Cluster Analysis in R

June 6, 2016

Contents

Read in data

We're continuing our exploration of college features, so read in the College Scorecard data:

```
require(dplyr)
require(ggplot2)

colleges = read.delim('./data/colleges.tsv',
                      sep = '\t',
                      header = TRUE)
```

We're also going to use a couple new libraries, `stringr` and `ggdendro`, so go ahead and install those now. You can use the `packages` menu, or the following commands:

```
install.packages('stringr', dependencies = TRUE)
install.packages('ggdendro', dependencies = TRUE)
```

Let's use a realistic application that we might use clustering to solve. Say that I work for a granting agency, like the Gates Foundation, and I want to identify colleges that have high numbers of low-income, and first generation college attendees because I want to give those colleges additional funding. How should I identify which schools to fund?

k-means clustering

One approach would be to use *k*-means to identify clusters of schools meeting our granting requirements.

```
college_features =
  colleges %>%
    select(institution_name, first_gen_share, poverty_rate,
           family_income_median, median_earnings) %>%
    na.omit() %>%
    distinct()

# run k-means clustering
kmeans_cluster = kmeans(select(college_features, -institution_name), 3)
```

```r
# check what attributes are in the kmeans object
attributes(kmeans_cluster)
```

```
## $names
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
##
## $class
## [1] "kmeans"
```

```r
# Find which cluster the observations belong to
head(kmeans_cluster$cluster, 10)
```

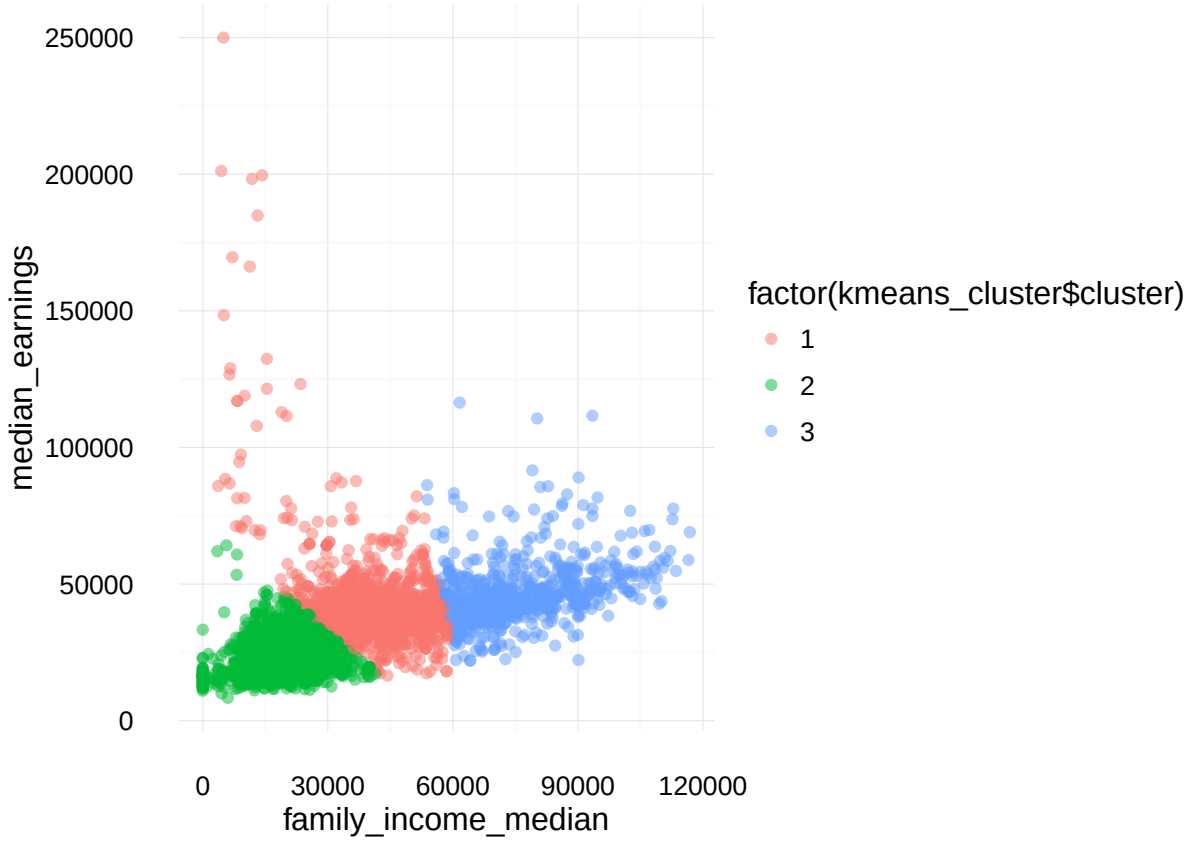
```
##  [1] 2 1 1 1 2 3 2 1 2 3
```

```r
# centers
kmeans_cluster$centers
```

```
##   first_gen_share poverty_rate family_income_median median_earnings
## 1       0.4250142     8.579652             40221.74        41790.14
## 2       0.5483327    12.901803             19231.73        25758.38
## 3       0.2808411     6.285941             75559.67        46343.79
```

Now we can plot the clusters and how they relate to two of the variables. Note that the clusters are labeled by numbers as seen in the figure legend.

```r
# plot 4 clusters
ggplot(college_features,
       aes(x = family_income_median,
           y = median_earnings,
           color = factor(kmeans_cluster$cluster))) +
  geom_point(alpha = 0.50) +
  theme_minimal()
```

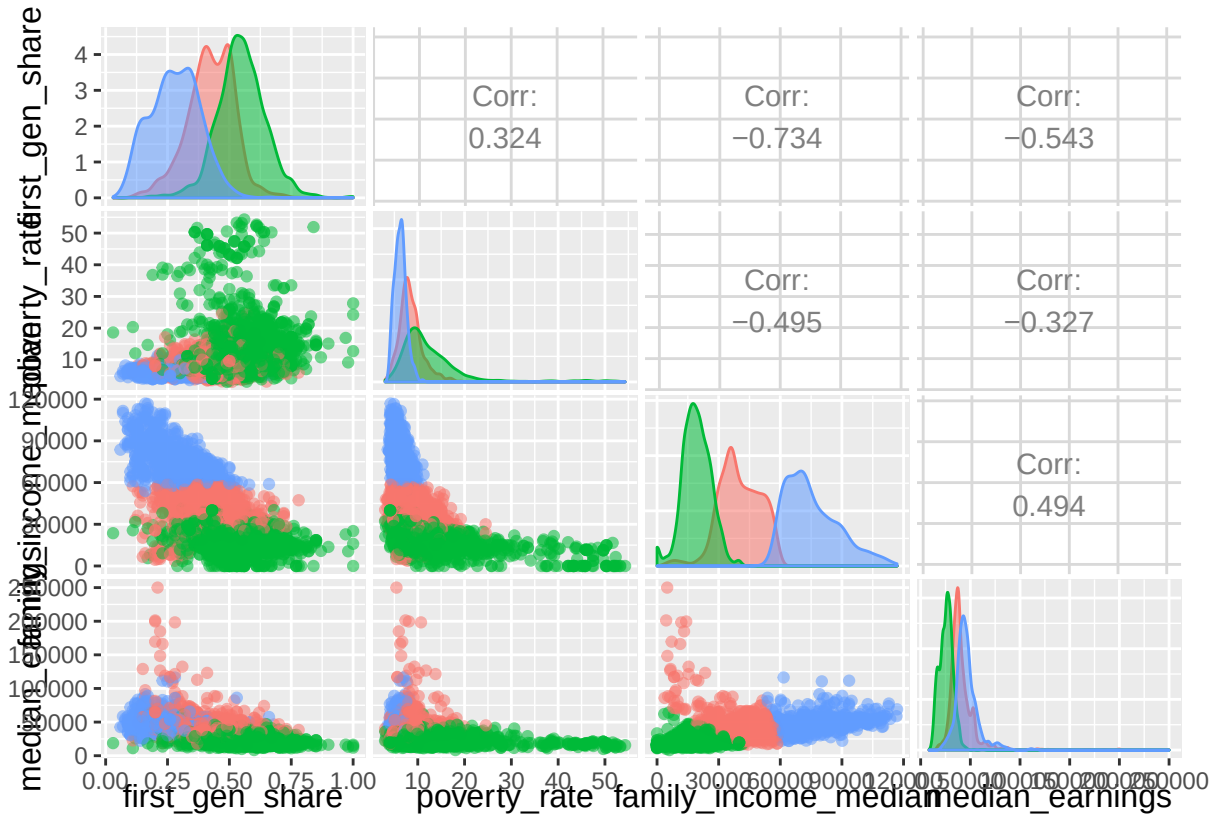


We can use our trusty friend `ggpairs` to visualize the resulting clusters:

```
require(GGally)

college_features =
  college_features %>%
    mutate(cluster = factor(kmeans_cluster$cluster))

ggpairs(college_features,
        lower = list(mapping = aes(color = cluster, alpha = 0.20)),
        diag = list(mapping = aes(fill = cluster, color = cluster, alpha = 0.50)),
        upper = list(mapping = aes(group = cluster)),
        columns = c('first_gen_share', 'poverty_rate', 'family_income_median', 'median_earnings'))
```



What schools should we give money to? (Use the earlier plot to figure out which cluster number to select.)

```
grant_candidates =
  college_features %>%
    filter(cluster == 1)
```

Hierarchical Clustering

Can we find similarities in the colleges based on the composition of majors?

```
require(stringr)
require(ggdendro)

# take a sample to readablility
sample = colleges[sample(nrow(colleges), 50), ]

# select all the columns that contain the string "_major_perc"
majors = sample[, str_detect(names(sample), '_major_perc')]

# put the institution name back on
majors$institution_name = sample$institution_name

# remove missing values
majors = na.omit(majors)
```

```r
# compute the euclidean distance
euclidean = dist(select(majors, -institution_name),
                 method = 'euclidean')

attributes(euclidean)
```

```
## $Size
## [1] 50
##
## $Labels
##  [1] "3747" "1624" "5951" "5110" "30"   "973"  "5477" "639"  "2946" "6851"
## [11] "56"   "151"  "4695" "5443" "6146" "4078" "1708" "5313" "3639" "6107"
## [21] "3034" "6507" "4459" "1390" "3031" "3692" "4999" "1644" "574"  "554"
## [31] "1698" "6848" "5216" "5851" "2865" "4161" "43"   "2307" "7166" "5852"
## [41] "803"  "5433" "563"  "7115" "5147" "1224" "5319" "4807" "1393" "1948"
##
## $Diag
## [1] FALSE
##
## $Upper
## [1] FALSE
##
## $method
## [1] "euclidean"
##
## $call
## dist(x = select(majors, -institution_name), method = "euclidean")
##
## $class
## [1] "dist"
```

```r
# hierarchical clustering
hier = hclust(euclidean)

attributes(hier)
```

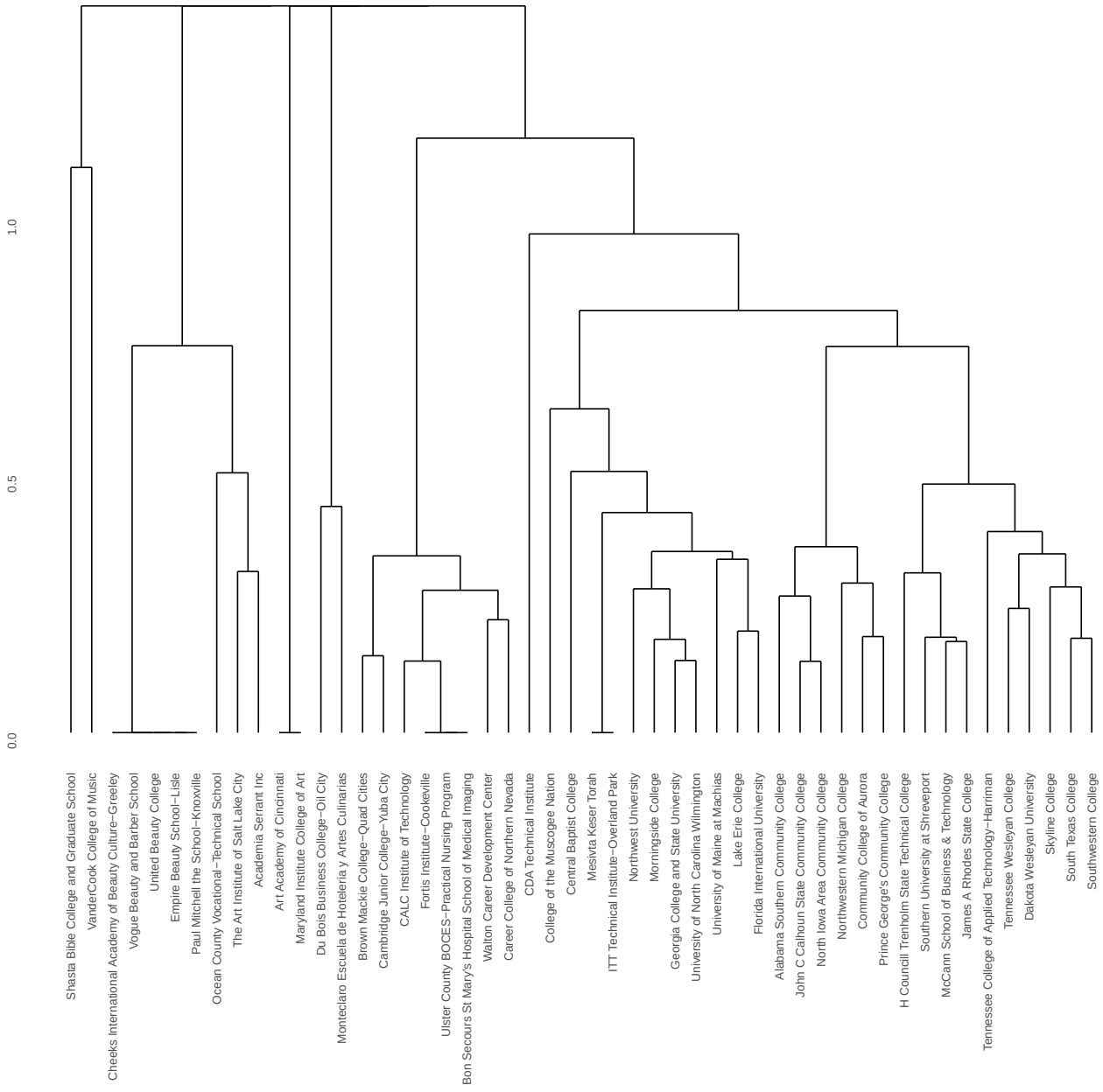
```
## $names
## [1] "merge"       "height"      "order"       "labels"      "method"
## [6] "call"        "dist.method"
##
## $class
## [1] "hclust"
```

```r
# label by id
hier$labels
```

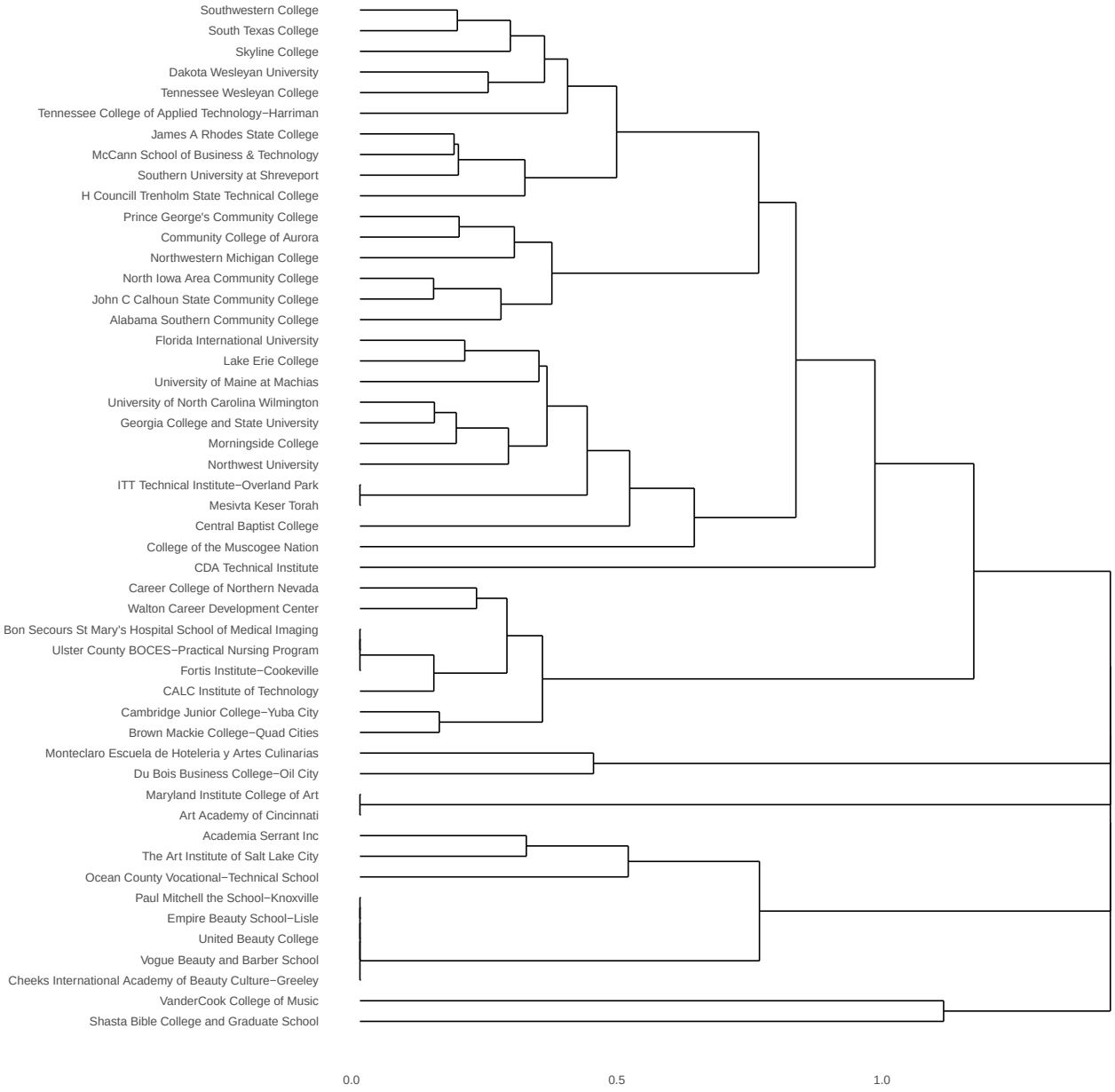
```
##  [1] "3747" "1624" "5951" "5110" "30"   "973"  "5477" "639"  "2946" "6851"
## [11] "56"   "151"  "4695" "5443" "6146" "4078" "1708" "5313" "3639" "6107"
## [21] "3034" "6507" "4459" "1390" "3031" "3692" "4999" "1644" "574"  "554"
## [31] "1698" "6848" "5216" "5851" "2865" "4161" "43"   "2307" "7166" "5852"
## [41] "803"  "5433" "563"  "7115" "5147" "1224" "5319" "4807" "1393" "1948"
```

```
hier$labels = majors$institution_name

# plot dendrogram
ggdendrogram(hier, rotate = FALSE, size = 2)
```




```
ggdendrogram(hier, rotate = TRUE, size = 2)
```



Do the schools we identified as candidates for grants cluster by area of study?

```
# extract the dendrogram data
dendro_data = dendro_data(hier)

attributes(dendro_data)
```

```
## $names
## [1] "segments"    "labels"      "leaf_labels" "class"
##
## $class
## [1] "dendro"
```

```
dendro_data$labels = unique(merge(dendro_data$labels,
                          select(college_features, institution_name, cluster),
                          by.x = 'label',
                          by.y = 'institution_name',
                          all.x = TRUE))

ggplot(segment(dendro_data)) +
  geom_segment(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_text(data = label(dendro_data),
            aes(label = label, x = x, y = 0, hjust = 0, color = cluster),
            size = 2) +
  coord_flip() +
  scale_y_reverse(expand = c(0.25, 0)) +
  theme_minimal() +
  theme(legend.position = 'bottom')
```