           :11941   1st Qu.:    0
Median :1.000   19     : 9834   사무직         :10258   Median :    0
Mean   :1.392   29     : 8097   2차산업 종사자: 8262   Mean   : 1718
3rd Qu.:2.000   14     : 5910   자영업         : 6523   3rd Qu.: 4400
Max.   :5.000   34     : 5702   3차산업 종사자: 4084   Max.   :10000
                (Other):10571   (Other)        :14564
CRDT_LOAN_CNT     MIN_CNTT_DATE     TOT_CRLN_AMT
Min.   : 0.0000   Min.   :     0   Min.   :        0
1st Qu.: 0.0000   1st Qu.:     0   1st Qu.:        0
Median : 0.0000   Median :     0   Median :        0
Mean   : 0.1628   Mean   : 19776   Mean   :  1184114
3rd Qu.: 0.0000   3rd Qu.:     0   3rd Qu.:        0
Max.   :11.0000   Max.   :201604   Max.   :101000000

 TOT_REPY_AMT       CRLN_OVDU_RATE    CRLN_30OVDU_RATE
```

```
Min.   :        0   Min.   :  0.000   Min.   :  0.0000
1st Qu.:        0   1st Qu.:  0.000   1st Qu.:  0.0000
Median :        0   Median :  0.000   Median :  0.0000
Mean   :   873509   Mean   :  2.527   Mean   :  0.2256
3rd Qu.:        0   3rd Qu.:  0.000   3rd Qu.:  0.0000
Max.   :101000000   Max.   :100.000   Max.   :100.0000

LT1Y_CLOD_RATE     STRT_CRDT_GRAD   LTST_CRDT_GRAD    PREM_OVDU_RATE
Min.   :  0.0000   Min.   :0.0000   Min.   : 0.0000   Min.   :  0.00
1st Qu.:  0.0000   1st Qu.:0.0000   1st Qu.: 0.0000   1st Qu.:  0.00
Median :  0.0000   Median :0.0000   Median : 0.0000   Median :  3.00
Mean   :  0.1901   Mean   :0.1493   Mean   : 0.2937   Mean   :  8.01
3rd Qu.:  0.0000   3rd Qu.:0.0000   3rd Qu.: 0.0000   3rd Qu.: 10.00
Max.   :100.0000   Max.   :7.0000   Max.   :10.0000   Max.   :100.00

LT1Y_PEOD_RATE  AVG_STLN_RATE    STLN_REMN_AMT       LT1Y_STLN_AMT
0      :74734   Min.   :  0.00   Min.   :        0   Min.   :        0
10미만 :12744   1st Qu.:  0.00   1st Qu.:        0   1st Qu.:        0
20미만 : 7334   Median :  0.00   Median :        0   Median :        0
30미만 : 3080   Mean   : 16.99   Mean   :  1602091   Mean   :   930065
40미만 : 1792   3rd Qu.:  0.00   3rd Qu.:        0   3rd Qu.:        0
50미만 :  866   Max.   :100.00   Max.   :101000000   Max.   :101000000
(Other): 1702
LT1Y_SLOD_RATE    GDINS_MON_PREM    SVINS_MON_PREM    FMLY_GDINS_MNPREM
Min.   :  0.000   Min.   :      0   Min.   :      0   Min.   :      0
1st Qu.:  0.000   1st Qu.:      0   1st Qu.:      0   1st Qu.:      0
Median :  0.000   Median :  50000   Median :      0   Median :  80000
Mean   :  2.586   Mean   : 127566   Mean   :  91989   Mean   : 183984
3rd Qu.:  0.000   3rd Qu.: 160000   3rd Qu.:      0   3rd Qu.: 230000
Max.   :100.000   Max.   :4000000   Max.   :4000000   Max.   :4000000

FMLY_SVINS_MNPREM   MAX_MON_PREM         TOT_PREM
Min.   :       0   Min.   :       0   Min.   :         0
1st Qu.:       0   1st Qu.:   70000   1st Qu.:   5000000
Median :       0   Median :  190000   Median :  11000000
Mean   :  136584   Mean   :  373290   Mean   :  20693277
3rd Qu.:  100000   3rd Qu.:  390000   3rd Qu.:  23000000
Max.   :10000000   Max.   :10000000   Max.   :1000000000

FMLY_TOT_PREM        CNTT_LAMT_CNT    LT1Y_CTLT_CNT     AUTR_FAIL_MCNT
Min.   :         0   Min.   :0.0000   Min.   :0.00000   Min.   : 0.000
1st Qu.:   6000000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 0.000
Median :  15000000   Median :0.0000   Median :0.00000   Median : 0.000
Mean   :  30766772   Mean   :0.1154   Mean   :0.02282   Mean   : 2.718
3rd Qu.:  34000000   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.: 2.000
Max.   :1000000000   Max.   :7.0000   Max.   :6.00000   Max.   :61.000

FYCM_PAID_AMT       FMLY_CLAM_CNT     FMLY_PLPY_CNT         AGE
Min.   :        0   Min.   :  0.000   Min.   : 0.000   45     :19426
1st Qu.:        0   1st Qu.:  0.000   1st Qu.: 0.000   50     :18116
Median :   300000   Median :  1.000   Median : 1.000   55     :15643
Mean   :  4039497   Mean   :  4.218   Mean   : 1.235   40     :15116
3rd Qu.:  1900000   3rd Qu.:  5.000   3rd Qu.: 2.000   35     :12217
Max.   :300100000   Max.   :171.000   Max.   :22.000   60     : 9381
                                                       (Other):12353
SEX       AVG_CALL_TIME     AVG_CALL_FREQ  TEL_MBSP_GRAD
*:  434   Min.   :    0.0   Min.   :   0    :46989
1:46218   1st Qu.:   69.0   1st Qu.:  41   E: 4103
```

```
 2:55600   Median :  171.0   Median : 135   Q:13913
           Mean   :  281.6   Mean   : 169   R:20235
           3rd Qu.:  361.0   3rd Qu.: 242   W:17012
           Max.   :10000.0   Max.   :1900

     ARPU          MON_TLFE_AMT    CBPT_MBSP_YN MOBL_FATY_PRC
Min.   :    -1   Min.   :     0   N:58896      Min.   :      0
1st Qu.: 30000   1st Qu.: 40000   Y:43356      1st Qu.:      0
Median : 40000   Median : 60000                Median : 700000
Mean   : 43802   Mean   : 72439                Mean   : 530267
3rd Qu.: 60000   3rd Qu.: 80000                3rd Qu.: 900000
Max.   :500000   Max.   :950000                Max.   :1200000

 TEL_CNTT_QTR    NUM_DAY_SUSP     CRMM_OVDU_AMT     TLFE_UNPD_CNT
Min.   :19934   Min.   :   0.00   Min.   :      0   Min.   :0.000000
1st Qu.:20113   1st Qu.:   0.00   1st Qu.:      0   1st Qu.:0.000000
Median :20132   Median :   0.00   Median :      0   Median :0.000000
Mean   :20122   Mean   :  17.11   Mean   :  11028   Mean   :0.003697
3rd Qu.:20143   3rd Qu.:   0.00   3rd Qu.:      0   3rd Qu.:0.000000
Max.   :20162   Max.   :2700.00   Max.   :1200000   Max.   :2.000000

LT1Y_MXOD_AMT     PAYM_METD LINE_STUS    MOBL_PRIN
Min.   :      0    : 2878   S:  1949   Min.   :      0
1st Qu.:      0   G: 4625   U:100303   1st Qu.:      0
Median :      0   K:33027              Median : 100000
Mean   :  19157   O:58076              Mean   : 190114
3rd Qu.:      0   R: 3646              3rd Qu.: 320000
Max.   :1600000                        Max.   :1100000
```

2. 데이터 전처리 (NA값 채우기)

1) OCCP_NAME_G 직업 살펴보기

Hide

```
summary(full.data$OCCP_NAME_G)
```

```
             *  1차산업 종사자  2차산업 종사자  3차산업 종사자
          1204            1205            9816            8421
 고소득 전문직          공무원  기업/단체 임원            기타
          1247            5184            1064            1710
   단순 노무직     단순 사무직          사무직 예체능계 종사자
           840            4186           16924             954
       운전직          자영업          전문직            주부
          2172            9663            5142           28137
          학생            NULL
          3912             471
```

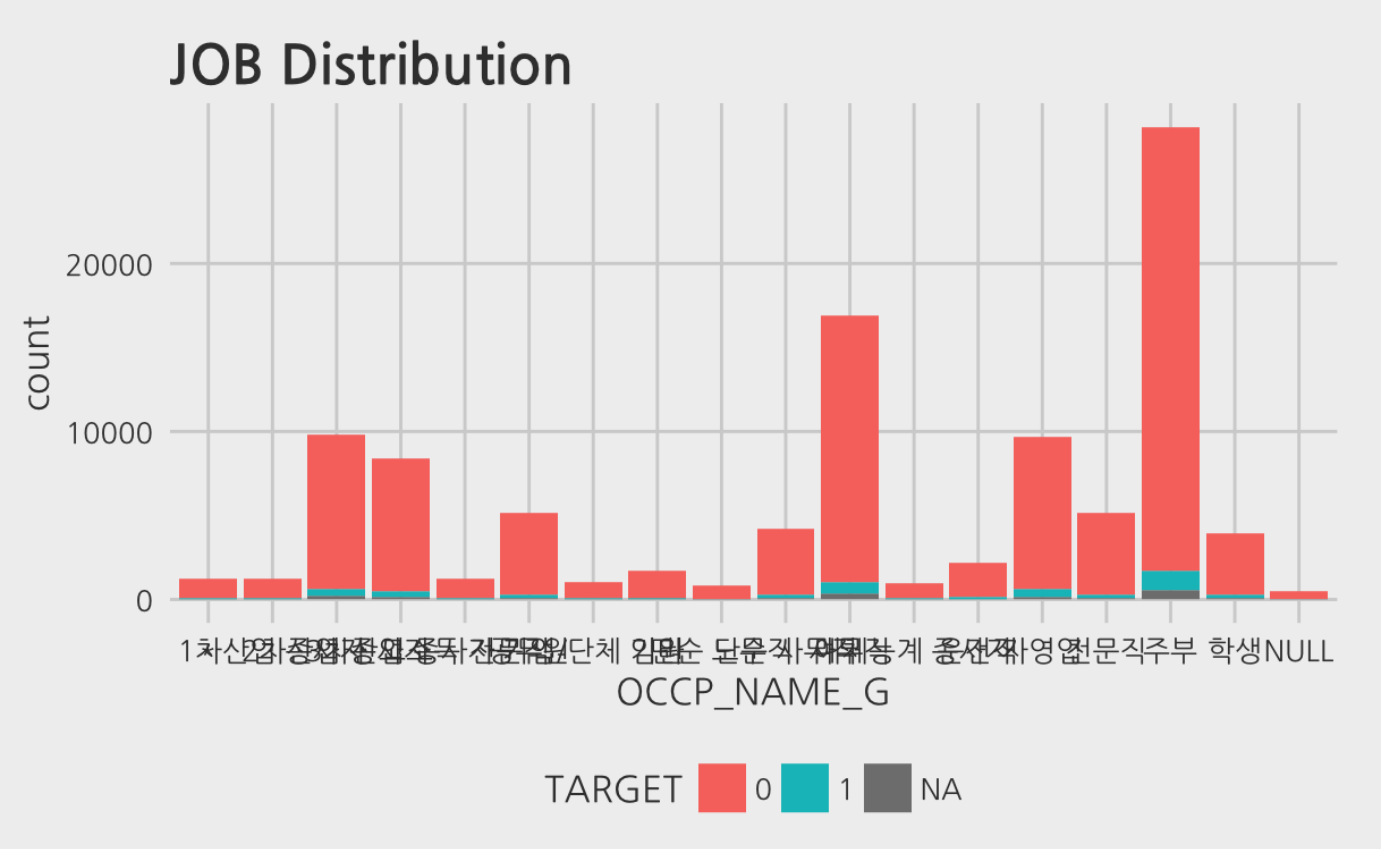
summary 결과,

- 결측치(NA값)가 “NULL”로 되어있는 것을 볼 수 있다.
- 알 수 없는 경우와 직업이 없는 경우 등이 * 로 비식별 처리되었다.

분포를 보기 위해 시각화해보면,

Hide

```
full.data %>%
  ggplot(aes(OCCP_NAME_G,fill=TARGET)) +
  geom_bar() +
  ggtitle("JOB Distribution")
```



*로 비식별된 데이터는 비식별의 특수성을 고려하여 하나의 범주로 처리한다.

그리고 NULL 데이터는 어떻게 처리해야 할까? 우선 직업별 연체자의 비율을 살펴보자.

Hide

```
# 직업별 연체자의 비율
# NULL이면서 TARGET이 1(연체한)인 사람의 비율 : 0.05818966
nrow(subset(train.data, train.data$OCCP_NAME_G == "NULL" & train.data$TARGET == 1))/s
um(train.data$OCCP_NAME_G == "NULL")
```

```
[1] 0.05818966
```

Hide

```
# 주부이면서 TARGET이 1(연체한)인 사람의 비율 : 0.04106657
nrow(subset(train.data, train.data$OCCP_NAME_G == "주부" & train.data$TARGET == 1))/su
m(train.data$OCCP_NAME_G == "주부")
```

```
[1] 0.04106657
```

Hide

```
# 사무직이면서 TARGET이 1(연체한)인 사람의 비율 : 0.04191545
nrow(subset(train.data, train.data$OCCP_NAME_G == "사무직" & train.data$TARGET == 1))/s
um(train.data$OCCP_NAME_G == "사무직")
```

```
[1] 0.04191545
```

Hide

```
# 3차산업 종사자이면서 TARGET이 1(연체한)인 사람의 비율 : 0.04386707
nrow(subset(train.data, train.data$OCCP_NAME_G == "3차산업 종사자" & train.data$TARGET ==
 1))/sum(train.data$OCCP_NAME_G == "3차산업 종사자")
```

```
[1] 0.04386707
```

Hide

```
# 전문직이면서 TARGET이 1(연체한)인 사람의 비율 : 0.04322824
nrow(subset(train.data, train.data$OCCP_NAME_G == "전문직" & train.data$TARGET == 1))/s
um(train.data$OCCP_NAME_G == "전문직")
```

```
[1] 0.04322824
```

Hide

```
# 공무원이면서 TARGET이 1(연체한)인 사람의 비율 : 0.03555294
nrow(subset(train.data, train.data$OCCP_NAME_G == "공무원" & train.data$TARGET == 1))/s
um(train.data$OCCP_NAME_G == "공무원")
```

```
[1] 0.03555294
```

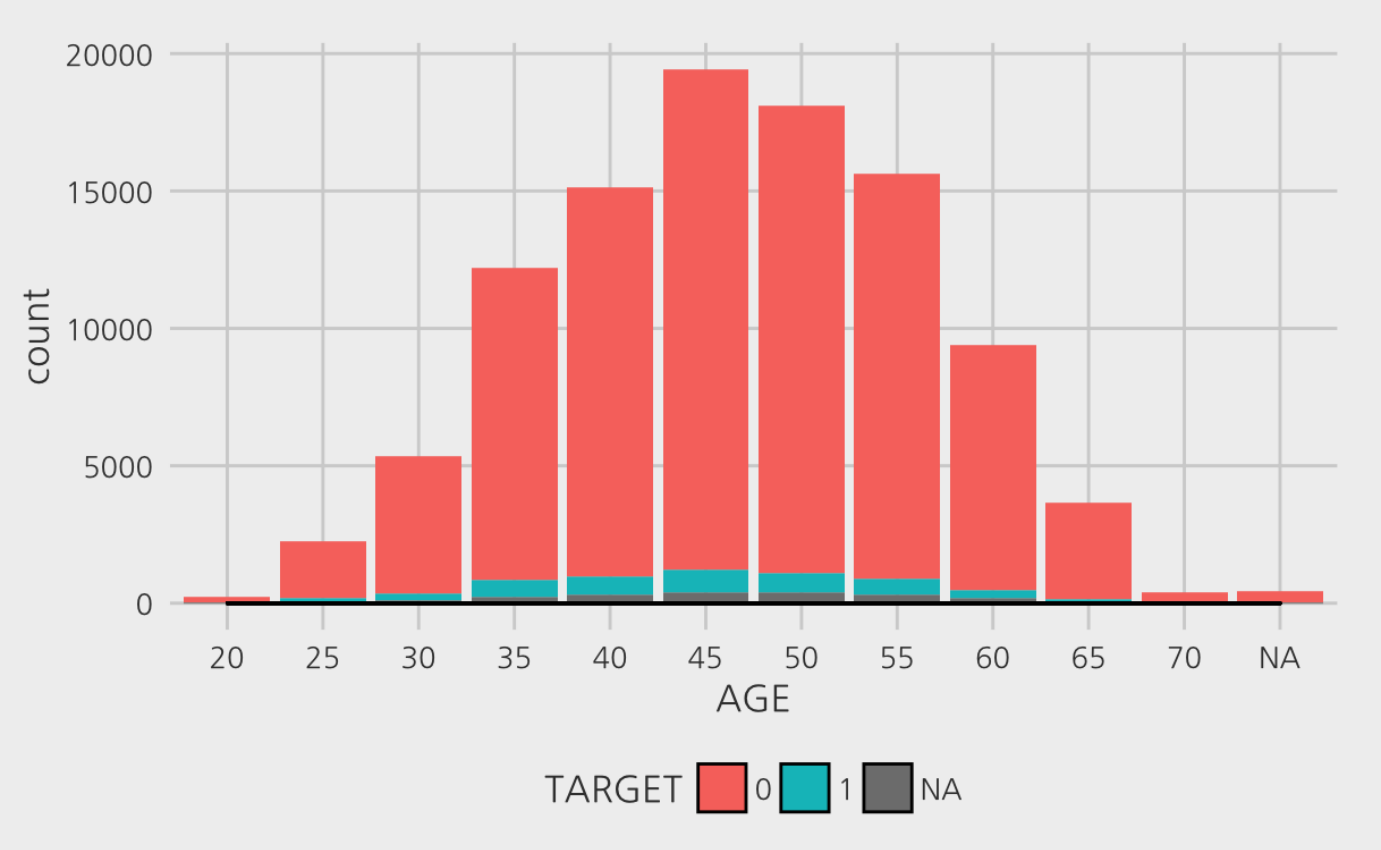
다음 결과에서 NULL인 경우의 연체자의 비율이 다른 직업군보다 연체율이 높다.

그러므로 NULL인 데이터를 하나의 범주로 처리한다.

2) AGE 데이터 살펴보기

Hide

```
# *로 비식별된 데이터를 NA로 대체한다.
full.data$AGE[full.data$AGE == "*"] <- NA
# 그래프로 분포 시각화
full.data %>%
  ggplot(aes(AGE, fill=TARGET)) +
  geom_bar() +
  geom_density(alpha=.5)
```



Hide

```
summary(full.data$AGE)
```

```
    *     20     25     30     35     40     45     50     55     60     65     70 
    0   245   2254   5360 12217 15116 19426 18116 15643   9381   3652    408 
NA's 
  434 
```

AGE데이터의 결측값은 434개이며, 전체 데이터의 약 0.4%정도이다. 통계적으로 영향을 미치는 정도가 적지만 삭제하기엔 손실되는 데이터가 아깝다.

분류모델인 의사결정나무로 예측하여 대체하자 의사결정나무는 계산 비용이 낮아 대규모의 데이터 셋에서도 비교적 빠르게 연산이 가능하다.

결측치처리

Hide

```
##### Decision Tree #####
col.pred <- c("HSHD_INFR_INCM","FMLY_GDINS_MNPREM","TOT_PREM","AUTR_FAIL_MCNT", "AGE"
,
              "FMLY_PLPY_CNT","CPT_LNIF_CNT","ECT_LNIF_CNT","TOT_LNIF_AMT","CRDT_OCC
R_MDIF",
              "CRDT_CARD_CNT","OCCP_NAME_G","SEX","PAYM_METD","MATE_OCCP_NAME_G")
# train에 쓸 데이터프레임을 만든다.
df.pred <- full.data[!is.na(full.data$AGE), col.pred]
# Decision Tree model
AGE.rpart <- rpart(as.factor(AGE) ~ .,
                           data = df.pred,
                           method = "class",
                           na.action=na.omit)
# 결측치 채우기
full.data$AGE[is.na(full.data$AGE)] <- predict(AGE.rpart,
                           full.data[is.na(full.data$AGE),col.pred],
                           type="class")
summary(full.data$AGE)
```

```
    *    20    25    30    35    40    45    50    55    60    65    70
    0   245  2254  5360 12359 15116 19426 18408 15643  9381  3652   408
```

Hide

```
# 빈 factor 레벨 지우기
full.data$AGE <- as.character(full.data$AGE)
full.data$AGE <- as.factor(full.data$AGE)
```

3) SEX 성별 데이터 살펴보기

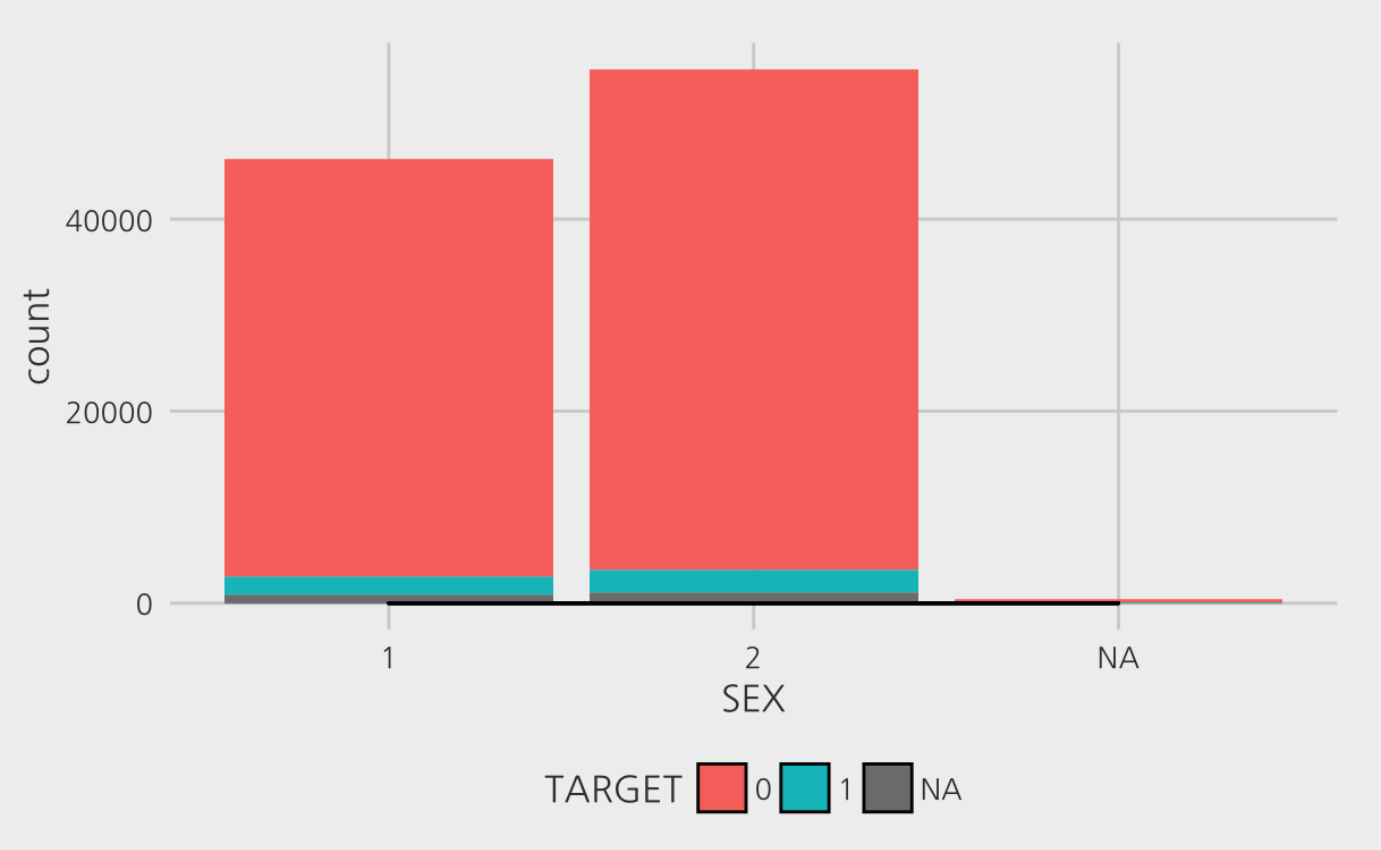
Hide

```
full.data$SEX[full.data$SEX == "*"] <- NA
summary(full.data$SEX)
```

```
    *     1     2  NA's
    0 46218 55600   434
```

Hide

```
full.data %>%
  ggplot(aes(SEX, fill=TARGET)) +
  geom_bar() +
  geom_density(alpha=.5)
```



2(여성)데이터가 더 많으며,

SEX(성별)데이터의 결측치는 434개로 전체 데이터 중 0.4% 정도이다.

마찬가지로 의사결정나무로 예측하여 결측값 채우기

결측치처리

Hide

```
##### Decision Tree #####
full.data$SEX[full.data$SEX == "NULL"] <- NA
col.pred3 <- c("HSHD_INFR_INCM","FMLY_GDINS_MNPREM","TOT_PREM","AUTR_FAIL_MCNT", "AG
E",
               "FMLY_PLPY_CNT","CPT_LNIF_CNT","SPART_LNIF_CNT","ECT_LNIF_CNT","TOT_LN
IF_AMT",
               "CRDT_OCCR_MDIF","SPTCT_OCCR_MDIF","CRDT_CARD_CNT","OCCP_NAME_G","SEX"
,
               "MON_TLFE_AMT","TEL_CNTT_QTR", "TEL_MBSP_GRAD" ,"PAYM_METD","MATE_OCCP
_NAME_G"
               )
df.pred3 <- full.data[!is.na(full.data$SEX), col.pred3]
SEX.rpart <- rpart(as.factor(SEX) ~ .,
                           data = df.pred3,
                           method = "class",
                           na.action=na.omit)
full.data$SEX[is.na(full.data$SEX)] <- predict(SEX.rpart,
                           full.data[is.na(full.data$SEX),col.pred3],
                           type="class")
summary(full.data$SEX)
```

```
    *     1     2
    0 46308 55944
```

Hide

```
# 빈 factor레벨 지우기
full.data$SEX <- as.character(full.data$SEX)
full.data$SEX <- as.factor(full.data$SEX)
#잘 들어갔는지 확인
summary(full.data$SEX)
```

```
    1     2
46308 55944
```

4) LAST_CHLD_AGE 막내자녀나이 데이터 살펴보기

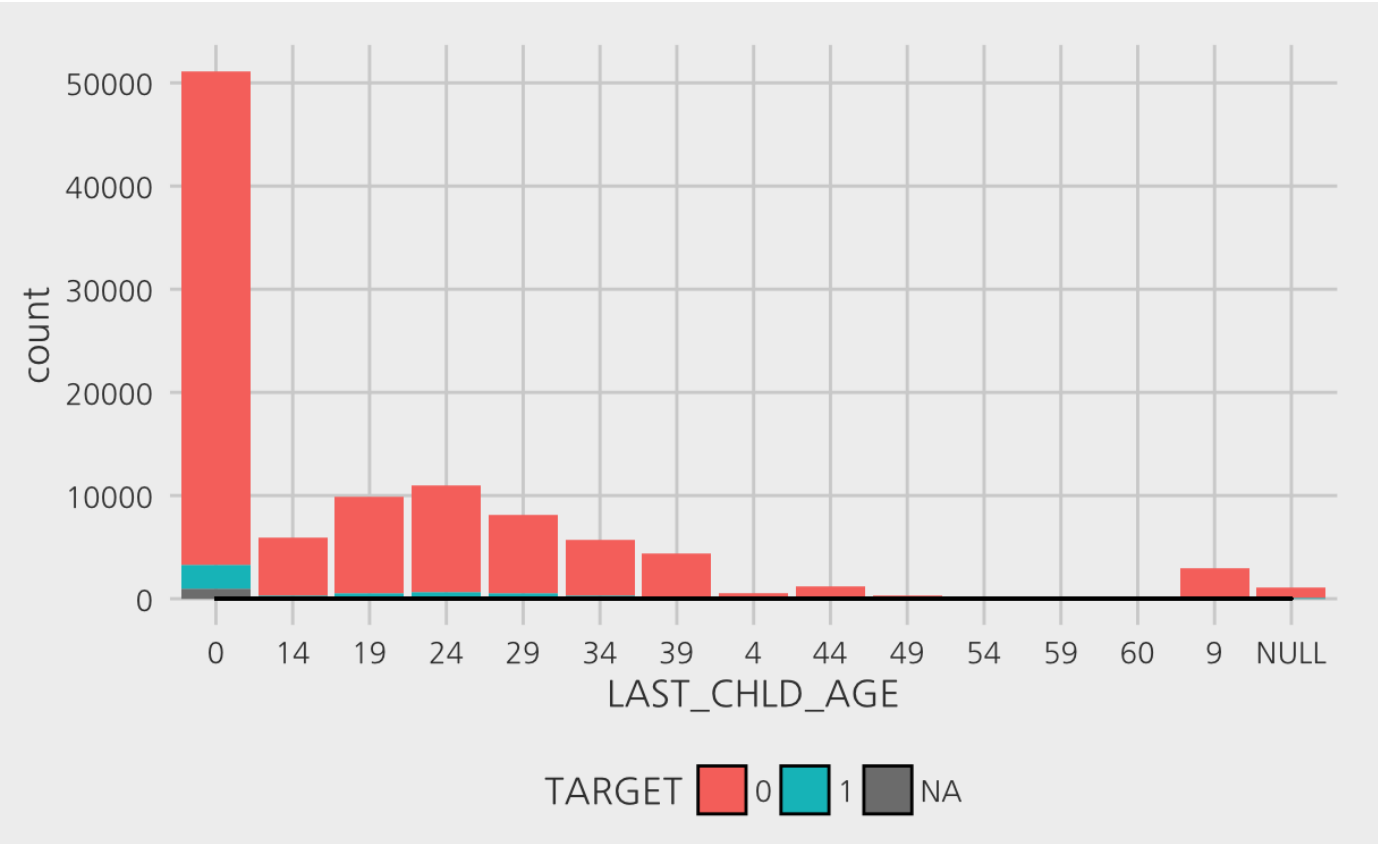
Hide

```
summary(full.data$LAST_CHLD_AGE)
```

```
    0    14    19    24    29    34    39     4    44    49    54    59
51146  5910  9834 10992  8097  5702  4371   552  1248   335    46    17
   60     9  NULL
   33  2928  1041
```

Hide

```
full.data %>%
  ggplot(aes(LAST_CHLD_AGE, fill=TARGET)) +
  geom_bar() +
  geom_density(alpha=.5)
```



0인 (나이 또는 존재 자체를 를 알 수 없는) 데이터 수가 너무 많다.

NULL 데이터도 0으로 대체한다. (변수설명에 NULL=0 이라고 되어있으므로)

Hide

```
full.data$LAST_CHLD_AGE[is.na(full.data$LAST_CHLD_AGE)] <- 0
```

5) PAYM_METD 납부방법 데이터 살펴보기

Hide

```
summary(full.data$PAYM_METD)
```

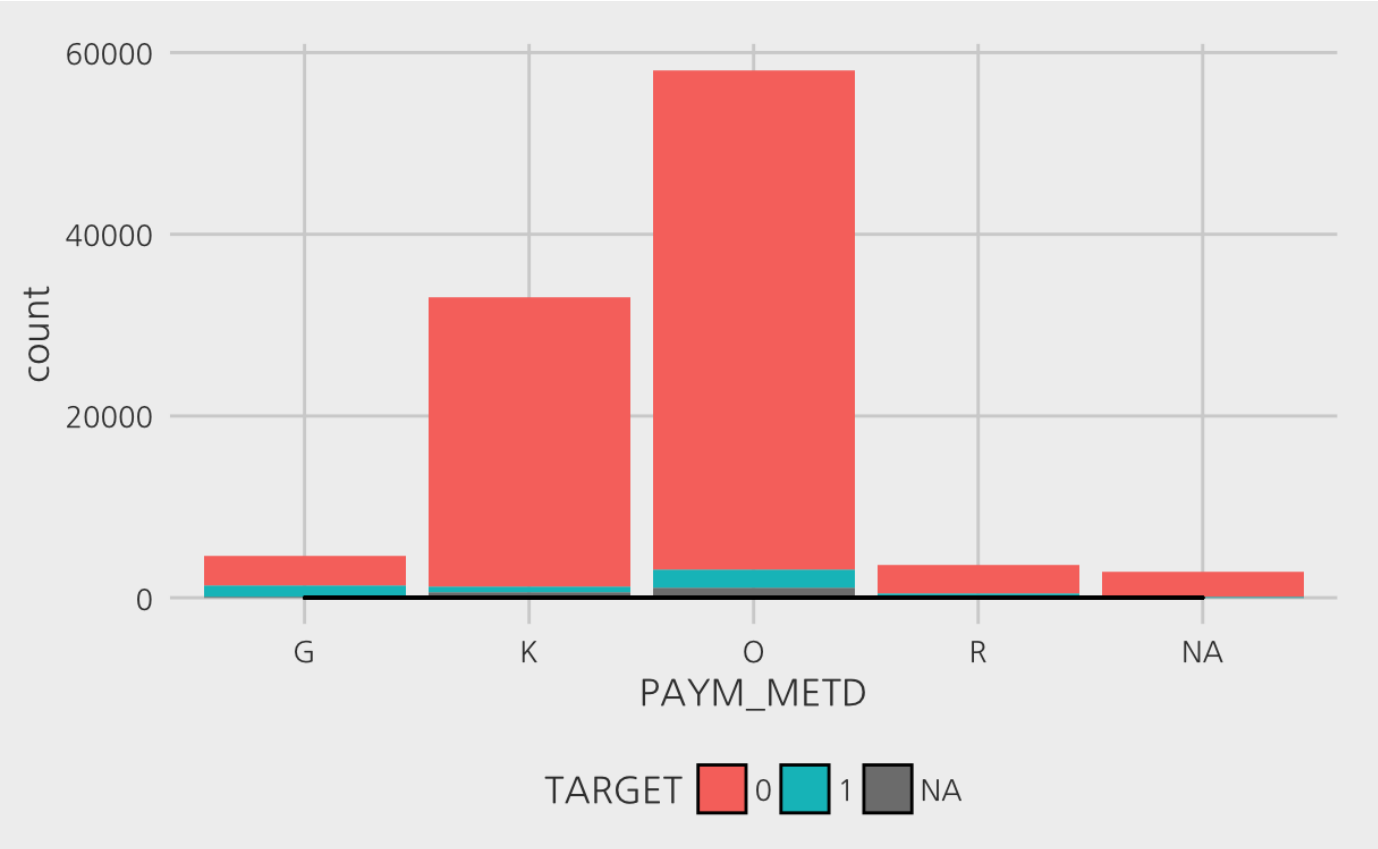
```
          G     K     O     R
 2878  4625 33027 58076  3646
```

결측,비식별 " “데이터 2878개 약 3%

시각화해보면,

Hide

```
full.data$PAYM_METD[full.data$PAYM_METD==""] <- NA
full.data %>%
  ggplot(aes(PAYM_METD, fill=TARGET)) +
  geom_bar() +
  geom_density(alpha=.5)
```



범주가 적으므로 MICE(다중대체법)로 예측하여 채워넣는다. 다음에 살펴볼 TEL_MBSP_GRAD(멤버쉽 등급 데이터)결측치와 같이 처리하도록 한다.

6) TEL_MBSP_GRAD 멤버쉽 등급 데이터 살펴보기

Hide

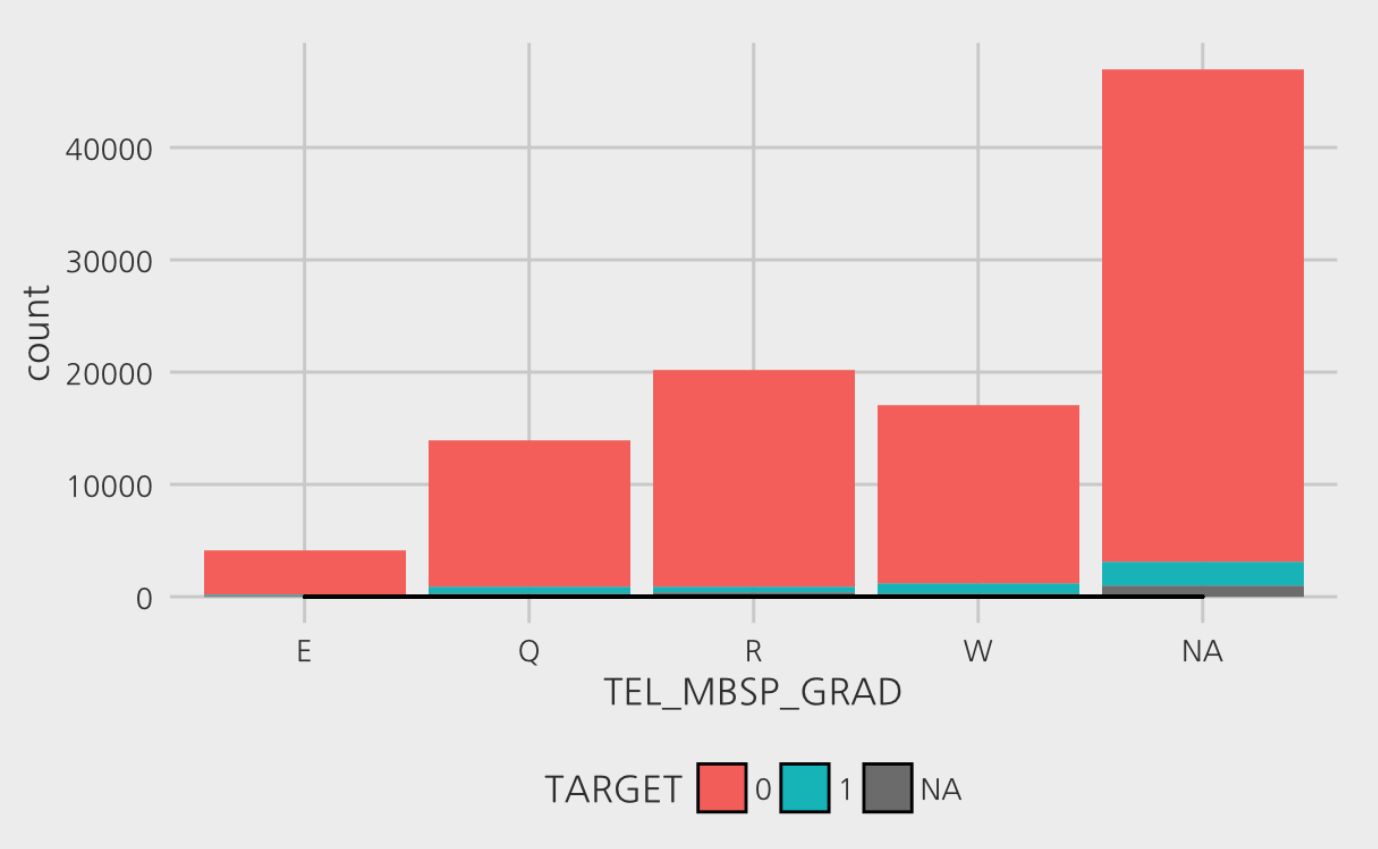
```
summary(full.data$TEL_MBSP_GRAD)
```

```
          E     Q     R     W
46989  4103 13913 20235 17012
```

46989개 데이터의 멤버쉽 등급을 알 수 없다. 전체 데이터의 약 45% 정도이다.

Hide

```
full.data$TEL_MBSP_GRAD[full.data$TEL_MBSP_GRAD==""] <- NA
full.data %>%
  ggplot(aes(TEL_MBSP_GRAD, fill=TARGET)) +
  geom_bar() +
  geom_density(alpha=.5)
```



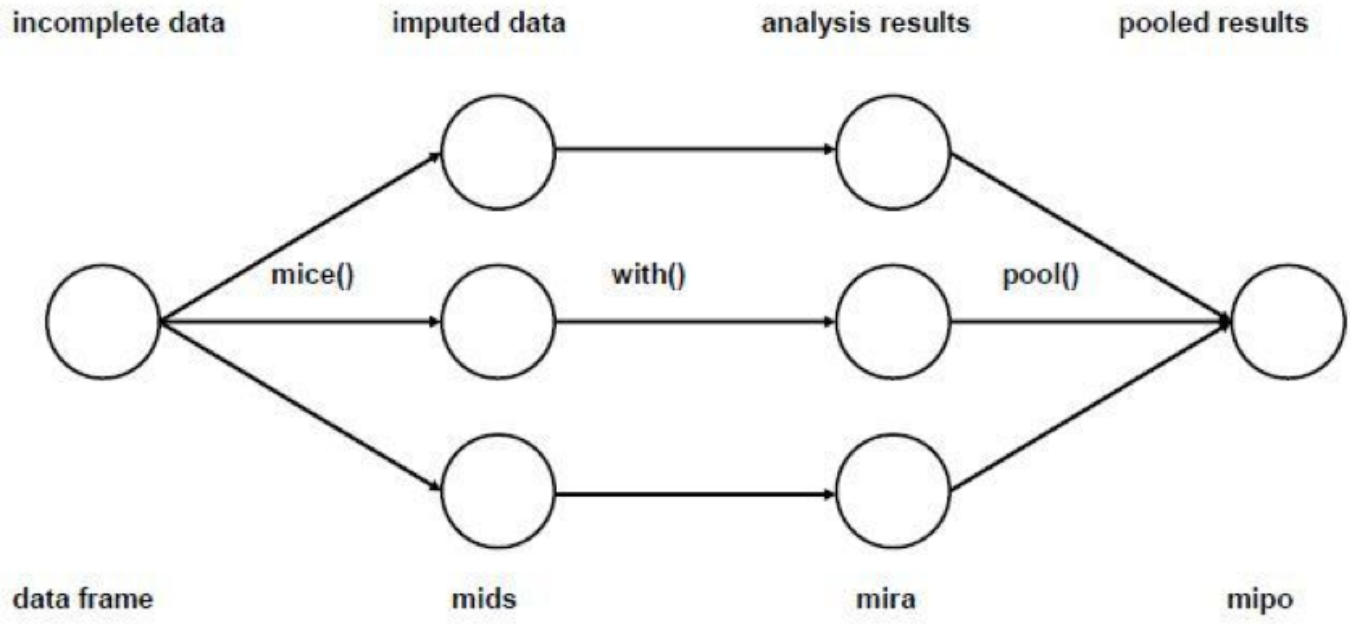
멤버쉽 등급은 전년도 누적 납부금액, 가입기간 및 연체 여부를 합산해 VIP, Gold, Silver,일반으로 구분된다.

ARPU(가입자매출), MON_TLFE_AMT(납부요금), CBPT_MBSP_YN(결합상품가입여부), TEL_CNTT_QTR(가입년월), CRMM_OVDU_AMT(당월연체금액), TLFE_UNPD_CNT(납부일미준수횟수), LT1Y_MXOD_AMT(년간최대연체금액) 변수들을 이용해 멤버쉽 등급을 예측하여 대체한다.

대체 방법으로 **MICE (multiple imputation) 다중대체법**을 사용한다.

MICE(Multivariate Imputation) 다중대체법

: 시뮬레이션을 반복하여 누락된 데이터를 채워넣는 방법



1. simulation을 통하여 누락된 자료를 채운 dataset을 3-5개 만든다.(default = 5)
2. 각각의 dataset에 대해 표준적인 통계방법을 적용하여 분석
3. 분석 결과를 하나의 결과로 통합한다.

TEL_MBSP_GRAD 멤버쉽 등급, PAYM_METD 납부방법 결측치 대체하기

Hide

```
##### MICE (multiple imputation) #####
# install.packages("mice")
library(mice)
# 빈 factor 레벨 삭제하기
full.data$TEL_MBSP_GRAD <- as.character(full.data$TEL_MBSP_GRAD)
full.data$TEL_MBSP_GRAD <- as.factor(full.data$TEL_MBSP_GRAD)
full.data$PAYM_METD <- as.character(full.data$PAYM_METD)
full.data$PAYM_METD <- as.factor(full.data$PAYM_METD)
mice.col <- c("TEL_MBSP_GRAD","ARPU", "MON_TLFE_AMT", "CBPT_MBSP_YN", "TEL_CNTT_QTR",
 "CRMM_OVDU_AMT", "TLFE_UNPD_CNT", "LT1Y_MXOD_AMT", "PAYM_METD")
imp = mice(full.data[, mice.col],method = c("cart"),seed=1234)
```

```
 iter imp variable
  1   1  TEL_MBSP_GRAD  PAYM_METD
  1   2  TEL_MBSP_GRAD  PAYM_METD
  1   3  TEL_MBSP_GRAD  PAYM_METD
  1   4  TEL_MBSP_GRAD  PAYM_METD
  1   5  TEL_MBSP_GRAD  PAYM_METD
  2   1  TEL_MBSP_GRAD  PAYM_METD
  2   2  TEL_MBSP_GRAD  PAYM_METD
  2   3  TEL_MBSP_GRAD  PAYM_METD
  2   4  TEL_MBSP_GRAD  PAYM_METD
  2   5  TEL_MBSP_GRAD  PAYM_METD
  3   1  TEL_MBSP_GRAD  PAYM_METD
  3   2  TEL_MBSP_GRAD  PAYM_METD
  3   3  TEL_MBSP_GRAD  PAYM_METD
  3   4  TEL_MBSP_GRAD  PAYM_METD
  3   5  TEL_MBSP_GRAD  PAYM_METD
  4   1  TEL_MBSP_GRAD  PAYM_METD
  4   2  TEL_MBSP_GRAD  PAYM_METD
  4   3  TEL_MBSP_GRAD  PAYM_METD
  4   4  TEL_MBSP_GRAD  PAYM_METD
  4   5  TEL_MBSP_GRAD  PAYM_METD
  5   1  TEL_MBSP_GRAD  PAYM_METD
  5   2  TEL_MBSP_GRAD  PAYM_METD
  5   3  TEL_MBSP_GRAD  PAYM_METD
  5   4  TEL_MBSP_GRAD  PAYM_METD
  5   5  TEL_MBSP_GRAD  PAYM_METD
```

Hide

```
full.data[, mice.col] <- complete(imp) # merge the imputed values into original data
 set
# 빈 factor레벨 지우기
full.data$PAYM_METD <- as.character(full.data$PAYM_METD)
full.data$PAYM_METD <- as.factor(full.data$PAYM_METD)
full.data$TEL_MBSP_GRAD <- as.character(full.data$TEL_MBSP_GRAD)
full.data$TEL_MBSP_GRAD <- as.factor(full.data$TEL_MBSP_GRAD)
```

금액 데이터 단위 1원으로 맞추기

- SCI 데이터 * 1000
- 한화생명 소득금액 데이터 * 10000

Hide

```
# SCI 데이터 - 단위 : 1000원
full.data$TOT_LNIF_AMT <- (full.data$TOT_LNIF_AMT) * 1000
full.data$TOT_CLIF_AMT <- (full.data$TOT_CLIF_AMT) * 1000
full.data$BNK_LNIF_AMT <- (full.data$BNK_LNIF_AMT) * 1000
full.data$CPT_LNIF_AMT <- (full.data$CPT_LNIF_AMT) * 1000
full.data$CB_GUIF_AMT <- (full.data$CB_GUIF_AMT) * 1000
# 한화생명 소득금액 데이터 - 단위 : 10000원
full.data$CUST_JOB_INCM <- (full.data$CUST_JOB_INCM) * 10000
full.data$HSHD_INFR_INCM <- (full.data$HSHD_INFR_INCM) * 10000
# SKT데이터 단위는 1원
```

금액 데이터 변환 작업

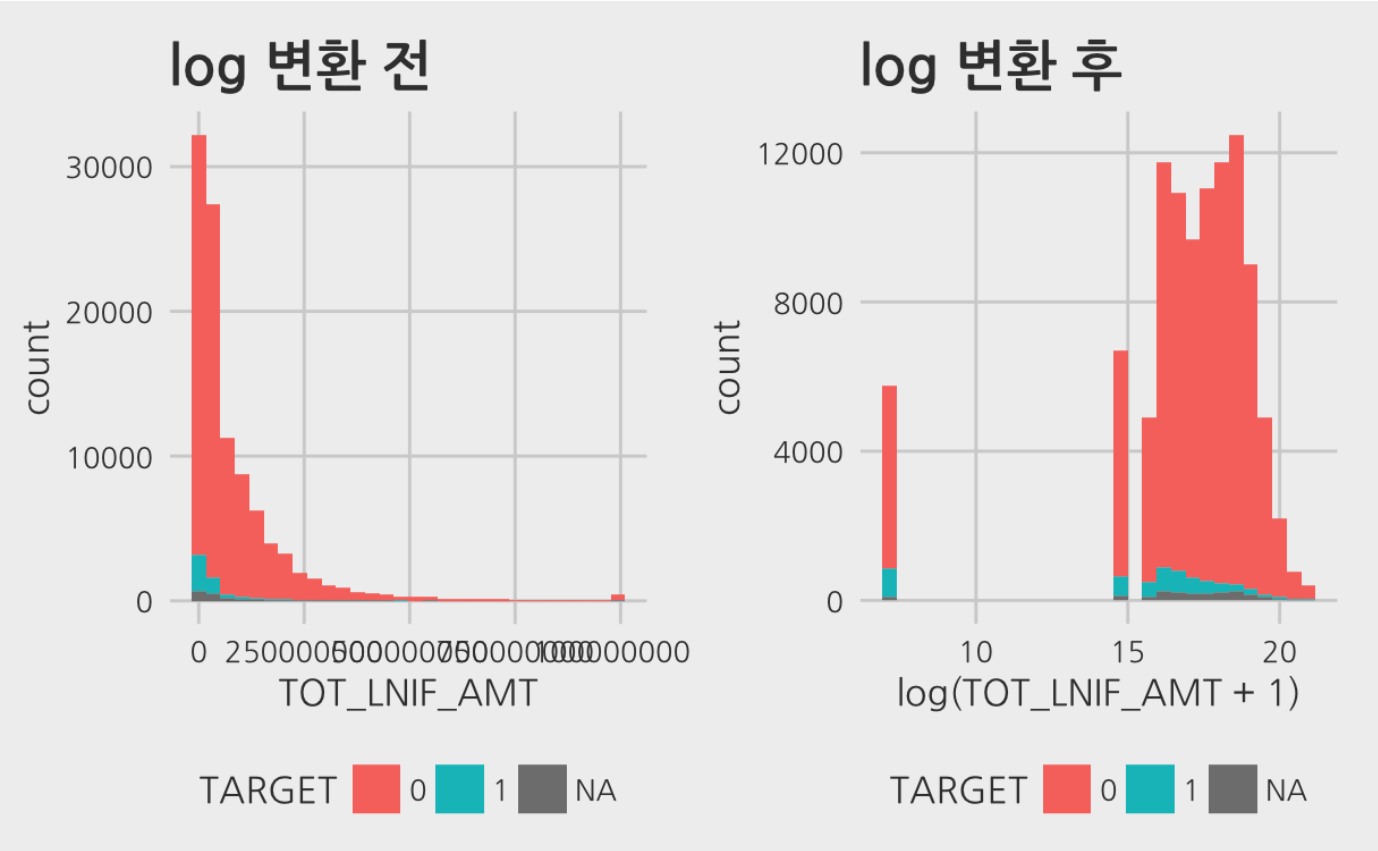
Hide

```
library(gridExtra)
```

```
다음의 패키지를 부착합니다: 'gridExtra'

The following object is masked from 'package:randomForest':

    combine

The following object is masked from 'package:dplyr':

    combine
```

Hide

```
# TOT_LNIF_AMT 대출정보 현재 총 금액 로그변환
plot1 <- full.data %>% ggplot(aes(TOT_LNIF_AMT, fill=TARGET)) + geom_histogram(bins =
 30) + ggtitle("log 변환 전")
plot2 <- full.data %>% ggplot(aes(log(TOT_LNIF_AMT + 1), fill=TARGET)) + geom_histogr
am(bins = 30) + ggtitle("log 변환 후")
grid.arrange(plot1, plot2,  nrow=1, ncol=2)
```



일부 금액 데이터를 히스토그램으로 시각화해 보면 왼쪽 그래프처럼 데이터가 한쪽으로 몰려있다.

정규화 하기 위해 log또는 sqrt변환을 해준다.

다른 변수도 마찬가지로 확인후 변환한다.

Hide

```
# plot3 <- full.data %>% ggplot(aes(LT1Y_STLN_AMT, fill=TARGET)) + geom_histogram(bin
s = 30) + labs(title="변화 전 데이터")
# plot4 <- full.data %>% ggplot(aes(log(LT1Y_STLN_AMT + 1), fill=TARGET)) + geom_hist
ogram(bins = 30) + labs(title="log변환 후")
# plot5 <- full.data %>% ggplot(aes(sqrt(LT1Y_STLN_AMT), fill=TARGET)) + geom_histogr
am(bins = 30) + labs(title="sqrt변환 후")
# grid.arrange(plot3, plot4, plot5 ,nrow=1, ncol=3)
#
# # 정규성 검정
# shapiro.test(sample(full.data$LT1Y_STLN_AMT,1000,replace = FALSE))
# shapiro.test(sample(log(full.data$LT1Y_STLN_AMT+1),1000,replace = FALSE))
# shapiro.test(sample(sqrt(full.data$LT1Y_STLN_AMT),1000,replace = FALSE))
```

Hide

```
########################################금액데이터 변환#####################################
#
full.data$TOT_LNIF_AMT = log(full.data$TOT_LNIF_AMT+1)
full.data$TOT_CLIF_AMT = log(full.data$TOT_CLIF_AMT+1)
full.data$BNK_LNIF_AMT = log(full.data$BNK_LNIF_AMT+1)
full.data$CPT_LNIF_AMT = log(full.data$CPT_LNIF_AMT+1)
full.data$CB_GUIF_AMT =  log((full.data$CB_GUIF_AMT)+1)
full.data$TOT_CRLN_AMT = log((full.data$TOT_CRLN_AMT)+1)
full.data$TOT_REPY_AMT = log((full.data$TOT_REPY_AMT)+1)
full.data$STLN_REMN_AMT = log(full.data$STLN_REMN_AMT+1)
full.data$LT1Y_STLN_AMT = log(full.data$LT1Y_STLN_AMT+1)
full.data$FYCM_PAID_AMT = log(full.data$FYCM_PAID_AMT+1)
full.data$MON_TLFE_AMT =sqrt(full.data$MON_TLFE_AMT)
full.data$CRMM_OVDU_AMT = log(full.data$CRMM_OVDU_AMT+1)
full.data$LT1Y_MXOD_AMT = sqrt(full.data$LT1Y_MXOD_AMT)
full.data$GDINS_MON_PREM = log(full.data$GDINS_MON_PREM+1)
full.data$SVINS_MON_PREM = log(full.data$SVINS_MON_PREM+1)
full.data$MAX_MON_PREM = log(full.data$MAX_MON_PREM+1)
full.data$TOT_PREM = log(full.data$TOT_PREM+1)
full.data$FMLY_TOT_PREM = log(full.data$FMLY_TOT_PREM+1)
full.data$FMLY_GDINS_MNPREM = sqrt(full.data$FMLY_GDINS_MNPREM)
full.data$FMLY_SVINS_MNPREM = log(full.data$FMLY_SVINS_MNPREM+1)
full.data$MOBL_PRIN = sqrt(full.data$MOBL_PRIN)
```

3. 데이터 나누기 (train, test)

Hide

```
# write.csv(full.data, file="full.data.csv",row.names = FALSE)
# reload.data <- read.csv("full.data.csv",header = T)
# full.data <- reload.data
data.df <- full.data[1:100233,]
test.df <- full.data[100234:102252,]
# data.df를 다시 train, test데이터로 나누기 (80:20 비율로)
parts <- createDataPartition(data.df$TARGET, p=0.8)
train.data <- data.df[parts$Resample1,]
dim(train.data)
```

```
[1] 80187    69
```

Hide

```
test.data <- data.df[-parts$Resample1,]
dim(test.data)
```

```
[1] 20046    69
```

4. 변수 중요도 파악하기

1) 병렬처리를 위한 Parallel패키지 사용하기

Hide

```
# Parallel 셋팅
library(parallelMap)
library(parallel)
parallelStartSocket(cpus = detectCores())
```

```
Parallelization was not stopped, doing it now.Stopped parallelization. All cleaned u
p.
Starting parallelization in mode=socket with cpus=4.
```

Hide

```
# parallelStop()
```

필터 방법으로 모델링에 사용할 변수를 선택하기로 한다.

먼저 분산이 0에 가까운 변수를 찾아 제외한다.

(분산이 0에 가까운 변수는 서로 다른 관찰을 구분하는 데 별 소용이 없다. 따라서 데이터 모델링에서도 그리 유용하지 않다.)

2) nearZeroVar() 함수를 사용해 분산이 0에 가까운 변수를 찾기

Hide

```
# 분산이 0에 가까운 변수를 찾아 제외한다.
library(mlbench)
nearZeroVar(train.data)
```

```
 [1]  2 15 16 24 26 27 28 29 30 31 32 33 37 38 39 41 43 48 55 56 63 64 65
[24] 66 68 69
```

Hide

```
mean.var <- train.data[,-nearZeroVar(train.data)]
# 69개의 변수 중 43개의 변수가 선택되었다.
names(mean.var)
```

```
 [1] "CUST_ID"           "BNK_LNIF_CNT"      "CPT_LNIF_CNT"
 [4] "SPART_LNIF_CNT"    "ECT_LNIF_CNT"      "TOT_LNIF_AMT"
 [7] "TOT_CLIF_AMT"      "BNK_LNIF_AMT"      "CPT_LNIF_AMT"
[10] "CRDT_OCCR_MDIF"    "SPTCT_OCCR_MDIF"   "CRDT_CARD_CNT"
[13] "CTCD_OCCR_MDIF"    "OCCP_NAME_G"       "CUST_JOB_INCM"
[16] "HSHD_INFR_INCM"    "ACTL_FMLY_NUM"     "CUST_FMLY_NUM"
[19] "LAST_CHLD_AGE"     "MATE_OCCP_NAME_G"  "CRDT_LOAN_CNT"
[22] "PREM_OVDU_RATE"    "LT1Y_PEOD_RATE"    "AVG_STLN_RATE"
[25] "GDINS_MON_PREM"    "FMLY_GDINS_MNPREM" "MAX_MON_PREM"
[28] "TOT_PREM"          "FMLY_TOT_PREM"     "CNTT_LAMT_CNT"
[31] "AUTR_FAIL_MCNT"    "FYCM_PAID_AMT"     "FMLY_CLAM_CNT"
[34] "FMLY_PLPY_CNT"     "AGE"               "SEX"
[37] "TEL_MBSP_GRAD"     "ARPU"              "MON_TLFE_AMT"
[40] "CBPT_MBSP_YN"      "MOBL_FATY_PRC"     "TEL_CNTT_QTR"
[43] "PAYM_METD"
```

3) 랜덤포레스트 모델을 사용해 중요한 변수 찾기

3. 변수 중요도 평가

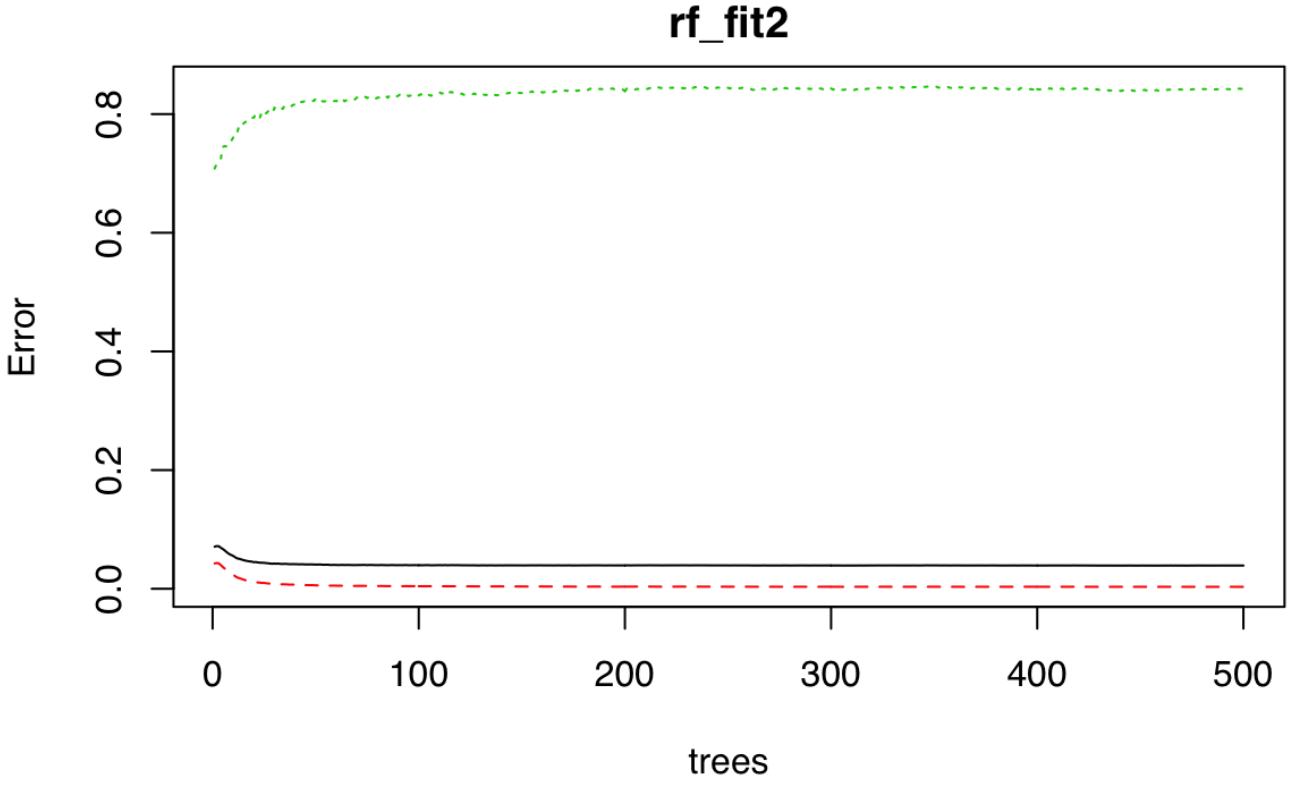
랜덤포레스트 모델은 변수의 중요도를 평가하고 모델링에 사용할 변수를 선택하는 데 사용할 수 있다.

변수 중요도는 변수가 정확도(Accuracy)와 노드 불순도(Node Impurity) 개선에 얼마만큼 기여하는지로 측정된다.

이렇게 구한 변수 중요도는 다른 모델에 사용할 변수를 선택하는 데 사용할 수 있다.

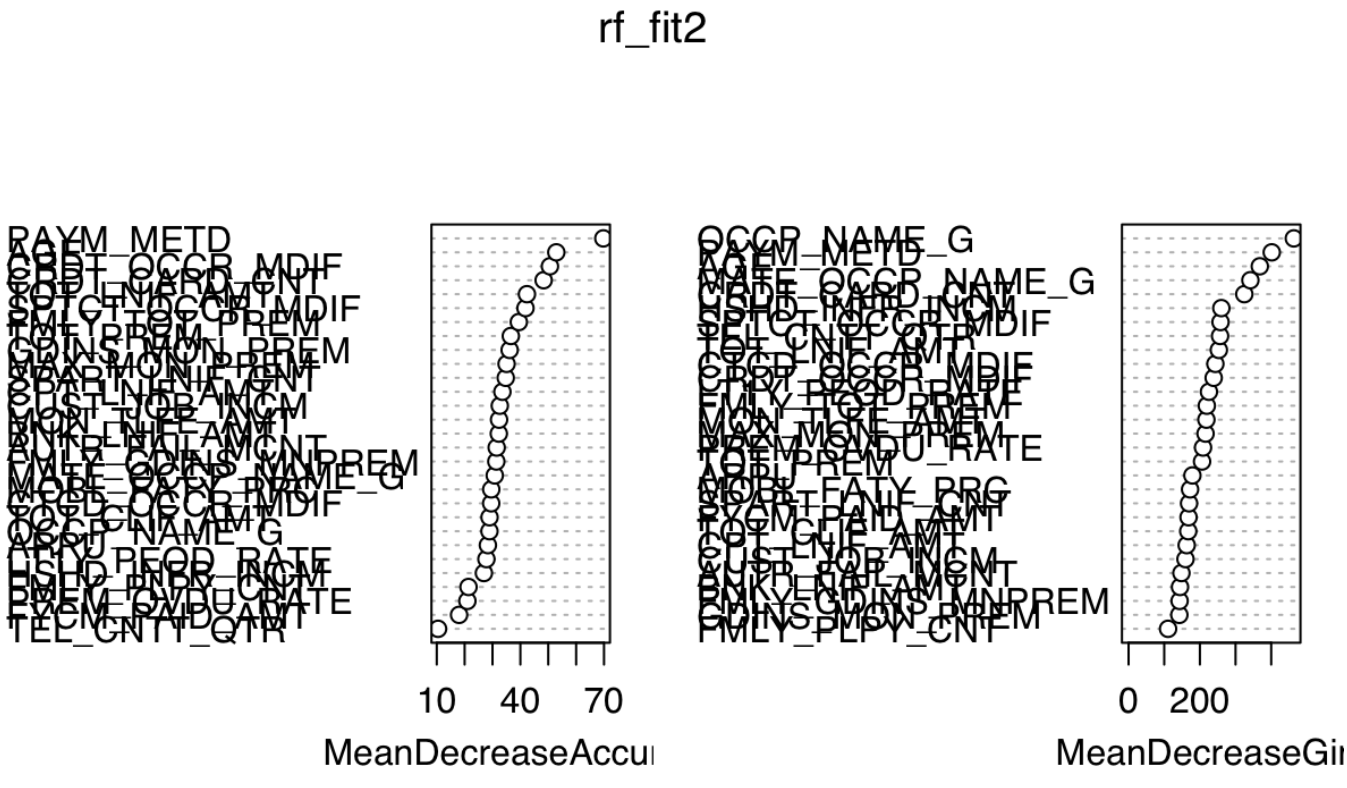
Hide

```
# 위에서 필터링한 43개의 변수 중 중요할 것 같은 변수 30개를 넣어서 랜덤포레스트 모델로 돌려보기
col.pred <- c( "CTCD_OCCR_MDIF" ,"MATE_OCCP_NAME_G"  ,"PREM_OVDU_RATE","SPART_LNIF_CN
T",
               "TOT_LNIF_AMT","TOT_CLIF_AMT","LT1Y_PEOD_RATE","TARGET","OCCP_NAME_G",
               "CUST_JOB_INCM","HSHD_INFR_INCM","GDINS_MON_PREM","FMLY_GDINS_MNPREM",
               "TOT_PREM","FMLY_TOT_PREM","MAX_MON_PREM","SPTCT_OCCR_MDIF","CRDT_CARD
_CNT",
               "FMLY_PLPY_CNT","AGE","ARPU", "AUTR_FAIL_MCNT","FYCM_PAID_AMT", "MON_T
LFE_AMT",
               "MOBL_FATY_PRC","TEL_CNTT_QTR","PAYM_METD","BNK_LNIF_AMT","CPT_LNIF_AM
T","CRDT_OCCR_MDIF")
rf_fit2 <- randomForest(as.factor(TARGET) ~. , data = train.data[,col.pred] ,mtry = 8
 , importance =TRUE)

plot(rf_fit2)
```



Hide

```
tmp <- importance(rf_fit2)
# tmp
varImpPlot(rf_fit2)
```



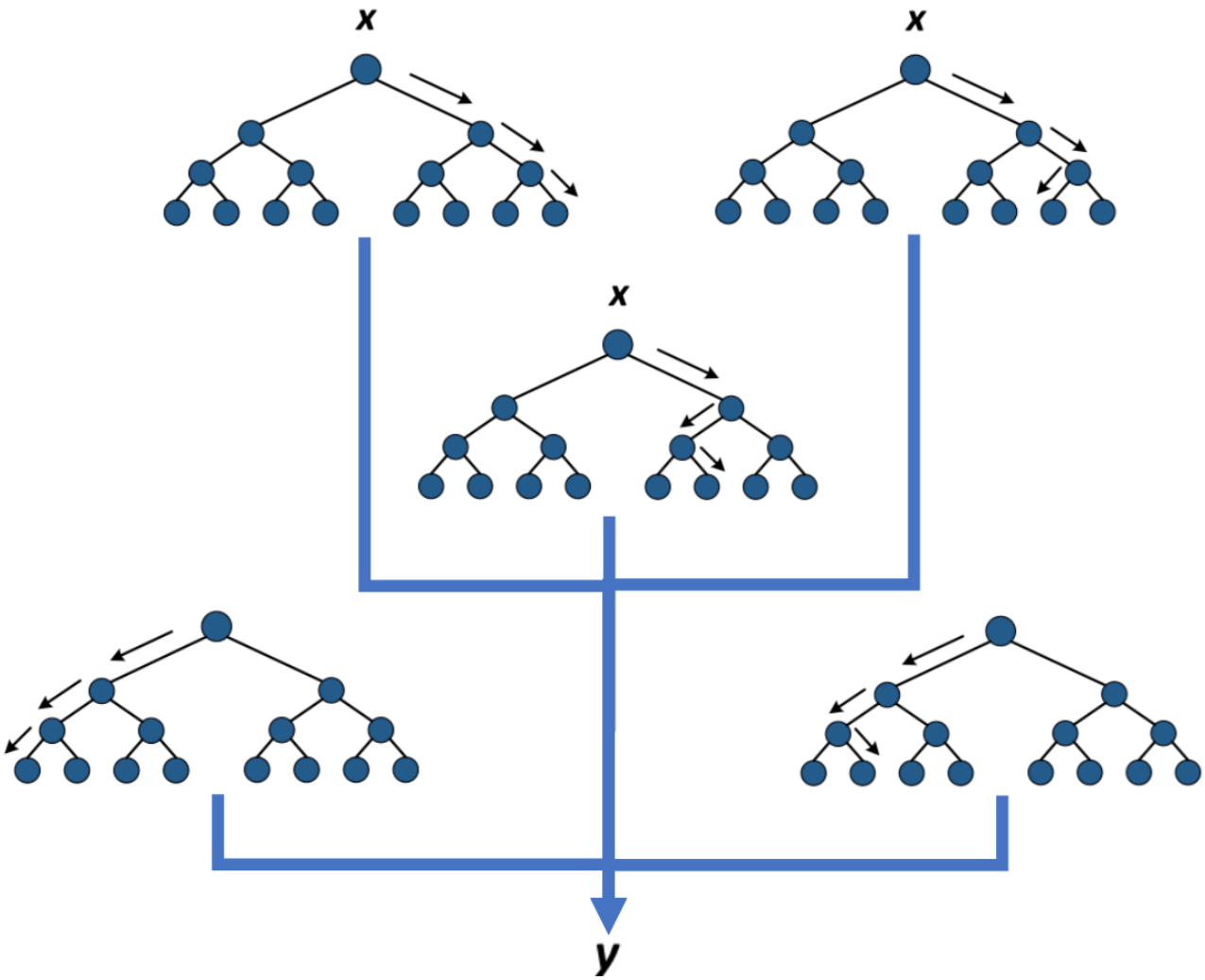
Hide

```
# 정확도 개선에 중요한 변수(MeanDecreaseAccuracy가 높은 변수) 9개 빼놓기 (TARGET포함 10개)
col.pred.9 <- c("PAYM_METD","CRDT_CARD_CNT","AGE","CRDT_OCCR_MDIF",
  "TOT_LNIF_AMT","SPTCT_OCCR_MDIF","FMLY_TOT_PREM","TOT_PREM", "CPT_LNIF_AMT","TARGE
T")
col.pred.9
```

```
 [1] "PAYM_METD"        "CRDT_CARD_CNT"    "AGE"
 [4] "CRDT_OCCR_MDIF"   "TOT_LNIF_AMT"     "SPTCT_OCCR_MDIF"
 [7] "FMLY_TOT_PREM"    "TOT_PREM"         "CPT_LNIF_AMT"
[10] "TARGET"
```

TARGET이 연체하는 경우(1) , 상환하는 경우(0) 를 분류하여 예측하는 것이 목적이다. 그러므로 적절한 모델은 로지스틱 회귀분석, SVM, 의사결정나무, 랜덤포레스트 등이다. 이 중 랜덤 포레스트와 SVM모델을 사용하기로 했다. 둘 중 결과가 좋은 모델을 선택하여 튜닝하고, 성능을 높이는 과정을 반복한다.

랜덤포레스트 모델



주어진 데이터로부터 여러개의 모델을 학습한 다음, 예측 시 여러 모델의 예측 결과들을 종합해 사용하여 정확도를 높이는 앙상블 기법

1. Bagging으로 나온 여러 sub-data로 Decision tree 모델을 만듦
2. 각 결정트리로부터 얻어진 결과를 평균, 확률등 종합하여 최종 결과를 도출
3. 장점

- Random sampled sub-data로 모델간 correlation 방지
- Bagging이 결합되면서 Overfitting 문제를 모델 자체적으로 해결할 수 있음

5. 랜덤포레스트 모델 사용하기

1) 모델 적용, 튜닝하기

Hide

```
library(mlr)
```

```
필요한 패키지를 로딩중입니다: ParamHelpers
replacing previous import 'BBmisc::isFALSE' by 'backports::isFALSE' when loading 'ml
r'
다음의 패키지를 부착합니다: 'mlr'

The following object is masked from 'package:caret':

    train
```

Hide

```
train.data$TARGET <- as.factor(train.data$TARGET)
test.data$TARGET <- as.factor(test.data$TARGET)
# Task 생성
traintask <- makeClassifTask(data = train.data[,-1], target = "TARGET")
testtask <- makeClassifTask(data = test.data[,-1], target = "TARGET")
getParamSet("classif.randomForest")
```

```
                   Type  len   Def   Constr Req Tunable Trafo
ntree             integer    -   500 1 to Inf   -    TRUE     -
mtry              integer    -     - 1 to Inf   -    TRUE     -
replace           logical    -  TRUE        -   -    TRUE     -
classwt     numericvector <NA>     - 0 to Inf   -    TRUE     -
cutoff      numericvector <NA>     -   0 to 1   -    TRUE     -
strata            untyped    -     -        -   -   FALSE     -
sampsize    integervector <NA>     - 1 to Inf   -    TRUE     -
nodesize          integer    -     1 1 to Inf   -    TRUE     -
maxnodes          integer    -     - 1 to Inf   -    TRUE     -
importance        logical    - FALSE        -   -    TRUE     -
localImp          logical    - FALSE        -   -    TRUE     -
proximity         logical    - FALSE        -   -   FALSE     -
oob.prox          logical    -     -        -   Y   FALSE     -
norm.votes        logical    -  TRUE        -   -   FALSE     -
do.trace          logical    - FALSE        -   -   FALSE     -
keep.forest       logical    -  TRUE        -   -   FALSE     -
keep.inbag        logical    - FALSE        -   -   FALSE     -
```

Hide

```
# learner 생성
rf.lrn <- makeLearner("classif.randomForest", predict.type = "response", par.vals = l
ist(ntree = 200, mtry = 3))
```

Hide

```
#set tunable parameters
#grid search to find hyperparameters
# 모델 Tuning
getParamSet(rf.lrn)
```

```
                     Type  len   Def   Constr Req Tunable Trafo
ntree             integer    -   500 1 to Inf   -    TRUE     -
mtry              integer    -     - 1 to Inf   -    TRUE     -
replace           logical    -  TRUE        -   -    TRUE     -
classwt     numericvector <NA>     - 0 to Inf   -    TRUE     -
cutoff      numericvector <NA>     -   0 to 1   -    TRUE     -
strata            untyped    -     -        -   -   FALSE     -
sampsize    integervector <NA>     - 1 to Inf   -    TRUE     -
nodesize          integer    -     1 1 to Inf   -    TRUE     -
maxnodes          integer    -     - 1 to Inf   -    TRUE     -
importance        logical    - FALSE        -   -    TRUE     -
localImp          logical    - FALSE        -   -    TRUE     -
proximity         logical    - FALSE        -   -   FALSE     -
oob.prox          logical    -     -        -   Y   FALSE     -
norm.votes        logical    -  TRUE        -   -   FALSE     -
do.trace          logical    - FALSE        -   -   FALSE     -
keep.forest       logical    -  TRUE        -   -   FALSE     -
keep.inbag        logical    - FALSE        -   -   FALSE     -
```

Hide

```
rf.lrn$par.vals <- list(ntree = 200L,
                        importance=TRUE,
                        cutoff=c(0.80,0.20)
)
# parameter space 지정
params <- makeParamSet(
  makeIntegerParam("mtry",
                   lower = 4,
                   upper = 9),
  makeIntegerParam("nodesize",
                   lower = 30,
                   upper = 50)
)
# Validation strategy 지정
rdesc <- makeResampleDesc("CV",
                          iters=5L)
# Optimization technique 지정
ctrl <- makeTuneControlRandom(maxit = 5L)
```

Hide

```
tune <- tuneParams(learner = rf.lrn
                   ,task = traintask
                   ,resampling = rdesc
                   ,measures = list(acc)
                   ,par.set = params
                   ,control = ctrl
                   ,show.info = T)
```

```
[Tune] Started tuning learner classif.randomForest for parameter set:
            Type len Def   Constr Req Tunable Trafo
mtry     integer   -   -   4 to 9   -    TRUE     -
nodesize integer   -   - 30 to 50   -    TRUE     -
With control class: TuneControlRandom
Imputation value: -0
Exporting objects to slaves for mode socket: .mlr.slave.options
Mapping in parallel: mode = socket; cpus = 4; elements = 5.
[Tune] Result: mtry=4; nodesize=42 : acc.test.mean=0.958
```

Hide

```
# 결과
# [Tune] Started tuning learner classif.randomForest for parameter set:
#             Type len Def   Constr Req Tunable Trafo
# mtry     integer   -   -  2 to 10   -    TRUE     -
# nodesize integer   -   - 10 to 50   -    TRUE     -
# With control class: TuneControlRandom
# Imputation value: -0
# Exporting objects to slaves for mode socket: .mlr.slave.options
# Mapping in parallel: mode = socket; cpus = 4; elements = 5.
# [Tune] Result: mtry=5; nodesize=44 : acc.test.mean=0.96
# [Tune] Started tuning learner classif.randomForest for parameter set:
#             Type len Def   Constr Req Tunable Trafo
# mtry     integer   -   -   4 to 9   -    TRUE     -
# nodesize integer   -   - 30 to 50   -    TRUE     -
# With control class: TuneControlRandom
# Imputation value: -0
# Exporting objects to slaves for mode socket: .mlr.slave.options
# Mapping in parallel: mode = socket; cpus = 4; elements = 5.
# [Tune] Result: mtry=4; nodesize=41 : acc.test.mean=0.958
```

Hide

```
tune
```

```
Tune result:
Op. pars: mtry=4; nodesize=42
acc.test.mean=0.958
```

Hide

```
#using hyperparameters for modeling
rf.tree <- setHyperPars(rf.lrn, par.vals = tune$x)
#train a model
rforest <- train(rf.tree, traintask)
getLearnerModel(rforest)
```

```
Call:
 randomForest(formula = f, data = data, classwt = classwt, cutoff = cutoff,      ntre
e = 200L, importance = TRUE, mtry = 4L, nodesize = 42L)
               Type of random forest: classification
                     Number of trees: 200
No. of variables tried at each split: 4

        OOB estimate of  error rate: 4.29%
Confusion matrix:
      0    1 class.error
0 75691 1066  0.01388798
1  2374 1056  0.69212828
```

2) 튜닝한 모델로 예측하기

Hide

```
#make predictions
rfmodel <- predict(rforest, testtask)
rfmodel2<- data.frame(rfmodel$data)
data.frame(rfmodel)
```

	id	truth	response
	<int>	<fctr>	<fctr>
1	1	0	0
2	2	0	0
3	3	0	0
10	4	0	0
14	5	0	0
16	6	0	0
20	7	0	0
35	8	0	0
37	9	1	1
43	10	0	0

1-10 of 20,046 rows Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
# Parallelization 종료
# parallelStop()
```

3) 예측한 결과 분석하기

Hide

```
tt <- confusionMatrix(data = rfmodel2$response, rfmodel2$truth, positive = "1")
tt
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 18935   587
         1   254   270

               Accuracy : 0.958
                 95% CI : (0.9552, 0.9608)
    No Information Rate : 0.9572
    P-Value [Acc > NIR] : 0.2955

                  Kappa : 0.3706
 Mcnemar's Test P-Value : <0.0000000000000002

            Sensitivity : 0.31505
            Specificity : 0.98676
         Pos Pred Value : 0.51527
         Neg Pred Value : 0.96993
             Prevalence : 0.04275
         Detection Rate : 0.01347
   Detection Prevalence : 0.02614
      Balanced Accuracy : 0.65091

       'Positive' Class : 1
```

Hide

```
TP <- tt$table[4]
TN <- tt$table[1]
FP <- tt$table[2]
FN <- tt$table[3]
recall <- TP/(TP+FN)
pre <- TP / (TP+FP)
f <- 2*(pre*recall)/(pre+recall)
f
```

```
[1] 0.391021
```

튜닝 후 F1 값을 0.3361204에서 0.391021로 높일 수 있었다.

8. SVM 모델 사용하기

1) 모델 적용 전에 데이터 스케일링, 센터링 작업

Hide

```
glimpse(train.data[,col.pred.9])
```

```
Observations: 80,187
Variables: 10
$ PAYM_METD       <fctr> G, G, O, G, O, O, R, O, O, K, K, O, K, K, K,...
$ CRDT_CARD_CNT   <int> 4, 1, 4, 2, 2, 5, 4, 6, 0, 3, 3, 6, 5, 3, 1, ...
$ AGE             <fctr> 35, 45, 45, 40, 60, 40, 55, 40, 25, 25, 55, ...
$ CRDT_OCCR_MDIF  <int> 1, 1, 1, 121, 1, 37, 1, 49, 13, 13, 1, 0, 37,...
$ TOT_LNIF_AMT    <dbl> 15.60744, 16.86008, 18.76428, 16.30050, 14.91...
$ SPTCT_OCCR_MDIF <int> 25, 0, 1, 121, 0, 0, 25, 49, 0, 0, 121, 0, 0,...
$ FMLY_TOT_PREM   <dbl> 15.20181, 15.20181, 15.60727, 17.31202, 15.20...
$ TOT_PREM        <dbl> 15.20181, 15.20181, 13.81551, 16.21341, 15.20...
$ CPT_LNIF_AMT    <dbl> 14.914456, 0.000000, 0.000000, 16.012846, 0.0...
$ TARGET          <fctr> 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,...
```

Hide

```
col.scale <- c("CRDT_CARD_CNT","CRDT_OCCR_MDIF","TOT_LNIF_AMT","SPTCT_OCCR_MDIF",
                  "FMLY_TOT_PREM","TOT_PREM", "CPT_LNIF_AMT")
scale.df <- train.data
scale.df[,col.scale] <- data.frame(scale(scale.df[,col.scale],scale = TRUE, center =
TRUE))
scaled.train.data <- scale.df[,col.pred.9]
glimpse(scaled.train.data)
```

```
Observations: 80,187
Variables: 10
$ PAYM_METD       <fctr> G, G, O, G, O, O, R, O, O, K, K, O, K, K, K,...
$ CRDT_CARD_CNT   <dbl> 0.49873228, -1.14407059, 0.49873228, -0.59646...
$ AGE             <fctr> 35, 45, 45, 40, 60, 40, 55, 40, 25, 25, 55, ...
$ CRDT_OCCR_MDIF  <dbl> -0.5869268, -0.5869268, -0.5869268, 3.5229638...
$ TOT_LNIF_AMT    <dbl> -0.46715559, -0.01448400, 0.67364124, -0.2167...
$ SPTCT_OCCR_MDIF <dbl> 0.38619713, -0.48648671, -0.45157936, 3.73730...
$ FMLY_TOT_PREM   <dbl> -1.03417456, -1.03417456, -0.70952879, 0.6554...
$ TOT_PREM        <dbl> -0.84106249, -0.84106249, -2.03250934, 0.0283...
$ CPT_LNIF_AMT    <dbl> 1.4192695, -0.6790331, -0.6790331, 1.5738011,...
$ TARGET          <fctr> 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,...
```

test 데이터도 마찬가지로 적용

Hide

```
scale.test.df <- test.data
scale.test.df[,col.scale] <- data.frame(scale(scale.test.df[,col.scale],scale = TRUE,
 center = TRUE))
scaled.test.data <- scale.test.df[,col.pred.9]
glimpse(scaled.test.data)
```

```
Observations: 20,046
Variables: 10
$ PAYM_METD       <fctr> O, O, O, O, K, O, O, K, G, O, O, K, O, K, O,...
$ CRDT_CARD_CNT   <dbl> -0.59965412, -0.59965412, 0.48622361, -0.0567...
$ AGE             <fctr> 50, 50, 60, 40, 40, 30, 50, 55, 35, 50, 60, ...
$ CRDT_OCCR_MDIF  <dbl> -0.5876326, -0.6225007, -0.5876326, -0.587632...
$ TOT_LNIF_AMT    <dbl> -0.298987907, 0.050786970, -0.116824629, -0.2...
$ SPTCT_OCCR_MDIF <dbl> -0.4926993, -0.4926993, 0.3739776, -0.4580322...
$ FMLY_TOT_PREM   <dbl> 0.25545189, 0.72434336, -0.22145695, 0.724343...
$ TOT_PREM        <dbl> 0.53882315, -0.35541530, 0.02958560, 1.039499...
$ CPT_LNIF_AMT    <dbl> -0.6791395, -0.6791395, 1.4262056, 1.5812558,...
$ TARGET          <fctr> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,...
```

2) SVM 모델 적용 후 결과 분석

Hide

```
library(mlr)
traintask <- makeClassifTask(data = scaled.train.data, target = "TARGET")
testtask <- makeClassifTask(data = scaled.test.data, target = "TARGET")
#load svm
getParamSet("classif.ksvm")
```

```
                     Type  len    Def
scaled              logical    -   TRUE
type               discrete    -  C-svc
kernel             discrete    - rbfdot
C                   numeric    -      1
nu                  numeric    -    0.2
epsilon             numeric    -    0.1
sigma               numeric    -      -
degree              integer    -      3
scale               numeric    -      1
offset              numeric    -      1
order               integer    -      1
tol                 numeric    -  0.001
shrinking           logical    -   TRUE
class.weights numericvector <NA>      -
fit                 logical    -   TRUE
cache               integer    -     40
                                                 Constr Req Tunable Trafo
scaled                                                -   -    TRUE     -
type              C-svc,nu-svc,C-bsvc,spoc-svc,kbb-svc   -    TRUE     -
kernel        vanilladot,polydot,rbfdot,tanhdot,lap...   -    TRUE     -
C                                           0 to Inf   Y    TRUE     -
nu                                          0 to Inf   Y    TRUE     -
epsilon                                   -Inf to Inf   Y    TRUE     -
sigma                                       0 to Inf   Y    TRUE     -
degree                                      1 to Inf   Y    TRUE     -
scale                                       0 to Inf   Y    TRUE     -
offset                                    -Inf to Inf   Y    TRUE     -
order                                     -Inf to Inf   Y    TRUE     -
tol                                         0 to Inf   -    TRUE     -
shrinking                                             -   -    TRUE     -
class.weights                               0 to Inf   -    TRUE     -
fit                                                   -   -   FALSE     -
cache                                       1 to Inf   -    TRUE     -
```

Hide

```
ksvm <- makeLearner("classif.ksvm", predict.type = "response")
#Set parameters
pssvm <- makeParamSet(
makeDiscreteParam("C", values = 2^(-2:2)), # cost
makeDiscreteParam("sigma", values = 2^(-2:2)) # Kernel
)
#specify search function
ctrl <- makeTuneControlGrid()
#tune model
res <- tuneParams(ksvm, task = traintask, resampling = rdesc, par.set = pssvm, contro
l = ctrl,measures = acc)
```

```
[Tune] Started tuning learner classif.ksvm for parameter set:
          Type len Def         Constr Req Tunable Trafo
C     discrete   -   - 0.25,0.5,1,2,4   -    TRUE     -
sigma discrete   -   - 0.25,0.5,1,2,4   -    TRUE     -
With control class: TuneControlGrid
Imputation value: -0
Exporting objects to slaves for mode socket: .mlr.slave.options
Mapping in parallel: mode = socket; cpus = 4; elements = 25.
```

결과가 랜덤포레스트 모델에 비해 좋지않다.그래서 최종 모형을 랜덤포레스트로 정함.

랜덤포레스트 모델파라미터 조정 , 튜닝과정을 통해 성능을 개선하도록 한다.

---

7. 튜닝한 랜덤포레스트로 최종 예측 후 제출

Hide

```
test.df$LAST_CHLD_AGE[test.df$LAST_CHLD_AGE == NA] <- 0
rfmodel1 <- predict(rforest, newdata = test.df[,-1])
rfmodel2<- data.frame(rfmodel1$data)
write.csv(rfmodel2, "submit.csv")
```

8. 결과

제출결과 본선진출 X

진출팀 F1 score는 0.4 ~ 0.5

고려해봐야할 것

- up sampling, down sampling을 통한 데이터 Balance맞추기
- Stacking 시도