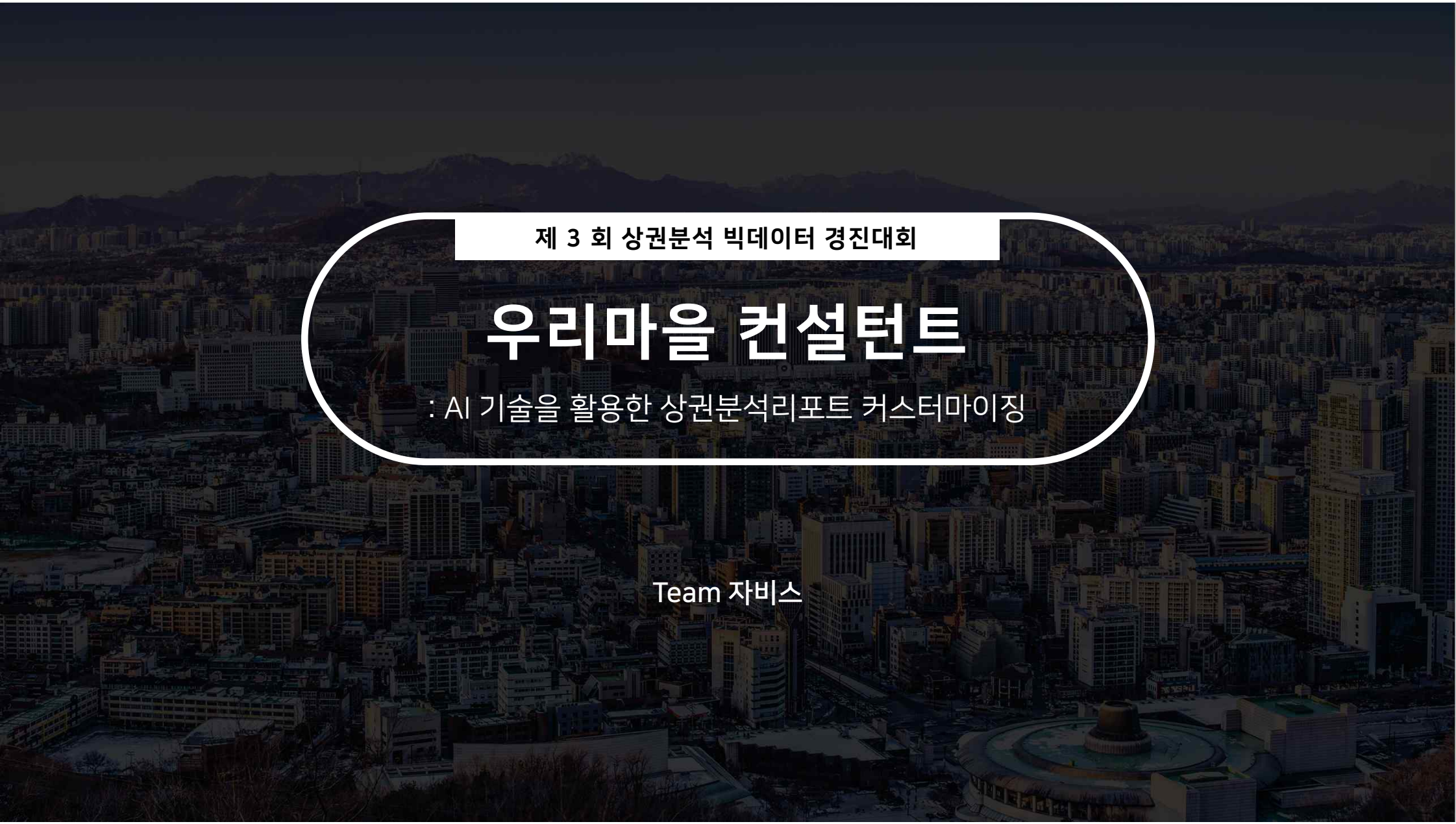


제 3 회 상권분석 빅데이터 경진대회

우리마을 컨설턴트

: AI 기술을 활용한 상권분석리포트 커스터마이징

Team 자비스

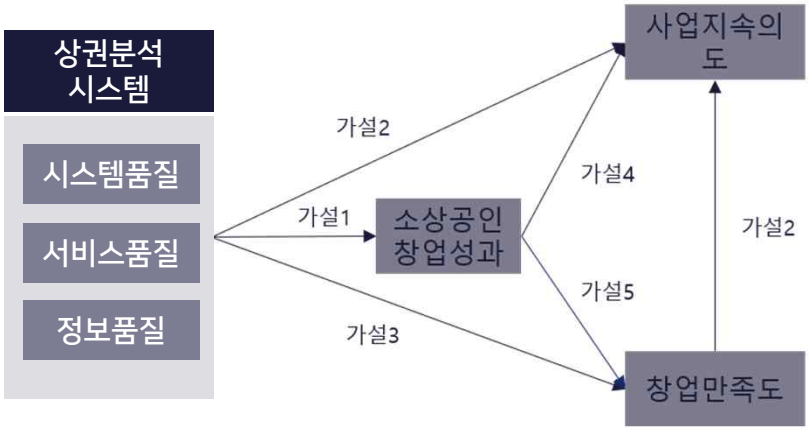
Contents

01 연구 배경 – a.상황분석

지금은 소상공인의 질적 성장이 필요한 시점!

서비스 품질의 고도화로 효율적인 목표 달성이 가능하다

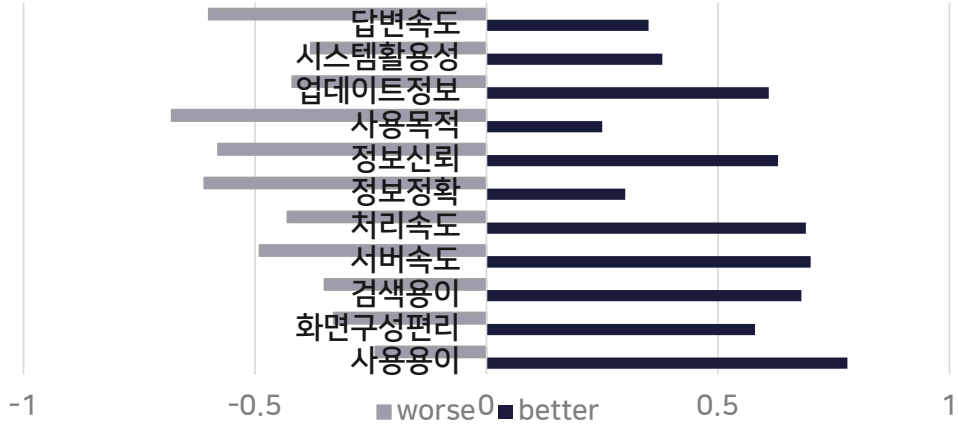
정보시스템 제공 모형



출처 : 소상공인 경쟁력 강화의 지원제도에 관한 연구

창업성과 ► 창업 만족도 ►사업 지속의도 ►질적성장
순으로 영향을 미치며, 전체 품질 중
서비스 품질의 정의(+)가설이 가장 유의함

상권분석 서비스 품질 Better/Worse 지수



출처 : 소상공인 경쟁력 강화의 지원제도에 관한 연구

Worse 지수가 가장 높은 사용목적은 일원적 특성을 가져
충족되지 않을 때 불만족이 크게 증가하지만
충족 시 전체적 만족도를
크게 높일 수 있는 잠재적 항목

01 연구 배경 – b.달성 목적

뿐만 아니라, 소상공인의 분석 서비스 활용도 제고를 위해

개별 상황에 쉽게 적용할 수 있도록 서비스의 보완이 필요하다

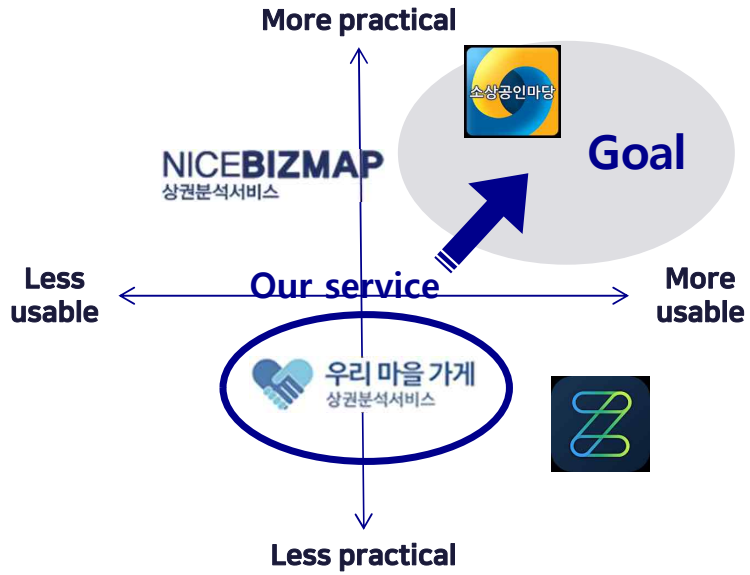
상권분석 서비스에 대한 이해도 부족

구분	빈도	비율(%)
제공서비스 정보의 부정확성	110	41.7
장애발생감소	40	15.2
이용방법에 대한 안내 확대	**56**	**21.2**
정보해석을 위한 정보 재가공	**53**	**20.1**
보안성 확보 (로그인 기능 추가 시)	5	1.9
합계	**264**	**100.0**

출처 : 상권정보시스템 정책성과 분석 및 향후 정책방안

이용 방법 및 제공 데이터 해석 측면에서
구체적인 가이드의 미제공으로 기능을
제대로 활용하지 못하고 있는 상황

기존 서비스 대비 포지셔닝



뿐만 아니라, 소상공인의 분석 서비스 활용도 제고를 위해

개별 상황에 쉽게 적용할 수 있도록 서비스의 보완이 필요하다

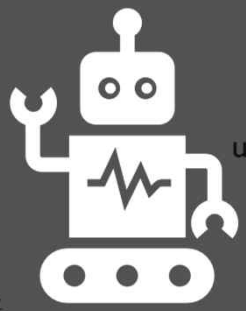
우리마을 컨설턴트

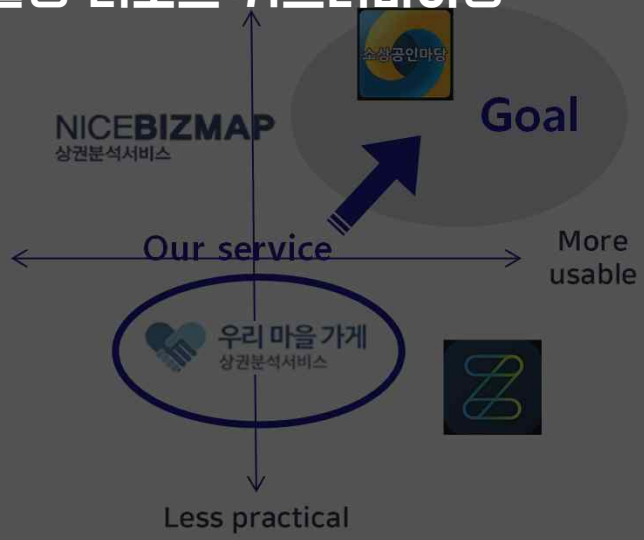
: 소상공인들의 마케팅 전략 수립을 위한 AI 활용 리포트 커스터마이징

상권분석 서비스에 대한 어려움 부정 기존 서비스 대비 포지셔닝

구분	빈도	비율(%)
제공서비스 정보의 부정확성	110	41.7
장애발생감소	40	15.2
이용방법에 대한 안내 확대	**56**	**21.2**
정보해석을 위한 정보 재가공	**53**	**20.1**
보안성 확보 (로그인 기능 추가 시)	5	1.9
합계	**264**	**100.0**

출처 : 상권정보시스템 정책성과 분석 및 향후 정책방안

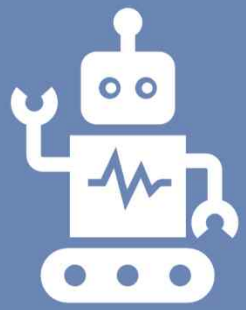




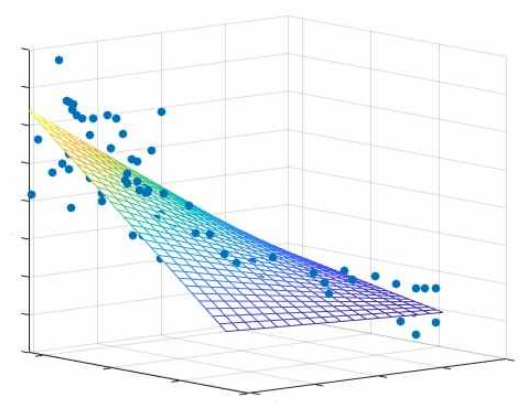
이용 방법 및 제공 데이터 해석 측면에서
구체적인 가이드의 미제공으로 기능을
제대로 활용하지 못하고 있는 상황

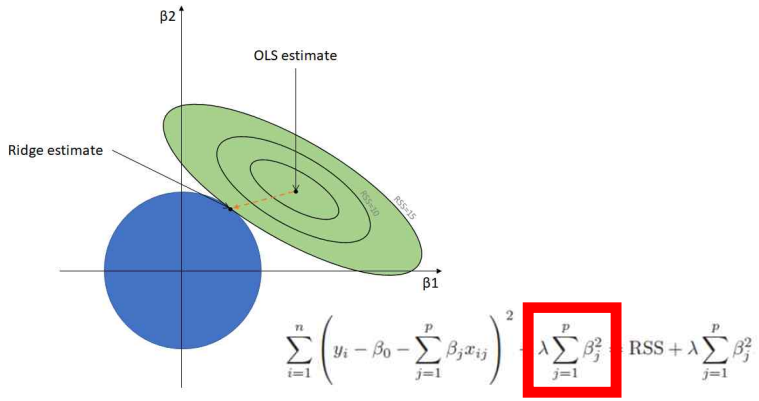
Ⅰ. 창업 접수 모델

: 소상공인들의 마케팅 전략 수립을 위한 AI 활용 리포트 커스터마이징



≫ 활용 기술 : Multi Regression & Ridge





- **<u>다중 선형 회귀</u>**: 여러 개의 독립변수로 연속형 종속변수를 예측하기 위한 회귀모형으로, $Y = a + \beta_1 x_1 + \beta_2 x_2 + \cdots$ 와 같은 회귀식에서 각 **공변량 $(x_1, x_2, \ldots)$이 한 단위 변할 때 종속변수는 편회귀 계수 $(\beta_1, \beta_2, \ldots)$ 만큼 변한다**고 해석함.

- **<u>Ridge 회귀</u>**: 규제가 추가된 선형 회귀 버전으로, 비용함수에 규제항(λ)을 추가하여 중요하지 않은 특성에 대한 가중치를 가능한 작게 만듦. 이러한 파라미터 축소 작업은 **다중 공선성 문제를 해결**하는데 가장 유용하게 쓰이며 **모델의 복잡도를 줄이는 데에도 효과적임**. L-2 Norm을 사용하며, 변수가 많고 계수의 크기가 거의 동일한 크기일 때 성능이 좋음.

기 창업자용 서비스 (점수)

≫ 서비스 기능 및 목적

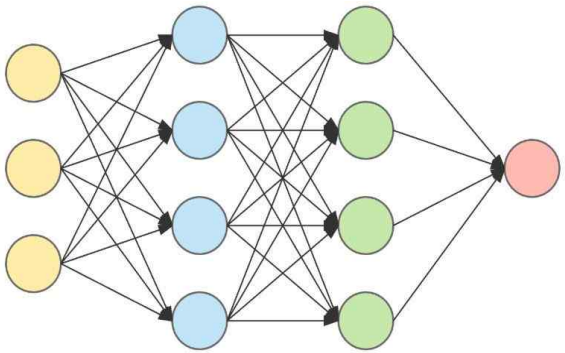
▪ **기능**:
사용자는 본인 업소의 최대 실적(매출 금액, 매출 건수)을 달성한 **연령, 요일, 시간대**를 입력하여 **주변 지역, 동일 업종 업소들의 현황과 자신의 업소를 비교**할 수 있음

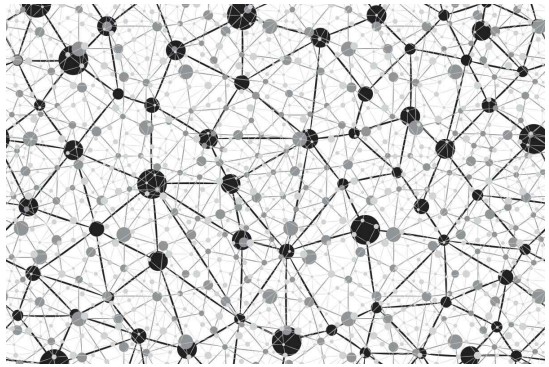
▪ **목적**:
맞춤형 서비스, # 구체적 인사이트 제공, # user-friendly, # 품질 고도화, # 실시간 데이터, # 사용목적적합성

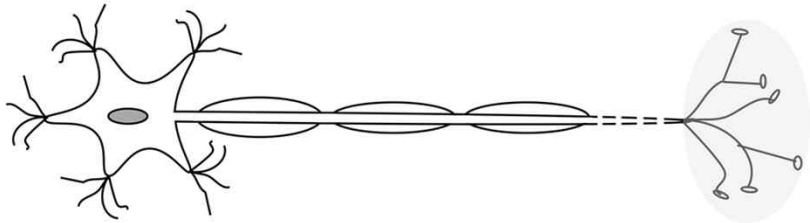
⇨ 소상공인 개별 상황에 **맞춤형 서비스** 제공

⇨ 소상공인의 **질적 성장 및 품질 경쟁력 향상**

≫ 활용 기술 : Neural Network







- **신경망**: 생물학의 신경망에서 영감을 얻은 통계학적 학습 알고리즘으로, 인간의 뇌가 패턴을 인식하는 방법을 모사한 방법. Feedforward와 Backpropagation을 반복하는 **지도학습 방식**, 각 단계에서 **활성화함수와 비용 최소화함수**를 적용하여 각 입력층에 대한 **최선의 가중치를 출력함**

기 창업자용 서비스 (등급)

≫ 서비스 기능 및 목적

▪ **기능**:
사용자는 본인 업소의 최대 실적(매출 금액, 매출 건수)을 달성한 연령, 요일, 시간대를 입력하여 주변 지역, 동일 업종 업소들의 현황과 자신의 업소를 비교할 수 있음

▪ **목적**:
맞춤형 서비스, # 구체적 인사이트 제공, # user-friendly, # 품질 고도화, # 실시간 데이터, # 사용목적적합성

⇨ 소상공인 개별 상황에 맞춤형 서비스 제공

⇨ 소상공인의 질적 성장 및 품질 경쟁력 향상

≫ 원본 데이터:
SELNG 테이블 中 아래 항목들만 가져와 변수 “Selling”에 할당

▪ 업소의 기본 정보 (1~6)과 연령, 요일, 시간대별 매출금액 및 매출건수의 합(19~56) 항목으로 데이터 셋 구성

1	블록코드	BLCK_CD
2	매출년월	TA_YM
3	표준업종코드	KSIC_CD
4	가맹점수	MCT_CNT
5	매출액	AMT
6	매출건수	CNT
7	최소 매출액	MIN_AMT

19	10 대 매출금액의 합	AMT_10
20	10 대 매출건수의 합	CNT_10
21	20 대 매출금액의 합	AMT_20
33	11시까지 매출금액의 합	AMT_T11
34	11시까지 매출건수의 합	CNT_T11
53	금요일매출금액합	AMT_FRI
54	금요일매출건수합	CNT_FRI
55	토요일매출금액합	AMT_SAT
56	토요일매출건수합	CNT_SAT

≫ 데이터 변환: 활용 목적에 맞도록 실적 항목의 의미 변환

▪ 하나의 행이 업소 단위가 아닌 블록 단위이므로 블록에 속해 있는 가맹점 수로 실적 항목을 나눠 줌

<CODE>

```
# 원본 데이터
Selling_origin <- read.csv(file = "SELNG.csv", sep = "|")
View(Selling)

# 필요한 열 전체 추출
Selling <- Selling_origin %>% dplyr::select(c(1:6, 19:56))
```

```
# 매출액/매출건수를 가맹점 수로 나누어 하나의 행을 하나의 업소로 취급
for (i in c(7:44)){
  Selling[,i] <- Selling[,i]/Selling$MCT_CNT
}
```

≫ 데이터 생성:
업소별 최대 매출액/매출건수를 달성한 연령, 요일, 시간대 추출

▪ 연령, 요일, 시간대별 하위 테이블(Sales_amt_, Sales_cnt_) 생성 후 최대 실적 추출 후 데이터프레임으로 변환

▪ 최대 실적에 매핑되는 연령, 시간, 요일의 인덱스 값을 찾아 리스트(which_max_)로 저장

≫ 데이터 통합 :
앞서 생성한 테이블들을 하나의 테이블로 통합

▪ 최대 실적 값 테이블과 최대 실적 인덱스 테이블을 하나의 테이블로 통합 후 열 이름 지정

▪ 통합한 테이블에서 사용할 열(블록코드, 업종코드, 연월, 연령/요일/시간대별 최대 실적, 전체 실적)만 추출

<CODE>

```
# 매출액이 MAX가 되는 나이, 요일, 시간 테이블 추출
## MAX 작업을 위해 원하는 열 추출하여 하위 테이블 생성
Sales_amt_age <- Selling %>% dplyr::select(AMT_10,  AMT_20,  AMT_30,  AMT_40,  AMT_50,  AMT_60)
Sales_amt_days <- Selling %>% dplyr::select(AMT_SUN, AMT_MON, AMT_TUE, AMT_WED, AMT_THU, AMT_FRI, AMT_SAT)
Sales_amt_times <- Selling %>% dplyr::select(AMT_T06, AMT_T11, AMT_T14, AMT_T17, AMT_T21, AMT_T24)
Sales_cnt_age <- Selling %>% dplyr::select(CNT_10, CNT_20, CNT_30, CNT_40, CNT_50, CNT_60)
Sales_cnt_days <- Selling %>%  dplyr::select(CNT_SUN, CNT_MON, CNT_TUE, CNT_WED, CNT_THU, CNT_FRI, CNT_SAT)
Sales_cnt_times <- Selling %>%  dplyr::select(CNT_T06, CNT_T11, CNT_T14, CNT_T17, CNT_T21, CNT_T24)
```

```
# 나이, 요일, 시간 별로 MAX(index값) 테이블 생성
## Age
which_max_age_amt <- as.list(0)

for(i in 1:nrow(Sales_amt_age)){
  row_list <- c(Sales_amt_age[i, 1], Sales_amt_age[i, 2], Sales_amt_age[i, 3], Sales_amt_age[i, 4],
                Sales_amt_age[i, 5], Sales_amt_age[i, 6])
  which_max_age_amt[[i]] <- which.max(row_list)
}
```

```
## 블록코드, 업종코드, 총매출액이랑 cbind
age_amt_max <- cbind(which_max_age_amt_df_t, Max_amt_age_df)
days_amt_max <- cbind(which_max_days_amt_df_t, Max_amt_days_df)
times_amt_max <- cbind(which_max_times_amt_df_t, Max_amt_times_df)
View(age_amt_max)

### 열 이름 변경
names(age_amt_max)[1] <- "Age_index"
names(days_amt_max)[1] <- "Days_index"
names(times_amt_max)[1] <- "Times_index"
names(age_amt_max)[2] <- "Age_amt"
names(days_amt_max)[2] <- "Days_amt"
names(times_amt_max)[2] <- "Times_amt"
```

```
# 사용할 열만 추출 후 열 이름 rename
AMT_MAX <- cbind(Selling$BLCK_CD, Selling$KSIC_CD, Selling$TA_YM, age_amt_max, days_amt_max, times_amt_max, Selling$AMT)

rename <- dplyr::rename
AMT_MAX <- rename(AMT_MAX, "BLCK_CD"="Selling$BLCK_CD", "KSIC_CD"="Selling$KSIC_CD",
                  "TA_YM"="Selling$TA_YM", "AMT"="Selling$AMT")
```

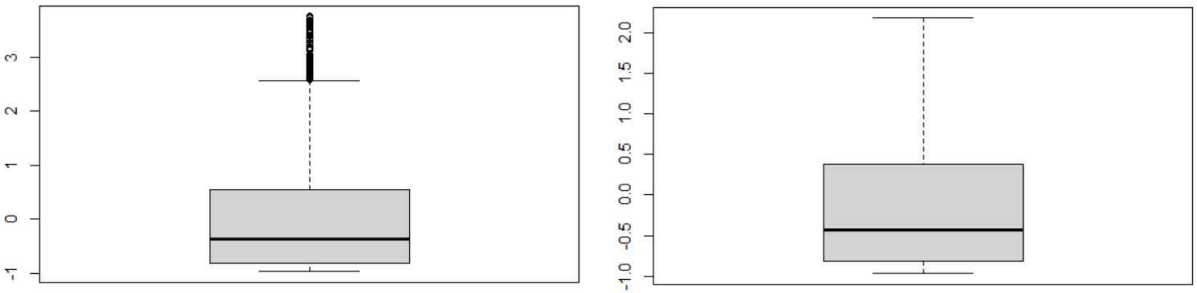
02 창업점수 모델 – b.데이터 전처리

Team Javis

≫ DATA JOIN

	KSIC_CD	TA_YM	Age_index	Age_amt	Days_index	Days_amt	Times_index	Times_amt	AMT	OLD_ADRES
2	357860	45	201907	3	104000	7	94000	4	116000	192000 강남구
3	359790	45	201907	3	260000	4	196250	4	194250	454250 강남구
4	365617	45	201907	6	2320000	4	2210000	4	2708000	4274000 강남구
5	366478	45	201907	4	447500	5	325000	5	325000	447500 강남구
6	369553	45	201908	3	518000	6	331000	4	547000	772000 강남구
7	371492	45	201908	4	453250	4	383250	4	474250	1004250 강남구
8	378236	45	201908	6	5.00E+06	2	5.00E+06	4	5.00E+06	5.00E+06 강남구
9	381330	45	201909	3	204500	7	264500	4	131000	341500 강남구
10	383294	45	201909	4	180000	6	215000	4	125000	322500 강남구
11	390833	45	201910	3	1000250	7	310150	4	573350	1337650 강남구
12	391279	45	201910	3	2390000	3	1550000	5	1210000	2730000 강남구

≫ 지역, 업종별 데이터 분할 및 정제



이상치 제거 전/후 Box plot

<CODE>

```
## 상가업소의 블록코드와 지번주소 불러오기
CNFM <- read.csv(file = "C://Users//bigdata15//Desktop//복사본 CNFM_PRMISN.csv")
region <- CNFM %>% dplyr::select(BLCK_CD, OLD_ADRES)
region$OLD_ADRES <- substr(region$OLD_ADRES, 7, 9)
region_split <- split(region, region$OLD_ADRES)   # 5개 지역구와 1개의 null 집합 확인
```

```
### 강남구와 강북구만 남겨두고 제거
region <- region[!(region$OLD_ADRES == ""),]
region <- region[!(region$OLD_ADRES == "노원구"),]
region <- region[!(region$OLD_ADRES == "성북구"),]
region <- region[!(region$OLD_ADRES == "송파구"),]
region_split <- split(region, region$OLD_ADRES)

## 블록코드 기준 innerjoin
query_innerjoin <- "

SELECT *
FROM 'AMT_MAX'
INNER JOIN 'region' ON 'region'.BLCK_CD = 'AMT_MAX'.BLCK_CD
"
```

```
# 1차 split - 지역별
Gangnam_amt <- AMT_FIN[AMT_FIN$OLD_ADRES == "강남구", ]
Gangbuk_amt <- AMT_FIN[AMT_FIN$OLD_ADRES == "강북구", ]

# 업종 기준 2차 split
amt_sectors_gn <- split(Gangnam_amt, Gangnam_amt$KSIC_CD)
amt_sectors_gb <- split(Gangbuk_amt, Gangbuk_amt$KSIC_CD)
```

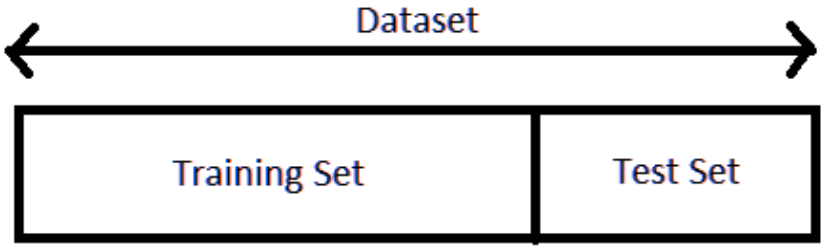
```
# 이상치 처리============
# 이상치 확인
boxplot(`55`$AMT)
boxplot(`55`$AMT)$stats
`55`$AMT <- ifelse(`55`$AMT < 100 | `55`$AMT > 9978125, NA, `55`$AMT)

# 결측치 제거
`55` <- na.omit(`55`)
sum(is.na(`55`$AMT))
nrow(`55`)
# ==============
```

창업점수 모델 – b.데이터 전처리

≫ 데이터 정제 후 더미화:
범주형 설명변수(index)에 대해 1, 0의 값으로 인코딩 후 최종 데이터 정제

- 기준 변수를 잡기 위해 히스토그램으로 각 인덱스 그룹별 최대 빈도의 인덱스 선택
- 기준 변수를 제외한 인덱스에 대해 1,0의 값을 활용하여 인코딩 후 factor 처리
- 낮은 분산을 갖는 변수를 식별하여 제거

≫ train/test set split



<CODE>

```r
# `55_BC`로 채택 후 표준화
`55` <- transform(`55`, z.BC = scale(BC))
head(`55`)

hist(`55`$z.BC, breaks = 100)
# ======
```

```r
# train, test set split =======
set.seed(100) # 난수 생성

index = sample(1:nrow(`55`), 0.7 * nrow(`55`))
nrow(`55`)

train = `55`[index,] # Create the training data # 70%
test = `55`[-index,] # Create the test data # 30%
```

```r
# 각 요소 개수 확인 ====
## 빈도수 높은 변수로 더미변수의 기준 채택
hist(`55`$Age_index) # 2 채택
hist(`55`$Days_index) # 7 채택
hist(`55`$Times_index) # 1 채택
```

```r
# encoding=======
### train
train <- transform(train,
                   Age_index_1 = ifelse(Age_index=="1", 1, 0),
                   Age_index_3 = ifelse(Age_index=="3", 1, 0),
                   Age_index_4 = ifelse(Age_index=="4", 1, 0),
                   Age_index_5 = ifelse(Age_index=="5", 1, 0),
                   Age_index_6 = ifelse(Age_index=="6", 1, 0),
                   Days_index_1 = ifelse(Days_index=="1", 1, 0),
                   Days_index_2 = ifelse(Days_index=="2", 1, 0),
                   Days_index_3 = ifelse(Days_index=="3", 1, 0),
                   Days_index_4 = ifelse(Days_index=="4", 1, 0),
                   Days_index_5 = ifelse(Days_index=="5", 1, 0),
                   Days_index_6 = ifelse(Days_index=="6", 1, 0),
                   Times_index_2 = ifelse(Times_index=="2", 1, 0),
                   Times_index_3 = ifelse(Times_index=="3", 1, 0),
```

창업점수 모델 – b.데이터 전처리

≫ 최종 데이터 셋

- 예시 데이터 (amt_gn_55)
- 매출지표별(매출액, 매출건수), 지역별, 업종별 각각의 데이터셋 완성

	Age_10	Age_30	Age_40	Age_50	Age_60	Days_SUN	...
1	0	0	0	0	0	1	
2	0	0	1	0	0	0	
3	0	0	0	0	0	0	
4	0	1	0	0	0	0	
5	0	0	0	1	0	0	

입력 변수 – 최대 지출 연령, 요일, 시간대에 대한 더미변수

AMT	AMT_bc	AMT_grade
1625000	103.49463	C
251000	66.80829	D
5050000	134.44177	A
10100000	157.57961	A
1000000	92.44956	C

목적 변수

≫ 다중 선형 회귀 분석: 모델 적용

- Data set: 강남 숙박업종 대상
- Training 입력 변수: 최대 매출 나이, 요일, 시간대에 대한 더미 변수
- Training 목표 변수: 매출 금액에 대하여 정규화된 숫자 변수

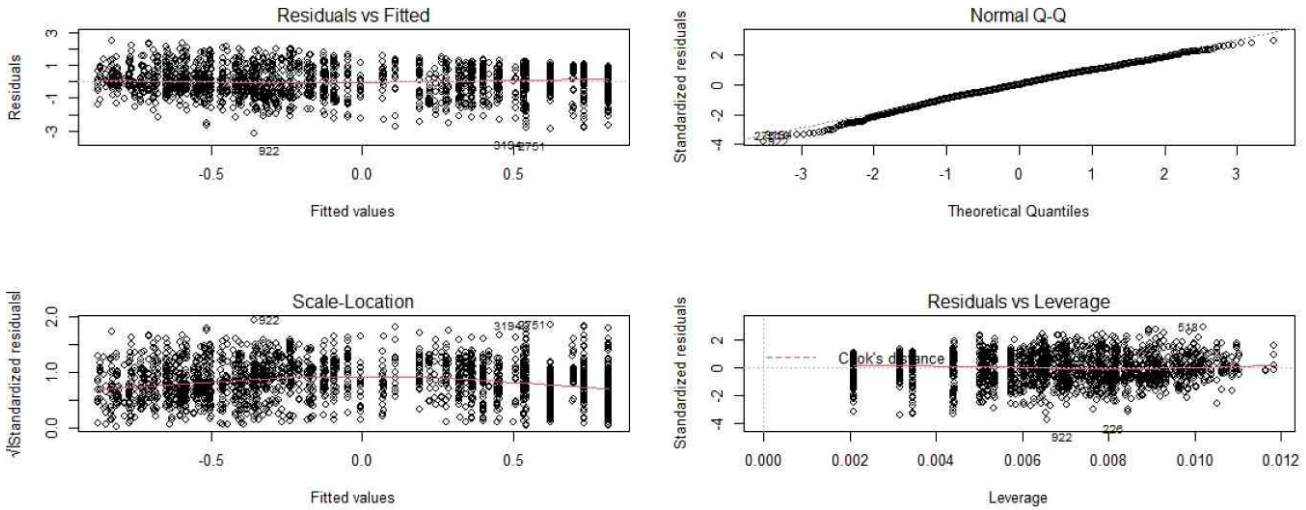
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.343e+02  1.507e+00  89.125  < 2e-16 ***
X           -4.642e-04  5.615e-04  -0.827   0.4085
Age_10      -1.608e+01  1.054e+01  -1.525   0.1273
Age_30      -2.060e+00  1.370e+00  -1.504   0.1327
Age_40      -2.512e+00  1.601e+00  -1.569   0.1168
Age_50      -3.247e+00  1.894e+00  -1.714   0.0866
Age_60      -1.585e+01  3.393e+00  -4.670 3.18e-06 ***
Days_SUN    -7.770e+00  1.591e+00  -4.885 1.10e-06 ***
Days_MON    -1.062e+01  2.049e+00  -5.183 2.38e-07 ***
Days_TUE    -1.265e+01  1.984e+00  -6.376 2.19e-10 ***
Days_WED    -1.081e+01  1.943e+00  -5.561 3.00e-08 ***
Days_THU    -1.127e+01  1.962e+00  -5.745 1.04e-08 ***
Days_FRI    -9.293e+00  1.939e+00  -4.794 1.74e-06 ***
Times_11    -3.340e+01  2.811e+00 -11.879  < 2e-16 ***
Times_14    -3.409e+01  2.108e+00 -16.172  < 2e-16 ***
Times_17    -3.427e+01  1.948e+00 -17.592  < 2e-16 ***
Times_21    -2.647e+01  1.523e+00 -17.386  < 2e-16 ***
Times_24    -2.124e+01  1.808e+00 -11.747  < 2e-16 ***
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1

Residual standard error: 25.62 on 2324 degrees of freedom
Multiple R-squared:  0.3244,    Adjusted R-squared:  0.31
F-statistic: 65.64 on 17 and 2324 DF,  p-value: < 2.2e-16
```

≫ 분석 결과

- 모델 적합 후 생성된 각 변수에 대한 회귀 계수를 고려하여 선형회귀식 도출 가능
- 회귀식 예시: $Y = (-1.58e + 01) * X_1 + \cdots + (-2.124e + 01) * X_{16}$

≫ 다중 선형 회귀 분석: 모델 검증

- 학습이 완료된 다중선형회귀 모델이 적합한 지 검증하는 과정
- plot을 통해 분포의 독립성과 이상치의 유무를 살피는 등의 분산분석 결과, 모델의 이상은 없음



≫ Ridge 회귀 분석: 모델 적용

- 다중 선형 회귀 모델과 같은 데이터셋을 대상으로 lambda 값을 직접 튜닝하여 모델 적용
- RMSE 점수가 약 0.82 이므로 입력 변수의 설정은 적절했다고 판단

Ridge regression 모델링 적용 CODE

```
## 람다값 설정
lambdas <- 10 ^ seq(2, -3, by = -.1)

# 모델링 - parameter 수, alpha값 (ridge는 0), 분포 모형, 람다값 설정
ridge_reg = glmnet(x_train, y_train, nlambda = 5, alpha = 0, family = 'gaussian', lambda = lambdas)
summary(ridge_reg)
```

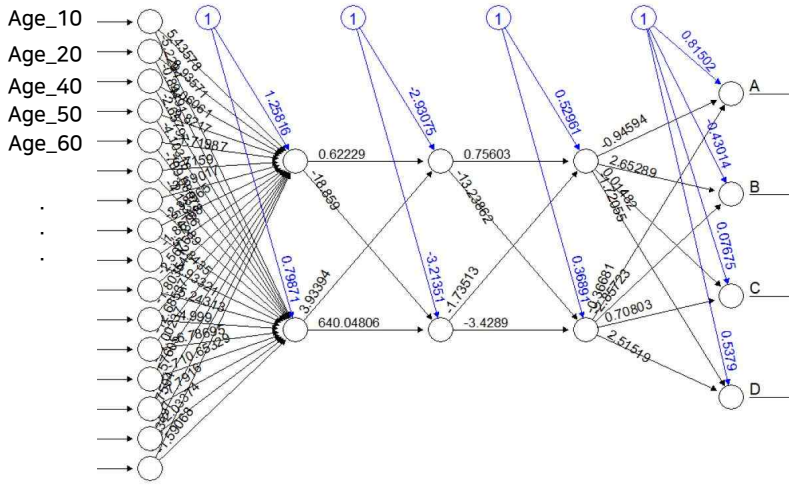
모델 매개변수 튜닝 과정 CODE

```
# hyperparameter lamda value tuning (자동) ====
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0, lambda = lambdas)
optimal_lambda <- cv_ridge$lambda.min
optimal_lambda # 최적값 = 0.2511886 # 표준화+정규화 : 0.01258925
```

≫ 신경망 분석: 목표 변수를 등급으로 범주화하여 분류기 생성

- 2개씩 3개의 은닉층을 갖는 신경망을 생성하여 매출지표의 범주 형태인 A, B, C, D 등급으로 분류하도록 학습
- neuralnet 패키지를 사용하여 error 값은 sum of squared error 사용, 역치 0.04, 최대 스텝수를 1e6으로 설정

<신경망 모델 생성>



<모델 학습 결과>

Prediction	Reference A	B	C	D
A	162	92	42	15
B	49	67	69	34
C	0	3	4	5
D	36	64	173	188

정확도

약 42.0%

≫ 분석 결과

- 완성된 모델에 테스트용 데이터셋의 입력값들을 대입하여 도출된 예측값을 실제 테스트용 데이터셋의 목표 변수 값과 비교
- 42.0%의 정확도로 입력 변수에 대한 매출 등급을 도출
- 하나의 지역과 업종 대상이므로 데이터의 수가 적고, 훈련 데이터 자체의 분류 형태가 명확하지 않아 정확도에 영향을 미침

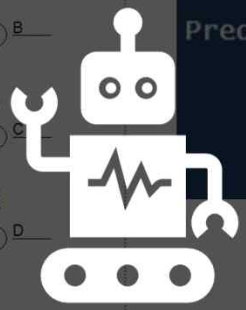
≫ 신경망 분석: 목표 변수를 등급으로 범주화하여 분류기 생성

- 2개씩 3개의 은닉층을 갖는 신경망을 생성하여 매출지표의 범주 형태인 A, B, C, D 등급으로 분류하도록 학습
- neuralnet 패키지를 사용하여 error 값은 sum of squared error 사용, 역치 0.04, 최대 스텝수를 1e6으로 설정

Ⅱ. 경쟁도 군집화 모델

: 소상공인들의 마케팅 전략 수립을 위한 AI 활용 리포트 커스터마이징

<신경망 모델 생성>



<모델 학습 결과>

Prediction \ Reference	A	B	C	D
A	162	92	42	15
B	49	67	69	34
C	0	3	4	5
D	36	64	173	188

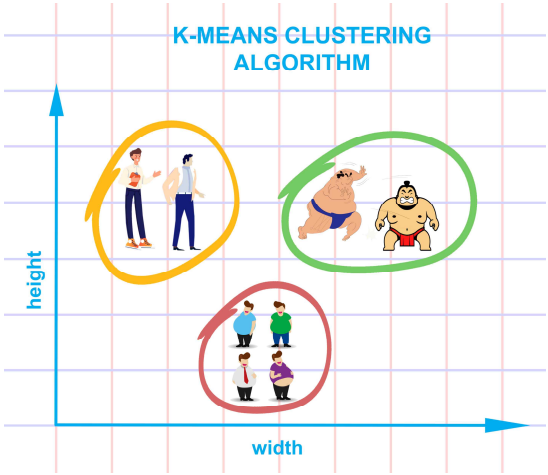
정확도

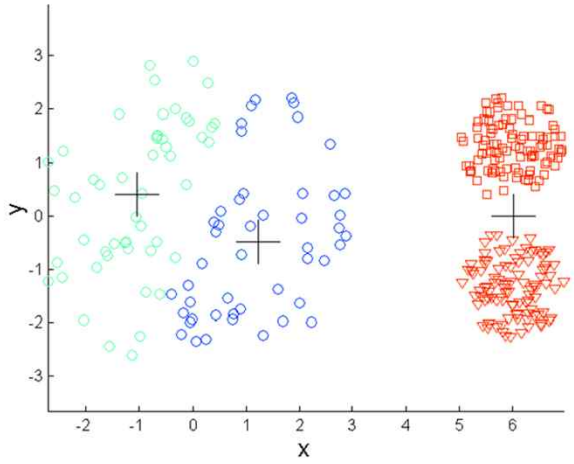
약 42.0%

≫ 분석 결과

- 완성된 모델에 테스트용 데이터셋의 입력값들을 대입하여 도출된 예측값을 실제 테스트용 데이터셋의 목표 변수 값과 비교
- 42.0%의 정확도로 입력 변수에 대한 매출 등급을 도출
- 하나의 지역과 업종 대상이므로 데이터의 수가 적고, 훈련 데이터 자체의 분류 형태가 명확하지 않아 정확도에 영향을 미침

≫ 활용 기술 : K-means Clustering





- **K-means**: 분리형 군집화 알고리즘 중 하나로, 라벨링 되지 않은 데이터 안에서 패턴과 구조를 발견하는 **비지도 학습 방법**. K개의 클러스터 개수를 정한 후 중심과의 **평균 거리를 최소화**하는 방식으로 유사한 군집을 분류함. 거리 계산에는 **유클리디안, 맨하탄 거리** 등이 대표적으로 사용됨.

기 창업자용 & 창업예정자용 서비스

≫ 서비스 기능 및 목적

▪ **기능**:
사용자는 본인 업소가 속한 행정동, 업종을 입력하여 **해당 군집의 경쟁도 및 이에 따른 전략을 파악**할 수 있음 (경쟁도는 해당 군집의 가맹점 수를 활용하며, 평균 영업 개월 수와 함께 군집을 특징 지음)

▪ **목적**:
직접적 활용, # 맞춤형 서비스, # 사용목적적합성, # 인사이트, # 상권 이해, # 창업 성과

⇨ **구체적인 분석 결과**를 제공하는 **인사이트형 리포트** 구성

⇨ 소상공인의 **질적 성장 및 품질 경쟁력 향상**

≫ 원본 데이터:
SELNG 테이블 中 아래의 열들만 선택해 변수 “Selling”에 할당

▪ 업소의 기본 정보 (1~6)과 재방문건수 및 평균 영업개월수(57~63)항목으로 데이터 셋 구성

1	블록코드	BLCK_CD
2	매출년월	TA_YM
3	표준업종코드	KSIC_CD
4	가맹점수	MCT_CNT
5	매출액	AMT
6	매출건수	CNT
7	최소 매출액	MIN_AMT

57	10 대 재방문 매출건수의 합	RCNT_10
58	20 대 재방문 매출건수의 합	RCNT_20
59	30 대 재방문 매출건수의 합	RCNT_30
60	40 대 재방문 매출건수의 합	RCNT_40
61	50 대 재방문 매출건수의 합	RCNT_50
62	60 대 재방문 매출건수의 합	RCNT_60
63	평균영업개월수	MCT_SALES

≫ 데이터 변환:
활용 목적에 맞도록 실적 항목의 의미 변환

▪ 연령별로 재방문건수에 대해 총 집계로 변환
▪ 이후 비율로 변환을 위해 총매출건수(CNT) 항목도 추출

<CODE>

```
# 필요한 열 추출
Selling <- Selling_origin %>% dplyr::select(c(1:6, 57:63))
View(Selling)
```

```
# 재방문 수 연령대 전체 sum
RCNT <- Selling %>% dplyr::select(RCNT_10, RCNT_20, RCNT_30, RCNT_40, RCNT_50, RCNT_60)
RCNT <- apply(RCNT, 1, sum)
RCNT <- as.data.frame(RCNT)

# 전체 매출 건수 추출
CNT <- Selling %>% dplyr::select(CNT)
```

02 경쟁도 군집화 모델 – b.데이터 전처리

≫ DATA JOIN :
상가업소테이블(CNFM)의 지역정보를 조인으로 가져옴

- 업소의 기본 정보 (1~6)과 재방문건수 및 평균영업개월수(57~63)항목으로 데이터 셋 구성
- 매출 점수 모델(1)과 같은 방법으로 조인 수행
- 모델에 학습시킬 데이터로 최신 데이터인 2019년 12월 데이터 사용

-> 해당 데이터에 대해 업종별 분할

≫ 최종 테이블 생성

- 45번 (예제) 업종에 대해 각 항목들을 행정동별 계산
- 가맹점수(MCT_CNT)는 단순 합, 재방문건수(RCNT), 매출건수(CNT)의 경우 합하여 재방문율(RCNT/CNT)로 변환, 평균영업개월수(MCT_SALES)는 전체 평균으로 계산

<CODE>

```
# 지역코드/업종코드-대분류와 매핑
## 상가업소 테이블의 블록코드와 지번주소 불러오기
CNFM <- read.csv(file = "C://Users//bigdata15//Desktop//복사본 CNFM_PRMISN.csv")
region <- CNFM %>% dplyr::select(BLCK_CD, OLD_ADRES)

# 행정동만 추출
region$OLD_ADRES <- substr(region$OLD_ADRES, 11, 13)

# 행정동별 데이터 split
region_split <- split(region, region$OLD_ADRES)   # 53개 지역구와 1개의 null 집합 확인

# 행정동명이 비어있는 데이터 삭제
region <- region[!(region$OLD_ADRES == ""),]
region_split <- split(region, region$OLD_ADRES)
```

```
# 최신(2019년 12월) 자료로 선택
`201912` <- AMT_FIN[AMT_FIN$TA_YM == "201912", ]

# 업종별 split
amt_sector <- split(`201912`, `201912`$KSIC_CD)
```

```
# 최종 테이블 생성 (예제: 45번 업종)
## 가맹점수, 평균영업개월수, 매출 건수 행정동별 계산
MCT_CNT <- aggregate(MCT_CNT~OLD_ADRES, dat_45, sum)
MCT_SALES <- aggregate(MCT_SALES~OLD_ADRES, dat_45, mean)
RCNT <- aggregate(RCNT~OLD_ADRES, dat_45, sum)
CNT <- aggregate(CNT~OLD_ADRES, dat_45, sum)
```

```
## cbind
dat_45 <- cbind(MCT_CNT, MCT_SALES, RCNT, CNT)

## 행정동 중복 column 제거
dat_45 <- dat_45[, c(-3, -5, -7)]

## 재방문건수를 재방문율로 변경
dat_45[4] <- dat_45[4] / dat_45[5] ## RCNT/CNT
dat_45[5] <- NULL # CNT 열 삭제
```

≫ 최종 테이블 생성

▪ 전체 18개의 업종에 대해 작업 후 업종별 행정동명 뒤에 라벨을 붙여 식별해줌 (ex. 청담동_45)
▪ 행정동 별 분포 확인 후 평균영업개월수 및 재방문율에 대해 표준화한 전체 데이터를 dat 테이블에 할당하여 마무리

최종 테이블 >

	OLD_ADRES	MCT_CNT	MCT_SALES	RCNT	V5
1	개포2	-0.70046969	-0.07206240	0.06250000	개포2 _45
2	개포4	-0.70046969	0.71425732	0.06250000	개포4 _45
3	개포동	0.88443142	-0.42741842	0.12280702	개포동 _45
4	논현2	-0.24764080	-0.67692371	0.15384615	논현2 _45
5	논현동	0.20518809	-0.54587043	0.14062500	논현동 _45
6	대치2	-0.47405524	-0.33668922	0.03846154	대치2 _45
7	대치4	-0.47405524	-0.31652718	0.00000000	대치4 _45
8	대치동	0.65801698	-0.57107298	0.02631579	대치동 _45
9	도곡동	-0.92688413	-0.54839068	0.02564103	도곡동 _45
10	미아동	1.11084587	0.68023387	0.12598425	미아동 _45
11	번1동	-0.24764080	-0.91508786	0.09259259	번1동 _45
12	번2동	-0.70046969	0.15476060	0.09090909	번2동 _45

군집화 기준

업종, 지역별 라벨

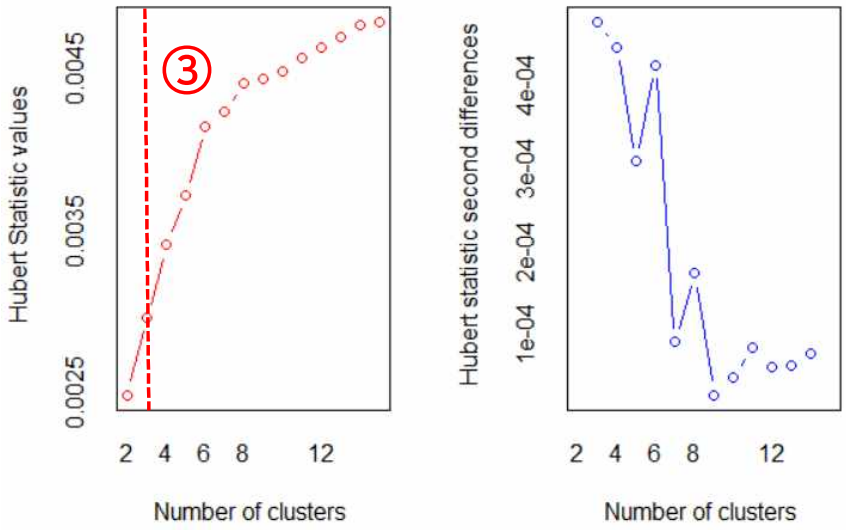
<CODE>

```
## 행정동_업종코드 라벨 생성
for (i in 1:nrow(dat_45)) {
  dat_45[i, 5] <- paste(dat_45[i, 1], "_45")

}
```

```
## 행정동별 분포 확인
ggplot(data = dat_45) +
  geom_density(mapping = aes(x = MCT_SALES, colour = OLD_ADRES))

summary(dat_45)

## 표준화
dat_45[2:3] <- scale(dat_45[2:3])
```

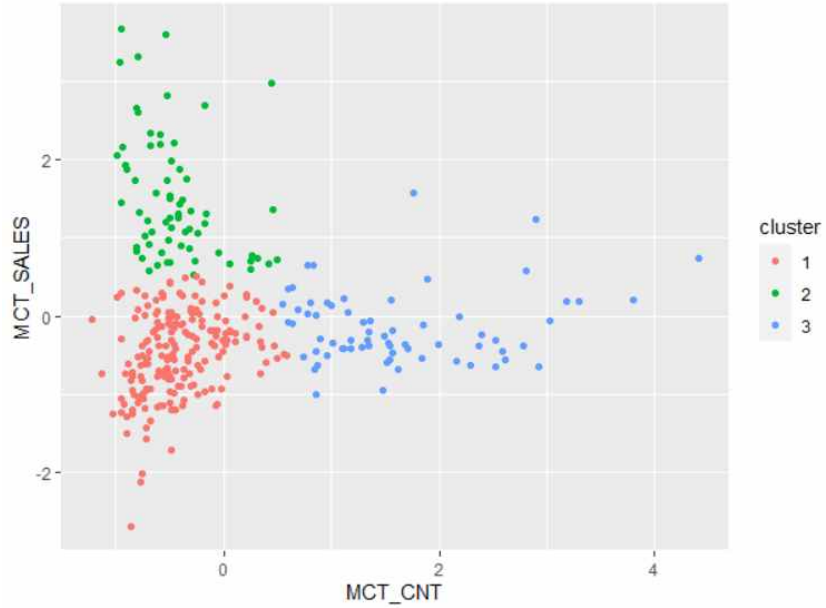
```
## rbind
dat <- rbind(dat_45, dat_46, dat_47, dat_55, dat_56, dat_59, dat_68, dat_69,
             dat_71, dat_73, dat_74, dat_75, dat_85, dat_86, dat_90, dat_91, dat_95, dat_96)
```

1) K 개수 선정



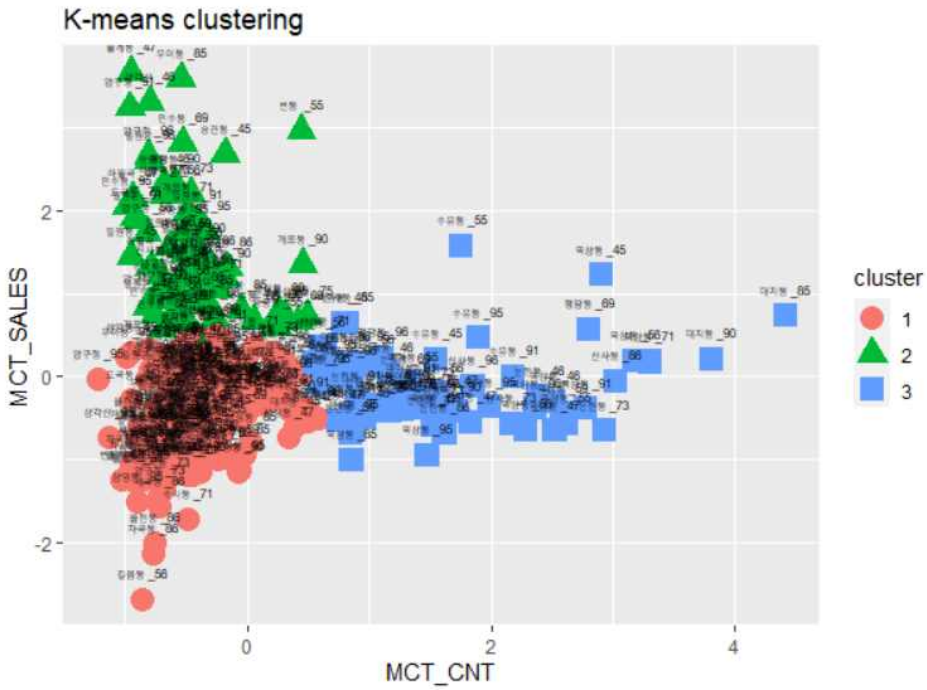
주성분 결과에 따라 K = 3으로 결정

2) 군집화 결과



K = 3일 때 군집화 결과

3) 군집에 행정동_업종 라벨 할당



군집별로 해당 행정동_업종 list와 경쟁도 및 시장 성숙도에 관한 특징을 추출할 수 있음

Clust 1 **행정동_업종코드 list : 길음동_56, 압구정_95, 자곡동_86, …**

저밀도 신생 군집:
행정동 내 경쟁도가 낮고 평균 영업 개월 수도 비교적 낮음

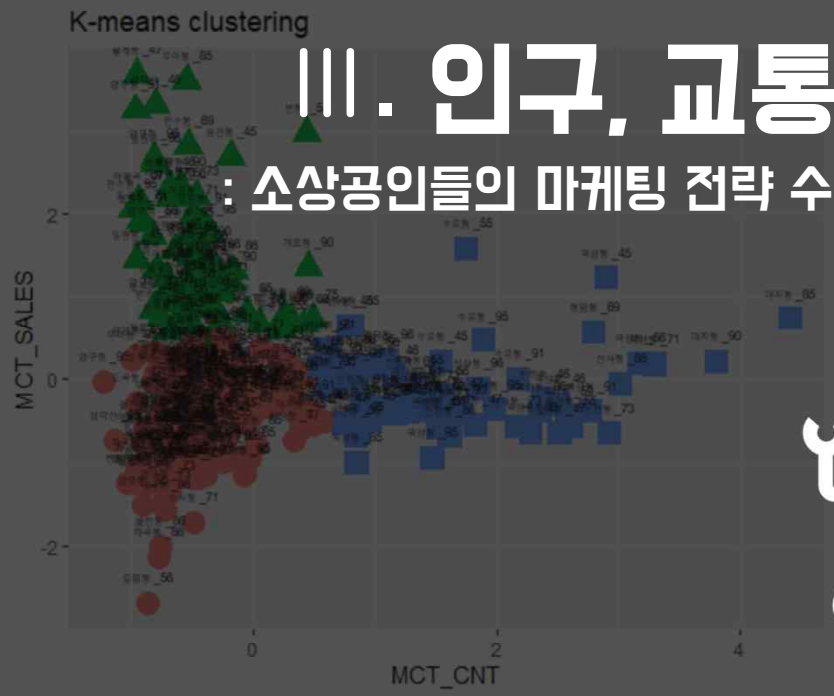
Clust 2 **행정동_업종코드 list : 개포동_90, 우이동_85, 압구정_96, …**

저밀도 정착 군집:
행정동 내 경쟁도가 낮고 평균 영업 개월 수는 비교적 높음

Clust 3 **행정동_업종코드 list : 수유동_55, 역삼동_45, 대치동, …**

완전 경쟁 군집:
행정동 내 경쟁도가 높고 평균 영업 개월 수도 비교적 높음

3) 군집에 행정동_업종 라벨 할당

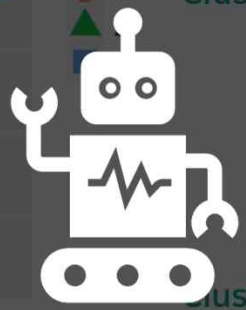


Clust 1 행정동_업종코드 list : 길음동_56, 압구정_95, …

III. 인구, 교통 특성 군집화 모델

: 소상공인들의 마케팅 전략 수립을 위한 AI 활용 리포트 커스터마이징

Clust 2 행정동_업종코드 list : 개포동_90, 우이동_85, 압구정_96, …

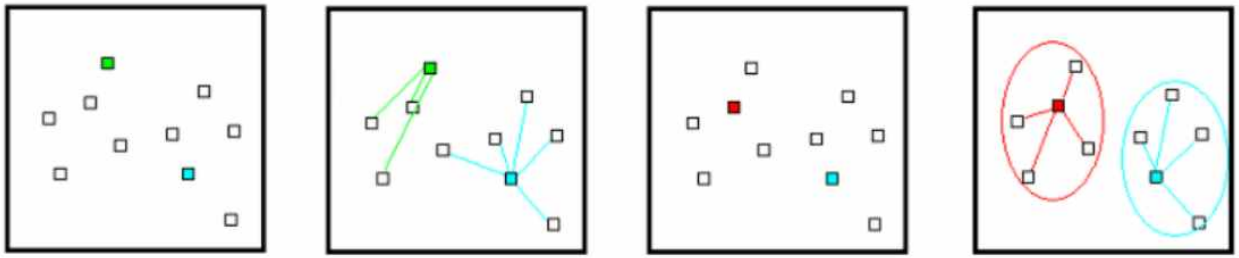


Clust 3 행정동_업종코드 list : 수유동_55, 역삼동_45, 대치동, …

군집별로 해당 행정동_업종 list와 경쟁도 및 시장 성숙도에 관한 특징을 추출할 수 있음

02 인구 및 교통 특성 군집화 모델 – a.프로젝트 개요

≫ 활용 기술 : K-medoids Clustering



- **K-medoids 군집화**: PAM(Partitioning Around Medoids) 이라고도 불리며, 평균을 이용하여 이상치에 민감한 K-means clustering보다 강건한 방법. 평균 대신 **대표주자(medoid)을 선택하고, 더 좋은 군집을 만드는 대표주자가 있으면 대체함**. 소규모 자료 적용에는 유용하지만, 대규모 자료 적용에는 불안정한 특징.

- **Gower 거리**: **혼합형 데이터**의 유사도를 측정하는 가장 널리 알려진 방식으로, 양적변수가 포함된 데이터에도 적용할 수 있음. 선택된 변수들을 [0,1] 사이의 값으로 표준화 시킨 후, 모든 변수들 간의 거리를 가중 평균하여 합한 값을 사용함.

창업예정자용 서비스

≫ 서비스 기능 및 목적

▪ 기능:
사용자는 개업 희망 업종과 지역(행정동)을 입력하여 해당 군집의 **전반적인 인구 및 교통 특성을 제공** 받고 이를 기준으로 군집 내 **유사한 업종과 지역을 추천 받을 수 있음**

▪ 목적:
창업예정자, # 실시간 추천 서비스,
다양한 상권 정보 고려, # 상권분석 리포트 고도화, # 창업 자신감, # 지적 공유

⇨ 창업 예정자의 **불안감 해소** 및 **다양한 창업 환경 고려**

⇨ 소상공인의 **질적 성장 및 품질 경쟁력 향상**

≫ 원본 데이터

▪ 지역과 업종 매핑 정보, 인구 정보, 교통 정보를 다음 10가지 테이블에서 불러온다.

BC외국인 시간대 매출	BC_FORN_TM_SELNG	외국인 시간대 매출데이터
BC외국인 요일별 매출	BC_FORN_WK_SELNG	외국인 요일별 매출데이터
매출금액	SELNG	블록별 서비스업종 매출집계를 관리한다
서울시 단기 외국인 생활인구	FORN_LONG_RESD	장기 외국인 생활인구를 관리한다
서울시 단기 외국인 생활인구	FORN_TEMP_RESD	단기 외국인 생활인구를 관리한다
길단위유동인구	F L P O P _ M D I M	길단위 유동인구정보를 관리한다
블록집계 _ 유동인구	BLCK_FLPOP	블록단위 유동인구정보를 관리한다
임대시세	RENT_CURPRC	행정동별임대시세
주거인구	RESDNG_POPLTN	상주인구 정보를 관리한다
직장인구	WRC_POPLTN	직장인구 정보를 관리한다
가구소득	NEW_HUSCOM	가구소득 정보를 관리한다
아파트단지정보	APT_HSMP_INFO	아파트단지 정보를 관리한다
아파트동정보	APT_APTCMPL_INFO	아파트동정보 정보를 관리한다
아파트호정보	APT_HO_INFO	아파트호정보 정보를 관리한다
버스정거장정보	STTN_INFO	버스정거장정보 정보를 관리한다
지하철역정보	SUBWAY_STATN	지하철역정보 승하차인원정보를 관리한다
집객시설정보	VIATR_FCLTY_INFO	집객시설정보 정보를 관리한다
학교정보	SCHOOL_INFO	학교정보 정보를 관리한다
시군구정보	SIGNGU_DIM	시군구차원을 관리한다.
행정동정보	ADSTRD_DIM	행정동코드를 관리한다.
그리드코드정보	GRID50_DIM	그리도 50차원을 관리한다.
블록정보	BLCK_DIM	블록차원을 관리한다.
발달상권정보	DEVLOP_TRDAR_DIM	발달상권정보를 관리한다.
골목상권정보	ALLEY_TRDAR_DIM	골목상권정보를 관리한다.
공통코드정보	CMMN_CL_CD	공통코드정보를 관리한다.
업종매핑 테이블	/C_INDUTY_MAPNG_NE	업종매핑테이블정보

≫ DATA JOIN: 블록코드 및 업종코드 매핑 테이블 생성

- BLCK_DIM의 블록코드와 ADSTRD_DIM의 행정동을 블록코드 기준으로 합친다.
- SELNG 테이블의 블록코드와 업종코드를 사용하여 지역 정보와 업종 정보를 담고 있는 테이블을 생성한다.

블록코드	행정동코드	행정동명	업종코드	업종명
10019	56182	삼성동	56112	분석전문점
21319	56182	삼성동	85221	외국어학원
22231	90218	대치4동	86102	자동차정비소
40181	90218	대치4동	47110	미용실
48229	90218	대치4동	85221	외국어학원

<CODE>

```
## 지역 및 업종 코드 테이블
Block_table <- BLCK_DIM[, -c(2, 11, 13)]
Adm_table <- ADSTRD_DIM[, c(1:9)]

Block_Sector_table <- SELNG[, c("BLCK_CD", "TA_YM", "KSIC_CD")]
month <- split(Block_Sector_table, Block_Sector_table$TA_YM)
Block_Sector_table <- month$'201912'

## 업종 테이블
Sector_table <- SVC_INDUTY_MAPNG_NEW[, c("SVC_INDUTY_CD_NM", "INDUTY_CD")]
names(Sector_table) <- c("업종명", "KSIC_CD")
Sector_table <- merge(Block_Sector_table, Sector_table, by = 'KSIC_CD')
Sector_table <- merge(Sector_table, Block_table, by = 'BLCK_CD')
Sector_table <- merge(Sector_table, Adm_table, by = 'ADSTRD_CD')
Sector_table <- Sector_table %>% dplyr::select(BLCK_CD, ADSTRD_NM, KSIC_CD, 업종명)
Sector_table <- Sector_table[order(Sector_table$BLCK_CD), ]
```

≫ 데이터 변환: 인구정보 테이블 생성

- 블록코드 단위의 유동인구, 상주인구, 직장인구 테이블을 각각 성별, 나이, 요일, 시간대에 대한 MAX 테이블로 정제한다.
- 유동인구, 상주인구, 직장인구 테이블을 블록코드 단위로 조인하여 하나의 인구정보 테이블을 생성한다.

<CODE>

```
### 상주인구 테이블 정제
Resident_table <- RESDNG_POPLTN[, c(1:12)]
Resident_table$Gender_max <- apply(Resident_table[, c(5:6)], 1, which.max)
Resident_table$Age_max <- apply(Resident_table[, c(7:12)], 1, which.max)
Resident_table <- within(Resident_table, {
  Gender = character(0)
  Gender[Gender_max == 1] = "Male"
  Gender[Gender_max == 2] = "Female"
  Gender = factor(Gender, level = c("Male", "Female"))

  Age = character(0)
  Age[Age_max == 1] = "10대"
  Age[Age_max == 2] = "20대"
  Age[Age_max == 3] = "30대"
  Age[Age_max == 4] = "40대"
  Age[Age_max == 5] = "50대"
  Age[Age_max == 6] = "60대"
  Age = factor(Age, level = c("10대", "20대", "30대", "40대", "50대", "60대"))
})
```

≫ 결측치 처리: 평균값으로 대체

- 인구정보 테이블에서 결측치가 발생한 블록코드는 해당 행정동의 평균값으로 대체한다.

<CODE>

```
### 결측치 처리 - 해당 행정동의 평균으로 대체
Population[!complete.cases(Population),]

Population[Population$ADSTRD_CD == 11680580, ]
Population[860, '가구수'] <- mean(Population[Population$ADSTRD_CD == 11680580, ]$가구수)
Population[860, '평균가구소득'] <- 0
Population[860, '평균가구소득'] <- mean(Population[Population$ADSTRD_CD == 11680580, ]$평균가구소득)
```

≫ 데이터 변환: 교통정보 테이블 생성

- X, Y좌표 단위의 버스정보 데이터를 블록코드 단위로 환산한다.
- 해당 블록 안에 해당하는 행 정보가 있으면 'Bus' 열을 1로 지정하여 버스정류장이 있음을 표시한다.

<CODE>

```
## 교통정보 테이블 - X,Y 좌표를 블록코드,행정동 단위로 변경

### 버스유무 테이블
Bus_table <- STTN_INFO[, c("STTN_ID", "XCNTS_VALUE", "YDNTS_VALUE")]
Bus_table[!complete.cases(Bus_table),]
Bus_table <- Bus_table[complete.cases(Bus_table),]    # 결측치 제거
rownames(Bus_table) <- NULL

Bus <- Block_table[, c('BLCK_CD', 'ADSTRD_CD')]
Bus$Bus <- NA

for (i in (1:nrow(Bus))){
  Block_bus <- Bus_table[Bus_table$XCNTS_VALUE >= Block_table[i, 'XCNTS_MIN_VALUE'] &
                         Bus_table$XCNTS_VALUE <=  Block_table[i, 'XCNTS_MAX_VALUE'] &
                         Bus_table$YDNTS_VALUE >=  Block_table[i, 'YDNTS_MIN_VALUE'] &
                         Bus_table$YDNTS_VALUE <=  Block_table[i, 'YDNTS_MAX_VALUE'], ]
  Bus[i, 'Bus'] = ifelse(nrow(Block_bus) == 0, 0, 1)
}
```

≫ DATA JOIN: 인구정보 테이블과 교통정보 테이블을 블록코드 단위로 조인

- 지역블록 및 업종 테이블을 기본키로 사용하여 이를 기준으로 합친다.

기본키 1 기본키 2 <data set>

BLCK_CD	ADSTRD_NM	업종명	유동인구수	유동인구성별	유동인구나이	유동인구시간	유동인구요일	상주인구수	상주인구성별	상주인구나이	직장인구수	직장인구성별	직장인구나이	가구수	평균가구소득	행정동	Bus	Subway
164	논현2동	일식음식점	10011	Female	30대	6	FRI	10	Female	60대	65	Male	30대	1	2391548	논현2동	0	0
213	청담동	세탁소	6628	Female	60대 이상	6	FRI	82	Female	20대	21	Male	30대	28	3749032	청담동	0	0
1698	인수동	청과상	8097	Female	60대 이상	6	SUN	42	Female	60대	22	Male	40대	16	2054794	인수동	0	0
1698	인수동	청과상	8097	Female	60대 이상	6	SUN	42	Female	60대	22	Male	40대	16	2054794	인수동	0	0
1698	인수동	청과상	8097	Female	60대 이상	6	SUN	42	Female	60대	22	Male	40대	16	2054794	인수동	0	0

≫ Group_by: 집단별 요약 테이블 생성

- 블록코드 단위의 최종 데이터셋을 행정동 별로 합쳐 새로운 요약 데이터셋을 생성한다.
- 수치형 변수는 행정동 내 블록코드들의 합산 값으로, 범주형 변수는 행정동 내 블록코드들의 최대 빈도수를 갖는 값으로 환산한다.

Group_by

<data set>

행정동	유동인구수	유동인구성별	유동인구나이	유동인구시간	유동인구요일	상주인구수	상주인구성별	상주인구나이	직장인구수	직장인구성별	직장인구나이	가구수	평균가구소득	버스수	지하철수
개포1동	12691	Female	60대 이상	6	FRI	596	Female	60대	2882	Male	50대	627	13941934	1	0
개포2동	173922	Female	60대 이상	6	SUN	1445	Female	60대	1121	Male	50대	668	49019469	2	0
개포4동	932510	Female	60대 이상	6	MON	11448	Female	60대	4090	Male	40대	4031	240390482	20	0
논현1동	2678434	Female	30대	6	FRI	17727	Female	30대	33268	Male	30대	6314	559004523	21	3

Data Shape: 35x16

업종명	유동인구수	유동인구성별	유동인구나이	유동인구시간	유동인구요일	상주인구수	상주인구성별	상주인구나이	직장인구수	직장인구성별	직장인구나이	가구수	평균가구소득	버스수	지하철수
DVD방	713354	Female	30대	21	THU	2282	Male	30대	9825	Male	30대	944	93370824	8	0
PC방	1904669	Female	60대 이상	21	THU	10182	Male	60대	21595	Male	30대	3835	291772847	43	0
가구	1632118	Female	30대	11	THU	11059	Female	30대	56980	Male	30대	5704	373036172	32	2
가방	749547	Female	30대	21	FRI	3358	Female	60대	20263	Female	30대	1221	171045515	13	1

Data Shape: 98x16

인구 및 교통 특성 군집화 모델 – c.분석 결과

≫ Gower 유사도 측정

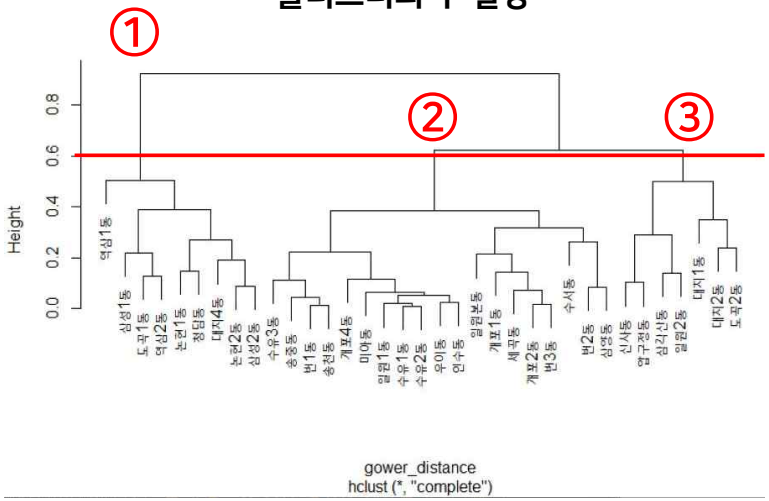
군집화 기준 변수: 8개의 범주형과 6개의 숫자형의 혼합 형태

행정동	유동인구수	유동인구성별	유동인구나이	유동인구시간	유동인구요일	상주인구수	상주인구성별	상주인구나이	직장인구수	직장인구성별	직장인구나이	가구수	평균가구소득	버스수	지하철수
개포1동	12691	Female	60대 이상	6	FRI	596	Female	60대	2882	Male	50대	627	13941934	1	0
개포2동	173922	Female	60대 이상	6	SUN	1445	Female	60대	1121	Male	50대	668	49019469	2	0
개포4동	932510	Female	60대 이상	6	MON	11448	Female	60대	4090	Male	40대	4031	240390482	20	0
논현1동	2678434	Female	30대	6	FRI	17727	Female	30대	33268	Male	30대	6314	559004523	21	3
논현2동	2520309	Female	30대	6	THU	13655	Female	30대	38398	Male	30대	5166	623359517	28	5

업종명	유동인구수	유동인구성별	유동인구나이	유동인구시간	유동인구요일	상주인구수	상주인구성별	상주인구나이	직장인구수	직장인구성별	직장인구나이	가구수	평균가구소득	버스수	지하철수
DVD방	713354	Female	30대	21	THU	2282	Male	30대	9825	Male	30대	944	93370824	8	0
PC방	1904669	Female	60대 이상	21	THU	10182	Male	60대	21595	Male	30대	3835	291772847	43	0
가구	1632118	Female	30대	11	THU	11059	Female	30대	56980	Male	30대	5704	373036172	32	2
가방	749547	Female	30대	21	FRI	3358	Female	60대	20263	Female	30대	1221	171045515	13	1
가전제품	655791	Female	30대	6	FRI	3292	Female	30대	13312	Male	30대	1294	148101678	10	1

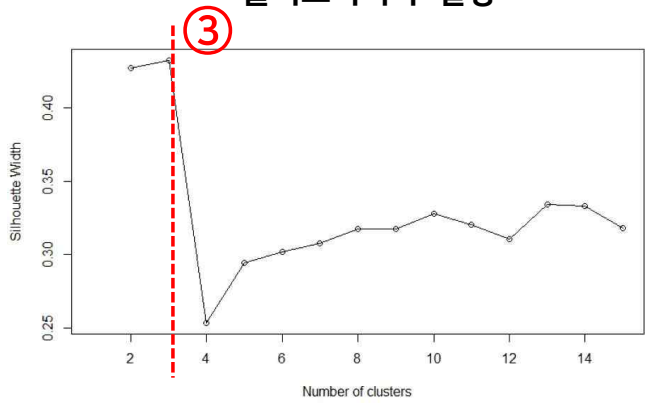
≫ 행정동별 데이터 분석: 클러스터 수 결정

응집형 계층적 군집 분석을 통한 클러스터의 수 결정



두 군집 간의 최장 거리를 군집간 거리로 정의하는 최장연결법으로 계층적 군집 분석 알고리즘을 생성하였다. 생성된 군집의 계층 트리를 살펴보면 height 0.6에서 계층도를 잘라내었을 때 군집의 수 3개로 가장 적절해 보인다.

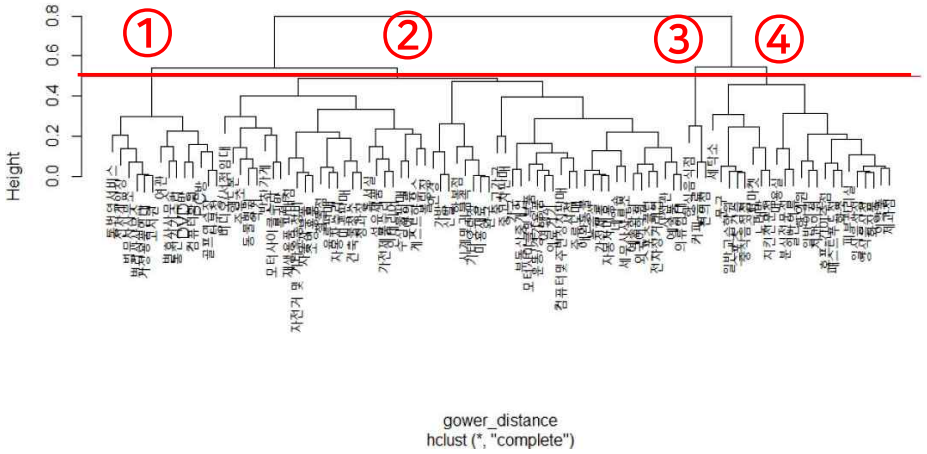
Silhouette 방법을 통한 클러스터의 수 결정



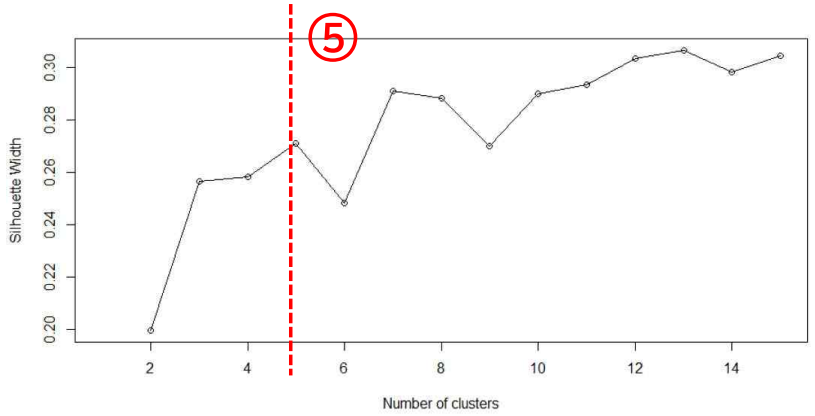
Silhouette 방법을 사용하여 클러스터의 수에 따른 silhouette 너비를 살펴본 결과, 클러수터의 수를 3개로 하였을 때 가장 효과적으로 그룹이 나뉠 것으로 보인다. 따라서 행정동은 클러스터의 수를 3개로 하여 군집화를 진행하였다.

≫ 업종별 데이터 분석: 클러스터 수 결정

응집형 계층적 군집 분석을 통한 클러스터의 수 결정



Silhouette 방법을 통한 클러스터의 수 결정



두 군집 간의 최장 거리를 군집간 거리로 정의하는 최장연결법으로 계층적 군집 분석 알고리즘을 생성하였다. 생성된 군집의 계층 트리를 살펴보면 height 0.5를 기준으로 군집을 생성하는 것이 적절해 보이지만, 군집의 수가 명확하지 않아 shilhouette 방법으로 결정하였다.

Silhouette 방법을 사용하여 클러스터의 수에 따른 silhouette 너비를 살펴본 결과, 클러스터의 수가 6일 때를 제외하고는 많을수록 더 좋은 군집이 형성되는 것처럼 보이지만, 적절한 군집 개수를 설정하기 위해 업종은 클러스터의 수를 5개로 하여 군집화를 진행하였다.

≫ 행정동 군집 결과

Clust 1 행정동 list : 개포1동, 개포2동, 미아동, 변1동, 변2동 …

유동 인구 성별	
남	0
여	19

유동 인구 나이	
10대	0
30대	0
60대	19

유동 인구 시간	
06시	18
11시	1
21시	0

유동 인구 요일	
SUN	17
MON	1
THE	0
THU	0
FRI	1
SAT	0

상주 인구 성별	
남	4
여	15

상주 인구 나이	
10대	1
30대	0
40대	0
60대	18

직장 인구 성별	
남	16
여	3

직장 인구 나이	
30대	0
40대	8
50대	11

…

→ 군집별로 해당 행정동 list와 인구, 교통 특성을 살펴볼 수 있다.

Clust 2 행정동 list : 논현1동, 논현2동, 대치4동, 삼성1동, 삼성2동 …

Clust 3 행정동 list : 대치1동, 대치2동, 도곡2동, 신사동, 압구정동 …

≫ 업종 군집 결과

Clust 1 업종 list : DVD방, 가구, 가정용품임대, 고시원, 법무사사무소, 변리사사무소, …

유동인구 성별	
남	6
여	8

유동인구 나이	
20대	0
30대	14
60대	0

유동인구 시간	
06시	1
11시	9
14시	0
17시	1
21시	3

유동인구 요일	
SUN	1
THU	13
FRI	0
SAT	1

상주인구 성별	
남	12
여	2

상주인구 나이	
20대	1
30대	8
40대	1
50대	0
60대	4

직장인구 성별	
남	13
여	1

직장인구 나이	
20대	0
30대	13
40대	1
50대	0
60대	0

…

Clust 2 업종 list : PC방, 가전제품수리, 건축물청소, 노래방, 모터사이클및부품, 모터사이클수리, …

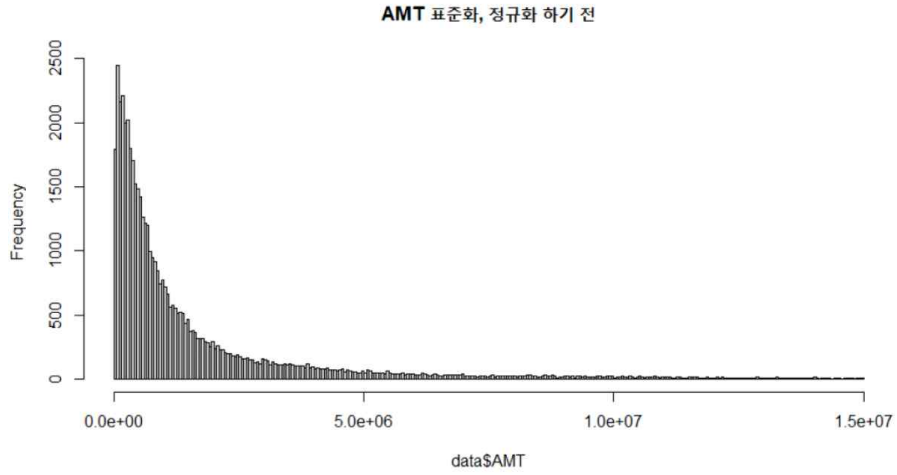
Clust 3 업종 list : 가방, 가전제품, 게스트하우스, 골프연습장, 기타오락장, 네일숍, …

Clust 4 업종 list : 당구장, 문구, 분식전문점, 사진관, 스포츠 강습, 양식음식점, …

Clust 5 업종 list : 미용실, 커피전문점, 편의점, 한식음식점

모델 보완점

1) 매출 점수 프로젝트

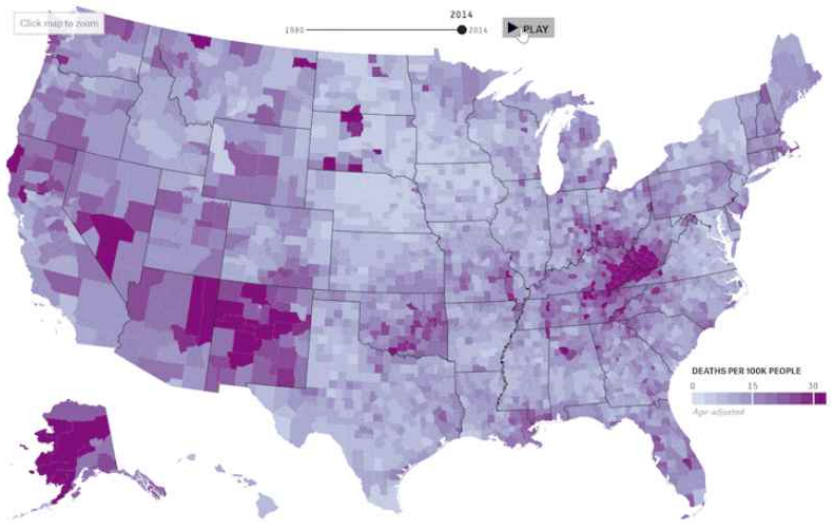


- 블록코드 단위 매출액과 매출 건수 자체의 분포에 있어서, 표준화 및 정규화를 진행하여도 왜도가 잡히지 않아 모델의 정확도가 높게 나타나지 않았다. 만약 매출액/건수의 분포를 왜곡하는 원인을 찾아 조정한다면 더욱 정확하게 매출금액/건수를 예측할 수 있을 것이다.

- 타겟 연령층, 요일, 시간대에 입각한 매출 점수를 도출하는 것은 가능하지만, 그 이후에 소상공인들이 매출 점수를 높이기 위한 전략을 함께 마련해준다면 더 유용한 정보를 제시할 수 있다.

모델 보완점

2) K-means를 통한 지역 경쟁도 군집화



- 3년치 데이터를 기반으로 분류한 결과이므로 “평균 영업 개월 수“ 및 “가맹점 수“ 항목에 있어 실제 해당 지역, 업종의 상황을 정확하게 반영하지 못하였을 수 있다.
- 최종 군집화 결과를 시간 흐름에 따라 지도 상에 시각화 하여 경쟁도에 대한 업종별 변화 양상을 제안할 수 있다.

모델 보완점

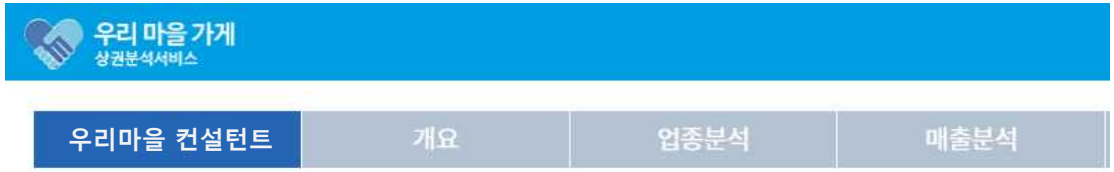
3) 인구 및 교통 특성 기반 군집화

- 행정동 단위로 군집화를 진행하였는데, 행정동이 아닌 지역블록단위로 군집화를 진행하여 지도상에 시각화 해본다면 지역블록의 인구와 교통 특성을 더욱 세밀하게 반영할 수 있을 것이다. 마찬가지로, 강남구와 강북구 뿐만 아니라 서울시 전체의 행정동 데이터로 군집화를 진행한다면 참조할 만한 데이터의 수가 많아지므로 군집 내 유사도가 더욱 높아질 것이다.

- 인구 특성과 교통 특성을 같이 반영하는 군집화가 아닌 인구 특성 기준 군집과 교통 특성 군집을 따로 생성해본다면 해당 특성을 더욱 정확하게 반영하는 군집이 형성될 것으로 기대된다.

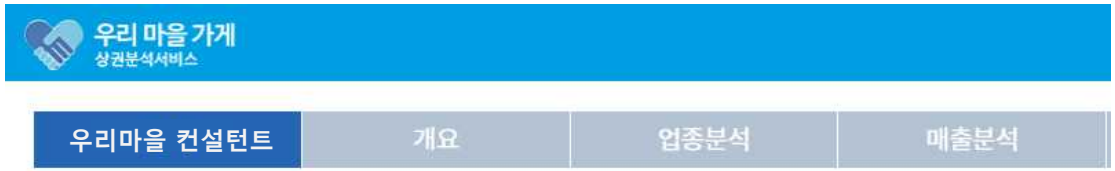
‘우리마을 컨설턴트’ 화면 구성 프로토타입

‘우리마을 컨설턴트’ 화면 구성 프로토타입



어떤 서비스를 원하십니까?

‘우리마을 컨설턴트’ 화면 구성 프로토타입



≫ 희망 업종: 여행사

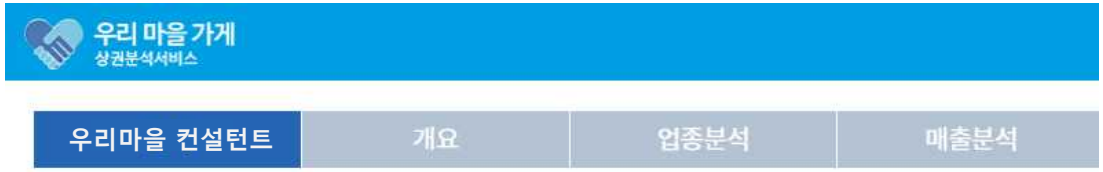
[여행사] 주변 주요 유동인구 성별은 [여자] 입니다.
…
[여행사] 주변 지역 지하철수는 [5]개 입니다.

모든 특성을 고려해 보았을 때, [여행사]와 비슷한 업종 목록에는
[DVD방, 가구, 가정용품임대, 고시원, 법무사사무소, …]가 있습니다.

≫ 희망 지역의 경쟁도 수준

해당 지역은 [완전경쟁 지역] 입니다.
[완전경쟁 지역] 의 특징은 ---입니다.
또다른 [완전경쟁 지역] 으로는 ---가 있습니다.
[완전경쟁 지역] 에서 주로 취해야 할 마케팅 전략은 ---입니다.

‘우리마을 컨설턴트’ 화면 구성 프로토타입



≫ 매출 실적 점수

매출 점수 30 매출 등급 A

현재 [삼성동], [분식전문점]의 매출 대비 당신의 점수 입니다.

≫ 현재 지역의 경쟁도 수준

해당 지역은 [완전경쟁 지역] 입니다.
[완전경쟁 지역] 의 특징은 ---입니다.
또다른 [완전경쟁 지역] 으로는 ---가 있습니다.
[완전경쟁 지역] 에서 주로 취해야 할 마케팅 전략은 ---입니다.

기대 효과

1) 소상공인들의 현 상황을 반영하는 맞춤 서비스 제공

서비스를 이용하는 사람들로부터 더 많은 입력을 받아 고려해감으로써 이용자들의 개별 상황을 더욱 구체적으로 반영하는 방향으로 상권분석서비스를 고도화 할 수 있다.

2) AI 기술을 도입한 발 빠른 정보 제시

실시간으로 쏟아지는 대용량의 데이터를 신경망, 클러스터링 등의 AI 기술을 통해 효과적으로 정보를 정리하여 제시할 뿐 아니라, 새로운 정보까지 제시할 수 있다.

3) 투명하고 공정한 지역 경제 활성화

고도화된 분석 서비스를 공공 사이트에서 제공함으로써 소상공인들 간의 정보 격차를 줄이고 지역 경쟁에서 현재 자신의 위치를 정확하게 파악하여 모든 소상공인들이 정보 사회에서 도태되지 않도록 돕는다.