DSCI353-353m-453: Class w06a-p3 Clustering, Kmeans on Employment Data

Profs: R. H. French, L. S. Bruckman, P. Leu, K. Davis, S. Cirlos

TAs: W. Oltjen, K. Hernandez, M. Li, M. Li, D. Colvin

21 February, 2023

Contents

6.1.3.1 Clustering, is distinct from Classification

- It is covered in Chapter 10, section 10.3 of ISLR

Clustering refers to a very broad set of techniques

- for finding subgroups, or clustering clusters, in a data set.
- When we cluster the observations of a data set,
  - we seek to partition them into distinct groups
  - so that the observations within each group are quite similar to each other,
  - while observations in different groups are quite different from each other.
- Of course, to make this concrete,
  - we must define what it means for two or more observations
  - to be similar or different.
- Indeed, this is often a domain-specific consideration
  - that must be made based on knowledge of the data being studied.

Since clustering is popular in many fields,

- there exist a great number of clustering methods.
- Here we focus on perhaps the two best-known clustering approaches:
  - K-means clustering
  - and hierarchical clustering.

6.1.3.2 Kmeans on Employment Data

- Our modeling goal is to use k-means clustering
  - to explore employment by race and gender.
- This is a good example for those who are new to k-means
  - and want to understand how to apply it to a real-world data set.
- There are two datasets, `tidytuesday-employed`
  - and also `tidytuesday-earn`
  - we won't be using `earn`

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.0
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.2.1     v stringr 1.5.0
## v readr   2.1.4     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
# read in the data
employed <- read_csv("./data/tidytuesday-employed.csv")
```

```
## Rows: 8184 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): industry, major_occupation, minor_occupation, race_gender
## dbl (3): industry_total, employ_n, year
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Let's start by focusing on

- the industry and occupation combinations available in this data,
  - and average over the years available.
- We aren't looking at any time trends,
  - but instead at the demographic relationships.

```r
employed_tidy <- employed %>%
  filter(!is.na(employ_n)) %>%
  group_by(occupation = paste(industry, minor_occupation), race_gender) %>%
  summarise(n = mean(employ_n)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'occupation'. You can override using the
## `.groups` argument.
```

Let's create a dataframe ready for k-means.

- We need to center and scale the variables we are going to use,
  - since they are on such different scales:
  - the proportions of each category
    * who are Asian, Black, or women
  - and the total number of people in each category.

```r
employment_demo <- employed_tidy %>%
  filter(race_gender %in% c("Women", "Black or African American", "Asian")) %>%
  pivot_wider(names_from = race_gender,
              values_from = n,
              values_fill = 0) %>%
  janitor::clean_names() %>%
  left_join(
    employed_tidy %>%
      filter(race_gender == "TOTAL") %>%
      select(-race_gender) %>%
      rename(total = n)
```

```
  ) %>%
  filter(total > 1e3) %>%
  mutate(across(c(asian, black_or_african_american, women), ~ . / (total)),
         total = log(total),
         across(where(is.numeric), ~ as.numeric(scale(.)))) %>%
  mutate(occupation = snakecase::to_snake_case(occupation))
```

```
## Joining with `by = join_by(occupation)`
```

```
employment_demo
```

```
## # A tibble: 230 x 5
##    occupation                                         asian black~1    women  total
##    <chr>                                              <dbl>   <dbl>    <dbl>  <dbl>
##  1 agriculture_and_related_construction_and_extr~ -0.553 -0.410   -1.31    -1.48
##  2 agriculture_and_related_farming_fishing_and_f~ -0.943 -1.22    -0.509    0.706
##  3 agriculture_and_related_installation_maintena~ -0.898 -1.28    -1.38    -0.992
##  4 agriculture_and_related_manage_ment_business_~ -1.06  -1.66    -0.291    0.733
##  5 agriculture_and_related_management_business_a~ -1.06  -1.65    -0.300    0.750
##  6 agriculture_and_related_office_and_administra~ -0.671 -1.54     2.23    -0.503
##  7 agriculture_and_related_production_occupations -0.385 -0.0372 -0.622   -0.950
##  8 agriculture_and_related_professional_and_rela~ -0.364 -1.17     0.00410 -0.782
##  9 agriculture_and_related_protective_service_oc~ -1.35  -0.647   -0.833   -1.39
## 10 agriculture_and_related_sales_and_related_occ~ -1.35  -1.44     0.425   -1.36
## # ... with 220 more rows, and abbreviated variable name
## #   1: black_or_african_american
```

```
## # A tibble: 230 x 5
##    occupation                          asian black_or_african_a…    women  total
##    <chr>                               <dbl>               <dbl>    <dbl>  <dbl>
##  1 agriculture_and_related_construct… -0.553             -0.410   -1.31    -1.48
##  2 agriculture_and_related_farming_f… -0.943             -1.22    -0.509    0.706
##  3 agriculture_and_related_installat… -0.898             -1.28    -1.38    -0.992
##  4 agriculture_and_related_manage_me… -1.06              -1.66    -0.291    0.733
##  5 agriculture_and_related_managemen… -1.06              -1.65    -0.300    0.750
##  6 agriculture_and_related_office_an… -0.671             -1.54     2.23    -0.503
##  7 agriculture_and_related_productio… -0.385             -0.0372 -0.622   -0.950
##  8 agriculture_and_related_professio… -0.364             -1.17     0.00410 -0.782
##  9 agriculture_and_related_protectiv… -1.35              -0.647   -0.833   -1.39
## 10 agriculture_and_related_sales_and… -1.35              -1.44     0.425   -1.36
## # … with 220 more rows</dbl></dbl></dbl></dbl></chr>
```

6.1.3.2.1 Implement k-means clustering

- In the `stats` package
  - is the`kmeans` function
- Now we can implement k-means clustering,
  - starting out with three centers.
- What does the output look like?

```
?stats::kmeans
employment_clust <-
  stats::kmeans(select(employment_demo, -occupation), centers = 3)
summary(employment_clust)
```

```
##              Length Class  Mode
```

```
## cluster      230    -none- numeric
## centers       12    -none- numeric
## totss          1    -none- numeric
## withinss       3    -none- numeric
## tot.withinss   1    -none- numeric
## betweenss      1    -none- numeric
## size           3    -none- numeric
## iter           1    -none- numeric
## ifault         1    -none- numeric
```

```
##              Length Class  Mode
## cluster      230    -none- numeric
## centers       12    -none- numeric
## totss          1    -none- numeric
## withinss       3    -none- numeric
## tot.withinss   1    -none- numeric
## betweenss      1    -none- numeric
## size           3    -none- numeric
## iter           1    -none- numeric
## ifault         1    -none- numeric
```

The original format of the output

- isn't as practical to deal with in many circumstances,
- so we can load the `broom` package (part of `tidymodels`)
  - and use verbs like `tidy()`.
- This will give us the centers of the clusters we found:

```
library(broom)
tidy(employment_clust)
```

```
## # A tibble: 3 x 7
##     asian black_or_african_american    women  total  size withinss cluster
##     <dbl>                     <dbl>    <dbl>  <dbl> <int>    <dbl> <fct>
## 1 -0.0753                      1.14  -0.00594 -0.317    60     133. 1
## 2 -0.788                      -0.680 -0.867   -0.619    78     145. 2
## 3  0.717                      -0.169  0.739    0.732    92     230. 3
```

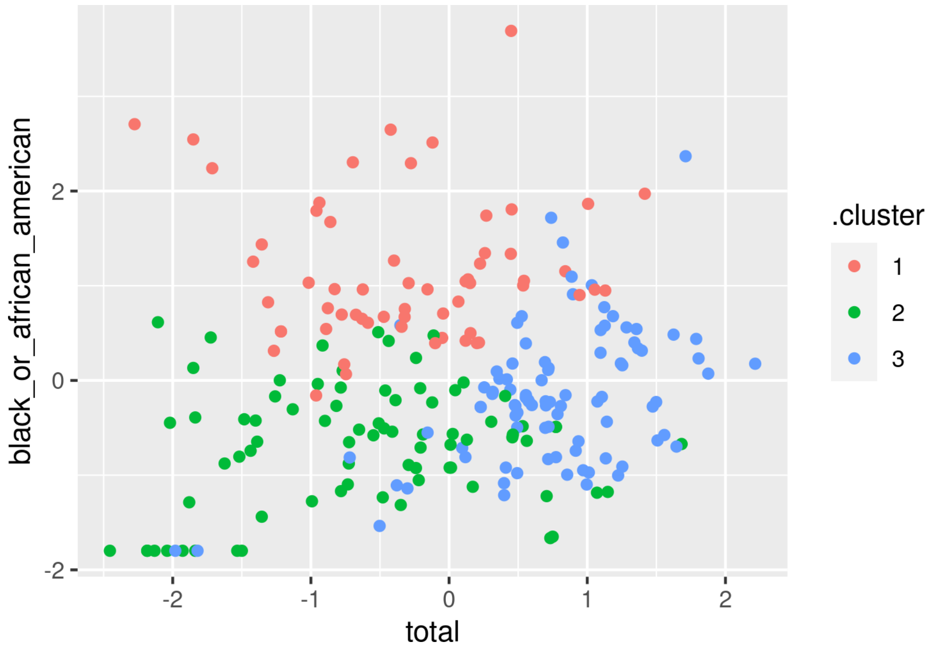
```
## # A tibble: 3 x 7
##      asian black_or_african_american  women  total  size withinss cluster
##      <dbl>                     <dbl>  <dbl>  <dbl> <int>    <dbl> <fct>
## 1  1.46                       -0.551  0.385  0.503    45     125. 1
## 2 -0.732                      -0.454 -0.820 -0.655    91     189. 2
## 3  0.00978                     0.704  0.610  0.393    94     211. 3</fct></dbl></int></dbl></dbl></d
```

If we `augment()` the clustering results with our original data,

- we can plot any of the dimensions of our space,
  - such as **total employed** vs. **proportion who are Black**.
- We can see here that
  - there are really separable clusters
  - but instead a smooth, continuous distribution
    - from low to high along both dimensions.
  - Switch out another dimension like `asian`
    - to see that projection of the space.

```
augment(employment_clust, employment_demo) %>%
  ggplot(aes(total, black_or_african_american, color = .cluster)) +
```

```
  geom_point()
```

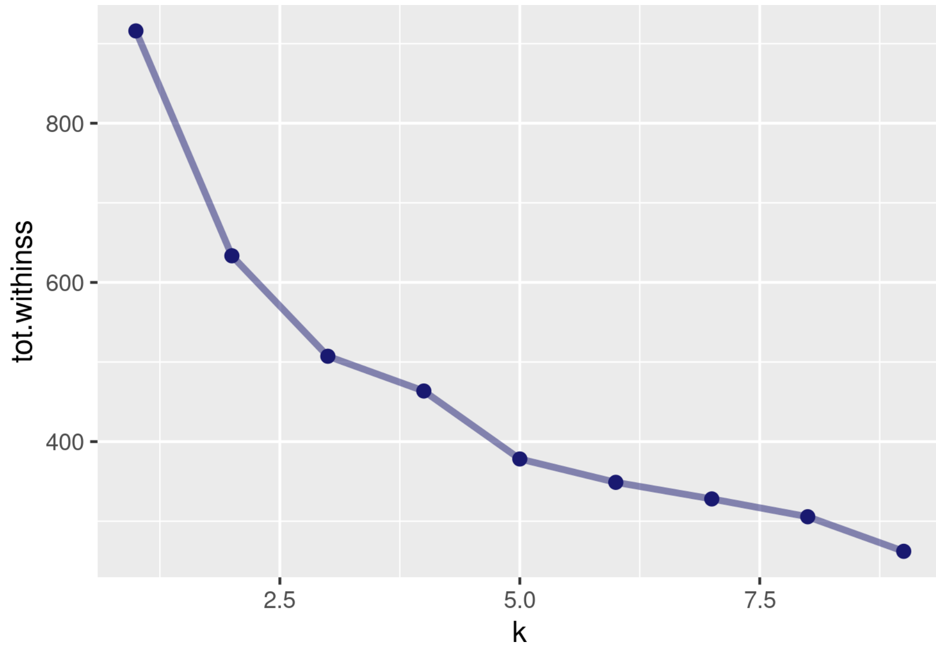


6.1.3.2.2 Choosing the value of `k` in Kmeans We used `k = 3` but how do we know that's right?

- There are lots of complicated
  - or "more art than science" ways of choosing `k`.
- One way is to look at
  - the total within-cluster sum of squares
  - and see if it stops dropping off so quickly at some value for k.
- We can get that from another verb from broom,
  - `glance()`
- Let's try lots of values for `k`
  - and see what happens to the total sum of squares.

```
kclusts <-
  tibble(k = 1:9) %>%
  mutate(kclust = map(k, ~ kmeans(select(
    employment_demo, -occupation
  ), .x)),
  glanced = map(kclust, glance),
  )

kclusts %>%
  unnest(cols = c(glanced)) %>%
  ggplot(aes(k, tot.withinss)) +
  geom_line(alpha = 0.5,
            size = 1.2,
            color = "midnightblue") +
  geom_point(size = 2, color = "midnightblue")
```



I don't see a major "elbow"

- but I'd say that k = 5 looks pretty reasonable.
- Let's fit k-means again.

```
final_clust <-
  kmeans(select(employment_demo, -occupation), centers = 5)
```

To visualize this final result,

- let's use `plotly`
  - and add the occupation name
  - to the hover
- so we can mouse around
  - and see which occupations are more similar.

```
library(plotly)
```

```
## Error: package or namespace load failed for 'plotly' in loadNamespace(j <- i[[1L]], c(lib.loc, .libPa
##  there is no package called 'viridisLite'
```

```
p <- augment(final_clust, employment_demo) %>%
  ggplot(aes(total, women, color = .cluster, name = occupation)) +
  geom_point()

ggplotly(p, height = 500)
```

```
## Error in ggplotly(p, height = 500): could not find function "ggplotly"
```

Remember that you can switch out the axes

- for asian or black_or_african_american
  - to explore dimensions.

6.1.3.3 Links

- Julia Silge, "Getting started with k-means and #TidyTuesday employment status", Feb. 2021.

- Z. Huang, "Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values," Data Mining and Knowledge Discovery, vol. 2, no. 3, pp. 283–304, Sep. 1998, doi: 10.1023/A:1009769707641. [Online]. Available: http://link.springer.com/article/10.1023/A:1009769707641.
- K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained K-means Clustering with Background Knowledge," in Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco, CA, USA, 2001, pp. 577–584 [Online]. Available: http://dl.acm.org/citation.cfm?id=645530.655669.
- W. Zhao, H. Ma, and Q. He, "Parallel K-Means Clustering Based on MapReduce," in Cloud Computing, vol. 5931, M. G. Jaatun, G. Zhao, and C. Rong, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 674–679 [Online]. Available: http://link.springer.com/10.1007/978-3-642-10665-1_71.