Market Segmentation

Sejal Deepak Kankriya

2024-02-20

Market Segmentation

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

The 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

Read Dataset and Data Conversion to Proper Data Format

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

```r
# fread m_marketing_campaign.csv and save it as df
# Read the data
df <- fread("m_marketing_campaign.csv")
# Examine the data
head(df)
```

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

```r
# Convert Year_Birth to Age (assume that current date is 2014-07-01)
df[, Age := 2014 - Year_Birth]
df$Age <- as.integer(df$Age)

# Dt_Customer is a date (it is still character), convert it to membership days (i.e. number of days per
# Convert Dt_Customer to MembershipDays

df[, MembershipDays := as.Date("2014-07-01", format = "%Y-%m-%d") - as.Date(Dt_Customer, format = "%d-%
df$MembershipDays <- as.integer(df$MembershipDays)
```

```r
# Summarize Education column (use table function)
table(df$Education)
```

```
##
##  Associate   Bachelor HighSchool     Master        PhD
##        200       1114         54        363        478
```

```r
# Lets treat Education column as ordinal categories and use simple levels for distance calculations
# Assuming following order of degrees:
#    HighSchool, Associate, Bachelor, Master, PhD
# factorize Education column (hint: use factor function with above levels)
df$Education <- factor(df$Education, levels = c("HighSchool", "Associate", "Bachelor", "Master", "PhD")
```

```r
# Summarize Marital_Status column (use table function)
table(df$Marital_Status)
```

```
##
## Divorced  Married   Single Together    Widow
##      232      857      471      573       76
```

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

df$Marital_Status <- factor(df$Marital_Status)
dummy <- dummyVars(~ Marital_Status, data = df)
dummies_conv <- data.frame(predict(dummy, newdata = df))

# dummies
colnames(dummies_conv) <- c("Divorced", "Married", "Single", "Together", "Widow")
df <- cbind(df, dummies_conv)
```

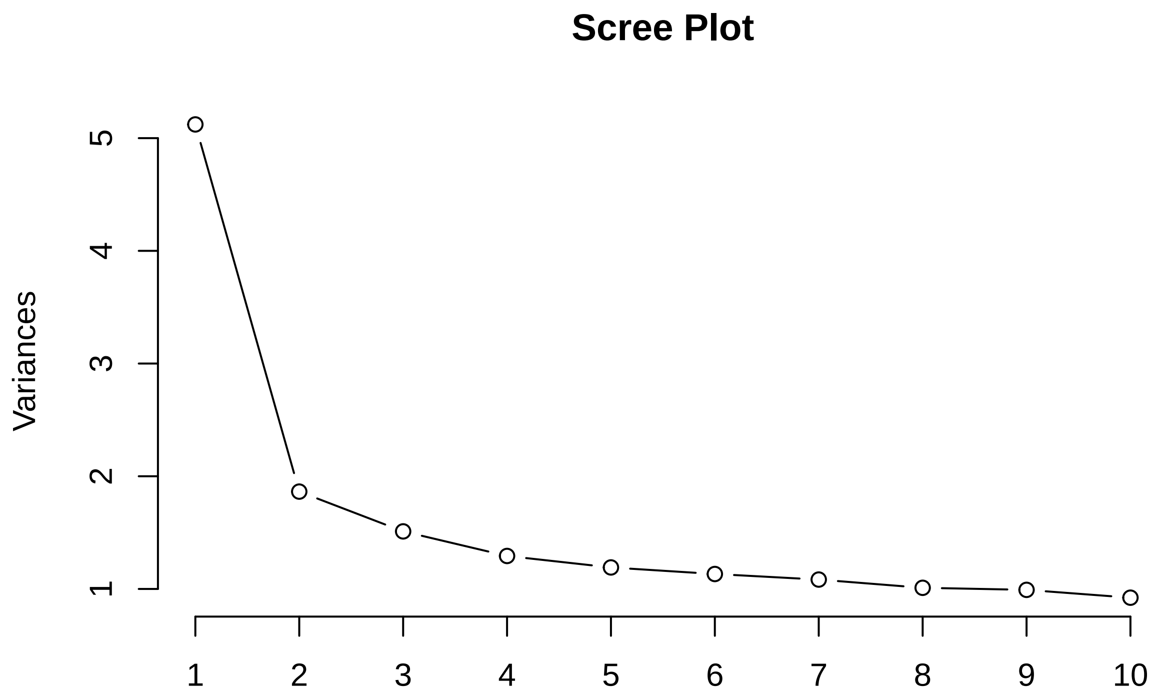
```r
# lets remove columns which we will no longer use:
# remove ID, Year_Birth, Dt_Customer, Marital_Status
# and save it as df_sel
# df_sel <- df %>% select(-ID, -Year_Birth, -Dt_Customer, -Marital_Status)

df_sel <- subset(df, select = -c(ID, Year_Birth, Dt_Customer, Marital_Status))

# Convert Education to integers
# use as.integer function, if you use factor function earlier
# properly then HighSchool will be 1, Associate will be 2 and so on)

df_sel$Education <- as.integer(df_sel$Education)
# head(df_sel)
```

```r
# lets scale
# 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 <- scale(df_sel)
# head(df_scale)
```

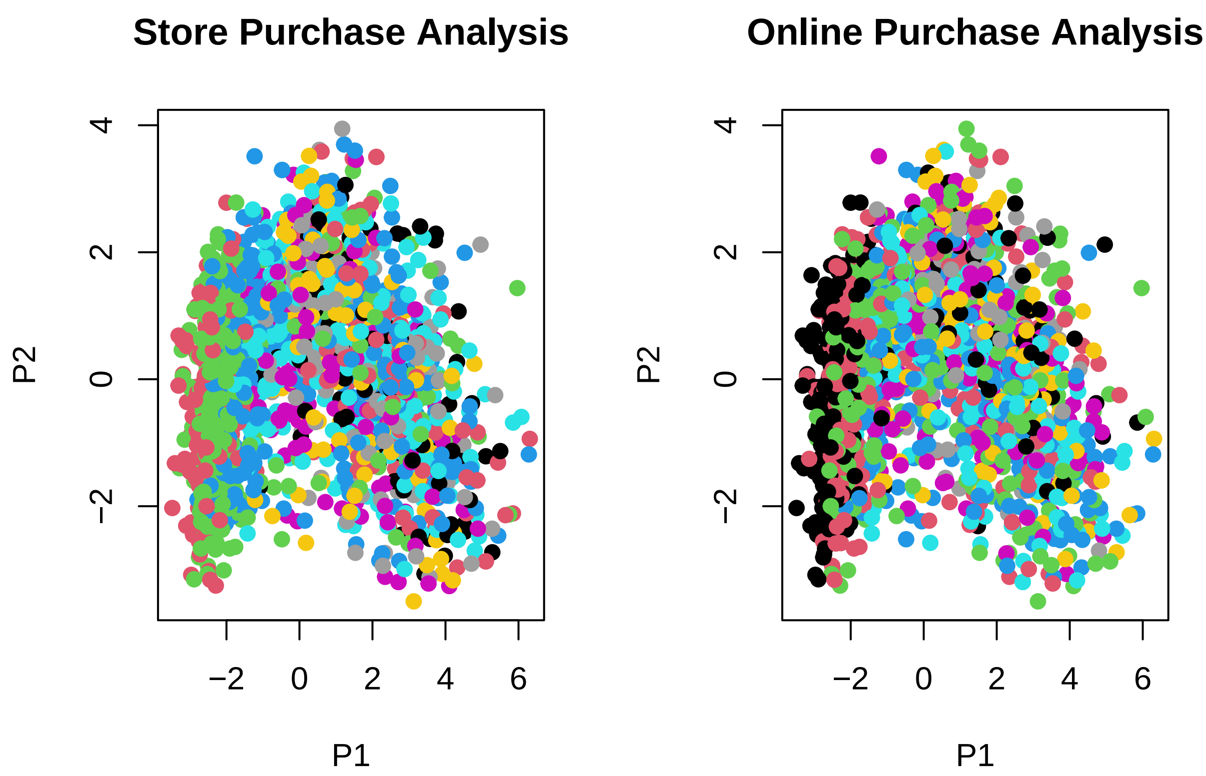
PCA

Run PCA

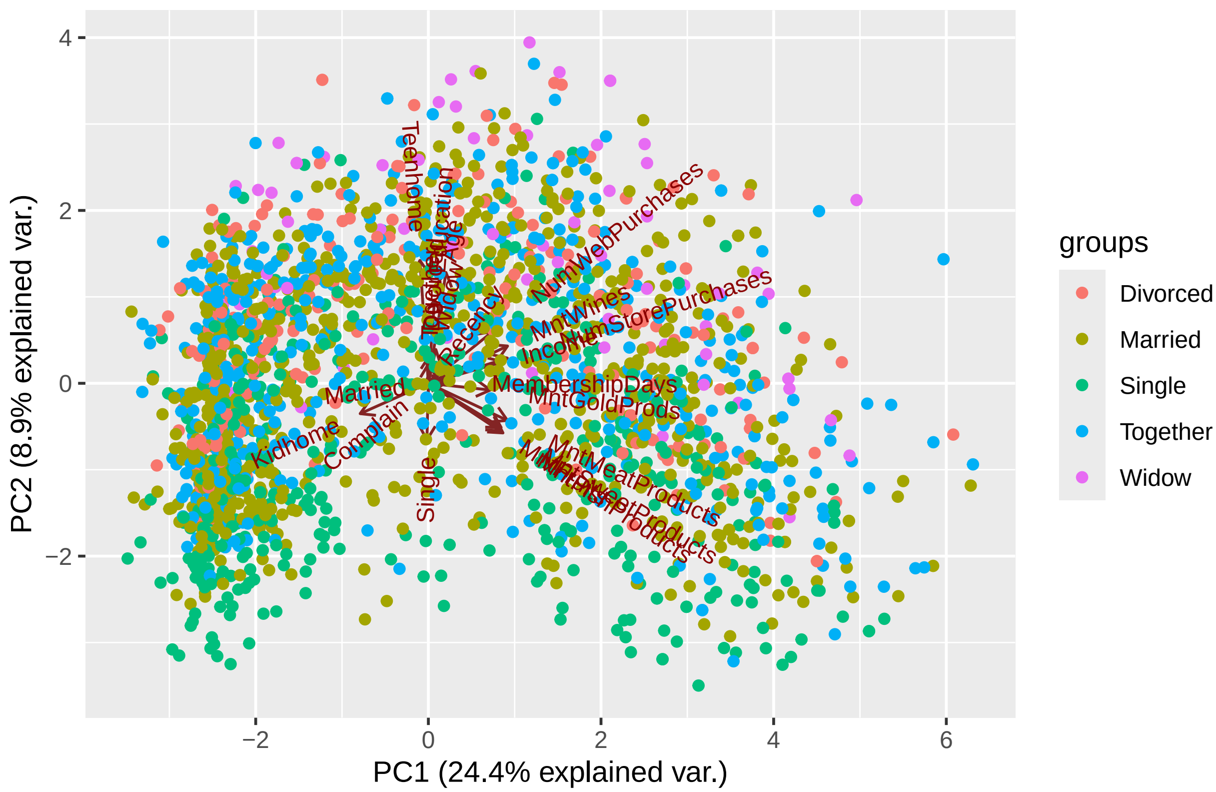
```r
# Run PCA on df_scale, make biplot and scree plot/percentage variance explained plot
# save as pc_out, we will use pc_out$x[,1] and pc_out$x[,2] later for plotting

pc_out <- prcomp(df_scale, scale = TRUE)
# pc_out

plot(pc_out, type="line", main="Scree Plot")
```



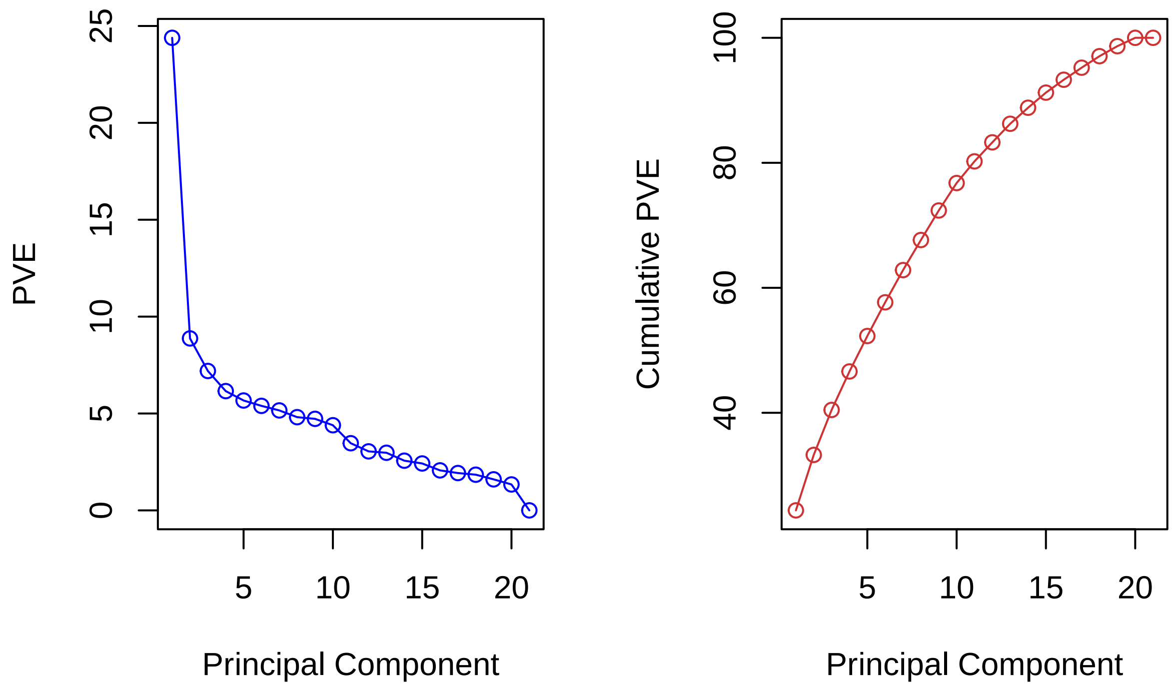

```r
# df_scale_hc <- hclust(dist(df_scale), method='complete')
# df_ct <- cutree(df_scale_hc, k=5)

par(mfrow = c(1, 2))
plot(x=pc_out$x[,1],y=pc_out$x[,2], col = df$NumStorePurchases, pch = 19,
    xlab = "P1", ylab = "P2", main = "Store Purchase Analysis")
plot(x=pc_out$x[,1],y=pc_out$x[,2], col = df$NumWebPurchases, pch = 19,
    xlab = "P1", ylab = "P2", main = "Online Purchase Analysis")
```




```r
# Biplot
ggbiplot(pc_out, scale=0, groups = as.factor(df$Marital_Status))
```




```
# percentage variance explained plot

pve <- 100 * pc_out$sdev^2 / sum(pc_out$sdev^2)
par(mfrow = c(1, 2))
plot(pve,  type = "o", ylab = "PVE",
    xlab = "Principal Component", col = "blue")
plot(cumsum(pve), type = "o", ylab = "Cumulative PVE",
    xlab = "Principal Component", col = "brown3")
```



The variables related to customer purchases (NumStorePurchases and NumWebPurchases) do not show any clear clusters or patterns. However, we can see some patterns in the variable Recency, which is a measure of how many days have passed since the customer's last purchase. According to the plotly function, there are three distinct groups of customers based on Recency, with different colors representing each group.

Furthermore, the scree plot shows that the first principal component explains more than 30% of the total variance, while the second principal component explains only about 20%. The first two principal components account for more than half of the total variance. This shows that these two components may be capturing some underlying elements or variables that are driving customer behavior. However, further research is required to discover these elements and comprehend how they relate to client purchases and other variables in the dataset.

Cluster with K-Means

use K-Means method for clustering

Selecting Number of Clusters

Select optimal number of clusters using elbow method.

```
set.seed(2)
km_out_list <- lapply(1:10, function(k) list(
  k=k,
  km_out=kmeans(df_scale, k, nstart = 20)))
```

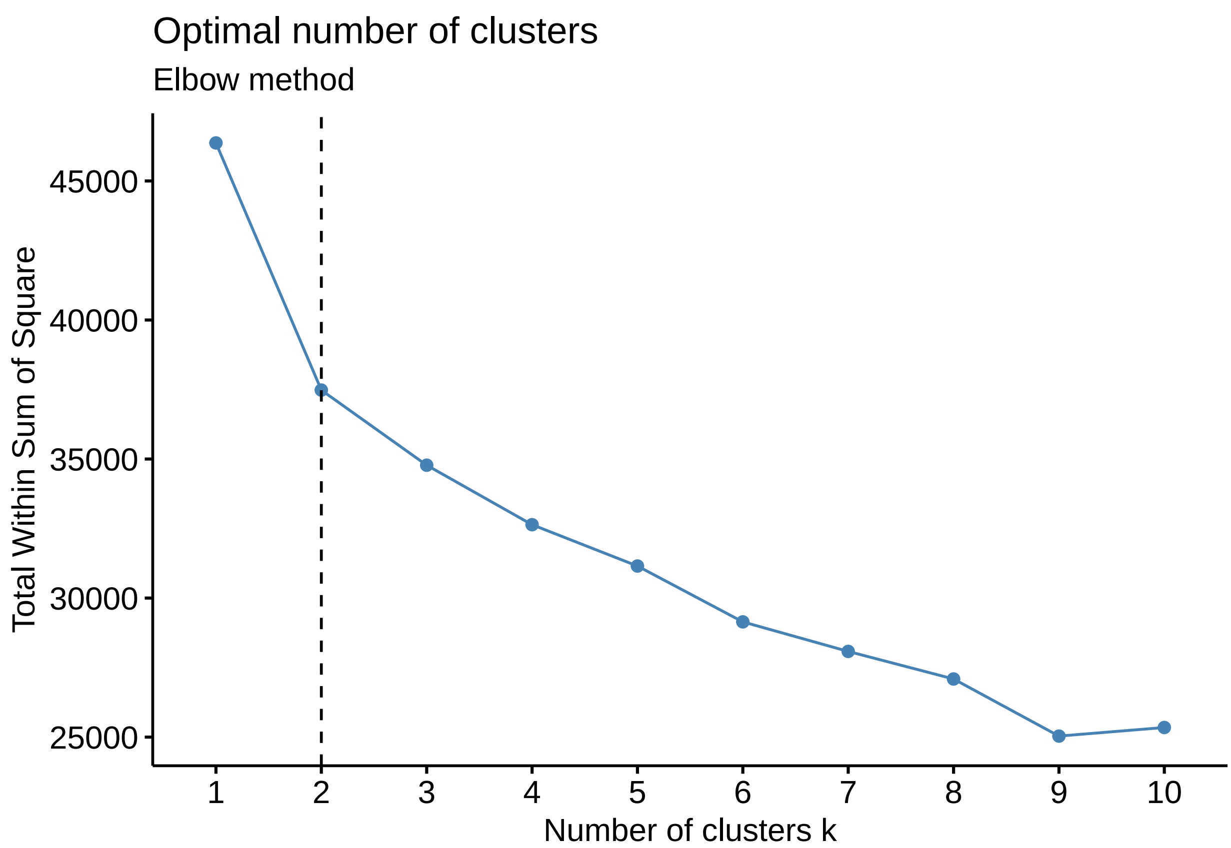
```r
km_results <- data.frame(
  k=sapply(km_out_list, function(k) k$k),
  totss=sapply(km_out_list, function(k) k$km_out$totss),
  tot_withinss=sapply(km_out_list, function(k) k$km_out$tot.withinss)
  )
km_results
```

```
##     k totss tot_withinss
## 1   1 46368     46368.00
## 2   2 46368     37479.33
## 3   3 46368     34778.20
## 4   4 46368     32802.66
## 5   5 46368     31240.88
## 6   6 46368     29646.63
## 7   7 46368     27543.27
## 8   8 46368     27107.45
## 9   9 46368     25036.27
## 10 10 46368     24551.77
```

```r
plot_ly(km_results,x=~k,y=~tot_withinss) %>% add_markers() %>% add_paths()
```

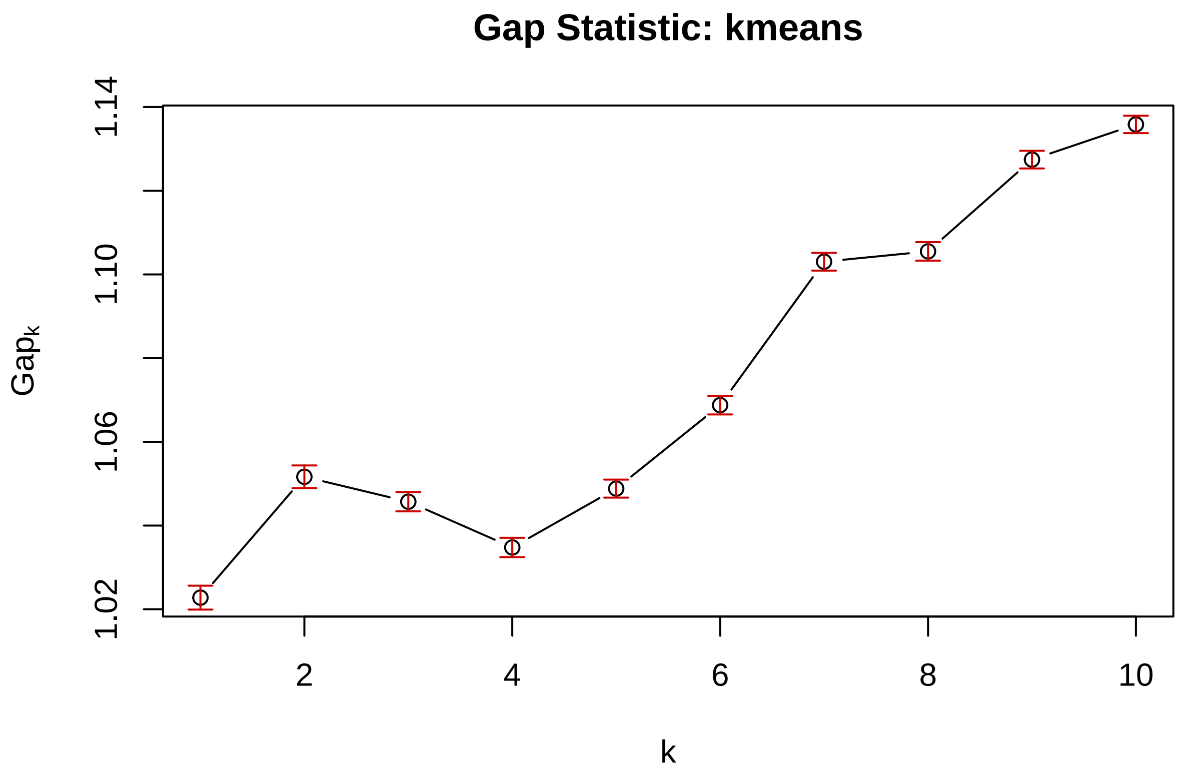
```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed, please
```

```r
# Elbow method
set.seed(4)
fviz_nbclust(df_scale, kmeans, method = "wss",k.max=10, nstart=20, iter.max=20) +
  geom_vline(xintercept = 2, linetype = 2)+
  labs(subtitle = "Elbow method")
```

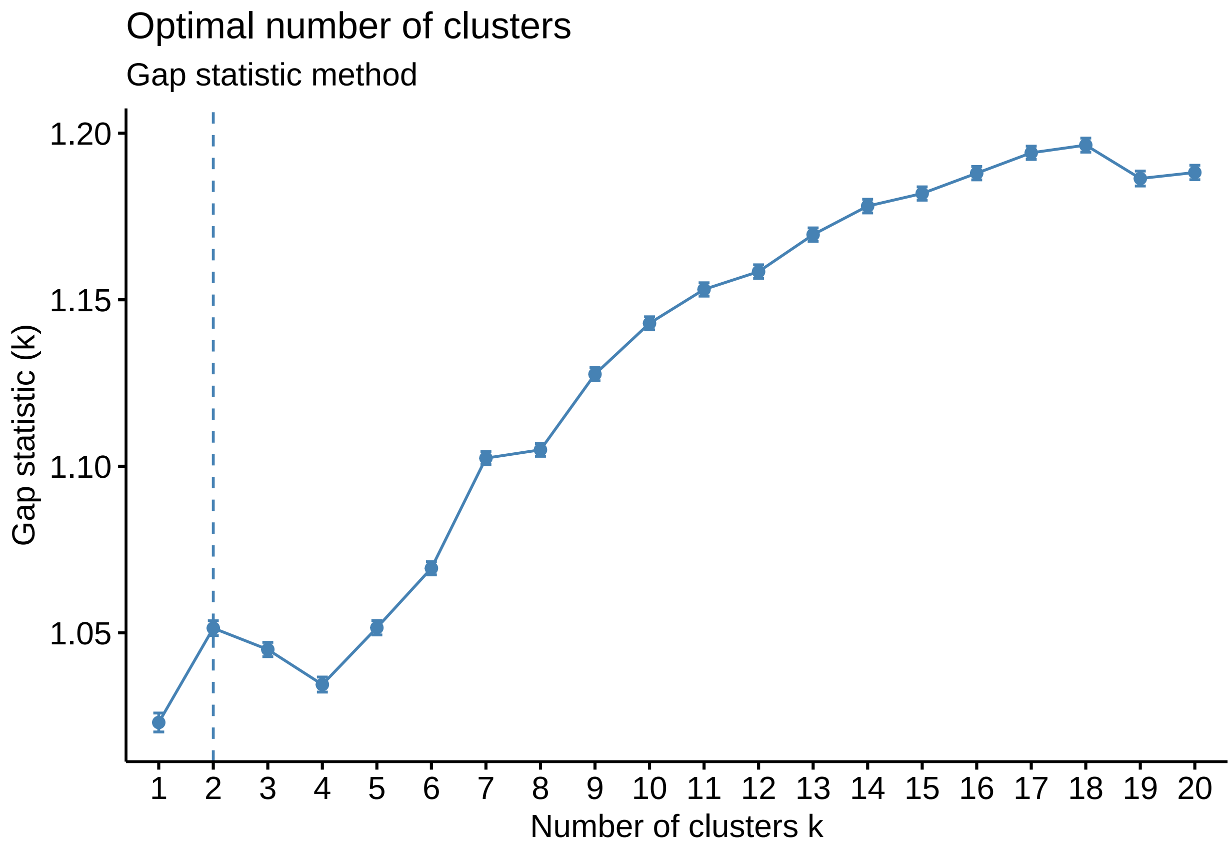


Select optimal number of clusters using Gap Statistic.

```
set.seed(3)
gap_kmeans <- clusGap(df_scale, kmeans, nstart = 20, K.max = 10, B = 100, iter.max=30)
plot(gap_kmeans, main = "Gap Statistic: kmeans")
```

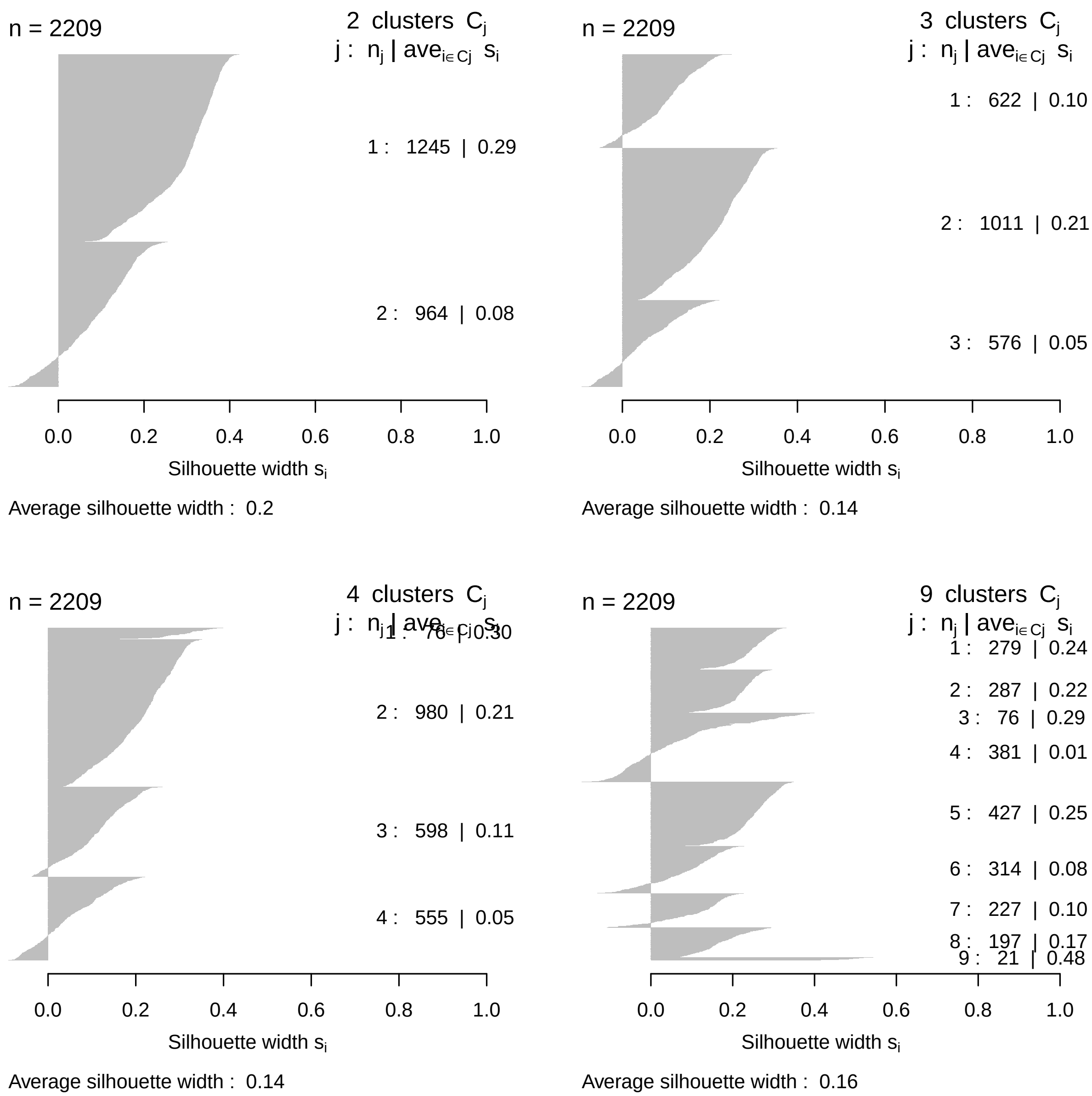



```
# Gap statistic
fviz_nbclust(df_scale, kmeans, method = "gap_stat", nboot = 20,k.max=20, nstart=20, iter.max=40) +
  labs(subtitle = "Gap statistic method")
```



Select optimal number of clusters using Silhouette method.

```
set.seed(5)

#Silhouette
par(mar = c(5, 2, 4, 2), mfrow=c(2,2))
for(k in c(2,3,4,9)) {
  kmeans_ct <- kmeans(df_scale, k, nstart=20)
  si <- silhouette(kmeans_ct$cluster, dist = dist(df_scale))
  plot(si, main="")
}
```

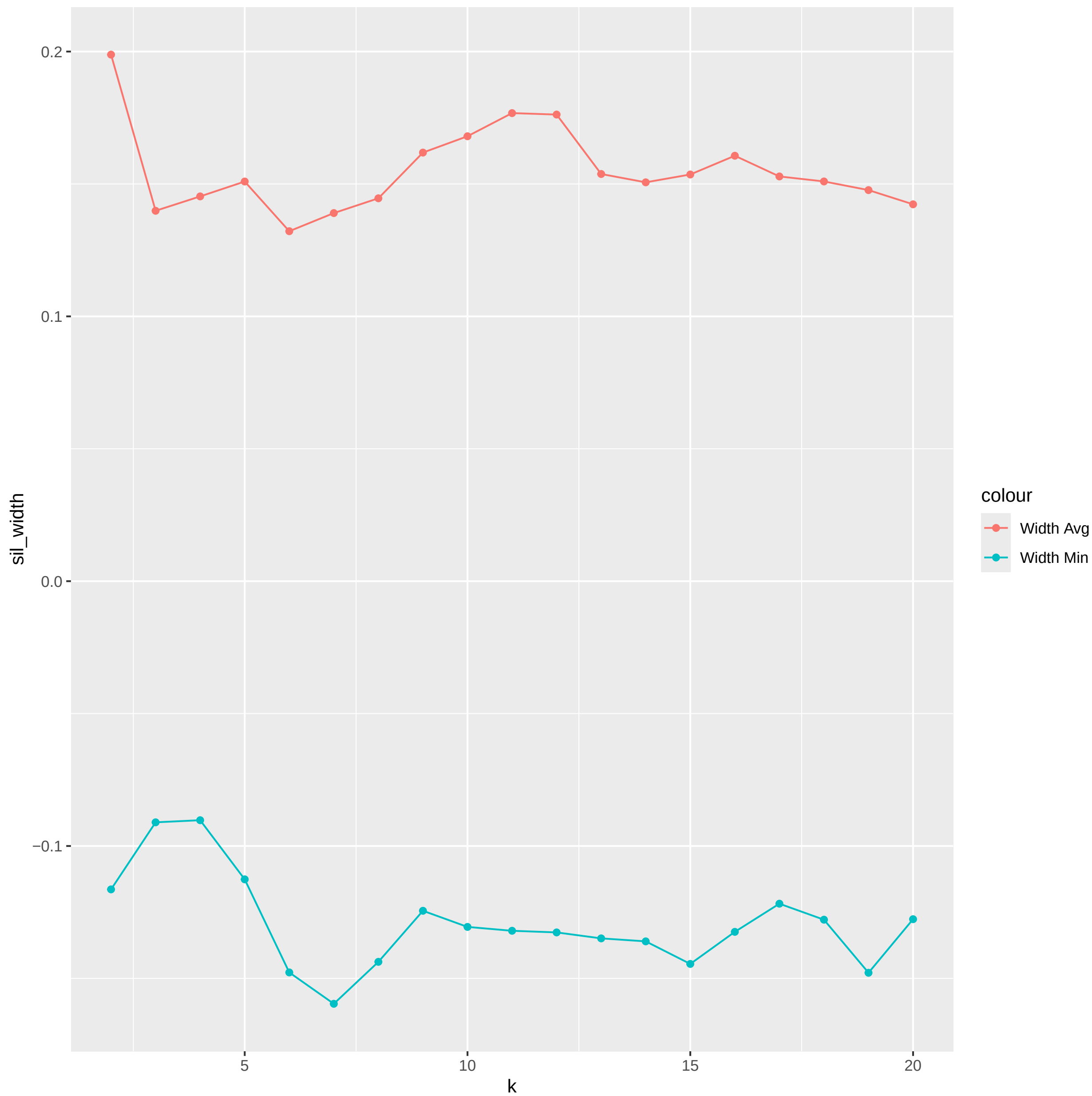



```
par(mar = c(1, 1, 1, 1), mfrow=c(1,1))
```

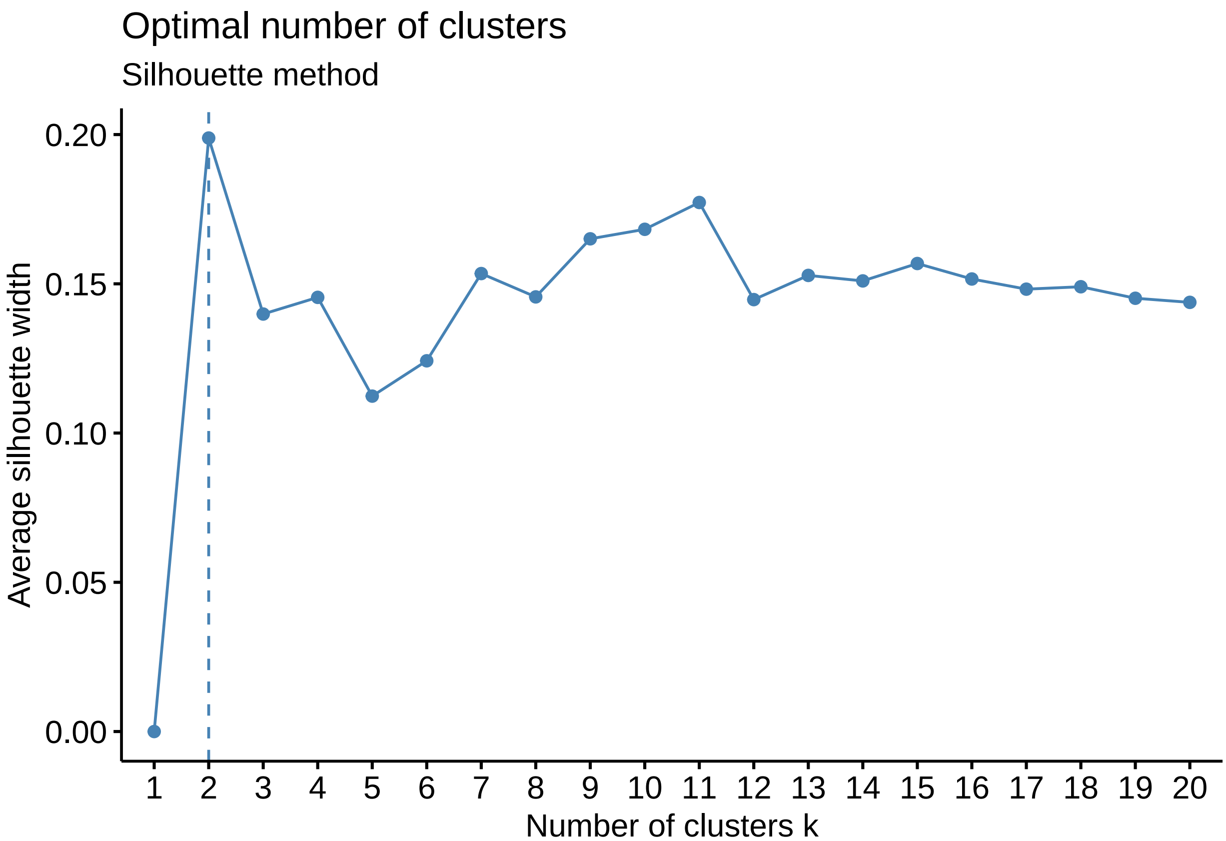
```
set.seed(1)

#Silhouette
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)
```

```
ggplot(si_df, aes(x=k,y=sil_width,color="Width Avg"))+geom_point()+geom_line()+
  geom_point(aes(y=sil_width_min,color="Width Min"))+geom_line(aes(y=sil_width_min,color="Width Min"))
```




```
fviz_nbclust(df_scale, kmeans, method = "silhouette", nboot = 20,k.max=20, nstart=20, iter.max=40)+
  labs(subtitle = "Silhouette method")
```



Deciding which k to choose based on elbow, gap statistics and silhuettes as well as clustering task (market segmentation for advertisement purposes, that is two groups don't provide sufficient benefit over a single groups).

It has been concluded that the ideal number of k is two, according to the elbow method, the gap method, and the silhouette method. However, it is not enough to segment customers into two segments. A clustering should be conducted with 3 as a value of k since, based on the elbow plot, gap statistics, and Silhouette methods, this is the second-most optimal number of k.

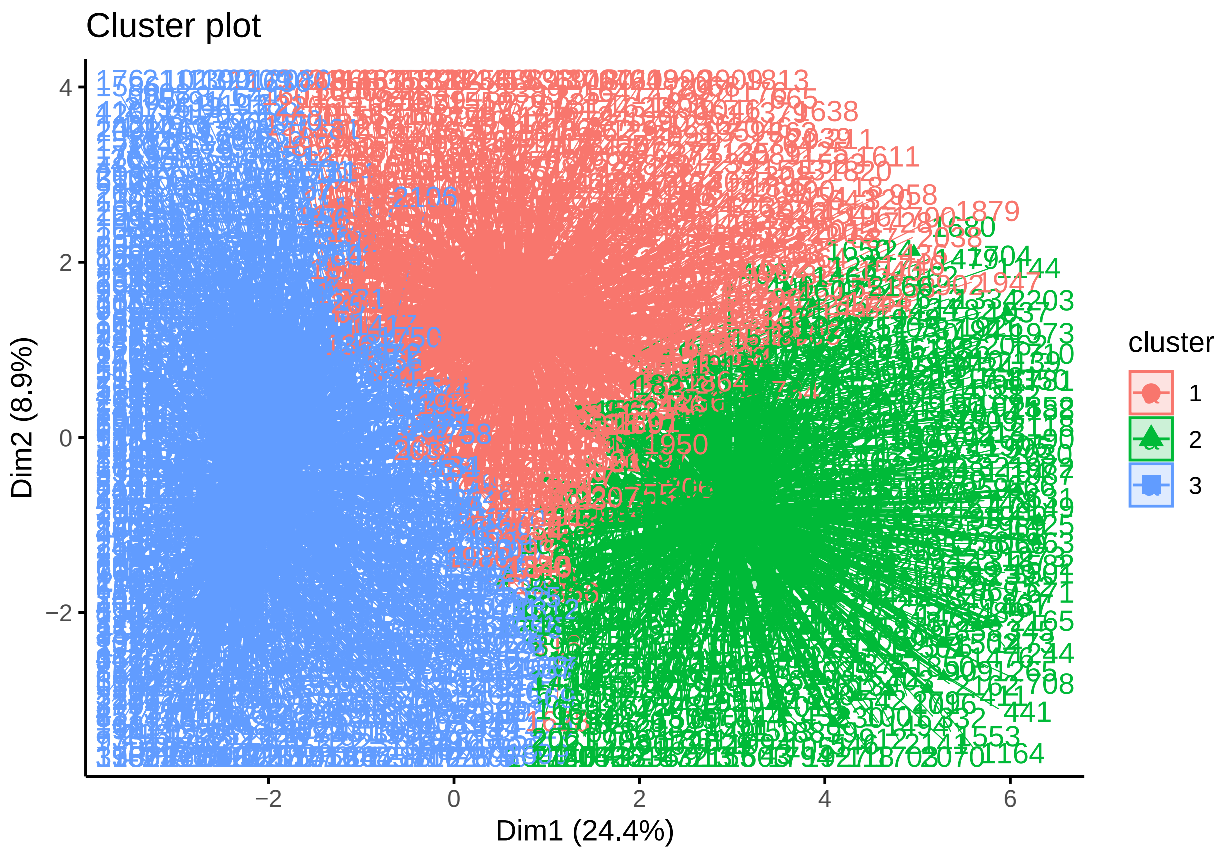
It is stated that two groups do not provide sufficient benefit over a single group for the market segmentation task, implying that at least three groups are required to obtain meaningful segmentation. As a result, based on the elbow, gap statistics, and silhouette results, as well as the clustering task requirements, it is recommended to select three clusters for the market segmentation analysis.

Clusters Visulalization

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.

```
# Computing kmeans with new selected k value

set.seed(6)
km_out <- kmeans(df_scale, 3, nstart = 25)
# km_out
```

```r
fviz_cluster(km_out, data = df_scale, ellipse.type = "euclid", star.plot = TRUE,
             repel = TRUE, ggtheme = theme_classic())
```



The k-means clustering algorithm identified three distinct groups based on the plot. The red cluster is located in the plot's upper right corner and is distinguished by high values of the variables represented by the x and y axes. This category may include customers who are high spenders and make frequent purchases.

The green cluster is located in the plot's bottom left corner and is distinguished by low values of the variables represented by the x and y axes. Customers in this category may be low spenders who make infrequent purchases.

The blue cluster is located in the top left corner of the plot and is distinguished by high x-axis values and low y-axis values. Customers in this category may be high spenders who make infrequent purchases.

Characterizing Cluster

Perform descriptive statistics analysis on obtained cluster. Based on that does one or more group have a distinct characteristics.

```r
# df$cluster <- as.factor(km_out$cluster)
df_sel <- df_sel %>%
  mutate(cluster = km_out$cluster)

stat_analysis <- df_sel %>%
  group_by(cluster) %>%
  summarise_all(list(mean = mean, sd = sd))
```

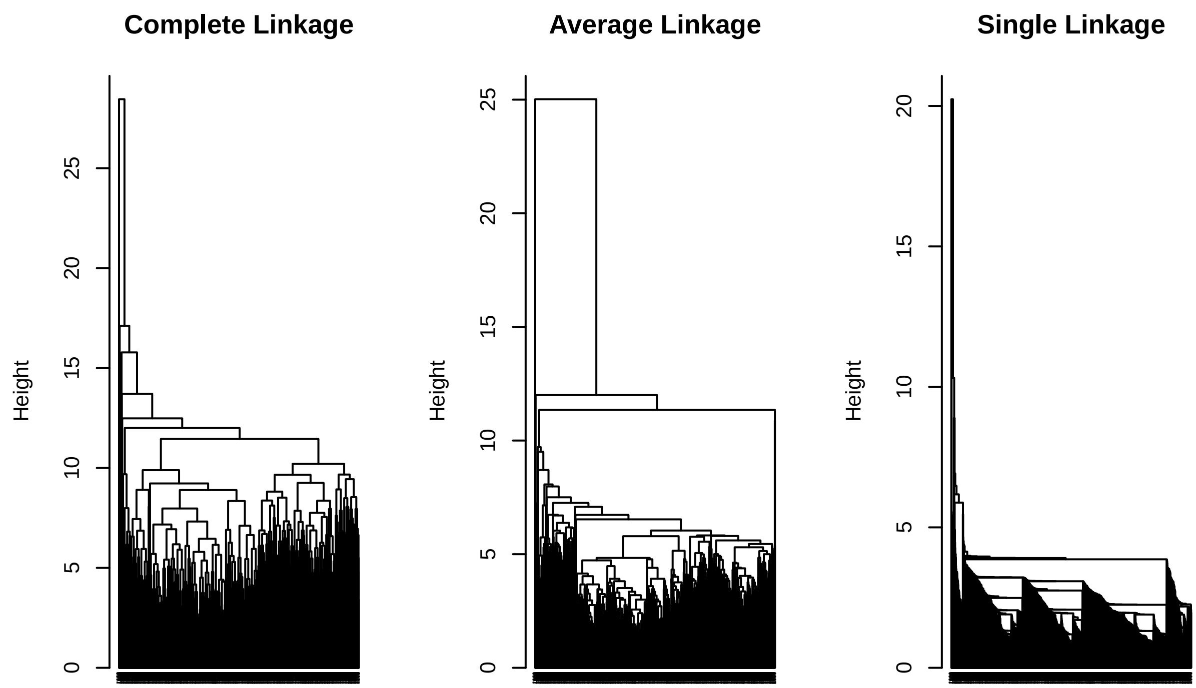
```r
print(stat_analysis)
```

```
## # A tibble: 3 x 43
##   cluster Education_mean Income_mean Kidhome_mean Teenhome_mean Recency_mean
##     <int>          <dbl>       <dbl>        <dbl>         <dbl>        <dbl>
## 1       1           3.82      59398.       0.144          0.886         47.3
## 2       2           3.33      76172.       0.0716         0.206         50.2
## 3       3           3.30      34712.       0.828          0.434         49.5
## # i 37 more variables: MntWines_mean <dbl>, 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>,
## #   Divorced_mean <dbl>, Married_mean <dbl>, Single_mean <dbl>,
## #   Together_mean <dbl>, Widow_mean <dbl>, Education_sd <dbl>, ...
```

Cluster with Hierarchical Clustering

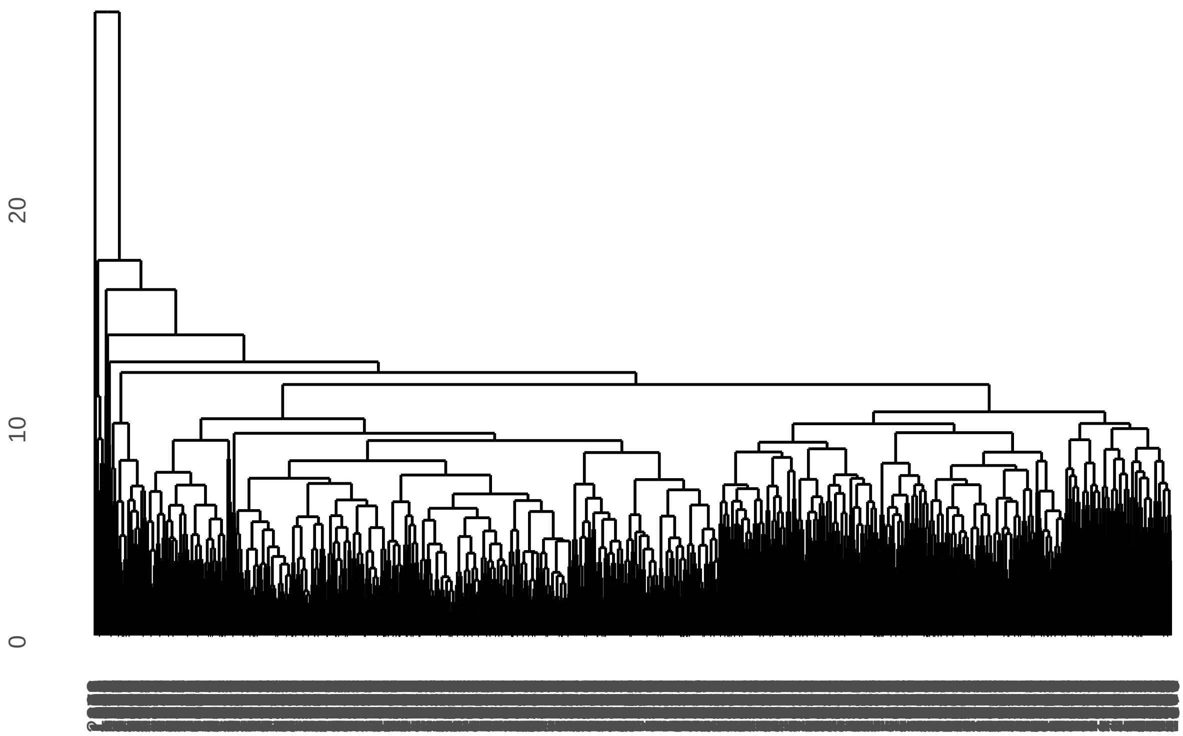
Perform clustering with Hierarchical method. Try complete, single and average linkage. Plot dendagram, based on it choose linkage and number of clusters.

```r
set.seed(6)
distance_matrix <- dist(df_scale, method = "euclidean")
hc_complete <- hclust(distance_matrix, method = "complete")
hc_avg <- hclust(distance_matrix, method = "average")
hc_single <- hclust(distance_matrix, method = "single")

par(mfrow = c(1, 3))
plot(hc_complete, main = "Complete Linkage",
    xlab = "", sub = "", cex = .3, hang = -1)
plot(hc_avg, main = "Average Linkage",
    xlab = "", sub = "", cex = .3, hang = -1)
plot(hc_single, main = "Single Linkage",
    xlab = "", sub = "", cex = .3, hang = -1)
```



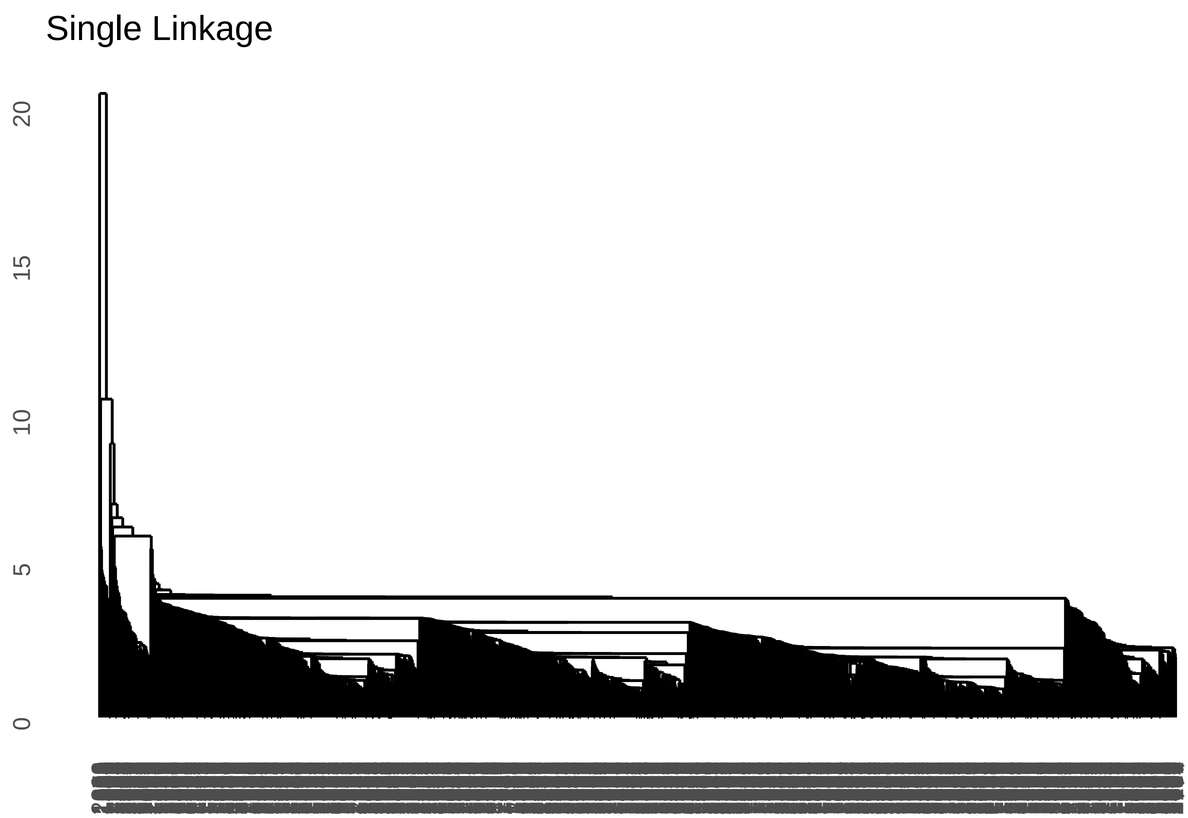

```
# ggplot
ggdendrogram(hc_complete, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro =
 labs(title='Complete Linkage')
```

Complete Linkage



```
ggdendrogram(hc_avg, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro = TRUE)
 labs(title='Average Linkage')
```


```
ggdendrogram(hc_single, segements=TRUE, labels=TRUE, leaf_labels = TRUE, rotate=FALSE, theme_dendro = T
 labs(title='Single Linkage')
```



In my opinion, scaling is essential in order to avoid bias in conclusions. In the dendrograms, the maximum number of distinct clusters is generated by the method of complete linkage, whereas the minimum number of distinct clusters is generated by the method of single linkage. Clusters with an average linkage are located in the middle.

If there are elbows in the data, this indicates a natural break in the data, which can be used to determine the number of clusters. A slight elbow can be observed around three clusters using the full and average linkage methods. In contrast, the single linkage method does not indicate a clear correlation between the two.