

## 다면량분석 과제 2

2015170378

정은영

### [K-Means Clustering]

K-Means clustering 은 각 변수의 값의 범위에 영향을 받을 수 있으므로 scale( )함수를 사용하여 모든 변 수의 평균을 0, 표준편차를 1로 만드는 정규화를 수행하시오.

```

1 # Multivariate Data Analysis Assignment #2
2 # 2015170378 Jung Eun-young
3
4 install.packages("ISLR")
5 install.packages("clValid")
6 install.packages("plotrix")
7
8 library(ISLR)
9 library(clValid)
10 library(plotrix)
11
12 data(College)
13 View(College)
14
15 c_Private <- College[,1]
16 cX <- College[,-1]
17 cX_scaled <- as.matrix(scale(cX, center = TRUE, scale = TRUE))
--
```

# [Q1-1] clValid()함수를 사용하여 K-Means Clustering 의 군집 수를 2개부터 10개까지 증가시켜 가면서 internal 및 stability 관련 타당성 지표 값을 산출하시오. 총 소요 시간은 얼마인가? Dunn index 와 Silhouette index 기준으로 가장 최적의 군집 수는 몇 개로 판별이 되었는가?

```

19 # -----[K-means Clustering]-----
20
21 # [Q1-1]
22 ## internal validation
23
24 start_time <- Sys.time()
25 cX_clValid <- clValid(cX_scaled, 2:10, clMethods="kmeans",
26                         validation=c("internal","stability"))
27 end_time <- Sys.time()
28
29 ## view results
30 end_time - start_time
31 summary(cX_clValid)
--
```

-> 결과값

```
> end_time - start_time
Time difference of 20.07694 secs
```

```
> summary(cX_clValid)

Clustering Methods:
kmeans

Cluster sizes:
2 3 4 5 6 7 8 9 10

Validation Measures:
          2      3      4      5      6      7      8      9      10
kmeans APN    0.1287  0.0436  0.1344  0.1815  0.2321  0.1748  0.1632  0.2372  0.3118
          AD     5.0040  4.3162  4.2363  4.1467  4.1325  3.9275  3.8215  3.8121  3.8180
          ADM    0.7046  0.1872  0.6012  0.8891  1.1200  0.6406  0.5745  0.8429  1.1216
          FOM    0.9627  0.8144  0.7938  0.7653  0.7755  0.7569  0.7439  0.7389  0.7334
          Connectivity 100.0270 194.8433 279.2139 258.9845 300.7508 283.7972 398.0944 434.5683 458.6734
          Dunn    0.0842  0.0611  0.0439  0.0481  0.0481  0.0577  0.0577  0.0679  0.0947
          Silhouette 0.3201  0.2421  0.1966  0.1883  0.1840  0.1908  0.1682  0.1500  0.1343

Optimal Scores:
          Score   Method Clusters
APN        0.0436 kmeans 3
AD         3.8121 kmeans 9
ADM        0.1872 kmeans 3
FOM        0.7334 kmeans 10
Connectivity 100.0270 kmeans 2
Dunn       0.0947 kmeans 10
Silhouette 0.3201 kmeans 2
```

Optimal Scores 를 보면 Dunn index 기준으로 Cluster 10 개, Silhouette 는 2 개로 판별이 되었다.

#[Q1-2] K=3 으로 군집화를 10 회 반복 수행하고 회차마다 각 군집의 Center 와 Size 를 확인해보시오. 10 회 반복 수행 시 모두 동일한 군집이 생성되었는가?

```
33 ##[Q1-2]
34 #n은 반복수, k는 군집수
35 Kmeans_Cluster_Info <- function(k,n){
36   for(i in 1:n){
37     cX_kmc <- kmeans(cX_scaled,k)
38     print(paste("n=",i))
39     print(cX_kmc$centers)
40     print(cX_kmc$size)
41   }
42 }
43 # Perform K-Means Clustering with the best K determined by Silhouette
44 Kmeans_Cluster_Info(3,10)
```

Kmeans Clustering 함수를 만들어 군집의 Center 와 Size 를 확인한다.

Kmeans\_Cluster\_Info(k,n) -> k 는 군집수, n 은 반복수로 설정했다.

```
[1] "n= 6"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
2 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
3 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
[1] 438 93 246
[1] "n= 7"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
3 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
[1] 438 93 246
[1] "n= 8"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
2 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
3 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
[1] 438 93 246
[1] "n= 9"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
2 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
3 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
[1] 438 93 246
[1] "n= 10"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
2 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
3 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
[1] 438 93 246
[1] "n= 1"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
3 -0.183946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
[1] 246 438 93
[1] "n= 2"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
3 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
[1] 246 438 93
[1] "n= 3"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 1.83946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
3 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
[1] 246 438 93
[1] "n= 4"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
3 -0.183946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
[1] 246 438 93
[1] "n= 5"
   Apps    Accept    Enroll Top10perc Top25perc F.Undergrad P.Undergrad  Outstate Room.Board    Books    Personal    PhD    Terminal    S.F.Ratio perc.alumni    Expend    Grad.Rate
1 -0.03489665 -0.1165637 -0.2330693 0.8552988 0.8370369 -0.3018808 -0.3693397 1.0480076 0.7172261 0.0590233 -0.37425038 0.7370637 0.7356034 -0.6350344 0.8164997 0.80516666 0.7674493
2 -0.37097263 -0.3607812 -0.3367498 -0.5291736 -0.5548376 -0.3116495 -0.1277753 -0.4746087 -0.3668251 -0.1024233 0.03786354 -0.5604491 -0.5571596 0.2325641 -0.3323420 -0.43979658 -0.3538001
3 -0.183946866 2.0074927 2.2024888 0.2298334 0.3990083 2.2662920 1.5787435 -0.5368952 -0.1695508 0.3262545 0.81162753 0.6898820 0.6782520 0.5844663 -0.5945498 -0.05849565 -0.3637427
[1] 246 438 93

```

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	개수
1	-0.0348967	-0.1165637	-0.2330693	0.8552988	0.8370369	-0.3018808	-0.3693397	1.0480076	0.7172261	0.0590233	-0.37425038	0.7370637	0.7356034	-0.6350344	0.8164997	0.80516666	0.7674493	246
2	-0.3709726	-0.3607812	-0.3367498	-0.5291736	-0.5548376	-0.3116495	-0.1277753	-0.4746087	-0.3668251	-0.1024233	0.03786354	-0.5604491	-0.5571596	0.2325641	-0.3323420	-0.43979658	-0.3538001	438
3	1.83946866	2.0074927	2.2024888	0.2298334	0.3990083	2.266292	1.5787435	-0.5368952	-0.1695508	0.3262545	0.81162753	0.6898820	0.6782520	0.5844663	-0.5945498	-0.05849565	-0.3637427	93
	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate									
1	0.7172261	0.0590233	-0.3742504	0.7370637	0.7356034	-0.6350344	0.8164997	0.80516666	0.7674493									
2	-0.3668251	-0.1024233	0.03786354	-0.5604491	-0.5571596	0.2325641	-0.3323424	-0.4397966	-0.3538001									
3	-0.1695508	0.3262545	0.81162753	0.689882	0.678252	0.5844663	-0.5945498	-0.0584957	-0.3637427									

위와 같은 형태로 10회 모두 동일한 군집이 생성되었다.

[Q1-3] K=10 으로 군집화를 10회 반복 수행하고 회차마다 각 군집의 Center 와 Size 를 확인해보시오. 10회 반복 시 모두 동일한 군집이 생성되었는가?

46 #Q1-3]

47 Kmeans\_Cluster\_Info(10,10)

위 문제와 같은 함수를 적용한다.

10회 반복 시 모두 동일한 군집이 생성되지 않았다.

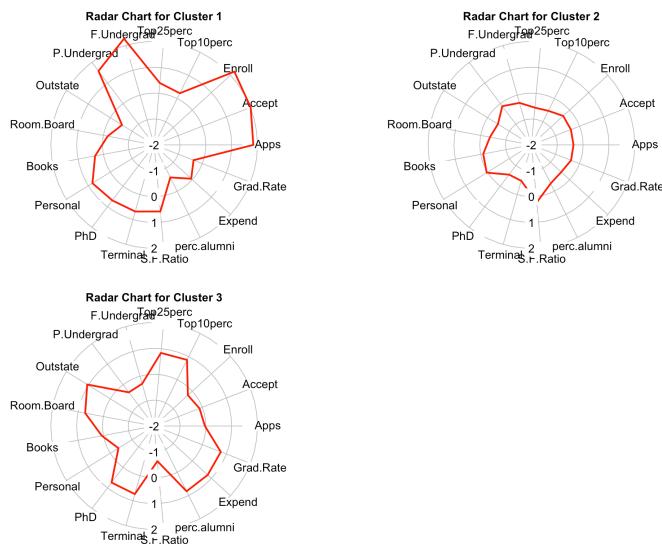
[1] "n= 1"																	
Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	
1	-0.5579585	0.5425601	-0.5572306	-0.52956904	-0.5993240	-0.5190916	-0.2818520	0.2383798	0.3400934	-0.3254858	-0.2200246	-1.44094505	1.15047745	-0.2700426	-0.1505230	0.406759001	-0.09672032
2	0.8841270	0.2898371	0.1777568	0.22591790	1.7367692	0.01122088	-0.3791958	1.91853780	1.45307202	0.3657724	-0.3759381	1.18981526	1.14083211	-1.34077395	1.3847654	2.36713932	1.33648387
3	-0.2665151	-0.2368425	-0.3514070	0.55311958	0.6550482	-0.3898923	-0.3795969	0.8913593	0.53230433	-0.1232996	-0.4803642	0.73478377	0.72340767	-0.45362978	0.6943944	0.43887252	0.61244359
4	-0.5328275	-0.5388125	-0.4962672	-0.9274469	-0.9272044	-0.48936442	-0.2731670	-0.93477995	-0.88343129	-0.0141455	0.1945290	-0.81583141	-0.90641449	0.53880565	-0.7487190	-0.20781009	
5	-0.3728963	-0.3217776	-0.3185237	-0.41817392	-0.4074462	-0.31418046	-0.2257921	-0.0692434	0.20559347	4.3618481	1.8656465	-1.24304281	-0.31942591	-0.26610597	0.6469289	-0.05642529	-0.19103258
6	-0.4257578	-0.4077041	-0.4377852	-0.2415036	-0.2145792	-0.44501037	-0.2556285	0.03155933	0.14394675	-0.1470667	-0.1504861	-0.09933102	-0.0263391	-0.06961582	0.1807274	-0.24083234	0.23772906
7	1.3408822	1.5763061	0.7291886	0.05714775	0.2184218	1.75515411	0.7317473	-0.540813076	-0.21384804	0.3984263	0.8042901	0.60950818	0.57051580	0.58107354	0.5884536	-0.14437750	0.24795149
8	3.4509076	3.6828480	3.6468906	0.90369116	1.0842120	3.63652378	1.4231845	-0.38359468	0.07842834	0.3389666	0.4599732	0.98177880	0.85618005	0.56616159	-0.4231739	0.10513103	
9	0.6436298	0.6429272	0.1422261	-0.33697076	-0.3587188	0.47318241	1.5626469	-0.72989051	-0.39838828	0.51251524	0.2041719	0.57200391	0.56889070	0.07061701	0.67458088	-0.02878908	-0.37039475
10	0.1906410	0.2169157	0.3793516	-0.62447720	-0.4097901	0.43628968	0.4479307	-0.13285182	-0.68766951	-0.2320124	0.3202565	0.17782770	0.18422258	1.09590019	-0.7914682	-0.65394225	-0.70635337
[1] 106 57 152 88 11 181 49 26 13 94																	
[1] "n= 2"																	
Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	
1	-0.0338293	-0.13767838	-0.2874409	1.5114976	1.39661817	-0.3657286	-0.49405362	1.73771800	0.0521926	-0.6699814	1.10555253	0.17623345	-0.6742882	0.7105739	-0.6196488	-0.81918918	
2	1.8462382	0.7253886	0.2867531	1.867531	1.93861487	0.43620459	-0.2790918	1.72872855	1.60368341	0.1014514	1.31859631	1.17813435	-0.1929552	0.33525115	3.9614421	0.24653386	
3	0.27291875	0.2915438	0.1820283	0.4371385	0.62018931	0.04626612	0.02037456	0.75150426	0.61347889	0.48833735	0.66824434	0.14572404	-0.35218283	0.1685655	0.39939736		
4	-0.5428293	-0.53591426	-0.5368117	0.6993760	-0.90596356	-0.50765430	-0.29830993	0.6304556	0.11738563	0.1236603	-0.14578121	-0.4657809	0.08253174	-0.5106961	-0.8739524		
5	-0.0748893	-0.1164794	-0.2046668	-0.35287045	-0.32862871	-0.0072947	-0.26667380	0.18373878	0.0809780	0.1805107	0.53276260	-0.4833373	-0.1708165	0.1744854	0.03870645	-0.3119127	0.21890905
6	3.4110019	3.6443475	3.6479462	0.9068405	0.17027484	3.574294	1.38443683	0.3606833	0.14727016	0.33776732	0.4909426	0.52022592	-0.44559014	0.2511664	0.1067066		
7	-0.5026296	-0.4931431	-0.3585714	0.3524744	0.3217768	0.4495835	0.47217015	0.18071805	0.5821652	0.2233309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
8	-0.4027638	-0.37267786	-0.4049942	0.2459428	0.3217105	-0.47499476	-0.40749476	0.1878105	0.6458535	0.23223309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
9	0.0145825	0.03201891	0.1541641	-0.70210572	0.00112108	0.2389397	1.8214800	-0.05264947	0.23635304	0.1233309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
10	1.0168276	1.2629369	1.2606287	-0.25694947	0.07445779	0.7827513	1.9024886	-0.5793825	0.43814261	0.33055032	0.03120191	0.06846827	0.5593825	-0.67279218	0.23988080	0.73373979	
[1] 59 21 62 122 10 27 170 137 109 82																	
[1] "n= 3"																	
Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	
1	-0.0338293	-0.13767838	-0.2874409	1.5114976	1.39661817	-0.3657286	-0.49405362	1.73771800	0.0521926	-0.6699814	1.10555253	0.17623345	-0.6742882	0.7105739	-0.6196488	-0.81918918	
2	1.8462382	0.7253886	0.2867531	1.867531	1.93861487	0.43620459	-0.2790918	1.72872855	1.60368341	0.1014514	1.31859631	1.17813435	-0.1929552	0.33525115	3.9614421	0.24653386	
3	0.27291875	0.2915438	0.1820283	0.4371385	0.62018931	0.04626612	0.02037456	0.75150426	0.61347889	0.48833735	0.66824434	0.14572404	-0.35218283	0.1685655	0.39939736		
4	-0.5428293	-0.53591426	-0.5368117	0.6993760	-0.90596356	-0.50765430	-0.29830993	0.6304556	0.11738563	0.1236603	-0.14578121	-0.4657809	0.08253174	-0.5106961	-0.8739524		
5	-0.0748893	-0.1164794	-0.2046668	-0.35287045	-0.32862871	-0.0072947	-0.26667380	0.18373878	0.0809780	0.1805107	0.53276260	-0.4833373	-0.1708165	0.1744854	0.03870645	-0.3119127	0.21890905
6	3.4110019	3.6443475	3.6479462	0.9068405	0.17027484	3.574294	1.38443683	0.3606833	0.14727016	0.33776732	0.4909426	0.52022592	-0.44559014	0.2511664	0.1067066		
7	-0.5026296	-0.4931431	-0.3585714	0.3524744	0.3217768	0.4495835	0.47217015	0.18071805	0.6458535	0.23223309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
8	-0.4027638	-0.37267786	-0.4049942	0.2459428	0.3217105	-0.47499476	-0.40749476	0.1878105	0.6458535	0.23223309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
9	0.0145825	0.03201891	0.1541641	-0.70210572	0.00112108	0.2389397	1.8214800	-0.05264947	0.23635304	0.1233309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
10	1.0168276	1.2629369	1.2606287	-0.25694947	0.07445779	0.7827513	1.9024886	-0.5793825	0.43814261	0.33055032	0.03120191	0.06846827	0.5593825	-0.67279218	0.23988080	0.73373979	
[1] 59 21 62 122 10 27 170 137 109 82																	
[1] "n= 4"																	
Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	
1	3.3271678	3.5456739	3.4989013	0.1856893	0.7371143	3.47233801	1.976293	-0.2733101	0.2142746	0.35942616	0.15056337	0.17233245	-0.30359598	0.4573361	0.39117275	0.27587454	0.01919899
2	-0.0748893	-0.1164794	-0.2046668	-0.35287045	-0.32862871	-0.0072947	-0.26667380	0.18373878	0.0809780	0.1805107	0.53276262	-0.20348322	-0.16090351	0.38019026	0.4535492	0.15334942	
3	0.27291875	0.2915438	0.1820283	0.4371385	0.62018931	0.04626612	0.02037456	0.75150426	0.61347889	0.48833735	0.66824434	0.14572404	-0.35218283	0.1685655	0.39939736		
4	-0.5428293	-0.53591426	-0.5368117	0.6993760	-0.90596356	-0.50765430	-0.29830993	0.6304556	0.11738563	0.1236603	-0.14578121	-0.4657809	0.08253174	-0.5106961	-0.8739524		
5	-0.0748893	-0.1164794	-0.2046668	-0.35287045	-0.32862871	-0.0072947	-0.26667380	0.18373878	0.0809780	0.1805107	0.53276262	-0.20348322	-0.16090351	0.38019026	0.4535492	0.15334942	
6	3.4110019	3.6443475	3.6479462	0.9068405	0.17027484	3.574294	1.38443683	0.3606833	0.14727016	0.33776732	0.4909426	0.52022592	-0.44559014	0.2511664	0.1067066		
7	-0.5026296	-0.4931431	-0.3585714	0.3524744	0.3217768	0.4495835	0.47217015	0.18071805	0.6458535	0.23223309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
8	-0.4027638	-0.37267786	-0.4049942	0.2459428	0.3217105	-0.47499476	-0.40749476	0.1878105	0.6458535	0.23223309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
9	0.0145825	0.03201891	0.1541641	-0.70210572	0.00112108	0.2389397	1.8214800	-0.05264947	0.23635304	0.1233309	0.03148428	0.04536216	0.15241784	-0.77171332	-0.6639911	-0.7121999	
10	1.0168276	1.2629369	1.2606287	-0.25694947	0.07445779	0.7827513	1.9024886	-0.5793825	0.43814261	0.33055032	0.03120191	0.06846827	0.5593825	-0.67279218	0.23988080	0.73373979	
[1] 29 79 21 117 88 100 28 133 55 77																	
[1] "n= 5"																	
Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.Ratio	perc.alumni	Expend	Grad.Rate	
1	-0.20625605	-0.2236051	-0.3615423	-0.5596133	-0.1919646	-0.4158323	-0.3405613	0.0526720	-0.1745970	-0.1745970	0.1745970	-0.1745970	-0.1432909	0.1316595	0.5671902		
2	3.4771352	3.5563767	3.4797343	0.1855994													

#[Q1-4] K=3 으로 군집화를 수행한 뒤, 각 변수들에 대해 정규화 이후의 값들을 이용하여 Radar chart 를 도시하시오. 이 중에서 상대적으로 더 유사한 두 개의 군집 쌍과 가장 다른 두 개의 군집 쌍을 판별해 보시오.

```

49  #[Q1-4]
50  cX_kmc <- kmeans(cX_scaled, 3)
51  # Compare the cluster info. and class labels
52  real_private <- c_Private
53  kmc_cluster <- cX_kmc$cluster
54  table(real_private, kmc_cluster)
55
56  # Compare each cluster for KMC
57  cluster_kmc <- data.frame(cX_scaled, clusterID = as.factor(cX_kmc$cluster))
58  kmc_summary <- data.frame()
59
60  for (i in 1:(ncol(cluster_kmc)-1)){
61    kmc_summary = rbind(kmc_summary,
62      tapply(cluster_kmc[,i], cluster_kmc$clusterID, mean))
63  }
64
65  colnames(kmc_summary) <- paste("cluster", c(1:3))
66  rownames(kmc_summary) <- colnames(cX)
67
68  par(mfrow = c(2,2))
69  for (i in 1:3){
70    plot_title <- paste("Radar Chart for Cluster", i, sep=" ")
71    radial.plot(kmc_summary[,i], labels = rownames(kmc_summary),
72      radial.lim=c(-2,2), rp.type = "p", main = plot_title,
73      line.col = "red", lwd = 2.5, show.grid.labels=1)
74  }
75  dev.off()

```



1,3 은 매우 상이하게 보인다.(top10perc) cluster2,3 0| 그림상 1 보다는 유사해 보인다.

#[Q1-5] 세 개의 두 군집 조합(Cluster 1 vs. Cluster 2, Cluster 1 vs. Cluster 3, Cluster 2 vs. Cluster 3)에 대 해서 각 변수별 차이의 유의성에 대해서 t-test를 수행하고 그 결과를 해석하시오.

```

77 #[Q1-5]
78 #Cluster_tTest(method,ClusterIndex(i),ClusterIndex(j))
79 Cluster_tTest <- function(m,n1,n2){
80
81   if(m=="kmc"){
82     cluster1 <- cX[cX_kmc$cluster == n1,]
83     cluster2 <- cX[cX_kmc$cluster == n2,]
84     t_result <- data.frame()
85   }else if(m=="hc"){
86     cluster1 <- cX[cluster_hc$cluster == n1,]
87     cluster2 <- cX[cluster_hc$cluster == n2,]
88     t_result <- data.frame()
89   }
90
91   for (i in 1:ncol(cX)) {
92
93     t_result[i,1] <- t.test(cluster1[,i], cluster2[,i],
94                             alternative = "two.sided")$p.value
95
96     t_result[i,2] <- t.test(cluster1[,i], cluster2[,i],
97                             alternative = "greater")$p.value
98
99     t_result[i,3] <- t.test(cluster1[,i], cluster2[,i],
100                            alternative = "less")$p.value
101   }
102   print(t_result)
103 }

105 #Cluster_tTest(method,ClusterIndex(i),ClusterIndex(j))
106 # Cluster 1 vs. Cluster 2
107 Cluster_tTest("kmc",1,2)
108
109 # Cluster 1 vs. Cluster 3
110 Cluster_tTest("kmc",1,3)
111
112 #Cluster 2 vs. Cluster 3
113 Cluster_tTest("kmc",2,3)

```

```

> # Cluster 1 vs. Cluster 2          > # Cluster 1 vs. Cluster 3
> Cluster_tTest("kmc",1,2)          > Cluster_tTest("kmc",1,3)
      V1        V2        V3      V1        V2        V3
1 3.058227e-24 1.529113e-24 1.0000000000 1 6.304911e-20 3.152455e-20 1.000000e+00
2 8.506840e-27 4.253420e-27 1.0000000000 2 8.170398e-24 4.085199e-24 1.000000e+00
3 1.835458e-35 9.177291e-36 1.0000000000 3 2.503837e-34 1.251919e-34 1.000000e+00
4 2.272173e-10 1.136087e-10 0.9999999999 4 1.137471e-06 9.999994e-01 5.687356e-07
5 4.337600e-16 2.168800e-16 1.0000000000 5 6.460040e-05 9.999677e-01 3.230020e-05
6 2.236426e-38 1.118213e-38 1.0000000000 6 2.834424e-38 1.417212e-38 1.000000e+00
7 1.399637e-12 6.998186e-13 1.0000000000 7 4.729789e-15 2.364895e-15 1.000000e+00
8 4.480168e-01 7.759916e-01 0.2240083907 8 1.769046e-40 1.000000e+00 8.845230e-41
9 6.579076e-02 3.289538e-02 0.9671046214 9 8.032827e-13 1.000000e+00 4.016413e-13
10 4.343236e-06 2.171618e-06 0.9999978284 10 1.001898e-02 5.009490e-03 9.949905e-01
11 4.085254e-10 2.042627e-10 0.9999999998 11 3.552026e-18 1.776013e-18 1.000000e+00
12 1.758412e-60 8.792060e-61 1.0000000000 12 3.961200e-01 8.019400e-01 1.980600e-01
13 6.774873e-57 3.387436e-57 1.0000000000 13 2.860576e-01 8.569712e-01 1.430288e-01
14 2.548733e-03 1.274366e-03 0.9987256337 14 2.177290e-19 1.088645e-19 1.000000e+00
15 1.159043e-03 9.994205e-01 0.0005795213 15 2.701272e-37 1.000000e+00 1.350636e-37
16 2.541693e-07 1.270846e-07 0.9999998729 16 2.225925e-14 1.000000e+00 1.112963e-14
17 9.221614e-01 5.389193e-01 0.4610807226 17 7.964679e-21 1.000000e+00 3.982339e-21

> #Cluster 2 vs. Cluster 3
> Cluster_tTest("kmc",2,3)
      V1        V2        V3
1 8.982960e-12 1.000000e+00 4.491480e-12
2 1.686494e-10 1.000000e+00 8.432471e-11
3 2.082504e-03 9.989587e-01 1.041252e-03
4 4.705503e-57 1.000000e+00 2.352752e-57
5 7.271735e-82 1.000000e+00 3.635867e-82
6 7.440255e-01 6.279872e-01 3.720128e-01
7 8.973497e-15 4.486749e-15 1.000000e+00
8 1.023578e-92 1.000000e+00 5.117892e-93
9 1.833853e-44 1.000000e+00 9.169264e-45
10 5.283634e-02 9.735818e-01 2.641817e-02
11 2.231088e-08 1.115544e-08 1.000000e+00
12 3.644528e-86 1.000000e+00 1.822264e-86
13 2.591698e-90 1.000000e+00 1.295849e-90
14 5.700492e-36 2.850246e-36 1.000000e+00
15 5.940226e-47 1.000000e+00 2.970113e-47
16 3.779045e-35 1.000000e+00 1.889523e-35
17 2.269348e-57 1.000000e+00 1.134674e-57

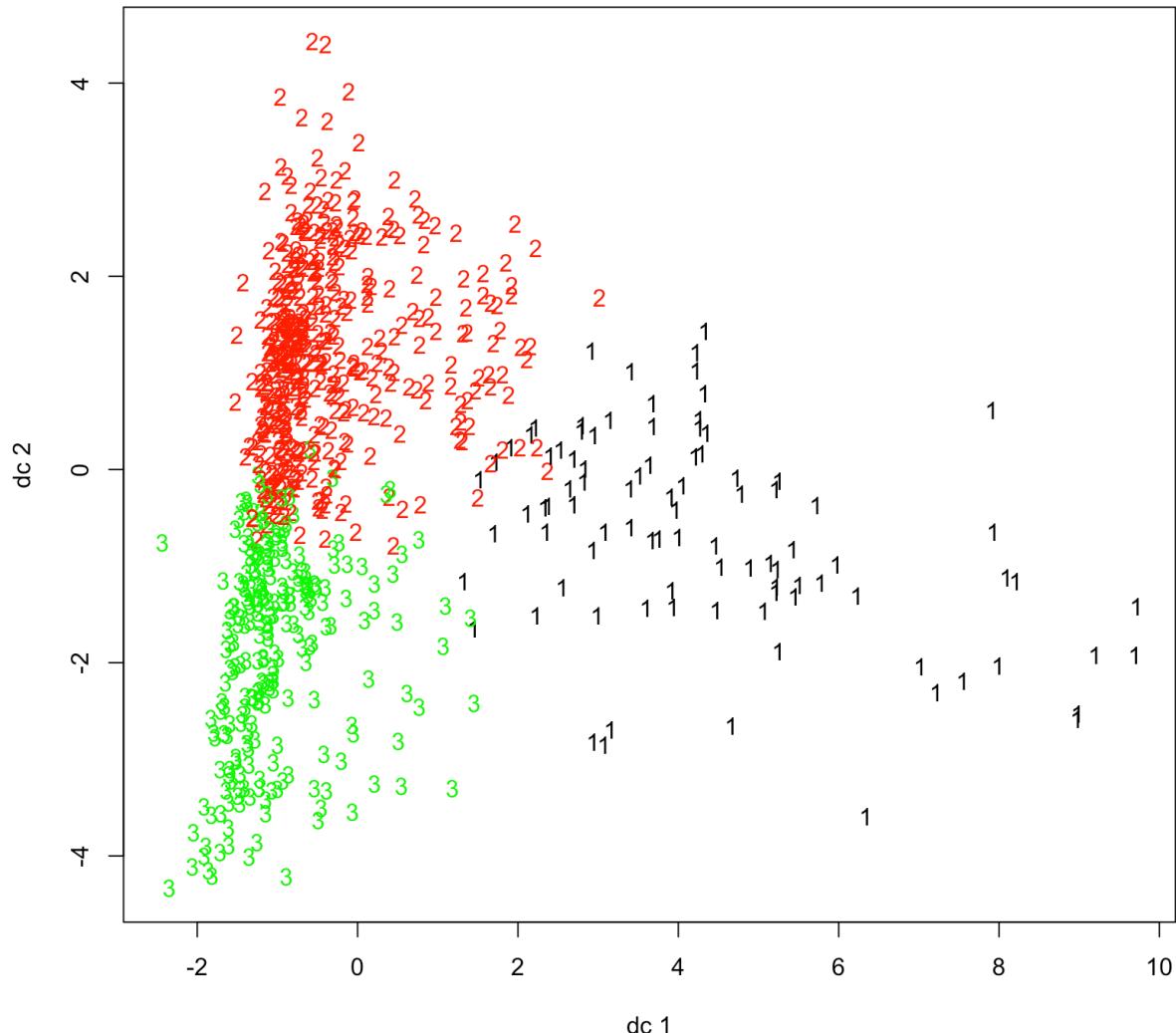
```

p-value 가 0.01 보다 작은 것은 통계적으로 유의미하다. one-sided 만 가지고 1,2 를 비교했을 때 8,9 번째 변수의 p-value 0.01 을 넘고 나머지는 0.01 보다 작아 통계적으로 유의미함을 보였다.

1,3 비교시 12,13 번 변수, 2,3 비교시 6 번 변수를 제외하고 통계적으로 유의미함을 확인할 수 있다.

#[Q1-6] K-Means Clustering 의 결과물을 시각화할 수 있는 방법을 웹에서 찾아 스스로 적용해보시오.

```
116 ##[Q1-6]  
117 install.packages("fpc")  
118 library(fpc)  
119 plotcluster(cX_scaled, cX_kmc$cluster, color=TRUE, shade=TRUE)
```



### [Hierarchical Clustering]

College Dataset에 대해 scale()함수를 사용하여 모든 변수의 평균을 0, 표준편차를 1로 만드는 정규화를 수행하시오.

kmeans 시 했던 데이터를 그대로 이용한다.

[Q2-1] clValid() 함수를 사용하여 Hierarchical Clustering의 군집 수를 2개부터 10개까지 증가시켜 가면서 internal 및 stability 관련 타당성 지표 값들을 산출하시오. 총 소요 시간은 얼마인가? K-Means Clustering과 비교할 경우 소요 시간의 차이가 있는가? Dunn index와 Silhouette index 기준으로 가장 최적의 군집 수는 몇 개로 판별이 되었는가?

```

123 # [Q2-1]
124
125 ## internal validation
126 start_time <- Sys.time()
127 cX_clValid <- clValid(cX_scaled, 2:10, clMethods="hierarchical",
128                         validation=c("internal","stability"))
129 end_time <- Sys.time()
130
131 ## view results
132 end_time - start_time
133 summary(cX_clValid)
134 plot(cX_clValid)

> end_time - start_time
Time difference of 14.73713 secs kmeans 랑 비교하여 조금 빠른 시간이 나왔다.

> summary(cX_clValid)

Clustering Methods:
hierarchical

Cluster sizes:
2 3 4 5 6 7 8 9 10

Validation Measures:
          2      3      4      5      6      7      8      9      10
hierarchical APN  0.0003  0.0003  0.0042  0.0090  0.0176  0.0210  0.0251  0.0377  0.0729
                  AD   5.3248  5.2914  5.2689  5.1870  5.1639  5.1406  5.1171  5.0968  5.0776
                  ADM  0.0075  0.0074  0.0425  0.1195  0.1569  0.1938  0.2021  0.2913  0.4527
                  FOM  0.9988  0.9944  0.9941  0.9812  0.9660  0.9658  0.9646  0.9575  0.9502
Connectivity    2.9290  5.8579  8.7869  22.0956  25.0246  31.3833  34.3123  46.5571  46.5571
Dunn           0.4033  0.4463  0.4393  0.1718  0.1718  0.1718  0.1718  0.1826  0.1826
Silhouette     0.6777  0.6464  0.5802  0.4806  0.4291  0.3481  0.3015  0.2422  0.2125

Optimal Scores:
          Score Method Clusters
APN      0.0003 hierarchical 3
AD       5.0776 hierarchical 10
ADM     0.0074 hierarchical 3
FOM      0.9502 hierarchical 10
Connectivity 2.9290 hierarchical 2
Dunn      0.4463 hierarchical 3
Silhouette 0.6777 hierarchical 2

```

Dunn index 기준으로 3개, Silhouette 기준으로 2개로 Optimal 군집개수가 나왔다.

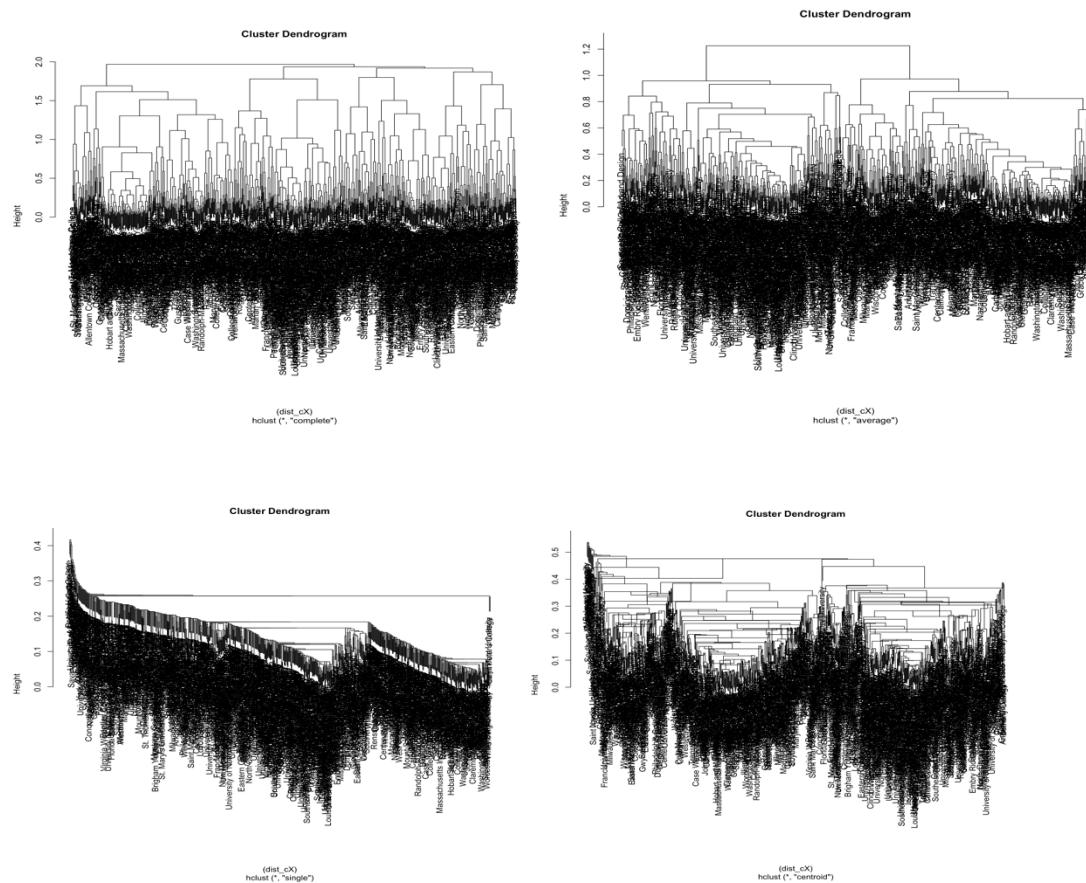
[Q2-2] Distance matrix 는 실습 자료에서 제공하는 spearman correlation 을 이용한 방식으로 산출한 뒤, hclust( ) 함수의 method 옵션을 다양하게 조절해 가면서 dendrogram 을 그려보시오. 군집화를 수행한다고 할 때 군집의 크기가 가장 극단적으로 차이가 날 것으로 예상되는 옵션은 무엇인가?

```

136  #[Q2-2]
137  ## Compute the similarity using the spearman coefficient
138
139 cor_Mat <- cor(t(cX_scaled), method = "spearman")
140 dist_cX <- as.dist(1-cor_Mat)
141
142 ## Perform hierarchical clustering
143 hcMethod <- c("complete", "average", "single", "mcquitty", "median", "centroid", "ward.D", "ward.D2")
144 for (item in hcMethod){
145   hr <- hclust((dist_cX), method = item, members=NULL)
146   plot(hr)
147 }
```

hcMethod <- c("complete", "average", "single", "mcquitty", "median", "centroid", "ward.D", "ward.D2")

를 이용하여 각각 방법에 대해 실행해보았다. 아래는 "complete", "average", "single" , "centroid" 이다.

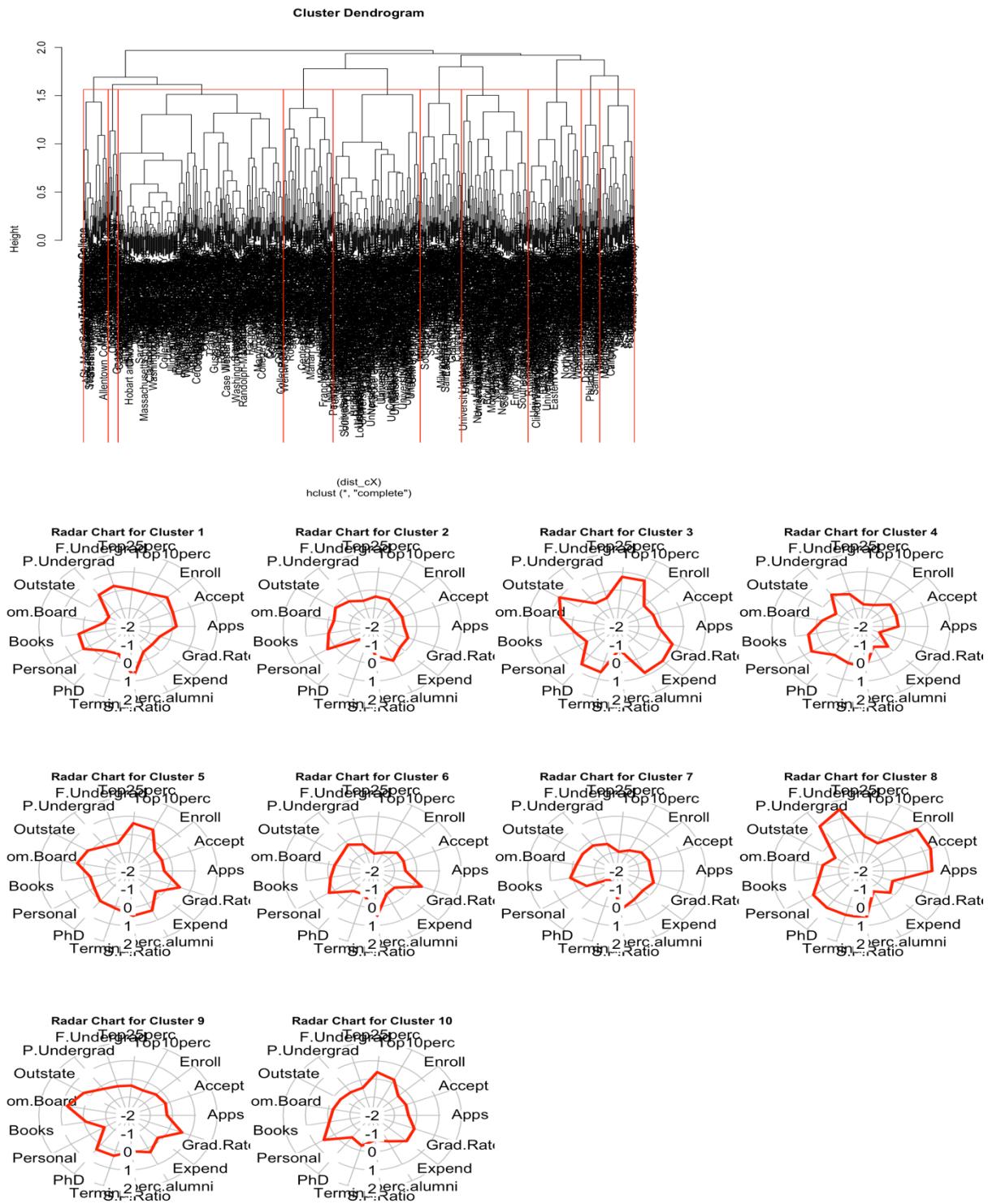


상대적으로 "single"이 가장 극단적으로 차이 날 것 같다.

[Q2-3] 생성된 Dendrogram 으로부터 10 개의 군집을 찾은 후 각 변수들에 대한 정규화 이후의 값들을 이용하여 Radar chart 를 도시하시오. 본인이 판단하기에 가장 차이가 극명하게 나타나는 두 군집 조합과 가장 유사한 것으로 보이는 두 군집 조합을 선택한 뒤 각 조합에 대해서 변수별 차이의 통계적 유의성을 t- test 를 통해 검증하시오.

```

150 #[Q2-3]
151 # Find the clusters
152 hr <- hclust((dist_cX), method = "complete", members=NULL)
153 mycl <- cutree(hr, k=10)
154 mycl
155
156 plot(hr)
157 rect.hclust(hr, k=10, border="red")
158
159
160 # Compare each cluster for HC
161 cluster_hc <- data.frame(cX_scaled, clusterID = as.factor(mycl))
162 hc_summary <- data.frame()
163
164 for (i in 1:(ncol(c_hc)-1)){
165   hc_summary = rbind(hc_summary,
166                       tapply(cluster_hc[,i], cluster_hc$clusterID, mean))
167 }
168
169 colnames(hc_summary) <- paste("cluster", c(1:10))
170 rownames(hc_summary) <- colnames(cX)
171 hc_summary
172
173 # Radar chart
174 par(mfrow = c(3,4))
175 for (i in 1:10){
176   plot_title <- paste("Radar Chart for Cluster", i, sep=" ")
177   radial.plot(hc_summary[,i], labels = rownames(hc_summary),
178               radial.lim=c(-2,2), rp.type = "p", main = plot_title,
179               line.col = "red", lwd = 3, show.grid.labels=1)
180 }
181 dev.off()
```



10 개의 Radar chart 를 도시하였을 때, 가장 유사한 것으로 보이는 두 군집은 cluster 6,7 이다.

차이가 극명하게 보이는 집단은 3,8 이다. 두개를 t-test 를 한 결과는 다음과 같다.

```

> # 차이가 가장 비슷한 집단 Cluster 6 vs. Cluster 7
> Cluster_tTest("hc", 6, 7)
      V1        V2        V3
1 4.922128e-01 7.538936e-01 0.246106410
2 4.373721e-01 7.813139e-01 0.218686054
3 7.094709e-01 6.452646e-01 0.354735443
4 5.347024e-02 9.732649e-01 0.026735122
5 3.464773e-01 8.267613e-01 0.173238655
6 5.351702e-01 7.324149e-01 0.267585109
7 5.441121e-01 2.720560e-01 0.727943973
8 2.053434e-01 8.973283e-01 0.102671715
9 5.288614e-01 7.355693e-01 0.264430721
10 4.208024e-01 7.895988e-01 0.210401201
11 3.024808e-03 1.512404e-03 0.998487596
12 7.422487e-05 3.711244e-05 0.999962888
13 1.949211e-03 9.746055e-04 0.999025395
14 1.192392e-01 5.961960e-02 0.940380400
15 3.777517e-02 9.811124e-01 0.018887586
16 3.120822e-03 9.984396e-01 0.001560411
17 1.730836e-03 8.654180e-04 0.999134582

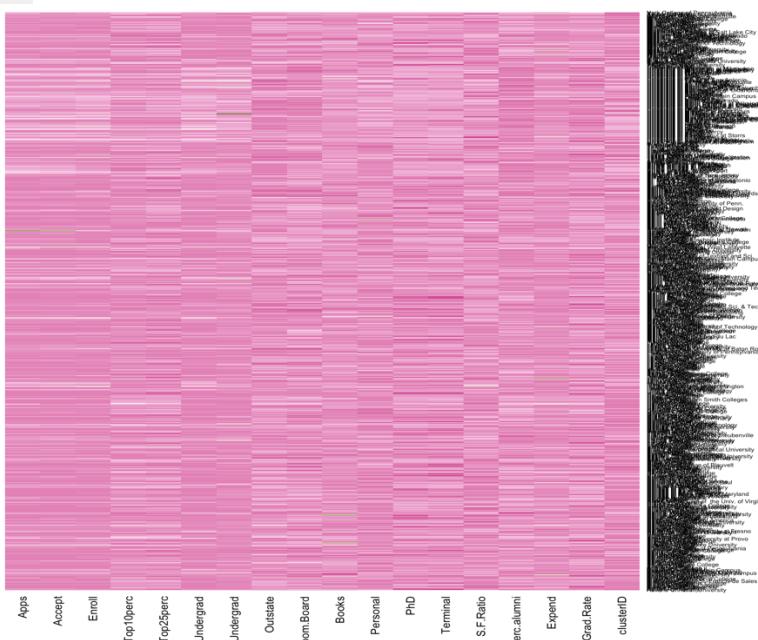
> # 차이가 가장 극명한 집단 Cluster 3 vs. Cluster 8
> Cluster_tTest("hc", 3, 8)
      V1        V2        V3
1 1.323863e-16 1.000000e+00 6.619313e-17
2 9.755304e-21 1.000000e+00 4.877652e-21
3 4.162327e-26 1.000000e+00 2.081164e-26
4 7.636000e-27 3.818000e-27 1.000000e+00
5 7.983715e-16 3.991858e-16 1.000000e+00
6 6.102748e-30 1.000000e+00 3.051374e-30
7 5.681459e-16 1.000000e+00 2.840730e-16
8 2.690637e-57 1.345319e-57 1.000000e+00
9 6.111057e-13 3.055528e-13 1.000000e+00
10 7.662318e-01 6.168841e-01 3.831159e-01
11 7.868836e-19 1.000000e+00 3.934418e-19
12 3.445097e-01 1.722548e-01 8.277452e-01
13 1.386780e-01 6.933899e-02 9.306610e-01
14 6.504037e-29 1.000000e+00 3.252019e-29
15 1.111755e-60 5.558776e-61 1.000000e+00
16 5.383138e-20 2.691569e-20 1.000000e+00
17 4.213407e-35 2.106703e-35 1.000000e+00

```

다음과 같이 비교한 결과 차이가 비슷한 집단의 p-value는 대부분 e-01 정도로 큰 값을 가졌으며, 즉 0.01 보다 대부분 큰 값을 가졌다. 즉 one-sided 로는 모두 0.01 보다 p-value 가 크기 때문에 비슷한 것을 확인할 수 있다. 차이가 극명한 집단은 10,12,13 변수를 제외하고 V1(one-sided)로 p-value 를 계산했을 때 모두 0.01 보다 작기 때문에 상이한 것을 확인할 수 있다.

[Q2-4] Hierarchical Clustering 의 결과물을 heatmap 을 사용하여 도시하는 방법을 웹에서 찾아 적용해보 시오.

```
191 # [Q2-4] |  
192 library(RColorBrewer)  
193 coul = colorRampPalette(brewer.pal(8, "PiYG"))(25)  
194 heatmap(data.matrix(cluster_hc), Rowv=NA, Colv=NA, col=coul, scale="column",  
margin=c(5,10))
```



## [DBSCAN]

실습에서 사용한 "Personal Loan.csv" 파일을 사용하여 다음 절차를 수행한 뒤 이후 물음에 답하시오. (1) 1 번, 5 번, 10 번 column 제외, (2) 평균 = 0, 표준편차 = 1로 정규화.

```

197 #[DBSCAN]
198 ploan <- read.csv("Personal Loan.csv")
199 ploan_x <- ploan[,-c(1,5,10)]
200 ploan_x_scaled <- scale(ploan_x, center = TRUE, scale = TRUE)
201
202 install.packages("factoextra")
203 install.packages("dbSCAN")
204
205 library(factoextra)
206 library(dbSCAN)
207 library(fpc)

```

[Q3-1] dbSCAN( )함수의 eps 옵션과 minPts 옵션에 대해 각각 최소 5 개 이상의 서로 다른 값(총 25 개 이 상의 조합)을 사용하여 생성된 군집의 수와 어느 군집에도 할당되지 않은 Noise points 수를 아래 표와 같이 정리하시오.

```

209 #[Q3-1]
210 # DBSCAN & Visualization
211 nMat <- matrix(c(1:100), nrow=25, ncol=4)
212 colnames(nMat) <- c("eps", "minPts", "군집 수", "Noise 수")
213
214 eps <- c(1.5, 2, 2.5, 3, 3.5)
215 minPts <- c(10, 11, 12, 13, 14)
216 for (i in 1:5){
217   for(j in 1:5){
218     nMat[(5*i-5)+j, 1] <- eps[i]
219     nMat[(5*i-5)+j, 2] <- minPts[j]
220     nMat[(5*i-5)+j, 3] <- length(unique(dbSCAN(ploan_x_scaled, eps = eps[i], MinPts = minPts[j])$cluster))-1
221     nMat[(5*i-5)+j, 4] <- length(which((dbSCAN(ploan_x_scaled, eps = eps[i], MinPts = minPts[j])$cluster)==0))
222   }
223 }
224 db_table<-as.data.frame(nMat)

```

위의 코드와 같이 for 문을 사용하여 코드를 작성하였다.

```
length(which((dbSCAN(ploan_x_scaled, eps = eps[i], MinPts = minPts[j])$cluster)==0))
```

이 부분은 cluster 의 0 Index 가 해당하는 부분이 노이즈이기 때문에 0 인 개수를 세는 방식으로 진행하였다.

```
length(unique(dbSCAN(ploan_x_scaled, eps = eps[i], MinPts = minPts[j])$cluster))-1
```

이 부분은 cluster 의 unique 즉, cluster index 의 개수를 샌 것인데, 0 은 노이즈기 때문에 제외했다.

그 결과는 다음과 같다.

	eps	minPts	군집 수	Noise 수
2	1.5	11	5	618
3	1.5	12	4	662
4	1.5	13	5	682
5	1.5	14	7	731
6	2.0	10	7	241
7	2.0	11	7	263
8	2.0	12	7	272
9	2.0	13	6	289
10	2.0	14	6	300
11	2.5	10	5	67
12	2.5	11	5	71
13	2.5	12	4	96
14	2.5	13	4	101
15	2.5	14	5	107
16	3.0	10	4	15
17	3.0	11	4	15
18	3.0	12	4	15
19	3.0	13	4	17
20	3.0	14	4	19
21	3.5	10	2	2
22	3.5	11	2	2
23	3.5	12	2	2
24	3.5	13	2	2
25	3.5	14	2	4

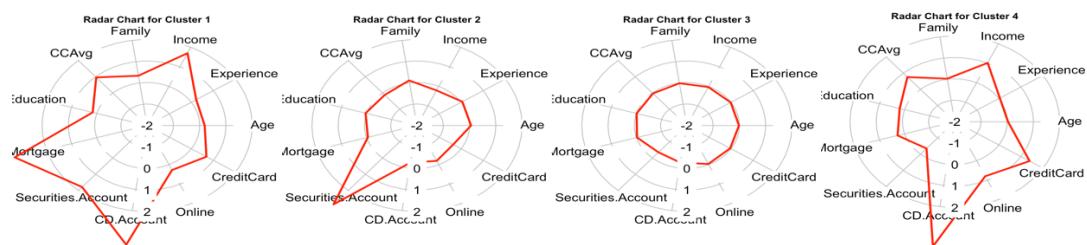
[Q3-2] 최소 3 개 이상의 군집이 판별된 eps/minPts 조합에 대하여 군집별 변수 Radar chart 를 도시하고 각 군집의 특성을 파악해 보시오.

```

226 ##[Q3-2]
227 DBSCAN_multishapes <- dbscan(ploan_x_scaled, eps = 3, MinPts = 10)
228
229 cluster_kmc <- data.frame(ploan_x_scaled, clusterID = as.factor(DBSCAN_multishapes$cluster))
230 kmc_summary <- data.frame()
231
232 for (i in 1:(ncol(cluster_kmc)-1)){
233   kmc_summary = rbind(kmc_summary,
234                         tapply(cluster_kmc[,i], cluster_kmc$clusterID, mean))
235 }
236 kmc_summary[,-1]
237
238 colnames(kmc_summary) <- paste("cluster", c(1:5))
239 rownames(kmc_summary) <- colnames(ploan_x)
240
241 par(mfrow = c(2,3))
242 for (i in 1:5){
243   plot_title <- paste("Radar Chart for Cluster", i, sep=" ")
244   radial.plot(kmc_summary[,i], labels = rownames(kmc_summary),
245                 radial.lim=c(-2,2), rp.type = "p", main = plot_title,
246                 line.col = "red", lwd = 2.5, show.grid.labels=1)
247 }
248 dev.off()

```

eps=3, MinPts = 10 으로 설정하여 군집이 4 개가 나오도록 설정하였다. noise 가 군집으로 분류됨으로 제외한다.



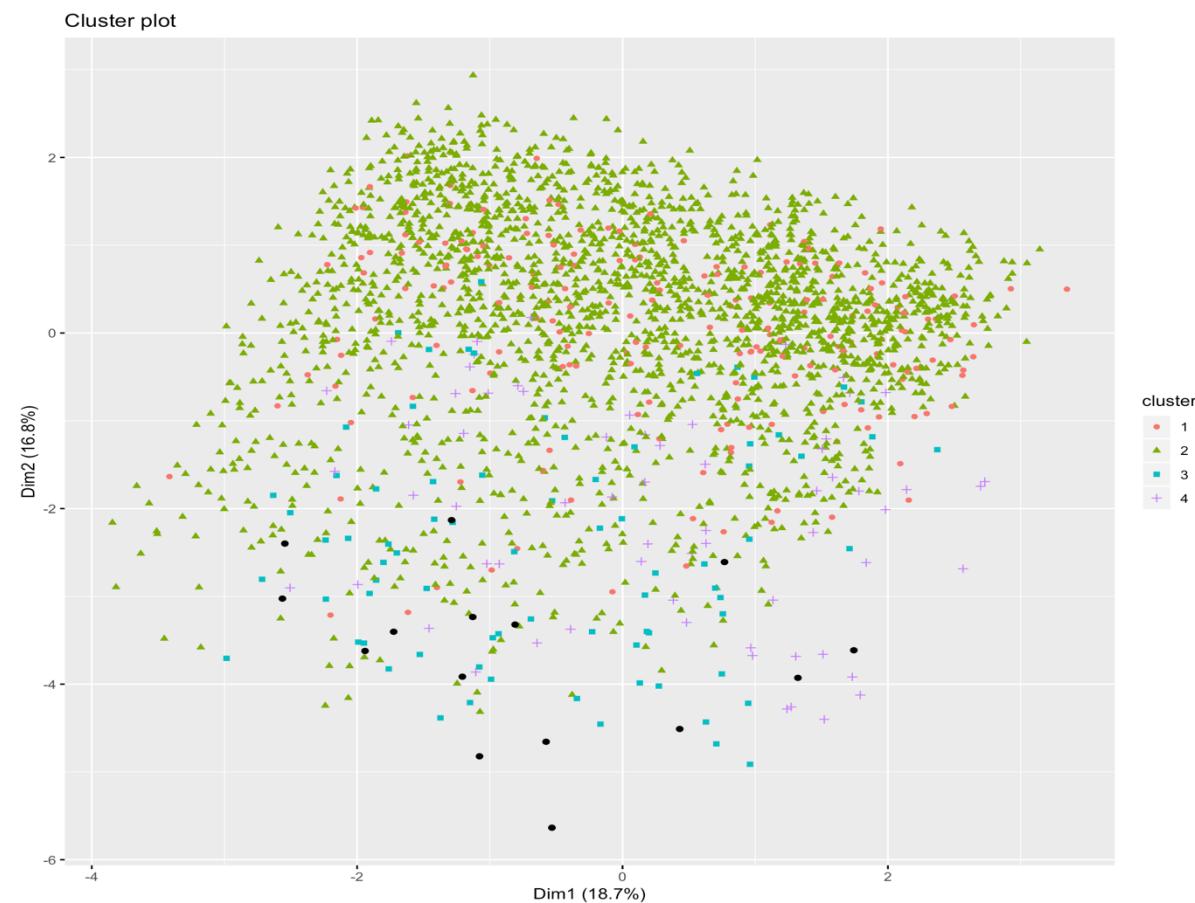
군집 2 의 경우 Securities Account 가 높게 나왔다. 각 군집의 튀어나온 부분으로 변수의 차이를 볼 수 있다. 즉 4 번째 군집은 CreditCard 가 loan ratio 에 대해 영향을 줄 수 있다.

[Q3-3] 원래 데이터를 PCA 를 이용하여 2 차원으로 축소시킨 후 [Q3-2]에서 선택한 군집의 수를 이용하여 군집들과 Noise points 를 2 차원 평면에 도시해 보시오.

```

250
251 # [Q3-3]
252 fviz_cluster(DBSCAN_multishapes, ploan_x_scaled, ellipse = FALSE, geom = "point",
253   show.clust.cent = FALSE)
254

```



검은색이 noise points 이다.