채널 D
2016125077 최재혁
2015125043 어성준
2015125080 표정진

데이터 사이언스

- 12주차 팀 과제 (공공 데이터를 활용한 군집분석)

목 차

데이터 셋 선정, 속성

- 데이터 셋 선정
  - 공공 데이터를 알아보던 중, 서울 공공데이터에서 청소년 5대 범죄의 데이터를 찾아 수행
  - 링크 : https://data.seoul.go.kr/dataList/datasetList.do
- 데이터 속성 조사
  - 구 별로 나와있는 데이터를 선택
  - 살인, 강도, 강간_추행, 절도, 폭력 등으로 구성됨

	A	B	C	D	E	F
1	구분	살인	강도	강간·추행	절도	폭력
2	중부	0	0	1	47	19
3	종로	0	0	2	21	16
4	남대문	0	0	0	14	3
5	서대문	0	3	9	89	85
6	혜화	0	0	2	19	17
7	용산	0	4	3	21	35
8	성북	0	0	9	54	71
9	동대문	0	2	4	112	126
10	마포	0	0	18	127	98
11	영등포	0	0	6	144	150
12	성동	0	0	3	42	52
13	동작	0	0	10	72	113
14	광진	0	0	12	115	191
15	서부	0	0	2	93	43
16	강북	0	0	8	82	147
17	금천	0	0	5	65	76
18	중랑	0	5	4	137	155
19	강남	0	0	9	66	44
20	관악	0	14	15	102	190

<사진 1> 지역별, 5대 범죄 발생 빈도 표

거리척도 계산

- 유클라디안 거리척도
  - 두 점을 잇는 가장 짧은 직선 거리로써, 5개의 속성들에 대해서 수학적 계산을 진행
  - Method를 "euclidean“ 으로 설정
  - 벡터 공간에 최단 연결 경로로써 가장 많이 사용하는 거리척도
- 맨하탄 거리척도
  - 가로, 세로로 블록 수를 더한 거리 척도
  - 2차원 공간에서는 적절하나 다차원 공간에서는 적절하지 않음

```
> m_dist <- dist(x, method = "manhattan")
> m_dist
       중부 종로 남대문 서대문 혜화 용산 성북 동대문 마포 영등포 성동 동작 광진 서부 강북
종로     30
남대문   50   22
서대문  119  147    169
혜화     31    3     21    148
용산     48   24     46    125   25
성북     67   95    117     52   96   79
동대문  177  205    227     70  206  185  120
마포     176  204    226     63  205  188  109     59
영등포  233  261    283    126  262  245  172     60   81
성동     40   58     80     89   59   42   37    147  146   203
동작     128  156    178     49  157  140   61     61   78   113   98
광진     251  279    301    138  280  263  184     78  111    76  221  123
서부     71   99     121     56  100   85   74    106  105   162   61   99  180
강북     170  198    220     73  199  182  105     57  104    67  140   46   81  121
금천     79   107    129     40  108   91   20    100   97   154   49   49  172   64   91
중랑     234  262    284    125  263  238  177     57   86    19  204  118   71  163   72
강남     52   80     102     67   81   64   39    135  124   187   38   76  199   35  120
관악     254  282    304    135  283  258  187     97  134   105  224  126   31  183   84
강서     343  371    393    226  372  351  276    166  181   110  313  215  142  272  173
강동     295  323    345    176  324  299  228    118  135    62  265  167  120  224  125
종암     77   105    127     42  106   89   22    100   99   156   47   51  174   70   93
구로     281  309    331    168  310  293  214    108  111    52  251  153  116  210  111
```

<사진 1> 맨하탄 거리척도

```
> x_dist <- dist(x, method = "euclidean")
> x_dist
              중부        종로      남대문      서대문        혜화        용산        성북
종로      26.191602
남대문    36.687873  14.899664
서대문    78.695616  97.175100 111.530265
혜화      28.089144   2.236068  15.000000  97.887691
용산      30.854497  19.442222  33.136083  84.622692  18.574176
성북      53.075418  64.521314  79.404030  37.815341  64.730209  49.365980
동대문   125.247754 142.790056 157.330862  47.286362 143.310851 128.712859  80.112421
마포      113.710158 134.966663 148.721216  41.267421 135.944842 124.281938  78.351771
영등포   163.079735 181.936802 196.328806  85.252566 182.565057 168.460678 119.791486
성동      33.436507  41.689327  56.515485  57.818682  41.892720  27.313001  23.259407
동작      97.683161 109.881755 124.755761  32.908965 109.949989  93.541435  45.705580
광진     185.280868 198.899472 213.749854 109.224539 198.977386 182.397917 134.647688
서부      51.894123  76.896034  88.572005  42.871902  78.434686  72.560320  48.518038
강북     132.883408 144.630564 159.449051  62.473995 144.585615 127.694949  81.000000
금천      59.908263  74.464757  89.190807  26.115130  74.873226  60.307545  12.727922
중랑     163.187009 181.124267 195.637420  85.047046 181.650764 166.907160 118.300465
강남      32.403703  53.460266  66.828138  47.106263  54.653454  46.454279  29.546573
관악     180.715246 192.878200 207.687265 106.541072 192.828940 175.584737 129.216872
강서     234.036322 256.396178 269.853664 161.412515 257.429602 245.652193 198.756132
강동     189.412249 210.049994 223.993303 113.727745 210.883854 197.587955 150.286393
종암      62.617889  75.478474  90.260733  29.223278  75.723180  60.340699  12.884099
구로     189.422807 210.924157 224.702025 115.021737 211.863163 199.562020 152.072351
서초      64.822835  77.820306  92.660671  27.549955  78.108898  63.039670  14.525839
양천     158.653711 180.208213 193.935556  84.746681 181.162910 169.041415 122.122889
송파     181.422160 201.345474 215.531900 104.599235 202.116303 188.610710 140.402991
노원     188.793538 201.799405 216.686409 113.145923 201.826658 185.070257 137.513636
```

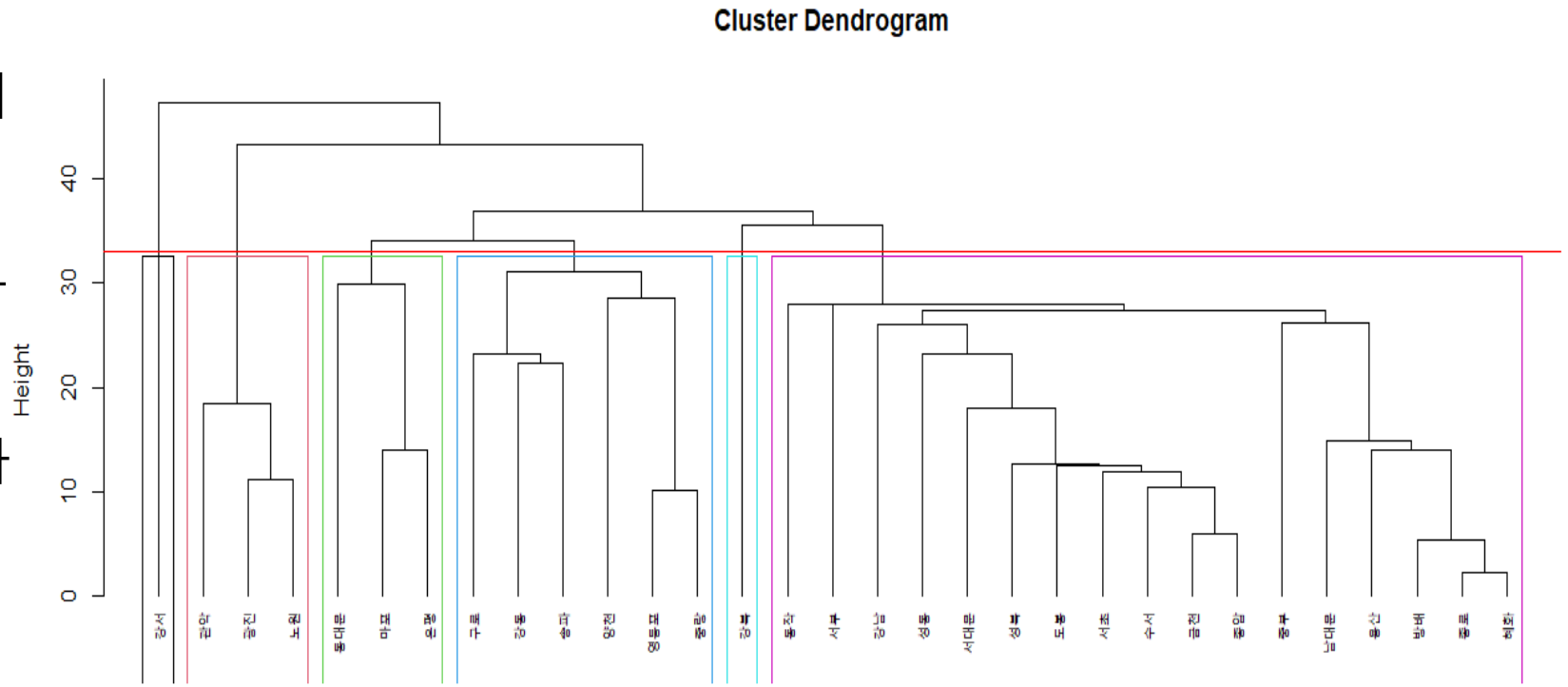
<사진 2> 유클라디안 거리척도

H-clust() 함수로 군집화

- Hclust() 함수 적용
  - 우선 거리척도로 유클라디안 거리척도를 이용
  - 거리 계산 기준은 최단연결법 (single)으로 사용
  - 군집화 개수는 총 6개로 설정하였으며, height 33에 선을 그어 덴드로그램에 표시

```
#전체적인 범죄 성향이 비슷한 집단으로 군집화 진행행
hc <- hclust(dist(x, method = "euclidean", diag = TRUE),
             method = "single")
hc
plot(hc, hang= -1, cex=0.8)
rect.hclust(hc, k = 6, border = 1:6)
abline(h = 33, col="red")
```

- Hclust() 적용 결과
  - 간단히 적용한 결과 총 6개의 군집이 생성된 것을 확인
  - 각 지역별로 5개의 범죄의 유사성을 가지고 군집화 진행
  - '강서' 지역 같은 경우, 유일하게 살인이 일어난 지역으로 독립적 군집형성 확인



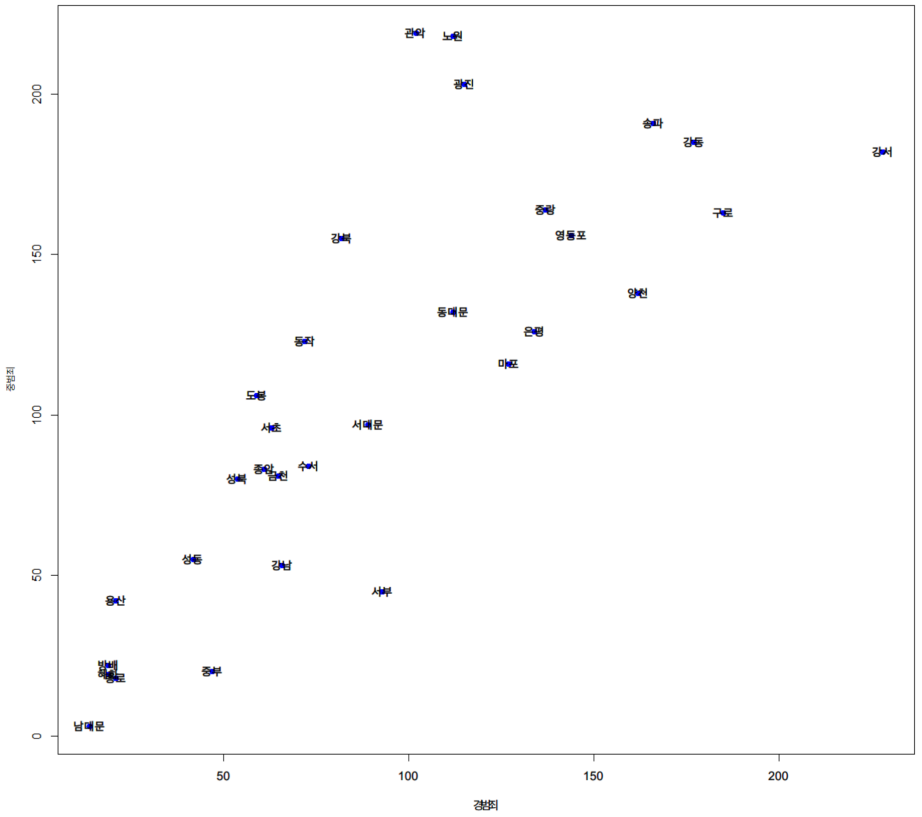
<사진 1> 군집화 덴드로그램

중범죄와 경범죄 - 1

- 데이터 전처리
  - 중범죄 : 살인, 강도, 강간_추행, 폭력
  - 경범죄 : 절도
  - 중/경범죄 분류 합을 구별로 구하여 전처리

- Scatter plot확인

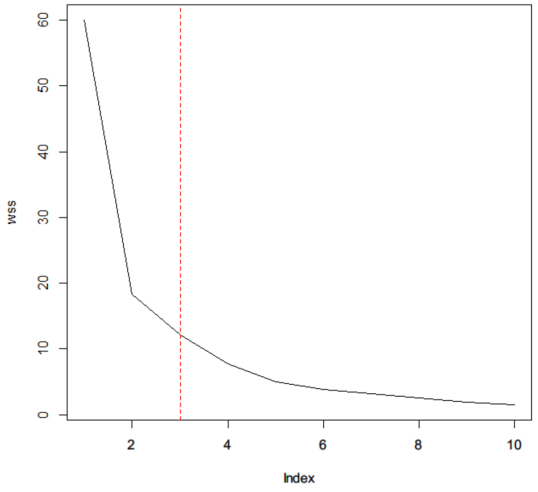
```
x_felony <- x[1:31,c(1,2,3,5)]
x_felony <- apply(x_felony,1,sum)
x_felony # 각 구의 중범죄 발생 합
y <- x[,4] # 경범죄 (절도) 발생 합 = 절도 발생 값
x <- cbind(x_felony,y)
colnames(x) <- c("중범죄","경범죄")
x
```



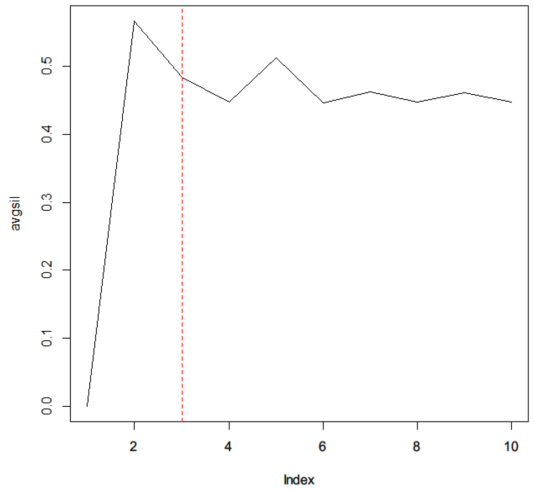
	중범죄	경범죄
중부	20	47
종로	18	21
남대문	3	14
서대문	97	89
혜화	19	19
용산	42	21
성북	80	54
동대문	132	112
마포	116	127
영등포	156	144
성동	55	42
동작	123	72
광진	203	115
서부	45	93
강북	155	82
금천	81	65
중랑	164	137
강남	53	66

중범죄와 경범죄 - 2

- 군집화 갯수
  - 클러스터링 집단 수를 1~10 plot 후 결과 확인
  - 기준 : 팔꿈치 포인트 & 평균 실루엣 값
  - 결과 : 2
- plot확인



<사진 1> 팔꿈치 포인트 Plot



<사진 2> 평균 실루엣 Plot

```r
result <- NULL
for (k in 1:10){
  result[[k]] <- kmeans(df, k, nstart = 25)
}

## 1) 팔꿈치 포인트
# The total within-cluster sum of square가 작을수록 좋습니다.
wss <- numeric(10)
for(k in 1:10){
  wss[k] <- result[[k]]$tot.withinss
}
plot(wss,type="l")
abline(v=3,col="red",lty=2)

## 2) 평균 실루엣
# 평균 실루엣이 클수록 좋습니다.
avgsil<-numeric(10)
for (k in 2:10){
  si <- summary(silhouette(result[[k]]$cluster, dist(df)))
  avgsil[k] <- si$avg.width
}
avgsil
plot(avgsil,type="l")
abline(v=3,col="red",lty=2)
```

중범죄와 경범죄 - 3

- 군집화(K-means)
  - BSS : 클러스터 “간”의 분산 정도
  - TSS : 클러스터 “내”의 분산 정도
  - BSS/TSS 값은 0.698
  - 1에 근접할 수록 좋은 클러스터링 결과
  - 비교적 좋다고 판단 가능함

  - 각 17과 14 사이즈 클러스터
  - 벡터 1 클러스터 : 범죄 발생 값이 낮은 집단
  - 벡터 2 클러스터 : 범죄 발생 값이 높은 집단

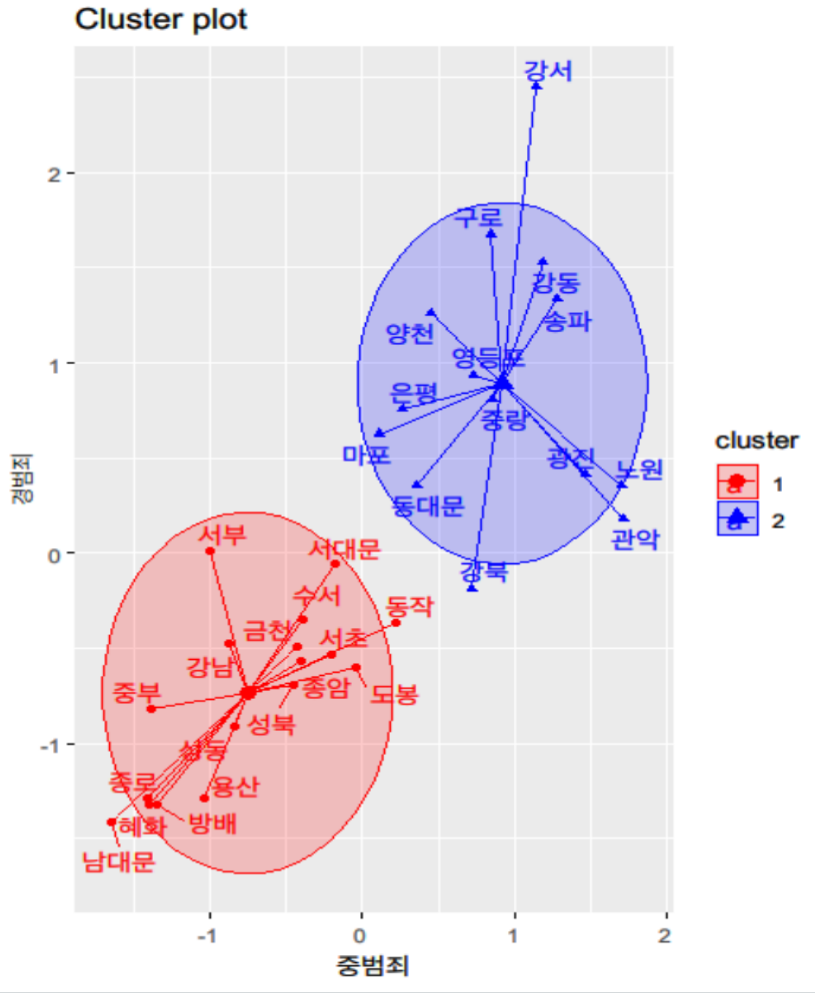
```
K-means clustering with 2 clusters of sizes 17, 14

Cluster means:
     중범죄    경범죄
1  60.41176  51.64706
2 167.71429 141.64286

Clustering vector:
  중부   종로 남대문 서대문   혜화   용산   성북 동대문   마포 영등포   성동   동작   광진   서부   강북   금천   중랑
     1      1      1      1      1      1      1      2      2      2      1      1      2      1      2      1      2
  강남   관악   강서   강동   종암   구로   서초   양천   송파   노원   방배   은평   도봉   수서
     1      2      2      2      1      2      1      2      2      2      1      2      1      1

Within cluster sum of squares by cluster:
[1] 31232.00 33848.07
 (between_SS / total_SS =  69.8 %)
```

```
fviz_cluster(km.res, data=x, palette=c("red","blue","green"),
             ellipse.type="euclid", star.plot=T, repel=T, ggtheme=theme())
```



	중범죄	경범죄	cluster
중부	20	47	1
종로	18	21	1
남대문	3	14	1
서대문	97	89	1
혜화	19	19	1
용산	42	21	1
성북	80	54	1
동대문	132	112	2
마포	116	127	2
영등포	156	144	2
성동	55	42	1
동작	123	72	1
광진	203	115	2
서부	45	93	1
강북	155	82	2
금천	81	65	1
중랑	164	137	2
강남	53	66	1
관악	219	102	2
강서	182	228	2
강동	185	177	2
종암	83	61	1
구로	163	185	2
서초	96	63	1
양천	138	162	2
송파	191	166	2
노원	218	112	2
방배	22	19	1
은평	126	134	2
도봉	106	59	1
수서	84	73	1

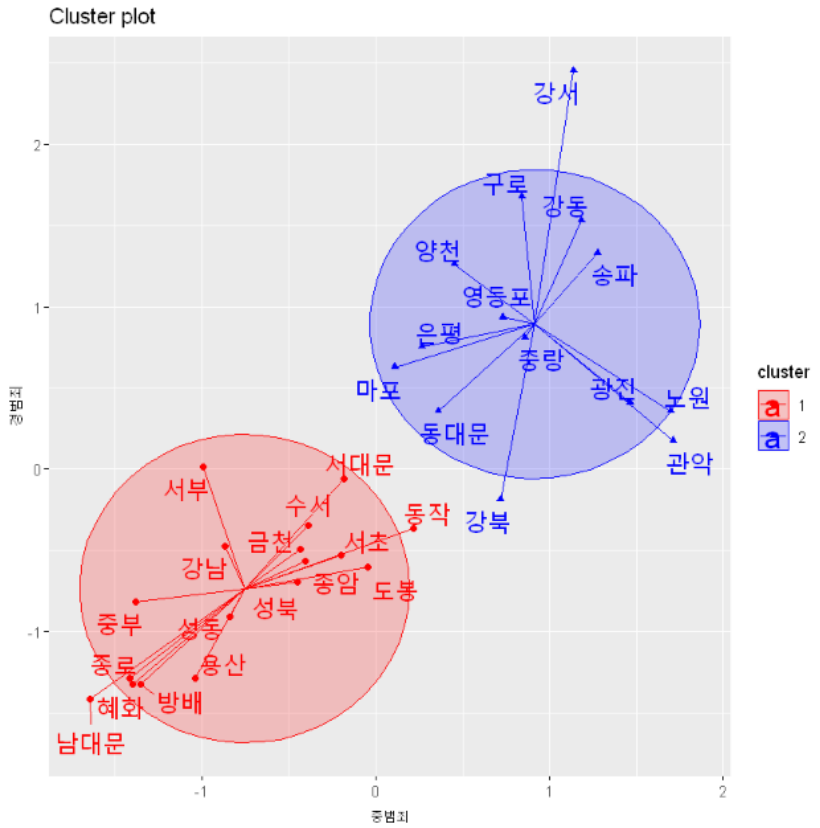
중범죄와 경범죄 - 4

- 군집화(k-Medoids)
  - Hard clustering
  - 이상치에 대해 강건함
  - 초기 medoids 집합을 찾은 뒤(빌드)
  - 목적 함수에 대한 지역적 최소값을 찾는다(스왑)

  - build: build 단계에서의 목적 함수의 값
  - swap: swap 단계에서의 목적 함수의 값

```
Medoids:
      ID 중범죄 경범죄
성북  7     80     54
중랑 17    164    137
Clustering vector:
  중부   종로 남대문 서대문   혜화   용산   성북 동대문   마포 영등포   성동
     1      1      1      1      1      1      1      2      2      2      1
  동작   광진   서부   강북   금천   중랑   강남   관악   강서   강동   종암
     1      2      1      2      1      2      1      2      2      2      1
  구로   서초   양천   송파   노원   방배   은평   도봉   수서
     2      1      2      2      2      1      2      1      1
Objective function:
   build     swap
51.56966 42.23114
```

```
fviz_cluster(pam.result,data=x,palette=c("red","blue","green"),
             ellipse.type="euclid",star.plot=T,repel=T,ggtheme=theme(), labelsize=20)
```

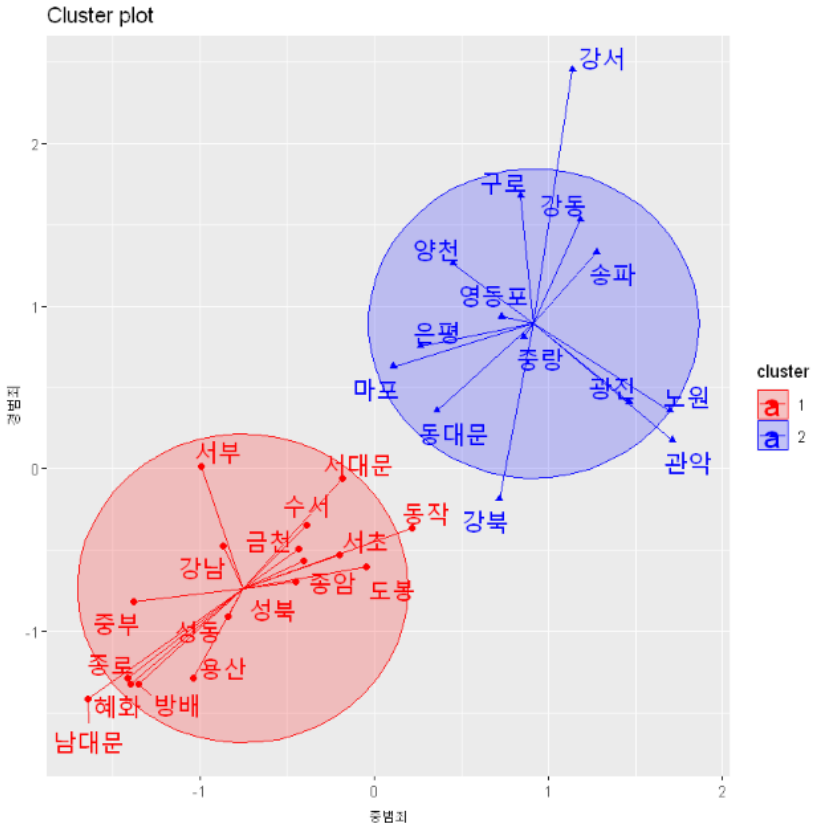


	중범죄	경범죄	cluster
중부	20	47	1
종로	18	21	1
남대문	3	14	1
서대문	97	89	1
혜화	19	19	1
용산	42	21	1
성북	80	54	1
동대문	132	112	2
마포	116	127	2
영등포	156	144	2
성동	55	42	1
동작	123	72	1
광진	203	115	2
서부	45	93	1
강북	155	82	2
금천	81	65	1
중랑	164	137	2
강남	53	66	1
관악	219	102	2
강서	182	228	2
강동	185	177	2
종암	83	61	1
구로	163	185	2
서초	96	63	1
양천	138	162	2
송파	191	166	2
노원	218	112	2
방배	22	19	1
은평	126	134	2
도봉	106	59	1
수서	84	73	1

중범죄와 경범죄 - 5

- 군집화(fuzzy)
  - Soft clustering
  - 특정 클러스터에 속할 확률을 계산

```
fviz_cluster(fanny.result,data=x,palette=c("red","blue","green"),
             ellipse.type="euclid",star.plot=T,repel=T,ggtheme=theme(), labelsize=20)
```



	중범죄	경범죄	cluster
중부	20	47	1
종로	18	21	1
남대문	3	14	1
서대문	97	89	1
혜화	19	19	1
용산	42	21	1
성북	80	54	1
동대문	132	112	2
마포	116	127	2
영등포	156	144	2
성동	55	42	1
동작	123	72	1
광진	203	115	2
서부	45	93	1
강북	155	82	2
금천	81	65	1
종랑	164	137	2
강남	53	66	1
관악	219	102	2
강서	182	228	2
강동	185	177	2
종암	83	61	1
구로	163	185	2
서초	96	63	1
양천	138	162	2
송파	191	166	2
노원	218	112	2
방배	22	19	1
은평	126	134	2
도봉	106	59	1
수서	84	73	1

  - dunn's coeff
  - 모든 제곱 멤버 계수의 합을 관측치 수로 나눈 값
  - 1에 가까울 수록 선명한 클러스터링을 나타냄

```
Fuzzyness coefficients:
dunn_coeff normalized
 0.6799792  0.3599585
Closest hard clustering:
  중부   종로 남대문 서대문   혜화   용산   성북 동대문   마포 영등포   성동
     1      1      1      1      1      1      1      2      2      2      1
  동작   광진   서부   강북   금천   종랑   강남   관악   강서   강동   종암
     1      2      1      2      1      2      1      2      2      2      1
  구로   서초   양천   송파   노원   방배   은평   도봉   수서
     2      1      2      2      2      1      2      1      1
```

군집화 비교

- 군집화 비교
  - 데이터에 대해 계층적 군집화, 세 가지 프로토타입 기반 군집화를 진행하였습니다.
  - 계층적 군집화와, 프토로타입 기반 군집화에서는 같은 k값을 주어도 서로 다른 군집 결과가 나타났습니다.
  - 세 가지의 프로토타입 기반 군집화에서는 모두 동일한 군집화의 결과가 나타났습니다.

결론 및 고찰

- 결론 및 고찰
  - 공공 데이터를 통해서 군집화를 진행하였는데, 데이터 벡터들 마다 연관성이 높은 분류대로 나눌 수 있어 데이터 탐색에 용의하다고 생각
  - 전체 데이터에 대하여 최적화 된 군집 수를 정할 수 있는 plot 과 지표를 구할 수 있어 쉽게 군집화 가능
  - 비지도 학습기에 해당하는 군집분석은 모델의 예측 성능보다는 데이터 셋 자체의 특성을 파악하고 싶을 때 사용하는 분석이라고 판단
  - 군집 분석을 통한 중, 경범죄 정도의 수준을 지역별로 나눌 수 있어, 분석을 통한 범죄 예측 및 대응이 가능할 수 있을 것으로 보임