BA_Assignment1_Team

2024-10-05

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Normalize whole data

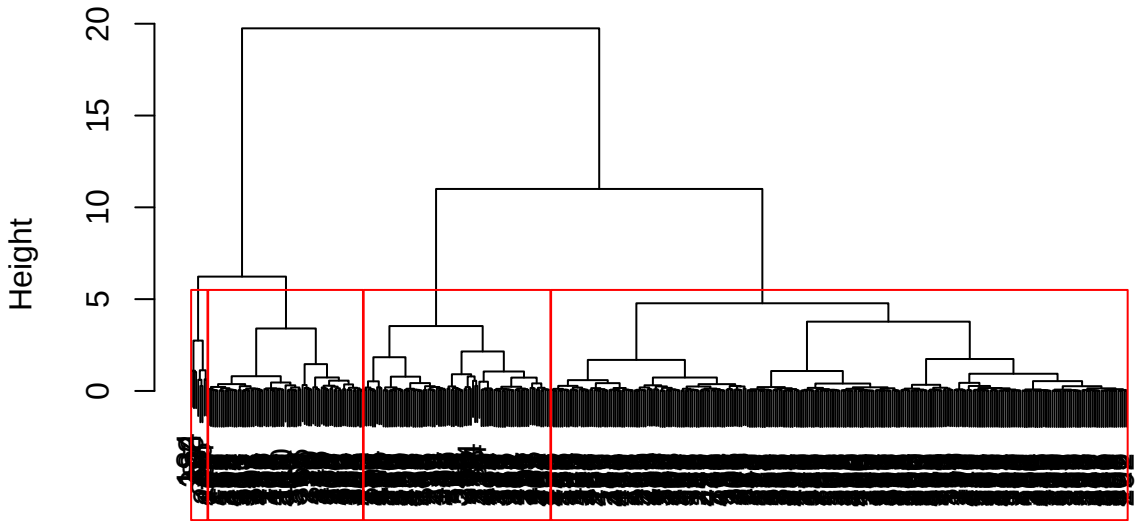
```
wholesale <- read.csv("Wholesale customers data.csv")

normalize = function(x){
  return((x-min(x))/(max(x)-min(x)))}

wholesale_normalized = wholesale %>% mutate_at(c(3:8), normalize)
```

Calculate the distance of whole data

```
wholesale_matrix = dist(wholesale_normalized[,3:8], method = "euclidean")
```

Hierarchical Clustering

```
hierarchical = hclust(wholesale_matrix, method = "ward.D")
plot(hierarchical, )
rect.hclust(hierarchical, k = 4)
```




```
wholesale_normalized$cluster = cutree(hierarchical, k = 4) # let's check out cluster centroids
wholesale_normalized %>% group_by(cluster) %>%
summarise_at(c(3:8), mean)
```

```
## # A tibble: 4 x 7
##   cluster  Fresh   Milk Grocery Frozen Detergents_Paper Delicatessen
##     <int>  <dbl>  <dbl>   <dbl>  <dbl>            <dbl>        <dbl>
## 1       1 0.0727 0.0439  0.0471 0.0318           0.0309       0.0218
## 2       2 0.243  0.0740  0.0627 0.110            0.0252       0.0480
## 3       3 0.0440 0.162   0.213  0.0229           0.227        0.0308
## 4       4 0.345  0.523   0.484  0.261            0.479        0.197
```

```
# Count the number of Retail and Horeca clients in each cluster
wholesale_normalized %>%
  group_by(cluster, Channel) %>%
  summarise(count = n()) %>%
  arrange(cluster)
```

```
## `summarise()` has grouped output by 'cluster'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 8 x 3
## # Groups:   cluster [4]
##   cluster Channel count
```

```
##     <int>   <int> <int>
## 1       1       1   217
## 2       1       2    54
## 3       2       1    74
## 4       2       2    14
## 5       3       1     4
## 6       3       2    69
## 7       4       1     3
## 8       4       2     5
```

```
# Count the number of regional clients in each cluster
wholesale_normalized %>%
  group_by(cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(cluster)
```

```
## `summarise()` has grouped output by 'cluster'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 11 x 3
## # Groups:   cluster [4]
##    cluster Region count
##      <int>  <int> <int>
##  1       1      1    50
##  2       1      2    28
##  3       1      3   193
##  4       2      1    14
##  5       2      2     7
##  6       2      3    67
##  7       3      1    13
##  8       3      2    10
##  9       3      3    50
## 10       4      2     2
## 11       4      3     6
```

=> We can assume K can be 3 or 4

Whole data K-Means Clustering (k = 4)

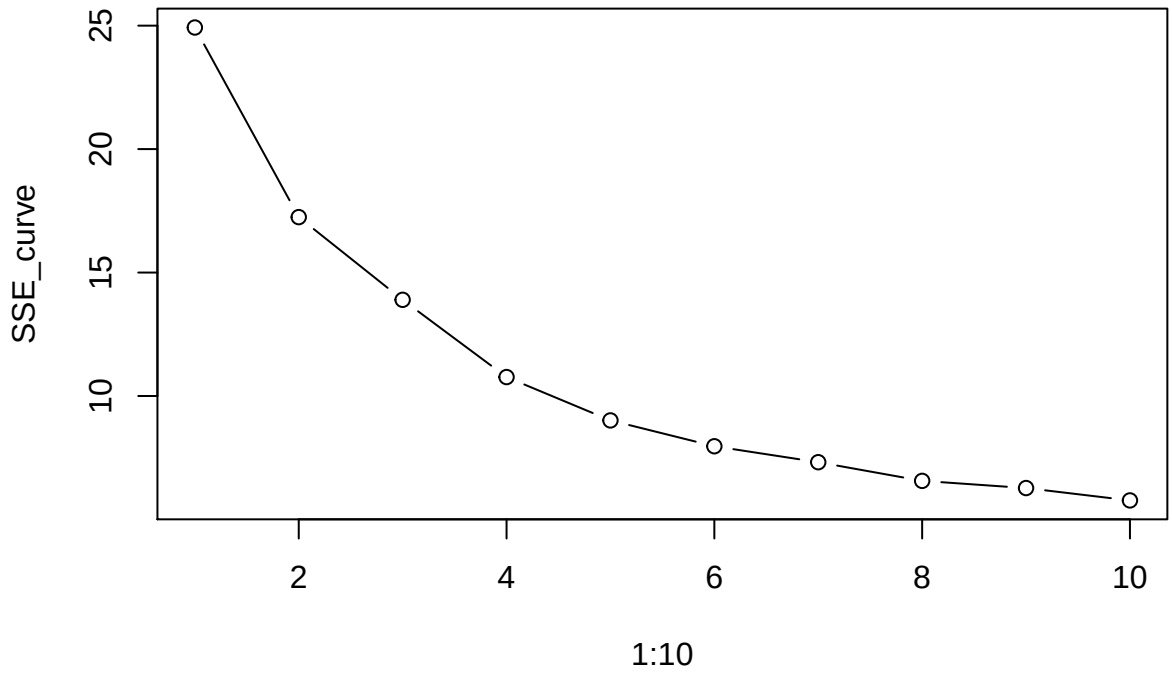
```
kcluster = kmeans(wholesale_normalized[,3:8], centers = 4)
kcluster$centers
```

```
##        Fresh       Milk    Grocery     Frozen Detergents_Paper Delicatessen
## 1 0.08396451 0.04147169 0.04173396 0.04340272       0.02436526   0.02204716
## 2 0.05181527 0.14567365 0.18337739 0.02360488       0.18012570   0.03889786
## 3 0.33971745 0.08543346 0.06919225 0.13961332       0.02373254   0.06913531
## 4 0.14232889 0.47184211 0.52312427 0.04979291       0.60925436   0.06132249
```

```
# Add the cluster assignments from k-means to the wholesale data
wholesale_normalized$kmeans_cluster <- kcluster$cluster
```

Whole data SSE Curve Evaluation (k = 4)

```
SSE_curve <- c()
for (n in 1:10) {
    kcluster = kmeans(wholesale_normalized[,3:8], n)
    sse = kcluster$tot.withinss
    SSE_curve[n] = sse}
plot(1:10, SSE_curve, type = "b")
```



Whole data Sihouette Coefficient Evaluation (k = 4)

```
library(cluster)
sc = silhouette(wholesale_normalized$cluster, dist = wholesale_matrix)
summary(sc)
```

```
## Silhouette of 440 units in 4 clusters from silhouette.default(x = wholesale_normalized$cluster, dist
##  Cluster sizes and average silhouette widths:
##         271          88          73           8
##  0.50647525  0.03415369  0.31031845 -0.09460692
## Individual silhouette widths:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.4102  0.2243  0.4302  0.3685  0.5877  0.6683
```

=> SSE Curve Evaluation seems ambiguous to tell k = 4, and there is a very small cluster (Cluster 4 with 8 members), which has a negative silhouette score (-0.09), indicating poor cluster quality. => Try K=3

Whole data K-Means Clustering (k = 3)

```r
kcluster3 = kmeans(wholesale_normalized[,3:8], centers = 3)
kcluster3$centers
```

```
##        Fresh       Milk    Grocery     Frozen Detergents_Paper Delicatessen
## 1 0.07612741 0.26781048 0.31604316 0.03156052       0.34516819   0.04830937
## 2 0.31612924 0.08086923 0.07042464 0.11874307       0.02650803   0.06815213
## 3 0.07297318 0.05468315 0.06047771 0.03993169       0.04513443   0.02316561
```

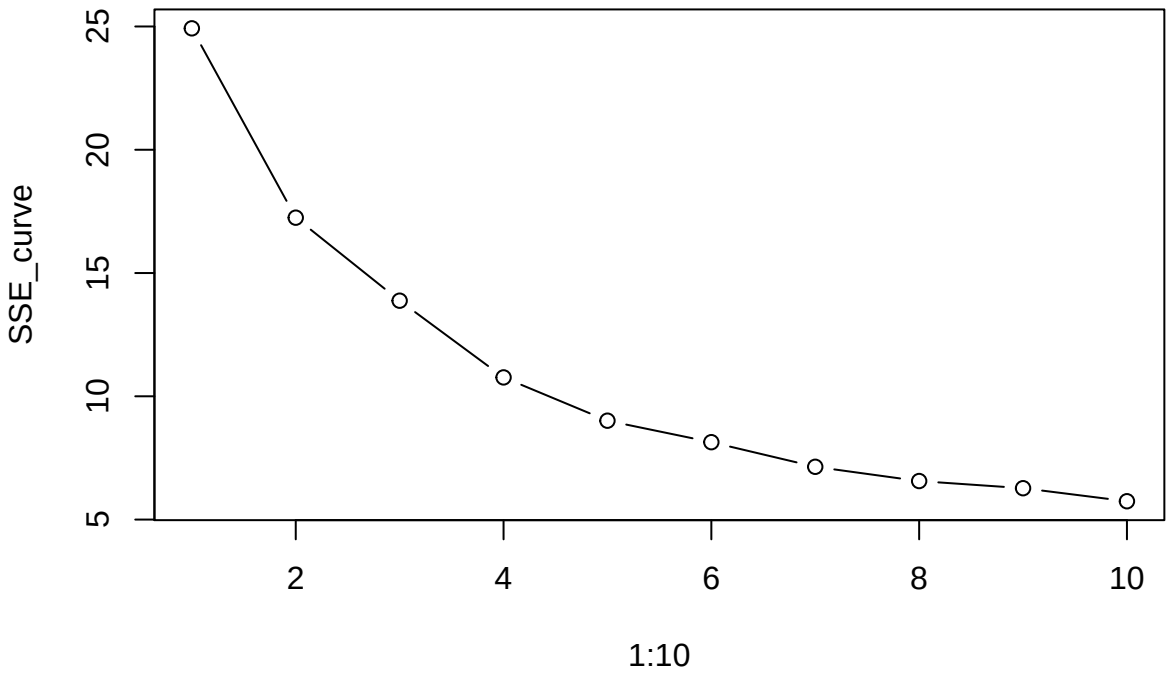
```r
# Add the cluster assignments from k-means to the wholesale data
wholesale_normalized$kmeans_cluster3 <- kcluster3$cluster

library(cluster)
sc = silhouette(wholesale_normalized$kmeans_cluster3, dist = wholesale_matrix)
summary(sc)
```

```
## Silhouette of 440 units in 3 clusters from silhouette.default(x = wholesale_normalized$kmeans_cluste
##  Cluster sizes and average silhouette widths:
##        41        61       338
## 0.1993003 0.0961455 0.5320006
## Individual silhouette widths:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.1810  0.2999  0.5179  0.4406  0.6217  0.6735
```

=> There is no negative silhouette widths, relatively well distributed cluster sizes and the mean is higher than k=4

Whole data SSE Curve Evaluation (k = 3)

```r
SSE_curve <- c()
for (n in 1:10) {
    kcluster3 = kmeans(wholesale_normalized[,3:8], n)
    sse = kcluster3$tot.withinss
    SSE_curve[n] = sse}
plot(1:10, SSE_curve, type = "b")
```



So We selected 'k=3' and using it for analysis

```
kcluster = kmeans(wholesale_normalized[,3:8], centers = 3)
kcluster$centers
```

```
##        Fresh       Milk    Grocery     Frozen Detergents_Paper Delicatessen
## 1 0.07297318 0.05468315 0.06047771 0.03993169       0.04513443   0.02316561
## 2 0.31612924 0.08086923 0.07042464 0.11874307       0.02650803   0.06815213
## 3 0.07612741 0.26781048 0.31604316 0.03156052       0.34516819   0.04830937
```

```
# Add the cluster assignments from k-means to the wholesale data
wholesale_normalized$kmeans_cluster <- kcluster$cluster

# Count how many Retail (Channel == 2) and Horeca (Channel == 1) clients are in each cluster
wholesale_normalized %>%
  group_by(kmeans_cluster, Channel) %>%
  summarise(count = n()) %>%
  arrange(kmeans_cluster)
```

```
## `summarise()` has grouped output by 'kmeans_cluster'. You can override using
## the `.groups` argument.
```

```
## # A tibble: 5 x 3
## # Groups:   kmeans_cluster [3]
##   kmeans_cluster Channel count
```

```
##            <int>   <int> <int>
## 1              1       1   245
## 2              1       2    93
## 3              2       1    53
## 4              2       2     8
## 5              3       2    41
```

```
# Count how many clients by regions are in each cluster
wholesale_normalized %>%
  group_by(kmeans_cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(kmeans_cluster)
```

```
## `summarise()` has grouped output by 'kmeans_cluster'. You can override using
## the `.groups` argument.
```

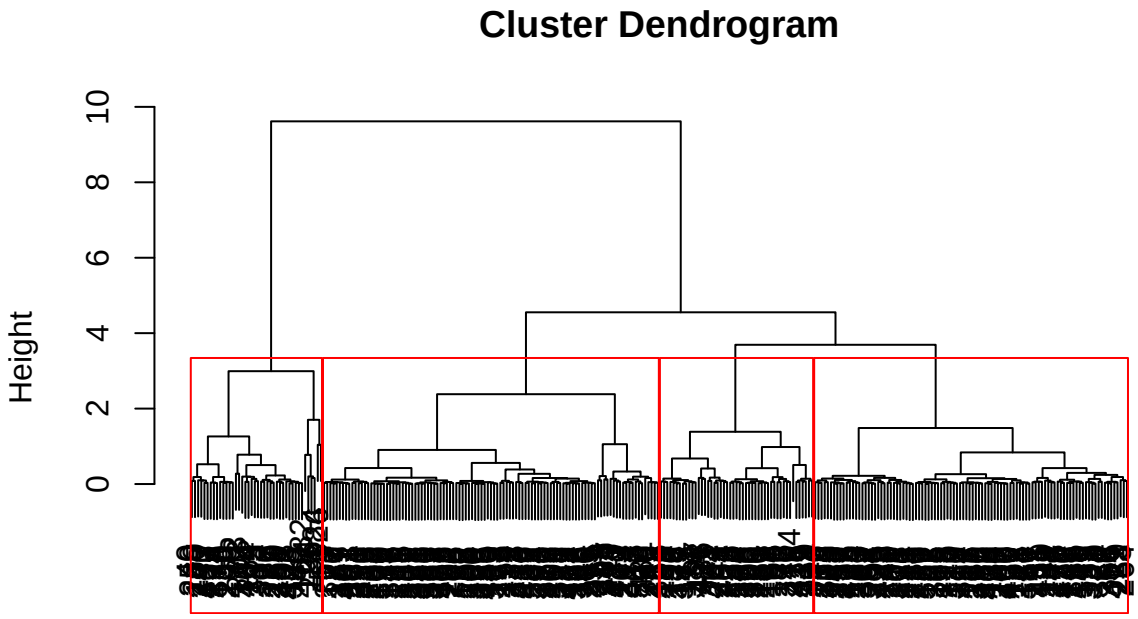
```
## # A tibble: 9 x 3
## # Groups:   kmeans_cluster [3]
##   kmeans_cluster Region count
##            <int>  <int> <int>
## 1              1      1    60
## 2              1      2    35
## 3              1      3   243
## 4              2      1    10
## 5              2      2     4
## 6              2      3    47
## 7              3      1     7
## 8              3      2     8
## 9              3      3    26
```

Divide data by two channels(horeaca, retail) Then, Normalize data and Calculate the distance

```
horeca_data <- subset(wholesale_normalized, Channel == 1)
retail_data <- subset(wholesale_normalized, Channel == 2)

library(stats)
horeca_matrix = dist(horeca_data[,3:8], method = "euclidean")
retail_matrix = dist(retail_data[,3:8], method = "euclidean")
```

Horeca Hierarchical Clustering

```
hierarchical = hclust(horeca_matrix, method = "ward.D")
plot(hierarchical, )
rect.hclust(hierarchical, k = 4)
```




```
horeca_data$cluster = cutree(hierarchical, k = 4) # let's check out cluster centroids
horeca_data %>% group_by(cluster) %>%
summarise_at(c(3:8), mean)
```

```
## # A tibble: 4 x 7
##   cluster  Fresh   Milk Grocery Frozen Detergents_Paper Delicatessen
##     <int>  <dbl>  <dbl>   <dbl>  <dbl>            <dbl>        <dbl>
## 1       1 0.120  0.0507  0.0400 0.162           0.0147       0.0290
## 2       2 0.119  0.0293  0.0353 0.0305          0.0175       0.0207
## 3       3 0.0319 0.0480  0.0412 0.0258          0.0234       0.0202
## 4       4 0.349  0.0770  0.0673 0.107           0.0186       0.0744
```

```
# Count the number of each Region clients in each cluster
horeca_data %>%
  group_by(cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(cluster)
```

```
## `summarise()` has grouped output by 'cluster'. You can override using the
## `.groups` argument.
```

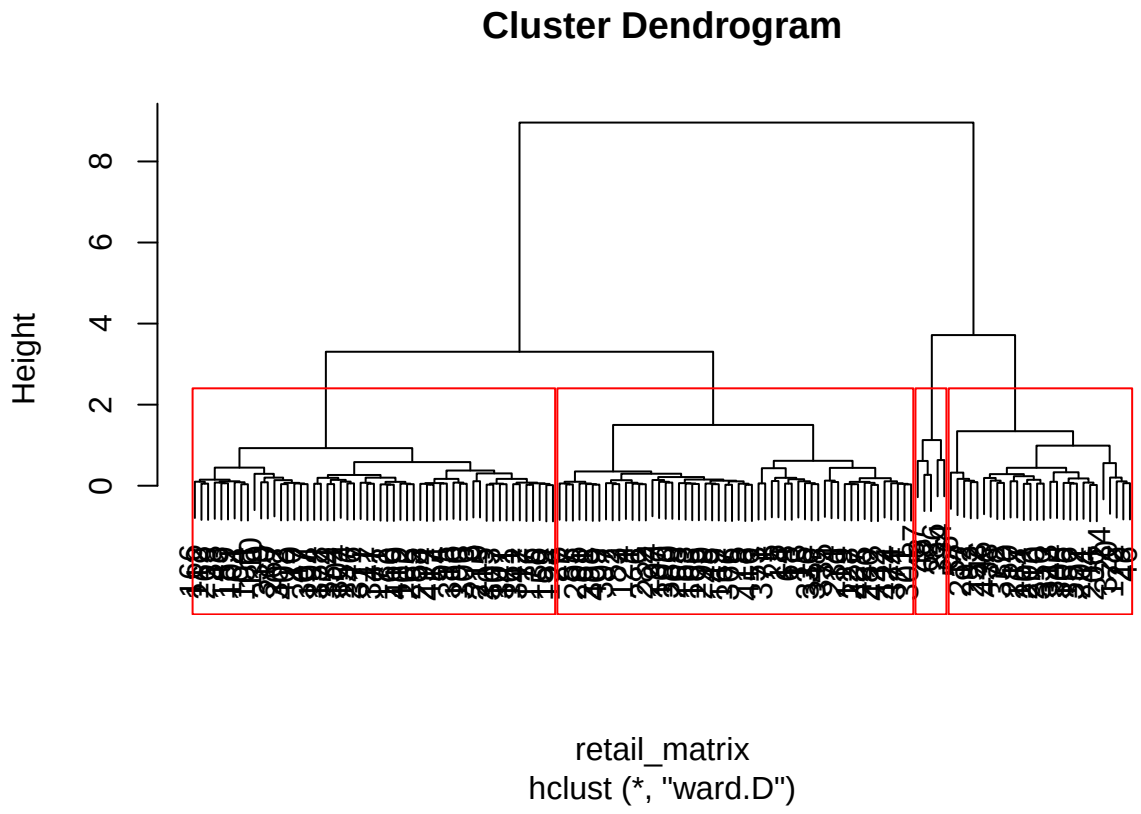
```
## # A tibble: 12 x 3
## # Groups:   cluster [4]
##    cluster Region count
```

```
##       <int>  <int> <int>
##  1        1      1     7
##  2        1      2     5
##  3        1      3    37
##  4        2      1    22
##  5        2      2    12
##  6        2      3    66
##  7        3      1    22
##  8        3      2     8
##  9        3      3    77
## 10        4      1     8
## 11        4      2     3
## 12        4      3    31
```

=> We can choose four or two clusters when we do target marketing to Horeca customers

Retail Hierarchical Clustering (k = 4)

```
hierarchical = hclust(retail_matrix, method = "ward.D")
plot(hierarchical, )
rect.hclust(hierarchical, k = 4)
```




```
retail_data$cluster = cutree(hierarchical, k = 4) # let's check out cluster centroids
retail_data %>% group_by(cluster) %>%
summarise_at(c(3:8), mean)
```

```
## # A tibble: 4 x 7
##   cluster  Fresh   Milk Grocery Frozen Detergents_Paper Delicatessen
##     <int>  <dbl>  <dbl>   <dbl>  <dbl>            <dbl>        <dbl>
## 1       1 0.123  0.0801  0.0902 0.0307           0.0766       0.0327
## 2       2 0.0344 0.124   0.160  0.0198           0.155        0.0292
## 3       3 0.0568 0.233   0.286  0.0298           0.320        0.0547
## 4       4 0.228  0.591   0.663  0.0429           0.734        0.0564
```

```
# Count the number of each Region clients in each cluster
retail_data %>%
  group_by(cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(cluster)
```

```
## `summarise()` has grouped output by 'cluster'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 11 x 3
## # Groups:   cluster [4]
##    cluster Region count
##      <int>  <int> <int>
##  1       1      1     4
##  2       1      2     6
##  3       1      3    44
##  4       2      1     7
##  5       2      2     8
##  6       2      3    40
##  7       3      1     7
##  8       3      2     4
##  9       3      3    17
## 10       4      2     1
## 11       4      3     4
```

=> We can choose four or two clusters when we do target marketing to Retail customers as well

Horeca K-Means Clustering

```
kcluster = kmeans(horeca_data[,3:8], centers = 4)
kcluster$centers
```

```
##        Fresh       Milk    Grocery     Frozen Detergents_Paper Delicatessen
## 1 0.07278819 0.03508138 0.03721027 0.03196383       0.02034084   0.01940537
## 2 0.13785812 0.07814782 0.05282421 0.17408454       0.01502259   0.04567301
## 3 0.54007799 0.40936962 0.18659438 0.62494795       0.05266510   0.43174107
## 4 0.34038949 0.04891656 0.05223674 0.06850998       0.01501890   0.03884845
```

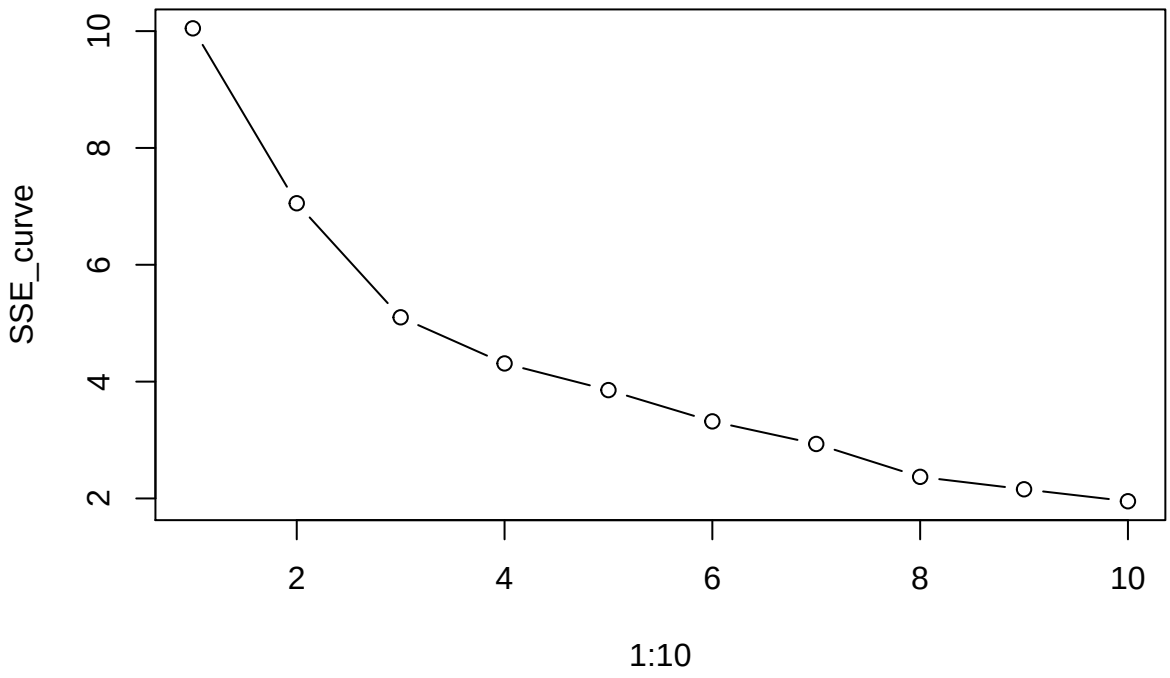
```
# Add the cluster assignments from k-means to the horeca data
horeca_data$kmeans_cluster = kcluster$cluster

# Count how many clients by regions are in each cluster
horeca_data %>%
  group_by(kmeans_cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(kmeans_cluster)
```

```
## `summarise()` has grouped output by 'kmeans_cluster'. You can override using
## the `.groups` argument.
```

```
## # A tibble: 11 x 3
## # Groups:   kmeans_cluster [4]
##    kmeans_cluster Region count
##             <int>  <int> <int>
##  1              1      1    47
##  2              1      2    20
##  3              1      3   151
##  4              2      1     5
##  5              2      2     5
##  6              2      3    29
##  7              3      2     1
##  8              3      3     2
##  9              4      1     7
## 10              4      2     2
## 11              4      3    29
```

=> When we did k=4 clustering, we also could check four types of horeca which can be coffee shops or diners, smaller food service establishments, restaurants and larger restaurants.

```
SSE_curve <- c()
for (n in 1:10) {
    kcluster = kmeans(horeca_data[,3:8], n)
    sse = kcluster$tot.withinss
    SSE_curve[n] = sse}
plot(1:10, SSE_curve, type = "b")
```



Retail K-Means Clustering

```
kcluster = kmeans(retail_data[,3:8], centers = 4)
kcluster$centers
```

```
##        Fresh       Milk   Grocery     Frozen Detergents_Paper Delicatessen
## 1 0.14226209 0.07764834 0.0896093 0.03233530       0.07074381   0.03670639
## 2 0.03427188 0.11962938 0.1527566 0.02020018       0.14899342   0.02645938
## 3 0.22826979 0.59101072 0.6625478 0.04291302       0.73415638   0.05644139
## 4 0.06052373 0.23949866 0.2880053 0.03107126       0.32728297   0.05826113
```

```
# Add the cluster assignments from k-means to the horeca data
retail_data$kmeans_cluster = kcluster$cluster

# Count how many clients by regions are in each cluster
retail_data %>%
  group_by(kmeans_cluster, Region) %>%
  summarise(count = n()) %>%
  arrange(kmeans_cluster)
```

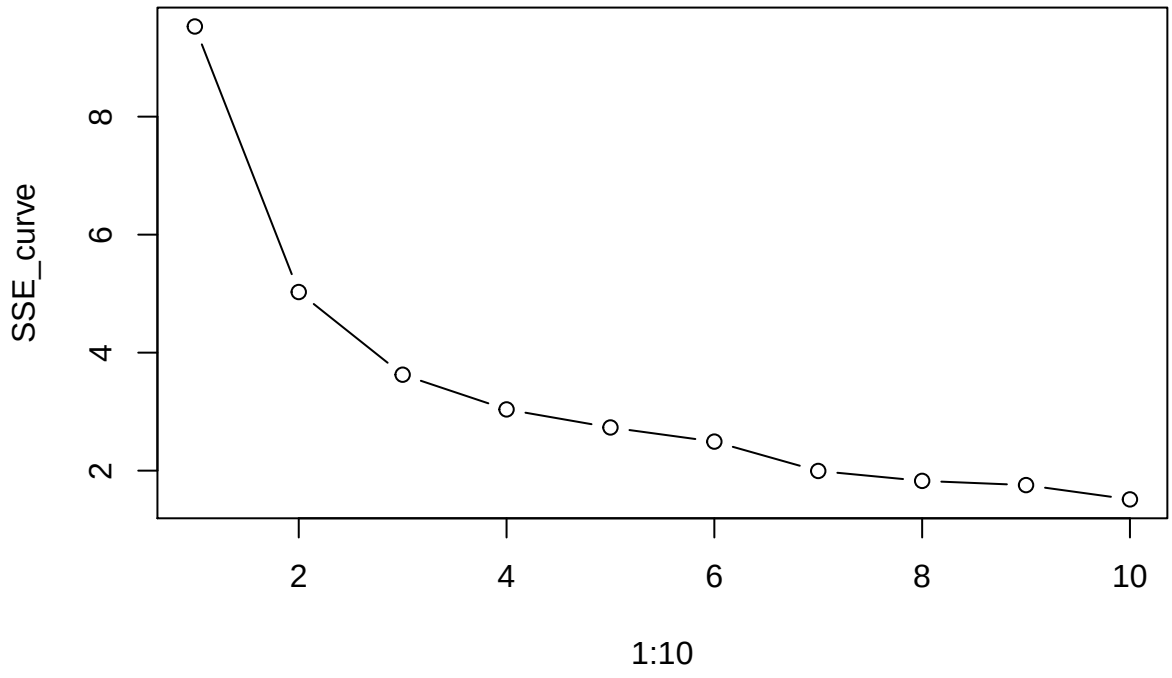
```
## `summarise()` has grouped output by 'kmeans_cluster'. You can override using
## the `.groups` argument.
```

```
## # A tibble: 11 x 3
## # Groups:   kmeans_cluster [4]
```

```
##    kmeans_cluster Region count
##             <int>  <int> <int>
##  1              1      1     3
##  2              1      2     5
##  3              1      3    36
##  4              2      1     9
##  5              2      2     9
##  6              2      3    49
##  7              3      2     1
##  8              3      3     4
##  9              4      1     6
## 10              4      2     4
## 11              4      3    16
```

=> When we did k=4 clustering, we also could check four types of retail which can be small retailers, standard grocery stores, larger grocery stores and very large supermarkets.

```
SSE_curve <- c()
for (n in 1:10) {
    kcluster = kmeans(retail_data[,3:8], n)
    sse = kcluster$tot.withinss
    SSE_curve[n] = sse}
plot(1:10, SSE_curve, type = "b")
```



Correlation matrix of spending categories

```r
spending_data <- wholesale_normalized %>% select(Fresh, Milk, Grocery, Frozen, Detergents_Paper, Delicat
cor_matrix <- cor(spending_data)
# Visualize the correlation matrix using a heatmap
library(corrplot)
```

```
## corrplot 0.94 loaded
```

```r
corrplot(cor_matrix, method = "circle")
```