Exploratory Cluster Analysis for Josie Schafer

Michael Flynn, Prior Analytics, LLC.

Cleaning Data

This first code block loads the data and performs any necessary cleaning, rescaling, etc.

First, there don't appear to be any missing values in any rows.

Second, for now I'm focusing primarily on broader demographic and institutional indicators for now, but also some more targeted variables that would likely help to explain disparate economic outcomes. For example, high-research universities granting PhDs or a higher number of community hospital beds. There are other variables that we could include that might be useful for some purposes (e.g. Medicare recipients by region) but I expect that these will be closely tracking other age-related demographic variables.

Third, I'm rescaling the variables by dividing the observed value by the largest value of X as follows:

$$\frac{X_i}{\max X}$$

This puts all observed values on a $0 - 1$ scale.

My understanding of clustering techniques is that when they calculate the distance between units, they will treat the scale of the variables equivalently. The idea here is to scale all of the cluster inputs so they are all on a $0 - 1$ scale, thereby treating all of them equivalently. That way variables with large values and large ranges don't dominate the clustering procedure.

That said, if there's reason to want to weight input variables differently for clustering we can explore that with more time.

```r
# Read in data and select relevant variables for clusters.
# Focus is on demographic and anchor institution variables.

# Read in raw data file
data <- readxl::read_xlsx(here("data/anchor regions analysis.xlsx"))
```

```r
# List of variables to include in clustering
varlist <- c("totpop_19",  # Total pop
             "popchange",  # pop change
             "medage",     # Median age
             "labfor",     # percent population in labor force
             "pov",       # percent population living in poverty
             "poc",       # people of color as percent of pop
             "highed",     # Percent population with at least bachelor's
             "forborn",    # Percent population foreign born
             "net_mig",    # Net domestic migration
             "highered_emp_qcew",
             "highered_estab_qcew",
             "hospital_emp_qcew",
             "hospital_estab_qcew",
             "inst_ipeds_enrollment_all",
             "inst_ipeds_doctoralunihighrese",
             "inst_ipeds_pellawards",
             "inst_hosp_ahacommunityhospitals",
             "inst_hosp_ahabeds",
             "inst_hosp_nihresearchfunding")

# Rescale the variables from 0-1
data.clean <- data |>
  mutate(across(all_of(varlist), # Variables to scale
                ~.x/max(.x),                                                   # Scale relative to
                .names = "{col}_max")) |>                                      # Add "max" suffix
  dplyr::select(MSA, ends_with("_max")) |>                                     # select chosen vari
  column_to_rownames("MSA")
```

Clustering Methods

Here I start with agglomerative/hierarchical clustering methods. The goal as I understand it is to find a happy medium number of groups that illustrates the variability across regions and anchor institutions while still being tractable for analyses.

The priority here is to construct clusters on the basis of 1) anchor institution characteristics, and 2) demographic characteristics of the surrounding region. For now I'll combine these into a single cluster, but we may want to think about constructing two clusters, one on the basis of demographic traits and the other on the basis of anchor institution traits. This would help parse out effects later if the client is interested in using these as predictors in subsequent regression analyses.

I'm going to create a few different clusters and we can compare the characteristics and performance of each, and then choose which one the client likes best.

I chose the "complete" method for the `hclust()` function because it generates a better distribution of clusters than the other methods. For example, others tend to produce either very flat distributions, in which case you may just as well use dummy variables for each MSA or city, or they produce oddly concentrated clusters with 80-90% of observatiosn falling into cluster group 2.

```
distance <- dist(data.clean) # calculate Euclidian distance between obs

hc.tree <- hclust(distance, method = "complete") # Create cluster groupings based on dista


# List of distances to use in generating clusters
cluster.size.list <- list("5" = 5,
                          "10" = 10,
                          "15" = 15,
                          "20" = 20,
                          "25" = 25,
                          "30" = 30,
                          "35" = 35,
                          "40" = 40)

cluster.ids <- map(
  .x = seq_along(cluster.size.list),
  .f = ~ cutree(hc.tree, k = cluster.size.list[[.x]])
                  ) |>
  bind_cols()
```

```
New names:
* `` -> `...1`
* `` -> `...2`
* `` -> `...3`
* `` -> `...4`
* `` -> `...5`
* `` -> `...6`
* `` -> `...7`
* `` -> `...8`
```

```
names(cluster.ids) <- c("cluster_5", "cluster_10", "cluster_15", "cluster_20", "cluster_25

data.out <- data.clean |>
  bind_cols(cluster.ids)
```

Choosing the optimal number of clusters

Figure 1 shows the distribution of the observations depending on the number of clusters chosen. In general, 25–35 clusters seems like a nice balance between parsimony and too much detail. Smaller numbers of clusters, like 5 or 10, group too many areas together (see the spike at group #1). In general we see there are regularly spikes like these, but we start to get more variability as we move towards the 25–30 range.

```
cluster.list <- list("cluster_5", "cluster_10", "cluster_15", "cluster_20", "cluster_25",



plot.out <- data.out |>
  dplyr::select(starts_with("cluster")) |>
  map2(
    .y = cluster.list,
    .f = ~{ggplot(data.out, aes(x = .)) +
    geom_histogram(bins = 40) +
        labs(title = .y)}
)


patchwork::wrap_plots(plot.out)
```

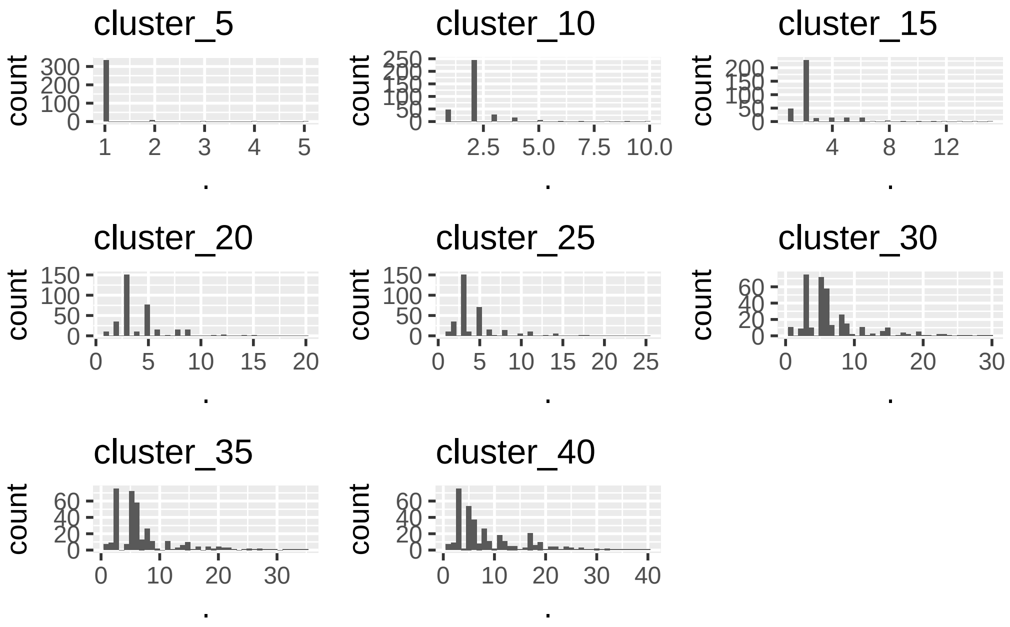

Figure 1: Histograms showing the distribution of clusters depending on the number of clusters chosen.

```
table.data <- data |>
  bind_cols(cluster.ids) |>
  dplyr::select(varlist, cluster_30) |>
  group_by(cluster_30) |>
  dplyr::summarise(across(everything(),
                          mean))
```

```
Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
i Please use `all_of()` or `any_of()` instead.
  # Was:
  data %>% select(varlist)

  # Now:
  data %>% select(all_of(varlist))

See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
names(table.data) <- c("Total Population (2019)",
                      "Population Change",
                      "Median Age",
                      "% Population in Labor Force",
                      "% Population in Poverty",
                      "% Population People of Color",
                      "$% Population with Bachelor's Degree",
                      "% Population Foreign Born",
                      "Net Domestic Migration",
                      "Higher Education Employment",
                      "Higher Education Establishments",
                      "Hospital Employment",
                      "Hospital Establishments",
                      "Higher Education Enrollment",
                      "High Research Doctoral Degree Institutions",
                      "Total Pell Grant Amounts Awarded",
                      "Hospitals/Community Hospitals",
                      "Hospital Beds",
                      "NIH Research Funding",
                      "Cluster")



table.out <- table.data |>
  kbl(longtable = TRUE) |>
  kable_styling(font_size = 8) |>
  scroll_box(height = "600px", width = "800px")

table.out
```

Total Population (2019)	Population Change	Median Age	% Population in Labor Force	% Population in Poverty	% Population P
1	491022.2	4.987475	38.27273	64.10909	
2	994230.7	7.819001	37.26667	63.83333	
3	214126.1	4.060407	39.85067	58.44000	
4	2319054.6	19.071504	36.69000	67.13000	
5	246575.0	7.490234	38.01528	64.73333	
6	577068.9	8.442607	37.72586	65.01379	
7	253319.8	10.775605	32.23077	61.05385	
8	299882.5	9.290796	35.95000	59.55769	
9	447905.1	8.002275	32.08667	58.36000	
10	575400.0	2.552712	39.55000	66.70000	
11	342088.6	10.687640	50.82727	47.78182	
12	125044.0	0.000000	67.40000	22.50000	
13	637543.3	18.178695	50.60000	52.03333	
14	328124.3	9.772405	39.70000	66.56667	
15	2197690.2	6.524644	37.91000	67.07000	
16	404417.0	227.829477	45.50000	57.40000	
17	386114.5	72.536067	40.30000	58.55000	
18	2821709.5	31.846792	39.80000	63.45000	

19	3461420.8	6.809083	37.22000	63.52000	
20	6090660.0	11.166374	41.00000	63.00000	
21	4761603.0	16.685736	36.70000	62.80000	
22	3344589.0	10.782462	38.05000	67.55000	
23	6373281.0	17.481792	35.35000	66.95000	
24	6196585.0	14.397978	37.00000	71.60000	
25	7320663.0	18.952678	34.80000	68.80000	
26	4832346.0	7.642613	38.70000	69.20000	
27	6079130.0	2.833259	38.80000	65.30000	
28	9508605.0	1.320708	37.50000	66.80000	
29	13249614.0	4.132679	36.80000	64.90000	
30	19294236.0	3.173788	38.60000	64.70000	