Cluster Analysis

DSC365: Introduction to Data Science

2026-11-03

Outline

- Unsupervised Learning Methods
- K-Means Clustering
- Agglomerative Clustering
- Clustering in practice
- Summary

IT IS ELECTION DAY. VOTE.

Announcements

Lab 6

- Due Tuesday November 10 at 11:59 pm in Blueline

Mini Project 3

- Due Thursday November 12 by 2:00pm in Blueline (peer review during class)

Suggested reading: Modern Data Science with R, Chapter 12

Unsupervised Learning Methods

Supervised vs. unsupervised

Supervised learning has *labeled* training data

Unsupervised learning

Clustering vs. Principal Components

1). *Today:* Clustering

2). *Thursday:* Principal Components

Cluster Analysis

Running example: `gapminder`

Goal:

Data: Focus on year 2015

```
1 library(dslabs)
2 gapminder_2015 <- gapminder %>%
3   filter(year == 2015)
4 glimpse(gapminder_2015)
```

```
Rows: 185
Columns: 9
$ country          <fct> "Albania", "Algeria", "Angola", "Antigua and Barbuda"…
$ year             <int> 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015,…
$ infant_mortality <dbl> 12.5, 21.9, 96.0, 5.8, 11.1, 12.6, NA, 3.0, 2.9, 27.9…
$ life_expectancy  <dbl> 78.00, 76.40, 59.60, 76.40, 76.50, 74.70, 75.72, 82.3…
$ fertility        <dbl> 1.78, 2.71, 5.65, 2.06, 2.15, 1.41, 1.66, 1.88, 1.50,…
$ population       <dbl> 2896679, 39666519, 25021974, 91818, 43416755, 3017712…
$ gdp              <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N…
$ continent        <fct> Europe, Africa, Africa, Americas, Americas, Asia, Ame…
$ region           <fct> Southern Europe, Northern Africa, Middle Africa, Cari…
```

Clustering Techniques

Hierarchical vs. Non-hierarchical

1). *Hierarchical* Clustering:

- Example Method:

2). *Non-Hierarchical* Clustering:

- Example Method:

K-Means Clustering

Hierarchical Method: K-Means Clustering

Overview: partitions the data into a **pre-specified number,** K, of non-overlapping clusters

Properties:

1.

2.

K-Means Clustering: Algorithm

Idea: Good clustering means within cluster variation is as small as possible

$$\text{minimize} \left(\sum_{j=1}^{K} W(C_j) \right)$$

- $C_1, \ldots, C_K$:
- $W(C_j)$:

K-Means Clustering: Algorithm (2/4)

Problem: very difficult minimization problem as almost K^n ways to partition n observations into K clusters.

Solution(s):

K-Means Clustering: Algorithm (2/4)

Steps:

1). Initial Cluster Assignment:

2). Iterate until the clusters stop changing.

Animation

K-Means Clustering: Algorithm (4/4)

With these steps will reach a **local optimum**:

How do you choose k?

- No definitive answer. Somewhat subjective.
- Some methods

Example: `gapminder`

The data is already "categorized" by continent and region. Clustering allows us to find other, possibly unconsidered, similarities between points.

Let's start with a two-variable clustering (`fertility` and `life_expectancy`) on countries in the Americas.

```
gap_fle <- gapminder_2015 %>%
  filter(continent == "Americas") %>%
  filter(fertility != 'NA') %>%
  select(fertility, life_expectancy)

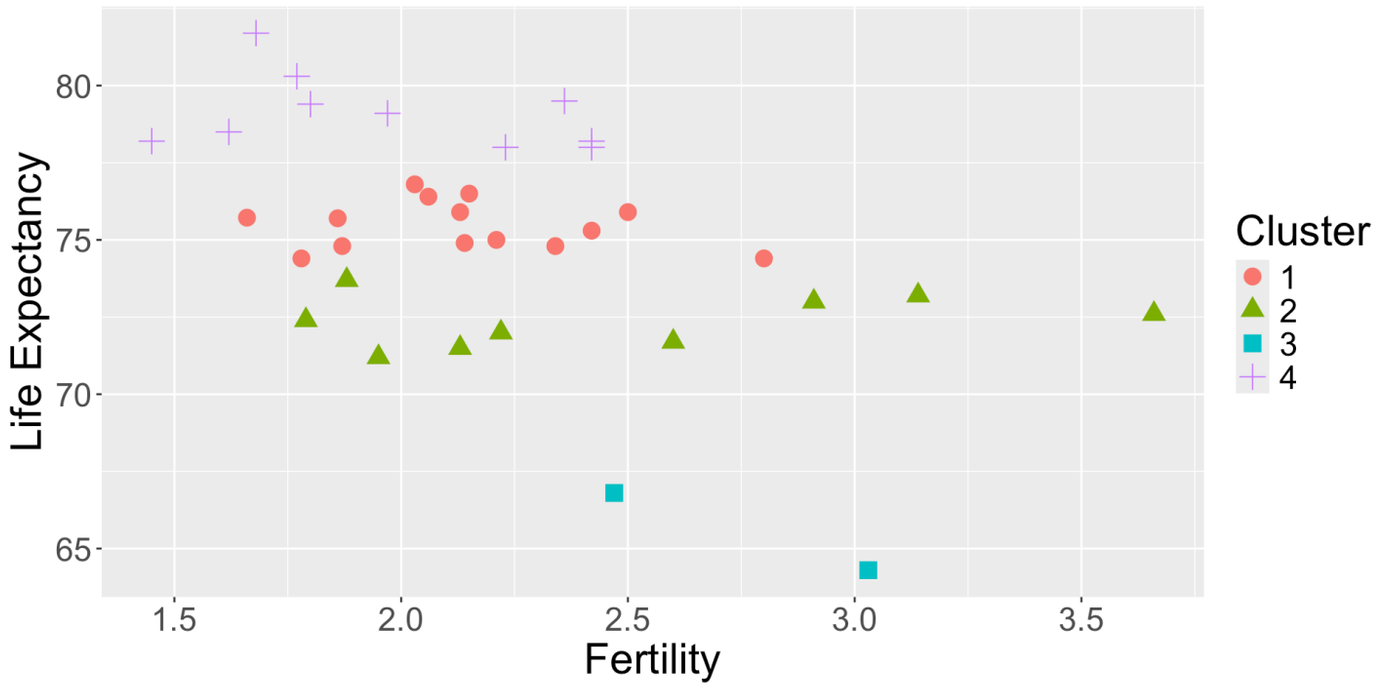
k <- 4  # define number of clusters

set.seed(4)
kmeans_clusters <- kmeans(gap_fle, centers = k, nstart = 10)
```

Example: gapminder (2/3)

What does this look like?

```
1  kmeans_clusters$cluster <- as.factor(kmeans_clusters$cluster)
2
3  ggplot(gap_fle, aes(x=fertility, y=life_expectancy)) +
4    geom_point(aes(color=kmeans_clusters$cluster, pch=kmeans_clusters$cluster), size = 4
5    labs(y = "Life Expectancy", x = "Fertility",
6         color = "Cluster", shape = "Cluster") +
7    presentation_theme
```

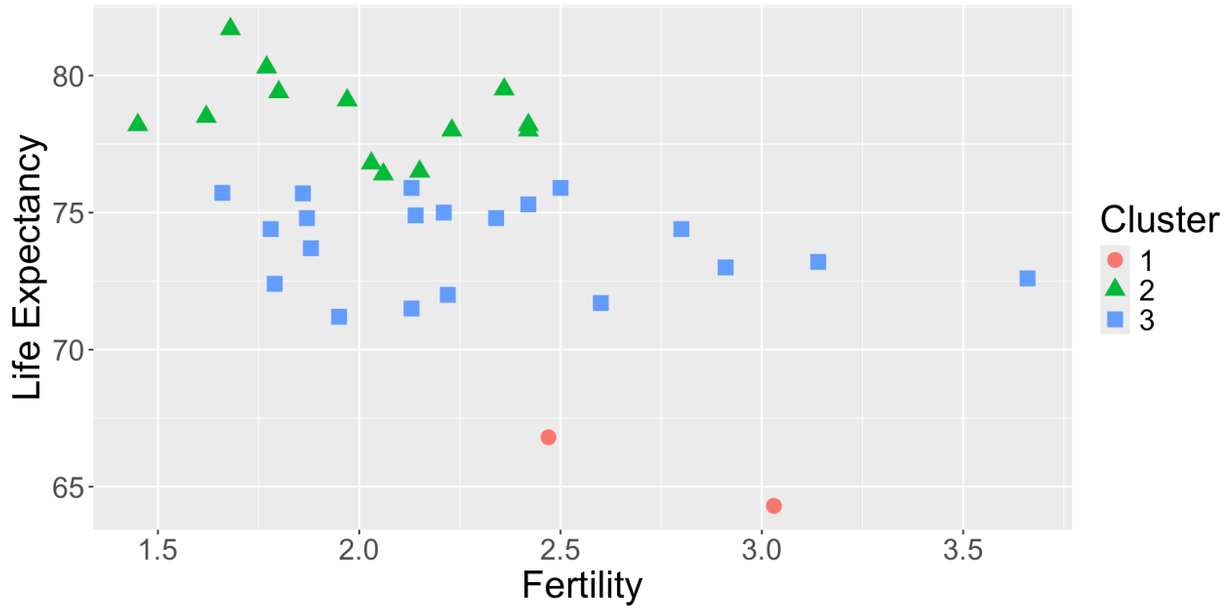


Example: gapminder (3/3)

What if we choose fewer clusters?

```r
k <- 3
kmeans_clusters <- kmeans(gap_fle, centers=k, nstart = 10)
kmeans_clusters$cluster <- as.factor(kmeans_clusters$cluster)

ggplot(gap_fle, aes(x=fertility, y=life_expectancy)) +
  geom_point(aes(color=kmeans_clusters$cluster, pch=kmeans_clusters$cluster), size = 4
  labs(y = "Life Expectancy", x = "Fertility",
       color = "Cluster", shape = "Cluster") +
  presentation_theme
```



Agglomerative Clustering

Hierarchical Method: Agglomerative Clustering

- Doesn’t require pre-specified number of clusters
- Tree-based representation of observations called a **dendrogram**
- **Agglomerative** (bottom-up) clustering: start from leaves and combine clusters up to trunk.
- Cut the tree to obtain a specific number of clusters

Hierarchical Clustering: Algorithm

Steps:

1). Define a **dissimilarity measure** between each pair of observations (usually Euclidean Distance).

2). Iterative:

3). Repeat until all observations are fused together into a single cluster.

Hierarchical Clustering: Algorithm

- **Dissimilarity** is difference between two *observations*
- **Linkage** defines dissimilarity between two *groups* of observations

Output will depend on both *linkage method* and *dissimilarity measure*

Example: `gapminder`

Step 1). Define a dissimilarity measure

- Two variables: `fertility` and `life_expectancy`
- calculate “distance” for every pair of countries
- display as *distance matrix*

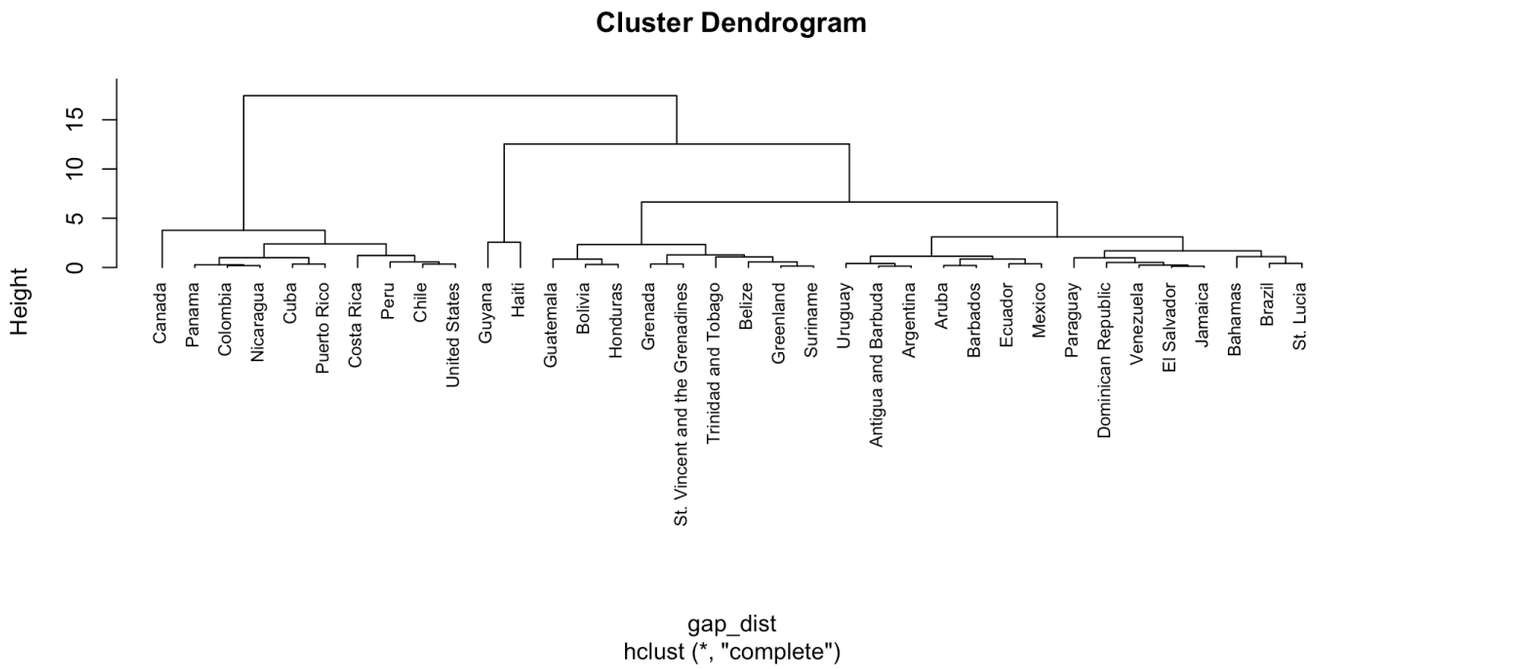
```
1  gap_fle <- gapminder_2015 %>%
2    filter(continent == "Americas") %>%                                    ①
3    select(fertility, life_expectancy, country)
4
5  rownames(gap_fle) <- gap_fle$country
6
7  gap_dist <- dist(gap_fle[,1:2])                                          ②
8
9  gap_dist_matrix <- gap_dist %>% as.matrix()
10 gap_dist_matrix[1:4, 1:4]                                               ③
```

```
                   Antigua and Barbuda Argentina     Aruba  Bahamas
Antigua and Barbuda           0.0000000 0.1345362 0.7889233 2.705993
Argentina                     0.1345362 0.0000000 0.9211406 2.812988
Aruba                         0.7889233 0.9211406 0.0000000 2.031945
Bahamas                       2.7059933 2.8129877 2.0319449 0.000000
```

Example: `gapminder` (2/2)

Steps 2 and 3). Cluster observations based on the similarity measure and linkage method - Plot dendrogram

```
1 clusters <- hclust(
2   gap_dist,                                                    ①
3   method = "complete")                                         ②
4
5 plot(clusters, hang = -1, cex = 0.75)
```



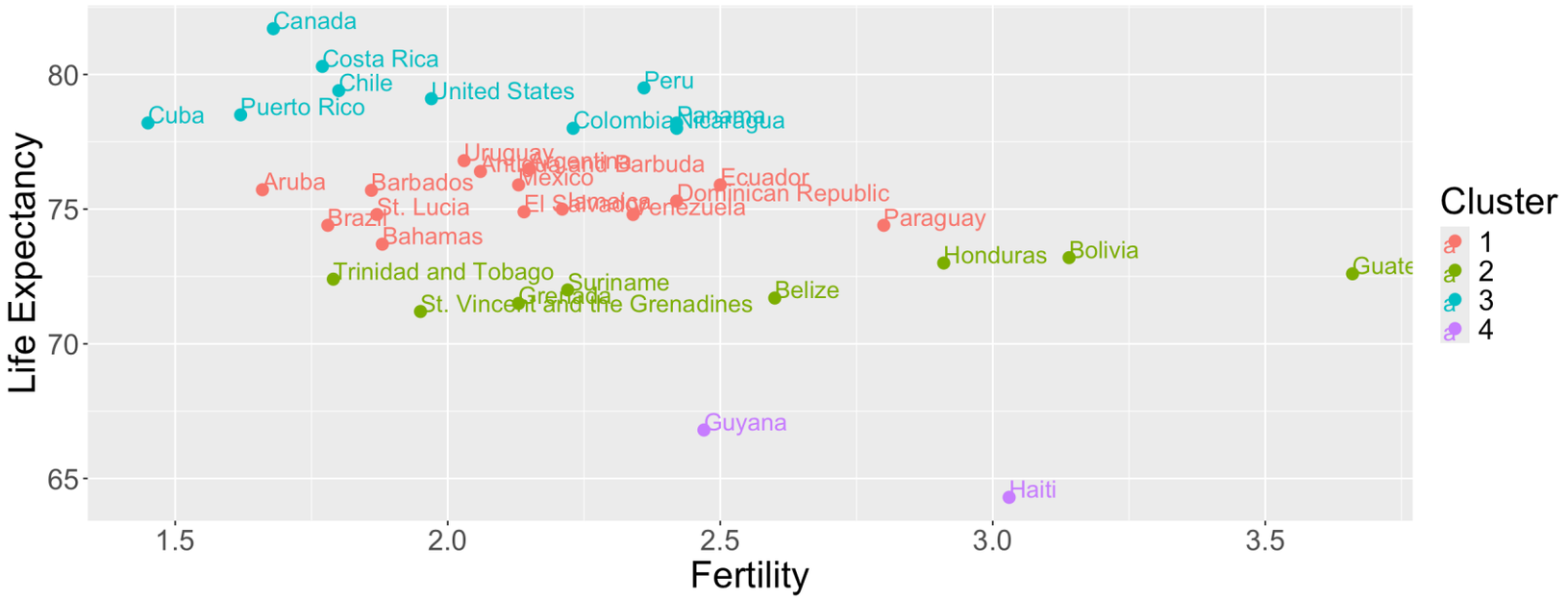
Reading a Dendrogram:

Specifying the number of clusters

Obtain a specific number of clusters by “cutting” the dendrogram using the `cutree` function

```
1 clus4 = cutree(clusters, 4)                                    ①
2
3 gap_clusters <- cbind(gap_fle, clus4)                          ②
```

Visualize with ggplot `gap_clusters %>% ggplot(...)` (full code not shown)



Clustering in practice

Practical issues

Standardizing Inputs

Results can be affected by variables with different scales (see also KNN notes)

Scaling data can mitigate this issue

```
data_scale = gapminder_2015 %>% ①
  filter(continent == "Americas") %>%
  filter(infant_mortality != 'NA') %>%
  select(infant_mortality, life_expectancy) %>%
  scale() %>% ②
  as.data.frame() ③
```

Summary

Hierarchical clustering:

- No need to decide the cluster number before the clustering
- Reproducible result
- Computationally slow

k-means clustering:

- Need to decide the cluster number first
- Randomness may lead to different result
- Computationally fast

Try It Yourself

Now let's add a third variable: `infant_mortality`. Do:

- K-Means clustering (try with different k values). Make a data visualization with `fertility` and `life_expectancy`.
- Hierarchical clustering. Decide on the number of clusters that seems appropriate. Make a data visualization with `fertility` and `life_expectancy` and compare to your k-means output.
- Decide if standardizing your inputs is necessary. If so redo the two steps above and compare your results.