# Assignment5

# Hierarchical Clustering

```
C_df<-read.csv("cereals.csv")
str(C_df)
```

```
## 'data.frame':    77 obs. of  16 variables:
##  $ name    : chr  "100%_Bran" "100%_Natural_Bran" "All-Bran" "All-Bran_with_Extra_
Fiber" ...
##  $ mfr     : chr  "N" "Q" "K" "K" ...
##  $ type    : chr  "C" "C" "C" "C" ...
##  $ calories: int  70 120 70 50 110 110 110 130 90 90 ...
##  $ protein : int  4 3 4 4 2 2 2 3 2 3 ...
##  $ fat     : int  1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium  : int  130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber   : num  10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo   : num  5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars  : int  6 8 5 0 8 10 14 8 6 5 ...
##  $ potass  : int  280 135 320 330 NA 70 30 100 125 190 ...
##  $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
##  $ shelf   : int  3 3 3 3 3 1 2 3 1 3 ...
##  $ weight  : num  1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups    : num  0.33 1 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ rating  : num  68.4 34 59.4 93.7 34.4 ...
```

```
head(C_df)
```

```
##                         name mfr type calories protein fat sodium fiber carbo
## 1                  100%_Bran   N    C       70       4   1    130  10.0   5.0
## 2          100%_Natural_Bran   Q    C      120       3   5     15   2.0   8.0
## 3                   All-Bran   K    C       70       4   1    260   9.0   7.0
## 4 All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## 5             Almond_Delight   R    C      110       2   2    200   1.0  14.0
## 6   Apple_Cinnamon_Cheerios   G    C      110       2   2    180   1.5  10.5
##   sugars potass vitamins shelf weight cups   rating
## 1      6    280       25     3      1 0.33 68.40297
## 2      8    135        0     3      1 1.00 33.98368
## 3      5    320       25     3      1 0.33 59.42551
## 4      0    330       25     3      1 0.50 93.70491
## 5      8     NA       25     3      1 0.75 34.38484
## 6     10     70       25     1      1 0.75 29.50954
```

```
library(DataExplorer)
introduce(C_df) # Missing values
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1   77      16                3                  13                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    4            74               1232        17296
```

```
C_df1<-na.omit(C_df) #Data set with missing values in the rows that have been omitted
```

# Apply hierarchical clustering to the data using normalization measures and Euclidean distance.

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.5     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.4     ✓ stringr 1.4.0
## ✓ readr   2.0.2     ✓ forcats 0.5.1
```

```
## ── Conflicts ─────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve
3WBa
```

```
library(dendextend)
```

```
##
## ---------------------
## Welcome to dendextend version 1.15.2
## 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/dend
extend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend)
)
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

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

```
library(cluster)
library(fastDummies)


#Categorical and numerical variables must be identified.

C_df1$name<-as.factor(C_df1$name)
C_df1$mfr<-as.factor(C_df1$mfr)
C_df1$type<-as.factor(C_df1$type)
C_df1$shelf<-as.factor(C_df1$shelf)


# Creating dummy variables
vaar<-colnames(C_df1)
n_var<-c("calories","protein","fat","sodium","fiber","carbo","sugars","potass","vitam
ins","weight","cups","rating")
cat_var<-C_df1[which(colnames(C_df1)%in%c('name','mfr','type','shelf'))]
cat_var<-data.frame(apply((C_df1[which(colnames(C_df1)%in%c('name','mfr','type','shel
f'))]),2,as.factor))
dummy_vars<-fastDummies::dummy_columns(cat_var %>% select(-name))
n_vars<-C_df1[,c(4:12,14:16)]
C_df2<-cbind(C_df1$name,dummy_vars,n_vars)%>% select(-c(mfr,type,shelf))
```

#Normalizing the data set

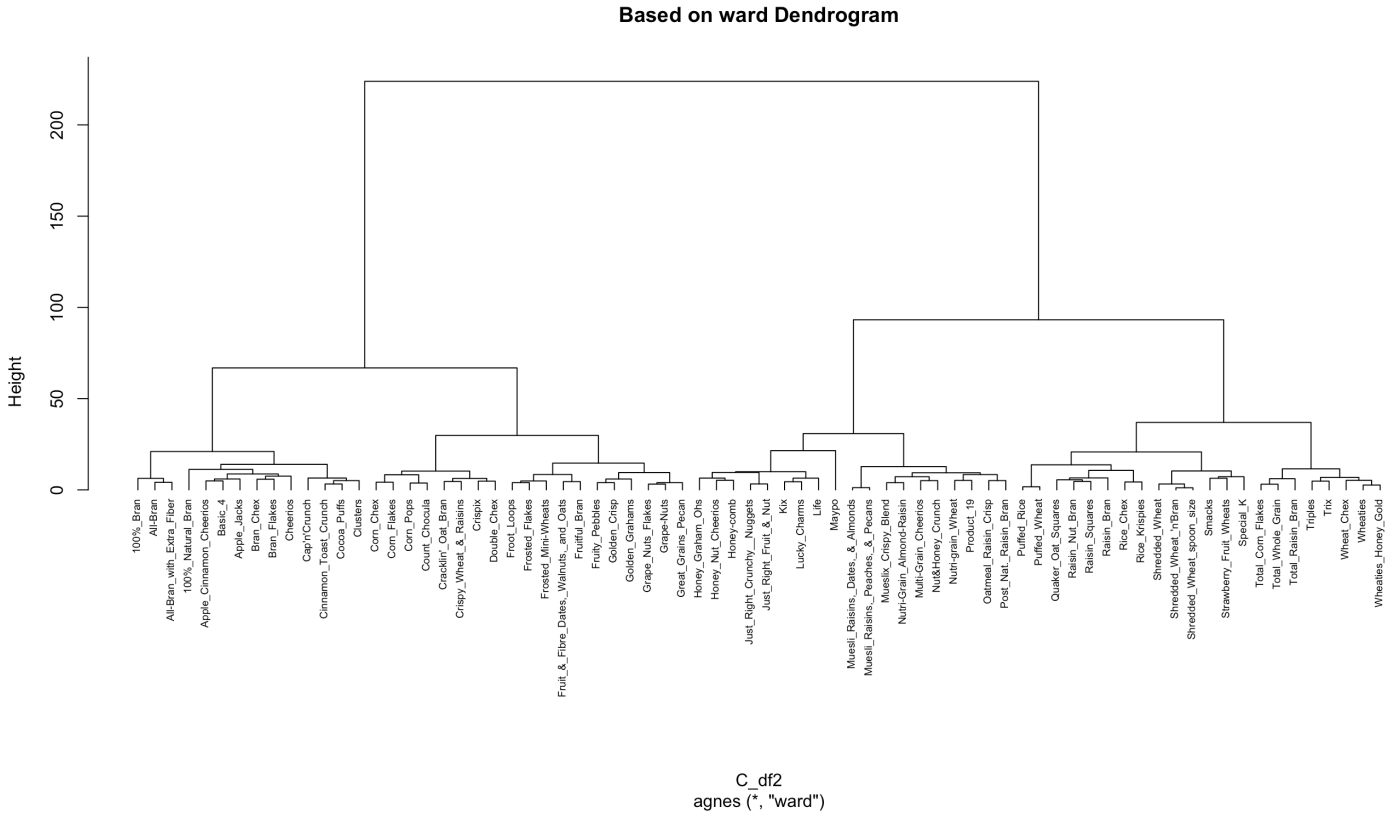
```
C_df2[,c(2:25)]<-scale(C_df2[,c(2:25)],scale = TRUE, center = TRUE)
```

Q1.Use Agnes to compare the clustering from single linkage, complete linkage, average linkage, and Ward. Choose the best method.

```
Hclustering1<- agnes(C_df2, method="complete")
Hclustering2<- agnes(C_df2, method = "average")
Hclustering3<- agnes(C_df2, method="single")
Hclustering4<- agnes(C_df2, method="ward")


ac<-c(Hclustering1$ac,Hclustering2$ac,Hclustering3$ac,Hclustering4$ac)
ac_method<-c(Hclustering1$method,Hclustering2$method,Hclustering3$method,Hclustering4
$method)
ac_df<-data.frame(ac_method, ac)
ac_df
```

```
##   ac_method        ac
## 1  complete 0.9360496
## 2   average 0.8779145
## 3    single 0.7194916
## 4      ward 0.9787193
```

```
pltree(Hclustering4,cex=0.6,hang=-1,main="Based on ward Dendrogram",labels=C_df2$'C_df1$name')
```



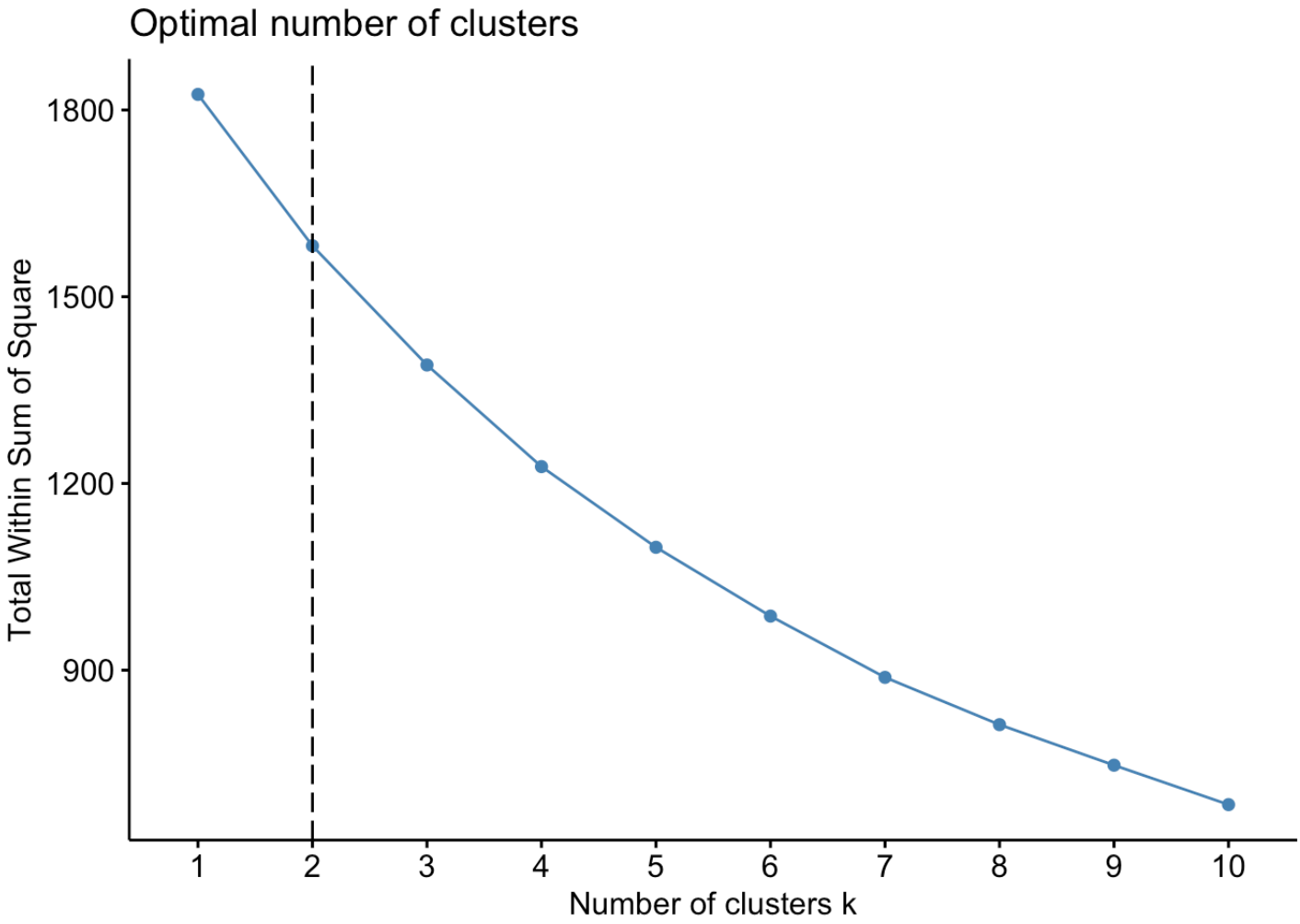

#According to the table above,the ward technique has the highest agglomerative coefficient,meaning it is the closest to one.As a result it produces the most clusters.

2.How many clusters would you choose?

```
fviz_nbclust(C_df2,hcut,method="wss")+geom_vline(xintercept=2,linetype=5)
```

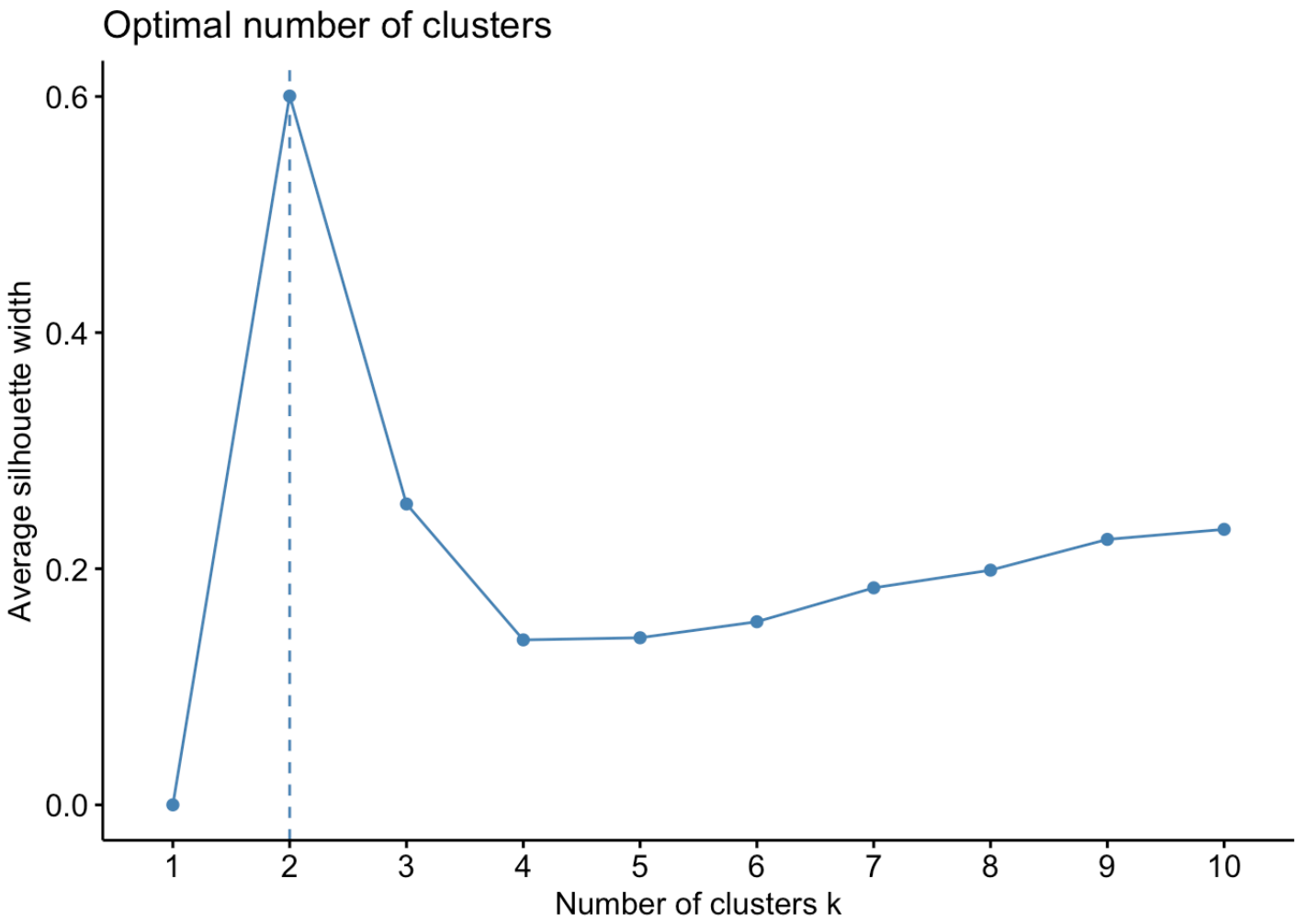
```
## Warning in stats::dist(x): NAs introduced by coercion
```

```
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion

## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
```



```
fviz_nbclust(C_df2,hcut,method = "silhouette")
```

```
## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion

## Warning in stats::dist(x): NAs introduced by coercion
```




```
C_df2<-C_df2%>% mutate(cluster=cutree(Hclustering4,k=2))
```

#I will select Two clusters based on the dendogram

3.Comment on the structure of the clusters and on their stability.

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(dplyr)
set.seed(12)
split_index<-createDataPartition(C_df2$rating,p=0.6,times=1,list=FALSE)

C_p1<-C_df2[ split_index, ]
C_p2<-C_df2[-split_index, ]


centroid1<-C_p1 %>% select_if(is.numeric) %>% filter(cluster==1) %>% colMeans()

centroid2<-C_p1 %>% select_if(is.numeric)%>% filter(cluster==2)%>% colMeans()

centroid<-rbind(centroid1,centroid2)


cluster_B<-data.frame(data=seq(1,nrow(C_p2),1),clusterB=rep(0,nrow(C_p2)))


for (x in 1:nrow(C_p2)) {cluster_B$clusterB<-which.min(as.matrix(get_dist(as.data.fra
me(rbind(centroid[,-25],C_p2[x,c(-1,-26)])))))[3,-3])

}


cluster_B<-cluster_B %>% mutate(orig_clusters=C_p2$cluster)

mean(cluster_B$clusterB==cluster_B$orig_clusters)
```

```
## [1] 0.3928571
```

# According to the comparison,the clusters are stable.

4. The elementary public schools would like to choose a set of cereals to include in their daily cafeterias

```
h_cereals<-data.frame(C_df2 %>% filter(cluster==2) %>% select_if(is.numeric) %>% colM
eans())
```

# Cluster Two has cereals that are high in protein,vitamins,carbohydrates and

fiber,but low in sodium and sugar.As a result,Cereals in cluster 2 can be used to maintain a healthy diet.