Problem Set #2: Clustering pt. 1

Di Tong

10/16/2019

```
knitr::opts_chunk$set(echo = TRUE)
```

Computation

1. Calculate Manahattan, Canberra, and Euclidean distances "by hand" (i.e., create the data, program each line, and make the calculations). What are the values for each measure?

```
# create the data
p <- c(1, 2)
q <- c(3, 4)

# calculate the Manahattan distance
man_dis <- 0
for (i in seq(1:length(p))) {
    man_dis <- man_dis + abs(p[i] - q[i])
}
cat("Manahattan distance is:", man_dis, "\n")
```

```
## Manahattan distance is: 4
```

```
# calculate the Canberra distance
can_dis <- 0
for (i in seq(1:length(p))) {
    can_dis <- can_dis + abs(p[i] - q[i]) / (abs(p[i]) + abs(q[i]))
}
cat("Canberra distance is:", can_dis, "\n")
```

```
## Canberra distance is: 0.8333333
```

```
# calculate the Euclidean distance
euc_dis <- 0
for (i in seq(1:length(p))) {
    euc_dis <- euc_dis + (p[i] - q[i])^2
}
euc_dis <- sqrt(euc_dis)
cat("Euclidean distance is:", euc_dis, "\n")
```

```
## Euclidean distance is: 2.828427
```

2. Use the dist() function in R to check your work. Were you right or wrong? (be honest in your reporting). If wrong, after debugging, where and why did you go wrong?

```
# create the matrix
x <- rbind(p, q)

# calculate the Manahattan distance
m_dist <- dist(x, method = "manhattan")
cat("Manahattan distance is:", m_dist, "\n")
```

```
## Manahattan distance is: 4
```

```r
# calculate the Canberra distance
c_dist <- dist(x, method = "canberra")
cat("Canberra distance is:", c_dist, "\n")
```

```
## Canberra distance is: 0.8333333
```

```r
# calculate the Euclidean distance
e_dist <- dist(x, method = "euclidean")
cat("Euclidean distance is:", e_dist, "\n")
```

```
## Euclidean distance is: 2.828427
```

3. What are the key differences between these measures, and why does it matter? How might you see these differences "in action" with these fictitious data?

Manahattan distance sums up the absolute distances in each dimension for a pair of objects. Therefore, it is always larger than or equal to the Euclidean distance, which can be conceptualized as the linear distance between two points (sums up the squared distances in each dimension for a pair of objects and takes the square root of the result). Canberra distance is a weighted version of the Manahattan distance (sum up the fraction differences between coordinates of a pair of objects). It is very sensitive to a small change when both coordinates are nearest to zero. Manahattan distance has the lowest complexity of calculation and least expansive in terms of computing, while the Euclidean distance has the highest complexity among the three. For an extremely large dataset, it would be easier to perform calculation with the Manahattan distance. The differences discussed above determines how the distance between data points get calculated and demonstated and hence affect our choice of distance measures that suit data with different size and density. In these fictitious data, we can see these differences reflected in the magnitude of distance values: the Manahattan distance is much larger than the rest two, while the Canberra distance is much smaller than the other ones.

4. Use some basic EDA techniques to present and discuss the data (e.g., visualize, describe in multiple ways, etc.)

```r
library(skimr)
```

```
##
## Attaching package: 'skimr'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

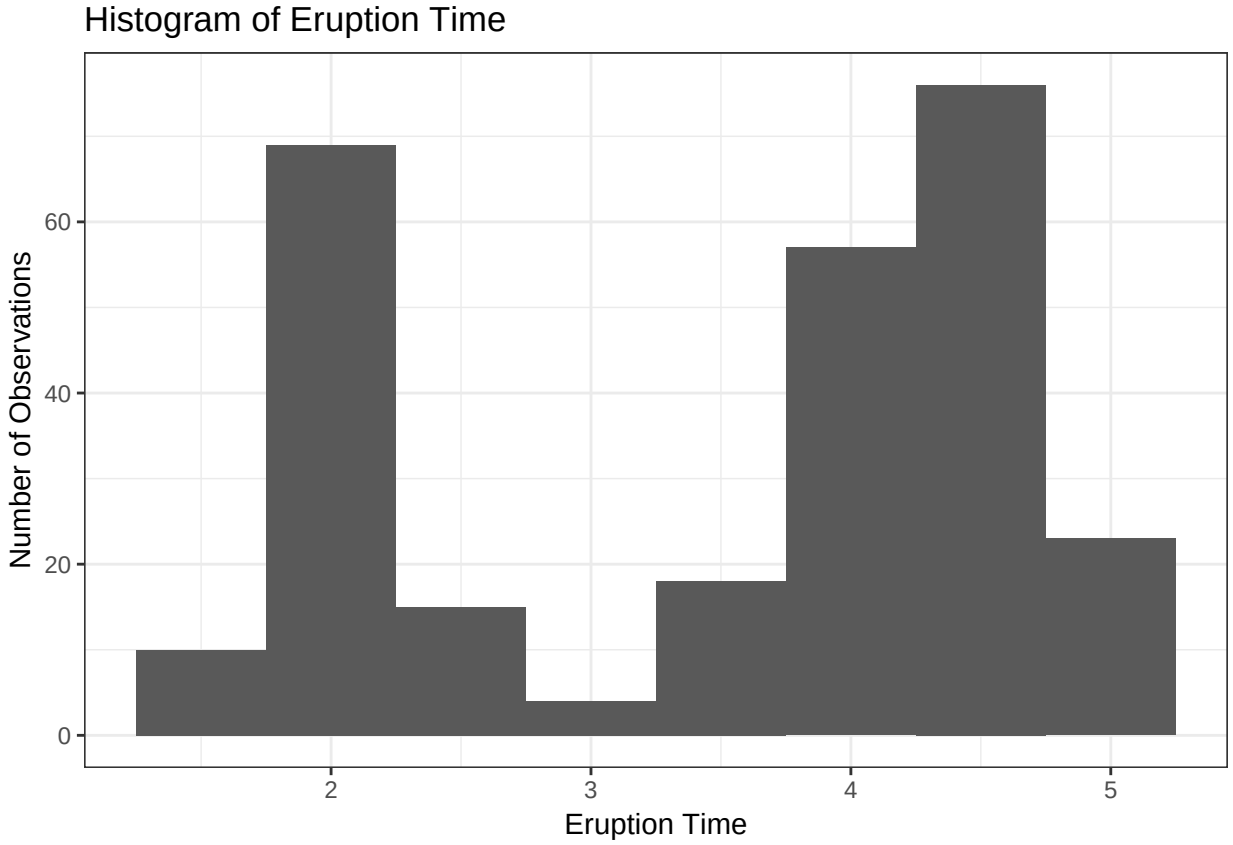
```r
# read in data
of <- faithful
#summarize data
skim(of)
```

```
## Skim summary statistics
##  n obs: 272
##  n variables: 2
##
## -- Variable type:numeric ------------------------------------------------------------------------
##   variable missing complete   n  mean    sd   p0   p25 p50   p75 p100
##  eruptions       0      272 272  3.49  1.14  1.6  2.16   4  4.45  5.1
##    waiting       0      272 272 70.9  13.59 43   58     76 82    96
##      hist
##
##
```

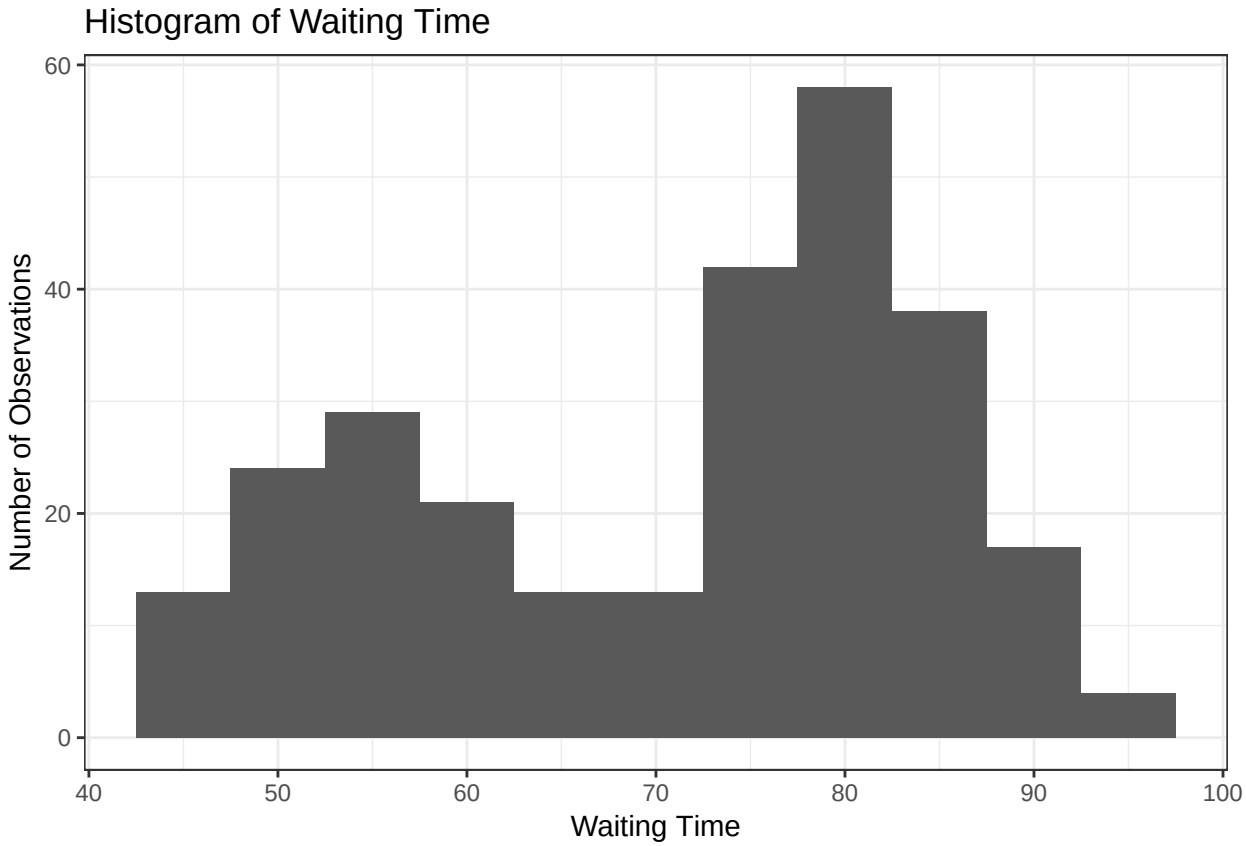
Judging from the standard deviation, the variation of eruption time is much smaller than that of waiting time. The average eruption time is 3.49 minutes, while the average waiting time is 70.9 minutes.

```
library(ggplot2)

# histogram
ggplot(data = of) +
  geom_histogram(aes(x = eruptions), binwidth = 0.5) +
  labs(x = "Eruption Time",
       y = "Number of Observations",
       title = "Histogram of Eruption Time") +
  theme_bw()
```

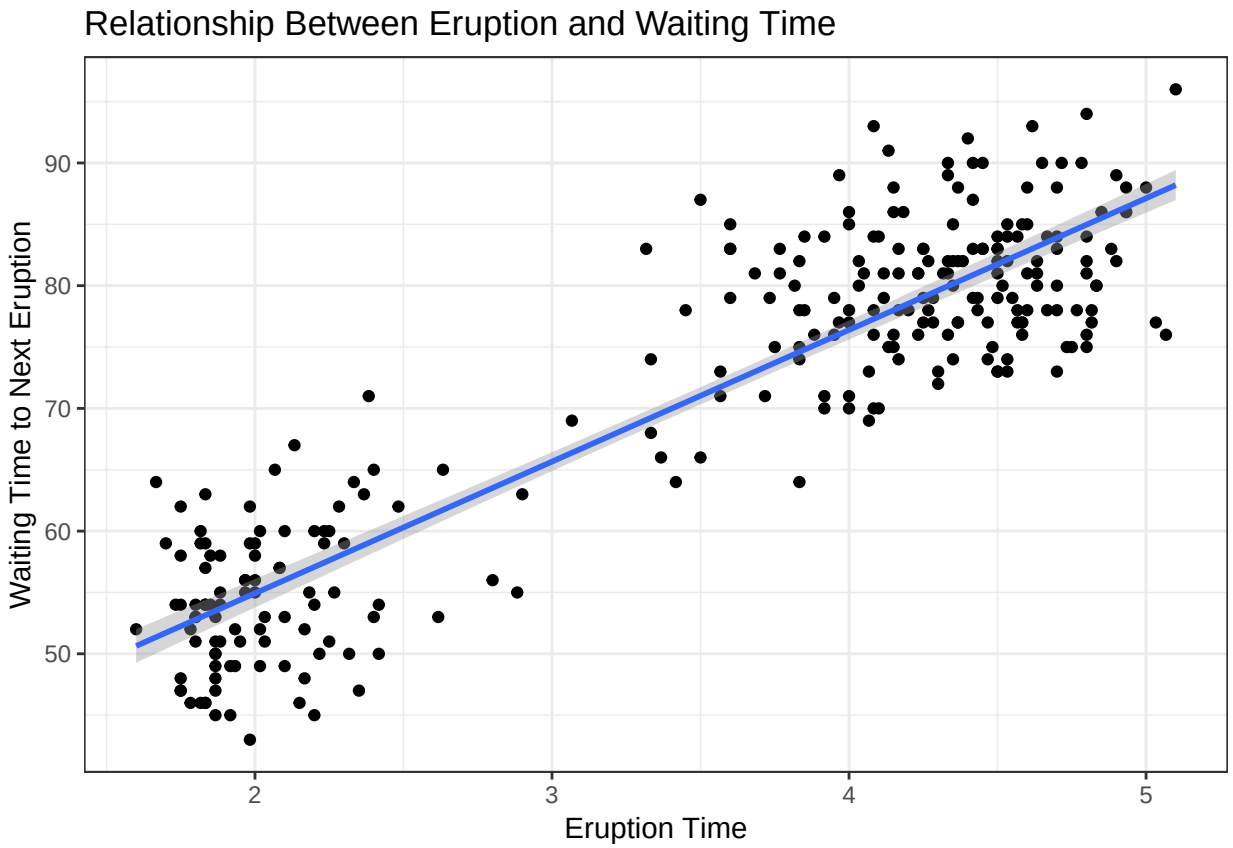



```
ggplot(data = of) +
  geom_histogram(aes(x = waiting), binwidth = 5) +
  labs(x = "Waiting Time",
       y = "Number of Observations",
       title = "Histogram of Waiting Time") +
  theme_bw()
```



The distribution of both eruption time and waiting time demonstrates some sort of "U" shape—there are more observations with small and large values, while fewer in the middle.

```
# scatter plot
ggplot(of, aes(x = eruptions, y = waiting)) +
  geom_point() +
  geom_smooth(method = lm) +
  labs(x = "Eruption Time",
       y = "Waiting Time to Next Eruption",
       title = "Relationship Between Eruption and Waiting Time") +
  theme_bw()
```



Overall, longer waiting time is correlated with longer eruption time. More importantly, there seems to be two clusters defined by waiting time and eruption time: one cluster that contains observations with short waiting time and eruption time; the other cluster that contains observations with long waiting time and eruption time.

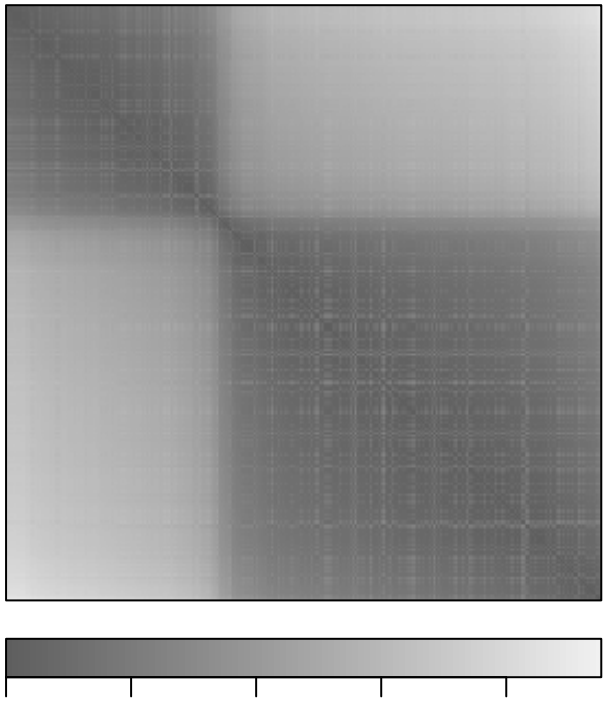
5. Calculate a dissimilarity matrix of these data.

```
# scale and calculate (and store) distance matrix
of_scaled <- scale(of)
of_dist <- dist(of_scaled, method = "euclidean")
```

6. Generate an ODI for the Old Faithful data. What do you see?

```
library(seriation)

# generate ODI
dissplot(of_dist)
```



The ODI shows two clusters defined by all variables in the old faithful dataset: one on the top left, which probably corresponds to the cluster of observations with short waiting and eruption time in the previous scatter plot; the other on the lower right, which probably corresponds to the cluster of observations with long waiting and eruption time in the previous scatter plot.

7. Using any munging tools you'd like (e.g., dplyr from the Tidyverse), create a subset of the data excluding the species feature, scaling the features, and calculating a dissimilarity matrix (think "pipe" for stacking functions to do this quickly, e.g.).

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------- t
```

```
## v tibble  2.0.1       v purrr   0.3.0
## v tidyr   0.8.2       v dplyr   0.8.0.1
## v readr   1.3.1       v stringr 1.4.0
## v tibble  2.0.1       v forcats 0.4.0
```

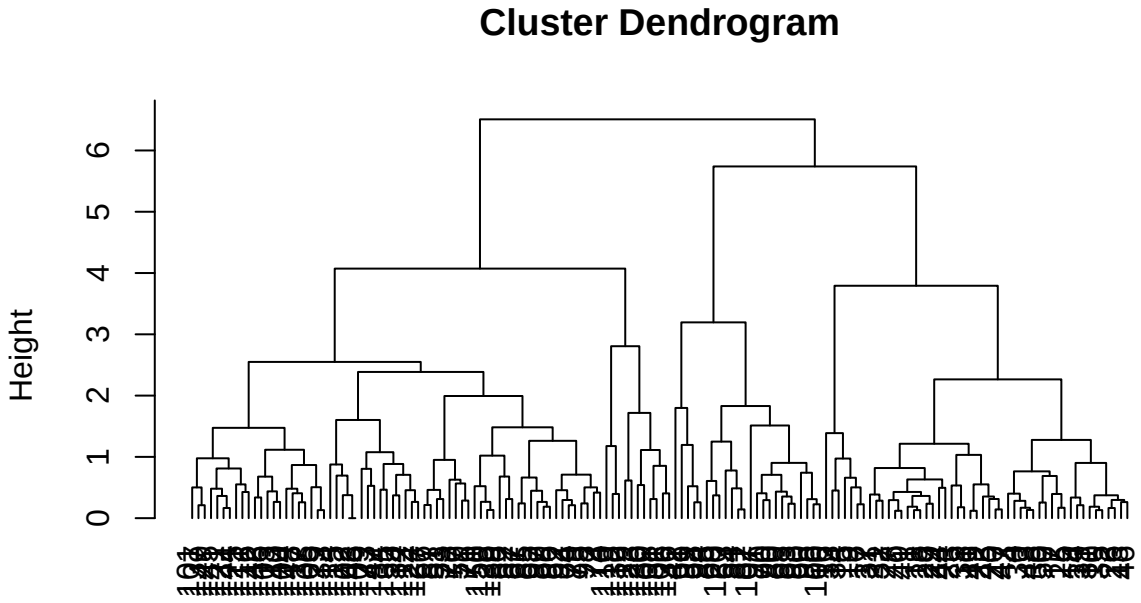
```
## -- Conflicts ------------------------------------------------------------------------------- tidyve
## x dplyr::filter() masks skimr::filter(), stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
data(iris)

# create a subset of the data excluding the species feature, scale the features and calculate distance
iris_sub <- iris %>%
  dplyr::select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
  scale() %>%
  dist()
```

8. Fit an agglomerative hierarchical clustering algorithm using complete linkage on your subset data and render the dendrogram of clustering results. What do you see?

```
hc_complete <- hclust(iris_sub,
                      method = "complete"); plot(hc_complete, hang = -1)
```



The 150 observations form 4 quite distinct clusters with different scales, which are further fussed into two major clusters. As the indexes of observations 1-150 do not denote any meaning substantively, I could not reach any substantively meaningful conclusion with this dendrogram. I might be able to make some substantive arguments if each observation is labeled with their species here. In that case I can possibly (1) examine if observations from each species are clustered together; (2) examine which species are more similar to each other.

9. Try cutting the tree at 2 and 3 branches and show these trees side-by-side. How do they differ?

```
library(dendextend)
```

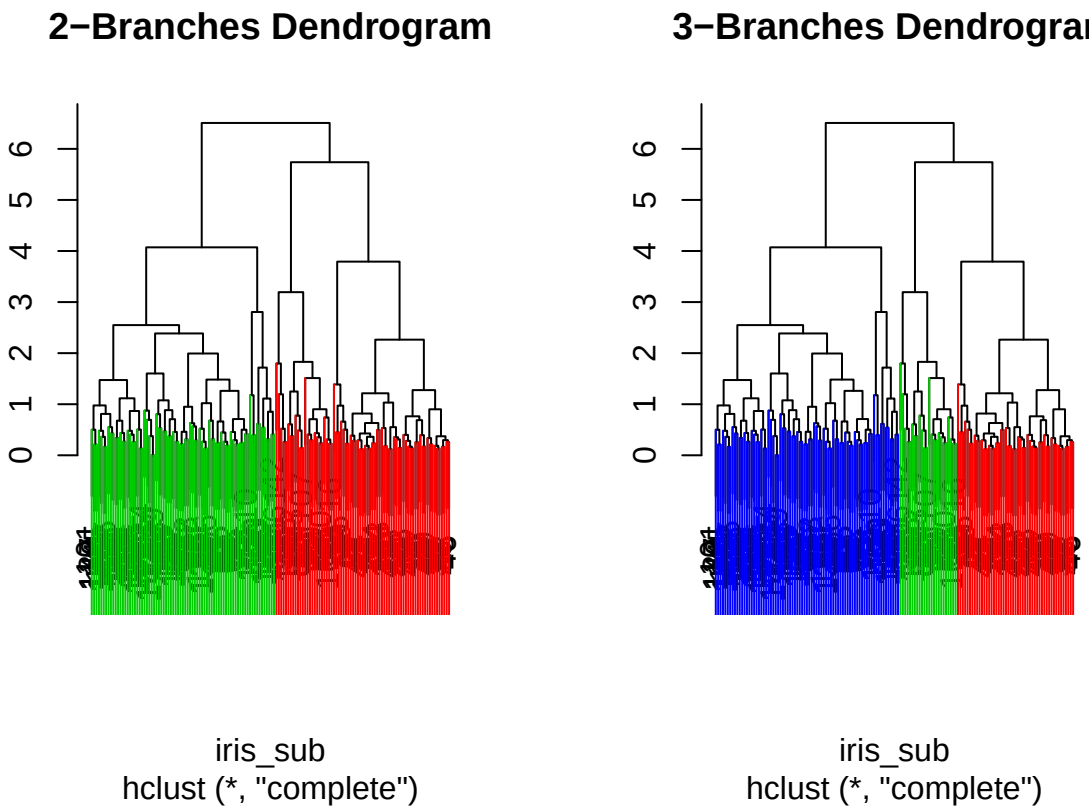
```
##
## ---------------------
## Welcome to dendextend version 1.12.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------

##
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(sparcl)

# cutting the tree at 2 and 3 branches and show these trees side-by-side
par(mfrow = c(1,2))
y = cutree(hc_complete, 2)
ColorDendrogram(hc_complete, y = y, labels = 1:150 , main = "2-Branches Dendrogram",
    branchlength = 80)
y = cutree(hc_complete, 3)
ColorDendrogram(hc_complete, y = y, labels = 1:150 , main = "3-Branches Dendrogram",
    branchlength = 80)
```



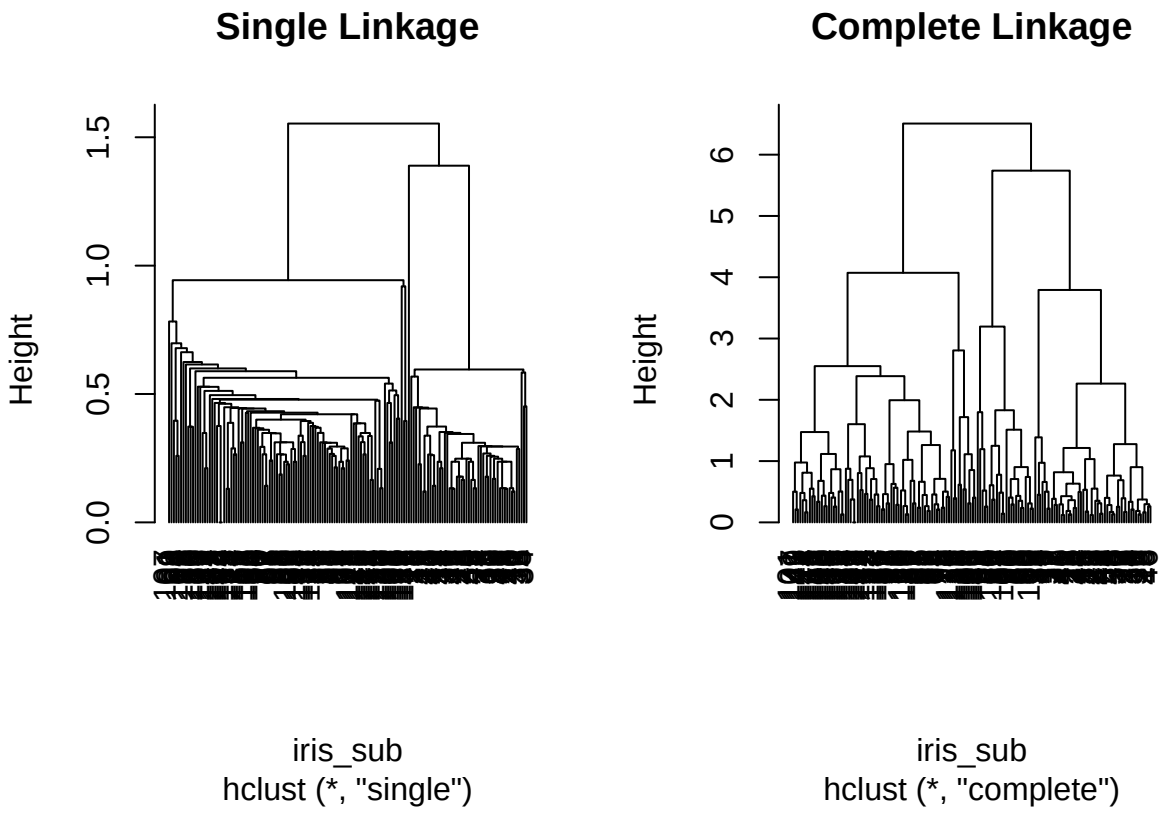

```
# show the difference
cuts <- cutree(hc_complete, k = c(2,3))
table(`2 Clusters` = cuts[,1],
      `3 Clusters` = cuts[,2])
```

```
##            3 Clusters
## 2 Clusters  1  2  3
##          1 49 24  0
##          2  0  0 77
```

The dendrograms and the table show that the first cluster remains the same for both trees. However, in the 2-branches dendrogram, the second and third cluster of the 3-branches dendrogram are fussed into one cluster.

10. Now fit the algorithm using single and complete linkage and present each dendrogram side-by-side. Discuss the differences. What effects can we see in the clustering patterns when using different linkage methods?

```
par(mfrow = c(1,2))
hc_single <- hclust(iris_sub,
                    method = "single"); plot(hc_single, hang = -1, main="Single Linkage")
hc_complete <- hclust(iris_sub,
                      method = "complete"); plot(hc_complete, hang = -1, main="Complete Linkage")
```



The single linkage method seems to fuse one observation at a time, resulting in longer branches and stringy-type clusters. The complete linkage seems to present a more balanced and prettier picture. Ultimately, the broadest pattern (in terms of 2-branch tree) identified by the two methods are similar (a bit different regarding the cluster-assignment for the data points in the middle).

Critical Thinking

1. You just assessed the clusterability of some feature space, R". Address the following questions:

a. How would you go about determining whether clustering made sense to consider or not?

I can explore the feature space to diagnose clusterability.

b. What are techniques you would use, and what might you be looking for from each?

(1) Simple distribution plots: I can create scatterplots for all observations according to certain features and see if the data points cluster in certain ways.
(2) VAT/ODI plots: I can calculate the distance matrix and construct VAT/ODI plot based on it. Clusters are indicated by dark blocks of pixels along the diagonal.
(3) Hopkins Statistic: I can use the Hopkins Statistic to mathenmatically test the the null hypothesis of spatial randomness in the data using a sparse sampling test. In general, if the Hopkins Statistic is above 0.5, we could say the data are non-random and hence "clusterable".

c. How might these techniques work together to motivate clustering or not?

If explorations based on all these techniques denote the same signal, for instance, both the scatter plot and

the VAT plot show same amount and shape of clusters and the Hopkins Statistic is above 0.5, we find good initial support for clusterability. In this case, we have grounds to proceed with clustering explorations.

d. And ultimately, can/should you proceed if you find little to no support for clusterability? Why or why not?

I don't think I should proceed if I find little to no support for clusterability through various means and combination of features. I understand that by selecting different sets of features and using different distance measures to calculate distance matrix, we might end up with very different outcomes in terms of clusterability. Therefore I definitely need to explore with a great many of possible situations before coming to an argument that there is little to no support for clusterability. But ultimately if I arrive at that point, I should not proceed with clustering for (1) this method does not make sense to the data; (2) this method could be computationally consuming yet there's very low probability to produce insightful analyses.

2. Locate (and read) a paper that applies the hierarchical agglomerative clustering technique. Address the following questions:

a. Describe the author(s) process.

Paper: "Segmentation of Expository Texts by Hierarchical Agglomerative Clustering" (https://arxiv.org/abs/cmp-lg/9709015)

The paper proposes a segementation method for expository texts based on hierarchical agglomerative clustering technique, which is able to identify a hierarchical discourse structure. The author uses the common lexical cohesion metrics as distance measure to fit a hierarchical agglomerative clustering algorithm on the Stargazers article, an expository text that discusses the conditions fore evolution of life in solar systems. Using paragraph as the basic unit of analsis for the HAC algorithm, the author chooses a slightly different operation than the general HAC applications: at each stage the author computes the similarity of the segment only to its two neighbors rather than all of the rest clusters in order to preserve the linear order in the text. Based on the dendrogram created by HAC, the author develops an algorithm to detect boundary between texts. The boundary detection judegement rests on two features of the dendrogram: the segment size (the number of leaves, i.e. paragraphs, it contains) and segment depth (the longest path in the subtree from the root to the leaves). Through the application of the notch rule and the cliff rule, Linear segmentation can be induced from the identified hierachical discourse structure. The evaluation of the proposed HAC segmentation algorithm is conducted through comparison with the segmentation results produced by Human judgements and by Hearst's TextTiling algorithm, which is genrally accepted as robust. The results of HAC matches quite well with the rest two.

b. Do they go through similar steps as we covered this week both in setting the stage for clustering (e.g., assessing clusterability, calculating distance, etc.), as well as in fitting the algorithm? If not, what did they omit and does this omission impact their findings in your opinion?

The article demonstrate all steps except for the clusterability diagnosis and the linkage method selection parts: data standardization, distance measure selection, algorithm fitting, pattern inspection through visualization and validation checking through comparing with other methods. The omitted clusterability diagnosis part does not really influence the validity of the finding, as there is a whole validation section that provides sufficient evidence to support the choice of HAC here. Yet though the results look good, the findings would be more scientifically solid if it is reproducible to readers. To acheive reproducibility, a first step should be clearly describing the whole analysis process, including specifying the choice of linkage method.

c. Describe at least one possible extension from the study that could emerge based on their findings.

The findings show that HAC identifies the necessary structure information for content analysis/comprehension (the segmentation of text). Based on these findings, we can seek to further identify the topics or titles for the segments identified by HAC.