Homework 3. Clustering Practice (80 Points)

Shri Harsha

2023-10-12

Part 1. USArrests Dataset and Hierarchical Clustering (20 Points)

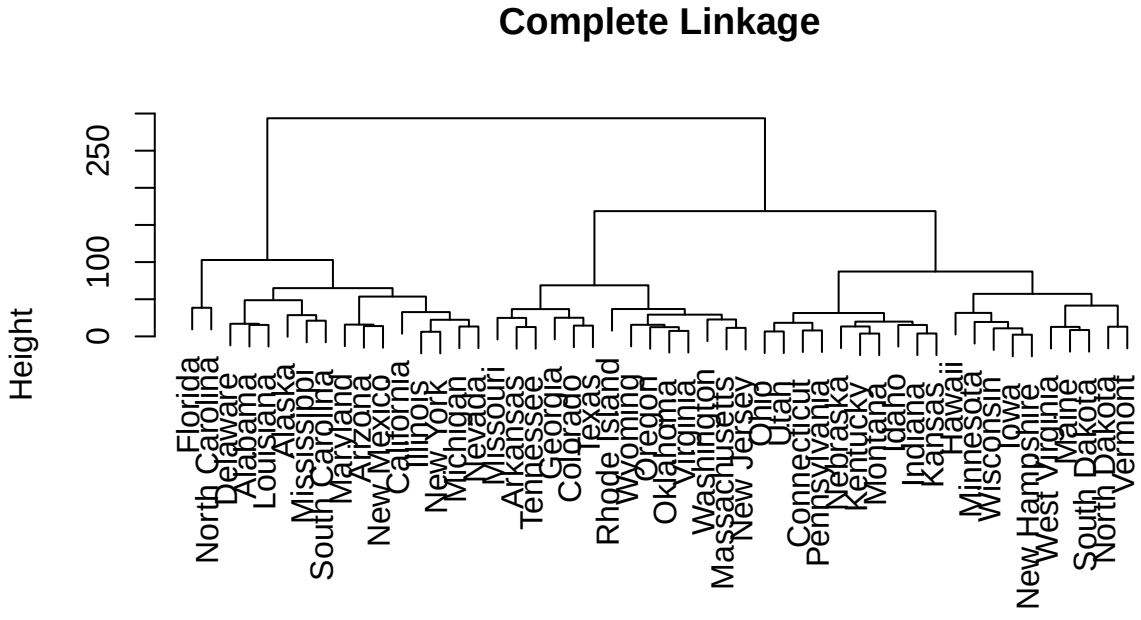
Consider the "USArrests" data. It is a built-in dataset you may directly get in RStudio. Perform hierarchical clustering on the observations (states) and answer the following questions.

```
head(USArrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

Q1.1. Using hierarchical clustering with complete linkage and Euclidean distance, cluster the states. (5 points)

```
set.seed(69)
us_arrests <- na.omit(USArrests)
hier_clust <- hclust(dist(us_arrests), method = "complete")
plot(hier_clust, main = "Complete Linkage", xlab = "", sub = "")
```



Q1.2. Cut the dendrogram at a height that results in three distinct clusters. Interpret the clusters. Which

states belong to which clusters? (5 points)

```
clusters <- cutree(hier_clust, k = 3)
state_names <- rownames(us_arrests)
cluster_data <- data.frame(State = state_names, Cluster = clusters)
cluster_states_1 <- cluster_data$State[cluster_data$Cluster == 1]
cluster_states_1
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
```

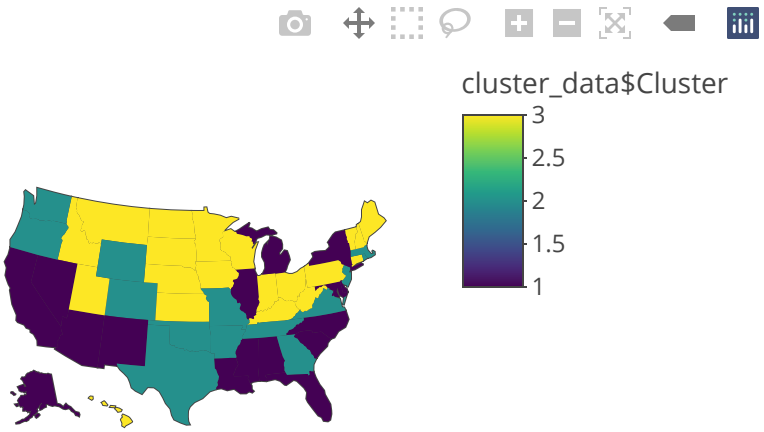
```
cluster_states_2 <- cluster_data$State[cluster_data$Cluster == 2]
cluster_states_2
```

```
##  [1] "Arkansas"      "Colorado"      "Georgia"       "Massachusetts"
##  [5] "Missouri"      "New Jersey"    "Oklahoma"      "Oregon"
##  [9] "Rhode Island"  "Tennessee"     "Texas"         "Virginia"
## [13] "Washington"    "Wyoming"
```

```
cluster_states_3 <- cluster_data$State[cluster_data$Cluster == 3]
cluster_states_3
```

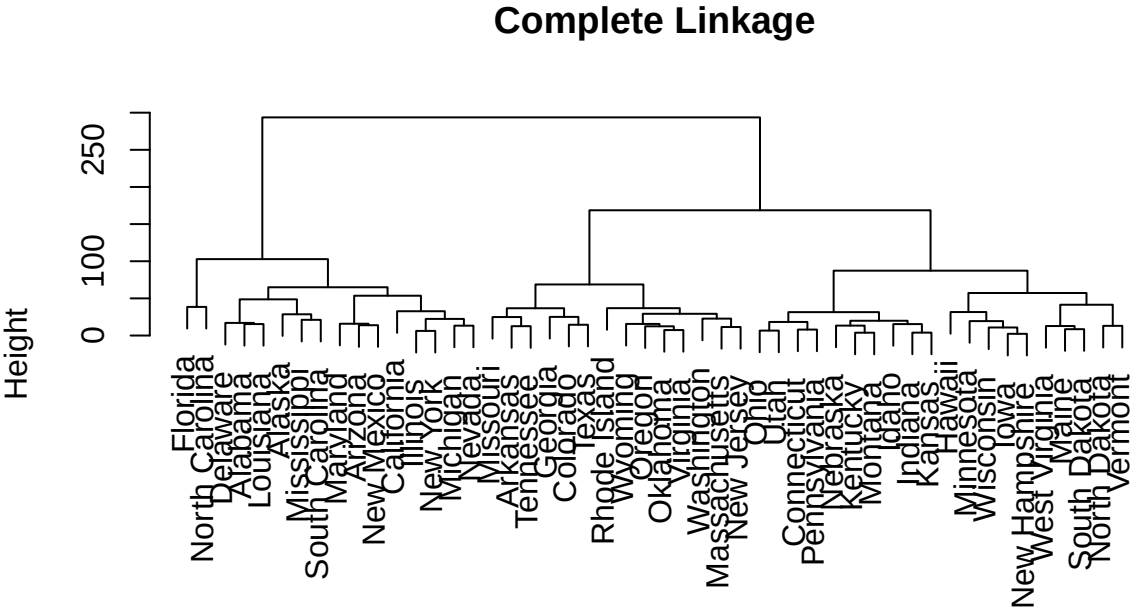
```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"          "Vermont"       "West Virginia" "Wisconsin"
```

```
g <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  lakecolor = toRGB('white')
)



plot_geo() %>%
  add_trace(
    z = ~cluster_data$Cluster, text = ~cluster_data$State, span = I(0),
    locations = ~state.abb, locationmode = 'USA-states'
  ) %>%
  layout(geo = g)
```



Q1.3 Hierarchically cluster the states using complete linkage and Euclidean distance, after scaling the variables to have standard deviation one. Obtain three clusters. Which states belong to which clusters?(5 points)

```
us_arrests <- na.omit(USArrests)
scaled_arrests <- scale(us_arrests)
hier_clust <- hclust(dist(us_arrests), method = "complete")
plot(hier_clust, main = "Complete Linkage", xlab = "", sub = "")
```




```
clusters <- cutree(hier_clust, k = 3)
state_names <- rownames(us_arrests)
cluster_data <- data.frame(State = state_names, Cluster = clusters)
cluster_states_1 <- cluster_data$State[cluster_data$Cluster == 1]
cluster_states_1
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
```

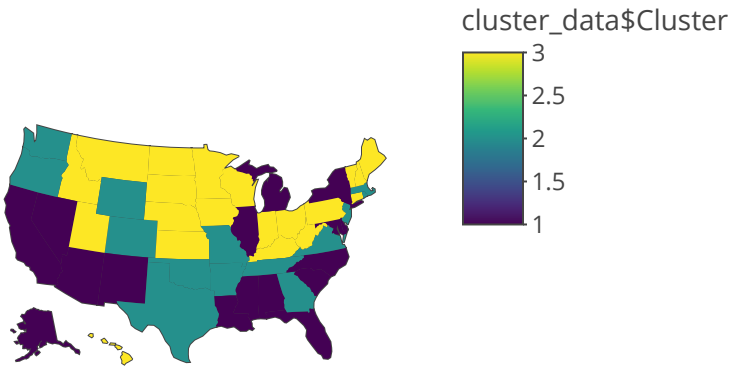
```
cluster_states_2 <- cluster_data$State[cluster_data$Cluster == 2]
cluster_states_2
```

```
##  [1] "Arkansas"      "Colorado"      "Georgia"       "Massachusetts"
##  [5] "Missouri"      "New Jersey"    "Oklahoma"      "Oregon"
##  [9] "Rhode Island"  "Tennessee"     "Texas"         "Virginia"
## [13] "Washington"    "Wyoming"
```

```
cluster_states_3 <- cluster_data$State[cluster_data$Cluster == 3]
cluster_states_3
```

```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"          "Vermont"       "West Virginia" "Wisconsin"
```

```
g <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  lakecolor = toRGB('white')
)



plot_geo() %>%
  add_trace(
    z = ~cluster_data$Cluster, text = ~cluster_data$State, span = I(0),
    locations = ~state.abb, locationmode = 'USA-states'
  ) %>%
  layout(geo = g)
```



Q1.4 What effect does scaling the variables have on the hierarchical clustering obtained? In your opinion, should the variables be scaled before the inter-observation dissimilarities are computed? Provide a justification for your answer. *(5 points)*

Answer: Scaling in hierarchical clustering is a process that adjusts the numerical values of variables to a common scale. It's done to prevent variables with larger ranges or variances from dominating the clustering process. Scaling ensures that each variable is given equal importance in the clustering analysis, allowing the relationships and patterns within the data to be accurately reflected. However, scaling is not always necessary; if your variables are already on a similar scale or scaling would distort their meaning, you can skip this step. The key is to make clustering results unbiased and based on the actual data patterns. In this particular example scaling made the dendogram a little balanced.

Part 2. Market Segmentation (60 Points)

An advertisement division of large club store needs to perform customer analysis the store customers in order to create a segmentation for more targeted marketing campaign

You task is to identify similar customers and characterize them (at least some of them). In other word perform clustering and identify customers segmentation.

This data-set is derived from https://www.kaggle.com/imakash3011/customer-personality-analysis

```
Colomns description:
People
  ID: Customer's unique identifier
  Year_Birth: Customer's birth year
  Education: Customer's education level
  Marital_Status: Customer's marital status
  Income: Customer's yearly household income
  Kidhome: Number of children in customer's household
  Teenhome: Number of teenagers in customer's household
  Dt_Customer: Date of customer's enrollment with the company
  Recency: Number of days since customer's last purchase
  Complain: 1 if the customer complained in the last 2 years, 0 otherwise

Products

  MntWines: Amount spent on wine in last 2 years
  MntFruits: Amount spent on fruits in last 2 years
  MntMeatProducts: Amount spent on meat in last 2 years
  MntFishProducts: Amount spent on fish in last 2 years
  MntSweetProducts: Amount spent on sweets in last 2 years
  MntGoldProds: Amount spent on gold in last 2 years

Place
  NumWebPurchases: Number of purchases made through the company's website
  NumStorePurchases: Number of purchases made directly in stores
```

Assume that data was current on 2014-07-01

Q2.1. Read Dataset and Data Conversion to Proper Data Format *(12 points)*

Read "m_marketing_campaign.csv" using `data.table::fread` command, examine the data.

```
# fread m_marketing_campaign.csv and save it as df (2 points)
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

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

```
##     transpose
```

```
market_df = fread('m_marketing_campaign.csv')
head(market_df)
```

```
##      ID Year_Birth Education Marital_Status Income Kidhome Teenhome Dt_Customer
## 1: 5524       1957  Bachelor         Single  58138       0        0  04-09-2012
## 2: 2174       1954  Bachelor         Single  46344       1        1  08-03-2014
## 3: 4141       1965  Bachelor       Together  71613       0        0  21-08-2013
## 4: 6182       1984  Bachelor       Together  26646       1        0  10-02-2014
## 5: 5324       1981       PhD        Married  58293       1        0  19-01-2014
## 6: 7446       1967    Master       Together  62513       0        1  09-09-2013
##    Recency MntWines MntFruits MntMeatProducts MntFishProducts MntSweetProducts
## 1:      58      635        88             546             172               88
## 2:      38       11         1               6               2                1
## 3:      26      426        49             127             111               21
## 4:      26       11         4              20              10                3
## 5:      94      173        43             118              46               27
## 6:      16      520        42              98               0               42
##    MntGoldProds NumWebPurchases NumStorePurchases Complain
## 1:           88               8                 4        0
## 2:            6               1                 2        0
## 3:           42               8                10        0
## 4:            5               2                 4        0
## 5:           15               5                 6        0
## 6:           14               6                10        0
```

```
# Convert Year_Birth to Age (assume that current date is 2014-07-01) (2 points)

market_df <- market_df %>%
  mutate(Age = year(as.Date("2014-07-01")) - Year_Birth)


# Dt_Customer is a date (it is still character), convert it to membership days (i.e. number of days per
# hint: note European date format, use as.Date with proper format argument (2 points)

market_df <- market_df %>%
  mutate(MembershipDays = as.integer(difftime(as.Date("2014-07-01"), as.Date(Dt_Customer, format = "%d-%

head(market_df)
```

```
##      ID Year_Birth Education Marital_Status Income Kidhome Teenhome Dt_Customer
## 1: 5524       1957  Bachelor         Single  58138       0        0  04-09-2012
## 2: 2174       1954  Bachelor         Single  46344       1        1  08-03-2014
## 3: 4141       1965  Bachelor       Together  71613       0        0  21-08-2013
## 4: 6182       1984  Bachelor       Together  26646       1        0  10-02-2014
## 5: 5324       1981       PhD        Married  58293       1        0  19-01-2014
## 6: 7446       1967    Master       Together  62513       0        1  09-09-2013
##    Recency MntWines MntFruits MntMeatProducts MntFishProducts MntSweetProducts
## 1:      58      635        88             546             172               88
## 2:      38       11         1               6               2                1
## 3:      26      426        49             127             111               21
## 4:      26       11         4              20              10                3
## 5:      94      173        43             118              46               27
## 6:      16      520        42              98               0               42
```

```
##    MntGoldProds NumWebPurchases NumStorePurchases Complain Age MembershipDays
## 1:           88               8                 4        0  57            665
## 2:            6               1                 2        0  60            115
## 3:           42               8                10        0  49            314
## 4:            5               2                 4        0  30            141
## 5:           15               5                 6        0  33            163
## 6:           14               6                10        0  47            295
```

```
# Summarize Education column (use table function) (2 points)


# Lets create a new column EducationLevel from Education
# Lets treat Education column as ordinal categories and use years in education as a levels
# for distance calculations (2 points)
# Assuming following order and years spend for education:
#    HighSchool (13 years), Associate(15 years), Bachelor(17 years), Master(19 years), PhD(22 years)
# create EducationLevel from Education
# hint: use recode function (in mutate statement)

# Define the years spent for each education level
years_for_education <- c("HighSchool" = 13, "Associate" = 15, "Bachelor" = 17, "Master" = 19, "PhD" = 2

# Create the "EducationLevel" column based on the specified order and years spent for education
market_df <- market_df %>%
  mutate(EducationLevel = recode(Education, !!!years_for_education))

head(market_df)
```

```
##      ID Year_Birth Education Marital_Status Income Kidhome Teenhome Dt_Customer
## 1: 5524       1957  Bachelor         Single  58138       0        0  04-09-2012
## 2: 2174       1954  Bachelor         Single  46344       1        1  08-03-2014
## 3: 4141       1965  Bachelor       Together  71613       0        0  21-08-2013
## 4: 6182       1984  Bachelor       Together  26646       1        0  10-02-2014
## 5: 5324       1981       PhD        Married  58293       1        0  19-01-2014
## 6: 7446       1967    Master       Together  62513       0        1  09-09-2013
##    Recency MntWines MntFruits MntMeatProducts MntFishProducts MntSweetProducts
## 1:      58      635        88             546             172               88
## 2:      38       11         1               6               2                1
## 3:      26      426        49             127             111               21
## 4:      26       11         4              20              10                3
## 5:      94      173        43             118              46               27
## 6:      16      520        42              98               0               42
##    MntGoldProds NumWebPurchases NumStorePurchases Complain Age MembershipDays
## 1:           88               8                 4        0  57            665
## 2:            6               1                 2        0  60            115
## 3:           42               8                10        0  49            314
## 4:            5               2                 4        0  30            141
## 5:           15               5                 6        0  33            163
## 6:           14               6                10        0  47            295
##    EducationLevel
## 1:             17
## 2:             17
## 3:             17
## 4:             17
## 5:             22
```

```
## 6:             19
```

```
# Summarize Marital_Status column (use table function)


# Lets convert single Marital_Status categories for 5 separate binary categories  (2 points)
# Divorced, Married, Single, Together and Widow, the value will be 1 if customer
# is in that category and 0 if customer is not
# hint: use dummy_cols from fastDummies or dummyVars from caret package, model.matrix
# or simple comparison (there are only 5 groups)
# Keep Marital_Status for later use


# Define the five categories
categories <- c("Divorced", "Married", "Single", "Together", "Widow")

martia_status = market_df$Marital_Status
# Use the pivot_wider function to convert Marital_Status into binary columns
market_df <- market_df %>%
  mutate(Value = 1) %>%   # Add a temporary Value column with 1
  pivot_wider(names_from = Marital_Status, values_from = Value, values_fill = 0)

market_df$Marital_Status = martia_status
head(market_df)
```

```
## # A tibble: 6 x 26
##      ID Year_Birth Education Income Kidhome Teenhome Dt_Customer Recency
##   <int>      <int> <chr>      <int>   <int>    <int> <chr>          <int>
## 1  5524       1957 Bachelor   58138       0        0 04-09-2012        58
## 2  2174       1954 Bachelor   46344       1        1 08-03-2014        38
## 3  4141       1965 Bachelor   71613       0        0 21-08-2013        26
## 4  6182       1984 Bachelor   26646       1        0 10-02-2014        26
## 5  5324       1981 PhD        58293       1        0 19-01-2014        94
## 6  7446       1967 Master     62513       0        1 09-09-2013        16
## # i 18 more variables: MntWines <int>, MntFruits <int>, MntMeatProducts <int>,
## #   MntFishProducts <int>, MntSweetProducts <int>, MntGoldProds <int>,
## #   NumWebPurchases <int>, NumStorePurchases <int>, Complain <int>, Age <int>,
## #   MembershipDays <int>, EducationLevel <dbl>, Single <dbl>, Together <dbl>,
## #   Married <dbl>, Divorced <dbl>, Widow <dbl>, Marital_Status <chr>
```

```
# lets remove columns which we will no longer use:
# remove ID, Year_Birth, Dt_Customer, Education, Marital_Status
# and save it as df_sel

df_sel <- market_df %>%
  select(-ID, -Year_Birth, -Dt_Customer, -Education, -Marital_Status)
head(df_sel)
```

```
## # A tibble: 6 x 21
##    Income Kidhome Teenhome Recency MntWines MntFruits MntMeatProducts
##     <int>   <int>    <int>   <int>    <int>     <int>           <int>
## 1  58138       0        0      58      635        88             546
## 2  46344       1        1      38       11         1               6
## 3  71613       0        0      26      426        49             127
## 4  26646       1        0      26       11         4              20
## 5  58293       1        0      94      173        43             118
```

```
## 6  62513       0        1      16      520        42              98
## # i 14 more variables: MntFishProducts <int>, MntSweetProducts <int>,
## #   MntGoldProds <int>, NumWebPurchases <int>, NumStorePurchases <int>,
## #   Complain <int>, Age <int>, MembershipDays <int>, EducationLevel <dbl>,
## #   Single <dbl>, Together <dbl>, Married <dbl>, Divorced <dbl>, Widow <dbl>
```

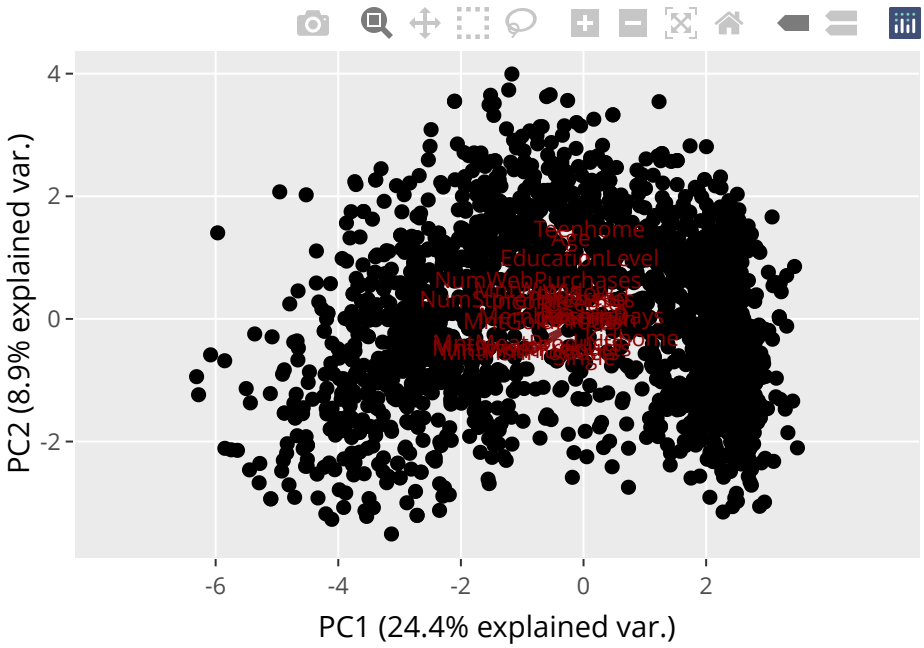
```
# lets scale (2 points)
# run scale function on df_sel and save it as df_scale
# that will be our scaled values which we will use for analysis
df_scale <- as.data.frame(scale(df_sel))
head(df_scale)
```

```
##       Income    Kidhome   Teenhome    Recency   MntWines  MntFruits
## 1  0.2339039 -0.8227362 -0.9281454  0.3082732  0.9766566  1.5488659
## 2 -0.2341403  1.0393789  0.9090170 -0.3826166 -0.8711997 -0.6370558
## 3  0.7686585 -0.8227362 -0.9281454 -0.7971505  0.3577432  0.5689700
## 4 -1.0158542  1.0393789 -0.9281454 -0.7971505 -0.8711997 -0.5616792
## 5  0.2400551  1.0393789 -0.9281454  1.5518749 -0.3914678  0.4182168
## 6  0.4075255 -0.8227362  0.9090170 -1.1425954  0.6361062  0.3930912
##   MntMeatProducts MntFishProducts MntSweetProducts MntGoldProds NumWebPurchases
## 1       1.6879549       2.4630607      1.481888649   0.85482704       1.4304941
## 2      -0.7180699      -0.6514171     -0.634215838  -0.73267383      -1.1252228
## 3      -0.1789421       1.3455128     -0.147755036  -0.03572223       1.4304941
## 4      -0.6556915      -0.5048534     -0.585569758  -0.75203360      -0.7601204
## 5      -0.2190425       0.1546831     -0.001816796  -0.55843593       0.3351868
## 6      -0.3081546      -0.6880580      0.363028806  -0.57779570       0.7002892
##   NumStorePurchases    Complain        Age MembershipDays EducationLevel
## 1        -0.5538715 -0.09794622  0.9853629      1.5300508     -0.4807422
## 2        -1.1683880 -0.09794622  1.2357656     -1.1889072     -0.4807422
## 3         1.2896778 -0.09794622  0.3176225     -0.2051387     -0.4807422
## 4        -0.5538715 -0.09794622 -1.2682609     -1.0603746     -0.4807422
## 5         0.0606449 -0.09794622 -1.0178582     -0.9516163      1.6431770
## 6         1.2896778 -0.09794622  0.1506875     -0.2990664      0.3688254
##       Single   Together    Married   Divorced      Widow
## 1  1.9205079 -0.5916806 -0.7959829 -0.3424856 -0.1887179
## 2  1.9205079 -0.5916806 -0.7959829 -0.3424856 -0.1887179
## 3 -0.5204599  1.6893359 -0.7959829 -0.3424856 -0.1887179
## 4 -0.5204599  1.6893359 -0.7959829 -0.3424856 -0.1887179
## 5 -0.5204599 -0.5916806  1.2557397 -0.3424856 -0.1887179
## 6 -0.5204599  1.6893359 -0.7959829 -0.3424856 -0.1887179
```

PCA

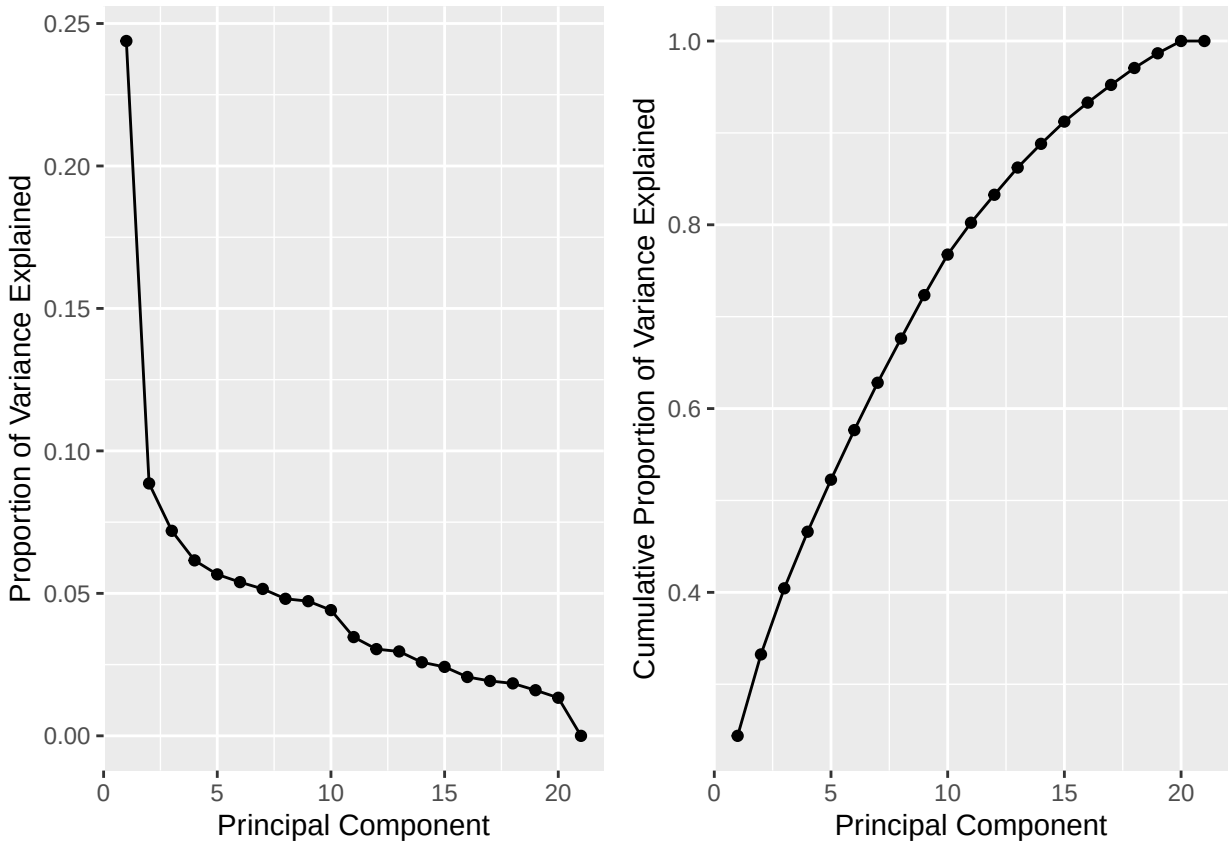
Q2.2. Run PCA, make biplot and scree plot *(6 points)*

```
# Run PCA on df_scale, make biplot and scree plot/percentage variance explained plot
# save as pc_out, we will use pc_out$x[,1] and pc_out$x[,2] later for plotting

pc_out <- prcomp(df_scale, center = TRUE, scale. = TRUE)
ggplotly(ggbiplot(pc_out, scale = 0,labels = pc_out$x %>% rownames()))
```

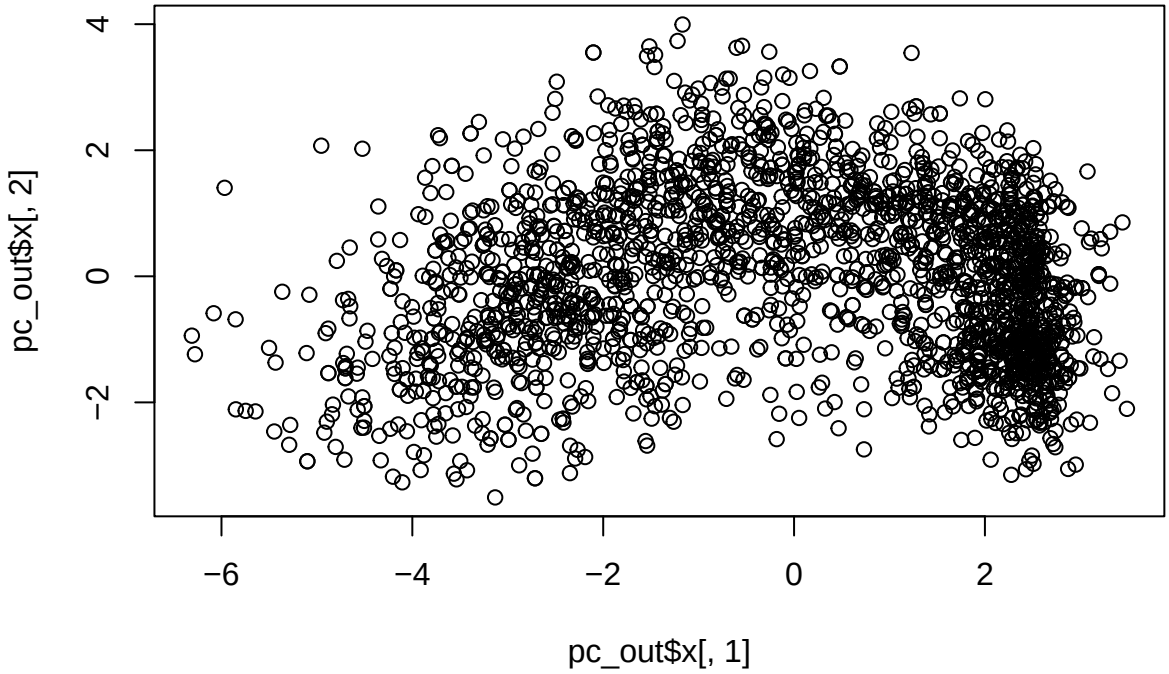



```
PVE <- tibble(
  PC=1:length(pc_out$sdev),
  Var=pc_out$sdev^2,
  PVE=Var/sum(Var),
  CumPVE=cumsum(PVE)
)
cowplot::plot_grid(
qplot(data=PVE,x=PC,y=PVE,geom=c("point","line"),
      xlab = "Principal Component",
      ylab = "Proportion of Variance Explained"),
qplot(data=PVE,x=PC,y=CumPVE,geom=c("point","line"),
xlab = "Principal Component",
ylab = "Cumulative Proportion of Variance Explained")
)
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
plot(pc_out$x[,1], pc_out$x[,2])
```



Q2.3 Comment on observation (any visible distinct clusters?) *(2 points)* Though there is a overlap I can see two clsters can maybe formed.

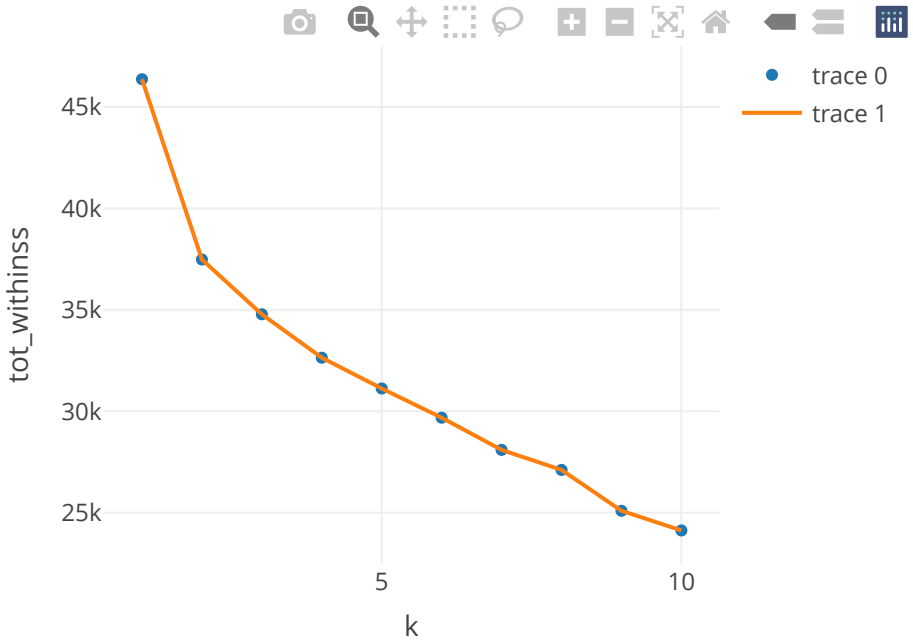
Cluster with K-Means

In questions Q2.4 to Q2.9 use K-Means method for clustering

Selecting Number of Clusters

Q2.4 Select optimal number of clusters using elbow method. *(4 points)*

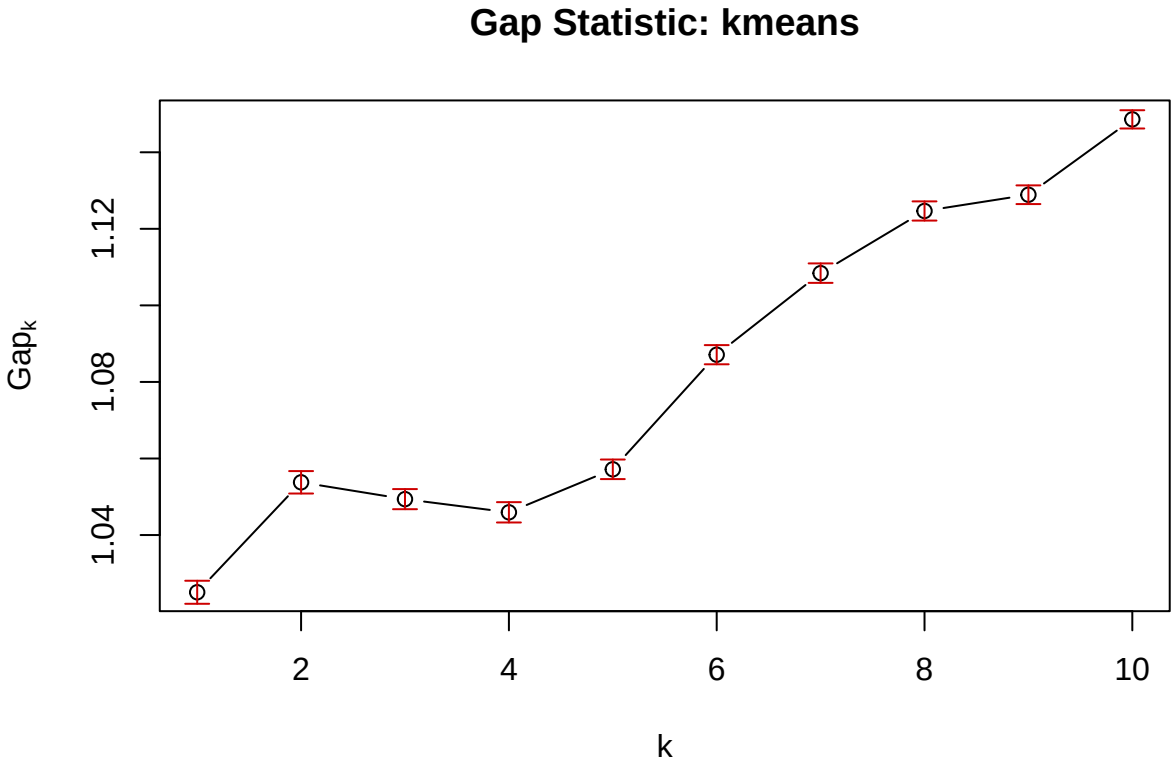
```
km_out_list <- lapply(1:10, function(k) list(
   k=k,
   km_out=kmeans(df_scale, k, nstart = 20)))
km_results <- data.frame(
k=sapply(km_out_list, function(k) k$k),
totss=sapply(km_out_list, function(k) k$km_out$totss), tot_withinss=sapply(km_out_list, function(k) k$km
plot_ly(km_results,x=~k,y=~tot_withinss) %>% add_markers() %>% add_paths()
```



We can see sharp dip at two 2 clusters. But as it has no significant meaning because of domain knowledge we will go with other dips where we find elbows 7 and 9. May be as suggested previously 9 clusters can be formed.

Q2.5 Select optimal number of clusters using Gap Statistic. *(4 points)*

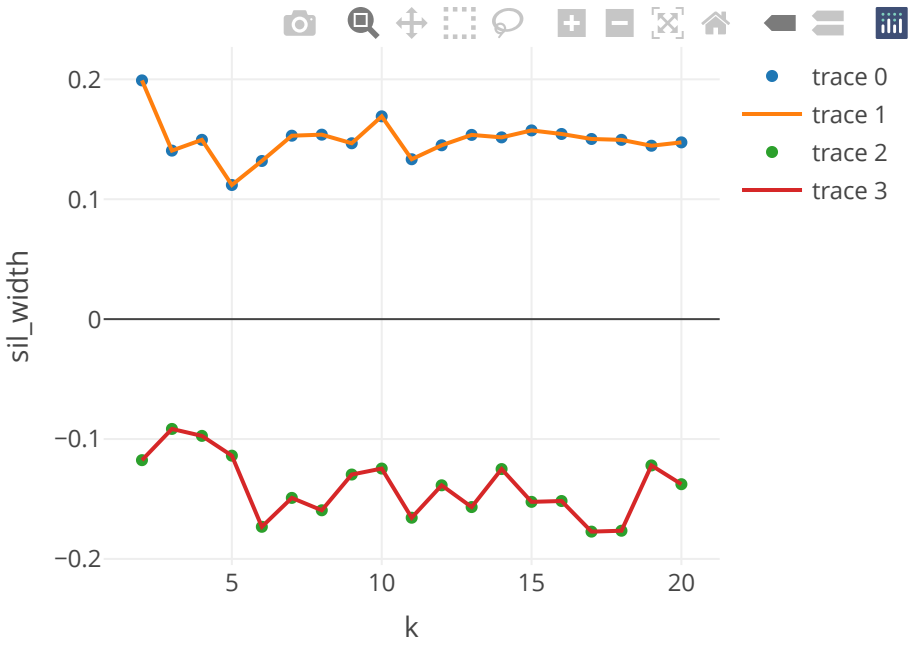
```
suppressWarnings({
   gap_kmeans <- clusGap(df_scale, kmeans, nstart = 20, K.max = 10, B = 100)
 })
 plot(gap_kmeans, main = "Gap Statistic: kmeans")
```



At 2 we have local maximum, so we can select 2 clusters as suggested above, but after that the Gap is continuous to increase there is no definitive peak to decide.

Q2.6 Select optimal number of clusters using Silhouette method. *(4 points)*

```
results <- lapply(2:20, function(k) {
   kmeans_cluster <- kmeans(df_scale, k, nstart=20)
   si <- silhouette(kmeans_cluster$cluster, dist = dist(df_scale))
   data.frame(k=k,sil_width=mean(si[,'sil_width']),sil_width_min=min(si[,'sil_width']))
 })
 si_df <- bind_rows(results)
 plot_ly(si_df, x=~k,y=~sil_width) %>%
   add_markers() %>% add_lines() %>%
   add_markers(y=~sil_width_min) %>% add_lines(y=~sil_width_min)
```



At 2 we have local maximum, so we can select 2 clusters as suggested above.

Q2.7 Which k will you choose based on elbow, gap statistics and silhouette as well as clustering task (market segmentation for advertisement purposes, that is two groups don't provide sufficient benefit over a single groups)?*(4 points)*

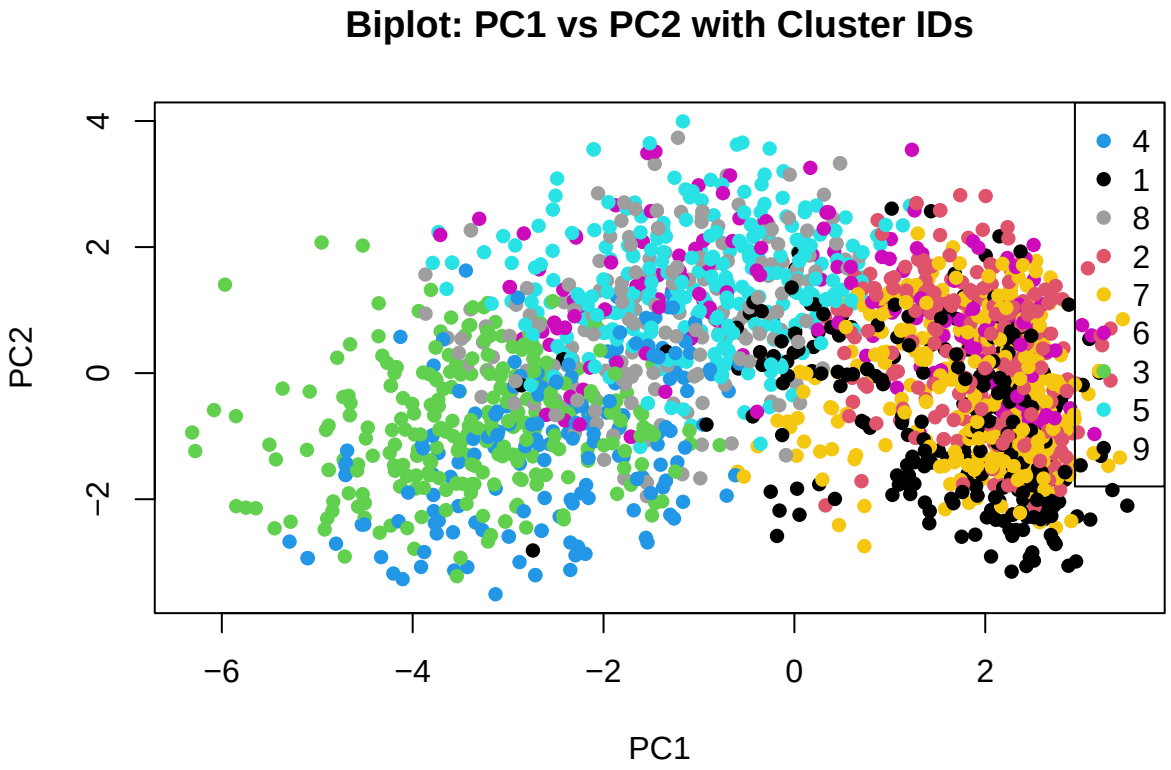
I feel like all there methods agree on 2 being the value of K. But as domain knowledge says having 2 clusters doesn't serve any purpose we will go with 9 clusters.

Though there are some local maximums for 9, 14 and 19 in silhouette method, we can still go with 9 as all three methods agreed on it.

Clusters Visulalization

Q2.8 Make k-Means clusters with selected k_kmeans (store result as km_out). Plot your k_kmeans clusters on biplot (just PC1 vs PC2) by coloring points by their cluster id.*(4 points)*

```
k_kmeans <- 9
 km_out <- kmeans(df_scale, centers = k_kmeans, nstart = 20)
 cluster_ids <- km_out$cluster

 plot(pc_out$x[, 1], pc_out$x[, 2], col = cluster_ids, pch = 16, xlab = "PC1", ylab = "PC2", main = "Bi
 legend("topright", legend = unique(cluster_ids), col = unique(cluster_ids), pch = 16)
```



Q2.9 Do you see any grouping? Comment on you observation.*(2 points)*

Answer Actually not. Everything seems pretty clumsy in 2 dimension may be in 3 dimensions we might actually see the clusters.

Characterizing Cluster

Q2.10 Perform descriptive statistics analysis on obtained cluster. Based on that does one or more group have a distinct characteristics? *(8 points)* Hint: add cluster column to original df dataframe

```
market_df$cluster<-as.factor(km_out$cluster)
 numerical_df <- market_df %>%
   select(-ID, -Year_Birth) %>%
   select_if(is.numeric)
 numerical_df$cluster <- market_df$cluster
 head(numerical_df)
```

```
## # A tibble: 6 x 22
##   Income Kidhome Teenhome Recency MntWines MntFruits MntMeatProducts
##    <int>   <int>    <int>   <int>    <int>     <int>           <int>
## 1  58138       0        0      58      635        88             546
## 2  46344       1        1      38       11         1               6
## 3  71613       0        0      26      426        49             127
## 4  26646       1        0      26       11         4              20
## 5  58293       1        0      94      173        43             118
## 6  62513       0        1      16      520        42              98
## # i 15 more variables: MntFishProducts <int>, MntSweetProducts <int>,
## #   MntGoldProds <int>, NumWebPurchases <int>, NumStorePurchases <int>,
## #   Complain <int>, Age <int>, MembershipDays <int>, EducationLevel <dbl>,
## #   Single <dbl>, Together <dbl>, Married <dbl>, Divorced <dbl>, Widow <dbl>,
## #   cluster <fct>
```

```r
cluster_stats_mean <- numerical_df %>%
  group_by(cluster) %>%
  summarise_all(list(mean = mean))
# View the statistics
print(cluster_stats_mean)
```

```
## # A tibble: 9 x 22
##   cluster Income_mean Kidhome_mean Teenhome_mean Recency_mean MntWines_mean
##   <fct>         <dbl>        <dbl>         <dbl>        <dbl>         <dbl>
## 1 1            36845.       0.729         0.473         47.6          84.5
## 2 2            36670.       0.794         0.551         51.3          54.3
## 3 3            77470.       0.0510        0.221         48.4         546.
## 4 4            73016.       0.04          0.234         52.0         597.
## 5 5            61752.       0.134         0.860         48.9         566.
## 6 6            49390.       0.472         0.633         49.4         288.
## 7 7            35878.       0.791         0.433         48.0          54.9
## 8 8            64669.       0.133         0.627         48.0         534.
## 9 9            45242.       0.667         0.524         53.0         169
## # i 16 more variables: MntFruits_mean <dbl>, MntMeatProducts_mean <dbl>,
## #   MntFishProducts_mean <dbl>, MntSweetProducts_mean <dbl>,
## #   MntGoldProds_mean <dbl>, NumWebPurchases_mean <dbl>,
## #   NumStorePurchases_mean <dbl>, Complain_mean <dbl>, Age_mean <dbl>,
## #   MembershipDays_mean <dbl>, EducationLevel_mean <dbl>, Single_mean <dbl>,
## #   Together_mean <dbl>, Married_mean <dbl>, Divorced_mean <dbl>,
## #   Widow_mean <dbl>
```

```r
# Median of each numerical columns grouped by cluster ID
cluster_stats_median <- numerical_df %>%
  group_by(cluster) %>%
  summarise_all(list(median = median))
# View the statistics
print(cluster_stats_median)
```

```
## # A tibble: 9 x 22
##   cluster Income_median Kidhome_median Teenhome_median Recency_median
##   <fct>           <dbl>          <dbl>           <dbl>          <dbl>
## 1 1               36230              1               0             49
## 2 2               34984              1               1             52
## 3 3               76624              0               0             50
## 4 4               72679              0               0             56
## 5 5               61833              0               1             51
## 6 6               48948              0               1             51
## 7 7               35688              1               0             47
## 8 8               65777              0               1             48
## 9 9               38998              1               0             49
## # i 17 more variables: MntWines_median <dbl>, MntFruits_median <dbl>,
## #   MntMeatProducts_median <dbl>, MntFishProducts_median <dbl>,
## #   MntSweetProducts_median <dbl>, MntGoldProds_median <dbl>,
## #   NumWebPurchases_median <dbl>, NumStorePurchases_median <dbl>,
## #   Complain_median <dbl>, Age_median <dbl>, MembershipDays_median <dbl>,
## #   EducationLevel_median <dbl>, Single_median <dbl>, Together_median <dbl>,
## #   Married_median <dbl>, Divorced_median <dbl>, Widow_median <dbl>
```

```r
# Mode of each numerical columns grouped by cluster ID
calculate_mode <- function(x) {
```

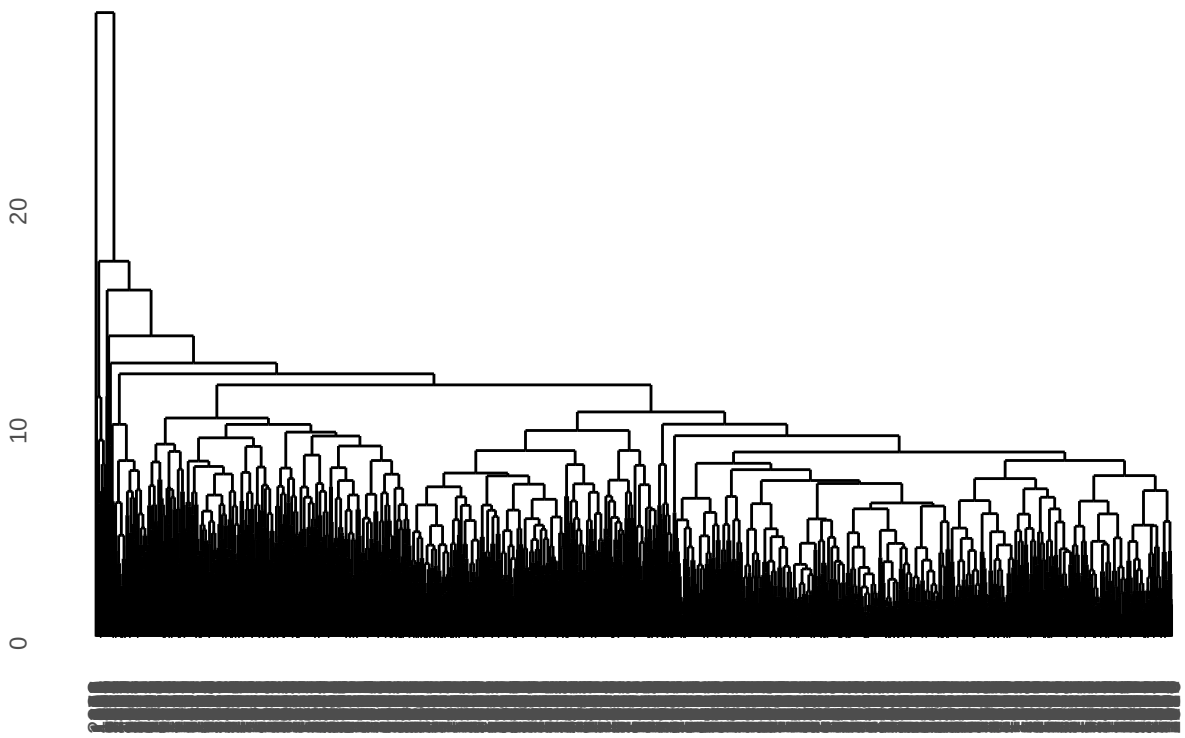
```
uniq_x <- unique(x)
uniq_x[which.max(tabulate(match(x, uniq_x)))]
}
cluster_stats_mode <- numerical_df %>%
  group_by(cluster) %>%
  summarise_all(list(mode = ~calculate_mode(.)))
# View the statistics
print(cluster_stats_mode)
```

```
## # A tibble: 9 x 22
##   cluster Income_mode Kidhome_mode Teenhome_mode Recency_mode MntWines_mode
##   <fct>         <int>        <int>         <int>        <int>         <int>
## 1 1              7500            1             0            2             4
## 2 2             30351            1             1           49             2
## 3 3             84618            0             0           54           398
## 4 4             82800            0             0           23           712
## 5 5             67445            0             1           72           656
## 6 6             63841            0             1           34             2
## 7 7              7500            1             0           30             5
## 8 8             83844            0             1           29           520
## 9 9             38998            1             0           92            16
## # i 16 more variables: MntFruits_mode <int>, MntMeatProducts_mode <int>,
## #   MntFishProducts_mode <int>, MntSweetProducts_mode <int>,
## #   MntGoldProds_mode <int>, NumWebPurchases_mode <int>,
## #   NumStorePurchases_mode <int>, Complain_mode <int>, Age_mode <int>,
## #   MembershipDays_mode <int>, EducationLevel_mode <dbl>, Single_mode <dbl>,
## #   Together_mode <dbl>, Married_mode <dbl>, Divorced_mode <dbl>,
## #   Widow_mode <dbl>
```

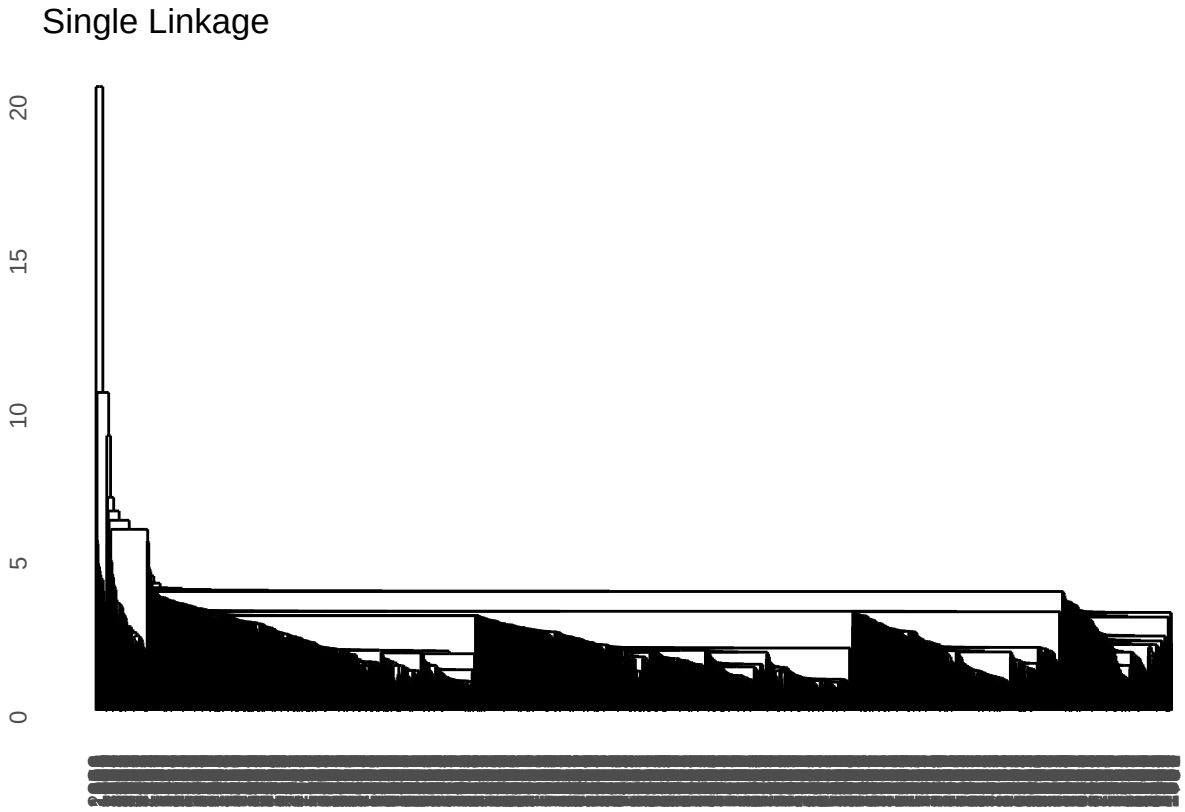
Cluster with Hierarchical Clustering

Q2.11 Perform clustering with Hierarchical method (Do you need to use scaling here?). Try complete, single and average linkage. Plot dendagram, based on it choose linkage and number of clusters, if possible, explain your choice. *(8 points)*

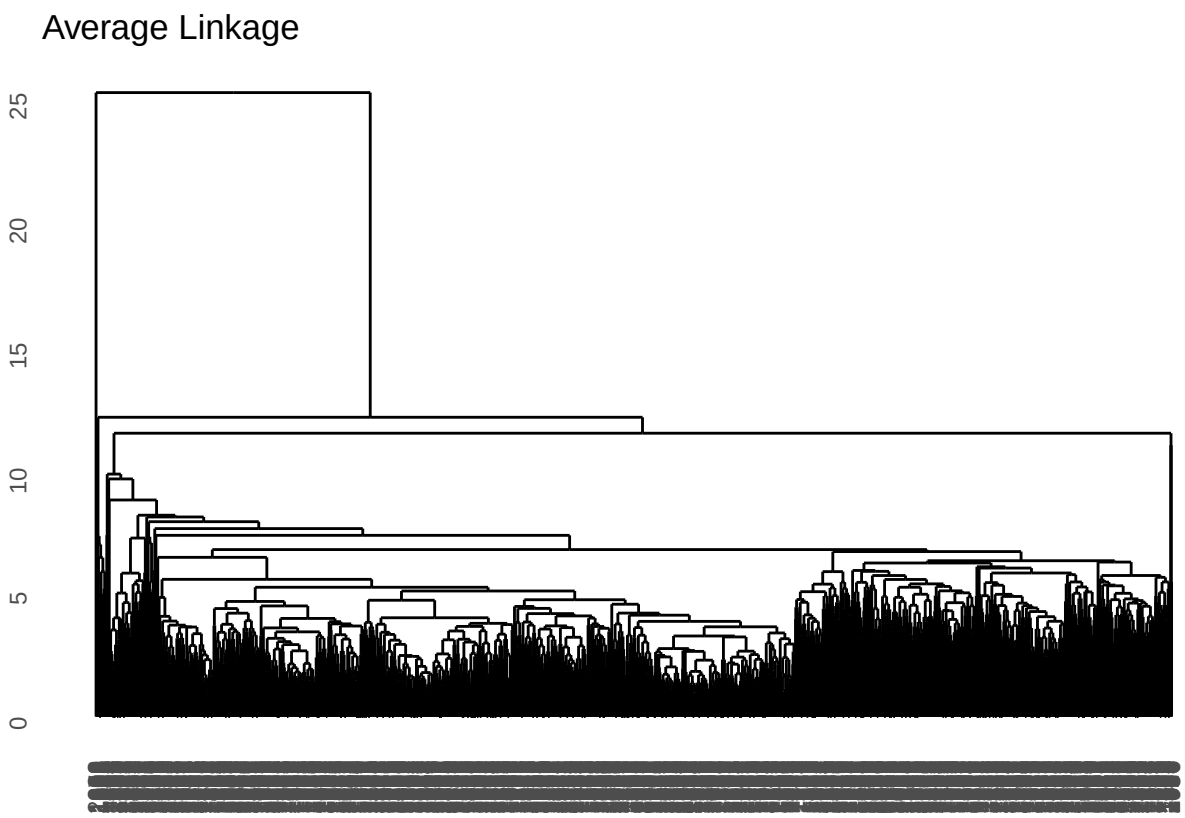
```
hc.complete <- hclust(dist(df_scale), method = "complete")
 ggdendrogram(hc.complete, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro =
  labs(title='Complete Linkage')
```




```
hc.single <- hclust(dist(df_scale), method = "single")
ggdendrogram(hc.single, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro = TR
 labs(title='Single Linkage')
```




```
hc.average <- hclust(dist(df_scale), method = "average")
 ggdendrogram(hc.average, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro =
  labs(title='Average Linkage')
```



All three Dendograms shows that they are highly imbalanced irrespective of the linkage. And also agrees that making 2 clusters is the best choice. But as 2 clusters doesn't have any significant meaning in the domain we can 9 clusters here too.

Additional grading criteria:

G3.1 Was all random methods properly seeded? *(2 points)*

Yes I set a global seed of 69 at the beginning of the RMD Files. So it the same seed will be used throughout the file.