신한은행 SHINHAN BANK

2018 빅콘테스트
2018 BIG CONTEST

빅데이터 분석 경진대회
2018 빅콘테스트[금융 분야]

나의 금융생활정보 지수

TiMIm
박채영 이채은 전지현 조윤영

TiMIm

① 한글로 읽으면 팀힘!
② Too Much Information(TMI), 많은 데이터를 다루는 팀

고려대학교 통계학과

박채영

전지현

조윤영

이채은

Table of Contents

Introduction

Introduction

1 Raw Data 결측값 채우기

약 1만7천명의 설문조사 Data의
금융거래정보에 존재하는
결측값 채우기

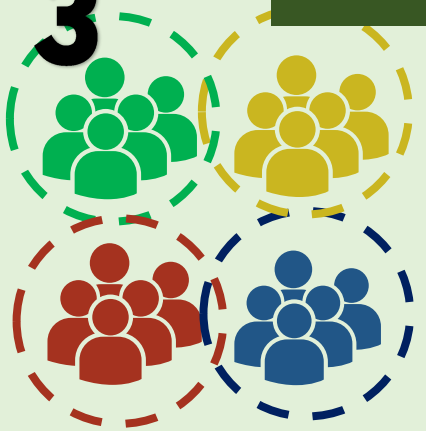
"Predictive Mean Matching(PMM) Imputation"

2 ①Data Set 완성

고객기본 정보를 모두 조합한
약 14만개 고객유형의
금융 거래 정보 예측하기

"Extreme Gradient Boosting(Xgboost)"

3 Peer Group



고객 유형을 유사한 집단으로
묶은 후 고객의 금융점수 제시

"Self Organizing Map(SOM)"

4 개인정보 최소화

상담시스템에 필요한
고객 기본 정보 최소화 하기

"Extreme Gradient Boosting(Xgboost)"

데이터 전처리

데이터 이름 정리

raw_man7	결측값이 존재하는 17,076개의 제공데이터 - DWX_BIGDATA_PEER2018	**raw_sip4 / sip4**	약 14만 개의 고객 조합으로 이루어진 데이터 - ①Data Set
man7	Raw data의 결측 값이 채워진 데이터	**Full**	Data Set의 금융정보까지 모두 채운 완성된 데이터

열 이름 동일화

```
> colnames(sip4)
 [1] "PEER_NO"        "SEX_GBN"        "AGE_GBN"        "JOB_GBN"        "ADD_GBN"        "INCOME_GBN"
 [7] "MARRY_Y"        "DOUBLE_IN"      "NUMCHILD"       "TOT_ASSET"      "ASS_FIN"        "ASS_REAL"
[13] "ASS_ETC"        "M_TOT_SAVING"   "M_JEOK"         "CHUNG_Y"        "M_FUND_STOCK"   "M_FUND"
[19] "M_STOCK"        "M_SAVING.INSUR" "M_CHUNG"        "TOT_DEBT"       "D_SHINYONG"     "D_DAMBO"
[25] "D_JUTEAKDAMBO"  "D_JEONSEA"      "RETIRE_NEED"    "FOR_RETIRE"     "TOT_YEA"        "TOT_JEOK"
[31] "TOT_CHUNG"      "TOT_FUND"       "TOT_ELS_ETE"    "TOT_SOBI"       "M_CRD_SPD"
> colnames(man7)
 [1] "X"              "idx"            "SEX_GBN"        "AGE_GBN"        "JOB_GBN"        "ADD_GBN"
 [7] "INCOME_GBN"     "MARRY_Y"        "DOUBLE_IN"      "NUMCHILD"       "TOT_ASSET"      "ASS_FIN"
[13] "ASS_REAL"       "ASS_ETC"        "M_TOT_SAVING"   "M_JEOK"         "CHUNG_Y"        "M_FUND_STOCK"
[19] "M_FUND"         "M_STOCK"        "M_SAVING_INSUR" "M_CHUNG"        "TOT_DEBT"       "D_SHINYONG"
[25] "D_DAMBO"        "D_JUTEAKDAMBO"  "D_JEONSEA"      "RETIRE_NEED"    "FOR_RETIRE"     "TOT_YEA"
[31] "TOT_JEOK"       "TOT_CHUNG"      "TOT_FUND"       "TOT_ELS_ETE"    "TOT_SOBI"       "M_CRD_SPD"
```

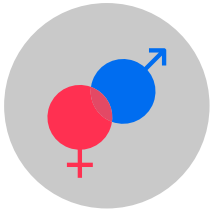
Raw Data와 Data Set의
월 저축액_저축성 보험의 명칭이
달라서 M_SAVING.INSUR을
M_SAVING_INSUR로 변경

데이터 전처리

FACTORIZATION

범주형 변수 직업 구분, 지역 구분, 결혼여부, 성별 맞벌이여부, 자녀 수, 청약보유여부

순서형 변수 연령, 가구소득구간



분석의 용의성을 위해 성별의 범주 레벨은 0과 1로 제한

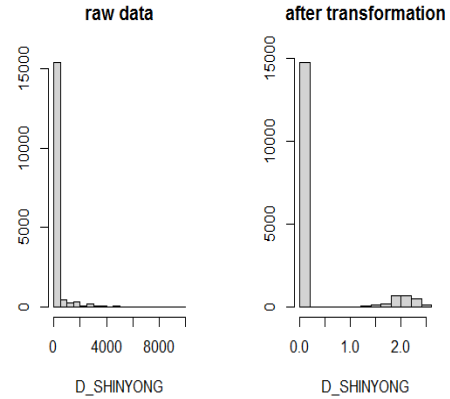
남성(1)

여성(2) → 여성(0)으로 처리

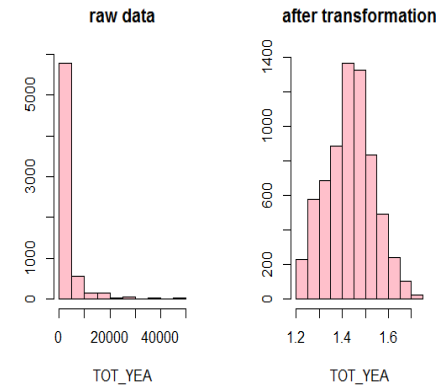
NORMALIZATION

Log 변환을 위해 0을 0.0000000001로 변경 후 0이 너무 많은 변수들을 제외한 나머지 변수들은 정규화 시행

정규화 시행 X



정규화 시행



총자산$^{0.25}$

금융자산$^{0.2}$

부동산자산$^{0.45}$

기타 자산$^{0.4}$

월총저축액$^{0.3}$

월 저축액_적금$^{0.2}$

은퇴 후 필요자금$^{0.09}$

금융상품 잔액_정기예금$^{0.05}$

금융상품 잔액_펀드$^{0.05}$

월총소비금액$^{0.1}$

log(금융상품 잔액_DLS/ELS/ETF 등)

제공데이터 결측값 채우기

① 결측값 확인

맞벌이여부
(DOUBLE_IN)
6524개

은퇴 후 필요자금
(RETIRE_NEED)
11736개

청약 잔액
(TOT_CHUNG)
9550개

자녀수
(NUMCHILD)
7158개

정기예금 잔액
(TOT_YEA)
10311개

펀드 잔액
(TOT_FUND)
14237개

ELS/DLS/ETF 잔액
(TOT_ELS_ETE)
16473 개

적금 잔액
(TOT_JEOK)
8877개

청약보유여부
(CHUNG_Y)
9550개

② 개인 정보 결측값 처리

데이터 설명에 따르면, 미혼 고객은 맞벌이 여부/자녀수에 응답을 하지 않았고 기혼 고객은 맞벌이 여부/자녀 수 모두에 의무적으로 응답을 해야 함.

맞벌이 여부 항목이 결측인 6524명 모두 미혼 고객

→ 미응답(3)으로 처리

자녀 수 항목이 결측인 7158명 중

미혼 고객 6524명 → 미응답(4)으로 처리

기혼 고객 634명 → 자녀 없음(0)으로 처리

③ 청약 관련 변수들 처리

데이터 설명에 따라, 청약 보유 여부의 결측값들은 결측이 아닌 미보유

NULL → 미보유(0)로 처리

분석의 용의성을 위해 청약 보유 여부의 범주 레벨은 0과 1로 제한

보유(5) → 보유(1)로 처리

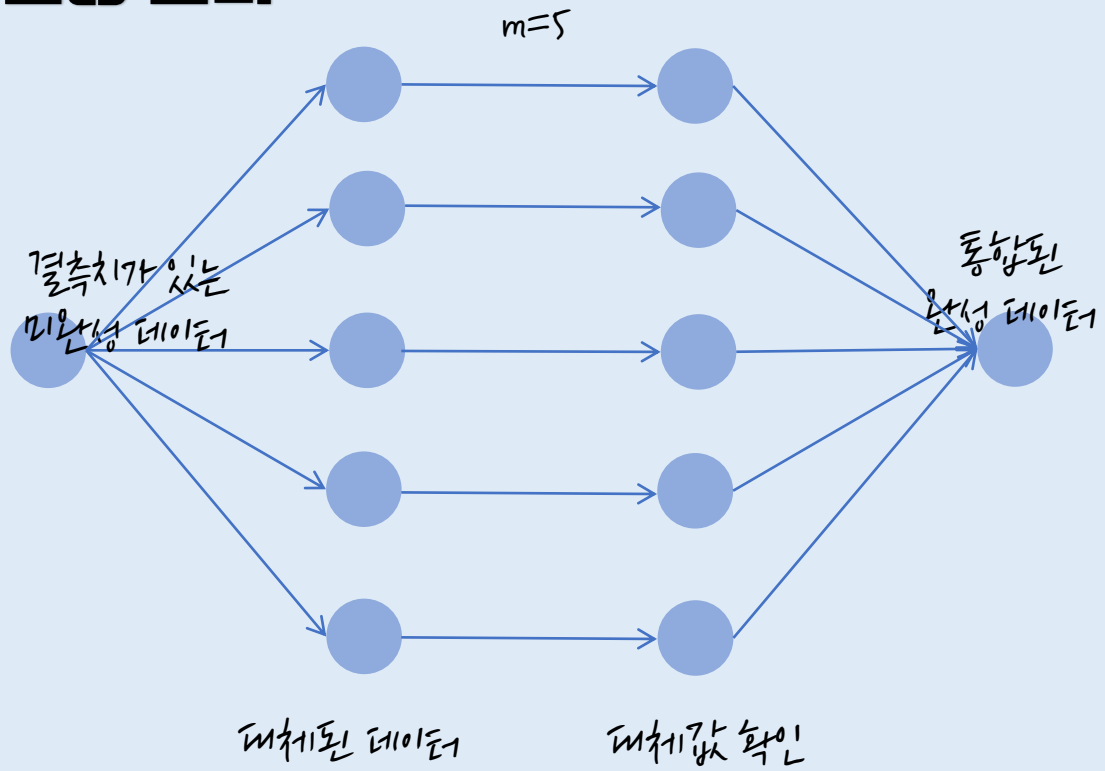
청약 잔액이 결측값인 고객들은 전부 청약을 미보유

결측값 → 0원으로 처리

④ Predictive Mean Matching(PMM)

개념 설명

진행 원리



Why Multiple Imputation(MI)?

- 결측값을 여러 번 예측하고 결과값들을 통합하여 대체하는 방식
- 대체 결과가 하나만 나오는 Single Imputation보다 정확도가 높기 때문에 택함

WHY PMM?

- PMM: MI의 한 기법
- 모형가정에 상대적으로 영향을 적게 받고 수치형 변수 예측에 적합한 Predictive Mean Matching을 채택

④ Predictive Mean Matching(PMM)

Procedure

진행 순서

결측값이 적은 순서:

1) 적금 잔액(TOT_JEOK)
2) 정기 예금 잔액(TOT_YEA)
3)은퇴 필요 자금(RETIRE_NEED)
4) 펀드 잔액(TOT_FUND)
5) ELS/DLS/ETF 잔액(TOT_ELS_ETE)

진행 방법

0. 결측값이 적은 순서대로 대체
1. 결측값이 있는 변수들 + 높은 상관관계가 있는 변수들(부동산 자산, 월 저축액_펀드/주식, 부채 잔액_담보대출, 부채 잔액_아파트/주택 담보 대출, 월평균 카드사용금액)을 제외한 나머지 변수들을 모두 이용해서 PMM으로 5개의 결측 대체값 생성
2. 만들어진 5개의 대체값의 평균으로 데이터 셋의 결측값에 대체
3. 대체 후 추정된 값들에 음수 존재여부와 기존 데이터와의 유사성을 산점도를 통해 확인

제공데이터 결측값 채우기

EXAMPLE

Raw data에서 idx, 상관관계가 높은 변수, 결측값이 존재하는 변수 제외

```
imp.totjeok<- mice(raw_man7[,-c(1,12,17,24,25,27,29,32,33,35)], m=5, maxit=30, seed=831)
```

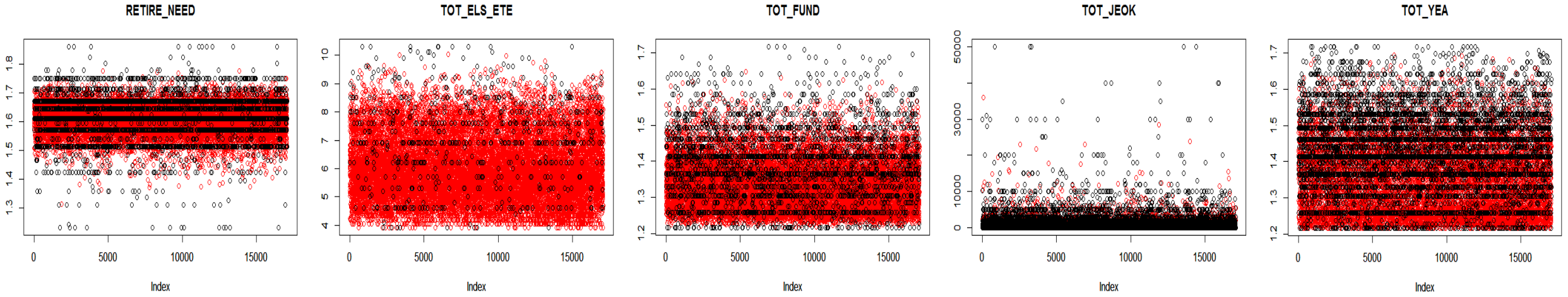
다섯개의 결측 대체값 생성

```
raw_man7[,30] <- apply(data.frame(complete(imp.totjeok,1)$TOT_JEOK,
complete(imp.totjeok,2)$TOT_JEOK, complete(imp.totjeok,3)$TOT_JEOK,
complete(imp.totjeok,4)$TOT_JEOK, complete(imp.totjeok,5)$TOT_JEOK), 1, mean)
```

평균으로 최종 결측값 대체

RESULTS

● 제공된 데이터값 ● 예측된 결측값



Data Set 완성

① 문제 확인

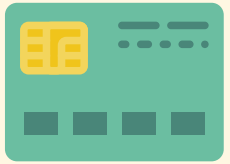
8가지 고객 기본 정보를 바탕으로 (141,750가지 조합)

- 성별: 남성 / 여성
- 연령: 20대 / 30대 / 40대 / 50대 / 60대
- 직업 구분: 사무직 / 공무원 / 전문직(근로직) / 전문직(자영업) / 판매서비스 / 기능직 / 일반자영업 / 프리랜서 / 대학원생 / 기타
- 지역 구분: 강남3구 / 강남(기타) / 광역시 / 경기도 / 기타
- 가구 소득구간: ~99만원 / 100~199만원 / 200~299만원 / 300~399만원 / 400~499만원 / 500~699만원 / 700만원~
- 결혼여부: 미혼 / 기혼
- 맞벌이: 외벌이 / 맞벌이
- 자녀수: 없음 / 1명 / 2명 / 3명이상

26가지 금융 정보 예측

: 총 자산, 금융자산, 부동산자산, 기타자산, 월총저축액, 월총저축액(적금), 청약보유여부, 월저축액_펀드/주식, 월저축액_펀드, 월저축액_주식, 월저축액_저축성보험, 월저축액_청약, 부채 잔액, 부채 잔액_신용대출, 부채 잔액_담보대출, 부채 잔액 _전세자금 대출부채 잔액_아파트/주택 담보대출, 월총소비금액, 은퇴후 필요자금, 노후자금용 월저축액, 금융상품 잔액_정기예금, 금융상품 잔액_펀드, 금융상품 잔액_적금, 금융상품 잔액_청약, 금융상품 잔액_ELS/DLS/ETF, 월평균카드사용금액

② 월평균카드사용금액 예측



데이터 설명에 따르면, 월평균카드사용금액은 **성별/연령/지역구분에 따른** 개인카드(신용+체크) 사용금액

성별	×	연령	×	지역 구분
여성		20대		강남3구
		30대		강남(기타)
		40대		광역시
남성		50대		경기도
		60대		기타

50가지 조합의 개인카드 사용금액

예)

강남3구에 사는 20대 여성 1,941,062(만원)

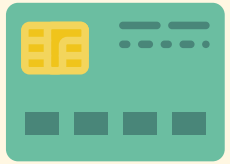
광역시에 사는 30대 남성 3,232,276(만원)

설정한 조건(성별, 연령, 지역구분)이 일치하는 값들로 데이터를 채움

제공 데이터의 월평균 카드 사용금액에 따라 left join함수로

새로운 Data Set의 월평균 카드 사용금액 채우기

② 월평균카드사용금액 예측



과정 코드

```
library("dplyr")
Temp_man7_2 <- man7[,c(2,3,5,35)]
```
Raw Data의 [성별, 연령, 주소, 월평균 카드사용금액]열만 추출

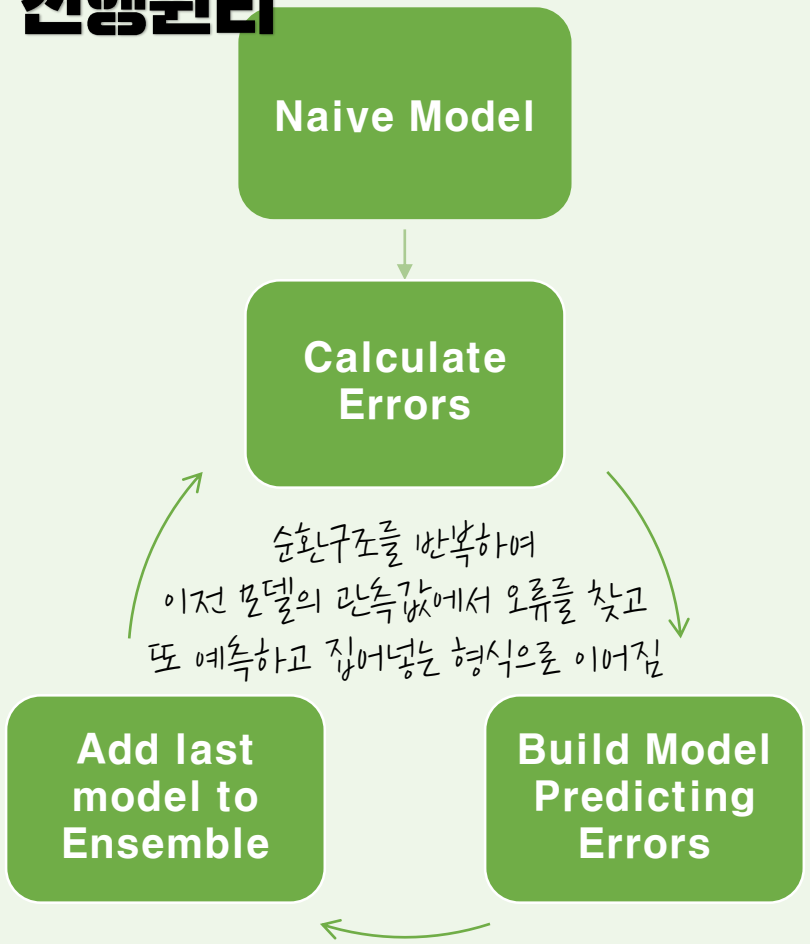
```
unique <- unique(temp_man7_2)
```
중복 없이 서로 다른 50가지 조합만 생성

```
z <- left_join(sip4,unique,by=c("SEX_GBN"="SEX_GBN","AGE_GBN"="AGE_GBN",
                                "ADD_GBN"="ADD_GBN"),all=TRUE)
sip4$M_CRD_SPD<-z$M_CRD_SPD.y
```
성별, 연령, 주소가 모두 일치하는 Data Set의 조합에 따라 월평균 카드 사용금액 채워넣기

```
> sum(is.na(sip4$M_CRD_SPD))
[1] 0
> summary(sip4$M_CRD_SPD)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1316000 1739000 2706000 2658000 3232000 5297000
> unique(sip4$M_CRD_SPD)
 [1] 1437457 1369899 1316087 1437017 1405288 1941062 1741465 1693477 1741706 1682269 3993789 3211358 3232276 3414512 3146348
[16] 3594431 2932871 2769733 2929480 2641576 5296554 3615840 3487272 3809610 3380323 4361528 3198903 2835021 3146000 2639625
[31] 4970759 3038873 2869737 3106659 2860741 3730941 2568835 2274012 2624100 2188927 3425282 1830279 1645889 1978722 1656130
[46] 2424907 1687950 1383457 1739355 1482185
```
결측값을 50가지 조합으로 모두 채움

③ Xgboost

진행원리



- Maching Learning의 기법
- 2016년도 Kaggle에서 우승한 기법으로 처음 등장
- 유용성을 널리 인정 받아 현재 다수의 Kaggle 참가자들이 사용

특징

- 빠르고 유연
- 최적의 결과를 위해 가장 중요한 것은 “Model Tuning”
  → Tuning 과정을 통해 과적합의 위험 감소
- 일반적인 트리는 분류에만 초점을 둔다면, Xgboost는 트리를 만들 때 CART 앙상블 모델을 사용 후 트리 부스팅을 사용해서 가중치를 최적화
  → 같은 분류 결과를 갖는 모델끼리도 모델의 우위 비교 가능

③ Xgboost

Model Tuning Parameters

최적의 결과를 얻기 위한 가장 중요한 과정

objective

목적함수;
이 프로젝트에서 범주형 변수 청약보유여부(CHUNG_Y)는 binary:logistic, 나머지는 트리를 사용하는 reg:linear를 사용.

nround

Boosting 횟수;
기본을 500으로 설정하고 조정했음

max_depth

트리 하나의 최대 깊이;
커지면 모델의 복잡도가 커짐. 기본값은 6.

gamma

트리의 깊이에 영향을 주는 모수;
클수록 보수적인 모델을 만듦. 기본값은 0.

colsample_bytree

각 트리를 설정할 때 열에서 표본을 추출할 비율;
표본 추출은 부스팅마다 한번씩 설정

subsample

트리를 뻗기 전에 train data로 표본을 추출하는 비율;
모든 데이터를 사용하지 않아 과적합을 방지해줌. 0 초과 1 미만

min_child_weight

자식노드에서 필요한 최소 가중치;
가중치가 더 작아지면 분할 과정을 멈춘다. 커질수록 보수적

eta

학습 속도(rate);
트리 스텝이 많으면 과적합이 일어날 수 있기에 부스팅 스텝마다 가중치를 줘서 부스팅 과정에 과적합이 일어나지 않도록 함.

③ Xgboost

① 회귀 모델의 R^2값을 기준으로 금융 정보 변수들 중 개인 정보가 가장 잘 설명하는 변수를 선택
② 1단계에서 선택한 변수를 독립 변수에 포함해서 같은 방식으로 다음 변수를 선택
③ 2단계의 과정을 반복해서 각 변수에 알맞은 회귀 모델을 선택해서 예측 순서와 영향력 높은 변수들을 정하기

```
rsq <- vector()   채워야 하는 금융 정보 중 연속형 변수들
for (i in c(11:16,18:34)){                          8가지 개인 정보와의 회귀식
  lm <-     lm(man7[,i]~SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+INCOME_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD,
                        data=man7)
  sum<-summary(lm)
  cont <- sum$adj.r.squared   각 회귀 모델의 설명력을 대표하는 Adjusted R squared 값
  rsq<-c(rsq,cont)
}
```

```
glm <-glm(man7$CHUNG_Y~SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+INCOME_GBN+MARRY_Y+DOUBLE_IN
                                +NUMCHILD, data=man7, family="binomial")
glm2 <- glm(man7$CHUNG_Y~1, data=full, family="binomial")
1-logLik(glm)/logLik(glm2)
```

청약 보유여부는 범주형 변수이기 때문에
logistic 회귀로 따로 모델 구하기

③ Xgboost

모델:
변수 ~ SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN

변수	R^2	변수	R^2	변수	R^2
TOT_ASSET	0.512	M_STOCK	0.012	RETIRE_NEED	0.457
ASS_FIN	0.272	M_SAVING INSUR	0.117	FOR_RETIRE	0.121
ASS_REAL	0.478	M_CHUNG	0.028	TOT_YEA	0.18
ASS_ETC	0.13	TOT_DEBT	0.095	TOT_JEOK	0.097
M_TOT_SAVING	0.319	D_SHINYONG	0.017	TOT_CHUNG	0.038
M_JEOK	0.086	D_DAMBO	0.082	TOT_FUND	0.285
M_FUND_STOCK	0.036	D_JUTEAKDAMBO	0.08	TOT_ELS_ETE	0.355
M_FUND	0.031	D_JEONSEA	0.025	TOT_SOBI	0.524
CHUNG_Y	0.038				

R^2이 가장 높은 변수를 먼저 선택

③ Xgboost


```
rsq <- vector()
for (i in c(11:16,18:33)){
  lm <-   lm(full[,i]~SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+INCOME_GBN+MARRY_Y
                      +DOUBLE_IN+NUMCHILD+TOT_SOBI, data=full)
  sum<-summary(lm)
  cont <- sum$adj.r.squared
  rsq<-c(rsq,cont)
}
```

채워야 하는 금융 정보 중 연속형 변수들 중 이미 예측한 TOT_SOBI 제외

8가지 개인 정보와 앞에서 예측한 TOT_SOBI의 회귀식

각 회귀 모델의 설명력을 대표하는 Adjusted R squared 값

```
glm <-
glm(full$CHUNG_Y~SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+INCOME_GBN+MARRY_Y
          +DOUBLE_IN+NUMCHILD+TOT_SOBI, data=full, family="binomial")
glm2 <- glm(full$CHUNG_Y~1, data=full, family="binomial")
1-logLik(glm)/logLik(glm2)
```

청약 보유여부는 범주형 변수이기 때문에 logistic 회귀로 따로 모델 구하기

③ Xgboost

모델:

변수 ~ SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI

R^2이 가장 높은 변수를 먼저 선택

변수	R^2	변수	R^2	변수	R^2
TOT_ASSET	0.5	M_STOCK	0.010	RETIRE_NEED	0.463
ASS_FIN	0.266	M_SAVING INSUR	0.095	FOR_RETIRE	0.093
ASS_REAL	0.464	M_CHUNG	0.026	TOT_YEA	0.175
ASS_ETC	0.122	TOT_DEBT	0.088	TOT_JEOK	0.073
M_TOT_SAVING	0.301	D_SHINYONG	0.013	TOT_CHUNG	0.033
M_JEOK	0.082	D_DAMBO	0.073	TOT_FUND	0.267
M_FUND_STOCK	0.029	D_JUTEAKDAMBO	0.07	TOT_ELS_ETE	0.317
M_FUND	0.024	D_JEONSEA	0.017	TOT_SOBI 제외	
CHUNG_Y	0.039				

③ Xgboost

순서	변수	R^2	Formula
1	TOT_SOBI	0.523268	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN
2	TOT_ASSET	0.509284	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+**TOT_SOBI**
3	ASS_REAL	0.868068	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+**TOT_ASSET**
4	RETIRE_NEED	0.478763	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET
5	TOT_ELS_ETE	0.439301	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET +**RETIRE_NEED**
6	ASS_FIN	0.758277	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET +RETIRE_NEED+**TOT_ELS_ETE**
7	ASS_ETC	**예외 ASS_ETC=TOT_ASSET-ASS_FIN-ASS_REAL**	
8	TOT_YEA	0.796608	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET +RETIRE_NEED+TOT_ELS_ETE+**ASS_FIN**
9	TOT_FUND	0.729014	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET +RETIRE_NEED+TOT_ELS_ETE+ASS_FIN+**TOT_YEA**

③ Xgboost

순서	변수	R^2	Formula
10	M_TOT_SAVING	0.436622	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+**TOT_FUND**
11	M_JEOK	0.359056	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+**M_TOT_SAVING**
12	TOT_JEOK	0.314099	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+**M_JEOK**
13	CHUNG_Y	0.303291	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+**TOT_JEOK**
14	M_SAVING INSUR	0.291614	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+**CHUNG_Y**
15	FOR_RETIRE	0.291464	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+**M_SAVING_INSUR**
16	M_CHUNG	0.265672	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+**FOR_RETIRE**
17	TOT_CHUNG	0.275504	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE +**M_CHUNG**
18	TOT_DEBT	0.196211	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE +M_CHUNG+**TOT_CHUNG**

③ Xgboost

순서	변수	R^2	Formula
19	*D_DAMBO*	0.853678	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+**TOT_DEBT**
20	*D_JUTEAKDAMBO*	0.781047	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT
21	*M_FUND_STOCK*	0.163831	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT
22	M_FUND	0.668222	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT+**M_FUND_STOCK**
23	D_SHINYONG	0.105482	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT+**M_FUND**
24	D_JEONSEA	0.081478	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT+M_FUND+**D_SHINYONG**
25(예외)	M_STOCK	0.422593	SEX_GBN+AGE_GBN+JOB_GBN+ADD_GBN+MARRY_Y+DOUBLE_IN+NUMCHILD+INCOME_GBN+TOT_SOBI+TOT_ASSET+RETIRE_NEED +TOT_ELS_ETE+ASS_FIN+TOT_YEA+TOT_FUND+M_TOT_SAVING+M_JEOK+TOT_JEOK+CHUNG_Y+M_SAVING_INSUR+FOR_RETIRE+M_CHUNG +TOT_CHUNG+TOT_DEBT+**M_FUND_STOCK**

③ Xgboost

사전작업:
- 범주형 변수인 성별, 연령, 직업, 주소, 소득수준, 결혼여부, 맞벌이여부, 자녀 수 변수를 dummy 변수로 만들기
- Dummy 변수까지 포함된 결측값이 채워진 Raw Data와 ①Dataset을 각각 행렬로 만든다

과정코드 – TOT_SOBI의 Xgboost

```
train_data99 <- man7.mat[,c(1:38,63)]
train_labels <- train_data99[,39]
train_data<-train_data99[,-39]

test_data99 <- sip4.mat[,c(1:38,63)]
test_labels <- train_data99[,39]
test_data<-train_data99[,-39]
```

채워진 Raw Data로 만든 행렬 중 개인정보 열들과 TOT_SOBI열 추출

train_labels에는 목표 변수인 TOT_SOBI열을,
train_data에는 나머지 변수들을 지정

③ Xgboost


```
dtrain <- xgb.DMatrix(data = train_data, label= train_labels)
dtest <- xgb.DMatrix(data = test_data, label= test_labels)
```

목표 변수는 label로 설정하고 예측 변수들은 data로 설정해서 train과 test data를 각자의 Dmatrix 객체로 지정

```
model_TOT_SOBI <- xgboost(data = dtrain, subsample = 1, nround = 500, eval_metric = "rmse",
                          eta = 0.025, max_depth = 4, gamma = 0.025, colsample_bytree=0.7,
                          min_child_weight = 0.6, seed=101, objective = "reg:linear")
```

최적 Parameter들을 설정해서 xgboost 모델 만들기

```
pred_TOT_SOBI <- predict(model_TOT_SOBI, dtest)
sip4_1<-sip4
sip4_1[,34]<-pred_TOT_SOBI
```

Parameter Tuning(뒷 슬라이드 참조)을 거친 모델로 데이터 예측

③ Xgboost

작은 이상치 / 높은 이상치

회귀모델보다 높은 R^2을 가진 모델 중 (예측값≤Raw Data의 최소값)또는 (예측값≥Raw Data의 최대값)인 개수의 비율이 낮은 모델을 최선으로 선택한 후 그래프를 확인한다.

예시

Tuning Parameters

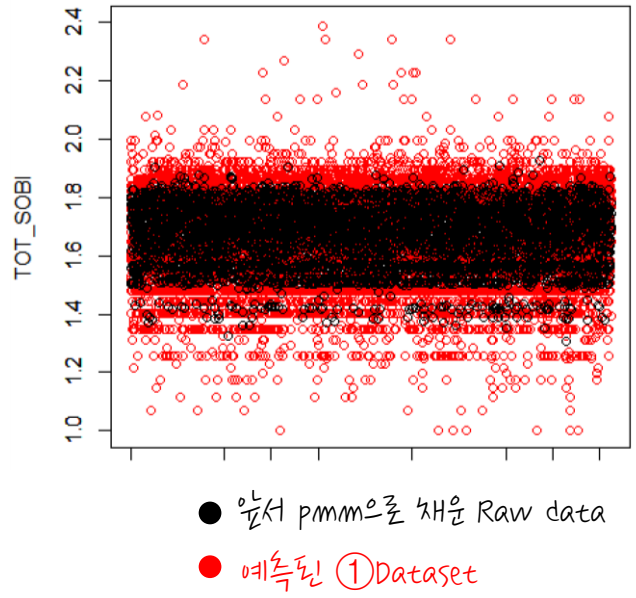
최우선 판단기준 / 두번째 판단 기준

Trials	nround	max_depth	min_child_weight	subsample	colsample	gamma	eta	RMSE	R^2	MAE	작은 이상치 비율	높은 이상치 비율
1	600	8	1	1	1	0	0.025	0.099	0.493	0.073	0.003	0.004
2	500	6	1	1	1	0	0.025	0.096	0.525	0.07	0.005	0.006
3	500	4	1	1	1	0	0.025	0.094	0.538	0.07	0.028	0.019
4	600	4	3	1	1	0	0.025	0.095	0.537	0.07	0.03	0.06
...												
n	500	4	0.6	1	0.7	0.025	0.025	0.094	0.539	0.07	0.028	0.017

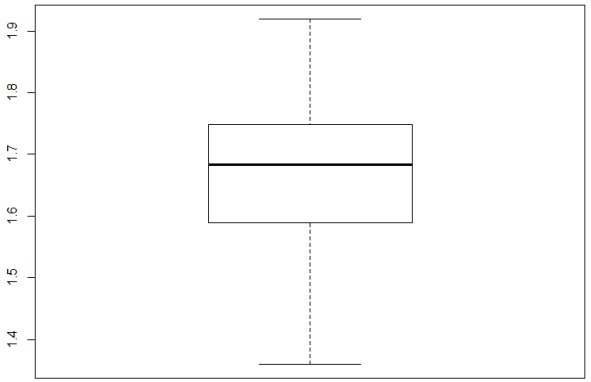
회귀 모델의 R^2 (0.523) 보다 높은 parameter들 중 이상치 비율이 낮은 Parameter 채택

③ Xgboost

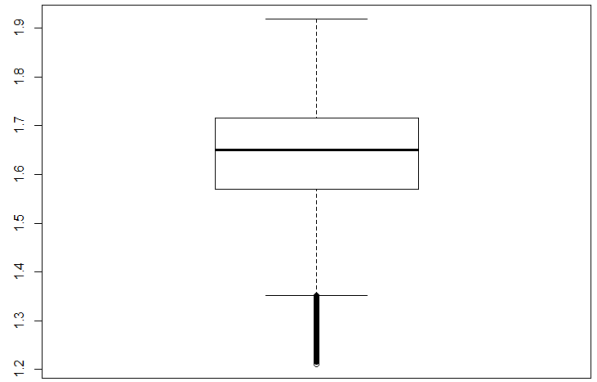
TOT_SOBI Xgboost 결과



PMM으로 채운 Raw Data의 TOT_SOBI



Xgboost로 예측한 전체 데이터의 TOT_SOBI



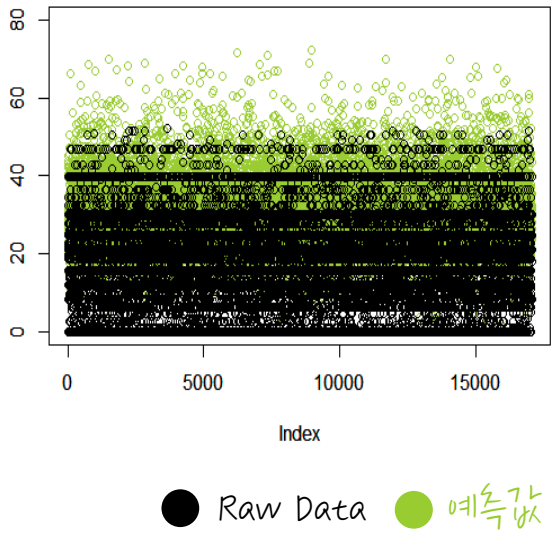
나머지 변수들도 최적의 Parameter을 가진 Xgboost로 채우기

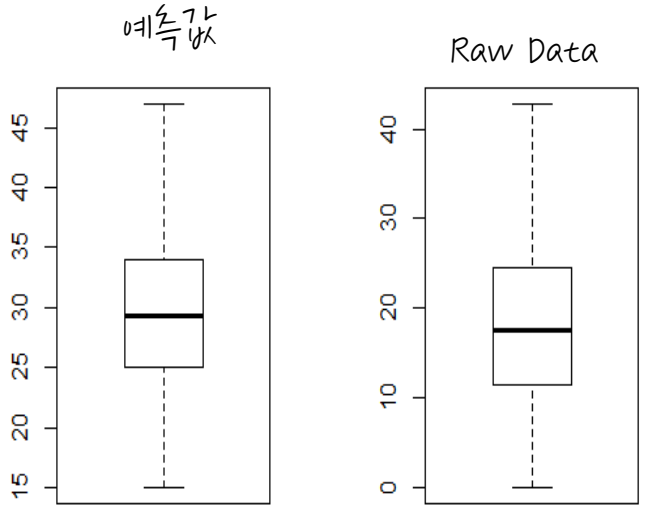
③ Xgboost

Raw Data에서
총 자산(TOT_ASSET) = 금융자산(ASS_FIN) + 부동산 자산(ASS_REAL) + 기타 자산(ASS_ETC),
월 저축액_펀드/주식(M_FUND_STOCK) = 월 저축액_펀드(M_FUND) + 월 저축액_주식(M_STOCK)의
관계가 100% 성립하기 때문에 ①Data Set에서도 동일할 것이라고 추론

ASS_ETC

Xgboost로 구한 ASS_FIN, ASS_REAL, TOT_ASSET의 산술로 ASS_ETC값 계산





M_STOCK

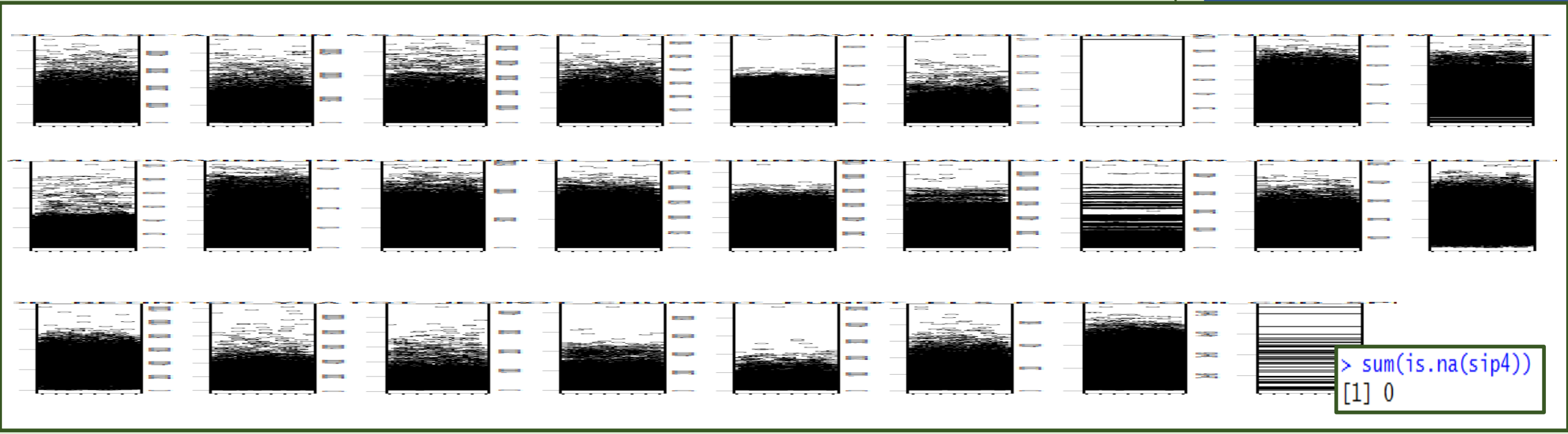
Xgboost로 채운 M_FUND_STOCK의
R squared 값이 약 0.3으로 설명력이 낮아
이 변수의 단순 산술로 다른 변수(M_STOCK)를
예측하는 것은 무리가 있다고 판단

```
> M_FUND_STOCK_TRAIN <- predict(model, dtrain2)
> postResample(M_FUND_STOCK_TRAIN,train_labels2)
     RMSE  Rsquared       MAE
17.315874  0.292941  6.924218
```

③ Xgboost

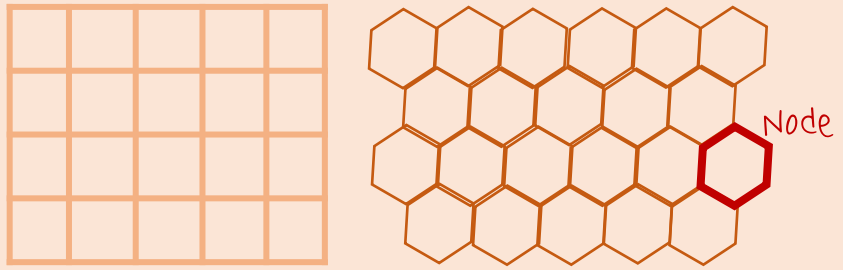
- ✓ 청약보유여부(CHUNG_Y)변수는 0.5보다 크면 1로, 그 외는 0으로 처리
- ✓ 정규화 과정을 거친 변수들은 back transformation을 거쳐 원상복귀
- ✓ 데이터 특성상 예측된 결과값에 음수가 나오기 힘든 변수들이기 때문에 음수는 모두 0으로 대체

완성된 ①Dataset의 금융 변수들의 산점도




Peer Group

Self Organizing Map(SOM)



SOM grid는 형태가 사각형이거나 육각형이다.
각 node은 neuron을 나타냄

- 군집화 분석 방법
- 목표 변수(label)에 대한 **사전 정보 없이** 시행되는 **비지도학습방법**
  : 주어진 입력패턴에 대하여 정확한 해답을 미리 주지 않고 스스로 학습할 수 있는 **자기 조직화 능력**을 이용
- 관측값의 특징에 따라 grid상에 알맞은 node 위에 위치

진행방식

1. 단위의 영향력을 없애는 Scaling
2. K-means 알고리즘으로 클러스터 개수별 Within Sum of Squares 그래프를 보고 꺾이는 지점(elbow)에서 클러스터 개수를 결정
3. 결정된 클러스터 개수에 따라 SOM을 이용하여 집단을 나누기

장점(선택이유)

- 많은 변수들이 있는 큰 데이터셋을 대표성을 가진 작은 차원으로 분류하기에 적합
- 그리드 상에 노드들이 위치하므로 인접한 노드들이 독립적이지 않음
- 클러스터마다 개체수가 유사하지 않아도 됨
- 다양하게 시각화 가능

Self Organizing Map(SOM)

과정 코드

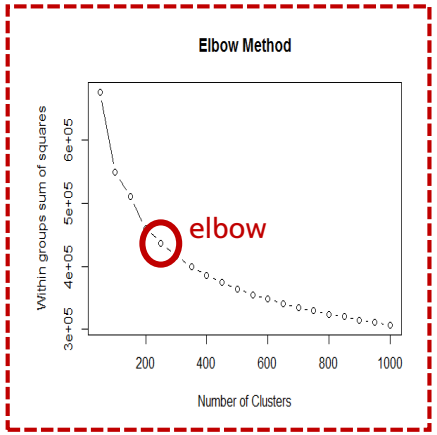
```
sip4_f<-sip4[,-c(1:9,16)]
```

완성된 141,750가지 고객 유형의 데이터 중 청약 보유 여부를 제외한 나머지 금융 정보 변수들만 새로운 데이터로 지정
→ 청약 보유 여부(CHUNG_Y)의 정보는 모두 월 저축액_청약(M_CHUNG) 변수에 포함되기 때문에 제외해도 무방

```
Sip4_f.mat<- scale(sip4_f)
```

```
wss<-vector()
for (i in seq(50,1000,by=50)){set.seed(123)
        wss[i] <- sum(kmeans(sip4_f.mat, centers=i, iter.max=60)$withinss)}
```

60회의 시도 내에 금융 정보 변수들을 각각 비슷한 i개의 집단으로 나누기




```
wss<-wss[!is.na(wss)]
plot(seq(50,1000,by=50), wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares", main="Elbow Method")
```

Self Organizing Map(SOM)

과정 코드

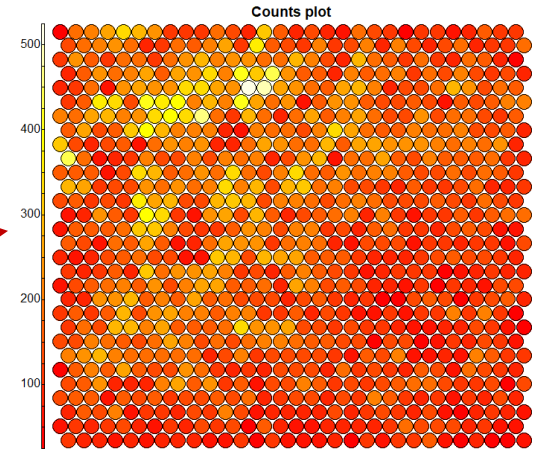
```
som_grid1<-somgrid(xdim=30,ydim=30, topo="hexagonal")
set.seed(101)
som_model1<-som(sip4_f.mat,grid=som_grid1,keep.data=TRUE,
rlen=1000)
```

(x축 30)*(y축30)으로 900개의 육각형 노드로 이루어진 그리드 만들기

1000회 반복해서 그리드 내에 관측값들 배치

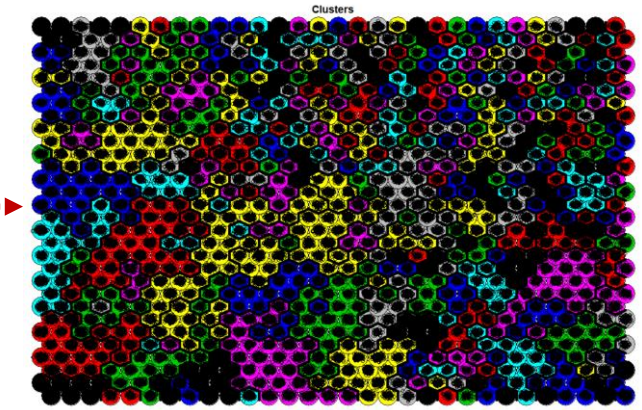
노드별 관측값 개수를 나타내는 그래프

```
plot(som_model1, type="count")
```



250개의 군집으로 그리드를 분류

```
som_cluster1<-cutree(hclust(dist(som_model1$codes[[1]])),250)
plot(som_model1, type="mapping", bgcol=som_cluster1, main="Clusters")
add.cluster.boundaries(som_model1, som_cluster1)
```




```
mapping_table<-sip4_som_mapping[,c(1,36)]
```

매핑 테이블 만들기

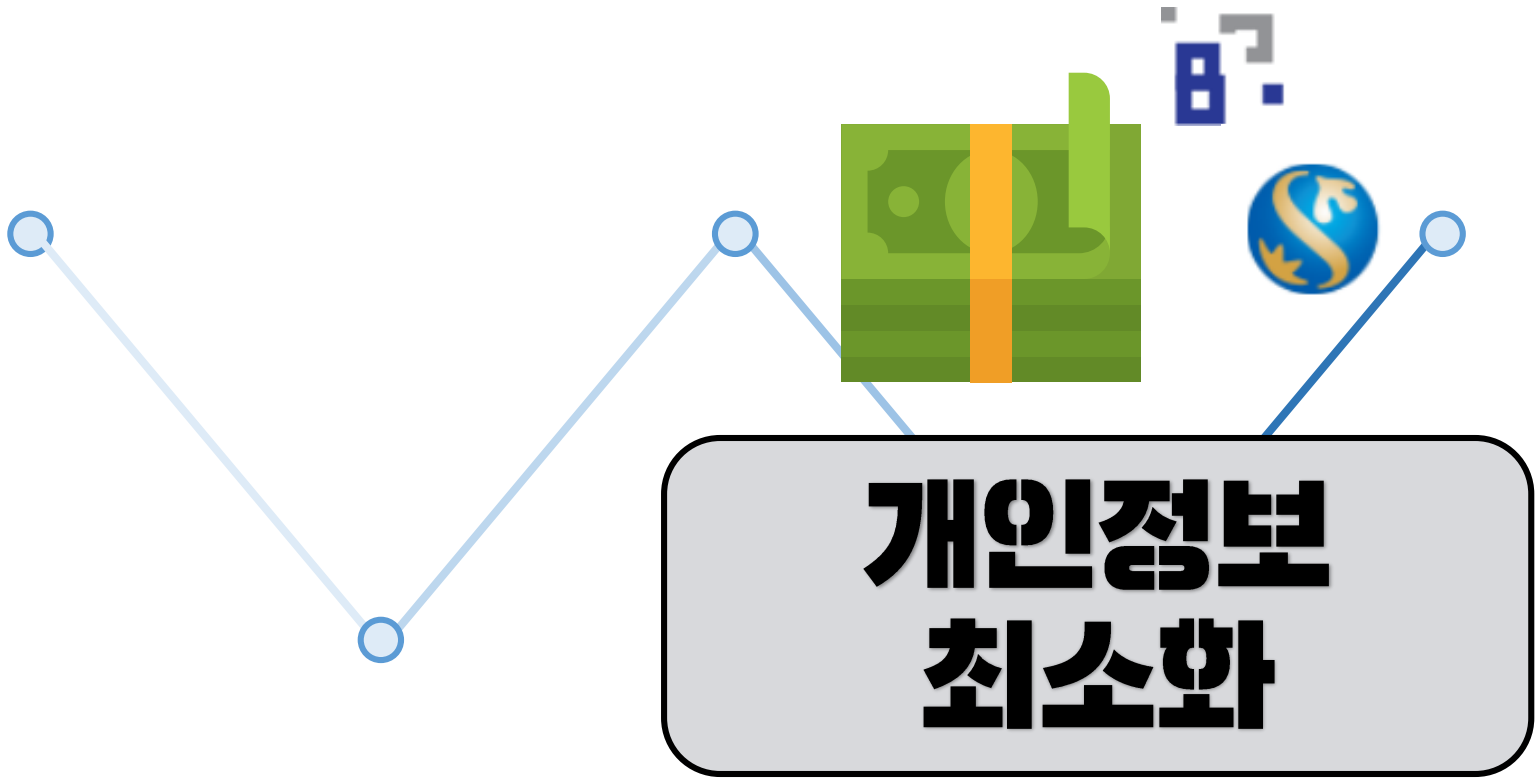
백분위 분포표 만들기

```
final<-data.frame(rep(0,102))
ccc<-data.frame()
for(i in 1:250 ){
  sub<-subset(sip4_som_mapping,group==i)
 quan=as.numeric(quantile(sub$ASS_FIN,seq(0,1,by=0.01)[-1])[1:100])
ccc<-c(i, "ASS_FIN", quan)
  final<-cbind(final,ccc)
}
final<-as.data.frame(t(as.matrix(final[,-1])))
colnames(final)<-c("group","type",paste0("quan",1:100))
rownames(final)<-1:750
```

250개의 그룹의 각 금융자산(ASS_FIN)의 상위 1%부터 100%까지 백분위수를 quan에 저장

M_TOT_SAVING과 TOT_SOBI도 마찬가지 과정으로 백분위 분포표를 만들어서 각 그룹의 백분위수를 차곡차곡 final에 저장

group	type	quan1	quan2	quan3	quan4	quan5	quan6	quan7	quan8	quan9	...	quan96	quan97	quan98	quan99	quan100
1	ASS_FIN	5705.101	5929.677	6070.174	6251.89	6370.483	6495.871	6773.298	6827.566	6895.922	...	10384.24	10407.58	10810.49	11315.04	12247.39
1	M_TOT_SAVING	87.23003	91.95969	98.29162	102.164	104.0214	106.6082	109.1666	115.258	118.5089	...	206.9729	210.2362	215.5553	219.4438	246.8823
1	TOT_SOBI	181.783	194.5974	195.336	196.6772	201.5864	204.7625	207.4416	209.3193	210.5755	...	333.1673	335.2875	336.634	344.7454	462.4243
2	ASS_FIN	4246.066	4480.339	4616.43	4679.896	4848.254	4933.074	5030.835	5046.567	5070.742	...	9663.651	9856.966	9969.191	10053.38	10574.68
2	M_TOT_SAVING	64.89036	70.29248	71.73083	73.29218	74.4988	77.66385	78.91542	80.41799	81.81395	...	185.7291	187.3505	198.5306	211.277	217.2485
2	TOT_SOBI	171.4525	181.5101	184.1318	186.8013	188.6139	189.1103	189.8465	196.7896	197.6262	...	317.1095	317.8919	324.8909	333.5376	356.8724
3	ASS_FIN	3135.016	3212.602	3318.91	3503.507	3565.268	3613.132	3691.142	3744.272	3790.64	...	6838.031	7119.354	7246.892	7455.426	8161.968
3	M_TOT_SAVING	73.03737	76.43846	77.32893	78.00077	79.64055	82.52398	84.30481	85.70831	87.32567	...	192.3993	195.0156	199.4721	206.4241	218.2434
3	TOT_SOBI	133.3189	136.3231	144.509	149.0391	151.6957	153.2611	156.9134	159.0771	160.582	...	338.8825	350.359	358.6268	370.2427	464.6884
4	ASS_FIN	3437.464	3553.402	3638.12	3656.834	3678.568	3727.685	3759.779	3784.237	3800.203	...	6315.236	6476.312	6865.995	7052.105	7079.881

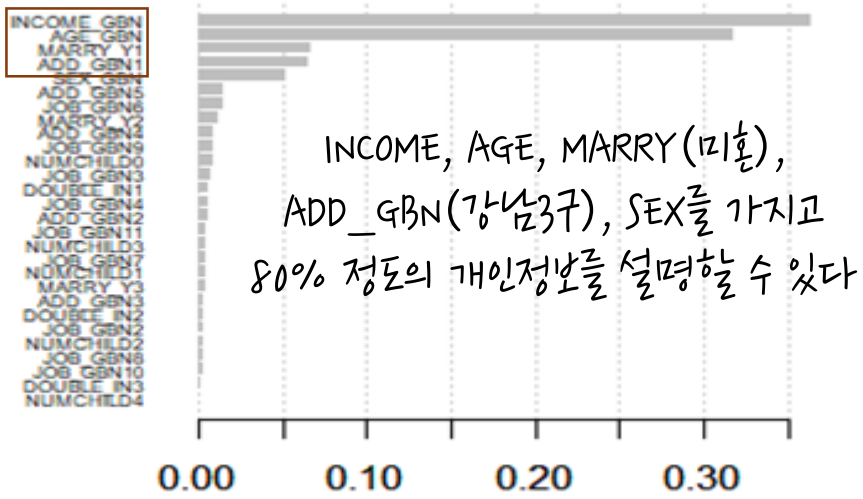


개인정보 최소화

고객 기본정보를 Train Data로 앞서 분류한 군집을 Train Label로 Xgboost 실행

Data set을 채울 때 실행했던 Xgboost는 모든 범주형 변수를 더미로 바꿔주는 one-hot-encoding을 실행했으나,
이번 Xgboost는 고객 기본정보(범주형 변수들)만 가지고 진행되기 때문에
순서형 변수인 소득(INCOME_GBN), 연령(AGE_GBN)을 수치형 변수로 취급

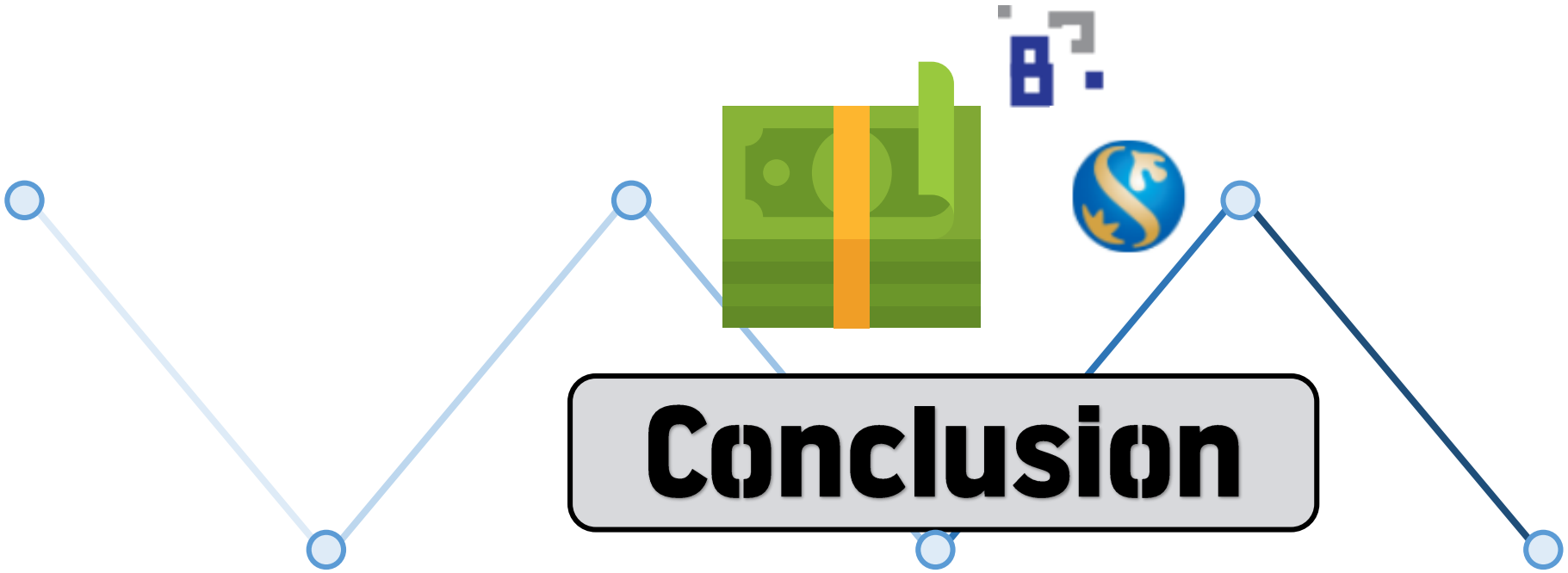
변수 중요도



INCOME, AGE, MARRY(미혼),
ADD_GBN(강남3구), SEX를 가지고
80% 정도의 개인정보를 설명할 수 있다

최소화된 개인정보

소득 | 연령 | 결혼여부 (미혼) | 거주지 (강남3구) | 성별



Conclusion

제출 결과물

(만원)

idx2	성별	연령구분	직업구분	지역구분	가구소득 (구간)	가구소득 (구간)	결혼여부	맞벌이 여부	자녀수	총자산	금융자산	부동산자산	기타자산
1	1	2	2	1	1	1	1	1	0	2,854	878	292	1,684
2	1	2	2	1	1	1	2	1	0	12,616	165	6,841	5,610
3	1	2	2	1	1	1	3	1	0	8,038	74	3,244	4,720
4	1	2	2	1	1	1	1	2	0	2,148	646	309	1,193
5	1	2	2	1	1	1	2	2	0	9,789	85	5,289	4,415
:	:	:	:	:	:	:	:	:	:	:	:	:	:
141,744	0	6	11	5	7	7	1	2	4	67,543	15,634	42,663	9,245
141,745	0	6	11	5	7	7	1	2	4	38,735	11,221	17,033	10,480
141,746	0	6	11	5	7	7	2	2	4	78,724	16,040	60,002	2,682
141,747	0	6	11	5	7	7	3	2	4	66,116	14,781	43,898	7,437
141,748	0	6	11	5	7	7	1	3	4	36,292	8,089	10,972	17,231
141,749	0	6	11	5	7	7	2	3	4	75,457	15,682	53,012	6,764
141,750	0	6	11	5	7	7	3	3	4	62,622	13,473	43,825	5,325

Full
① Dataset

idx2	Peer Group No.
1	105
2	105
3	105
:	:
73,721	58
73,722	80
73,723	79
73,724	243
:	:
141,748	115
141,749	25
141,750	97

Mapping
Table

제출 결과물

(만원)

Peer Group No.	비교대상 칼럼	백분위수1	백분위수2	백분위수3	백분위수4	백분위수5	...	백분위수96	백분위수97	백분위수98	백분위수99	백분위수100
1	금융자산	5705	5930	6070	6252	6370	...	10384	10408	10810	11315	12247
1	월저축금액	87	92	98	102	104	...	207	210	216	219	247
1	월소비금액	182	195	195	197	202	...	333	335	337	345	462
2	금융자산	4246	4480	4616	4680	4848	...	9664	9857	9969	10053	10575
2	월저축금액	65	70	72	73	74	...	186	187	199	211	217
2	월소비금액	171	182	184	187	189	...	317	318	325	334	357
:	:	:	:	:	:	:	:	:	:	:	:	:
250	금융자산	1136	1503	1681	1703	1711	...	3694	3760	3901	4103	4187
250	월저축금액	48	51	52	52	53	...	119	119	121	125	135
250	월소비금액	132	141	146	148	151	...	246	253	255	258	268

Peer Group
금액분포

한계점

제공 데이터가 전체 표본을 대표하지 않을 수도 있음

모집단의 표본인 Raw Data의 개인정보 변수들을 봤을 때 균일한 분포를 가지고 있지 않아서 금융정보의 변수들 역시 편향되어 있다

아마존워크스페이스의 컴퓨팅 파워가 낮음

NbClust를 할 때 요구되는 RAM에 못 미쳐서 실행하지 못하고 대안으로 Elbow Method를 실행했다

제공된 데이터에 비해 예측해야 하는 결측값이 너무 많음

대체값 예측 기법과 무관하게 결측 비율이 높을수록 정확도는 떨어지는데 제공 데이터에서 결측이 90% 이상인 변수가 있었다.

개인정보 변수들이 모두 범주형 변수들임

①Dataset을 채울 때 예측에 사용되는 개인정보 변수들이 모두 범주형이여서 정확도에 한계가 있었다.
소득이나 연령과 같은 변수들은 수치형으로 제공했으면 더 정확한 결과가 나왔을 것 같다.

참고 자료 목록

- https://stats.stackexchange.com/questions/204489/discussion-about-overfit-in-xgboost
- https://brunch.co.kr/@snobberys/137
- https://xgboost.readthedocs.io/en/latest/R-package/discoverYourData.html
- https://xgboost.readthedocs.io/en/latest/R-package/xgboostPresentation.html
- https://xgboost.readthedocs.io/en/latest/parameter.html
- https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/
- https://blog.naver.com/tjdudwo93/221071886633
- https://medium.com/@peteryun/ml-kaggle%EC%97%90-%EC%A0%81%EC%9A%A9%ED%95%B4%EB%B3%B4%EB%8A%94-xgboost-f1650342ba93
- http://hamelg.blogspot.com/2016/09/kaggle-home-price-prediction-tutorial.html
- http://hamelg.blogspot.com/2015/10/introduction-to-r-index.html
- https://www.kaggle.com/dansbecker/xgboost
- https://www.kaggle.com/rtatman/machine-learning-with-xgboost-in-r
- https://www.analyticsvidhya.com/blog/2016/03/tutorial-powerful-packages-imputing-missing-values/
- https://algobeans.com/2017/11/02/self-organizing-map/
- https://www.shanelynn.ie/self-organising-maps-for-customer-segmentation-using-r/
- https://users.ics.aalto.fi/jhollmen/dippa/node14.html#SECTION00521400000000000000
- https://www.kaggle.com/rtatman/machine-learning-with-xgboost-in-r
- https://clarkdatalabs.github.io/soms/SOM_NBA
- https://www.analyticsvidhya.com/blog/2016/01/xgboost-algorithm-easy-steps/
- https://www.kdnuggets.com/2016/03/xgboost-implementing-winningest-Kaggle-algorithm-spark-flink.html
- 외 다수

감사합니다