동국대학교
dongguk university




실습 4강
계층적 군집분석

실습 데이터

- 실습데이터와 실습과정은 Shmueli et al., Data Mining for Business Analytics, R Edition”에서 발췌
- 미국 전력회사 데이터 22개
- 9변수 (독립변수 8)

	A	B	C	D	E	F	G	H	I
1	Company	Fixed_char	RoR	Cost	Load_facto	Demand_g	Sales	Nuclear	Fuel_Cost
2	Arizona	1.06	9.2	151	54.4	1.6	9077	0	0.628
3	Boston	0.89	10.3	202	57.9	2.2	5088	25.3	1.555
4	Central	1.43	15.4	113	53	3.4	9212	0	1.058
5	Commonv	1.02	11.2	168	56	0.3	6423	34.3	0.7
6	NY	1.49	8.8	192	51.2	1	3300	15.6	2.044
7	Florida	1.32	13.5	111	60	-2.2	11127	22.5	1.241
8	Hawaiian	1.22	12.2	175	67.6	2.2	7642	0	1.652
9	Idaho	1.1	9.2	245	57	3.3	13082	0	0.309
10	Kentucky	1.34	13	168	60.4	7.2	8406	0	0.862
11	Madison	1.12	12.4	197	53	2.7	6455	39.2	0.623
12	Nevada	0.75	7.5	173	51.5	6.5	17441	0	0.768
13	New Engla	1.13	10.9	178	62	3.7	6154	0	1.897
14	Northern	1.15	12.7	199	53.7	6.4	7179	50.2	0.527
15	Oklahoma	1.09	12	96	49.8	1.4	9673	0	0.588
16	Pacific	0.96	7.6	164	62.2	-0.1	6468	0.9	1.4
17	Puget	1.16	9.9	252	56	9.2	15991	0	0.62
18	San Diego	0.76	6.4	136	61.9	9	5714	8.3	1.92
19	Southern	1.05	12.6	150	56.7	2.7	10140	0	1.108
20	Texas	1.16	11.7	104	54	-2.1	13507	0	0.636
21	Wisconsin	1.2	11.8	148	59.9	3.5	7287	41.1	0.702
22	United	1.04	8.6	204	61	3.5	6650	0	2.116
23	Virginia	1.07	9.3	174	54.3	5.9	10093	26.6	1.306

데이터 구조 확인과 전처리

```
# 데이터 읽기
utilities.df <- read.csv("Utilities.csv")

# Company 벡터 내용 복제하기
row.names(utilities.df) <- utilities.df[,1]

# 기존의 Company 벡터 삭제
utilities.df <- utilities.df[,-1]
```

	Company
1	Arizona
2	Boston
3	Central
4	Commonwealth
5	NY

	Company
Arizona	Arizona
Boston	Boston
Central	Central
ommonwealth	Commonwealth
NY	NY

	Fixed_charge
Arizona	1.06
Boston	0.89
Central	1.43
ommonwealth	1.02
NY	1.49

데이터 구조 확인과 전처리

```
# compute Euclidean distance
d <- dist(utilities.df, method = "euclidean")
```

```
> d
                Arizona      Boston     Central  Commonwealth          NY     Florida
Boston       3989.40808
Central       140.40286  4125.04413
Commonwealth 2654.27763  1335.46650  2789.75967
NY           5777.16767  1788.06803  5912.55291   3123.15322
Florida      2050.52944  6039.68908  1915.15515   4704.36310  7827.42921
Hawaiian     1435.26502  2554.28716  1571.29540   1219.56001  4342.09380  3485.67156
Idaho        4006.10419  7994.15599  3872.25763   6659.53457  9782.15818  1959.73108
Kentucky      671.27635  3318.27656   807.92079   1983.31435  5106.09415  2721.70630
Madison      2622.69900  1367.09063  2758.55966     43.64889  3155.09559  4672.82929
Nevada       8364.03105 12353.06270  8229.22328  11018.05781 14141.02258  6314.35909
New England  2923.13610  1066.57943  3058.70743    271.45273  2854.09948  4973.50684
Northern     1899.27982  2091.16049  2035.44152    756.83195  3879.16746  3949.09232
Oklahoma      598.55663  4586.30256   461.34167   3250.98459  6373.74325  1454.29260
Pacific      2609.04536  1380.74996  2744.50285     56.64463  3168.17746  4659.35626
Puget        6914.74206 10903.14646  6780.43031   9568.43443 12691.15511  4866.11165
San Diego    3363.06163   629.76075  3498.11301    710.29296  2414.69876  5413.09300
Southern     1063.00907  5052.33167   928.74925   3717.20296  6840.15029   988.04456
Texas        4430.25159  8419.61054  4295.01469   7084.37284 10207.39263  2380.12497
Wisconsin    1790.48565  2199.72167  1925.77256    864.27315  3987.33596  3840.22794
United       2427.58887  1562.21081  2563.63736    232.47687  3350.07312  4478.02887
Virginia     1016.61769  5005.08126   883.53546   3670.01819  6793.03530  1035.98148
```

데이터 구조 확인과 전처리

❖ 데이터 전처리

```
# normalize input variables (정규화)
utilities.df.norm <- sapply(utilities.df, scale)

# add row names: utilities (정규화 시킨 벡터에 추가)
row.names(utilities.df.norm) <- row.names(utilities.df)
```

- apply (input : array, output : array)
- lapply (input : list or vector, output : list)
- sapply (input : list or vector, output : vector or array)
- vapply (input : list or vector, output : vector or array)
- tapply (input : list or vector and factor, output : vector or array)
- mapply (input : list or vector, output : vector or array)

https://3months.tistory.com/389

데이터 구조 확인과 전처리

```
# compute normalized distance based on variables Sales and FuelCost
d.norm <- dist(utilities.df.norm[,c(6,8)], method = "euclidean")
```

```
> d.norm
             Arizona    Boston   Central  Commonwealth        NY  Florida  Hawaiian      Idaho
Boston       2.0103293
Central      0.7741795 1.4657027
Commonwealth 0.7587375 1.5828208 1.0157104
NY           3.0219066 1.0133700 2.4325285    2.5719693
Florida      1.2444219 1.7923968 0.6318918    1.6438566 2.6355728
Hawaiian     1.8852481 0.7402833 1.1560922    1.7460268 1.4116954 1.2288047
Idaho        1.2656380 3.1766540 1.7327770    2.0032300 4.1625615 1.7641233 2.8601888
Kentucky     0.4612918 1.5577377 0.4192538    0.6299937 2.5664387 1.0256629 1.4368218 1.6504169
Madison      0.7386496 1.7196319 1.1022872    0.1387579 2.7054453 1.7225099 1.8803606 1.9502960
Nevada       2.3694792 3.7565131 2.3759746    3.1060838 4.5970059 1.9715184 3.1853105 1.4795256
New England  2.4259752 0.6843933 1.7373219    2.1538314 0.8462906 1.8313804 0.6081070 3.4587708
Northern     0.5646572 1.9401658 1.1134329    0.3770043 2.9386369 1.6986240 2.0272242 1.7084093
Oklahoma     0.1826480 2.1660781 0.8550928    0.9373890 3.1745882 1.2436342 1.9970362 1.0834492
Pacific      1.5707796 0.4783340 0.9877719    1.2588346 1.4620188 1.3431847 0.5609973 2.7055789
Puget        1.9476675 3.5013904 2.0656431    2.6990600 4.3974331 1.7675811 2.9958483 0.9920924
San Diego    2.5090434 0.6796342 1.8367621    2.2029297 0.7156293 1.9534230 0.7260955 3.5637271
Southern     0.9136210 1.6344254 0.2764402    1.2785143 2.5584087 0.3667437 1.2050342 1.6586708
Texas        1.2479759 2.8905601 1.4281594    1.9988179 3.8311318 1.2779197 2.4632271 0.6000891
Wisconsin    0.5214913 1.6542554 0.8389668    0.2434079 2.6617861 1.4524174 1.7112561 1.7788126
United       2.7617447 1.1005949 2.0348238    2.5471162 0.9525069 2.0164926 0.8799342 3.7204215
Virginia     1.2523502 1.4792607 0.5103653    1.5020926 2.3286909 0.3138469 0.9294143 1.9807148
```

주요 함수 문법

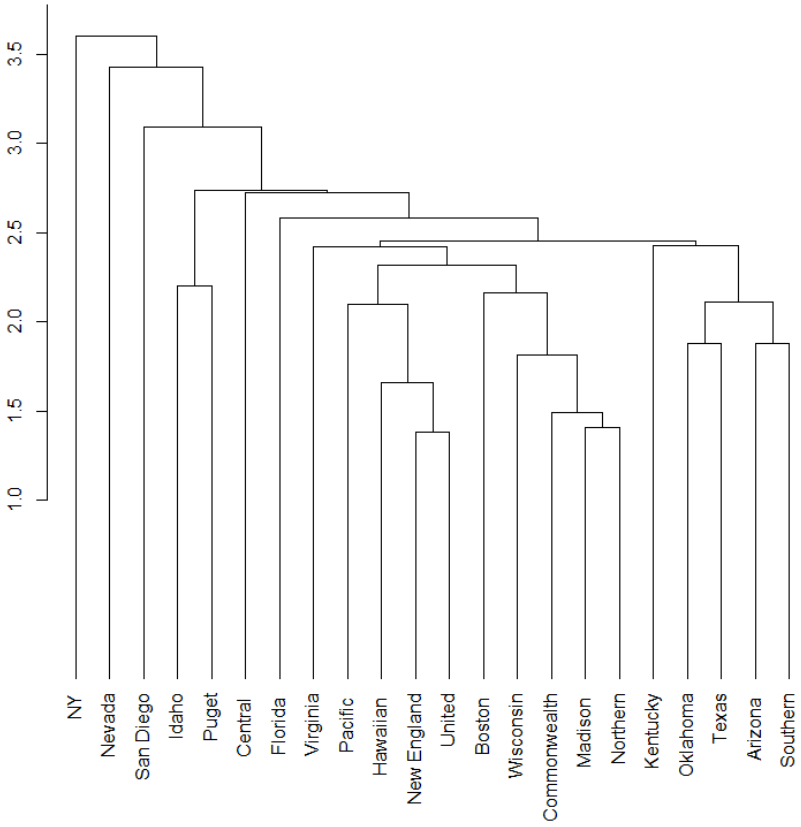
```
# compute normalized distance based on all 8 variables
d.norm <- dist(utilities.df.norm, method = "euclidean")

# hcluster를 이용하여 덴드로그램 설정
hc1 <- hclust(d.norm, method = "single")
plot(hc1, hang = -1, ann = FALSE)
hc2 <- hclust(d.norm, method = "average")
plot(hc2, hang = -1, ann = FALSE)
```

※ hclust 의 거리 계산 기준

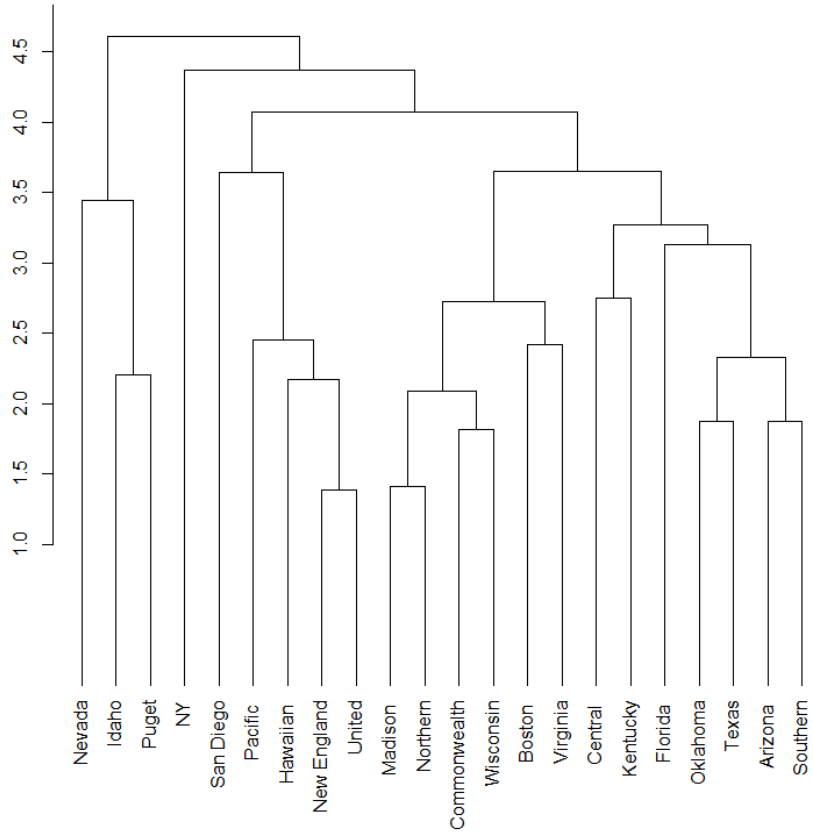
- single : 단일연결법
- complete : 완전연결법
- average : 평균연결법
- median : 중심연결법

모형 구축

```
> plot(hc1, hang = -1, ann = FALSE)
> plot(hc2, hang = -1, ann = FALSE)
```



hc1



hc2

모형 구축

- 덴드로그램을 절단하여 멤버쉽으로 할당

```
memb <- cutree(hc1, k = 6)
memb
```

```
> memb

    Arizona      Boston     Central Commonwealth          NY     Florida     Hawaiian
          1           1           2           1           3           1           1
      Idaho    Kentucky     Madison      Nevada New England    Northern    Oklahoma
          4           1           1           5           1           1           1
    Pacific      Puget  San Diego    Southern       Texas   Wisconsin      United
          1           4           6           1           1           1           1
   Virginia
          1
```

```
memb <- cutree(hc2, k = 6)
memb
```

```
> memb

     Arizona      Boston     Central Commonwealth          NY     Florida     Hawaiian
           1           2           1           2           3           1           4
       Idaho    Kentucky     Madison      Nevada New England    Northern    Oklahoma
           5           1           2           5           4           2           1
    Pacific      Puget  San Diego    Southern       Texas   Wisconsin      United
           4           5           6           1           1           2           4
   Virginia
           2
```

분석 결과 확인

❖ **Heatmap을 그리기 위한 레이블 부여**

```
# set labels as cluster membership and utility name
row.names(utilities.df.norm) <- paste(memb, ": ",
                                      row.names(utilities.df),
                                      sep = "")
```

	Fixed_charge	RoR	Cost	Load_factor
Arizona	1.06	9.2	151	54.4
Boston	0.89	10.3	202	57.9
Central	1.43	15.4	113	53.0
Commonwealth	1.02	11.2	168	56.0
NY	1.49	8.8	192	51.2
Florida	1.32	13.5	111	60.0

	Fixed_charge	RoR	Cost
1: Arizona	-0.29315791	-0.68463896	-0.417122002
2: Boston	-1.21451134	-0.19445367	0.821002037
1: Central	1.71214073	2.07822360	-1.339645796
2: Commonwealth	-0.50994695	0.20660702	-0.004413989
3: NY	2.03732429	-0.86288816	0.578232617
1: Florida	1.11597086	1.23153991	-1.388199680

분석 결과 확인

❖ Heatmap

rdocumentation.org/packages/stats/versions/3.6.2/topics/heatmap

Usage

```
heatmap(x, Rowv = NULL, Colv = if(symm)"Rowv" else NULL,
        distfun = dist, hclustfun = hclust,
        reorderfun = function(d, w) reorder(d, w),
        add.expr, symm = FALSE, revC = identical(Colv, "Rowv"),
        scale = c("row", "column", "none"), na.rm = TRUE,
        margins = c(5, 5), ColSideColors, RowSideColors,
        cexRow = 0.2 + 1/log10(nr), cexCol = 0.2 + 1/log10(nc),
        labRow = NULL, labCol = NULL, main = NULL,
        xlab = NULL, ylab = NULL,
        keep.dendro = FALSE, verbose = getOption("verbose"), ...)
```

Arguments

x	numeric matrix of the values to be plotted.
Rowv	determines if and how the *row* dendrogram should be computed and reordered. Either a `dendrogram` or a vector of values used to reorder the row dendrogram or `NA` to suppress any row dendrogram (and reordering) or by default, `NULL`, see 'Details' below.
Colv	determines if and how the *column* dendrogram should be reordered. Has the same options as the `Rowv` argument above and *additionally* when `x` is a square matrix, `Colv = "Rowv"` means that columns should be treated identically to the rows (and so if there is to be no row dendrogram there will not be a column one either).
distfun	function used to compute the distance (dissimilarity) between both rows and columns. Defaults to `dist`.
hclustfun	function used to compute the hierarchical clustering when `Rowv` or `Colv` are not dendrograms. Defaults to `hclust`. Should take as argument a result of `distfun` and return an object to which `as.dendrogram` can be applied.
reorderfun	`function(d, w)` of dendrogram and weights for reordering the row and column dendrograms. The default uses `reorder.dendrogram`.
add.expr	expression that will be evaluated after the call to `image`. Can be used to add components to the plot.
symm	logical indicating if `x` should be treated **symm**etrically; can only be true when `x` is a square matrix.
revC	logical indicating if the column order should be `rev`ersed for plotting, such that e.g., for the symmetric case, the symmetry axis is as usual.
scale	character indicating if the values should be centered and scaled in either the row direction or the column direction, or none. The default is `"row"` if `symm` false, and `"none"` otherwise.
na.rm	logical indicating whether `NA`'s should be removed.
margins	numeric vector of length 2 containing the margins (see `par(mar = *)`) for column and row names, respectively.
ColSideColors	(optional) character vector of length `ncol(x)` containing the color names for a horizontal side bar that may be used to annotate the columns of `x`.

분석 결과 확인

- **Heatmap 설정**

```
heatmap(as.matrix(utilities.df.norm), Colv = NA, hclustfun=hclust,
        col=rev(paste("gray",1:99,sep="")))
```