hw2

Arun Pandian

10/18/2019

Computation

1.

Calculating distance manually:

```
## ('Euclidean distance', 2.8284271247461903)
```

```
## ('Canberra distance', 0.8333333333333333)
```

```
## ('Manhattan distance', 4.0)
```

2.

Calculating distances using dist():

```
## Euclidean distance: 2.82842712474619
```

```
## Canberra distance: 0.833333333333333
```

```
## Manhattan distance: 4
```

The distances are the about the same barring slight differences in floating point arithmetic between python and R. One thing to note is that in python, the vectors are initialized as vectors of real numbers so that, they aren't treated as integers during division(eg. 1/2 will evaluate to 0, which is not desirable).

3.

The differences in the distances to use might be related to domain preferences as well as the characteristics of the data under question. In general, Euclidean distances may not be appropriate for small fractions as squaring them may lead to issues with floating point arithmetic. Likewise for Canberra distances, the same problem may apply. In those cases, Manhattan distances maybe more appropriate.

If sensitivity for very small values is a priority in clustering, the Canberra distance can be used(because of the denominator). Conversely, if sensitivity for high values is important then Euclidean and Manhattan distances may be more appropriate(although Euclidean distance requires squaring which may lead to overflow errors/more intensive calculations). This is because, Canberra with it's weighting factor will impede sensitivity towards big numbers.

Between Manhattan and Euclidean distances, the Manhattan distance is a natural fit, if the data under question is grid based data and it doesn't make sense to talk of Euclidean distances(eg. Chess board moves, Walking blocks in a city). Additionally, Euclidean distances may add further weight to dimensions with more variability (because of the squaring), and so could be more appropriate for low variance data-sets.

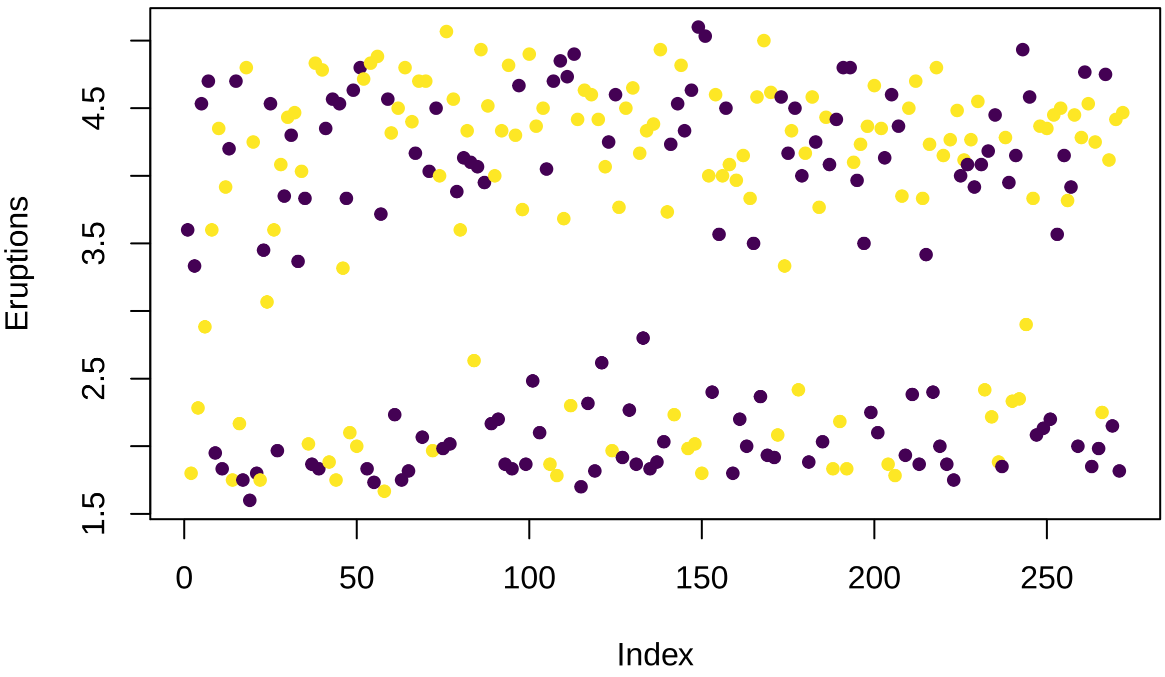
4.

```r
library(viridis)
```

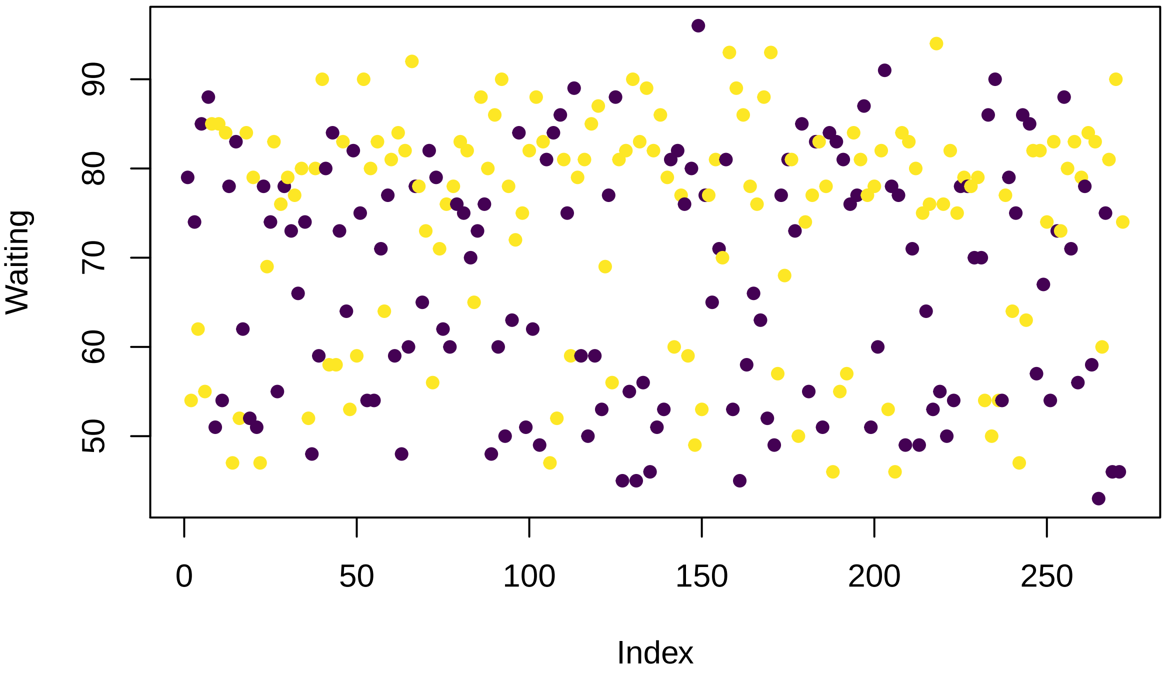
```
## Loading required package: viridisLite
```

```r
library(ggplot2)
```

```
x <- faithful
plot(x$eruptions, ylab='Eruptions', col=viridis(2), pch =19, cex = 0.8)
```

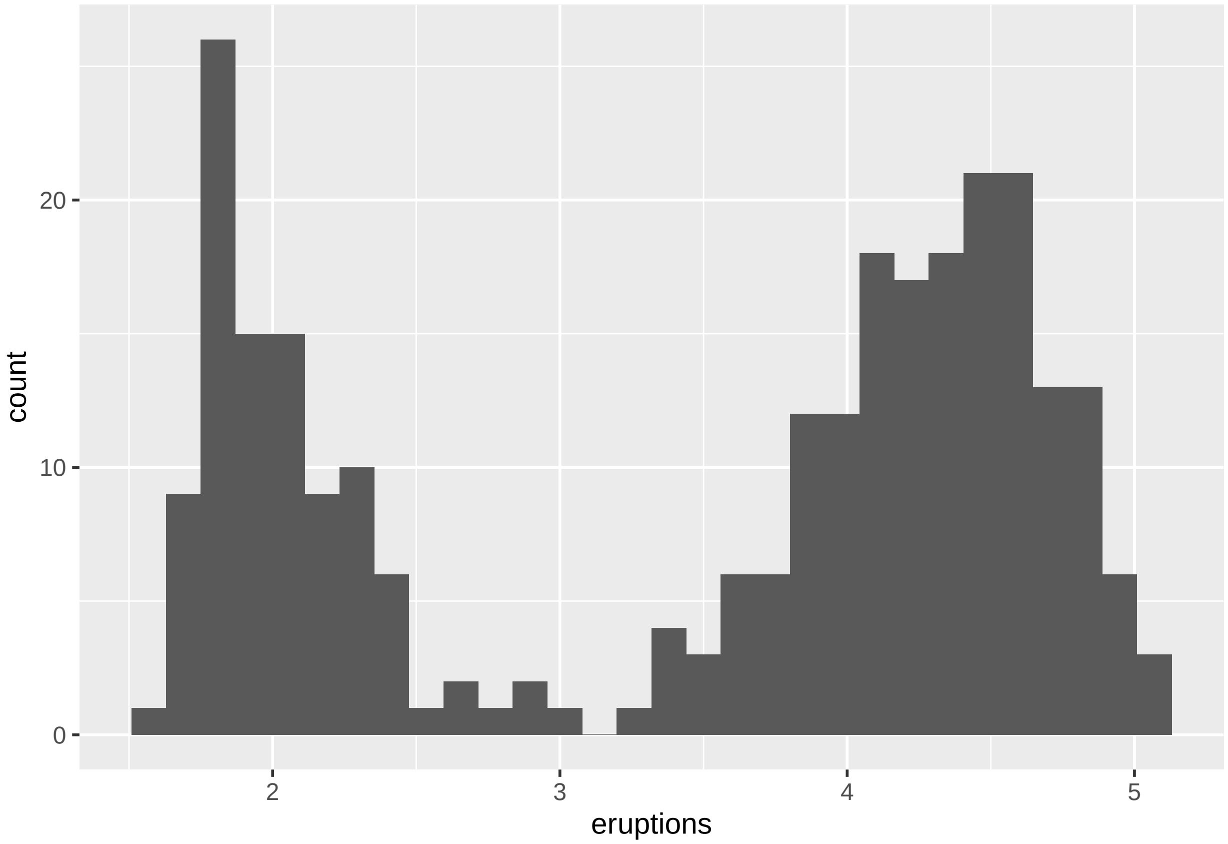


```
plot(x$waiting, ylab='Waiting', col = viridis(2), pch = 19, cex = 0.8)
```



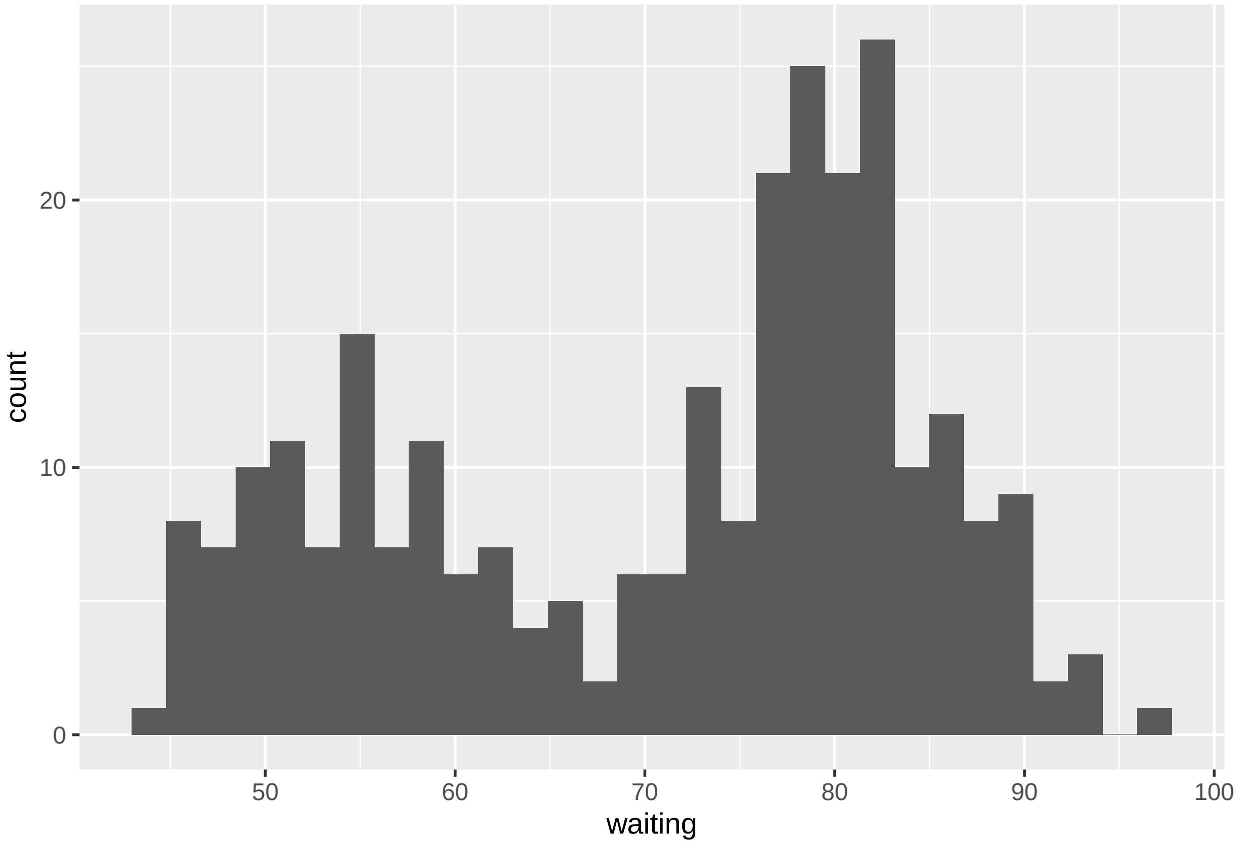
```
ggplot(x, aes(x=eruptions)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

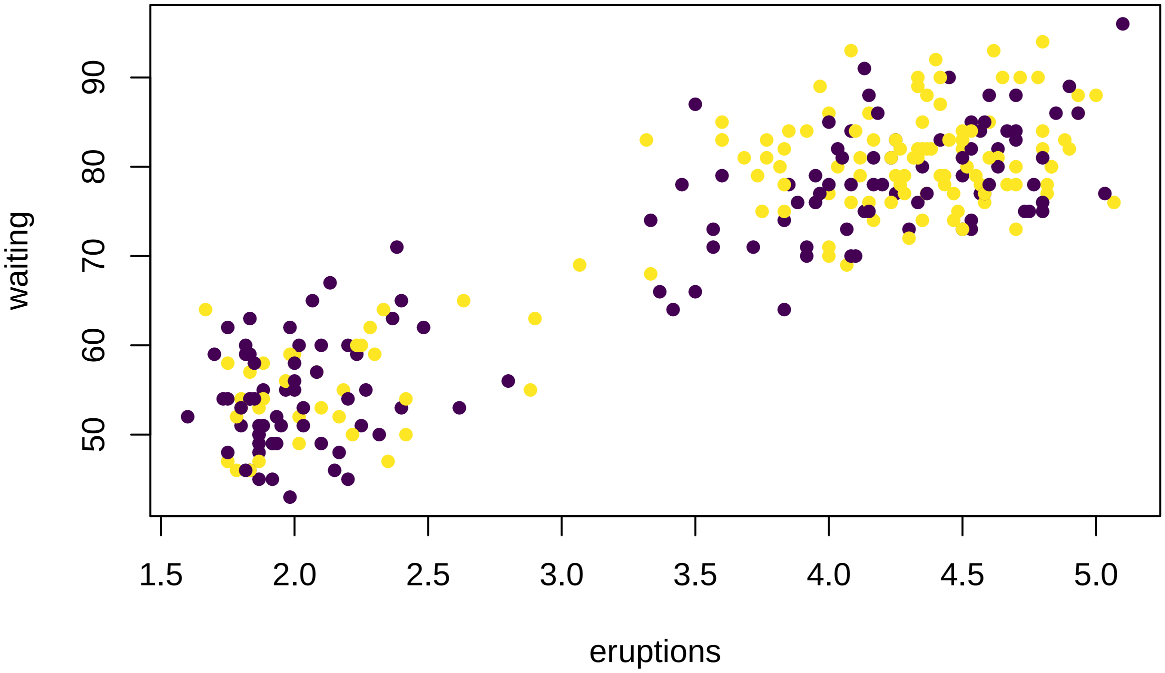


```
ggplot(x, aes(x=waiting))+geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
plot(x,col = viridis(2), pch = 19, cex = 0.8)
```



Looking at these plots, it seems like the data-set is clusterable. In the plot for Eruptions, there seems to be two natural groupings one on above y = 3.5 and another below. Likewise in the plot for Waiting, there seems to be two groups of symmetric shapes above and below y = 70. Looking at the histograms of both eruptions and waiting time, it seems like there are normal distributions centered around two means quite far apart.

Looking at the overall 2 dimensional plot, there seems to be two dense clusters at a distance from each other, and the waiting times seems correlated to eruptions. This suggests that the data is clusterable.

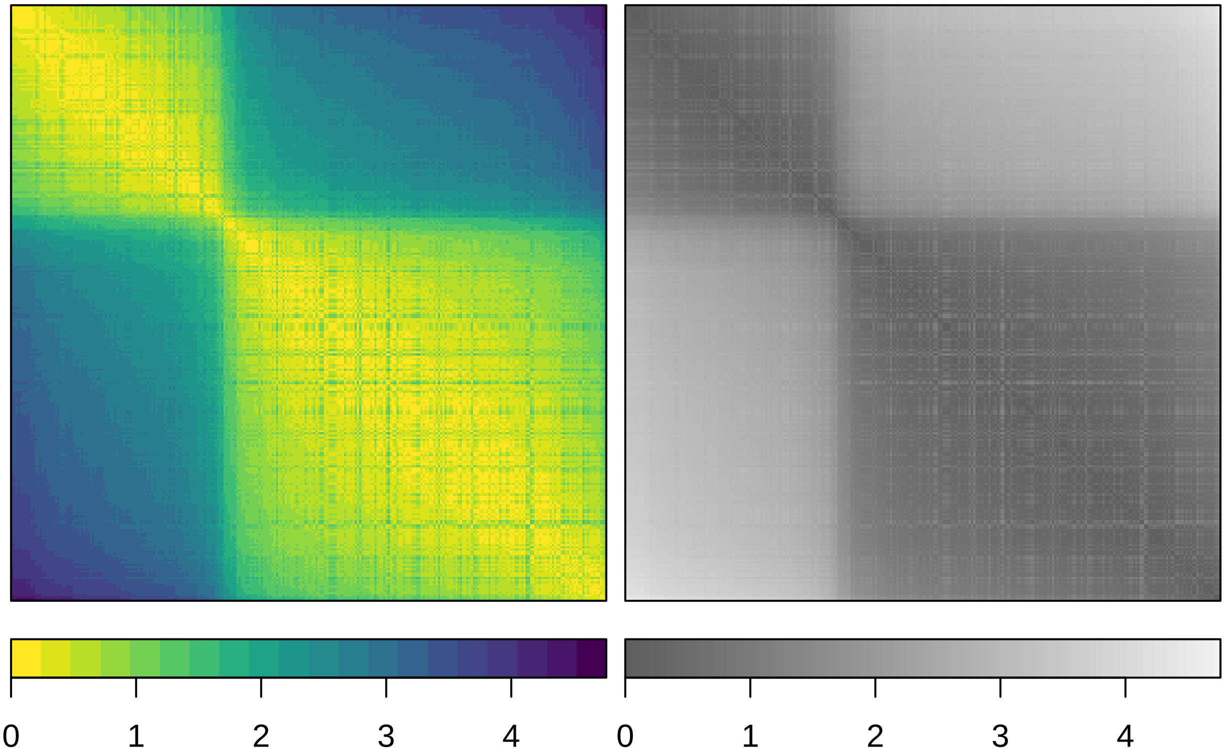
5.

Calculating dissimilarity matrix of the data after scaling the data and using euclidean distance since there seems to be no great many outliers to the visual groupings above:

```r
x_scaled <- scale(x)
x_dist <- dist(x_scaled,
               method = "euclidean")
```

6. Generating ODI matrix of the data, in two colorings:

Black/White as well as Viridis coloring which is popular since it's known to be easily interpretable for humans when using color plots(even those with certain degrees of color blindness).



It seems from the above two images that the data-set is clusterable and there are two natural groups from the very black two squares along the diagonal.

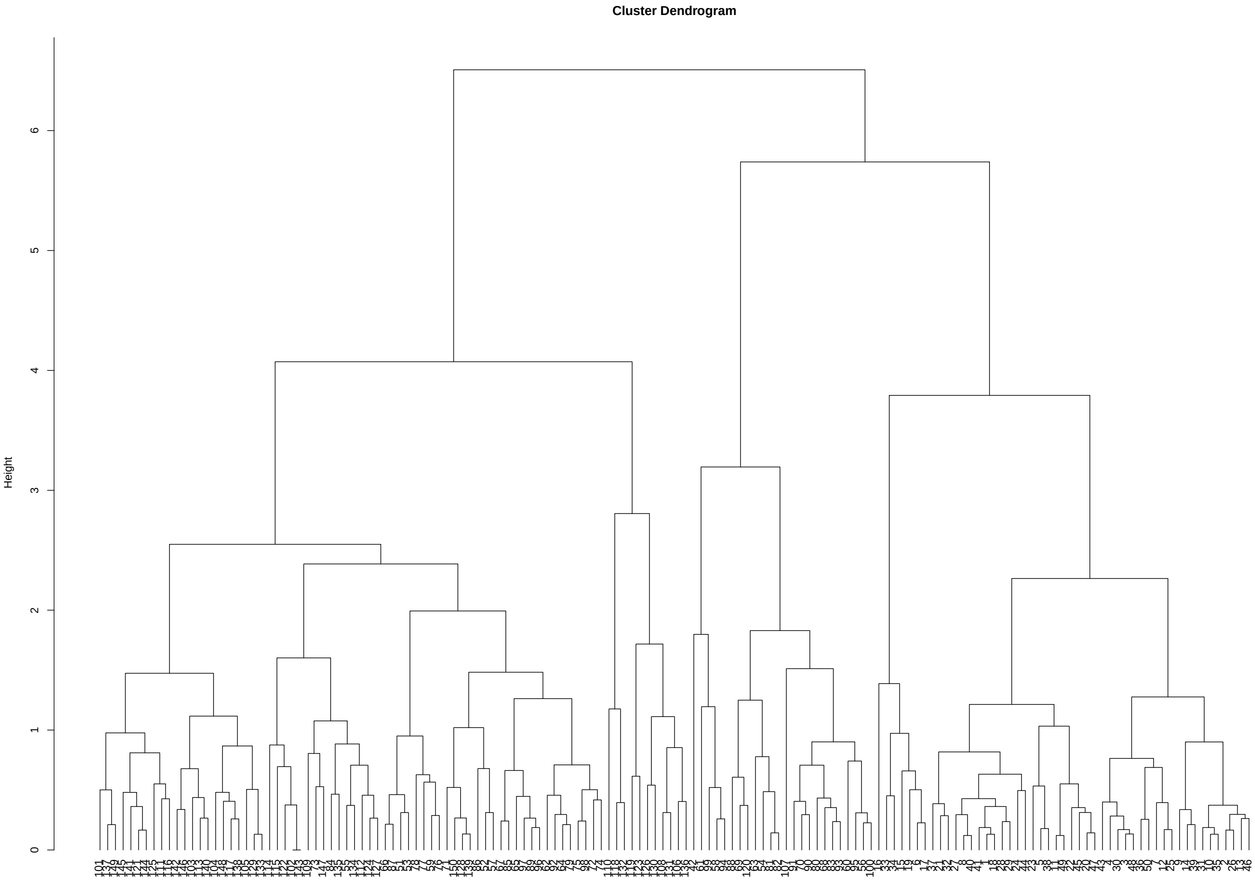
7.

```r
suppressPackageStartupMessages(library(tidyverse))
library(dplyr)
data <- iris
# exclude species column, scale and create dissimiliarity matrix
distance_matrix <- scale(iris[, -5])%>% dist(,
               method = "euclidean")
```

8.

Fitting HAC using complete linkage:

```
suppressPackageStartupMessages(library(dendextend))  # for "cutree" function
hc_complete <- hclust(distance_matrix, method="complete");plot(hc_complete,hang=-1,xlab="", sub="")
```

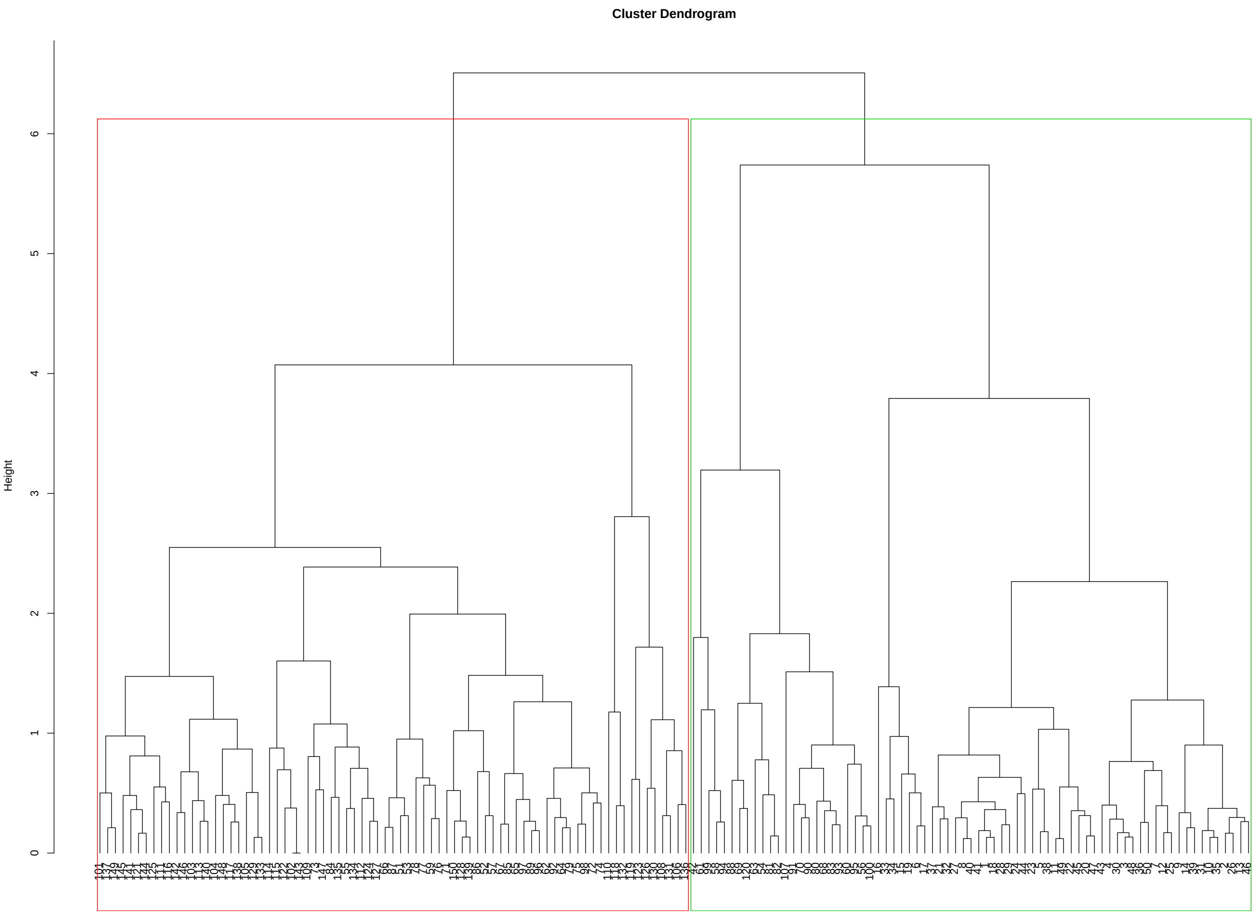


The major groupings emerge around height 4. Amongst the three clusters we know there are, two types(under topmost right clade) are closer to each other than the third(under the topmost left clade), based on this dendogram.

9.

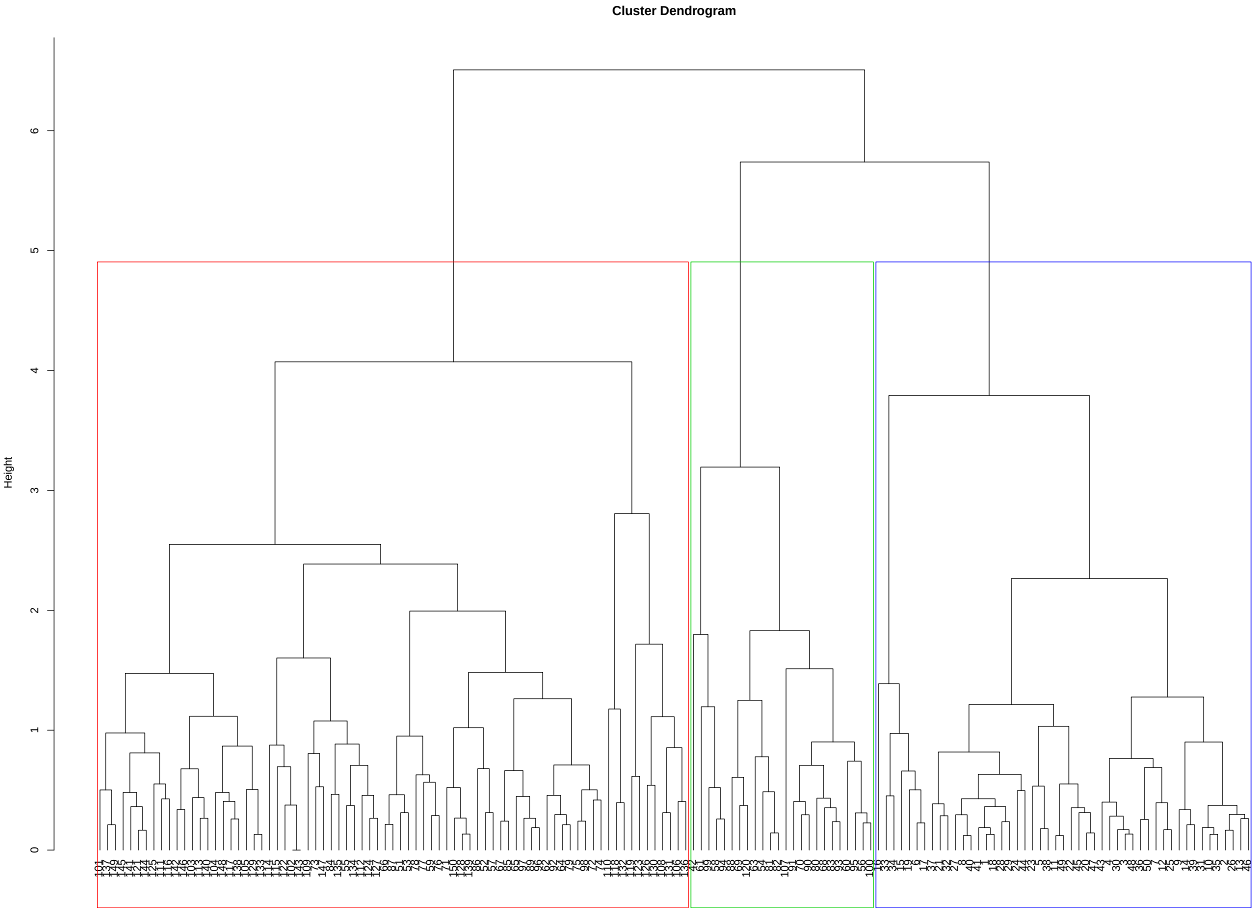
Cutting the dendogram at height 2:

```
#tried to remove the numbers at the bottom but didn't work
plot(hc_complete, xlab=NA, sub=NA, hang=-1)
rect.hclust(hc_complete, k = 2, border = 2:3)
```

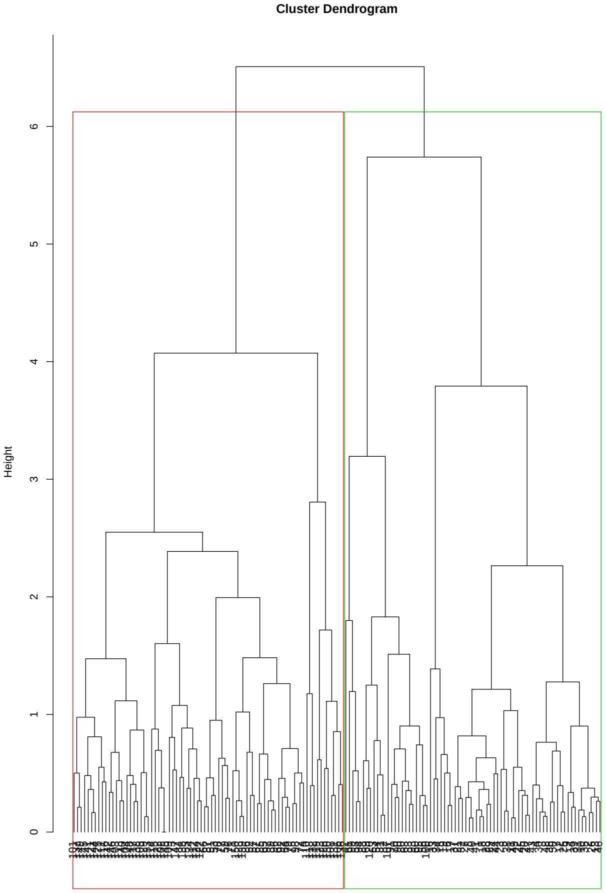


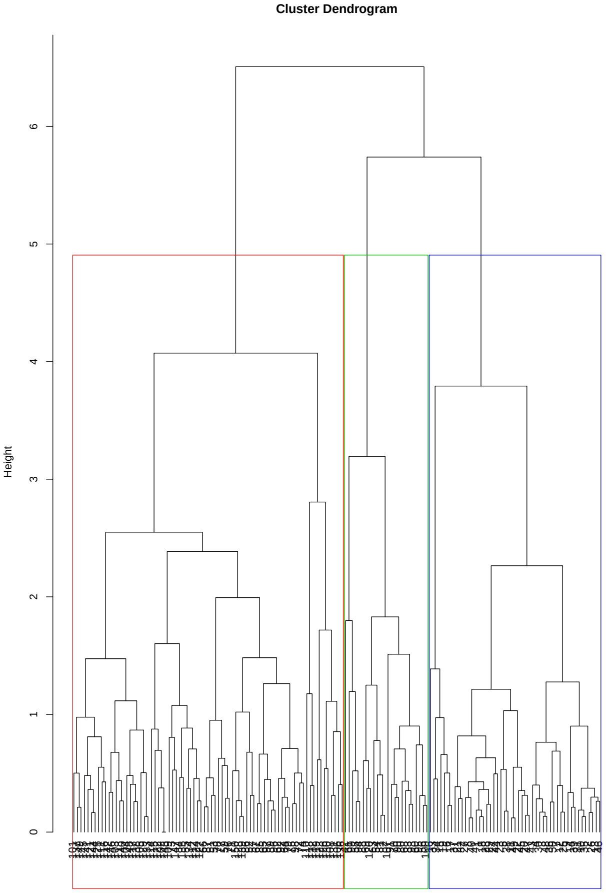
Cutting the dendogram at height 3:

```r
#tried to remove the numbers at the bottom but didn't work
plot(hc_complete, xlab="", sub="", hang =-1)
rect.hclust(hc_complete, k = 3, border = 2:5)
```



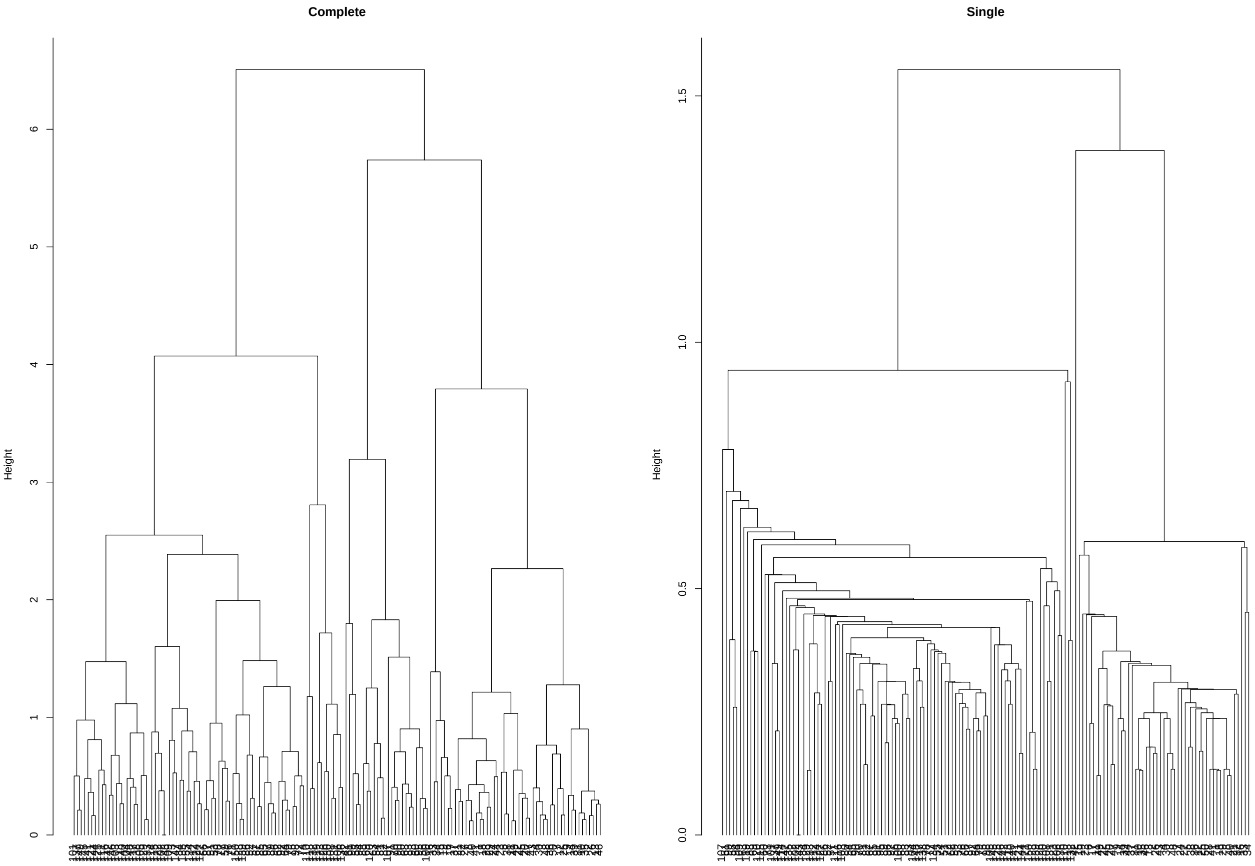
```
par(mfrow = c(1,2))
plot(hc_complete, xlab=NA, sub=NA, hang=-1)
rect.hclust(hc_complete, k = 2, border = 2:3)
plot(hc_complete, xlab=NA, sub=NA, hang=-1)
rect.hclust(hc_complete, k = 3, border = 2:4)
```





The dendogram cut into three seems a better fit for the data. Cutting the dendogram into two, means that one of the clusters has two major clusters below it which is formed around the same time as the cluster under the other clade. Cutting the dendogram into three clusters gives clusters that are formed around the same time. However, as mentioned above, two types(under topmost right clade) are closer to each other than the third(under the topmost left clade), based on this dendogram.

10.



Complete link clustering seems to have more parity in the size of clusters formed at a certain level, but there are too many points/clusters added at a much later time in single link clustering. This could be related to the compressed height in single link clustering(1.6 vs 6 in complete link). In the single link clustering, although it converges to clusters under similar circumstances as the complete link, the structure of the sub-clusters below are different and not concordant with our knowledge of the iris data-set, where two of the three types are similar. Part of the problem is that a lot of times, points seem to be added to a cluster, much after they are formed.

Critical Thinking

1. a)

Questions of clustering becomes relevant when we would like to know if there is an underlying structure to the data under question or would like to reduce the dimensionality of the data space. We do not necessarily know what the constituents of the structure will correspond to. In other words, unlike classification, the different possible labels are not known. We also need a measure of similarity between the different objects in the cluster. For eg. Clustering may not make sense for categorical data.

1. b)

EDA can be used initially to diagnose clusterability(both numerically and visually).

Visually, we can plot certain dimensions of the data-set (based on domain knowledge/intuition) and see if groupings exist. We can compare these plots to see if clustering will make sense. We can then visualize the entire data-set by reducing the dimensions to 2 or 3 using dimensionality reduction methods.

For instance we can check visually if, there is any grouping in terms of distance between the groups, if different groups have different densities or is there a difference in shape between partitions of the data-set.

ODI plots can be used which would ascertain if any groupings exist in the data, based on the provided distance function. The number of darker blocks along the diagonal is a measure of groupings in the data whose constituents that are quite "close" and the lighter blocks representing groups whose constituents are far apart.

Numerically, we can calculate the Hopkins test statistic (H) which tests if the data is the result of a uniform distribution (with no natural clustering), or not. Here again, we need to use the distance function. We conclude that the data is not from a uniform distribution and could be clusterable, if the Hopkins statistic is great than 0.5

1. c)

The initial visual analysis of distribution plots may be insightful and if the ODI plots correspond to the informal analysis, we are more certain to use clustering. If the initial visual analysis does not seem favorable to clustering, we may still use ODI plots, as visualizing distributions of reduced forms of high dimensional may lead to artifacts that obscure the underlying structure/groupings. However, even after applying ODI with an appropriate distance measure, if the data-set doesn't seem clusterable, caution should be applied in clustering the data. This is especially so, considering that clustering algorithms will usually return clusters, even if the underlying data is random and doesn't have natural groupings.

1. d)

If the data doesn't seem clusterable, it could be that adding an appropriate dimension to the data-set could make the data clusterable. Additionally, a more appropriate distance function could be found to the data-set. However if there is still low evidence for clusterability, it's better not to use clustering algorithms since, as mentioned above, clustering algorithms will return clusters even if the underlying data has not natural groupings/clusters.

2. a)

Reference: Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event co-reference resolution. Transactions of the ACL, 3:517–528. I will focus on a paper (Yang, 2015) that solves event-co-reference resolution using hierarchical distance-dependent clustering. Event co-reference is the problem of resolving different spans of text that refer to the same event into a cluster. For eg., consider the following sentence:

"Hilary Clinton as the Secretary of State spearheaded the Obama administration's support of the military-led coup in Honduras, which led to the predictable consequence of (. . . .) even though this violation of the Honduran constitution was opposed by most of Central and South Americas who also objected to the unilateral action by the United States. . . ".

In the sentence above, "military-led coup" and "which", "this violation of the Honduran constitution" refers to the same coup. This these spans of text would belong to the same cluster. Other clusters would contain other such co-reference spans of text("support", "unilateral action").

To this end, first they extract spans of the text that could refer to events. Then they cluster these events to get event-co-references. The distance between coherent events should be high and non-co-referent events be small for this clustering scheme to work. Interestingly, the learn the distance function during training. Their distance function takes as input, two events at a time. The events are featurized as word vectors along with other semantic and syntactic information. The distance function is then a (putatively) non-linear function of the features, as it's learning using Deep Learning that learns to provide the optimal (gold-standard) clustering. Once they have this pairwise distance function, they can use it to cluster their events.

To get this clustering, the authors use a non-parametric Hierarchical clustering model (the number of clusters being the unspecified parameter). The clustering is hierarchical because the clustering happens in two steps. In the first, they resolve event co-reference in the same document. In the second, the use this information to form event-co-reference cluster across documents. For eg. to resolve mention of the coup in one document (d1) to the mention of the coup in another document (d2), once they have resolved the event co-references within the same document.

This direction makes sense because within-document event co-reference is simpler than cross-document co-reference for obvious reasons.

2. b)

It's clear that, the problem of event co-reference can be transformed into a clustering problem and these clusters of events do exist. They assume that they would be able to learn this non-trivial distance function by using their training data. They also assume that this function can be learnt by the very specific deep learning model they use and that this function will generalize to unseen data, which are standard supervised learning assumptions. Because of this framework, they don't spend anytime at all on diagnosing clusterability.

2. c)

At the time this paper was written, it achieved state of the art results for event co-reference. This suggests that Hierarchical modelling was a good fit for this problem. However, later on, in our work published in CICLING (a popular NLP conference), we showed that we can achieve better results by completely avoiding clustering. We did however use some of the distance function features suggested in this paper but instead of clustering, we formed links between pairs of events that are very close by our enriched distance function and avoid forming links between events that aren't close. This suggests that, the distance function used in this paper, did contain information about the similarity of events and in general such functions can be calculated to capture complex semantic relationships in text.