Hierarchizing Cluster Analysis: Agglomerative and Divisive Methods

Nicholas Maue

PUBPOL 542

3/15/2021

For this analysis, we will be clustering observations based on COVID-19 deaths, as well as similar health and demographic characteristics. We will request a certain number of clusters, and the algorithm will put obsesrvations into one of those clusters based on variable values

We will be doing both agglomerative hierchical clustering, which determines the number of clusters from individual groups, as well as divisive clustering, which determines the number of clusters after starting from one large cluster that contains all observations. I will also compare the results of both types of clustering visually. The unit of analysis here is the county.

Reading in data

```
link='https://github.com/Public-Policy-COVID/students_merge/raw/main/Merged_d
ata.csv'

myfile=url(link)

covid=read.csv(file=myfile)
```

Reset row names to R format

```
row.names(covid)=NULL

str(covid,width = 50,strict.width='cut')

## 'data.frame':    133 obs. of  19 variables:
##  $ Number_of_beds       : num  3667 0 52 553 25 ..
##  $ Number_of_hospitals  : num  22 0 1 6 1 1 10 1..
##  $ Location             : chr  "Alameda_CA" "Al"..
##  $ Urban_Rural_Code     : chr  "Large central m"..
##  $ Deaths_COVID         : int  573 0 31 101 12 1..
##  $ Deaths_total         : int  10908 0 415 2313 ..
```

```
##  $ never                  : num  0.019 0.025 0.045..
##  $ rarely                 : num  0.008 0.085 0.013..
##  $ sometimes              : num  0.055 0.088 0.099..
##  $ frequently             : num  0.123 0.19 0.188 ..
##  $ always                 : num  0.795 0.612 0.655..
##  $ mask_score             : num  3.67 3.28 3.4 3.3..
##  $ total_population       : num  1671329 1129 3975..
##  $ white_total_pct        : num  49.3 67.9 89.7 85..
##  $ black_total_pct        : num  11.03 0.35 2.68 1..
##  $ aian_total_pct         : num  1.06 25.69 2.33 2..
##  $ asian_total_pct        : num  32.33 1.59 1.67 5..
##  $ nhopi_total_pct        : num  0.94 0 0.29 0.29 ..
##  $ multiracial_total_pct: num  5.35 4.43 3.38 4...
```

VARIABLE PREPARATION

First, we want to include the variable for COVID deaths and total in the cluster analysis, but it is currently a string variable. We will need to change to numeric.

```
covid$Deaths_COVID<-as.numeric(covid$Deaths_COVID)
covid$Deaths_total<-as.numeric(covid$Deaths_total)
```

Now, we will choose the variables to cluster. We will include COVID_Deaths, total deaths, total population, mask score, the number of hospital beds, the total percent of the population that is white, and the total percent of the population that is black. Although we only have total deaths and not deaths per 100K, clustering around total population should help control for variation due to population.

```
dfClus=covid[c('Number_of_beds','mask_score','Deaths_COVID','Deaths_total','N
umber_of_hospitals', 'black_total_pct','white_total_pct')]

summary(dfClus)

##  Number_of_beds      mask_score     Deaths_COVID   Deaths_total
##  Min.   :    0.0   Min.   :2.470   Min.   :   0   Min.   :    0
##  1st Qu.:   25.0   1st Qu.:3.301   1st Qu.:   0   1st Qu.:    0
##  Median :  131.0   Median :3.464   Median :  22   Median :  637
##  Mean   :  885.4   Mean   :3.428   Mean   : 206   Mean   : 2896
##  3rd Qu.:  553.0   3rd Qu.:3.591   3rd Qu.: 128   3rd Qu.: 2537
##  Max.   :26672.0   Max.   :3.822   Max.   :8034   Max.   :75463
##  Number_of_hospitals black_total_pct  white_total_pct
##  Min.   :  0         Min.   : 0.000   Min.   :49.28
##  1st Qu.:  1         1st Qu.: 0.770   1st Qu.:82.16
##  Median :  2         Median : 1.260   Median :88.64
##  Mean   :  5         Mean   : 2.318   Mean   :85.50
##  3rd Qu.:  4         3rd Qu.: 2.620   3rd Qu.:91.84
##  Max.   :112         Max.   :14.770   Max.   :96.13
```

Rescale the units into a new variable

```
dfClus=scale(dfClus)
summary(dfClus)
```

```
##  Number_of_beds      mask_score       Deaths_COVID      Deaths_total
##  Min.   :-0.3334   Min.   :-4.2726   Min.   :-0.2704   Min.   :-0.37704
##  1st Qu.:-0.3240   1st Qu.:-0.5659   1st Qu.:-0.2704   1st Qu.:-0.37704
##  Median :-0.2841   Median : 0.1612   Median :-0.2415   Median :-0.29411
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.:-0.1252   3rd Qu.: 0.7277   3rd Qu.:-0.1024   3rd Qu.:-0.04674
##  Max.   : 9.7118   Max.   : 1.7581   Max.   :10.2736   Max.   : 9.44771
##  Number_of_hospitals black_total_pct   white_total_pct
##  Min.   :-0.44686    Min.   :-0.8976   Min.   :-3.8920
##  1st Qu.:-0.35749    1st Qu.:-0.5994   1st Qu.:-0.3585
##  Median :-0.26812    Median :-0.4097   Median : 0.3379
##  Mean   : 0.00000    Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:-0.08937    3rd Qu.: 0.1169   3rd Qu.: 0.6818
##  Max.   : 9.56284    Max.   : 4.8214   Max.   : 1.1428
```

We will set Location as the row names, which will allow us to verify results for each county.

```
row.names(dfClus)=covid$Location
head(dfClus)
```

```
##                Number_of_beds mask_score Deaths_COVID Deaths_total
## Alameda_CA          1.0476322  1.0666781    0.4816583   1.04310240
## Alpine_CA          -0.3334445 -0.6640201   -0.2703586  -0.37704284
## Amador_CA          -0.3138601 -0.1465949   -0.2296736  -0.32301275
## Butte_CA           -0.1251719 -0.2090427   -0.1378042  -0.07590643
## Calaveras_CA       -0.3240289 -0.6104934   -0.2546096  -0.32691854
## Colusa_CA          -0.3153666  0.1790262   -0.2546096  -0.36194046
##                Number_of_hospitals black_total_pct white_total_pct
## Alameda_CA              1.51932965       3.3732540     -3.89196747
## Alpine_CA              -0.44686166      -0.7620594     -1.88666366
## Amador_CA              -0.35748933       0.1401204      0.44640953
## Butte_CA                0.08937233      -0.1618969      0.01654805
## Calaveras_CA           -0.35748933      -0.4794022      0.58611451
## Colusa_CA              -0.35748933      -0.3903458      0.60223432
```

Set seed to ensure replicability of results

```
set.seed(999)
```

Determine the sitance method and compute distance matrix

```
library(cluster)
dfClus_D=cluster::daisy(x=dfClus)
```

HIERARCHIZING AGGLOMERATIVE

Set the number of clusters

```
NumCluster=4
```

Next, apply the function:

```
library(factoextra)

## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://g
oo.gl/ve3WBa

res.agnes= hcut(dfClus_D,
                k = NumCluster,isdiss=T,
                hc_func='agnes',
                hc_method = "ward.D2")
```

Cluster and append results to data frame:

```
covid$agn=as.factor(res.agnes$cluster)
```

Let's check the first cluster results

```
covid[covid$agn==1,'Location']

##  [1] "Alameda_CA"       "Contra Costa_CA"   "Orange_CA"
##  [4] "Riverside_CA"     "Sacramento_CA"     "San Bernardino_CA"
##  [7] "San Diego_CA"     "San Francisco_CA"  "San Mateo_CA"
## [10] "Santa Clara_CA"   "Solano_CA"         "King_WA"
```

Let's check the results through a table

```
table(covid$agn)

##
##  1  2  3  4
## 12 45 75  1
```

The results indicate that Cluster 4 has only one observation.

```
covid[covid$agn==4,'Location']

## [1] "Los Angeles_CA"
```

Los Angeles County appears to be the sole observation.

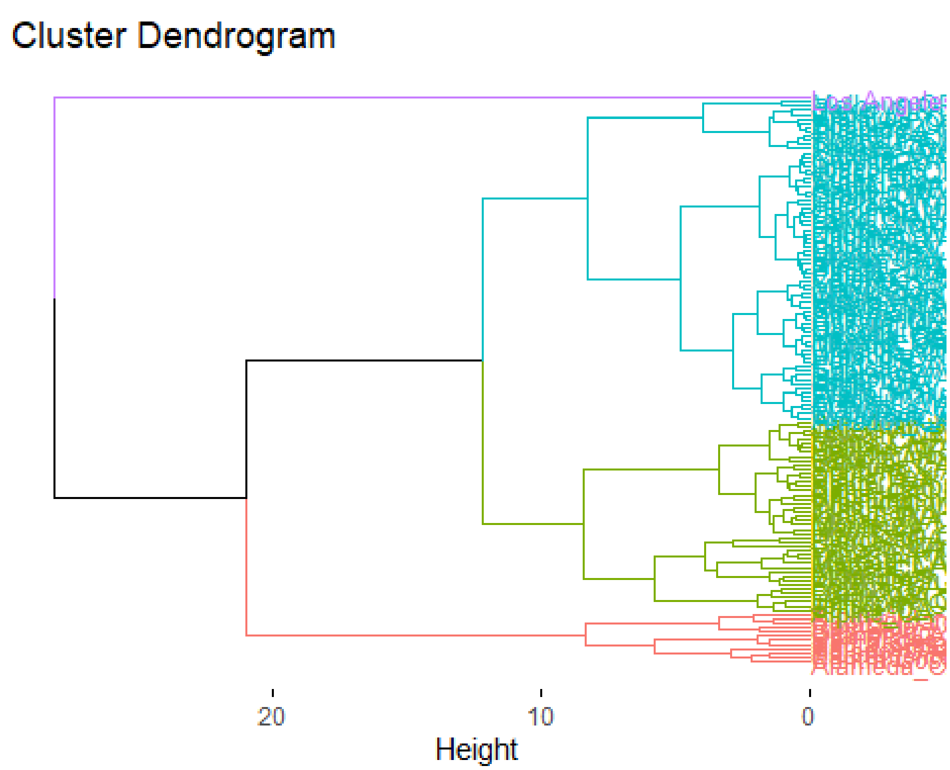
King County's cluster:

```
covid[covid$Location=="King_WA",'agn']

## [1] 1
## Levels: 1 2 3 4
```

VISUALIZING AGGLOMERATIVE RESULTS

We will produce a dendrogram of the cluster results

```
fviz_dend(res.agnes,k=NumCluster, cex = 0.7, horiz = T)
```



Visualize results with a silhouette plot

```r
library(factoextra)
fviz_silhouette(res.agnes)
```

```
##   cluster size ave.sil.width
## 1       1   12          0.24
## 2       2   45          0.14
## 3       3   75          0.47
## 4       4    1          0.00
```

It appears that clusters 1,2, and 3 each have negative silhouettes, which means these are poorly clustered

Saving and verifying individual silhouettes:

```r
agnEval=data.frame(res.agnes$silinfo$widths)
head(agnEval)
```

```
##                    cluster neighbor sil_width
## Alameda_CA               1        2 0.3919787
## King_WA                  1        2 0.3804824
## Sacramento_CA            1        2 0.3653891
## San Bernardino_CA        1        2 0.3530916
## San Diego_CA             1        2 0.3198629
## Riverside_CA             1        2 0.2560581
```

Check the observations with negative silhouettes

```
agnEval[agnEval$sil_width<0,]
```

```
##                cluster neighbor   sil_width
## San Mateo_CA         1        2 -0.20528803
## Lassen_CA            2        3 -0.01850048
## San Joaquin_CA       2        1 -0.05384687
## Jefferson_WA         2        3 -0.07849399
## Del Norte_CA         2        3 -0.13087586
## San Juan_WA          2        3 -0.19746628
## Ferry_WA             2        3 -0.23240216
## Okanogan_WA          2        3 -0.35433693
## San Benito_CA        3        2 -0.04121742
```

In total, there are nine observations that are poorly clustered: One in cluster 1, Seven in Cluster 2, and One in Cluster 3

HIERARCHIZING DIVISIVE METHOD

Apply the function (we will use the same number of clusters, 4, as the agglomerative method)

```
library(factoextra)

res.diana= hcut(dfClus_D, k = NumCluster,
                 hc_func='diana',
                 hc_method = "ward.D")
```

Clustering and appending results to data frame

```
covid$dia=as.factor(res.diana$cluster)
```

Querying the data frame

```
covid[covid$dia==1, 'Location']
```

```
##  [1] "Alameda_CA"       "Contra Costa_CA"  "Fresno_CA"        "Sacramento_
CA"
##  [5] "San Francisco_CA" "San Joaquin_CA"   "San Mateo_CA"     "Santa Clara
_CA"
##  [9] "Solano_CA"        "Pierce_WA"
```

Check the results by each cluster:

```r
table(covid$dia)

##
##   1   2   3   4
##  10 117   1   5
```

The results indicate that Cluster 2 has the large majority of observations. Let's check that one:

```r
covid[covid$dia==2, 'Location']

##   [1] "Alpine_CA"          "Amador_CA"          "Butte_CA"
##   [4] "Calaveras_CA"       "Colusa_CA"          "Del Norte_CA"
##   [7] "El Dorado_CA"       "Glenn_CA"           "Humboldt_CA"
##  [10] "Imperial_CA"        "Inyo_CA"            "Kern_CA"
##  [13] "Kings_CA"           "Lake_CA"            "Lassen_CA"
##  [16] "Madera_CA"          "Marin_CA"           "Mariposa_CA"
##  [19] "Mendocino_CA"       "Merced_CA"          "Modoc_CA"
##  [22] "Mono_CA"            "Monterey_CA"        "Napa_CA"
##  [25] "Nevada_CA"          "Placer_CA"          "Plumas_CA"
##  [28] "San Benito_CA"      "San Luis Obispo_CA" "Santa Barbara_CA"
##  [31] "Santa Cruz_CA"      "Shasta_CA"          "Sierra_CA"
##  [34] "Siskiyou_CA"        "Sonoma_CA"          "Stanislaus_CA"
##  [37] "Sutter_CA"          "Tehama_CA"          "Trinity_CA"
##  [40] "Tulare_CA"          "Tuolumne_CA"        "Ventura_CA"
##  [43] "Yolo_CA"            "Yuba_CA"            "Baker_OR"
##  [46] "Benton_OR"          "Clackamas_OR"       "Clatsop_OR"
##  [49] "Columbia_OR"        "Coos_OR"            "Crook_OR"
##  [52] "Curry_OR"           "Deschutes_OR"       "Douglas_OR"
##  [55] "Gilliam_OR"         "Grant_OR"           "Harney_OR"
##  [58] "Hood River_OR"      "Jackson_OR"         "Jefferson_OR"
##  [61] "Josephine_OR"       "Klamath_OR"         "Lake_OR"
##  [64] "Lane_OR"            "Lincoln_OR"         "Linn_OR"
##  [67] "Malheur_OR"         "Marion_OR"          "Morrow_OR"
##  [70] "Multnomah_OR"       "Polk_OR"            "Sherman_OR"
##  [73] "Tillamook_OR"       "Umatilla_OR"        "Union_OR"
##  [76] "Wallowa_OR"         "Wasco_OR"           "Washington_OR"
##  [79] "Wheeler_OR"         "Yamhill_OR"         "Adams_WA"
##  [82] "Asotin_WA"          "Benton_WA"          "Chelan_WA"
##  [85] "Clallam_WA"         "Clark_WA"           "Columbia_WA"
##  [88] "Cowlitz_WA"         "Douglas_WA"         "Ferry_WA"
##  [91] "Franklin_WA"        "Garfield_WA"        "Grant_WA"
##  [94] "Grays Harbor_WA"    "Island_WA"          "Jefferson_WA"
##  [97] "Kitsap_WA"          "Kittitas_WA"        "Klickitat_WA"
## [100] "Lewis_WA"           "Lincoln_WA"         "Mason_WA"
## [103] "Okanogan_WA"        "Pacific_WA"         "Pend Oreille_WA"
## [106] "San Juan_WA"        "Skagit_WA"          "Skamania_WA"
```

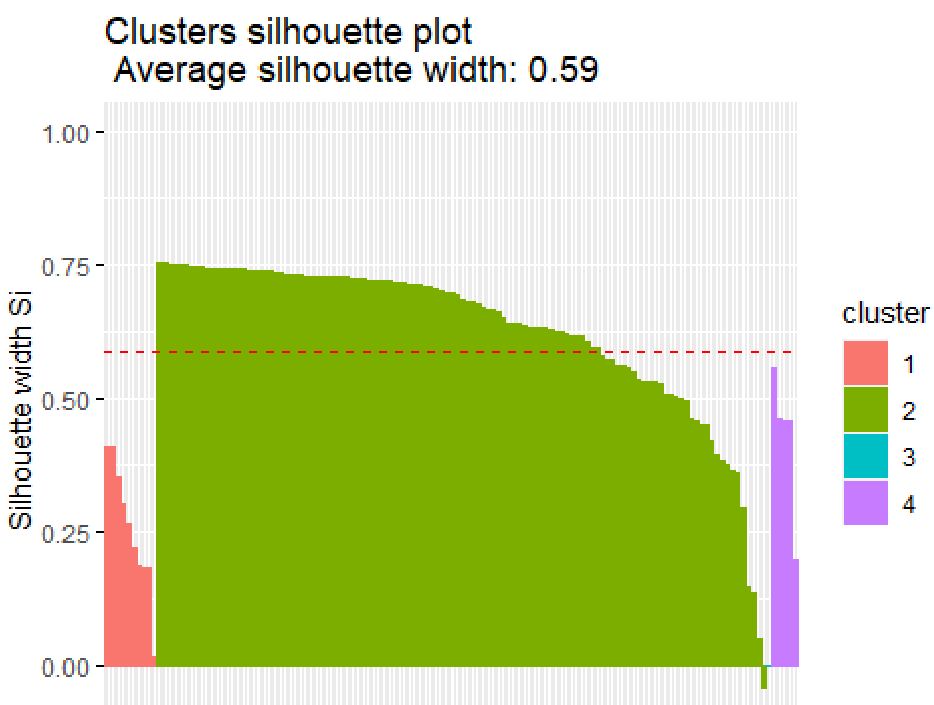
```
## [109] "Snohomish_WA"         "Spokane_WA"           "Stevens_WA"
## [112] "Thurston_WA"          "Wahkiakum_WA"         "Walla Walla_WA"
## [115] "Whatcom_WA"           "Whitman_WA"           "Yakima_WA"
```

Let’s check King County:

```
covid[covid$Location=="King_WA" , 'dia']

## [1] 4
## Levels: 1 2 3 4
```

Produce silhouettes to visualize results and report average silhouettes

```
library(factoextra)
fviz_silhouette(res.diana)

##   cluster size ave.sil.width
## 1       1   10          0.25
## 2       2  117          0.63
## 3       3    1          0.00
## 4       4    5          0.43
```



Cluster 2 has a negative silhouette, meaning it is poorly clustered.

Next we will save and verify silhouettes

```
diaEval=data.frame(res.diana$silinfo$widths)
head(diaEval)
```

```
##                   cluster neighbor sil_width
## San Joaquin_CA          1        4 0.4104780
## Contra Costa_CA         1        4 0.4078738
## Solano_CA               1        4 0.3537789
## San Francisco_CA        1        4 0.3027197
## Alameda_CA              1        4 0.2676310
## Sacramento_CA           1        4 0.2209684
```

Let's check the poorly clustered silhouette in Cluster 2

```
diaEval[diaEval$sil_width<0,]
```

```
##               cluster neighbor   sil_width
## Multnomah_OR        2        1 -0.04078789
```

It looks like Multnomah County is the poorly clustered result

COMPARING AGGLOMERATIVE AND DIVISIVE CLUSTERS

Prepare a bidimensional map

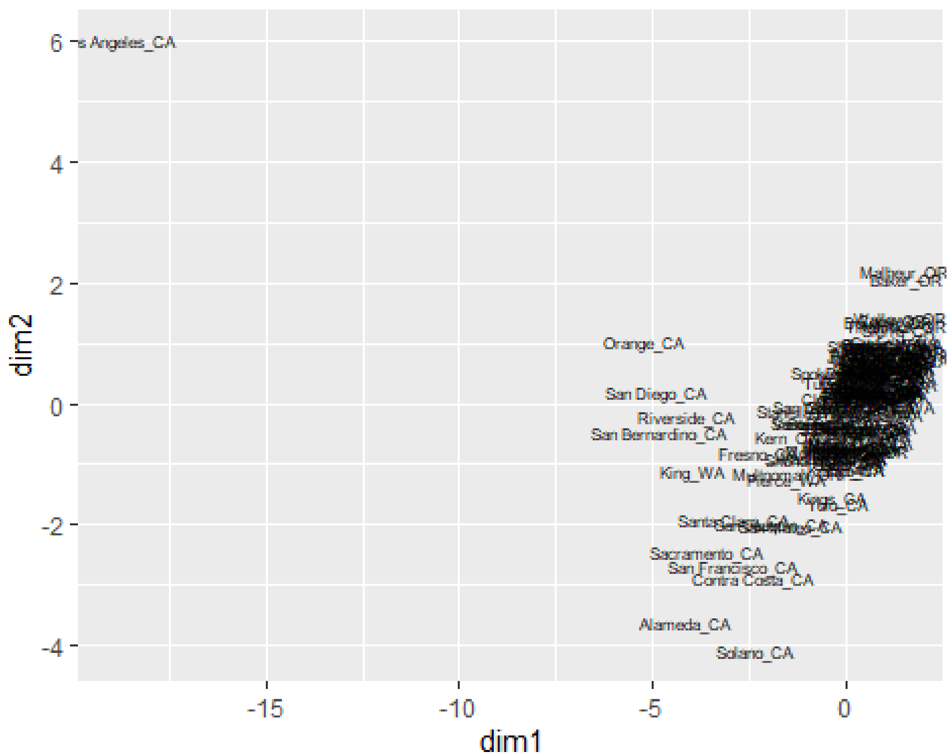
```
projectedData = cmdscale(dfClus_D, k=2)
```

Save coordinates to original data frame

```
covid$dim1 = projectedData[,1]
covid$dim2 = projectedData[,2]
```

Map the Clusters

```
base= ggplot(data=covid,
             aes(x=dim1, y=dim2,
                 label=Location))
base + geom_text(size=2)
```



Plot the Agglomerative Results

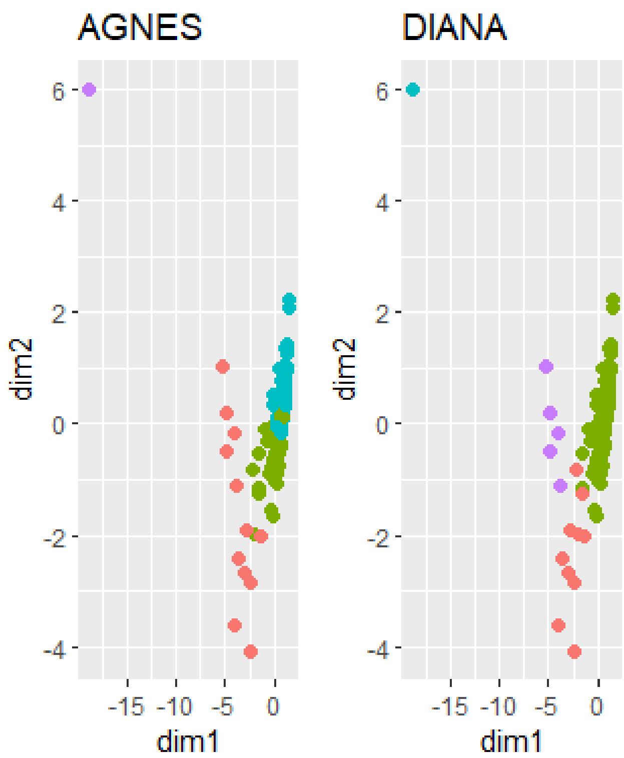
```
agnPlot=base + labs(title = "AGNES") + geom_point(size=2,
                                                 aes(color=agn),
                                                 show.legend = F)
```

Plot the Divisive Results

```
diaPlot=base + labs(title = "DIANA") + geom_point(size=2,
                                                 aes(color=dia),
                                                 show.legend = F)
```

Let’s look at the visual results

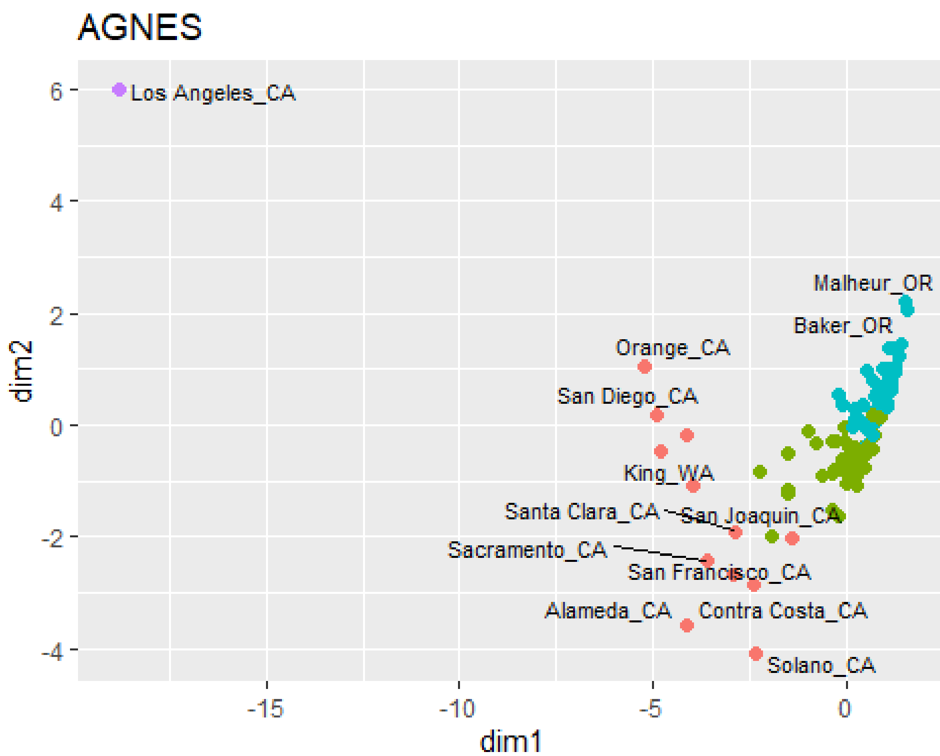
```
library(ggpubr)
ggarrange(agnPlot, diaPlot,ncol = 3)
```



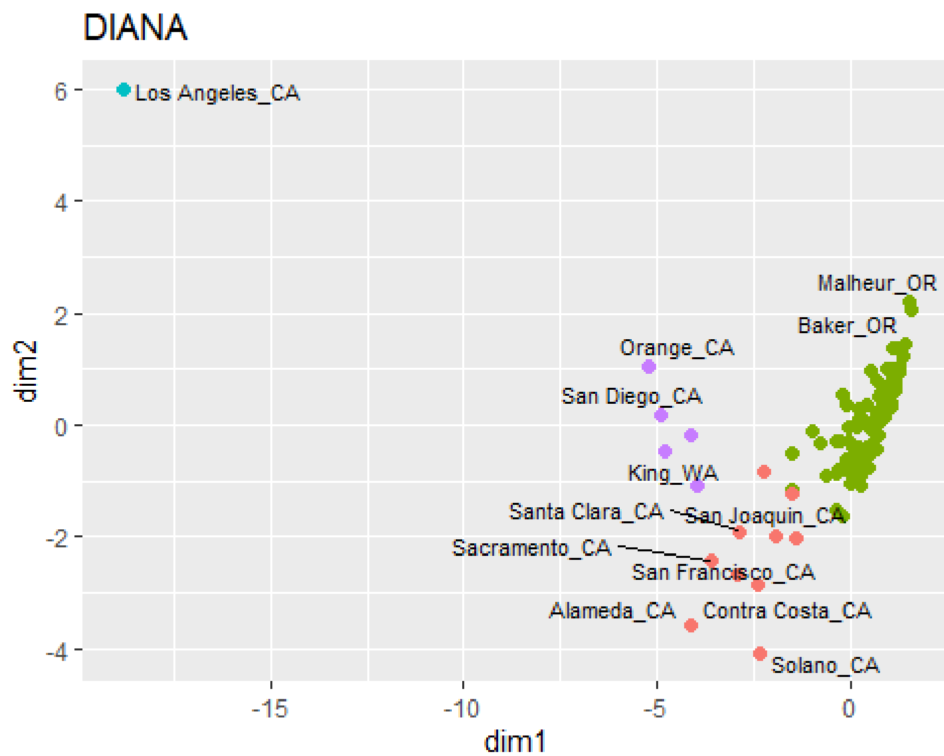
From the visual plots, the results for both hierarchical clustering methods appear to be pretty consistent.

We can label the two hierarchical clustering plots

```
library(ggrepel)
agnPlot + geom_text_repel(size=3,aes(label=Location))

## Warning: ggrepel: 120 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

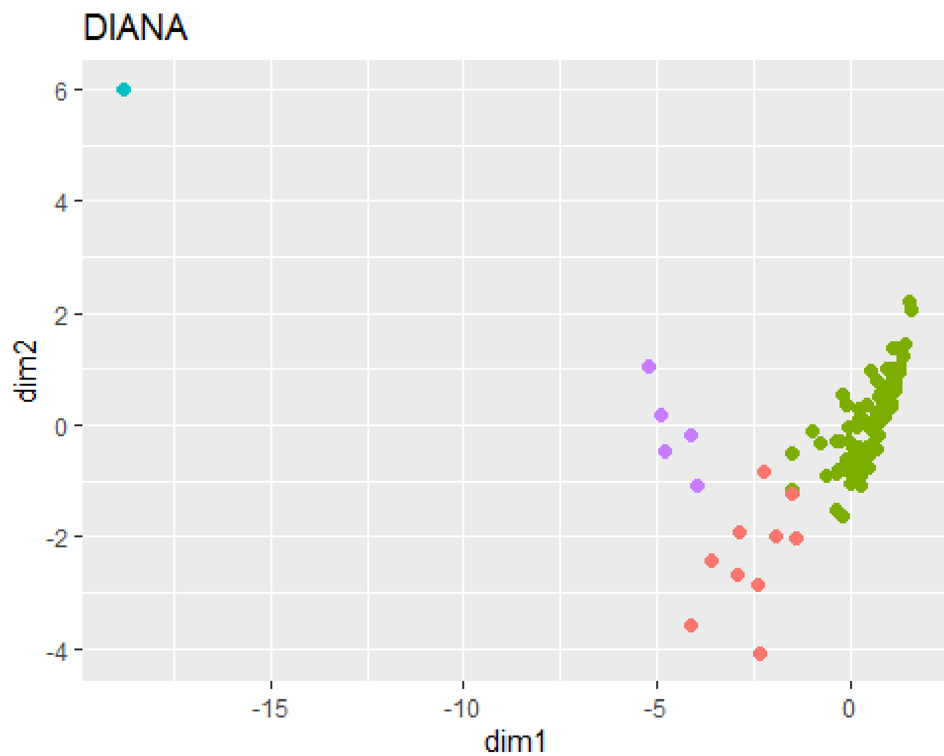



```
diaPlot + geom_text_repel(size=3,aes(label=Location))

## Warning: ggrepel: 120 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

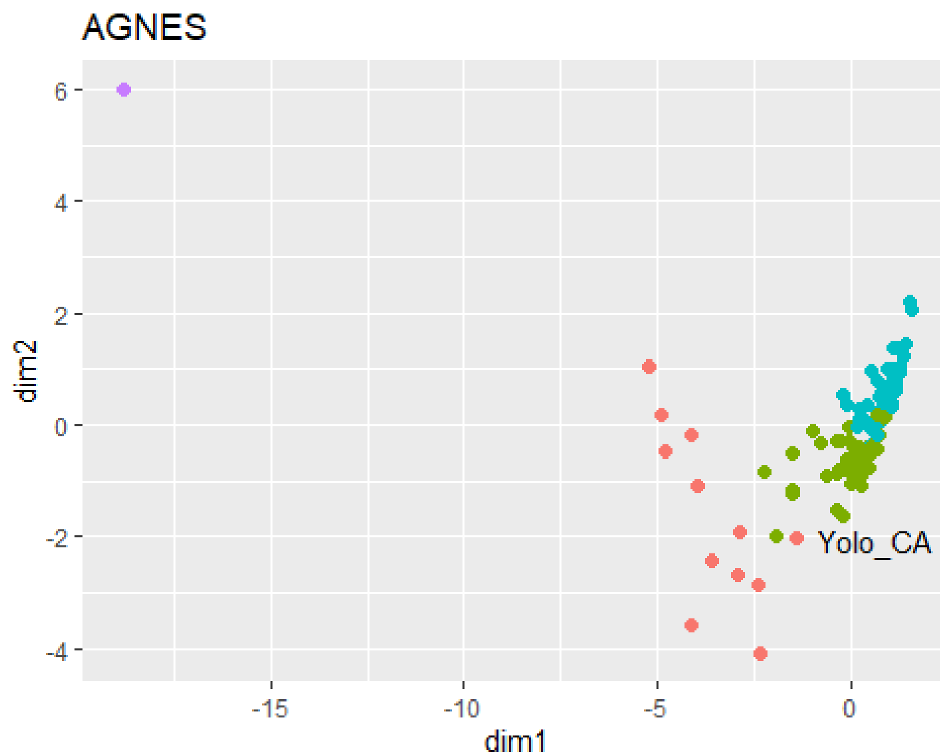


It looks like there are too many overlaps for the large majority of names to appear. Instead, we will need to try and label anomalies from the agn and dia plots

```
LABEL=ifelse(diaEval$sil_width<0, covid$Location,"")

diaPlot + geom_text_repel(aes(label=LABEL))

## Warning: ggrepel: 1 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
LABEL=ifelse(agnEval$sil_width<0, covid$Location,"")

agnPlot + geom_text_repel(aes(label=LABEL))

## Warning: ggrepel: 8 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



It looks like we still get some overlaps