201310630 이준석
201510657 정선민
201515001 김승원
201610593 김진영

INDEX

001
연구 목적

L-POINT 고객의 사용 내역 데이터를 바탕으로 고객 분석

RFM 모형 & 장바구니 분석

고객 세그먼트 분류

맞춤형 마케팅 방법 제안

결과값을 이용

002
데이터 분석 및 결과 도출 과정

L.POINT

품목별
업종별
시간대별

장바구니분석
RFM

사용한 분석 TOOL 소개

003
장바구니 분석(Market Basket Analysis)

장바구니 분석?

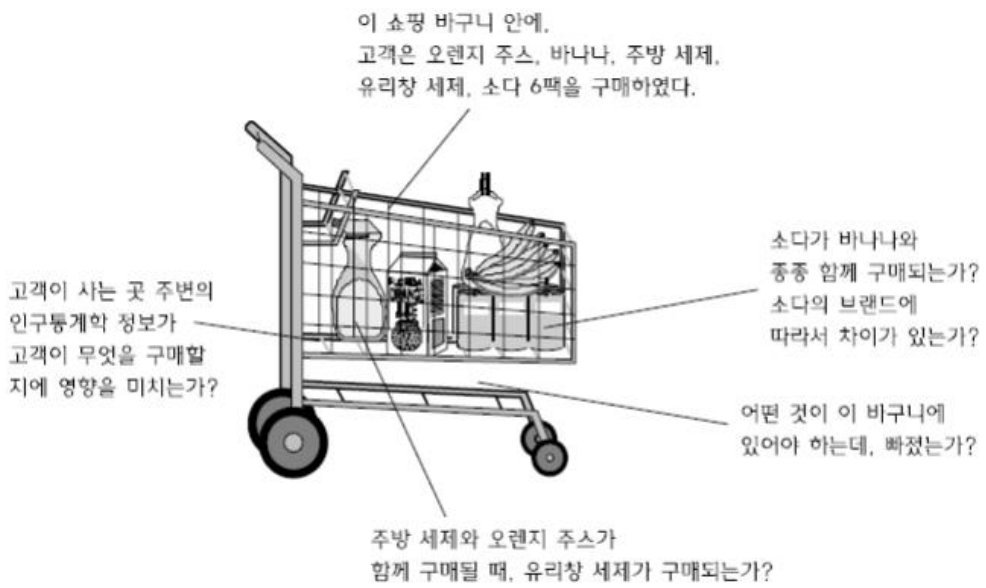
하나의 장바구니는 고객이 한 번의 구매에서 산 물건을 알려주는데

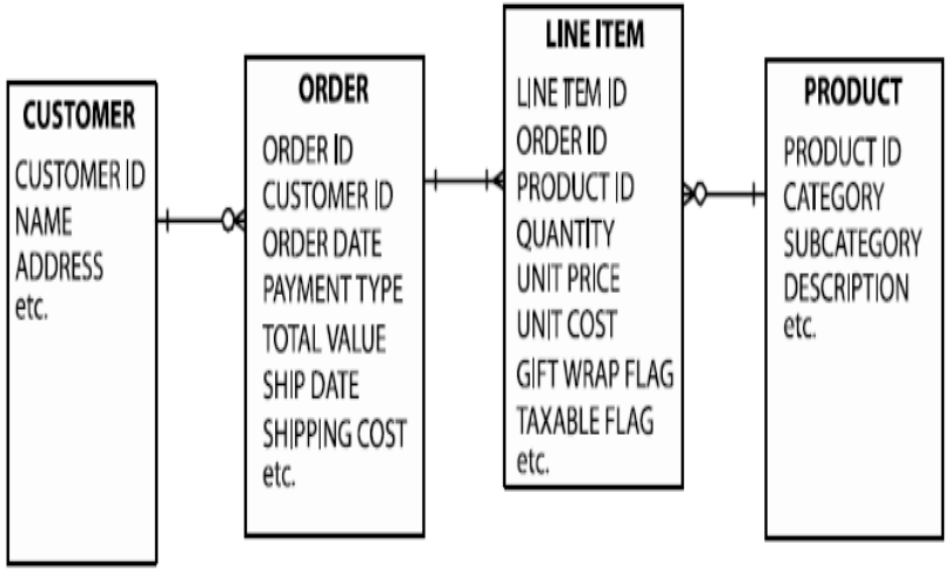
어떤 물품들이 함께 구매되는 경향이 있는지 분석하여

패턴 및 연관 규칙을 분석하는 기법

월마트는 장바구니 분석을 통해 **'맥주와 기저귀'** 의 연관규칙을 발견하고 마케팅 활동으로 활용

003
장바구니 분석(Market Basket Analysis)





- 장바구니 분석은 **하나의 기법을 지칭하는 것은 아님**
- POS 거래 데이터를 이해하는 것과 관련된 여러 분석 통칭
- **분석한 정보를 바탕으로 마케팅 방법으로 활용** 상품의 배치, 특정 상품에 대한 행사 여부, 쿠폰 발행 등

003
장바구니 분석(Market Basket Analysis)

고객	물품
1	오렌지 주스, 소다
2	우유, 오렌지 주스, 유리창 세제
3	오렌지 주스, 주방 세제
4	오렌지주소, 주방 세제, 소다
5	유리창 세제, 소다

	오렌지 주스	유리창 세제	우유	소다	주방 세제
오렌지 주스	4	1	1	2	1
유리창 세제	1	2	1	1	0
우유	1	1	1	0	0
소다	2	1	0	3	1
주방 세제	1	0	0	1	2

- **오렌지 주스**와 **소다**의 같이 팔리는 경우가 많다
- **주방 세제**는 **우유**와 같이 팔리지 X
- **우유**는 **소다 혹은 주방세제**와 같이 팔리지 X

003
고객가치분석(Recency/Frequency/Monetary)

구매 최근성 (Recency)　구매 빈도 (Frequency)　구매 금액 (Monetary)

- **세가지 변수**를 측정한 지표를 바탕으로 고객이 기업에 가져다 주는 수익에 얼마나 기여하는지를 분석하는 기법

- RFM 모형은 다양한 고객가치 측정 지표들 가운데 재무적인 가치 측정 뿐만 아니라, 관계 활동에 대한 질적 측면도 함께 고려한 고객가치 평가 모형

003
고객가치분석(Recency/Frequency/Monetary)

구매 최근성
(Recency)

고객의 **마지막 구매 시점**이 언제 인지 나타내는 변수

일반적으로 최근에 구매한 고객일수록 현재의 관계가 유의미하다고 판단

구매 빈도
(Frequency)

고객이 정해진 기간 동안 **얼마나 자주 구매** 했는지를 나타내는 변수

동일한 기간 동안 구매횟수가 많을수록 높은 점수가 부과

고객의 구매/이용활동성을 판단

구매 금액
(Monetary)

일정 기간 동안에 고객의 **총 구매 금액**을 나타내는 변수

구매액이 높을 수록 높은 점수를 획득

BUT, 지나치게 높은 구매액이 존재 -> 상한선을 두는 것이 전체적인 지수 왜곡을 방지

RFM 모형

RFM 지수 = a • 최근성(R) + b • 구매빈도(F) + c • 구매액(M)

- - a/b/c는 가중치를 나타냄
- 산업에 따라 R • F • M의 중요도가 다를 수 있으므로 그 중요도에 따라 다른 가중치를 적용하는 것이 합리적
- 경우에 따라 유의미한 변수만을 선별적으로 선택하여 적용할 수 있음

003
고객가치분석(Recency/Frequency/Monetary)

RFM 모형의 대표적인 활용 방법

- **고객 세그먼트**

R/F/M 각 변수에 따라 고객을 분류하여, 차별화된 마케팅을 위한 고객세분화에 활용

- **고객 스코어링**

각 고객의 RFM지수를 산출하여 고객을 평가하는 지수로 활용

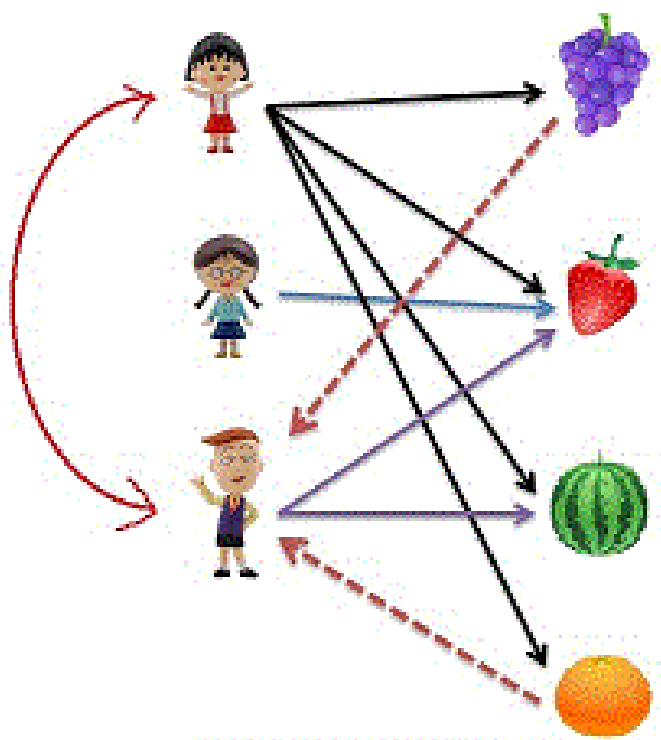
ex) 고객의 등급을 부여하여 고객군의 분류

003
협업 필터링 (Collaborative Filtering)

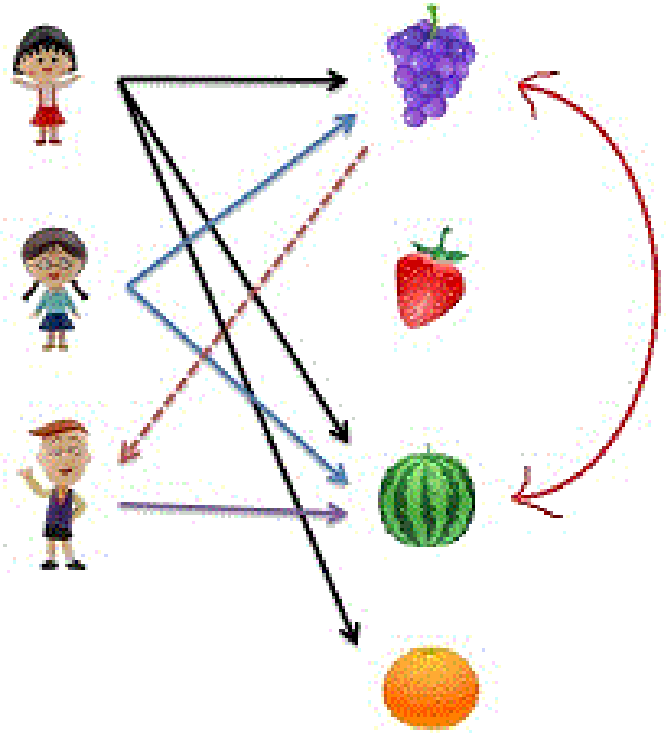
- 많은 사용자들로부터 얻은 기호정보에 따라 **사용자들의 관심사들을 자동적으로 예측**하게 해주는 방법

- 고객들의 선호도와 관심 표현을 바탕으로 선호도, 관심에서 비슷한 패턴을 가진 고객들을 식별해 내는 기법

003
협업 필터링 (Collaborative Filtering)



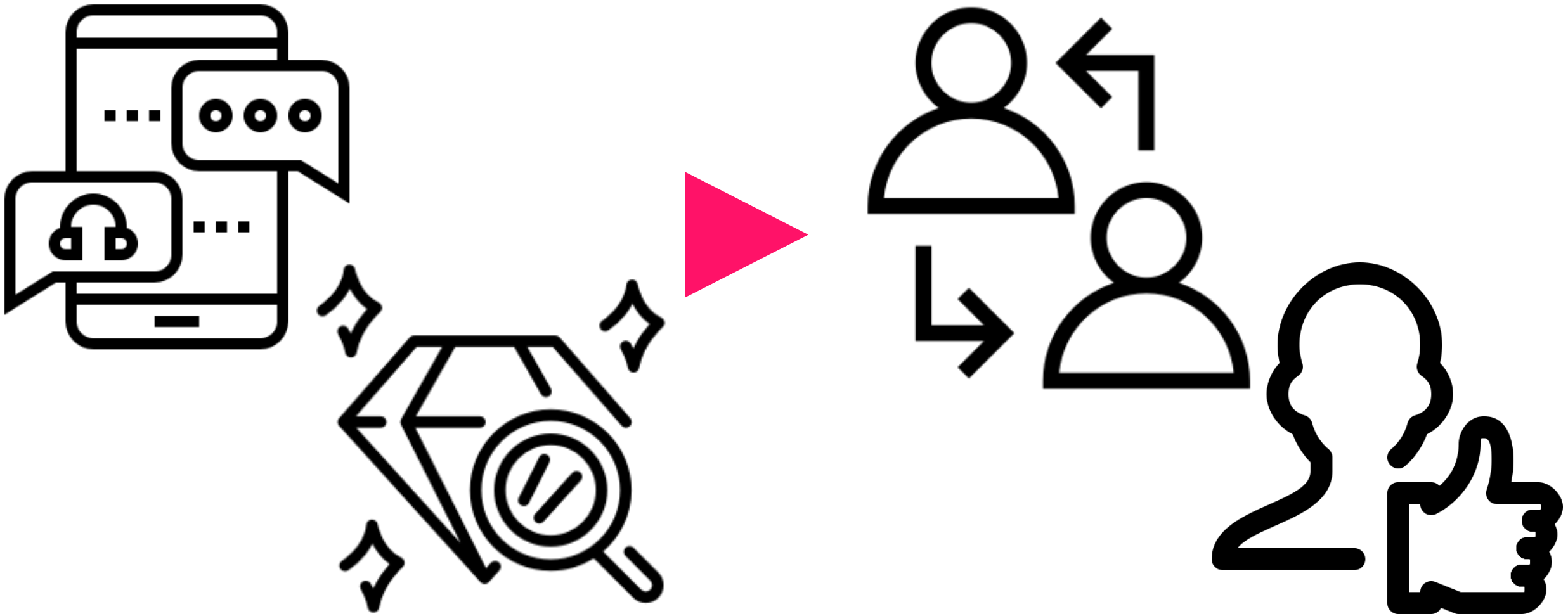
User-based filtering



Item-based filtering

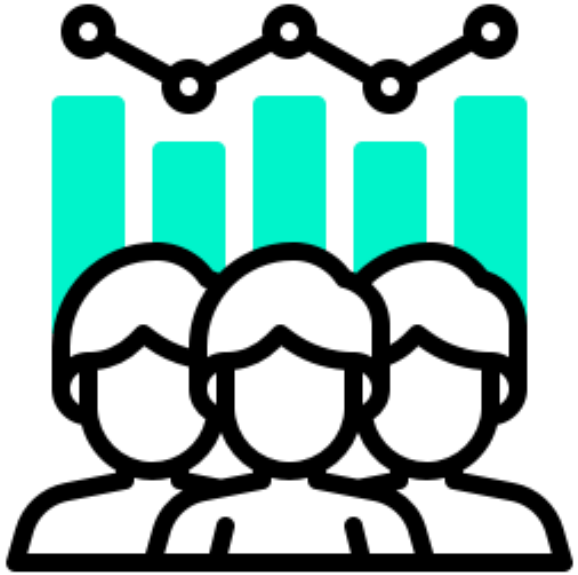
003
협업 필터링 (Collaborative Filtering)

사용자들의 과거의 경향이 미래에서도 그대로 유지 될 것이라는 전제



003
협업 필터링 (Collaborative Filtering)

기존의 어느 정도 예측이 가능한 고객들과
비슷한 패턴을 가진 고객들을 탐색에서도
그대로 유지 될 것이라는 전제



기존 고객들의 행동을 예측하기 위해
첫 번째 단계에서 찾은 비슷하다고 생각된
고객들의 행동을 수치화하여 사용.

003 종류
아이템 기반 협업 필터링

	$item_1$	$item_2$	$item_3$	...	$item_n$
$user_1$		5	2		1
$user_2$	3				
$user_3$	1		3		
.					
.					
.					
$user_{m-1}$	5		4		2
$user_m$		4			3

고객이 **선호도를 입력한 기존의 상품들**과 **예측하고자 하는 상품**과의 **유사도**를 계산하여

고객의 선호도를 예측하는 방법

데이터 분석 과정

004
전처리

쇼핑업종상품분류 - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
BIZ_UNIT,PD_S_C,PD_S_NM,PD_M_NM,PD_H_NM
A04,0341,단행본서적(직배),특수점서적,서적/음반
A04,0001,삼각김밥,삼각김밥,미반
A04,0002,The커진삼각김밥,삼각김밥,미반
A04,0003,말이김밥,김밥,미반
A04,0004,트레이김밥,김밥,미반
A04,0005,초밥,김밥,미반
A04,0006,도시락,도시락,미반
A04,0007,미니도시락,도시락,미반
A04,0008,기타,도시락,미반
A04,0009,국/찌개도시락,도시락,미반
A04,0010,떠먹는주먹밥,용기형주먹밥,미반
A04,0011,동그란주먹밥,동그란주먹밥/사각주먹밥,미반
A04,0012,사각주먹밥,동그란주먹밥/사각주먹밥,미반
A04,0013,안주/간식류,안주/간식류,미반
A04,0014,조리면,조리면,미반
A04,0015,삼각샌드,샌드위치,조리빵
A04,0016,토스트샌드,샌드위치,조리빵
A04,0017,햄버거,햄버거,조리빵
```

구매 - 메모장

서식(O) 보기(V)

```
IZ_UNIT,PD_
3,A01,0215,0
9,A01,0075,0
9,A01,0149,0
0,A01,0138,0
0,A01,0138,0
0,A01,0558,0
1,A01,0013,0
2,A01,0223,0
2,A01,0216,0
2,A01,0121,0
2,A01,0121,0
2,A01,0421,0
2,A01,0421,0
3,A01,0532,0
4,A01,0113,0
5,A01,0143,0
5,A01,0439,0
15356,004085,A01,0531,0025,20150411,14,2198000,1
```

쇼핑외업종상품구매 - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
ID,BIZ_UNIT,CRYM,U_AM,U_CT
09544,D03,201511,14400,1
09544,C01,201510,18000,1
09544,D01,201508,9500,2
09544,D01,201507,4200,1
09544,D03,201505,3900,1
09544,D01,201512,26000,3
09544,D01,201510,9500,1
09544,B03,201504,314283,5
09544,C01,201502,38000,3
09544,D01,201502,18100,1
09544,C01,201511,20000,1
09544,C01,201502,2500,1
09595,D03,201510,26100,4
09595,C02,201504,18900,1
09595,B03,201502,104716,2
09595,D03,201505,12200,1
09595,D03,201501,10500,1
09595,D03,201508,28700,5
```

고객DEMO - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
ID,GENDER,AGE_PRD,HOM_PST_NO
00001,1,60PRD,052
00002,2,60PRD,080
00003,2,60PRD,620
00004,1,60PRD,120
00005,1,60PRD,
00006,2,60PRD,058
00007,2,60PRD,052
00008,2,60PRD,480
00009,2,60PRD,470
00010,1,60PRD,055
00011,1,60PRD,072
00012,2,60PRD,061
00013,2,60PRD,620
00014,1,60PRD,062
00015,2,60PRD,036
00016,2,60PRD,055
00017,2,60PRD,056
00018,1,60PRD,055
```

004 전처리

데이터 로드

```
> system.time(shop <-read.table("쇼핑업종상품구매.txt",
+                               sep=",",
+                               header=TRUE,
+                               encoding="UTF-8",
+                               stringsAsFactors=FALSE))
 사용자  시스템 elapsed
  19.52    0.58   20.61
```

```
> system.time(shop <- fread("쇼핑업종상품구매.txt",
+                 sep=",",
+                 header=TRUE,
+                 encoding="UTF-8",
+                 stringsAsFactors = FALSE))
 사용자  시스템 elapsed
   0.96    0.17    0.35
```

```
23 shop <- fread("쇼핑업종상품구매.txt",
24                sep=",",
25                header=TRUE,
26                encoding="UTF-8",
27                stringsAsFactors = FALSE))
28
29 customer<-fread("고객DEMO.txt",
30                  sep=",",
31                  header=TRUE,
32                  encoding="UTF-8",
33                  stringsAsFactors=FALSE)
34
35 category<-fread("쇼핑업종상품분류.txt",
36                  sep=",",
37                  header=TRUE,
38                  encoding="UTF-8",
39                  stringsAsFactors=FALSE)
40
41 nonshop <-fread("쇼핑외업종상품구매.txt",
42                  sep=",",
43                  header=TRUE,
44                  encoding="UTF-8",
45                  stringsAsFactors=FALSE)
```

category	3788 obs. of 5 variables
customer	20000 obs. of 4 variables
nonshop	178659 obs. of 5 variables
shop	3641082 obs. of 9 variables

약 170MB인 txt파일을 data.frame으로 불러오려면 약 20초

data.table의 fread함수를 통해 약 50배 빠르게 불러옴

전처리 (customer, shop)

```r
# 각 데이터를 우선 살펴보자 ( customer )

str(customer)
head(customer,10)
summary(customer)

##gender, age_prd 속성 범주화
customer$GENDER <- as.factor(customer$GENDER)
customer$AGE_PRD <- as.factor(customer$AGE_PRD)

##우리는 지역 고려하지 않으므로 삭제
customer <- customer %>% select(-HOM_PST_NO)
```

```r
# 각 데이터를 우선 살펴보자 ( shop )

str(shop)
head(shop,10)
summary(shop)

## 점포코드 삭제
shop <- shop %>% select(-BR_C)

## biz_unit+pd_s_c로 봐야 어떤상품인지 알 수 있다/ 코드통합

shop <- shop %>%
  mutate(P_CODE=paste(BIZ_UNIT,PD_S_C,sep="_"))

## biz_unit + rct_no 로 봐야 영수증별로 알 수 있다 / 코드통합

shop <- shop %>%
  mutate(R_NUM=paste(BIZ_UNIT,RCT_NO,sep="_"))

## rct_no, biz_unit, pd_s_c 삭제

shop <- shop %>% select(-RCT_NO,-BIZ_UNIT,-PD_S_C)
```

4개의 데이터 모두 str(), head(), summary() 등의 함수를 통해 데이터를 확인 후

필요 없는 칼럼 삭제, 속성 변경, 파생변수 추가, 코드 변환 등 전처리

전처리 (category, nonshop)

```r
# 각 데이터를 우선 살펴보자 ( category )
str(category)
head(category,10)

# 각각의 카테고리가 몇 개씩 있을까?
length(unique(category$PD_H_NM))
length(unique(category$PD_M_NM))
length(unique(category$PD_S_NM))

## biz_unit마다 겹치는 중분류가 있는듯
head(category$PD_M_NM,100)
category %>% filter(PD_M_NM =="도시락") %>% head(100)

## biz_unit+pd_s_c로 봐야 어떤상품인지 알 수 있다/ 코드통합

category <- category %>%
  mutate(P_CODE=paste(BIZ_UNIT,PD_S_C,sep="_")) %>%
  select(P_CODE,PD_M_NM,PD_S_NM)
```

```r
# 각 데이터를 우선 살펴보자 ( nonshop )
summary(nonshop)
str(nonshop)
head(nonshop)

## BIZ_UNIT 범주화
nonshop$BIZ_UNIT <- as.factor(nonshop$BIZ_UNIT)

## 전체고객중 noshop을 이용한 고객은?(약88%의 고객)
length(unique(nonshop$ID))/length(customer$ID) * 100

## 알아보기 쉽게 코드를 바꿔보자
nonshop$BIZ_UNIT <- ifelse(nonshop$BIZ_UNIT=="B01","Hotel",
                    ifelse(nonshop$BIZ_UNIT=="B02","Travel",
                    ifelse(nonshop$BIZ_UNIT=="B03","Taxfree",
                    ifelse(nonshop$BIZ_UNIT=="C01","Theater",
                    ifelse(nonshop$BIZ_UNIT=="C02","Themepark",
                    ifelse(nonshop$BIZ_UNIT=="C03","baseball",
                    ifelse(nonshop$BIZ_UNIT=="D01","Fastfood",
                    ifelse(nonshop$BIZ_UNIT=="D02","Familyrestaurant",
                    ifelse(nonshop$BIZ_UNIT=="D03","Cafe",nonshop$BIZ_UNIT)))))))))
```

4개의 데이터 모두 str(), head(), summary() 등의 함수를 통해 데이터를 확인 후

필요 없는 칼럼 삭제, 속성 변경, 파생변수 추가, 코드 변환 등 전처리

분석을 위한 데이터 병합

```
177 ## customer + shop + category
178 shopping <- merge(customer, shop, by="ID")
179 shopping <- merge(shopping, category, by="P_CODE")
180
181 ## 필요한 컬럼만 남기자
182 shopping <- shopping %>%
183   select(ID,GENDER,AGE_PRD,DE_DT,DE_HR,
184          R_NUM, BUY_CT,BUY_AM,P_CODE,PD_S_NM,PD_M_NM)
185
186 ## customer + nonshop
187 notshopping <- merge(customer, nonshop, by="ID")
188
```

```
> head(shopping)
   ID GENDER AGE_PRD    DE_DT DE_HR      R_NUM BUY_CT BUY_AM P_CODE PD_S_NM PD_M_NM PD_H_NM
1:  1      1   60PRD 20150513    15 A01_378813      1  24530  A01_1     우육     육류     식품
2: 16      2   60PRD 20150328    18 A01_351601      1  18950  A01_1     우육     육류     식품
3: 16      2   60PRD 20151105    11 A01_166199      1  15830  A01_1     우육     육류     식품
4: 16      2   60PRD 20151105    11 A01_166199      1  15740  A01_1     우육     육류     식품
5: 16      2   60PRD 20150324    14 A01_312326      1  14000  A01_1     우육     육류     식품
6: 16      2   60PRD 20151102    14 A01_343144      1  12100  A01_1     우육     육류     식품
```

```
> head(notshopping)
   ID GENDER AGE_PRD   BIZ_UNIT    CRYM   U_AM U_CT
1:  1      1   60PRD    Taxfree 201502 270518    3
2:  1      1   60PRD       Cafe 201512   9200    1
3:  1      1   60PRD    Theater 201512   7500    1
4:  1      1   60PRD    Taxfree 201512  69887    1
5:  1      1   60PRD    Theater 201512  14000    1
6:  1      1   60PRD Themepark 201511  58000    1
```

Customer + shop + category => shopping

Customer + nonshop => notshopping

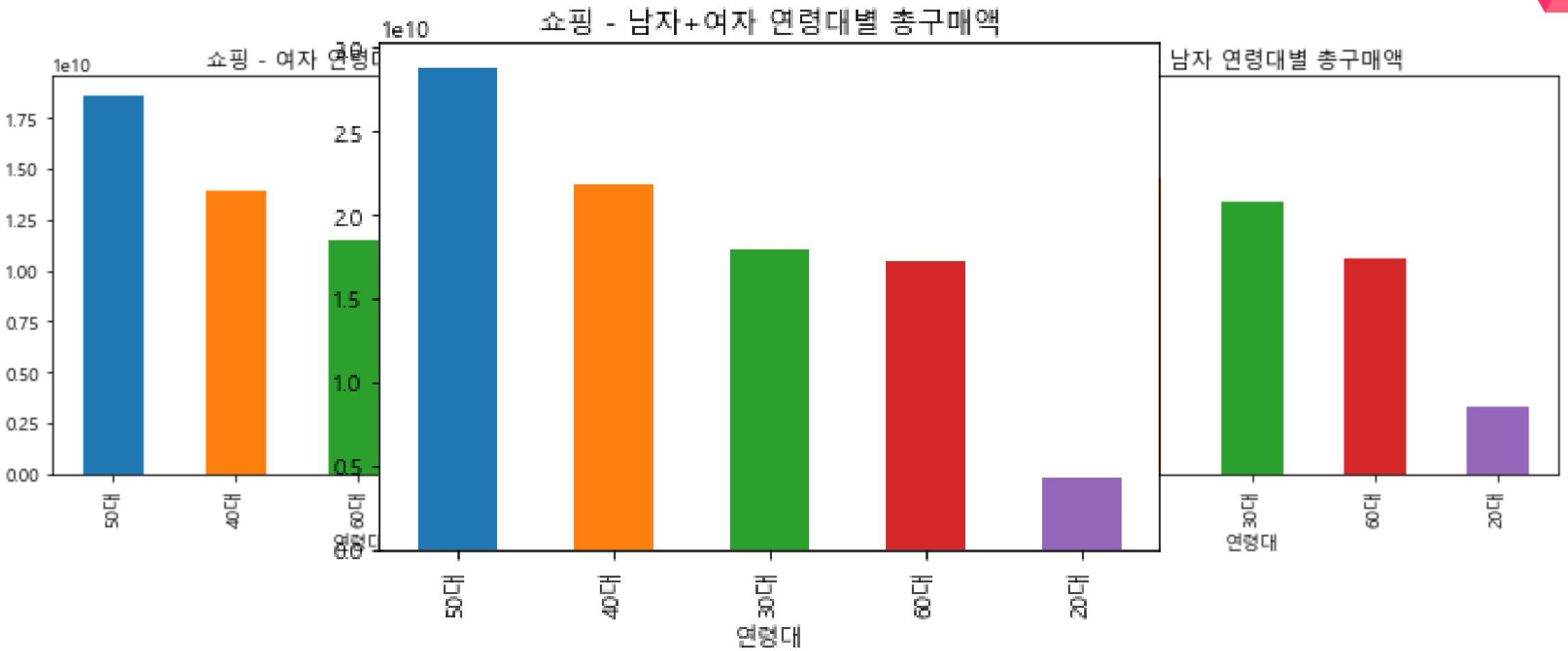
data.table의 merge() 통해 훨씬 빠르게 데이터 병합 가능

004
전처리

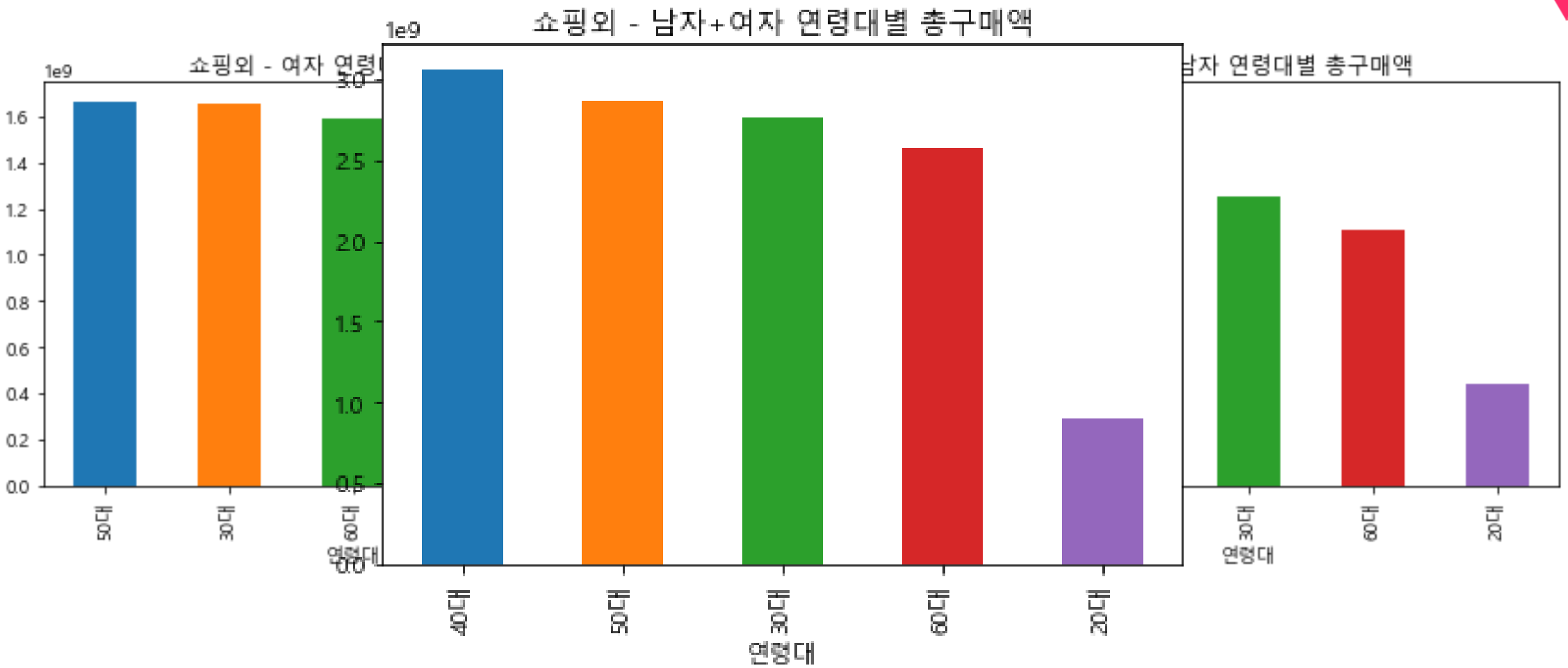
In [11]:
```
## 성별로 구분

# 남자만
male = shopping[shopping['성별'] == 1]
n_male = notshopping[notshopping['성별'] == 1]

#여자만
female = shopping[shopping['성별'] == 2]
n_female = notshopping[notshopping['성별'] == 2]
```

In [12]:
```
#성별간 나이대로 구분
#남자 20대~60대
M_20 = male[male['연령대'] == '20대']
M_30 = male[male['연령대'] == '30대']
M_40 = male[male['연령대'] == '40대']
M_50 = male[male['연령대'] == '50대']
M_60 = male[male['연령대'] == '60대']

n_M_20 = n_male[n_male['연령대'] == '20대']
n_M_30 = n_male[n_male['연령대'] == '30대']
n_M_40 = n_male[n_male['연령대'] == '40대']
n_M_50 = n_male[n_male['연령대'] == '50대']
n_M_60 = n_male[n_male['연령대'] == '60대']
```

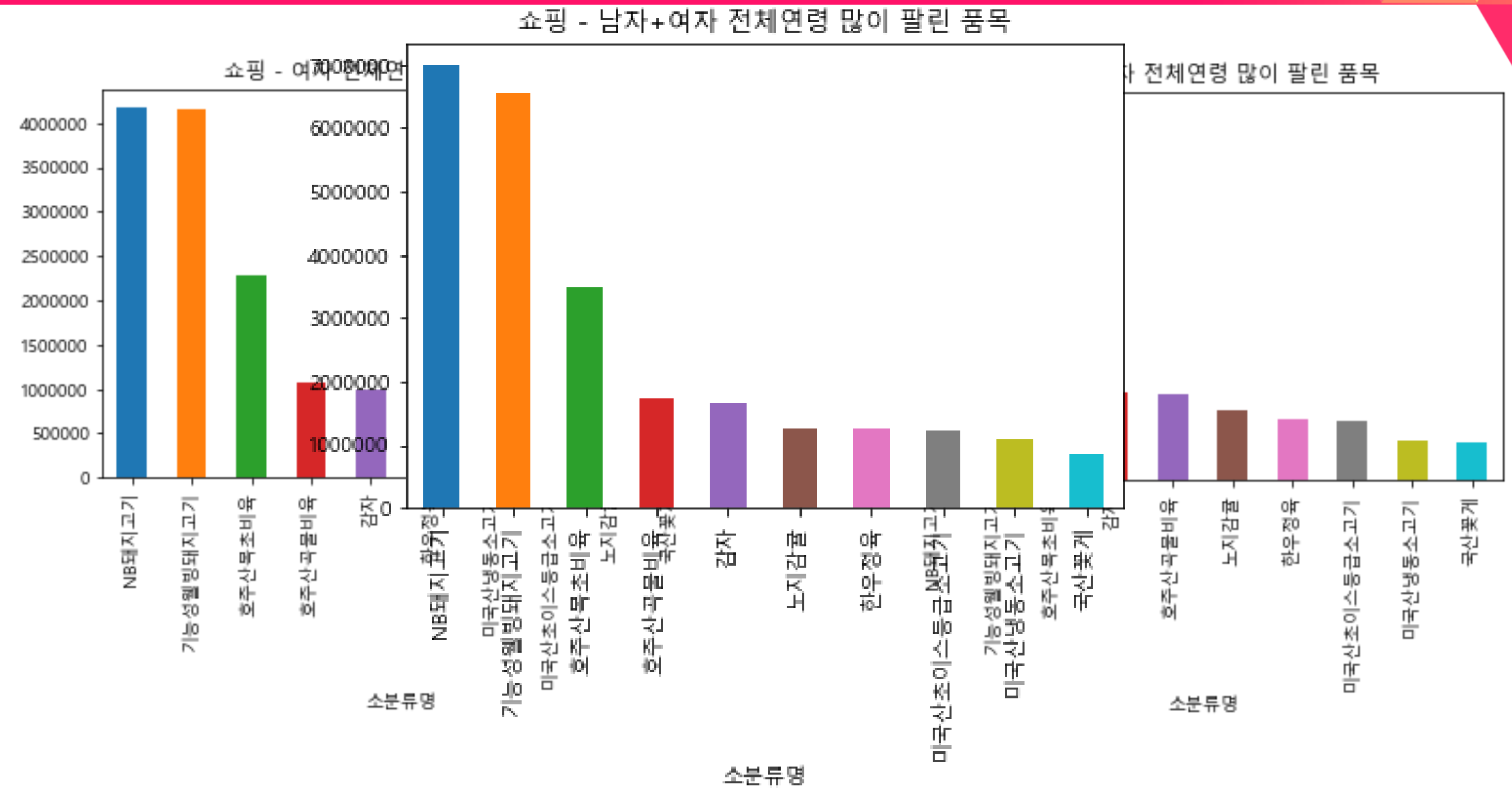
004
탐색적 데이터 분석



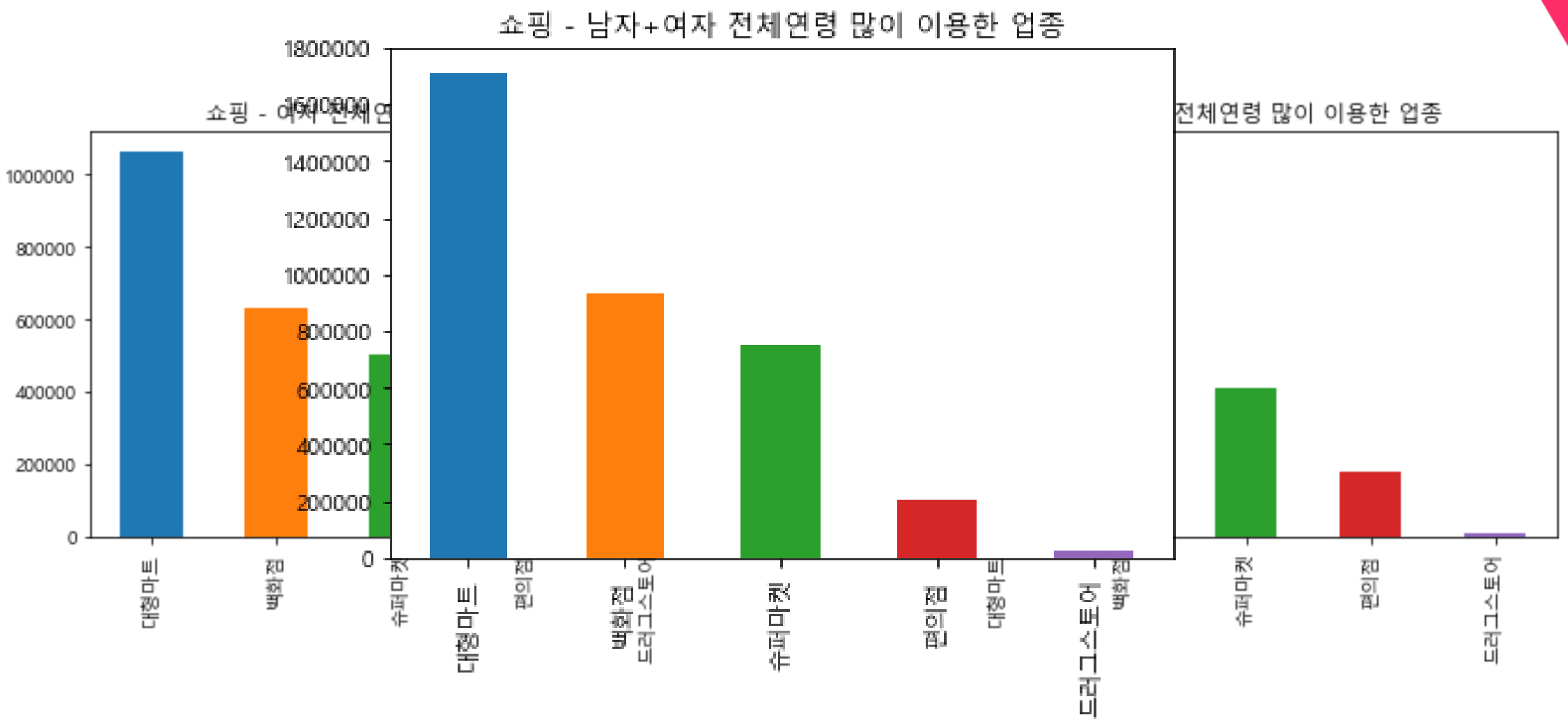
004
탐색적 데이터 분석



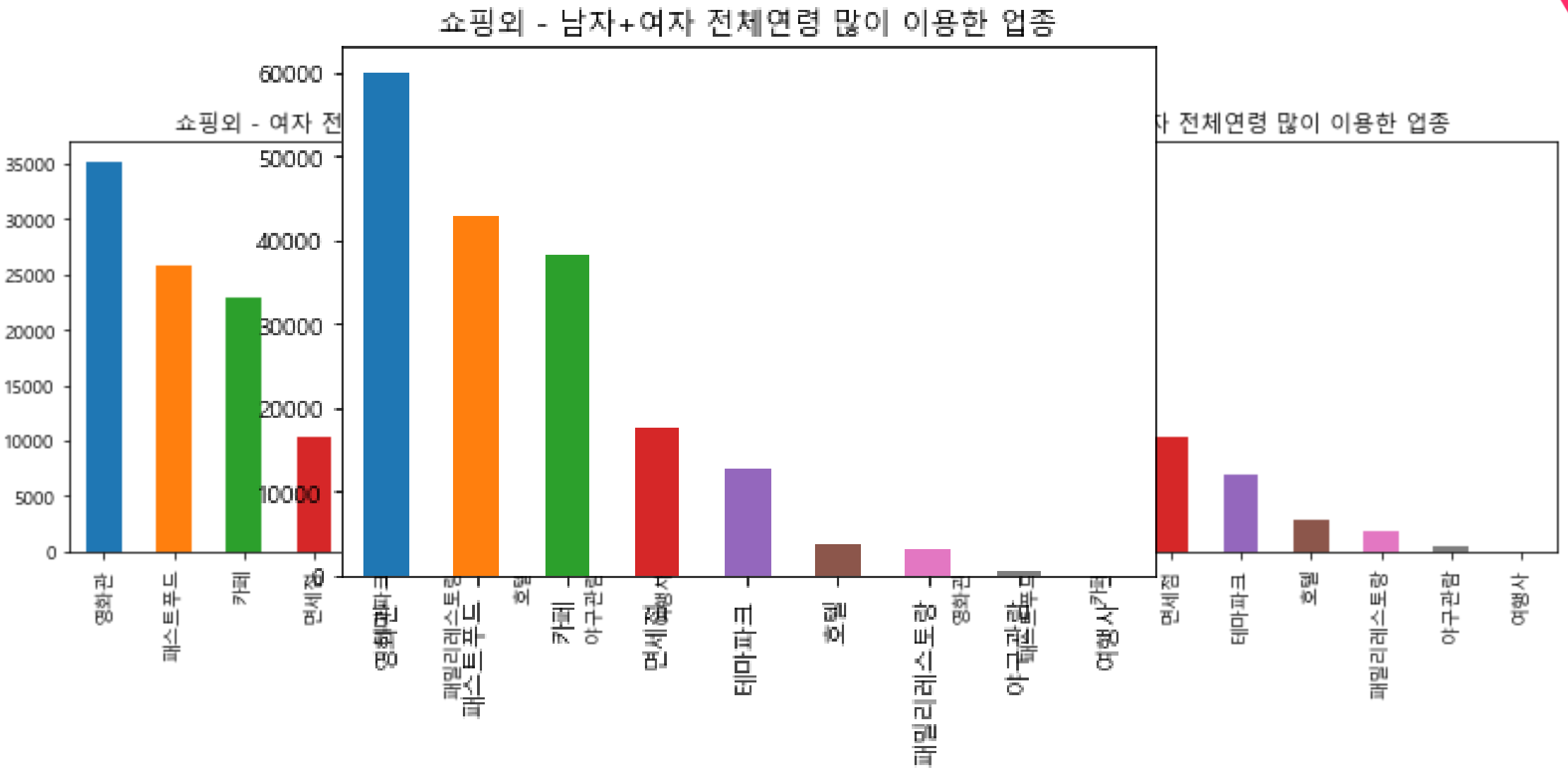
004
탐색적 데이터 분석



004
탐색적 데이터 분석

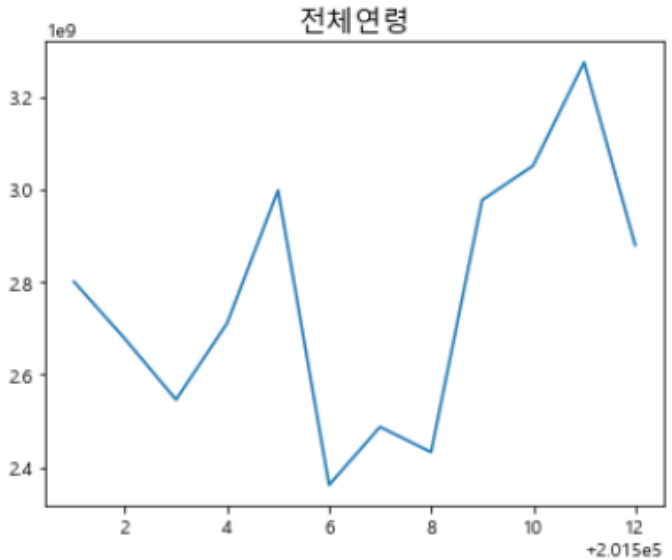


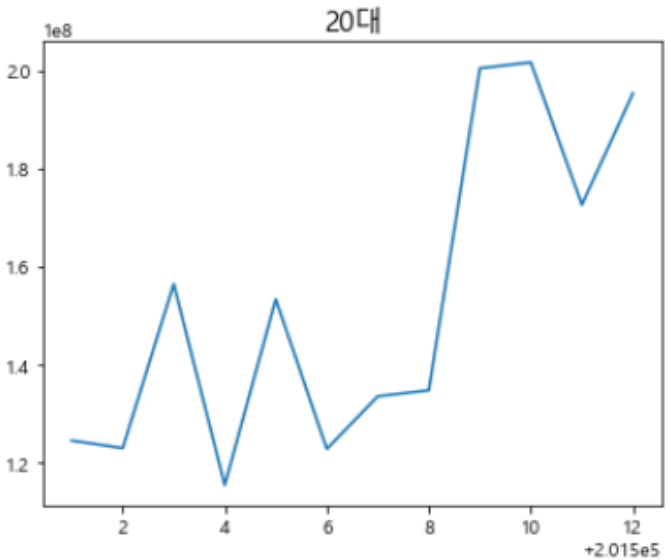
004
탐색적 데이터 분석

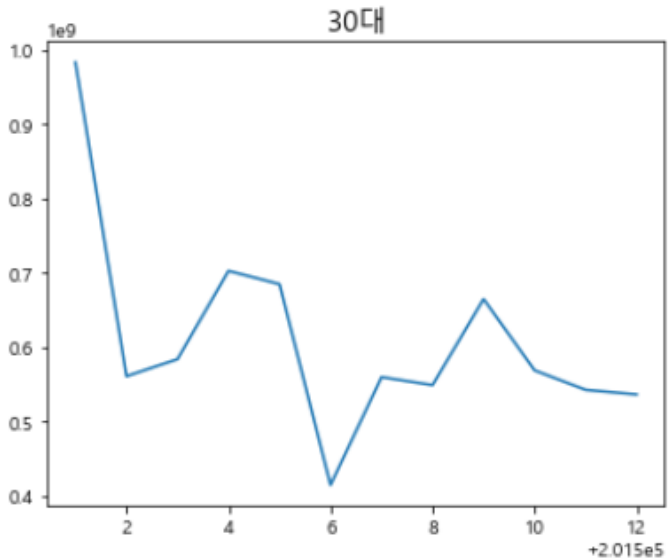


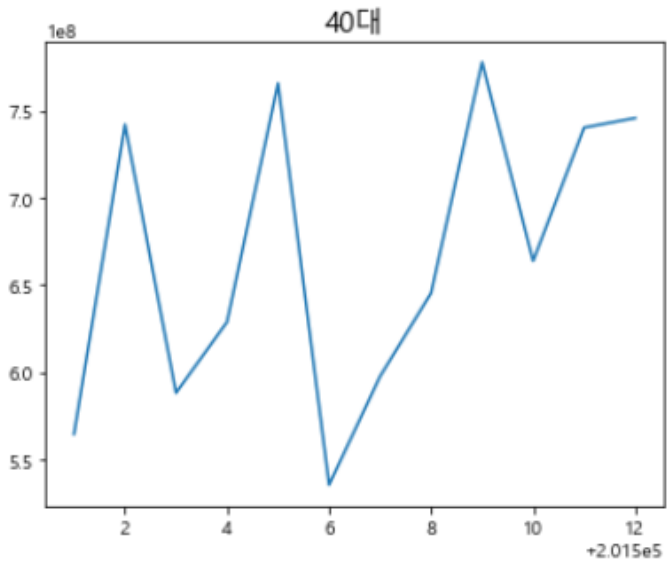
004
탐색적 데이터 분석

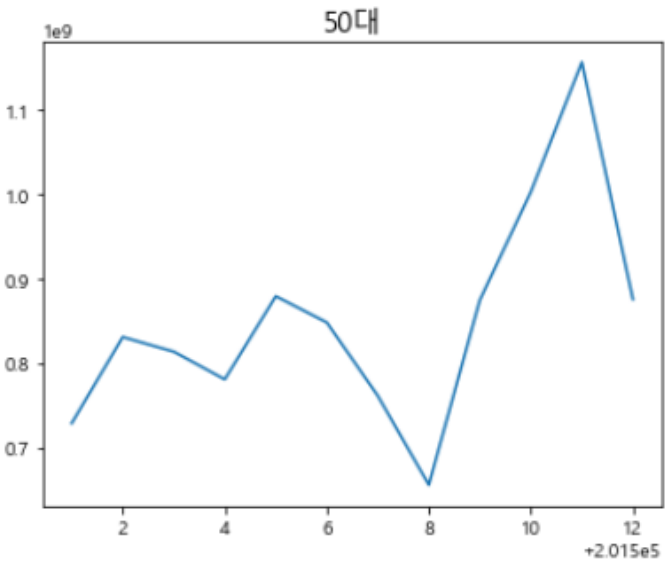
쇼핑 - 남자 연령대별 월별 총 매출액

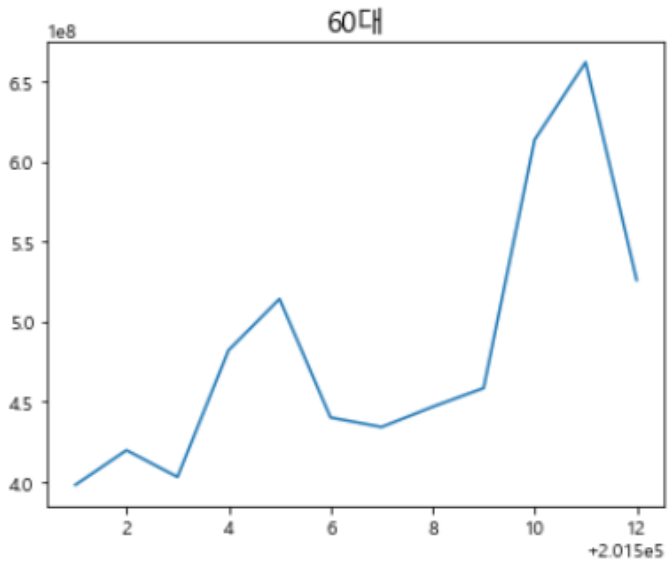




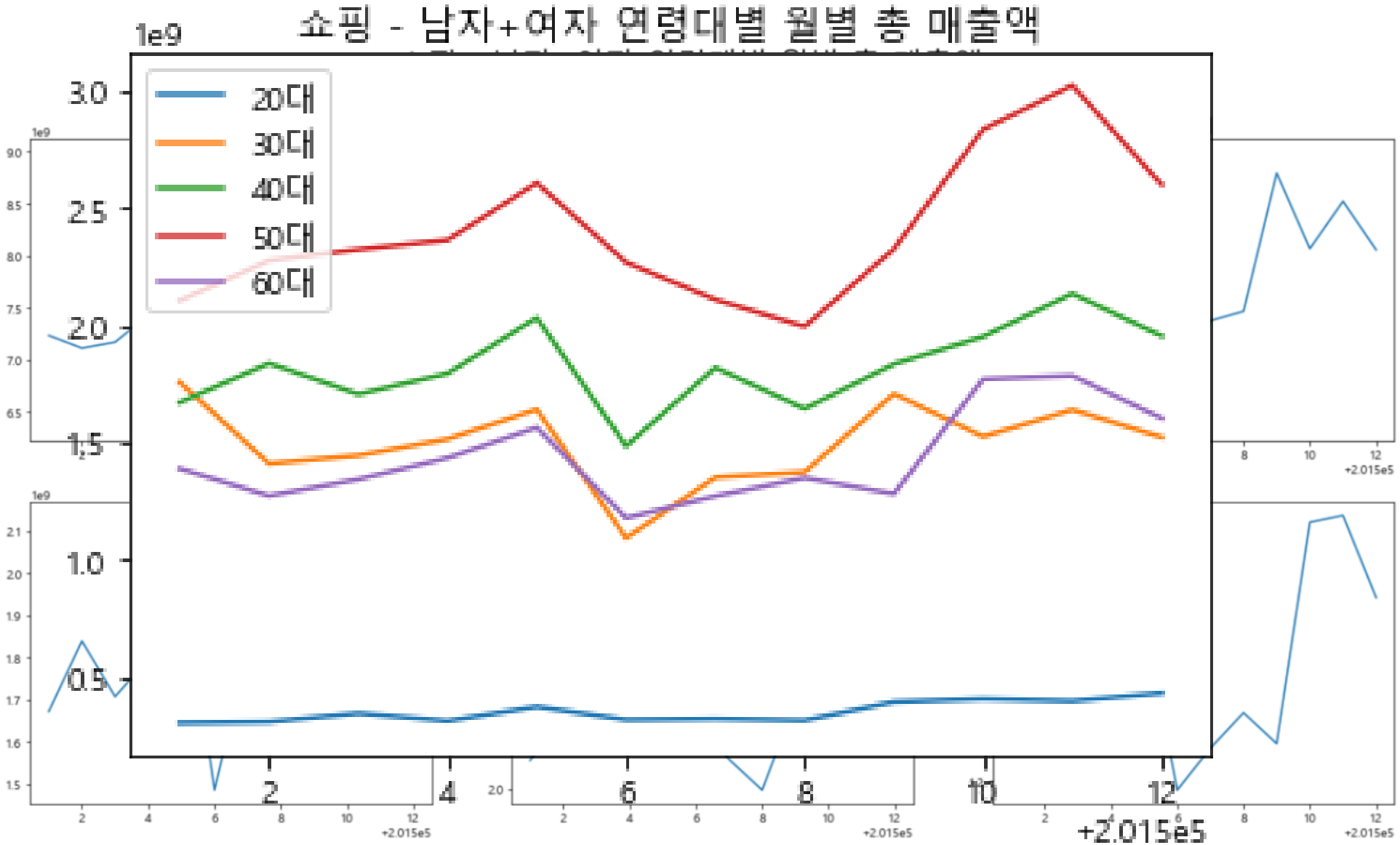




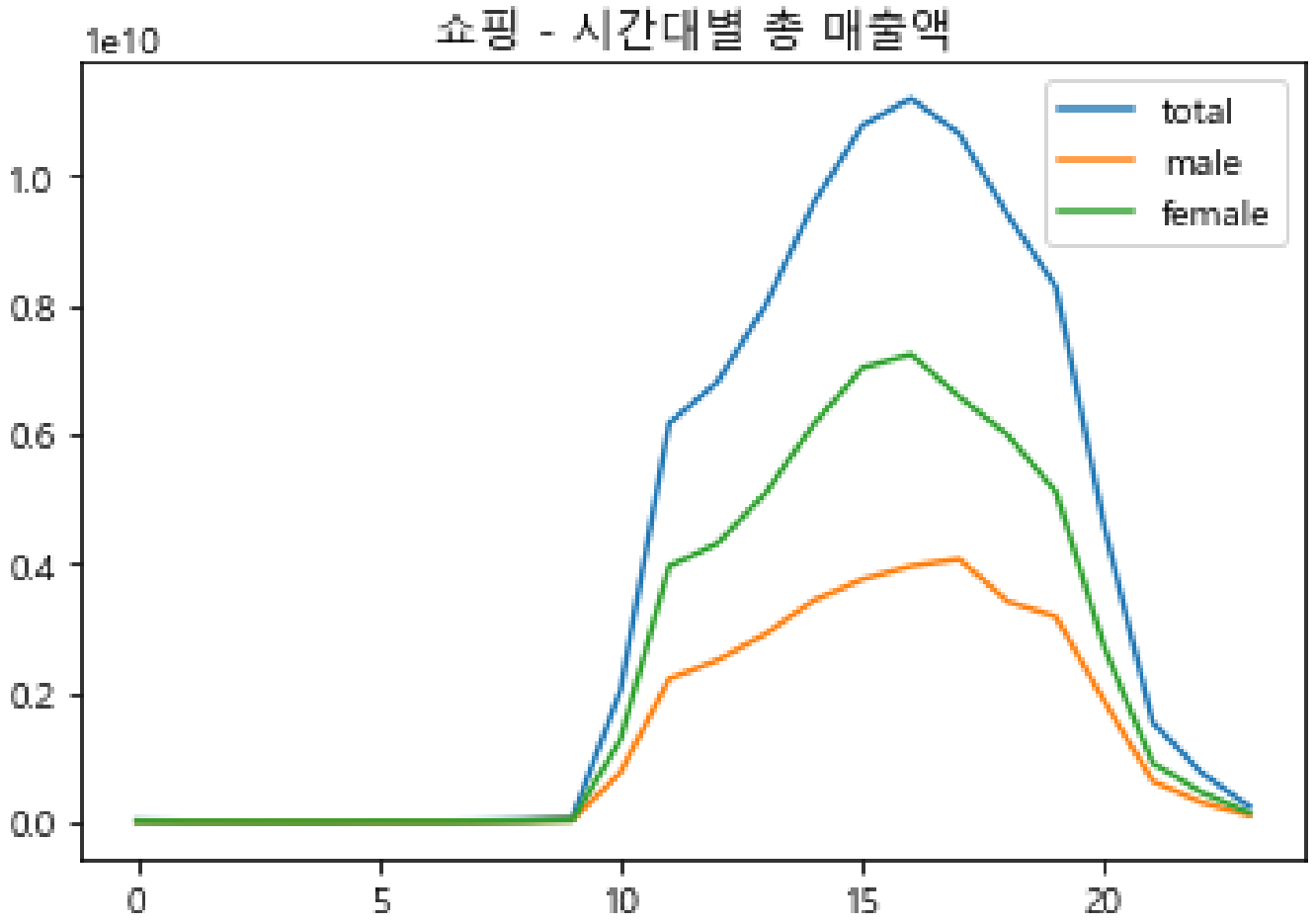




004
탐색적 데이터 분석



004
탐색적 데이터 분석



005
고객 세그먼트 분류

```
In [117]:

In [120]:

In [125]:

In [126]:

In [127]:
Out[127]:
```

```
In [25]:
# Arguments (x =
def RClass(x,p,d
    if x <= d[p]
        return 1
    elif x <= d[
        return 2
    else:
        return 3

# Arguments (x =
def FMClass(x,p,
    if x <= d[p]
        return 3
    elif x <= d[
        return 2
    else:
        return 1
```

```
In [26]:
rfmSegmentation[
rfmSegmentation[
rfmSegmentation[
```

```
In [27]:
rfmSegmentation[
```

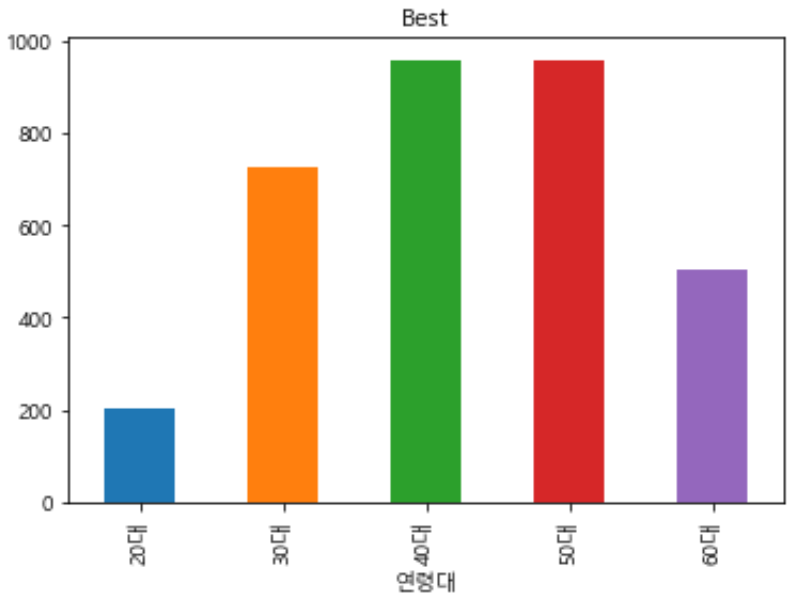
```
In [28]:
rfmSegmentation.
```

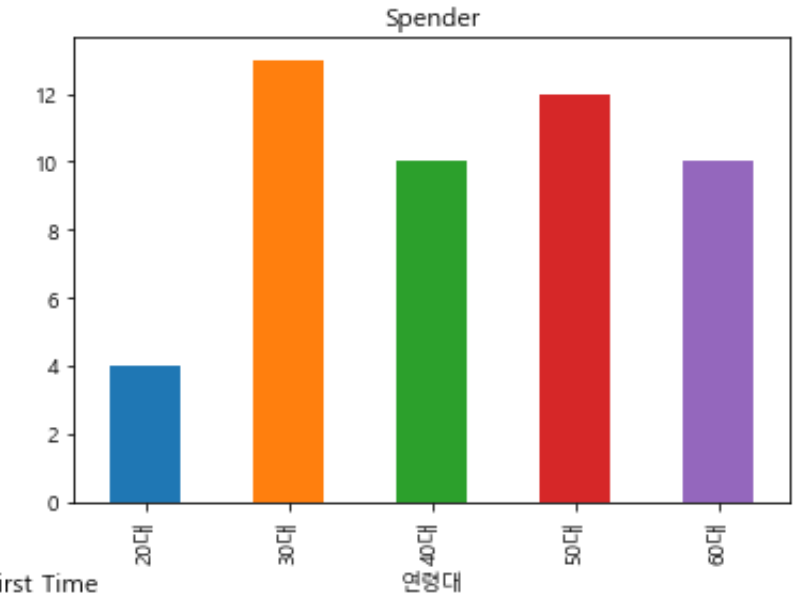
	Frequency	Monetary	Recency	R_Quartile	F_Quartile	M_Quartile	RFMClas
ID							
1	43	3695930	3	1	2	1	121
2	55	1979142	11	2	2	2	222
3	32	2272884	7	2	2	2	222
4	122	7970116	3	1	1	1	111
6	280	6458914	4	1	1	1	111
7	50	2650880	16	2	2	2	222
8	59	7635203	27	3	2	1	321
9	72	11241966	8	2	1	1	211
10	11	351800	9	2	3	3	233
11	18	3339228	52	3	2	2	322
13	8	502900	39	3	3	3	333
14	26	3020410	7	2	2	2	222
15	54	5268800	2	1	2	1	121
16	234	26818574	4	1	1	1	111
17	51	2546768	9	2	2	2	222
18	77	2576903	3	1	1	2	112
19	14	1467900	57	3	3	2	332
20	114	1455785	2	1	1	2	112

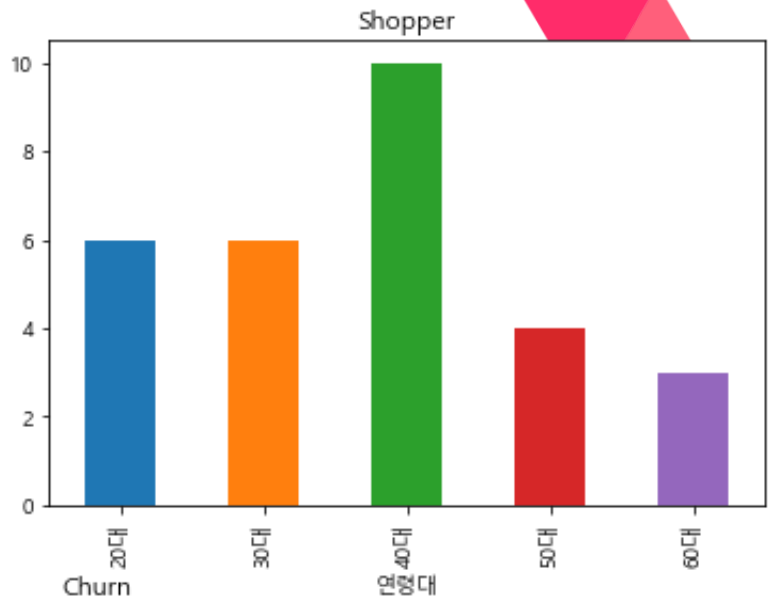
005
고객 세그먼트 분류

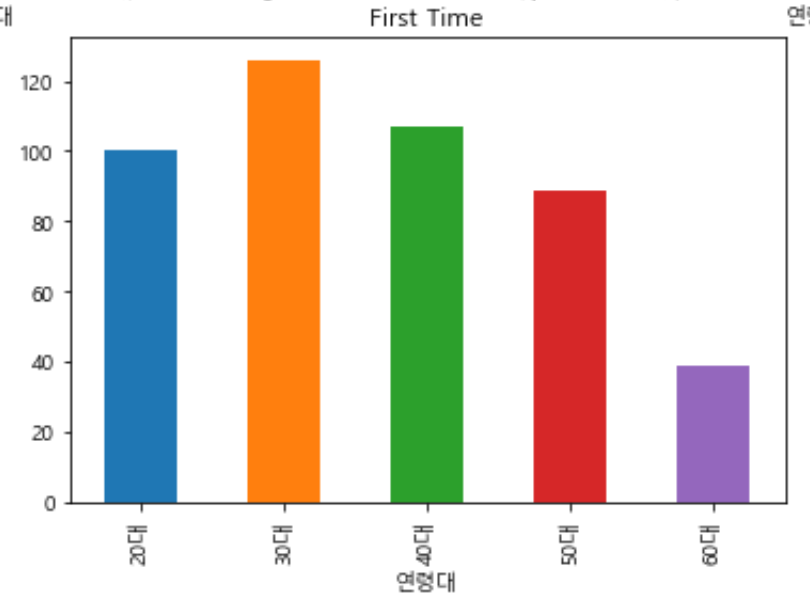
R↑ F↑ M↑	모든 지표에서 높은 우량고객	**Best**
R↓ F↓ M↑	한 번에 많이 쓰는 고객	**Spenders**
R↑ F↑ M↓	가성비를 추구하는 고객	**Shopper**
R↑ F↓ M↓	첫 방문 고객	**First Time**
R↓ F↑ M↑	중요한 고객인데 이탈한 고객	**Churn**

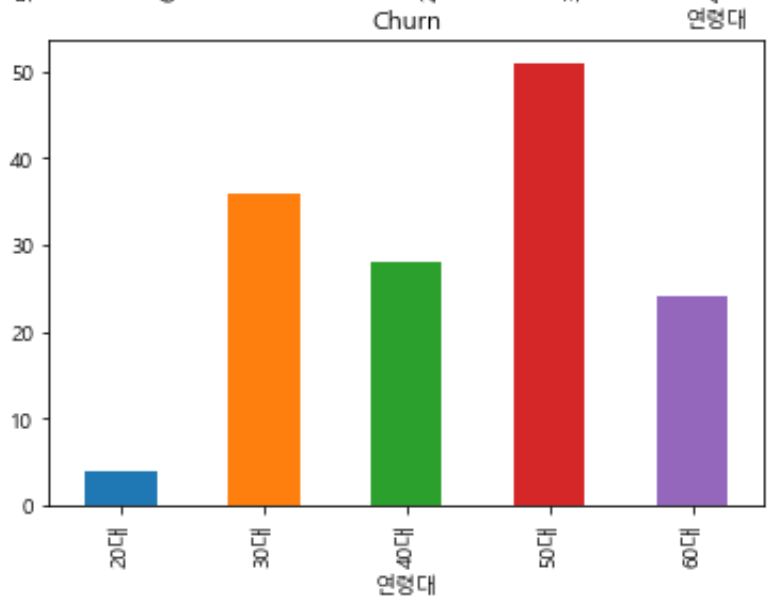
005
고객 세그먼트 분류











맞춤형 마케팅 제안

006
맞춤형 마케팅

Best R(최근성)↑ F(구매빈도)↑ M(구매금액)↑

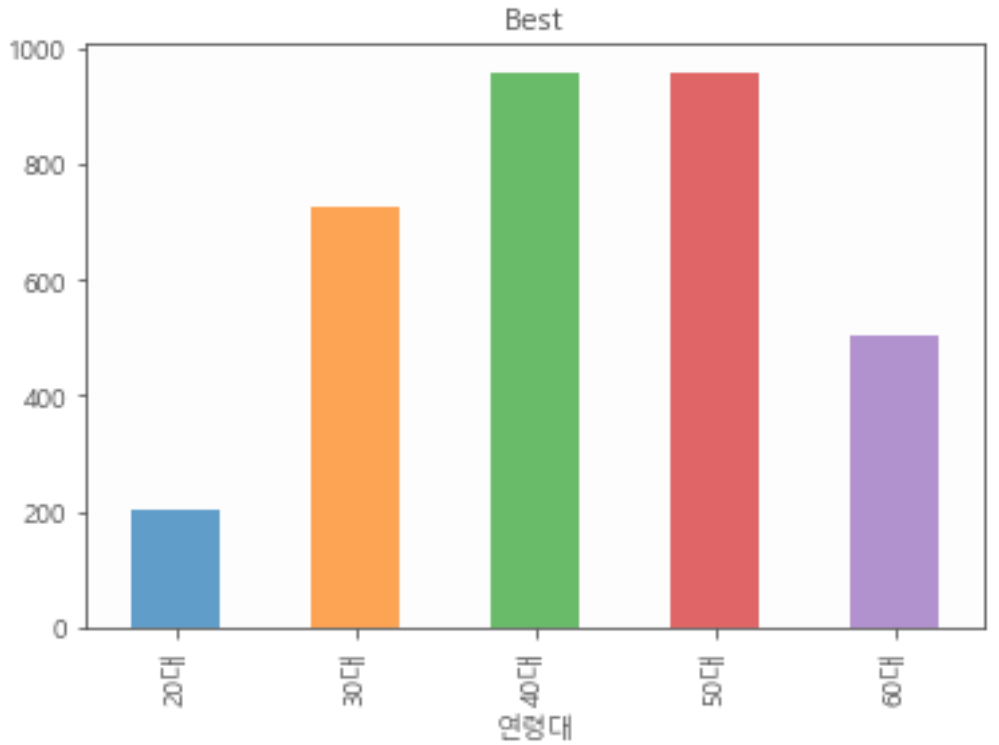
모든 측면에서 높은 비율을 보이는 '우량고객'

가격적인 측면 보다 서비스 측면으로 충성심 유지하고 감성을 자극

VIP 멤버쉽,
라운지 제공

공연, 문화시설
이용 티켓 제공

손편지



006
맞춤형 마케팅

Spender R(최근성)↓ F(구매빈도)↓ M(구매금액)↑

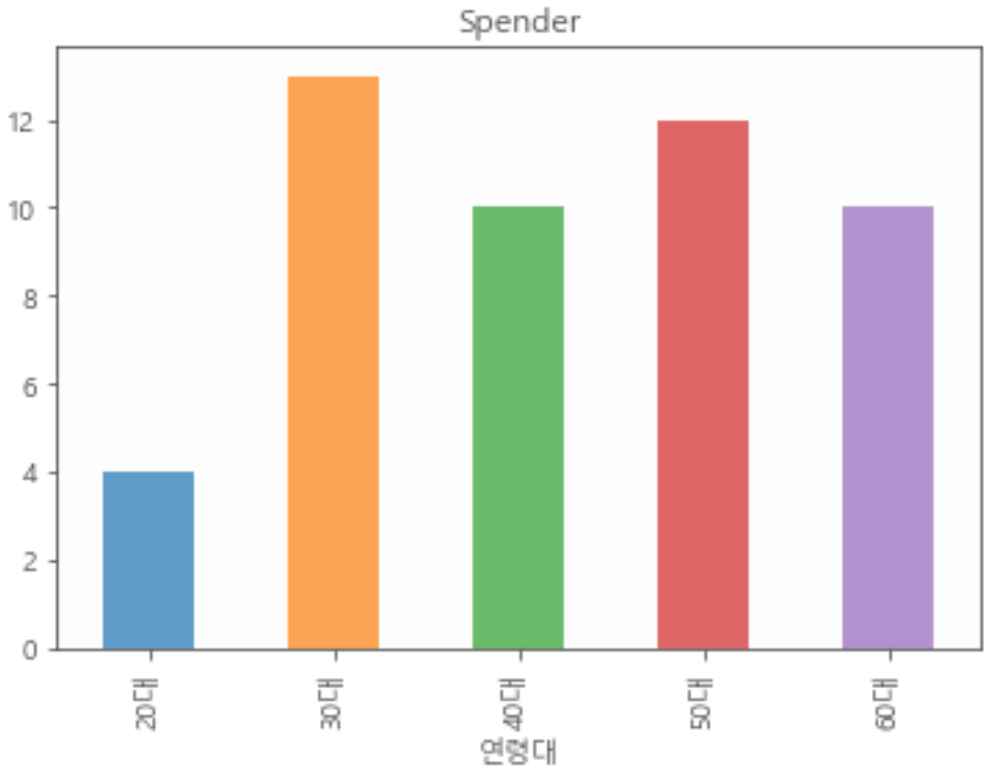
최근성과 구매빈도는 낮지만 구매금액이 높은 고객군

접근성이 낮은 고객

분석결과 특정 브랜드의 화장품을 찾는 여성 고객

온-오프라인
활용한
멀티채널 전략

특정 브랜드의
상품 추천



006
맞춤형 마케팅

Shopper R(최근성)↑ F(구매빈도)↑ M(구매금액)↓

40대 result. 식품

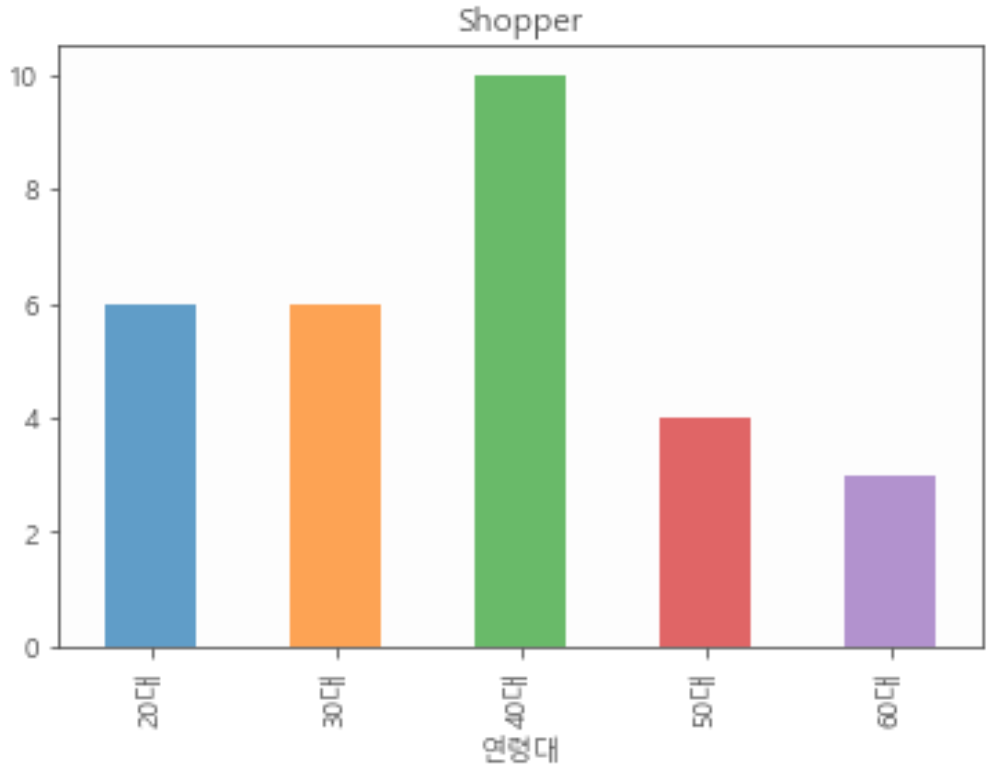
제품의 질과 가격 등을 꼼꼼히 평가

교육비와 내집 마련 지출이 집중되기 때문에 구매금액 저하, 가성비에 집중

번들링 상품

가성비 상품

Mother's Day



006

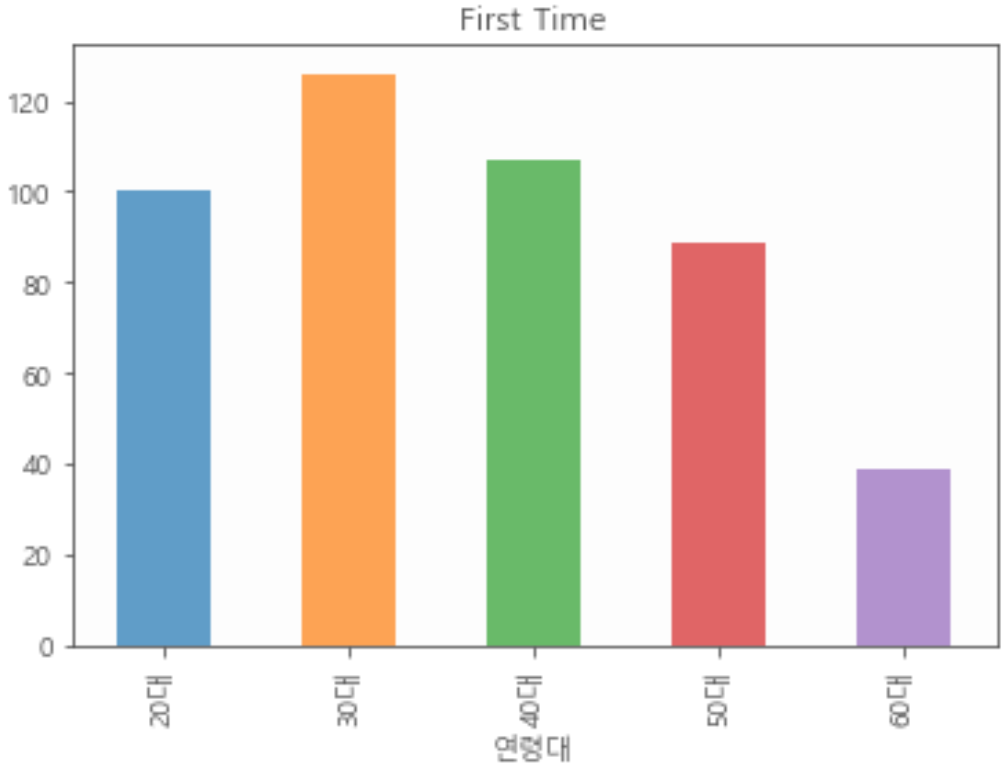
맞춤형 마케팅

First Time R(최근성)↑ F(구매빈도)↓ M(구매금액)↓

최근성이 높지만 구매빈도와 구매금액이 낮은 '첫 방문고객'
단골고객과 재 방문율을 높이기 위한 마케팅 전략
구매금액을 올리기 위한 구매유도

고객과의 커뮤니케이션 관계 구축

특정금액 이상을 구매시 상품권 증정



006
맞춤형 마케팅

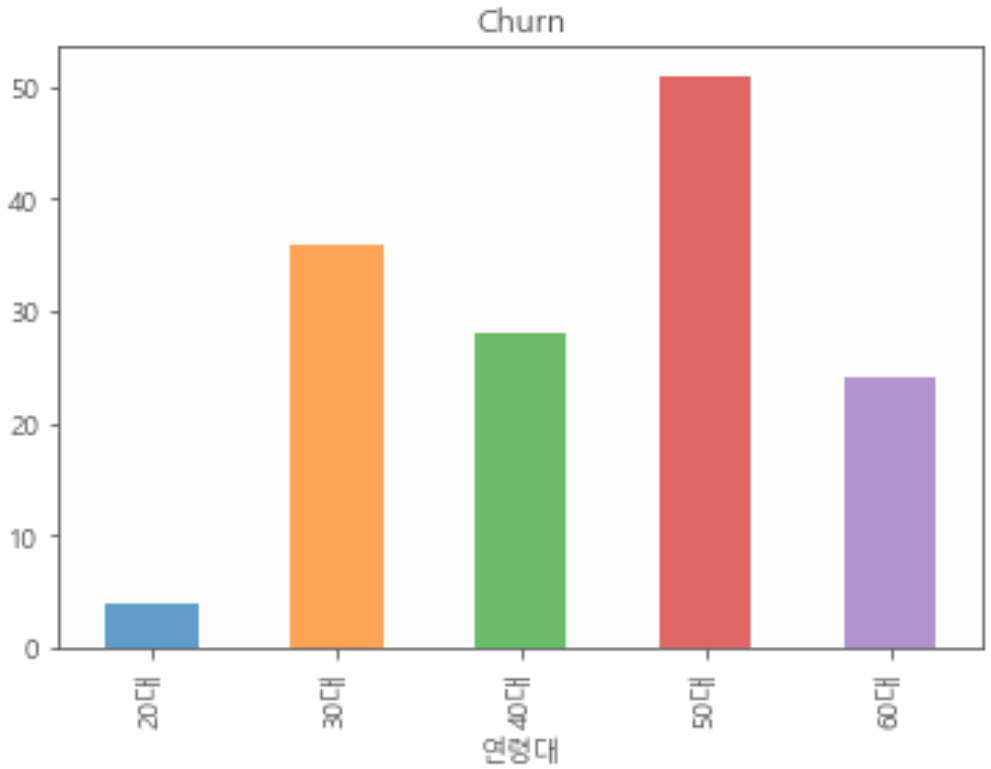
Churn R(최근성)↓ F(구매빈도)↑ M(구매금액)↑

중요한 고객이지만 최근성이 떨어지는 '이탈 고객'
이들을 다시 유효고객으로 전환한다면 매출의 상승을 기대할 수 있음
명확한 기준을 통해 매출에 기여도가 높은 고객을 선별하여 마케팅

이탈 고객을 위한 전담CS팀

과거 데이터 바탕 맞춤 상품 제안

이탈 고객을 위한 할인 혜택



수업시간에 배운 것
우리 데이터에
적용해보기

007
RFM + K-means를 활용한 고객 clustering & segmentation

RFM 변수 만들기

```
316  ## R
317
318  s_R <- shopping %>% group_by(ID) %>%
319    summarise(Recency = max(DE_DT))
320
321  # 2014년 구매고객 제외(2건)
322  s_R <- s_R %>% filter(!Recency < 20150101)
323
324  # 날짜의 차이를 구하기 위해 데이터 변환
325  s_R$Recency <- as.character(s_R$Recency)
326  s_R$Recency <- as.Date(s_R$Recency, format="%Y%m%d")
327
328  lastday <- "20151231"
329  lastday <- as.Date(lastday, format="%Y%m%d")
330  s_R$Recency <- lastday - s_R$Recency
331  s_R$Recency <- as.numeric(s_R$Recency)
332
333  ## F
334  ## 횟수만 알면 되니 영수증 중복 삭제
335  shopping_tmp <- shopping[!duplicated(shopping$R_NUM)]
336  length(unique(shopping_tmp$R_NUM))
337
338  s_F<- shopping_tmp %>% group_by(ID) %>%
339    summarise(Frequency = n())
340
341  ## M
342  s_M <- shopping %>% group_by(ID) %>%
343    summarise(Monetary = sum(BUY_AM))
344
```

```
> s_RFM %>% arrange(desc(Monetary)) %>% head(10)
      ID Recency Frequency  Monetary
1  13087       0       175 611749918
2   7278       0       215 342051200
3   2807       3       110 323160907
4   9038       0       262 310334084
5   6663       0       173 284605677
6   2363       0       116 273316980
7   3020       1       315 265696983
8   7039       0       310 240652200
9  11447       1        57 229759930
10  1584      13        54 214148828
```

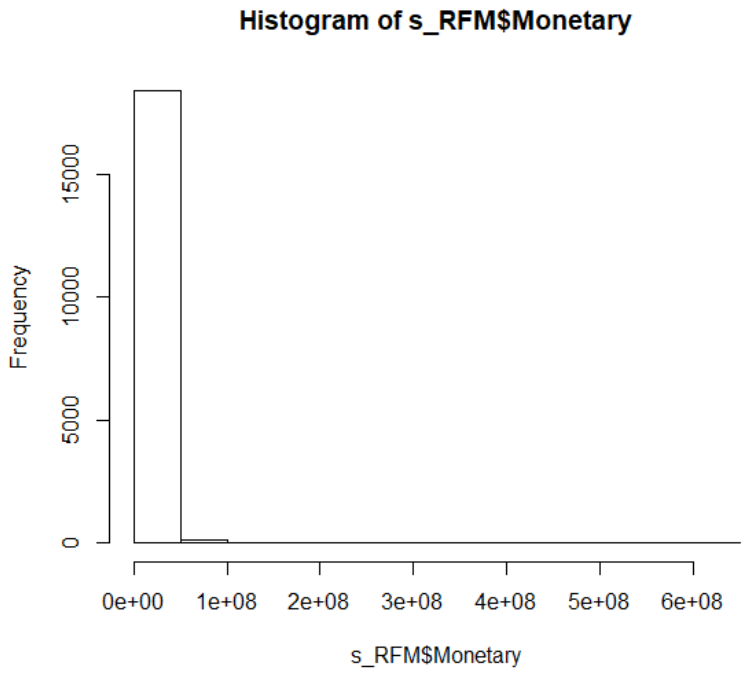
```
> summary(s_RFM)
       ID            Recency          Frequency         Monetary
 Min.   :    1   Min.   :  0.00   Min.   :   1.00   Min.   :       400
 1st Qu.: 4813   1st Qu.:  4.00   1st Qu.:  11.00   1st Qu.:    486975
 Median : 9620   Median : 11.00   Median :  35.00   Median :   1846536
 Mean   : 9749   Mean   : 41.67   Mean   :  56.69   Mean   :   4872003
 3rd Qu.:14638   3rd Qu.: 42.00   3rd Qu.:  82.00   3rd Qu.:   4875378
 Max.   :20000   Max.   :364.00   Max.   :1192.00   Max.   :611749918
```

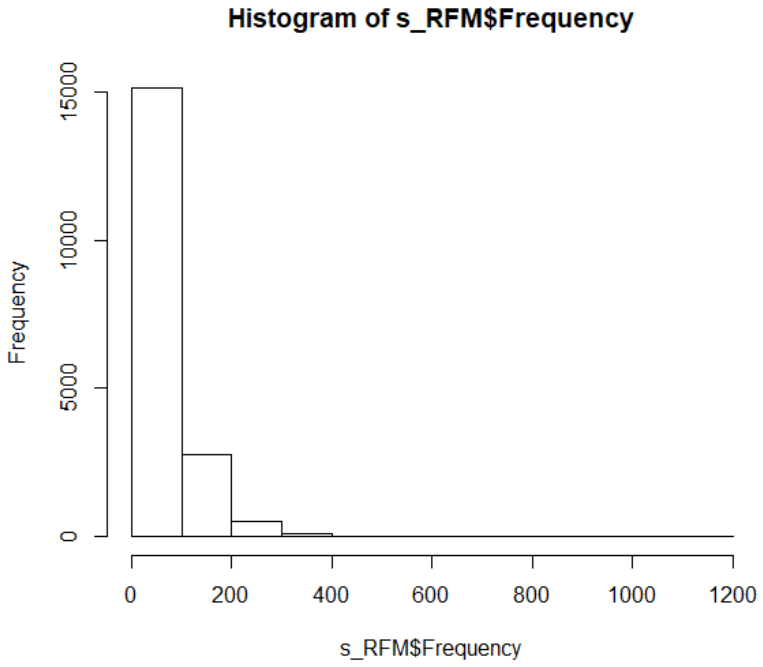
각각의 고객ID에 RFM을 부여

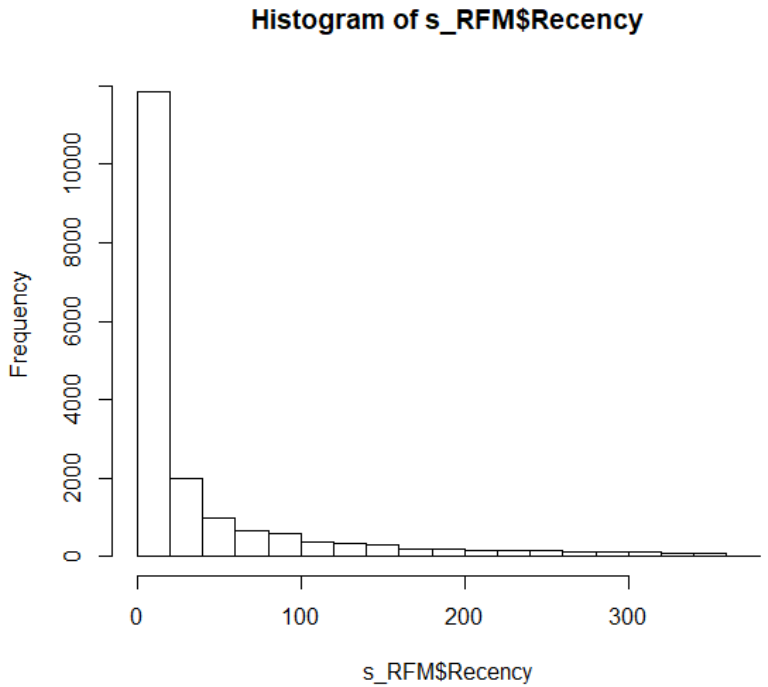
그런데 요약 통계량을 본 결과 매우 치우쳐진 분포를 보일 것으로 예상

007
RFM + K-means를 활용한 고객 clustering & segmentation

RFM 분포의 문제





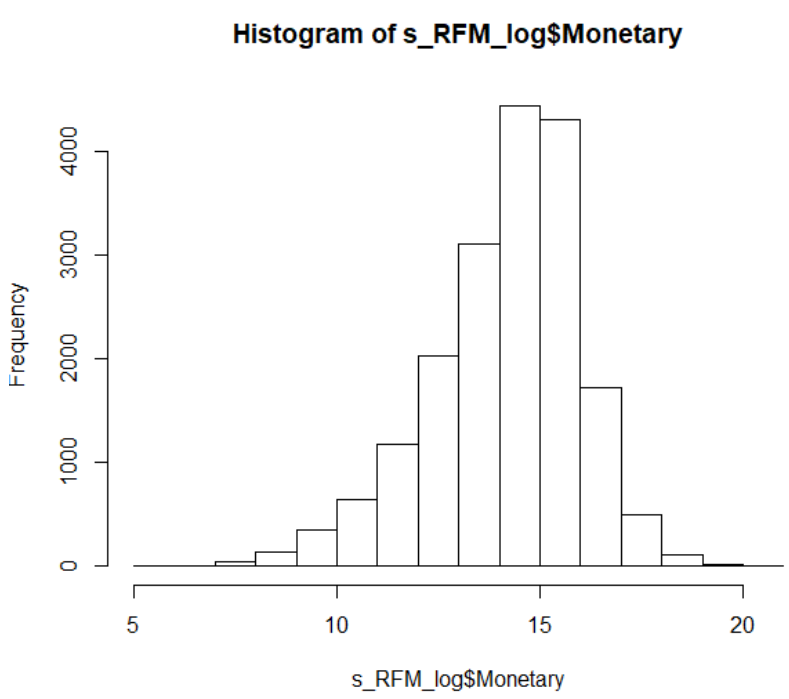


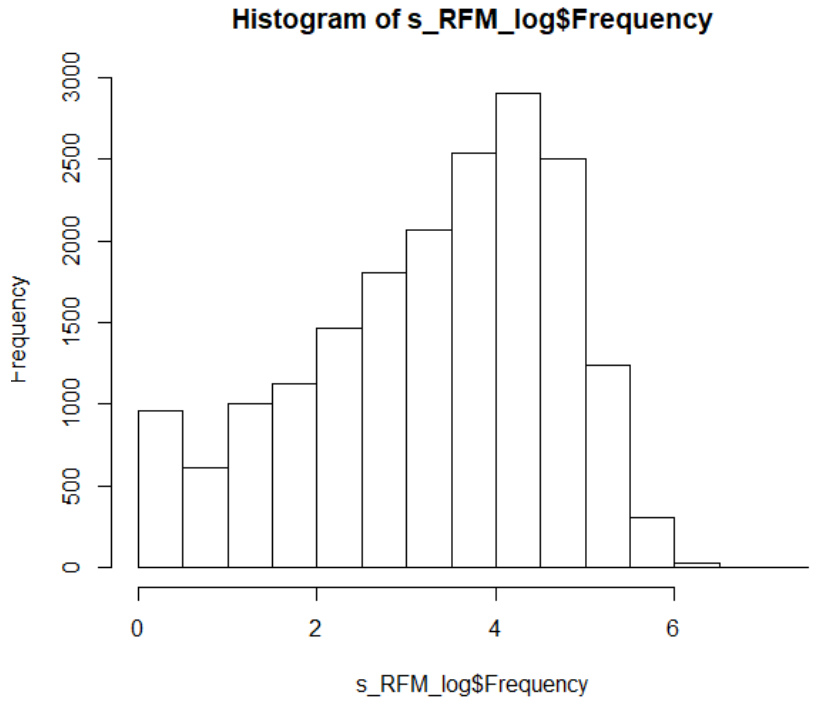
히스토그램을 통해 확인해보니 매우 치우쳐진 분포를 보인다

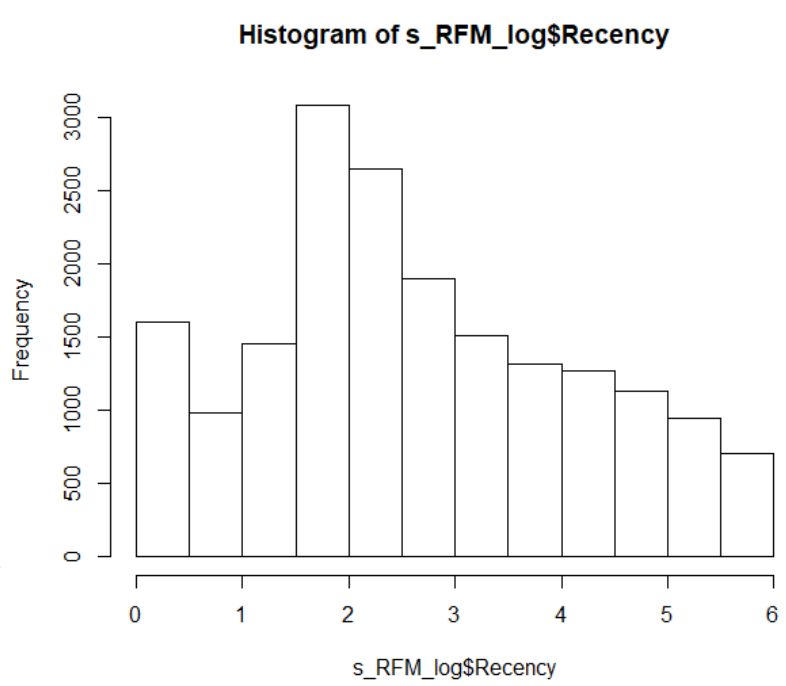
이는 파레토의 법칙을 떠올리게 한다..?

007
RFM + K-means를 활용한 고객 clustering & segmentation

자연로그를 취한 RFM의 분포







K-means를 통해 clustering하려면 거리를 계산 해야 하는데

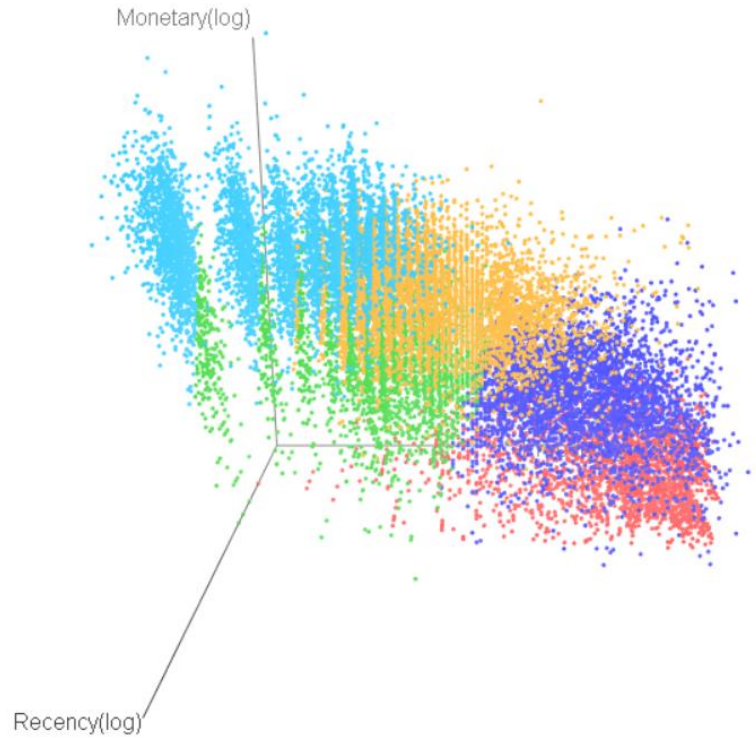
억단위의 큰 숫자를 쓰는 것은 옳지 않고 / 정규분포화를 위해서 자연로그를 취함

007
RFM + K-means를 활용한 고객 clustering & segmentation

k-means 3차원 시각화 해보기

```
## K-MEANS를 통해 클러스터링 해보자
## 우선 다섯개의 클러스터로 나눠서 3차원 시각화해서 보자

install.packages("car")
install.packages("rgl")
library(car)
library(rgl)

viz_3d <- s_RFM_log
RFM_km <- kmeans(s_RFM_log[2:4],
                 centers = 5)

## 이런것들을 통해 클러스터의 특징을 확인할 수 있다
RFM_km$centers
RFM_km$size
RFM_km$cluster

## 기존의 데이터프레임에 cluster를 추가하자
viz_3d[,("cluster")] <- RFM_km$cluster
viz_3d$cluster <- as.factor(viz_3d$cluster)
table(viz_3d$cluster)
```

```
colors <- c('red','orange','green3','deepskyblue',
            'blue','darkorchid4','violet','pink1','tan3','blac

scatter3d(x=viz_3d$Recency,
          y=viz_3d$Frequency,
          z=viz_3d$Monetary,
          groups=viz_3d$cluster,
          xlab="Frequency(log)",
          ylab="Monetary(log)",
          zlab="Recency(log)",
          surface.col=colors,
          axis.scales = FALSE,
          surface = FALSE,
          fit = "smooth",
          grid = TRUE,
          axis.col = c("black", "black", "black"))
```



Cluster를 몇 개로 쪼갤지는 elbow curve를 그려보거나 비즈니스상 요구를 확인해야 하지만

우선 시각화해보기 위해 5개로 나눠 3차원 시각화를 해봄

이런 방식으로 거리에 따라 클러스터가 묶인다는 것을 확인

https://towardsdatascience.com/apply-rfm-principles-to-cluster-customers-with-k-means-fef9bcc9ab16

007
RFM + K-means를 활용한 고객 clustering & segmentation

고객 segmentation 만들기

```
462 ## cluster를 몇개로쪼개야할지는 elbow curve를 그려보던지 비즈니스상 요구를 확인해야하지!
463 ## 다양한 레벨로 나누기 위해 우리는 15개로 가정하고 해보자
464
465 seg <- s_RFM_log
466 RFM_km2 <- kmeans(seg[2:4],
467                   centers = 15)
468
469 seg[,("cluster")] <- RFM_km2$cluster
470 seg$cluster <- as.factor(seg$cluster)
471 table(seg$cluster)
472
473 cluster_model <- RFM_km2$centers
474 cluster_model
475
476 cluster_name <- data.frame(cluster=c(1:15))
477 cluster_name
478
479 cluster_size <- as.data.frame(RFM_km2$size)
480 cluster_size <- rename(cluster_size, size = `RFM_km2$size`)
481
482 cluster_model <- cbind(cluster_model,
483                        cluster_name)
484
485 cluster_model <- cbind(cluster_model,
486                        cluster_size)
487
488 cluster_model <- cluster_model %>% select(cluster, size, Recency, Frequency, Monetary)
489 cluster_model <- rename(cluster_model, R = Recency)
490 cluster_model <- rename(cluster_model, F = Frequency)
491 cluster_model <- rename(cluster_model, M = Monetary)
499 cluster_model <- cluster_model %>% mutate(type= ifelse(R<quantile(cluster_model$R, probs = 0.4) &
500                                                        F>quantile(cluster_model$R, probs = 0.6)&
501                                                        M<quantile(cluster_model$R, probs = 0.4),"SHOPPER",
502                                                 ifelse(R<quantile(cluster_model$R, probs = 0.1) &
503                                                        F>quantile(cluster_model$R, probs = 0.9)&
504                                                        M>quantile(cluster_model$R, probs = 0.9), "VIP",
505                                                 ifelse(R<quantile(cluster_model$R, probs = 0.5) &
506                                                        F>quantile(cluster_model$R, probs = 0.5) &
507                                                        M>quantile(cluster_model$R, probs = 0.5), "BEST",
508                                                 ifelse(R<quantile(cluster_model$R, probs = 0.25) &
509                                                        F<quantile(cluster_model$R, probs = 0.25) &
510                                                        M<quantile(cluster_model$R, probs = 0.25), "NEW",
511                                                 ifelse(R>quantile(cluster_model$R, probs = 0.5) &
512                                                        F>quantile(cluster_model$R, probs = 0.5)  &
513                                                        M>quantile(cluster_model$R, probs = 0.5), "CHURN",
514                                                 ifelse(R>quantile(cluster_model$R, probs = 0.3) &
515                                                        F<quantile(cluster_model$R, probs = 0.3)&
516                                                        M>quantile(cluster_model$R, probs = 0.8), "SPENDER","UNCERTAIN"))))))))
```

```
> cluster_model
   cluster size         R         F         M       type
1        1  897 4.3309142 1.5331021 11.788437   SPENDER
2        2 1166 2.6298699 2.2648558 12.740032 UNCERTAIN
3        3  895 0.3791868 5.1271860 16.827919       VIP
4        4  924 5.3181657 0.5617639 11.381961   SPENDER
5        5  958 2.0051480 4.6736918 16.840602      BEST
6        6 1373 4.6355428 2.0857262 13.086306 UNCERTAIN
7        7 2230 1.7920064 4.6649989 15.580009      BEST
8        8 1568 3.9714884 3.0176228 14.178430     CHURN
9        9 1511 0.3376145 4.5473856 15.183692      BEST
10      10  385 2.5137209 0.9102434 11.020850   SPENDER
11      11  632 0.5900448 3.1777337 13.631651      BEST
12      12 1719 2.4056542 3.1701087 13.896556      BEST
13      13 1580 3.0880586 3.9388889 15.323046     CHURN
14      14  624 5.0434632 0.2308242  9.341309   SPENDER
15      15 2088 1.9565500 3.9594053 14.663841      BEST
```

디테일 하게 분류하기 위해 (임의대로) 15개의 clustering으로 나누어 봄

그후 클러스터를 회사의 사정에 맞춰 고객분류를 하면 됨

오른쪽처럼 다양하게 분류할 수 있음

https://towardsdatascience.com/apply-rfm-principles-to-cluster-customers-with-k-means-fef9bcc9ab16

007
RFM + Association rule을 활용한 상품 recommendation

장바구니분석

```
409 # 장바구니분석
410 library(arules)
411
412 ## 영수증별로 그룹(장바구니분석을 위한)
413 length(unique(shopping$R_NUM)) # 현재 영수증 100만개
414
415
416 ### 중분류
417 reciept_M <- shopping %>% select(R_NUM, PD_M_NM)
418
419 rM_list<- split(reciept_M$PD_M_NM,reciept_M$R_NUM)
420 rM_list[3]
421 rM_trans <- as(rM_list,"transactions")
422 summary(rM_trans)
423
424 # 가장 많이팔린건?
425 itemFrequencyPlot(rM_trans,topN=20)
426 itemFrequencyPlot(rM_trans,topN=20,type="absolute")
427
428 rM_rules <- apriori(rM_trans,
429                     parameter = list(supp=0.001,
430                                      conf=0.5))
431
432 inspect(sort(rM_rules,by="lift")[1:30])
433
```

```
> inspect(sort(rM_rules,by="lift")[1:30])
     lhs                                rhs          support     confidence lift     count
[1]  {공병공박스}                       => {소주}     0.001260120 0.8123850  53.23854 1325
[2]  {기타음주류}                       => {소주}     0.004336716 0.7140620  46.79507 4560
[3]  {기타음주류,맥주}                  => {소주}     0.001712812 0.5605353  36.73391 1801
[4]  {수산품,축산가공}                  => {기타}     0.001172625 0.5059499  35.10160 1233
[5]  {가공식품,농산물,축산가공}         => {기타}     0.001058501 0.5009001  34.75125 1113
[6]  {두부,양념채소,잎채소}             => {열매채소} 0.001023313 0.5915338  26.94114 1076
[7]  {버섯,양념채소,잎채소}             => {열매채소} 0.001103200 0.5852674  26.65574 1160
[8]  {두부,버섯,잎채소}                 => {열매채소} 0.001130780 0.5563875  25.34042 1189
[9]  {버섯,양념채소,열매채소}           => {잎채소}   0.001103200 0.5362922  24.78700 1160
[10] {열매채소,차별화돼지고기}          => {잎채소}   0.001029019 0.5174558  23.91640 1082
[11] {샐러드채소,양념채소}              => {열매채소} 0.001363783 0.5214545  23.74941 1434
[12] {나물,열매채소}                    => {잎채소}   0.001194499 0.5042152  23.30443 1256
[13] {뿌리채소,콩나물}                  => {열매채소} 0.001018558 0.5097573  23.21666 1071
[14] {계란,콩나물}                      => {두부}     0.001526410 0.5177419  23.14129 1605
[15] {버섯,뿌리채소}                    => {열매채소} 0.001922991 0.5063862  23.06313 2022
[16] {양념채소,잎채소}                  => {열매채소} 0.002498367 0.5024866  22.88553 2627
[17] {수산품,축산가공}                  => {농산물}   0.001828839 0.7890849  20.59913 1923
[18] {육류,축산가공}                    => {농산물}   0.001767972 0.7675475  20.03690 1859
[19] {수산품,젓갈/반찬}                 => {농산물}   0.001288651 0.7655367  19.98441 1355
[20] {기타,수산품}                      => {농산물}   0.002582058 0.7641430  19.94802 2715
[21] {기타,젓갈/반찬}                   => {농산물}   0.001801259 0.7630943  19.92065 1894
[22] {가공식품,기타,축산가공}           => {농산물}   0.001058501 0.7607656  19.85985 1113
[23] {젓갈/반찬,축산가공}               => {농산물}   0.001447474 0.7594810  19.82632 1522
[24] {기타,축산가공}                    => {농산물}   0.002870221 0.7522433  19.63738 3018
[25] {육류,젓갈/반찬}                   => {농산물}   0.001174527 0.7475787  19.51561 1235
[26] {수산품,육류}                      => {농산물}   0.001791748 0.7310827  19.08498 1884
[27] {기타,육류}                        => {농산물}   0.002466031 0.7306283  19.07312 2593
[28] {씨리얼}                           => {우유}     0.001099395 0.5688976  17.77044 1156
[29] {가공식품,육류}                    => {농산물}   0.001530214 0.6670813  17.41422 1609
[30] {기타,일용잡화}                    => {농산물}   0.001408482 0.6635305  17.32152 1481
```

우선 영수증별로 대분류, 중분류, 소분류별로 돌려봄

노이즈가 너무 커서 딱히 재밌는 결과는 나오지 않음

007
RFM + Association rule을 활용한 상품 recommendation

장바구니분석

```
463 ### shopping, notshopping 합쳐서 장바구니 분석해보자
464
465 head(notshopping)
466 head(shopping)
467
468 ns <- notshopping %>% select(ID, BIZ_UNIT)
469 ns <- rename(ns, PD_H_NM = BIZ_UNIT)
470 ss <- shopping %>% select(ID, PD_H_NM)
471 both <- bind_rows(ns,ss)
472 str(both)
473 head(both)
474
475 both_list<- split(both$PD_H_NM,both$ID)
476 both_trans <- as(both_list,"transactions")
477 itemFrequencyPlot(both_trans,topN=20)
478
479 new_rules <- apriori(both_trans,
480                      parameter = list(supp=0.2,
481                                       conf=0.7))
482
483 new_rules <- subset(new_rules, items %pin% c("Themepark"))
484
485 inspect(sort(new_rules,by="lift")[1:30])
486 inspect(new_rules)
```

```
> inspect(sort(new_rules,by="lift")[1:30])
     lhs                                  rhs          support   confidence lift     count
[1]  {Themepark,식품,음료}                => {과자}     0.2078914 0.8647292  1.614621 4136
[2]  {Themepark,음료}                     => {과자}     0.2255341 0.8543412  1.595225 4487
[3]  {Cafe,Themepark,식품,여성의류}       => {잡화}     0.2063835 0.9118366  1.559844 4106
[4]  {Themepark,과자,식품}                => {음료}     0.2078914 0.8944637  1.554315 4136
[5]  {Themepark,스포츠,식품}              => {잡화}     0.2008545 0.9059170  1.549718 3996
[6]  {Themepark,과자}                     => {음료}     0.2255341 0.8909849  1.548270 4487
[7]  {Cafe,Themepark,여성의류}            => {잡화}     0.2105554 0.9010540  1.541399 4189
[8]  {Themepark,식품,여성의류}            => {스포츠}   0.2067354 0.7478182  1.531272 4113
[9]  {Themepark,식품,잡화}                => {스포츠}   0.2008545 0.7469159  1.529425 3996
[10] {Themepark,식품,여성의류}            => {잡화}     0.2466951 0.8923636  1.526533 4908
[11] {Themepark,스포츠}                   => {잡화}     0.2051269 0.8893005  1.521293 4081
[12] {Themepark,스포츠,식품}              => {여성의류} 0.2067354 0.9324416  1.516218 4113
[13] {Themepark,여성의류}                 => {스포츠}   0.2125157 0.7368421  1.508797 4228
[14] {Cafe,Themepark,식품,잡화}           => {여성의류} 0.2063835 0.9277000  1.508508 4106
[15] {Themepark,잡화}                     => {스포츠}   0.2051269 0.7350504  1.505128 4081
[16] {Themepark,스포츠}                   => {여성의류} 0.2125157 0.9213336  1.498155 4228
[17] {Cafe,Themepark,잡화}                => {여성의류} 0.2105554 0.9208617  1.497388 4189
[18] {Themepark,여성의류}                 => {잡화}     0.2524252 0.8752178  1.497202 5022
[19] {Themepark,식품,잡화}                => {여성의류} 0.2466951 0.9173832  1.491732 4908
[20] {Cafe,Themepark,식품}                => {잡화}     0.2224680 0.8704031  1.488966 4426
[21] {Taxfree,Themepark,식품}             => {잡화}     0.2002513 0.8627111  1.475807 3984
[22] {Themepark,잡화}                     => {여성의류} 0.2524252 0.9045389  1.470846 5022
[23] {Themepark,식품,음료}                => {잡화}     0.2055793 0.8551119  1.462807 4090
[24] {Cafe,Themepark,여성의류,잡화}       => {식품}     0.2063835 0.9801862  1.456371 4106
[25] {Fastfood,Themepark,식품}            => {잡화}     0.2029153 0.8509696  1.455722 4037
[26] {Themepark,식품,여성의류,잡화}       => {Cafe}     0.2063835 0.8365933  1.454897 4106
[27] {Themepark,스포츠,잡화}              => {식품}     0.2008545 0.9791718  1.454864 3996
[28] {Themepark,여성의류,잡화}            => {식품}     0.2466951 0.9772999  1.452082 4908
[29] {Themepark,여성의류,잡화}            => {Cafe}     0.2105554 0.8341298  1.450613 4189
[30] {Taxfree,Themepark,잡화}             => {식품}     0.2002513 0.9755142  1.449429 3984
```

쇼핑과 쇼핑 외를 합쳐서 ID별로 돌려봄

한번 theme park를 가는 사람들의 연관분석을 해봄

007
RFM + Association rule을 활용한 상품 recommendation

VIP고객의 장바구니분석

```
593 ## 3. VIP
594
595 seg <- seg %>% filter(cluster==3)
596 VIP <- merge(seg,shopping,by="ID")
597 VIP <- VIP %>% select(-Recency, -Frequency, -Monetary)
598 length(unique(VIP$ID))  # 895명
599
600 #
601 VIP_R <- VIP %>% select(ID, PD_S_NM)
602 SR_list <- split(VIP_R$PD_S_NM,VIP_R$ID)
603 SR_list[0:5]
604
605 #
606 SR_trans <- as(SR_list,"transactions")
607 summary(SR_trans)
608 image(sample(SR_trans,1000,replace=FALSE))
609 itemFrequencyPlot(SR_trans,topN=20,type="absolute")
610
611
612 SR_rules <- apriori(SR_trans, parameter =
613                       list(support=0.5,
614                            confidence = 0.8))
```

```
> inspect(sort(SR_rules,by="lift"))
     lhs                    rhs              support   confidence lift     count
[1]  {채소}              => {청과}          0.5162011 0.9041096  1.324350 462
[2]  {감자스낵}          => {일반스낵}      0.5687151 0.8671210  1.270169 509
[3]  {일반스낵}          => {감자스낵}      0.5687151 0.8330606  1.270169 509
[4]  {상품군미지정}      => {청과}          0.5072626 0.8454376  1.238407 454
[5]  {떠먹는요구르트}    => {일반스낵}      0.5106145 0.8339416  1.221567 457
[6]  {떠먹는요구르트}    => {어묵}          0.5061453 0.8266423  1.193298 453
[7]  {상품군미지정}      => {기초 화장품}   0.5083799 0.8472998  1.105442 455
[8]  {브랜드샵}          => {기초 화장품}   0.5128492 0.8330309  1.086826 459
[9]  {한식델리}          => {기초 화장품}   0.5128492 0.8211091  1.071272 459
[10] {청과}              => {기초 화장품}   0.5597765 0.8199673  1.069782 501
[11] {수입식품}          => {기초 화장품}   0.5385475 0.8114478  1.058667 482
[12] {전문베이커리}      => {기초 화장품}   0.5575419 0.8074434  1.053443 499
[13] {식당가 한식}       => {기초 화장품}   0.5743017 0.8056426  1.051094 514
```

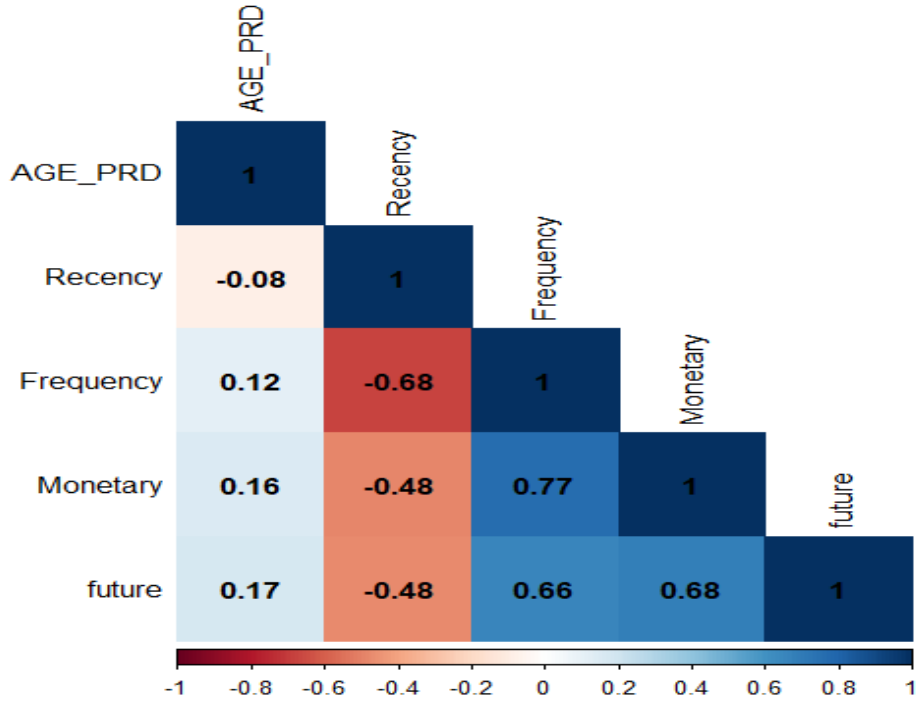
클러스터링에서 VIP고객들의 장바구니분석

좀 다르기는 하지만 상품추천을 하기엔 적합하지 않을 듯 함..

007
RFM + Regression을 활용한 미래 구매액 prediction

RMF2 테이블 / 상관행렬 HEATMAP

	ID	AGE_PRD	Recency	Frequency	Monetary	future
1	1	60	18	23	1183830	1830000
2	10	60	80	6	129000	193900
3	100	60	1	56	970180	2469644
4	1000	60	37	20	676660	1311872
5	10001	30	1	28	570130	2036640
6	10002	40	53	3	146400	26000
7	10003	50	27	6	66760	296610
8	10004	50	4	60	3483770	4245218
9	10005	50	7	44	1509670	17706348
10	10006	40	43	16	885180	3670704



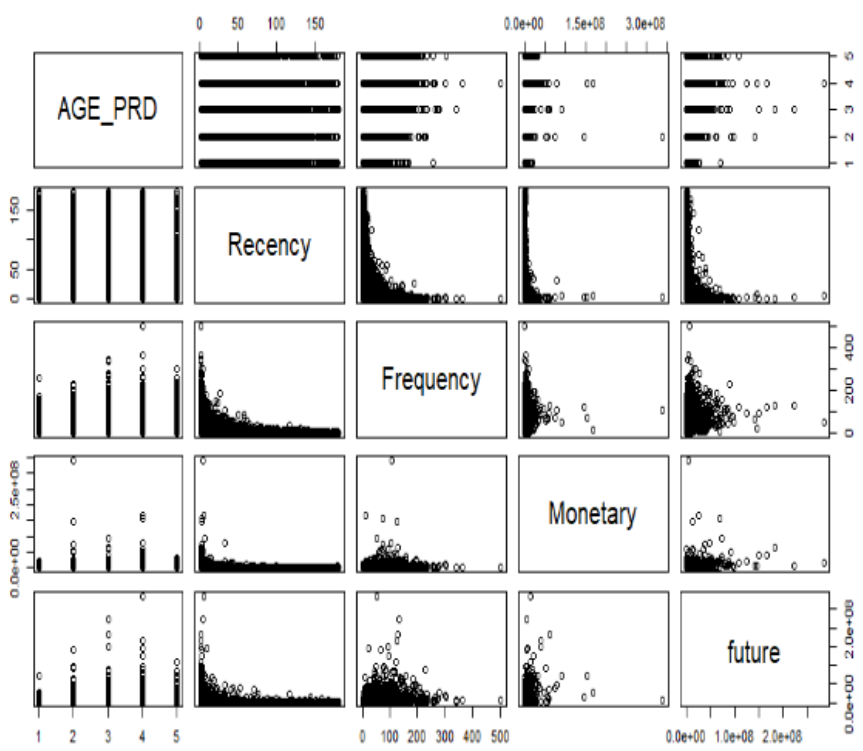
1년치 Transaction data를 절반으로 쪼갠 후

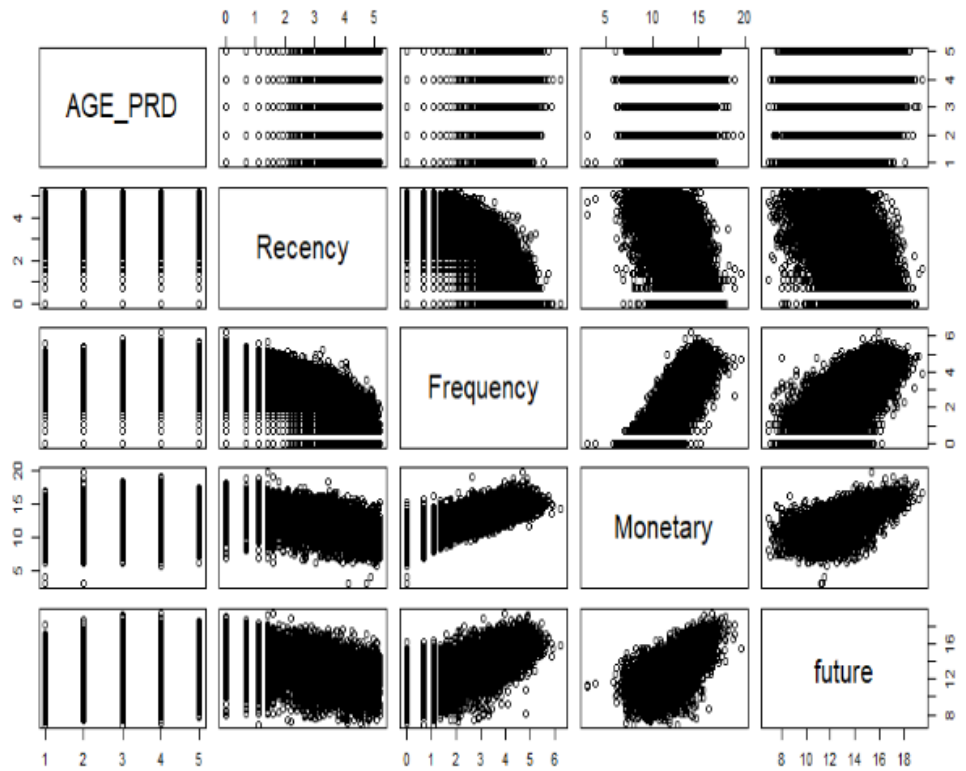
2분기 까지의 RFM + age를 통해 3,4분기의 구매액을 예측 해보자

전처리과정은 앞과 유사하므로 생략

007
RFM + Regression을 활용한 미래 구매액 prediction

로그를 취하기 전후의 plot





왼쪽 산점도 보면 확실히 20대는 미래구매액이 좀 적은 것 같고

오른쪽을 통해 Frequency, Monetary가 Future과 선형관계에 있음을 알 수 있다

007
RFM + Regression을 활용한 미래 구매액 prediction

트레이닝셋과 테스트셋 분리

(validation set 생략..)

```
765  set.seed(129)
766  n <- nrow(RFM2)
767  idx <- 1:n
768  training_idx <- sample(idx, n*0.7)
769  test_idx <-setdiff(idx,training_idx)
770  training <- RFM2[training_idx,]
771  test <- RFM2[test_idx,]
772
773  training <- training[2:6]
774  test <- test[2:6]
```

```
> str(training)
'data.frame':   11239 obs. of  5 variables:
 $ AGE_PRD  : num  2 1 5 4 4 4 2 4 5 2 ...
 $ Recency  : num  2.89 1.1 2.56 3.61 3.5 ...
 $ Frequency: num  3.61 3.04 2.08 2.64 1.1 ...
 $ Monetary : num  13.9 14.5 10.7 14.4 10.3 ...
 $ future   : num  14.5 13.66 13.08 14.75 9.91 ...
> str(test)
'data.frame':   4817 obs. of  5 variables:
 $ AGE_PRD  : num  5 2 3 3 5 3 5 3 4 4 ...
 $ Recency  : num  0 0 3.76 3.64 4.55 ...
 $ Frequency: num  4.025 3.332 2.773 3.258 0.693 ...
 $ Monetary : num  13.8 13.3 13.7 13.1 11 ...
 $ future   : num  14.7 14.5 15.1 13.8 11.8 ...
```

7:3으로 셋을 분리 (11239 : 4817 로 잘 분리 됨)

훌륭한 모델링을 하기 위해 변수를 잘 선정해야 하지만 그냥 다 넣고 이것저것 돌려는 것에 의의를 둠

007
RFM + Regression을 활용한 미래 구매액 prediction

Linear model

```
776 data_lm_full <- lm(future ~., data=training)
777 summary(data_lm_full)
778
779 predict(data_lm_full, newdata=test[1:5,])
780
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.214029   0.109523   65.87   <2e-16 ***
AGE_PRD      0.096213   0.009046   10.64   <2e-16 ***
Recency     -0.104469   0.010379  -10.07   <2e-16 ***
Frequency    0.357580   0.016302   21.93   <2e-16 ***
Monetary     0.418063   0.009889   42.28   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.137 on 11234 degrees of freedom
Multiple R-squared:  0.5241,    Adjusted R-squared:  0.5239
F-statistic:  3093 on 4 and 11234 DF,  p-value: < 2.2e-16
```

step변수를 사용한 Linear model

```
783 library(MASS)
784
785 data_step <- stepAIC(data_lm_full, scope=list(upper = ~.^2, lower=~1))
786 data_step
787 summary(data_step)
```

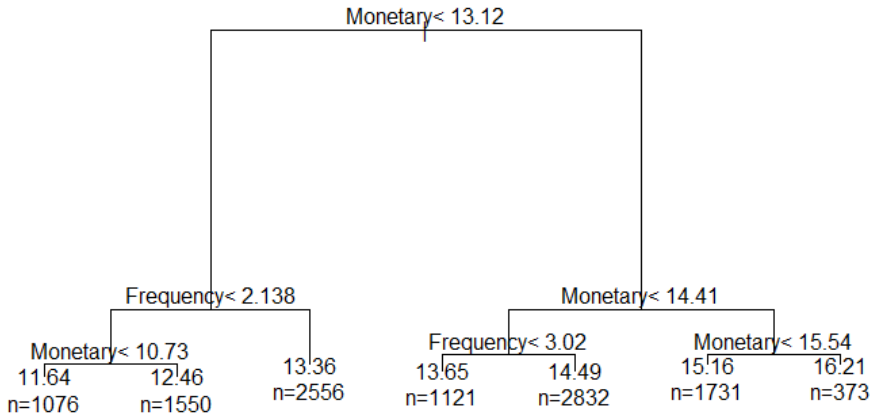
```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        10.673760   0.445446  23.962  < 2e-16 ***
AGE_PRD            -0.296558   0.080894  -3.666 0.000247 ***
Recency             0.021336   0.085462   0.250 0.802858
Frequency          -0.827923   0.091313  -9.067  < 2e-16 ***
Monetary            0.125589   0.039540   3.176 0.001496 **
Frequency:Monetary  0.096796   0.006334  15.283  < 2e-16 ***
Recency:Frequency   0.035334   0.009003   3.925 8.74e-05 ***
AGE_PRD:Monetary    0.039028   0.007792   5.009 5.56e-07 ***
AGE_PRD:Frequency  -0.043763   0.010912  -4.010 6.10e-05 ***
Recency:Monetary   -0.016402   0.007562  -2.169 0.030103 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.112 on 11229 degrees of freedom
Multiple R-squared:  0.5451,    Adjusted R-squared:  0.5448
F-statistic:  1495 on 9 and 11229 DF,  p-value: < 2.2e-16
```

변수간 2차 상호작용을 고려한 선형모델의 Adjusted R-squared가 약간 더 높음

007
RFM + Regression을 활용한 미래 구매액 prediction

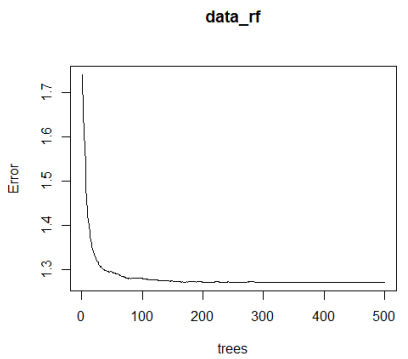
Tree model

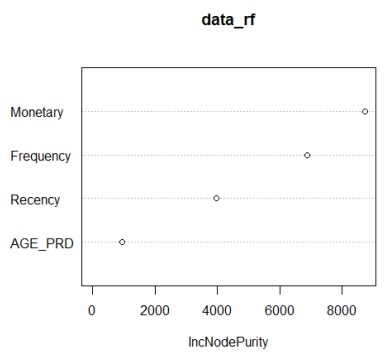
```
806  # 나무모형을 적용해보자
807  library(rpart)
808  data_tr <- rpart(future ~., data= training)
809  data_tr
810  printcp(data_tr)
811  summary(data_tr)
812
813  opar <- par(mfrow = c(1,1), xpd= NA)
814  plot(data_tr)
815  text(data_tr, use.n= TRUE)
816  par(opar)
```



Random Forest

```
830  install.packages("randomForest")
831  library(randomForest)
832
833  set.seed(184)
834  data_rf <- randomForest(future ~., training)
835  data_rf
836
837    #나무의 갯수에 따른 mse의 감소를 알려준다
838    #적당히 30개쯤 해도될듯???
839  plot(data_rf)
840
841    # 변수의 중요도를 알 수 있다
842    # 확실히 현재까지 구매액과 얼마나 자주구매해왔는지가 가장 큰 영향을 끼친다
843
844  varImpPlot(data_rf)
```





확실히 Random Forest를 실행하는데 시간이 오래 걸림

007
RFM + Regression을 활용한 미래 구매액 prediction

어떤 모델의 예측력이 가장 높을까?

```
852 # 선형모형, 스텝변수 선택한 선형 모형, 트리모델, 랜덤포레스트의 성능을 비교해보자
853 library(caret)
854 y_obs <- test$future
855 yhat_lm <- predict(data_lm_full, newdata=test)
856 yhat_step <- predict(data_step, newdata=test)
857 yhat_tr <- predict(data_tr, test)
858 yhat_rf <- predict(data_rf, newdata = test)
859
860 data.frame(lm = RMSE(y_obs, yhat_lm),
861            step = RMSE(y_obs, yhat_step),
862              tree = RMSE(y_obs, yhat_tr),
863                rf = RMSE(y_obs, yhat_rf))
864
865   # 스텝변수를 선택한 선형모델이 가장 예측력이 좋은 것을 알 수 있다 (최종모형사용)
866
```

	lm	step	tree	rf
1	1.189546	1.163688	1.224843	1.17735

	AGE_PRD	Recency	Frequency	Monetary	future	prediction
3822	4	172	2	4600	710248	77483.48
3823	2	63	2	18300	40600	111996.18
3829	2	21	61	4939760	1233770	4531755.73
3831	4	2	44	653710	3220331	1663176.69
3832	5	42	19	323418	486450	718255.04
3833	1	110	7	26200	25520	148907.41
3834	3	12	22	731700	1194412	1099273.43
3838	5	46	1	58000	50000	180177.90
3841	3	16	23	332640	3630576	736091.71
3847	2	137	2	5250	233400	80778.01
3848	2	11	16	725550	268100	862482.38

Step변수를 활용한 선형모델의 RMSE가 1.16으로 가장 낮음 => 최종 모델로 선정!

오른쪽에 exp()로 로그를 다 풀어서 실제 결과를 보니 꽤 많은 차이가 남..

008
출처

https://github.com/joaolcorreia/RFM-analysis/blob/master/RFM%20Analysis.ipynb

http://servicedesignplatform.com/archives/45

https://m.blog.naver.com/bestinall/221321162598

http://pyopyo03.tistory.com/14 [보노보노의 분석라이프]

http://hackability.kr/entry/Data-Mining-11-연관-법칙-Association-Rule-소개 [HACKABILITY]

연관분석(Association Rule)|작성자 인우기술

http://hackersstudy.tistory.com/126 [공대인들이 직접 쓰는 컴퓨터공부방]

위키트리, 위키백과

[중앙일보] 충동구매, 20대가 아니라 유아 둔 30대가 최고

008
출처

https://towardsdatascience.com/apply-rfm-principles-to-cluster-customers-with-k-means-fef9bcc9ab16

실리콘밸리 데이터과학자가 알려주는 따라하며 배우는 데이터과학 / 권재명

Data Mining Using RFM Analysis / Derya Birant / Dokuz Eylul University / Turkey

009
질의응답


THANK
YOU