KMU BIGDATA MBA

DATA MINING Personal Exam

빅데이터 경영 MBA 5기
U2016054 이병준
2016년 10월 29일 23시

CONTENTS INDEX

1. Data

사용 데이터

- 2조 Customer Signatures 중 수업시간에 만들었던 기본 15개 파생변수와 조별 과제에서 생성한 11개 데이터 사용.
  - 기본파생변수 : 환불금액, 환불횟수, 구매상품 다양성, 내점일수와 구매 주기, 내점 당 구매 건수, 주중 주말 구매패턴, 12/6/3개월 구매금액 및 횟수
  - 2조파생변수 : 가격선호도, 시즌선호도, 주구매상품, 휴면/이탈 고객, 주구매 시간대, 선호 할부, 선호 구매파트, 주이용지점, 선호할인율,
- 6조 CS Data "instMonth" 평균 할부기간 데이터 사용.

```
> head(user_tree)
  custid sex   rf_amt rf_cnt buy_brd visits API      NPPV  wk_amt  we_amt   wk_pat    amt12 nop12    amt6 nop6    amt3
1      4   2        0      0       6      4  91 1.500000  254300  127000 weekdays   381300     6  381300    6   89700
2      5   2        0      0       4      3 121 1.333333  129600   26100 weekdays   155700     4  155700    4  155700
3      6   2  -221730      4      14     15  24 1.933333  460900 1556500  weekend  2017400    29  786060   14   18060
4      7   2        0      0      10      7  52 1.571429  767000 1903000  weekend  2670000    11 2670000   11 2670000
5      8   2 -1934341     21      67    114   3 2.131579 6917729 3465820 weekdays 10383549   243 6040055  114 2425630
6      9   2  -820800      6      13      9  40 3.111111  457580  554100     none  1011680    28 1011680   28 1011680
  nop3 pr_pref sea_pref stay_out fav_time fav_paymthd fav_part_mean_amt fav_part_cnt main_store avg_disc fav_part_index
1    2 3-grade   Spring      stay      19시           1          90533.33            3    sinchon       4%              1
2    4 2-grade   Spring       out      19시           1          43200.00            3    sinchon      10%              2
3    3 3-grade     Fall      stay      18시           1          45424.17           12     Chunho       3%              3
4   11 4-grade   Spring       out      19시           3         183860.00            5    sinchon       7%              2
5   47 1-grade   Summer       out      18시           1          64108.94           85    sinchon       6%              2
6   28 1-grade   Spring       out      18시           3          20718.57            7      Muyuk       4%              4
  fav_good_index   PAPV P_group p_trend instCnt instMonth instRatio
1              1  90000       4     new       1       3.0      0.17
2              2  50000       4     new       0       0.0      0.00
3              3 130000       3    keep       9       3.0      0.31
4              4 380000       1     new      10       5.1      0.91
5              5  90000       3 loyalty      62       2.9      0.26
6              6 110000       3     new      21       3.4      0.75
```

2. Data Processing (데이터 전처리)

Library

```r
library(plyr)  # Join 및 데이터 전처리를 위해 호출
library(dplyr) # 데이터 전처리를 위해 호출
library(C50) # C50, caret, ROCR Decision Tree 생성을 위해 호출
library(caret)
library(ROCR)
library(combinat) # 조합을 사용하기 위해 호출
library(gtools) # 순열을 사용하기 위해 호출
library(Epi) # 시각화
```

Processing

```r
#### Signature만들기 ####

user <- read.csv("2_group.csv", stringsAsFactors = T)
user_tree <- user[,c(1,2,37:length(user))]
# 0 : 무효 1 : 남성 2: 여
user_tree <- user_tree[user_tree$sex!=0,] #무효는 제외
user_tree$sex <- factor(user_tree$sex)

#명목형 변수 한글사용 시 Model이 생성되지 않는 점과 Factor화
user_tree$wk_pat <- as.character(user_tree$wk_pat)
user_tree[user_tree$wk_pat=="주말형",]$wk_pat <- "weekend"
user_tree[user_tree$wk_pat=="주중형",]$wk_pat <- "weekdays"
user_tree[user_tree$wk_pat=="유형없음",]$wk_pat <- "none"
user_tree$wk_pat <- factor(user_tree$wk_pat)

user_tree$main_store <- as.character(user_tree$main_store)
user_tree[user_tree$main_store=="신촌점",]$main_store <- "Sinchon"
user_tree[user_tree$main_store=="천호점",]$main_store <- "Chunho"
user_tree[user_tree$main_store=="무역점",]$main_store <- "Muyuk"
user_tree[user_tree$main_store=="본점",]$main_store <- "Bon"
user_tree$main_store <- factor(user_tree$main_store)
```

2. Data Processing (데이터 전처리)

Processing

```
user_tree$group_member <- as.character(user_tree$group_member)
user_tree[user_tree$group_member=="그룹사",]$group_member <- "group"
user_tree[user_tree$group_member=="일반회원",]$group_member <- "normal"
user_tree$group_member <- factor(user_tree$group_member)

user_tree[user_tree$stay_out=="",]$stay_out <- "유지" # 해당사항이 안될 경우 유지로 판단.
user_tree$stay_out <- as.character(user_tree$stay_out)
user_tree[user_tree$stay_out=="이탈/휴면",]$stay_out <- "out"
user_tree[user_tree$stay_out=="유지",]$stay_out <- "stay"
user_tree$stay_out <- factor(user_tree$stay_out)

#선호 할부 개월에서 한글인 개월 삭제.
user_tree$fav_paymthd <- as.character(user_tree$fav_paymthd)
user_tree$fav_paymthd <- gsub(pattern = "개월",replacement = "",user_tree$fav_paymthd)
user_tree$fav_paymthd <- factor(user_tree$fav_paymthd)


# 한글 벡터를 포함하고 있는 데이터 항목들 전체 인덱스화 (우유 : 1, 치즈 : 2와 같은형태)
df <- data.frame(fav_part_index = 1:length(unique(user_tree$fav_part)), fav_part =unique(user_tree$fav_part))
user_tree <- join(user_tree,df,by="fav_part")
user_tree$fav_part_index <- factor(user_tree$fav_part_index)


df2 <- data.frame(fav_good_index = 1:length(unique(user_tree$fav_goodcd)), fav_goodcd =unique(user_tree$fav_goodcd))
user_tree <- join(user_tree,df2,by="fav_goodcd")
user_tree$fav_good_index <- factor(user_tree$fav_good_index)

#사용하면 안되는 항목 제거
user_tree <- user_tree[,-c(12,13,22,26,31)]

# 6조 데이터 Join
cust_6 <- read.csv("6custsig.csv", stringsAsFactors = F)
tmp_cust <- cust_6[,c("custid","PAPV","P_group","p_trend","instCnt","instMonth","instRatio")]
user_tree <- join(user_tree,tmp_cust,by="custid")
```

	fav_part_index	fav_part
1	1	스포츠캐주얼
2	2	패션잡화
3	3	영라이브
4	4	아동,스포츠
5	5	공산품
6	6	명품잡화

	fav_good_index	fav_goodcd
1	1	골프웨어
2	2	수입종합화장품
3	3	영캐쥬얼
4	4	싸롱화
5	5	임대을
6	6	미확인코너

3. User Defined Function

사용자 함수 생성 (자동 독립변수 할당 및 옵션 적용 함수)

- 함수생성이유
  - C5 모델생성의 경우 다양한 방법과 조건, Formula내 변수 설정 등 다각도로 접근이 가능하여 사용자 또는 분석가가 일일이 적용할 때 많은 시간과 노력이 필요하다고 판단되어 사용자 함수를 생성.
  - 파라미터
    - ✓ Train Data : 트레이닝용 데이터
    - ✓ Test Data : 테스트용 데이터
    - ✓ Formula 내 독립 변수 : 문자벡터
    - ✓ Control Parameters : 컨트롤 및 C5.0 모델링시 사용되는 옵션 값들 (Winnowing, Global Pruning, Boosting)
    - ✓ CF : Pruning Severity – 가지치기의 강도
  - 사용법

```
str <- c("fav_time","buy_brd","avg_disc","amt12")
# Sex ~ 독립변수 값 들에 할당 결과 : sex ~ fav_time + buy_brd + avg_disc + amt12 의 formula로 변환
sub <- c(T,F) # T,F의 값들 중 3개를 선택하여 표시 TTT,TFT 등등.
count <- c(seq(0,40,10)) # trials = 0 ~ 40 Step 10 형태로 입력.
train_data <- user.train[,-1] # Training Data
test_data <- user.test        # Testing Data
result <- make(user.train[,-1],user.test,str,sub,count)  # 함수적용.
```

  - 결과값.

```
> head(result6[order(result6$Accur, decreasing = T),])
       Accur                                                                Param                 Sub Cnt
22 0.7104618   sex ~ pr_pref + sea_pref + API + amt12 + instMonth + avg_disc +  FALSE, FALSE, TRUE  20
23 0.7104618     fav_part_index + fav_time + nop6 + fav_good_index + buy_brd +  FALSE, FALSE, TRUE  20
24 0.7104618                                                       main_store FALSE, FALSE, TRUE  20
82 0.7104618   sex ~ pr_pref + sea_pref + API + amt12 + instMonth + avg_disc +   TRUE, FALSE, TRUE  20
83 0.7104618     fav_part_index + fav_time + nop6 + fav_good_index + buy_brd +   TRUE, FALSE, TRUE  20
84 0.7104618                                                       main_store  TRUE, FALSE, TRUE  20
```

3. User Defined Function

사용자 함수 (자동 독립변수 할당 및 옵션 적용 함수)

```
make <- function(train_data,test_data,str,detail,cnt,CF=0.25){
  df <- data.frame(Accur = integer(), Param = character(), Sub = character(), Cnt = integer())
  # 결과 출력을 위한 빈 DataFrame 데이터형에 맞춰서 생성.
  permn_str <- permn(str) # 입력받은 독립변수들을 순열로 생성
  sub_comb <-  permutations(2,3,detail,repeats=TRUE)  # T,F로 들어온 값을 2개 중 3번 선택, 중복 허용
  # 반복 횟수 가져오기
  length_permn <- length(permn_str) # 순열의 크기
  length_sub <- NROW(sub_comb) # 생성된 Options의 크기
  length_count <- length(cnt) # trials의 크기.
  rows <- 0
  for(i in 1:length_permn){
    # 순열의 크기만큼 반복 (반복시키는 이유 - Winnowing=T에서 입력되는 독립변수 순서에 따라 사용률이 달라지고 모델도 달라짐)
    param <- paste(permn_str[[i]],collapse = "+") # 독립변수들끼리 +로 문자열 합치기 a + b + c
    for(j in 1:length_sub){ # Options만큼 반복.
      sub_1 <-  sub_comb[j,1] # Winnowing
      sub_2 <-  sub_comb[j,2] # noGlobalPruning
      sub_3 <- sub_comb[j,3] # Trials
      c5_options <- C5.0Control(winnow = sub_1, noGlobalPruning = sub_2, CF = CF) # 옵션 생성 Default CF = 0.25
      for(k in 1:length_count){ # trials 횟수 만큼 반복
        rows <- rows + 1
        params <- as.formula(gsub("\\\"","",paste("sex",param, sep=" ~ "))) # Formula 생성.
        if(cnt[k]>0){ # trials > 0 이상일때만
          c5_model <- C5.0(params, data=train_data, control = c5_options, rules=sub_3, trials=cnt[k])
        }else{
          c5_model <- C5.0(params, data=train_data, control = c5_options, rules=sub_3)
        }
        # 정확도 확보를 위한 처리.
        test_data$c5_pred <- predict(c5_model,test_data,type="class")
        test_data$c5_pred_prob <- round(predict(c5_model,test_data,type="prob"),2)
        Accur <- confusionMatrix(test_data$c5_pred, test_data$sex)
        AccurResult <- Accur$overall[[1]] # 해당 모델의 정확도 확인.
        param_in <- deparse(params) # Formula 를 문자열 형태로 변환
        sub <- toString(sub_comb[j,]) # List에 있는 T,F 값들을 문자열 형태로 변환
        df2 <- data.frame(Accur = AccurResult, Param = param_in, Sub = sub, Cnt = cnt[k])
        # DataFrame에 동적으로 삽입하기 위해 DF생성
        df <- rbind(df,df2) #df에 합치기.
      }
    }
    print(rows)
  }
  return(df) #모든 반복이후 결과값 출력
}
```

4. Decision Tree

Model Choice

- 모델선택
  - ✓ 돌렸던 변수 및 조건들에 대해 가장 정확도가 높은 모델 선정.
  - ✓ 옵션 winnowing : F, noGlobalPruning = F, rules = T, trials = 20
  - ✓ 독립변수: pr_pref + sea_pref + API + amt12 + instMonth + avg_disc + fav_part_index + fav_time + nop6 + fav_good_index + buy_brd + main_store

```
c5_options <- C5.0Control(winnow = F, noGlobalPruning = F)
c5_model <- C5.0(sex ~ pr_pref + sea_pref + API + amt12 + instMonth + avg_disc +
                    fav_part_index + fav_time + nop6 + fav_good_index + buy_brd +
                    main_store, data=train_data, control = c5_options, rules=T, trials=20)
test_data$c5_pred <- predict(c5_model,test_data,type="class")
test_data$c5_pred_prob <- round(predict(c5_model,test_data,type="prob"),2)
confusionMatrix(test_data$c5_pred, test_data$sex)
```

```
boost           7167(23.9%)   <<


          (a)   (b)    <-classified as
         ----  ----
         2858  6240    (a): class 1
          927 19959    (b): class 2


       Attribute usage:

       100.00% pr_pref
       100.00% sea_pref
       100.00% amt12
       100.00% instMonth
       100.00% avg_disc
       100.00% fav_part_index
       100.00% fav_time
       100.00% main_store
        96.93% fav_good_index
        95.28% API
        91.84% buy_brd
        80.66% nop6
```

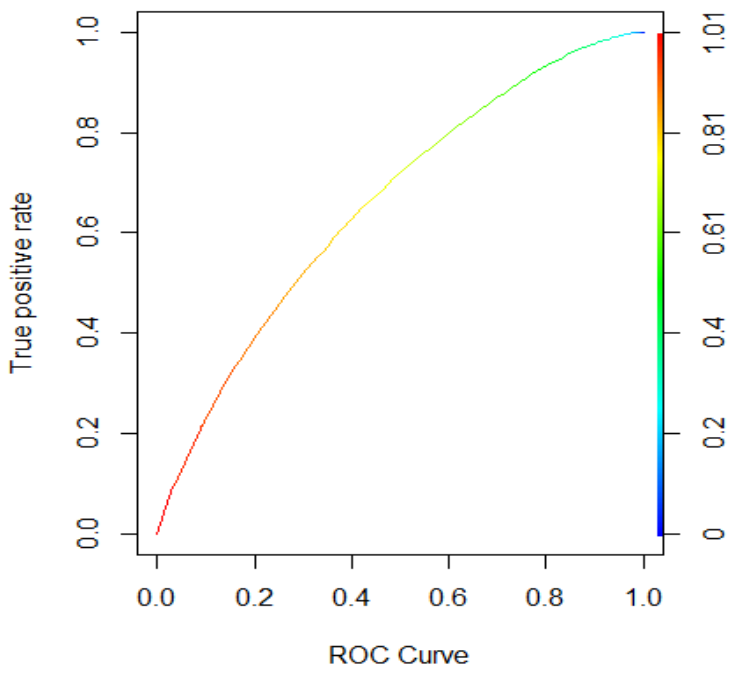
```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1  1296  1019
         2  4768 12904

               Accuracy : 0.7105
                 95% CI : (0.7041, 0.7167)
    No Information Rate : 0.6966
    P-Value [Acc > NIR] : 9.687e-06

                  Kappa : 0.1702
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.21372
            Specificity : 0.92681
         Pos Pred Value : 0.55983
         Neg Pred Value : 0.73019
             Prevalence : 0.30340
         Detection Rate : 0.06484
   Detection Prevalence : 0.11583
      Balanced Accuracy : 0.57027

       'Positive' Class : 1
```

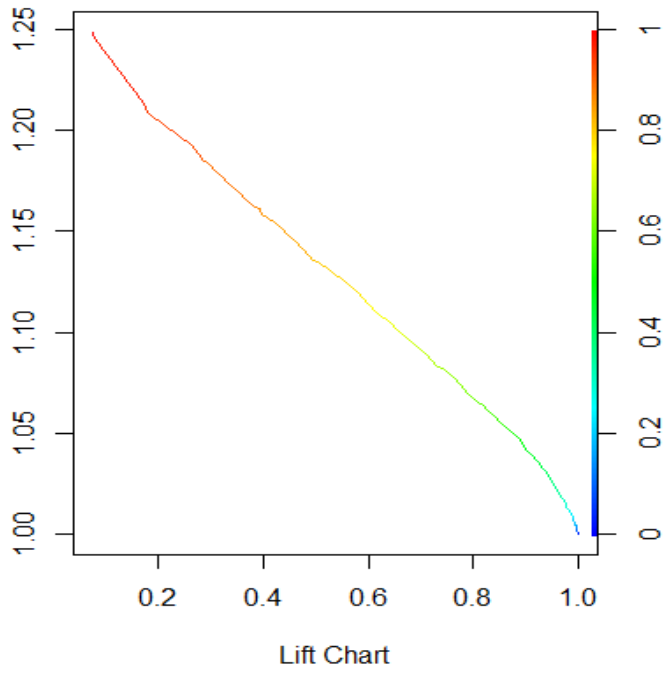
- No Information Rage : 0.6966
- Accuracy : 0.7105

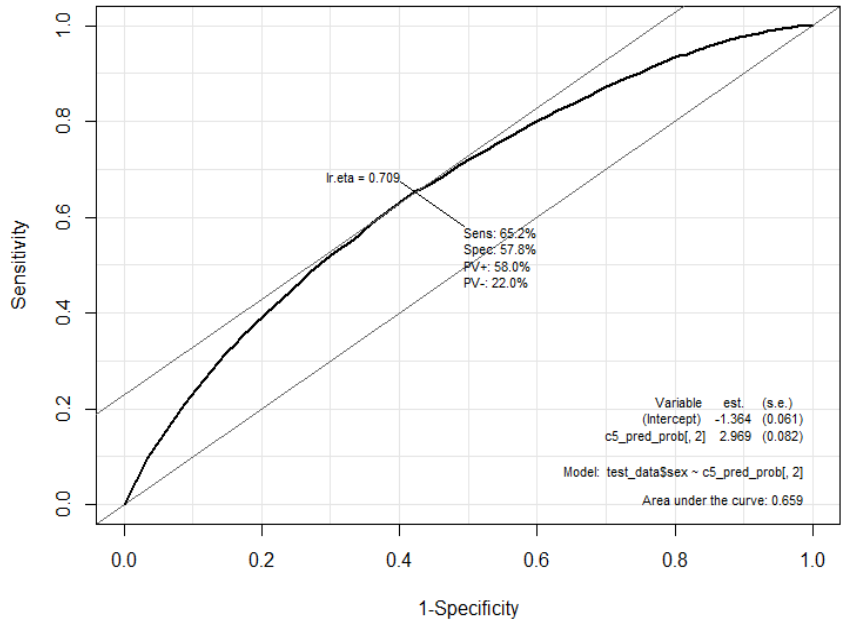
4. Decision Tree

Model Choice

- ROCcurve & LiftChart







- AUC : 0.66

```
> performance(c5_pred,"auc")@y.values[[1]]
[1] 0.6592522
```

✓ 일반적으로 덜 정확한 모델의 값이 출력 되었다. 0.5 < AUC <= 0.7

5. Conclusion

결론

- 느낀점
  - 구매 내역으로 고객의 성별을 구분하는 것에 있어서 많은 어려움이 있는 것으로 판단 됨.
  - 남성 또는 여성의 구매 패턴이 비슷하며 뚜렷한 구분이 되는 데이터가 존재 하지 않는 것으로 판단 됨.
  - CS 데이터 내 그룹사 및 나이 정보를 가지고 모델을 생성했을 경우 83% 이상의 성별을 매칭 할 수 있었습니다.


THANK YOU
Grand Central