workshop07

February 3, 2019

```
In [1]: library(tidyverse)
        library(ggrepel)

        # Plot size deppening on your screen resolution to 5 x 3
        options(repr.plot.width=4, repr.plot.height=4)
```

```
 Attaching packages  tidyverse 1.2.1
 ggplot2 2.2.1      purrr   0.2.5
 tibble  1.4.2      dplyr   0.7.5
 tidyr   0.8.1      stringr 1.3.1
 readr   1.1.1      forcats 0.3.0
 Conflicts  tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()
```


1 Welcome to Workshop 7

We first consider hierarchical clustering. Later we will consider k means clustering. Either procedure involves:
* If using numerical variables, adjusting for different scales by standardisation * Choosing a between-observations distance measure appropriate to the kind of variables, * Calculating a distance matrix for the distance between observations

For hierarchical clustering, also * Choosing a way of calculating the distance between clusters (This calculation uses the between-observations distance matrix.) * Obtain a hierarchical structure of nested clusters from which the appropriate number of clusters can be decided upon, and those clusters read off.

1.0.1 Exercise 1

Load the file `KTC.csv` into `ktc.df` and print the first rows. KTC (Know Thy Customer) is a financial advising company that provides personalised financial advice to its clients (see Camm et al, page 256.) KTC would like to segment its customers into several groups (clusters) so that the customers within a group are similar in certain key characteristics, and are dissimilar to customers who are not in the group.

```
In [2]: ktc.df <- read.table(file = "KTC.csv",
                            header = TRUE,
```

```
                          sep = ",")
ktc.df %>% head()
```

ID	Age	Female	Income	Married	Children	CarLoan	Mortgage
1	48	1	17546	0	1	0	0
2	40	0	30085	1	3	1	1
3	51	1	16575	1	0	1	0
4	23	1	20375	1	3	0	0
5	57	1	50576	1	0	0	0
6	57	1	37870	1	2	0	0

Clearly there are three numerical variables, and four binary variables. Ideally, all variables are used together to create the clusters, and this can be done in R, using the metric gower in the command daisy in the package cluster. It is important when mixing data types, the relative magnitudes of the data are comparable and weightings are chosen accordingly. We will keep things a little bit simpler by using all binary variables or all numerical variables. We will not be doing any examples where the data types are mixed.

Install the package `ade4` and `ggdendro` into the local directory and load the libraries.

```
In [3]: install.packages('ade4',lib='.', verbose=TRUE)
        library(ade4,lib.loc='.')
        install.packages('ggdendro',lib='.', verbose=TRUE)
        library(ggdendro,lib.loc='.')
```



```
system (cmd0): /usr/lib/R/bin/R CMD INSTALL
foundpkgs: ade4, /tmp/RtmpNQuLql/downloaded_packages/ade4_1.7-13.tar.gz
files: /tmp/RtmpNQuLql/downloaded_packages/ade4_1.7-13.tar.gz
1): succeeded '/usr/lib/R/bin/R CMD INSTALL -l '/srv/home/whtam4' /tmp/RtmpNQuLql/downloaded_pac
system (cmd0): /usr/lib/R/bin/R CMD INSTALL
foundpkgs: ggdendro, /tmp/RtmpNQuLql/downloaded_packages/ggdendro_0.1-20.tar.gz
files: /tmp/RtmpNQuLql/downloaded_packages/ggdendro_0.1-20.tar.gz
1): succeeded '/usr/lib/R/bin/R CMD INSTALL -l '/srv/home/whtam4' /tmp/RtmpNQuLql/downloaded_pac
```

Select all binary columns plus the `ID` field and store it into `ktcBinary.df`. The `ID` needs to be stored as a row name with `column_to_rownames(var = 'ID')`

```
In [10]: ktcBinary.df<- ktc.df %>%
         select(ID,Female,Married,CarLoan,Mortgage)%>%
         column_to_rownames(var = 'ID')
         ktcBinary.df
```

Female	Married	CarLoan	Mortgage
1	0	0	0
0	1	1	1
1	1	1	0
1	1	0	0
1	1	0	0
1	1	0	0
0	0	0	0
0	1	1	0
1	1	1	0
0	1	1	0
1	1	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	1	0	1
1	1	1	1
1	1	0	1
1	1	0	1
1	1	0	0
0	1	1	0
0	1	0	0
0	1	0	0
0	1	0	0
1	0	1	1
0	0	1	0
0	1	1	1
0	1	0	1
1	1	0	0
1	0	1	1
0	1	0	0

Calculate the simple matching distance with:

```
In [11]: d2<- dist.binary(ktcBinary.df, method = 2)
         d2
```

```
           1         2         3         4         5         6         7
2  1.0000000
3  0.7071068 0.7071068
4  0.5000000 0.8660254 0.5000000
5  0.5000000 0.8660254 0.5000000 0.0000000
6  0.5000000 0.8660254 0.5000000 0.0000000 0.0000000
7  0.5000000 0.8660254 0.8660254 0.7071068 0.7071068 0.7071068
8  0.8660254 0.5000000 0.5000000 0.7071068 0.7071068 0.7071068 0.7071068
9  0.7071068 0.7071068 0.0000000 0.5000000 0.5000000 0.5000000 0.8660254
10 0.8660254 0.5000000 0.5000000 0.7071068 0.7071068 0.7071068 0.7071068
11 0.5000000 0.8660254 0.5000000 0.0000000 0.0000000 0.0000000 0.7071068
12 0.7071068 0.7071068 0.7071068 0.8660254 0.8660254 0.8660254 0.8660254
```

```
13 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.8660254
14 0.8660254 0.5000000 0.5000000 0.7071068 0.7071068 0.7071068 1.0000000
15 0.8660254 0.5000000 0.8660254 0.7071068 0.7071068 0.7071068 0.7071068
16 0.8660254 0.5000000 0.5000000 0.7071068 0.7071068 0.7071068 1.0000000
17 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.8660254
18 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.8660254
19 0.5000000 0.8660254 0.5000000 0.0000000 0.0000000 0.0000000 0.7071068
20 0.8660254 0.5000000 0.5000000 0.7071068 0.7071068 0.7071068 0.7071068
21 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.5000000
22 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.5000000
23 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.5000000
24 0.7071068 0.7071068 0.7071068 0.8660254 0.8660254 0.8660254 0.8660254
25 0.7071068 0.7071068 0.7071068 0.8660254 0.8660254 0.8660254 0.5000000
26 1.0000000 0.0000000 0.7071068 0.8660254 0.8660254 0.8660254 0.8660254
27 0.8660254 0.5000000 0.8660254 0.7071068 0.7071068 0.7071068 0.7071068
28 0.5000000 0.8660254 0.5000000 0.0000000 0.0000000 0.0000000 0.7071068
29 0.7071068 0.7071068 0.7071068 0.8660254 0.8660254 0.8660254 0.8660254
30 0.7071068 0.7071068 0.7071068 0.5000000 0.5000000 0.5000000 0.5000000
```



```
           8         9        10        11        12        13        14
2
3
4
5
6
7
8
9  0.5000000
10 0.0000000 0.5000000
11 0.7071068 0.5000000 0.7071068
12 0.8660254 0.7071068 0.8660254 0.8660254
13 0.8660254 0.7071068 0.8660254 0.5000000 0.7071068
14 0.7071068 0.5000000 0.7071068 0.7071068 0.5000000 0.5000000
15 0.7071068 0.8660254 0.7071068 0.7071068 0.8660254 0.5000000 0.7071068
16 0.7071068 0.5000000 0.7071068 0.7071068 0.5000000 0.5000000 0.0000000
17 0.8660254 0.7071068 0.8660254 0.5000000 0.7071068 0.0000000 0.5000000
18 0.8660254 0.7071068 0.8660254 0.5000000 0.7071068 0.0000000 0.5000000
19 0.7071068 0.5000000 0.7071068 0.0000000 0.8660254 0.5000000 0.7071068
20 0.0000000 0.5000000 0.0000000 0.7071068 0.8660254 0.8660254 0.7071068
21 0.5000000 0.7071068 0.5000000 0.5000000 1.0000000 0.7071068 0.8660254
22 0.5000000 0.7071068 0.5000000 0.5000000 1.0000000 0.7071068 0.8660254
23 0.5000000 0.7071068 0.5000000 0.5000000 1.0000000 0.7071068 0.8660254
24 0.8660254 0.7071068 0.8660254 0.8660254 0.0000000 0.7071068 0.5000000
25 0.5000000 0.7071068 0.5000000 0.8660254 0.7071068 1.0000000 0.8660254
26 0.5000000 0.7071068 0.5000000 0.8660254 0.7071068 0.7071068 0.5000000
27 0.7071068 0.8660254 0.7071068 0.7071068 0.8660254 0.5000000 0.7071068
28 0.7071068 0.5000000 0.7071068 0.0000000 0.8660254 0.5000000 0.7071068
29 0.8660254 0.7071068 0.8660254 0.8660254 0.0000000 0.7071068 0.5000000
30 0.5000000 0.7071068 0.5000000 0.5000000 1.0000000 0.7071068 0.8660254
```

```
          15         16         17         18         19         20         21
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16 0.7071068
17 0.5000000 0.5000000
18 0.5000000 0.5000000 0.0000000
19 0.7071068 0.7071068 0.5000000 0.5000000
20 0.7071068 0.7071068 0.8660254 0.8660254 0.7071068
21 0.5000000 0.8660254 0.7071068 0.7071068 0.5000000 0.5000000
22 0.5000000 0.8660254 0.7071068 0.7071068 0.5000000 0.5000000 0.0000000
23 0.5000000 0.8660254 0.7071068 0.7071068 0.5000000 0.5000000 0.0000000
24 0.8660254 0.5000000 0.7071068 0.7071068 0.8660254 0.8660254 1.0000000
25 0.8660254 0.8660254 1.0000000 1.0000000 0.8660254 0.5000000 0.7071068
26 0.5000000 0.5000000 0.7071068 0.7071068 0.8660254 0.5000000 0.7071068
27 0.0000000 0.7071068 0.5000000 0.5000000 0.7071068 0.7071068 0.5000000
28 0.7071068 0.7071068 0.5000000 0.5000000 0.0000000 0.7071068 0.5000000
29 0.8660254 0.5000000 0.7071068 0.7071068 0.8660254 0.8660254 1.0000000
30 0.5000000 0.8660254 0.7071068 0.7071068 0.5000000 0.5000000 0.0000000
          22         23         24         25         26         27         28
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

```
19
20
21
22
23 0.0000000
24 1.0000000 1.0000000
25 0.7071068 0.7071068 0.7071068
26 0.7071068 0.7071068 0.7071068 0.7071068
27 0.5000000 0.5000000 0.8660254 0.8660254 0.5000000
28 0.5000000 0.5000000 0.8660254 0.8660254 0.8660254 0.7071068
29 1.0000000 1.0000000 0.0000000 0.7071068 0.7071068 0.8660254 0.8660254
30 0.0000000 0.0000000 1.0000000 0.7071068 0.7071068 0.5000000 0.5000000
          29
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30 1.0000000
```

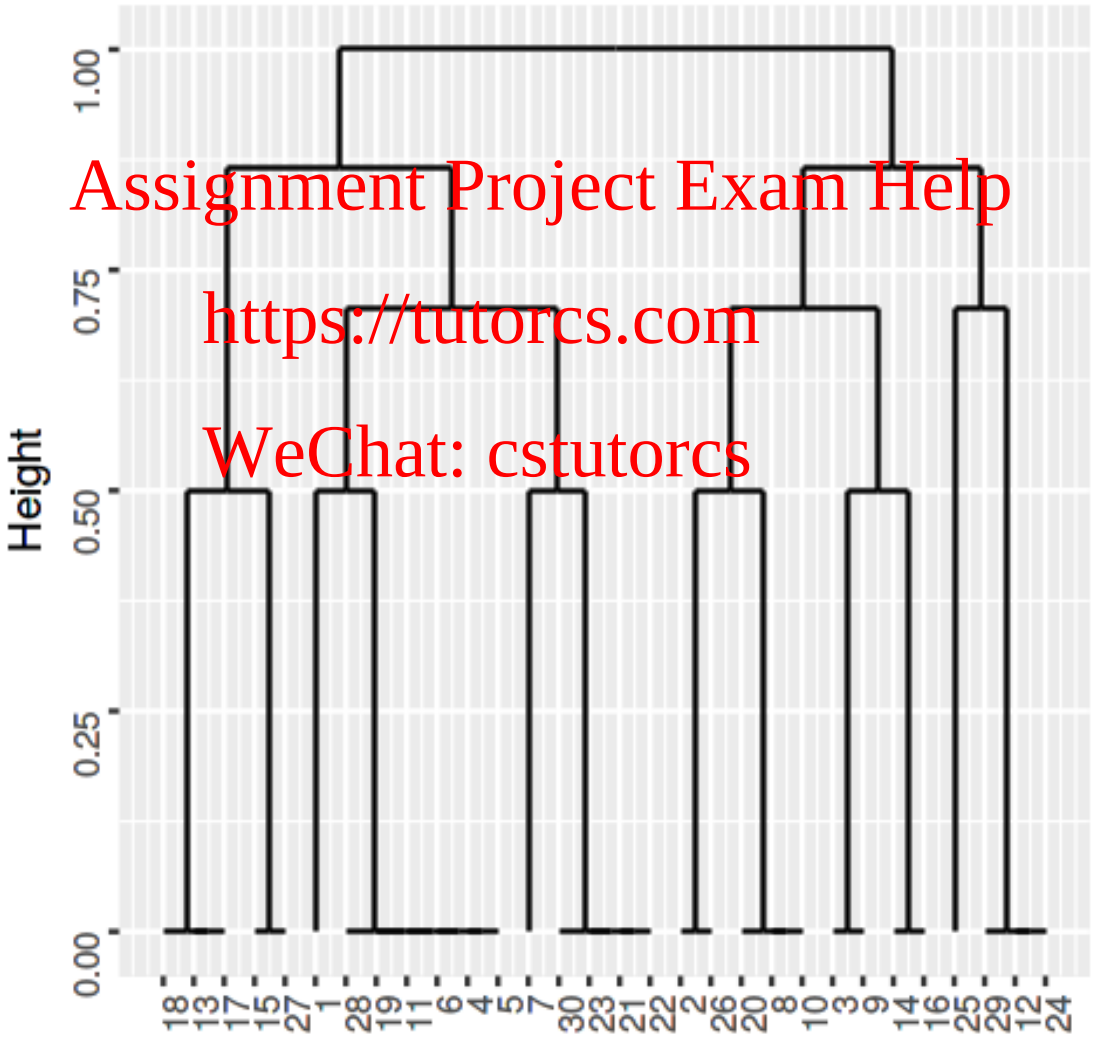

The method chosen can be anything from 1 to 10. We will restrict ourselves to method 1 (Jaccard) and method 2 (Simple Matching). I have named the distance d2 to indicate it was calculated by method 2. Thus in dist.binary, the "method" is the method of calculating the distance between two observations. dist.binary can be found in ade4.

Now this distance matrix (distances between observations) is to be fed into the hierarchical clustering command hclust, and we need to choose a method of measuring the distance between clusters. There are seven methods available. The ones of interest to us are "single", "complete", "average", and "centroid" as discussed in the lecture. On this occasion we use "average" and call the hierarchical clustering result hca

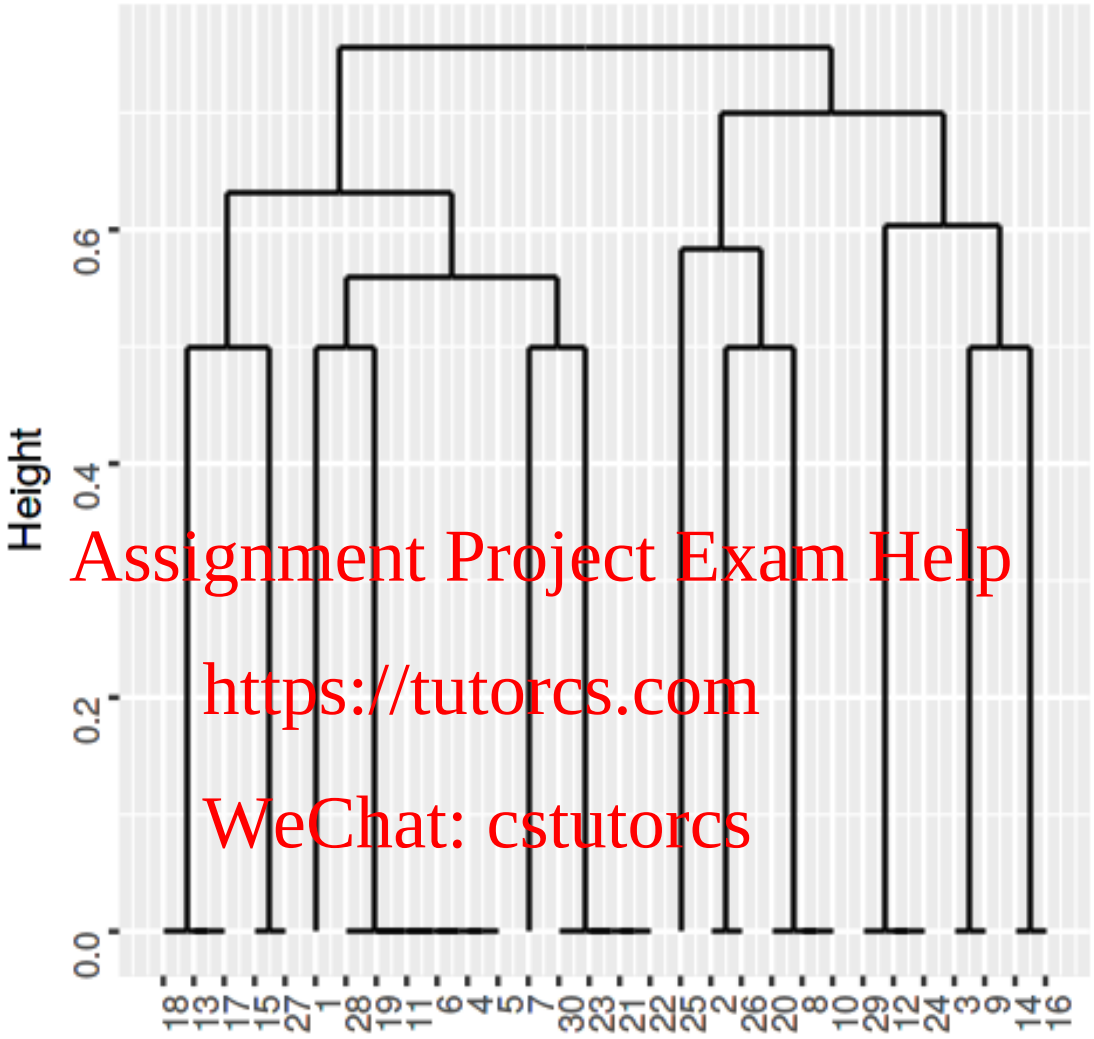
```
In [16]: hca<-hclust(d2, method = "average")
```

Let's plot the dendrogram (as this creates our usual ggplot object, we can use the standard commands for labelling):

```
In [15]: ggdendrogram(hca,theme_dendro =FALSE) +
             ylab("Height") +
             xlab('')
```



```
In [17]: ggdendrogram(hca,theme_dendro =FALSE) +
             ylab("Height") +
             xlab('')
```



So now we have obtained a dendrogram which tells us possible ways of forming clusters. This is fine for a small data set such as the current one. We can also use the cutree command which will cut a dendrogram into clusters. We can specify how many clusters we want, for example four clusters (k = 4):

```
In [23]: a4 <- cutree(hca, k=4)
```

or the height at which the tree should be cut, for example h = 0.6. Notice that this results in 5 clusters.

```
In [24]: a5 <- cutree(hca, h=0.6)
```

Add the columns a4 and a5 to the data frame ktc.df

In [25]:
```
ktc.df<- data.frame(ktc.df)
ktc.df$a4<-a4
ktc.df$a5<-a5
ktc.df
```

ID	Age	Female	Income	Married	Children	CarLoan	Mortgage	a4	a5
1	48	1	17546	0	1	0	0	1	1
2	40	0	30085	1	3	1	1	2	2
3	51	1	16575	1	0	1	0	3	3
4	23	1	20375	1	3	0	0	1	1
5	57	1	50576	1	0	0	0	1	1
6	57	1	37870	1	2	0	0	1	1
7	22	0	8877	0	0	0	0	1	1
8	58	0	24947	1	0	1	0	2	2
9	37	1	25304	1	2	1	0	3	3
10	54	0	24212	1	2	1	0	2	2
11	66	1	59804	1	0	0	0	1	1
12	52	1	26659	0	0	1	1	3	4
13	44	1	19736	1	1	0	1	4	5
14	66	1	55205	1	1	1	1	3	3
15	36	0	19475	1	0	0	1	4	5
16	38	1	22842	1	0	1	1	3	3
17	37	1	17730	1	2	0	1	4	5
18	46	1	41016	1	0	0	1	4	5
19	62	1	26909	1	0	0	0	1	1
20	31	0	22243	1	3	1	0	2	2
21	61	0	57881	1	2	0	0	1	1
22	50	0	16497	1	2	0	0	1	1
23	54	0	38447	1	0	0	0	1	1
24	27	1	15539	0	0	1	1	3	4
25	22	0	12640	0	2	1	0	2	2
26	56	0	41034	1	0	1	1	2	2
27	45	0	20810	1	0	0	1	4	5
28	39	1	20114	1	1	0	0	1	1
29	39	1	29359	0	3	1	1	3	4
30	61	0	24270	1	1	0	0	1	1

In [26]:
```
ktc.df %>%
    #select(Age,Female,Income,Married,Children,CarLoan,Mortgage,a5)%>%
        select(-ID,-a4) %>%
        group_by(a5) %>%
        summarise_all(mean)
```

a5	Age	Female	Income	Married	Children	CarLoan	Mortgage
1	50.00000	0.5833333	31597.17	0.8333333	1.000000	0	0.0000000
2	43.50000	0.0000000	25906.83	0.8333333	1.166667	1	0.3333333
3	48.00000	1.0000000	29856.50	1.0000000	0.750000	1	0.5000000
4	39.33333	1.0000000	23852.33	0.0000000	1.000000	1	1.0000000
5	41.60000	0.6000000	22953.40	1.0000000	0.600000	0	1.0000000

You should get a print out like: | a5 | Age | Female | Income | Married | Children | CarLoan | Mortgage | |—-|———-|———-|———-|———-|———-|———-|———-| | 1 | 50.00000 | 0.5833333 | 31597.17 | 0.8333333 | 1.000000 | 0 | 0.0000000 | | 2 | 43.50000 | 0.0000000 | 25906.83 | 0.8333333 | 1.166667 | 1 | 0.3333333 | | 3 | 48.00000 | 1.0000000 | 29856.50 | 1.0000000 | 0.750000 | 1 | 0.5000000 | | 4 | 39.33333 | 1.0000000 | 23852.33 | 0.0000000 | 1.000000 | 1 | 1.0000000 | | 5 | 41.60000 | 0.6000000 | 22953.40 | 1.0000000 | 0.600000 | 0 | 1.0000000 |

Questions: * Also calculate standard deviations (replace mean by sd). * How do you interpret the averages of the binary variables? * For each of the five clusters, examine the values of the binary variables, and describe what characteristics the cluster seems to have. * For example, consider the first cluster. On average those in cluster 1 * a little more likely to be female than male, * are likely to be married, * do not have a car loan or a mortgage.

In general, in characterising clusters, it is of interest to consider both variables that were included in creating the clusters and variables that were not included. Since we have used information only about binary variables informing the clusters it is particularly interesting if the means of numerical variables vary a lot between clusters. It provides confirmation that the clusters are meaningful. Also look at the numerical variables, which were not involved in determining the clusters, and see if a consistent pattern emerges.

Again, for example, on average those in cluster 1: * are a little older than those in the other clusters, * have higher income, * have one child

Also calculate the standard deviations, and see if the differences between clusters are large compared to the standard deviations of numerical variables. Exercise: Try the “single”, “centroid”, and “complete” methods of calculating the distance between clusters, and compare the results when you specify 5 clusters for each of these.

Exercise: Try the “single”, “centroid”, and “complete” methods of calculating the distances between clusters, and compare the results when you specify 4,5 clusters for each of these.

```
In [ ]: # your code here
        fail() # No Answer - remove if you provide an answer
```

1.0.2 Exercise 2: Hierarchical cluster analysis using numerical variables

Staying with the same data set, let’s try cluster analysis using only the numerical variables. So this time we pull out the numerical variables.

Suppose for example we calculate the Euclidean distances between points. If we use the variables as they stand, big differences in incomes will swamp any variation in number of children, or even Age. In fact, even if the differences between the average values of variables are relatively minor, they still have to be taken into account.

We therefore start by standardising the variables. Recall from an earlier stats unit, or from the presentation, that this is achieved for each variable by subtracting the mean and dividing the result by the standard deviation. You end up with every standardised variable having mean 0 and standard deviation 1.

Standardisation of all the variables in a data frame is easy in R. Simply type the command `scale`.

For our workflow `%>% scale`. To calculate the distance the function is `dist(ktcNumeric.df, method = "euclidean")`.

```
In [ ]: # your code here
        fail() # No Answer - remove if you provide an answer
```

1.0.3 Review Questions

Q: Briefly explain the four linkage methods we have discussed for determining the distance between clusters (single, average, complete and centroid), and how the results are likely to differ based on which method is used. 2. Explain how you would choose between the Simple Matching and the Jaccard method of calculating similarity between cases. 3. Suppose you have data on the following variables:
* Overseas holiday last year or not * Owner-occupier or renting home * Has children or not
Which method would you choose and why?
What would you do with:
* Travelled to Bali last year or not * Lives in Toorak or not
* Has more than three children or not

Q: Consider the vertical axis represented on the dendrogram, which has the default label Height. Explain what is being graphed on this axis.

Review the pros and cons of hierarchical cluster analysis.