2017195113

10/26/2020

#A.DATASET1: DATA WRANGLING —-

#1. Libraries

library(tidyverse) library(magrittr)

#2. Data Acquisition

data.raw<- read.csv("wine_data.csv")

data.raw %>% glimpse

## 3. Data Insepection

data.raw %>% str

data.raw %>% dim

data.raw %>% view

data.raw %>% names

data.raw %>% rownames

#Qusetion: How can we categorize wines in clusters?

## Problem 1: Cluster wines based on attributes

## Problem 2: What is the optimal number of clusters?

## First Question: How to clean the data? Missing values? Outliers?

#3.1 Dealing with NA

data.raw %>% anyNA

#no NA

data.raw %>% map_dbl(~sum(is.na(.)))

#no NA

#3.2 Dealing with whitespace

names(data.raw)

#no whitespace or .

#3.3 To lowercase

```
names(data.raw) %<>% tolower
names(data.raw)
data.raw %>% glimpse
#4. Checking/Converting data types
data.raw %>% dplyr::summarize(across(.cols = everything(), .fns = class))
#no need to change
#5. Descriptive Statistics
#5.1 Checking outliers
```

## fastest way to glimpse data distributions for discrete & continuous variables

```
data.raw %>% Hmisc::hist.data.frame()
#ash, magnesium, flavanoids suspicious
data.raw %>% skimr::skim()
library(outliers)
outlier(data.raw)
#proline also suspicious
#5.2 checking/changing typos
#ash
data.raw$ash %>% table #seems like data itself, no typos, ok
#magnesium
data.raw$magnesium %>% table
#1010 seems like a typo, extra 0
data.raw$magnesium %<>% substr(0, 3) %>% as.numeric
data.raw$magnesium %>% table
data.raw$magnesium %>% summary()
#changed
#flavanoids
data.raw$flavanoids %>% table
#seems like data itself, no typos, ok
#proline
data.raw$proline %>% table
#seems like data itself, no typos, ok
#check final
data.raw %>% summary()
#mean and medium are similar in all vars
```

#6. Final Check on raw data/ Getting dataset, dataset.z

data.raw %>% summary()

data.raw %>% boxplot()

data.raw %>% anyNA

data.raw %>% names

dataset = data.raw

dataset.z = dataset %>% map_df(scale)

## check result of standardization

dataset.z %>% head(30)

dataset.z %>% summary()

dataset.z %>% boxplot()

#B. KMEANS CLUSTERING —-

library(factoextra)

library(ggplot2)

seed <- 4

set.seed(seed)

#1. Determine optimal # of cluster

dataset.z %>% fviz_nbclust(., FUNcluster = kmeans, method = “wss”)

#OPTIMAL #: 3

dataset.z %>% fviz_nbclust(., FUNcluster = kmeans, method = “silhouette”)

#OPTIMAL #: 3

#2. Check between-cluster variance

#try out around 3,4 just in case

kmeans1 = kmeans(dataset.z, 3) %T>% print

#btw-cluster: 44.8

kmeans1$size

#even

kmeans2 = kmeans(dataset.z, 4) %T>% print

#btw-cluster: 47.8

kmeans2$size

#less even

## Check within SS

kmeans1$tot.withinss

kmeans2$tot.withinss #similar

#FINAL OPTIMAL #: cluster # of 3 is better, because of more even distribution, #and becuase the between-variance of clusters and wihinss does not significantly differ from 3 & 4

##4. Check nstart

kmeans1 = kmeans(dataset.z, 3) %T>% print

kmeans1.1 = kmeans(dataset.z,3, nstart = 50) %T>% print

kmeans1$tot.withinss

kmeans1.1$tot.withinss

#NSTART don't use nstart, not necessary here

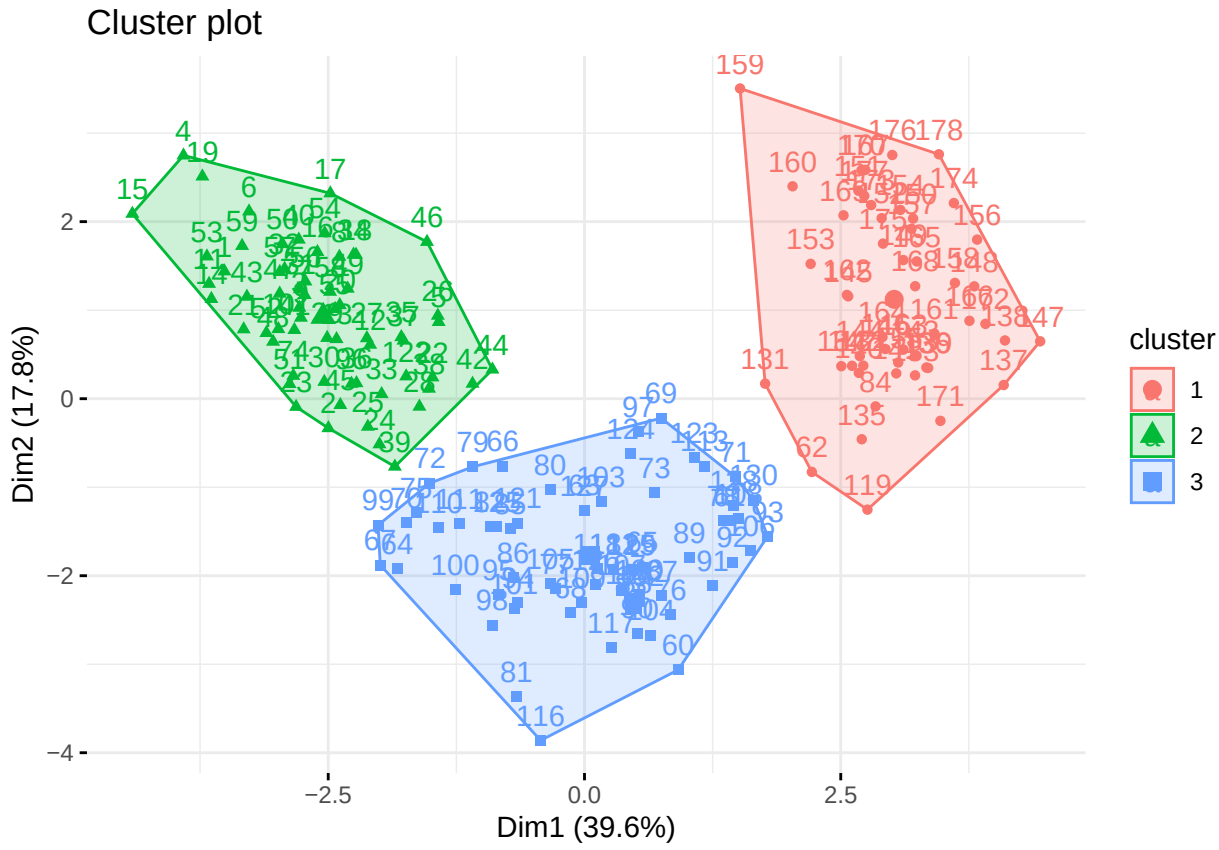
#4. put in cluster.kmeans to the original dataset.z

dataset.z$cluster.kmeans = kmeans1$cluster

dataset.z %>% names

#5. Viz

```
factoextra::fviz_cluster(kmeans1, dataset.z %>% select(-cluster.kmeans),
                         ggtheme = theme_minimal())
```




#ANALYSIS: NON OVERLAPPLING CLUSTERS

#C. HIERARCHICAL CLUSTERING —-

library(dendextend)

set.seed(seed)

#1. Euclidean distance, complete linkage

distance.matrix1 = dist(dataset.z, method = “euclidean”)

hc.euclidean.complete = hclust(distance.matrix1, method = “complete”)

#3. Correlation dissimaliry distance, complete

distance.matrix2 = as.dist(1-cor(t(dataset.z)))

hc.correlation.complete = hclust(distance.matrix2, method = “complete”)

#4. Comparing two methods

cutree(hc.correlation.complete, k = 3) %>% table

cutree(hc.euclidean.complete, k = 3) %>% table

# Result:

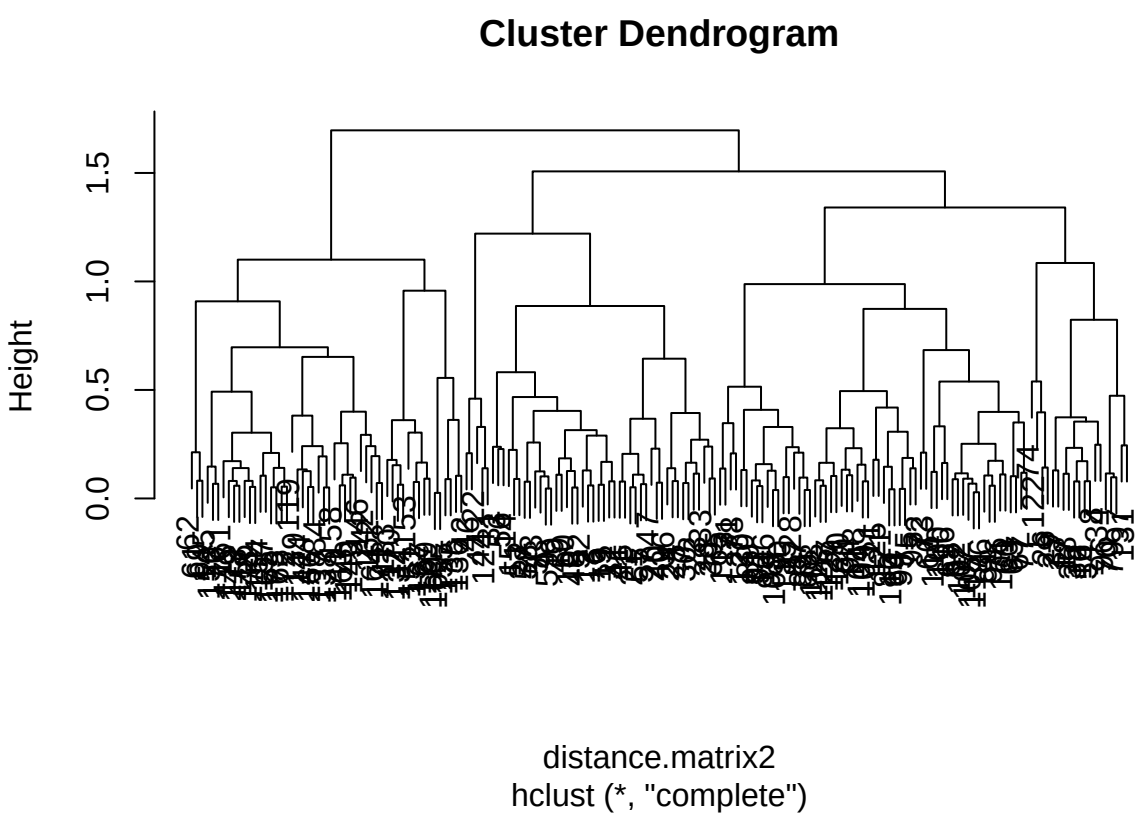
### euclidean distance is better, given that using complete linkage

#5. put hclust.cluster back in original dataset.z

dataset.z$cluster.hclust = cutree(hc.euclidean.complete, k = 3)

#6. Viz

```
hc.correlation.complete %>% plot
```




#ANALYSIS: EVEN CLUSTERS

#D. COMPARING KMEANS VS HCLUST —-

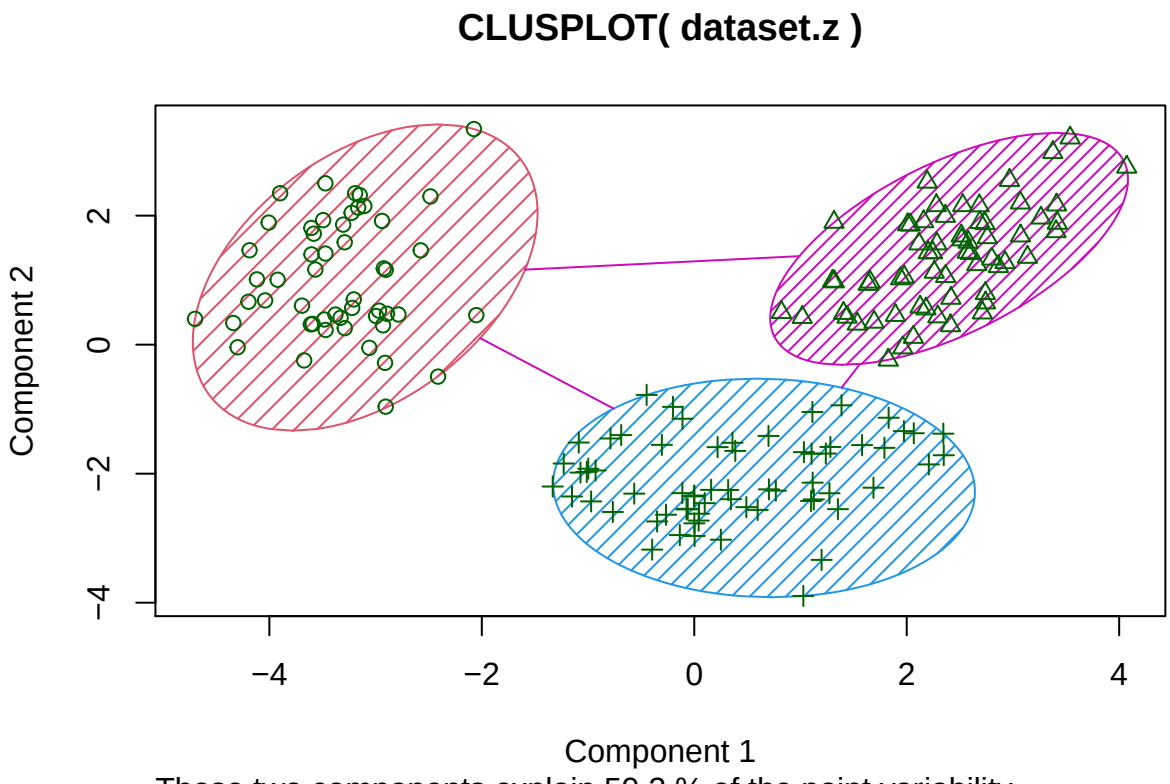
library(cluster)

dataset.z %>% names

## Visualise the kmeans and hierarchical clustering

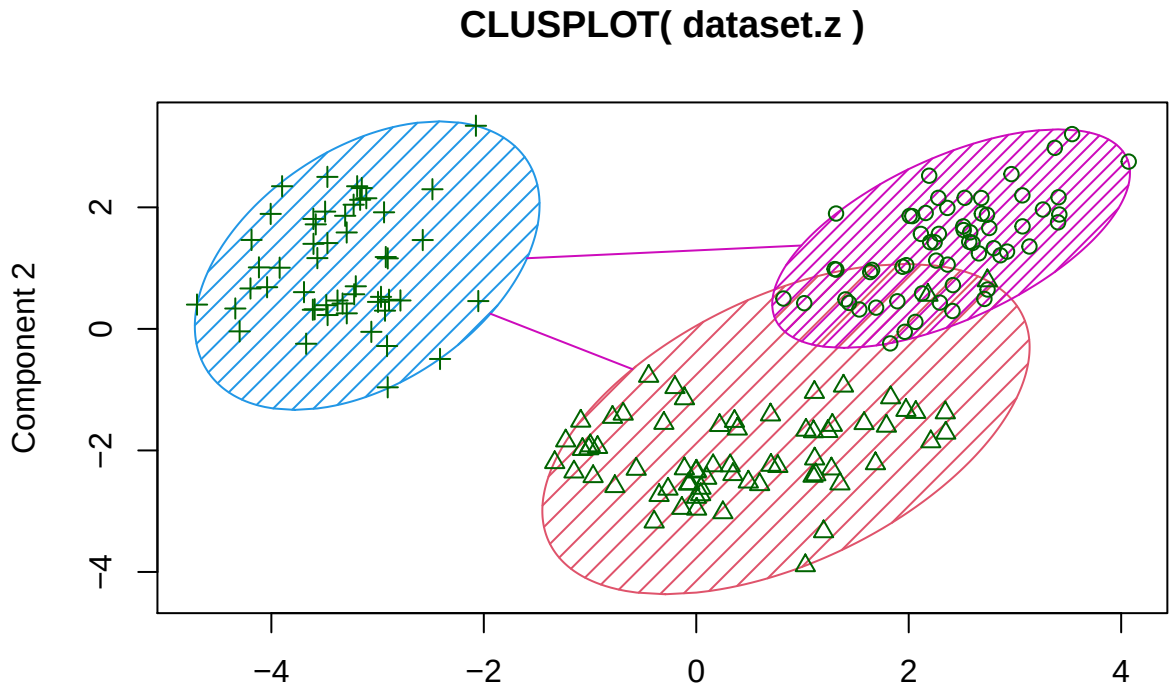
#1. Clusplot kmeans

```
clusplot(dataset.z, dataset.z$cluster.kmeans, color = TRUE, shade = TRUE)
```




#2. Clusplot Hclust

```
clusplot(dataset.z, dataset.z$cluster.hclust,
         color= TRUE, shade = TRUE)
```




#ANALYSIS: KMEANS BETTER, NON OVERLAPPING

#3. Extract the profiling attributes for each of the segments/clusters across all features: # Calculate average feature scores per cluster and compare with population statistics

population <- dataset.z %>% select(-cluster.kmeans, -cluster.hclust) %>% colMeans() %>% as.data.frame() %>% rownames_to_column() %>% set_names(c("attribute", "population")) %>% as_tibble() %>% print()

snake_plot <- function(data, group_variable, to_remove) { group.variable <- rlang::sym(group_variable) %T>% print data %>% select(-to_remove) %>% gather(attribute, value, -!!group_variable) %>% ggplot(aes(x = attribute, y = value, group = !!group.variable)) + geom_line(aes(color = as.factor(!!group.variable))) + coord_flip() }

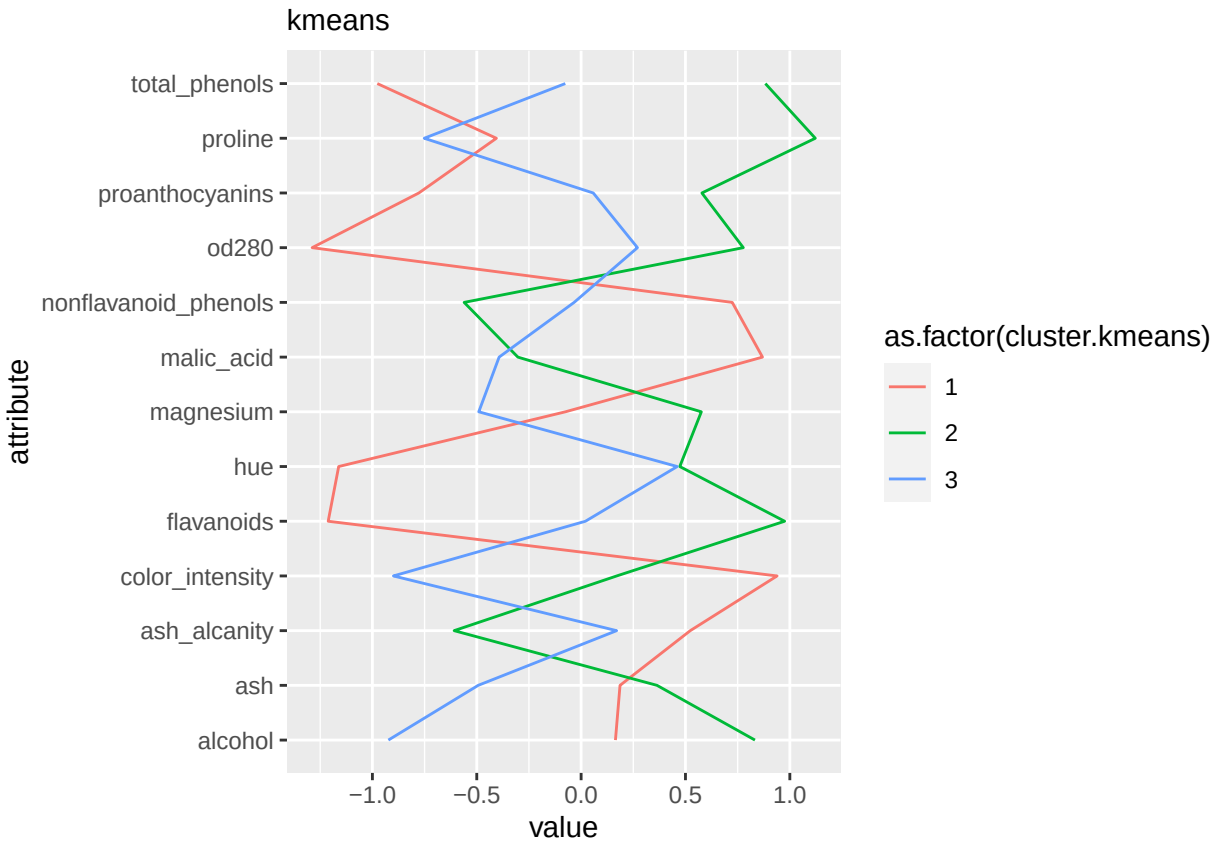
#4. Snakeplot kmeans

cluster.kmeans.means.z <- dataset.z %>% dplyr::group_by(cluster.kmeans) %>% dplyr::summarize(across(.cols= everything(), .fns = mean)) %>% as.data.frame

print(cluster.kmeans.means.z)

```
snake_plot(cluster.kmeans.means.z,
           group_variable = "cluster.kmeans",
           to_remove = "cluster.hclust"
) + labs(subtitle = "kmeans")
```

```
## cluster.kmeans
```



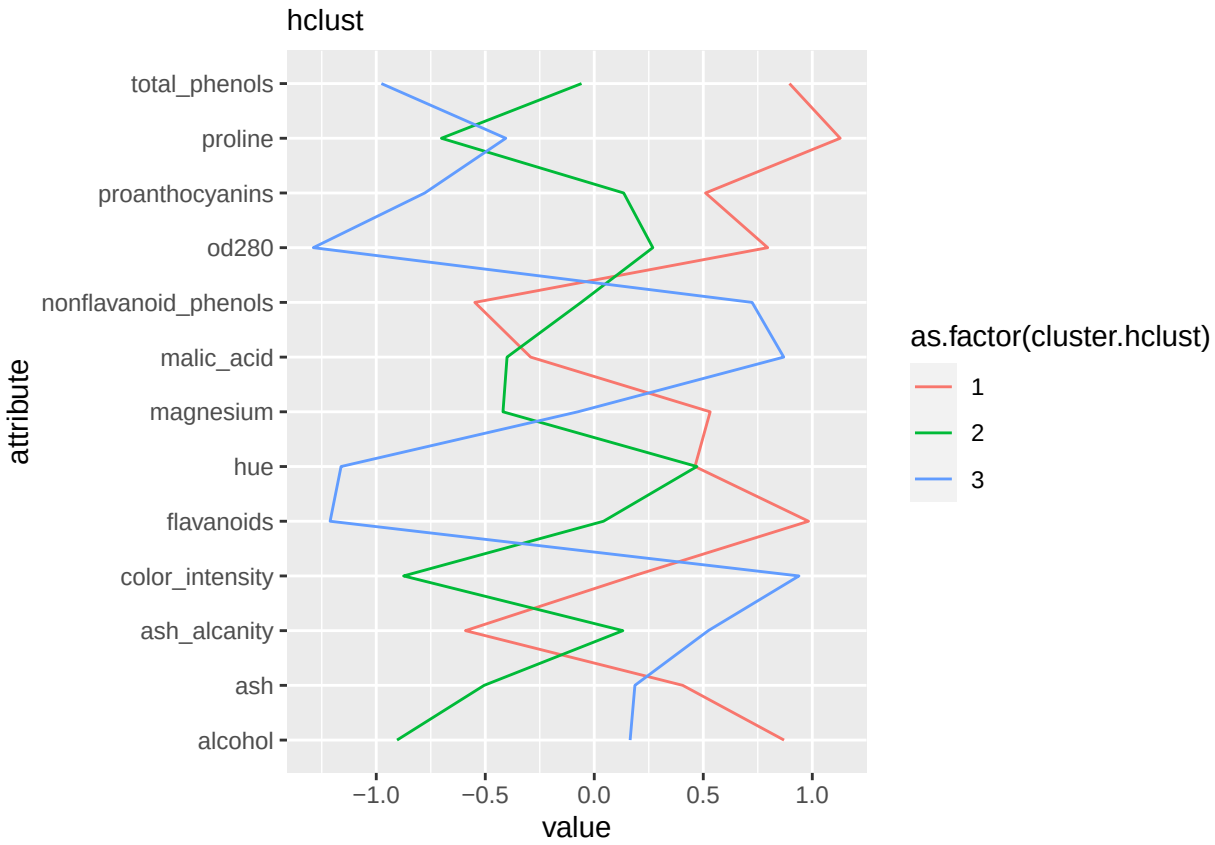

#5. Snakeplot hclust

cluster.hclust.means.z <- dataset.z %>% group_by(cluster.hclust) %>% summarise_all(.funs = mean) %>% as.data.frame()

print(cluster.hclust.means.z)

```
snake_plot(cluster.hclust.means.z,
           group_variable = "cluster.hclust",
           to_remove = "cluster.kmeans"
)+ labs(subtitle = "hclust")
```

```
## cluster.hclust
```




#loadings

print(cluster.kmeans.means.z)

print(cluster.hclust.means.z)

#ANALYSIS ON COMPARING KMEANS AND HCLUST (SNAKEPLOT & CLUSPLOT & DATA DISTRIBUTION ON CLUSTERS):

#wHICH IS BETTER, KMEANS OR HCLUST: When comparing kmeans and Hierarchical clustering with snake plot, both seems to be showing the distribution of each variables quite well. However, if we look at the clusplot, we can see that kmeans is better at showing dissimilarity between clusters as they are nonoverlapping, while hierarchical clusters are overlapping. Thus, kmeans is preferred.

#DATA DISTRIBUTION ON CLUSTERS: Looking at the kmeans snakeplot and loadings, cluster 1 has high amount of alcohol and flavonoids, proline. Cluster 2 has the lowest amount of alcohol with medium amount of flavonoids, and Cluster 3 has a relatively low amount of alcohol with very low amount of flavonoids. Wines in Cluster 1 would taste a lot of flavonoids, while Cluster 3 would have a sour and smoky taste due to high amount of acid and ash. Compared to the other clusters, Cluster 2 wines would have mild tastes.

#E.DATASET2: FA —-

library(psych)

library(GPArotation)

dataset2 = read.csv("us presidential data.csv")

#0. Cleaning data

dataset2 %>% names

dataset2 %>% str

dataset2 %<>% select(-c(PastUsed, FutureUsed, NumericContent, PresentUsed))

dataset2 %>% names

#0.1

dataset2 %<>% na.omit

dataset2 %>% anyNA

dataset2 %>% view

#0.2 Descriptive Statistics

#Checking outliers

## fastest way to glimpse data distributions for discrete & continuous variables

dataset2 %>% Hmisc::hist.data.frame()

dataset2$OppPartyCount %>% table

#typo of 0 at the end of 110

dataset2$OppPartyCount %<>% substr(0, 2) %>% as.numeric

dataset2$Extra %>% table

#ok

dataset2$Openn %>% table

#ok

dataset2$OwnPartyCount %>% table

#check

dataset2.z = dataset2 %>% map_df(scale)

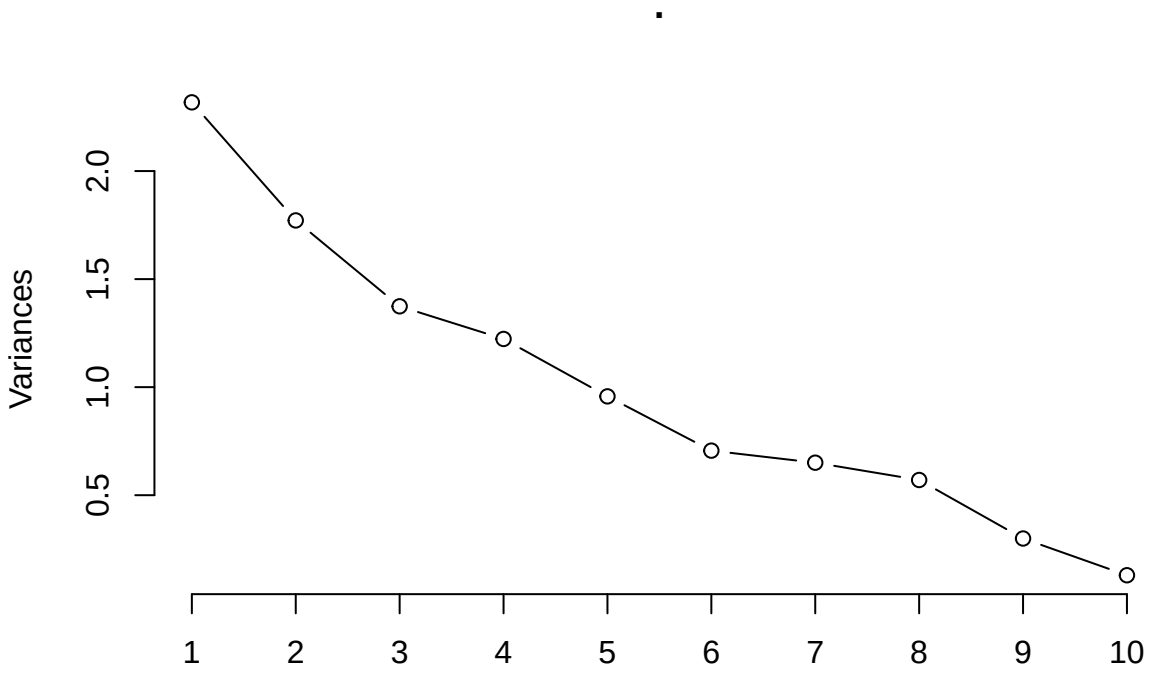
dataset2.z %>% view

#Question: What are the most important reasons for voting the president?

#REASON TO USE FA, not PCA: PCA is primarily for summarizing data in less characteristics while FA is for finding underlying causes for sthe variables. Thus, for finding the important reasons, FA is more appropriate.

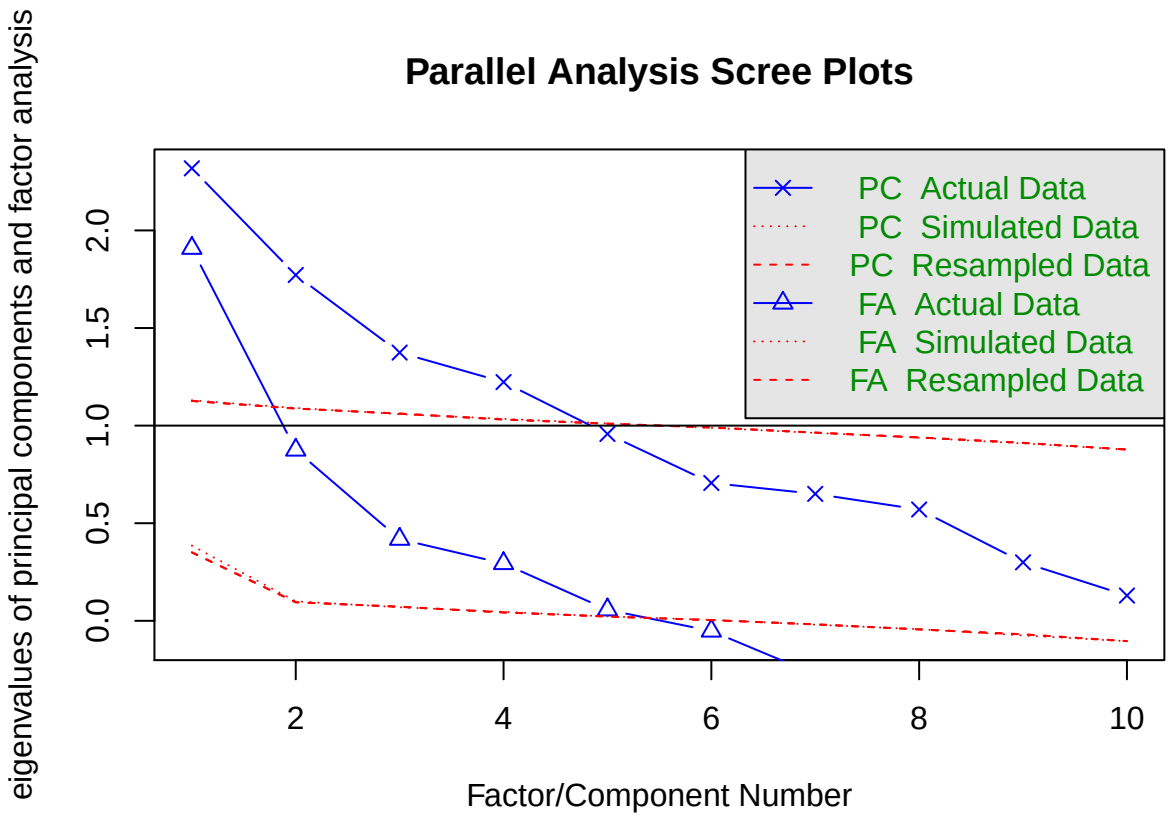
#1. FINDING OPTIMAL # OF FACTORS

```
dataset2.z %>% prcomp() %>% screeplot(type = "lines")
```




#OPTIMAL #: 5

```
fa.parallel(dataset2.z)
```




```
## Parallel analysis suggests that the number of factors =  5  and the number of components =  4
```

#OPTIMAL #: suggests 4 factors

#2. FA

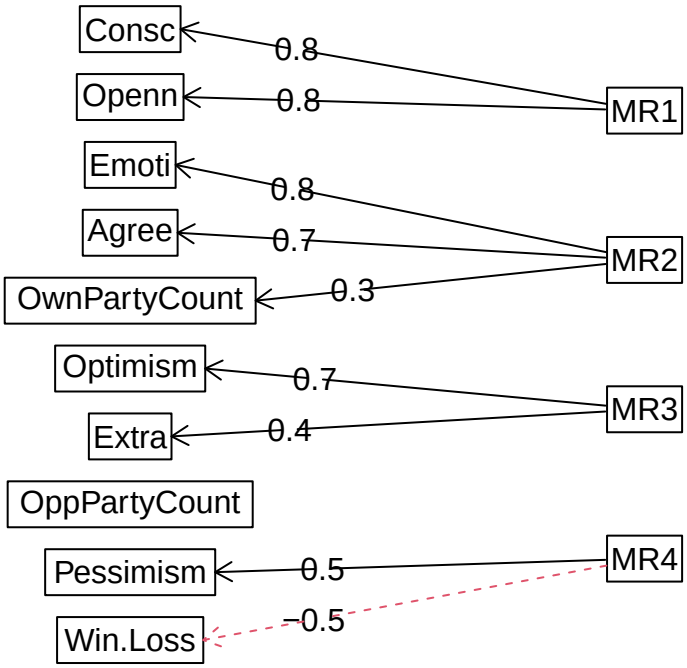
#2.1 FA according to fa.parallel fa = dataset2.z %>% fa( ., nfactors = 4) %T>% print

#CUMULATIVE VARIANCE: 0.50,Fit based upon off diagonal values = 0.97

#CUMULATIVE VARIANCE: TOO LOW, should I increase?

#CHECK DOUBLE LOADINGS: none

```
fa %>% fa.diagram
```



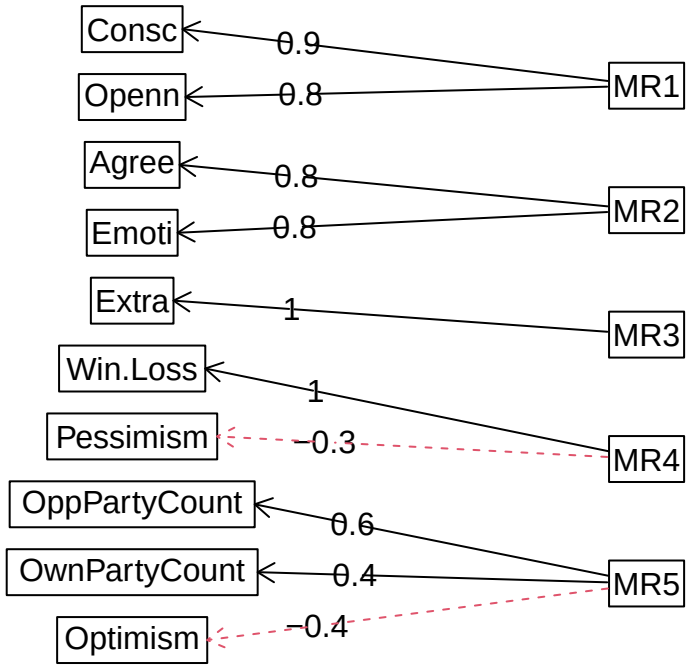

#2.1 FA according to screeplot

fa2 = dataset2.z %>% fa( ., nfactors = 5) %T>% print

#CUMULATIVE VARIANCE: 0.59,Fit based upon off diagonal values = 1

#CHECK DOUBLE LOADINGS:

```
fa2 %>% fa.diagram
```




#ANALYSIS ON COMPARING FA & FA2: Focusing on simple cumulative variance seems to render to much factors, making them no longer meaningful. Once the number of factors become 5 or larger, meaningless factors start to arise, seeing from fa diagram. Thus, keeping the number of factors to 4, as suggested by the fa.parallel seems to be a better solution.

#ANALYSIS ON FA (LOADINGS & DIAGRAM): #1. CHOOSING BTW PCA AND FA: As mentioned before, PCA is primarily for summarizing data in less characteristics while FA is for finding underlying causes for sthe variables. Thus, for finding the important reasons, FA is more appropriate.

#2. FA ANALYSIS based on what is the reason for voting the president: According to the FA Analysis, Winning an election is the reason related lightly with pessimistic personlaity. While other factors do not load on to MR4, Win.Loss is loaded solely on MR4, and the only related personality to that is, although weakly so, would be pessimism. Other personalities such as Consciousness, Openess does not have any effect on the winning of the presidential election according to FA.