Wholesale customers clustering

Raheem Bukhsh

2020-10-22

Contents

1 Introduction:

This is 2nd project of capstone HarvardX for Data Science Professional Certificate. In this project we will analyze a dataset from UCI Machine Learning Repository. This data set contains information about clients of a wholesale distributor. It includes annual spending in monetary units (m.u) for different product categories.

1.1 Objective:

The objective of this project is to classify and cluster different segments of wholesale customers through unsupervised learning methods, based on their spending prefrences.

1.2 Unsupervised Learning:

Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.
The algorithms are thus allowed to classify, label and/or group the data points contained within the data sets without having any external guidance in performing that task.
The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data. The clusters are modeled using a measure of similarity which is defined upon metrics such as Euclidean or probabilistic distance.

Common clustering algorithms include:

- Hierarchical clustering: builds a multilevel hierarchy of clusters by creating a cluster tree
- k-Means clustering: partitions data into k distinct clusters based on distance to the centroid of a cluster
- Gaussian mixture models: models clusters as a mixture of multivariate normal density components
- Self-organizing maps: uses neural networks that learn the topology and distribution of the data
- Hidden Markov models: uses observed data to recover the sequence of states

1.3 Customers Segments:

Customer segmentation is an important application of unsupervised learning.
Customer segmentation/clustering is the process of dividing a braod customer base or business markets, normall consisting of existing and potential customers into sub-groups of consumers(Segments) based on type of shared characteristics. In dividing or segmenting makets, researchers typically look for common characteristics such as shared needs,common interests, similar lifestyles, alike spending patterns or even same demographic profiles. The overall aim of segmentation is to identify the *most profitable segments* with *growth potential*. Using clustering techniques, companies can identify the several segments of customers allowing them to target the potential user base.

1.4 Wholesale Customers Data Set:

Wholesale customers is a multivariate data set with 440 observations and 8 variables which are as follows:
1. Channel: customers Channel - Horeca (Hotel/Restaurant/Cafe) or Retail channel (Nominal)
2. Region: customers Region - Lisnon, Oporto or Other (Nominal)
3. Fresh: annual spending (m.u.) on fresh products (Continuous)

4. Milk: annual spending (m.u.) on milk products (Continuous)
5. Grocery: annual spending (m.u.)on grocery products (Continuous)
6. Frozen: annual spending (m.u.)on frozen products (Continuous)
7. Detergents_Paper: annual spending (m.u.) on detergents and paper products (Continuous)
8. Delicassen: annual spending (m.u.)on and delicatessen products (Continuous)

The data has been downladed from the url "$https://archive.ics.uci.edu/ml/datasets/Wholesale+customers$"

2 Data Analysis:

First we upload the wholesale customers data set,

```
wholesale <- read.csv("C:\\Users\\Raheem\\Downloads\\Wholesale customers data.csv")
```

2.1 Data Exploration:

Upload following packages and libraries for data exploration and clustering.

```
library(corrplot)
library(dendextend)
library(stats)
library(RCurl)
library(factoextra)
library(cluster)
library(gridExtra)
library(tidyverse)
```

2.1.1 Dimensions:

```
dim(wholesale)
```

```
## [1] 440   8
```

The wholesale customers dataset has 440 observations for 8 variables.

2.1.2 Variables:

```
library(tidyverse)
glimpse(wholesale)
```

```
## Rows: 440
## Columns: 8
## $ Channel          <int> 2, 2, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2,...
## $ Region           <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,...
## $ Fresh            <int> 12669, 7057, 6353, 13265, 22615, 9413, 12126, 7579...
## $ Milk             <int> 9656, 9810, 8808, 1196, 5410, 8259, 3199, 4956, 36...
```

```
## $ Grocery          <int> 7561, 9568, 7684, 4221, 7198, 5126, 6975, 9426, 61...
## $ Frozen           <int> 214, 1762, 2405, 6404, 3915, 666, 480, 1669, 425, ...
## $ Detergents_Paper <int> 2674, 3293, 3516, 507, 1777, 1795, 3140, 3321, 171...
## $ Delicassen       <int> 1338, 1776, 7844, 1788, 5185, 1451, 545, 2566, 750...
```

It shows that 2 variables channel and region are integers but categorical and can be classified as nominal integers.

2.1.3 Summary Statistics:

Summary statistics is useful insight for the continuous integers with details about Minimum, maximum, Mean, Median and Standard Deviation.

- **Summary of Wholesale Dataset:**

```
summary(wholesale)
```

```
##     Channel          Region          Fresh             Milk
##  Min.   :1.000   Min.   :1.000   Min.   :     3   Min.   :   55
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:  3128   1st Qu.: 1533
##  Median :1.000   Median :3.000   Median :  8504   Median : 3627
##  Mean   :1.323   Mean   :2.543   Mean   : 12000   Mean   : 5796
##  3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.: 16934   3rd Qu.: 7190
##  Max.   :2.000   Max.   :3.000   Max.   :112151   Max.   :73498
##     Grocery          Frozen        Detergents_Paper    Delicassen
##  Min.   :    3   Min.   :   25.0   Min.   :    3.0   Min.   :    3.0
##  1st Qu.: 2153   1st Qu.:  742.2   1st Qu.:  256.8   1st Qu.:  408.2
##  Median : 4756   Median : 1526.0   Median :  816.5   Median :  965.5
##  Mean   : 7951   Mean   : 3071.9   Mean   : 2881.5   Mean   : 1524.9
##  3rd Qu.:10656   3rd Qu.: 3554.2   3rd Qu.: 3922.0   3rd Qu.: 1820.2
##  Max.   :92780   Max.   :60869.0   Max.   :40827.0   Max.   :47943.0
```

- **Standard deviation of variables:**

```
apply(wholesale, 2, sd)
```

```
##          Channel           Region            Fresh             Milk
##     4.680516e-01     7.742724e-01     1.264733e+04     7.380377e+03
##          Grocery           Frozen Detergents_Paper       Delicassen
##     9.503163e+03     4.854673e+03     4.767854e+03     2.820106e+03
```

Channel and Region are categorical variables thus their standard deviation can be ignored.

2.1.4 Channel and Region Insights:

- **Channel Distribution:**

```
table(wholesale$Channel)
```

```
##
##   1   2
## 298 142
```

1 is abbreviated for "HoReCa"(Hotel/Restaurant/Cafe) and 2 for "Retail" show the respective customers via each channel.

```
table(wholesale$Region)
```

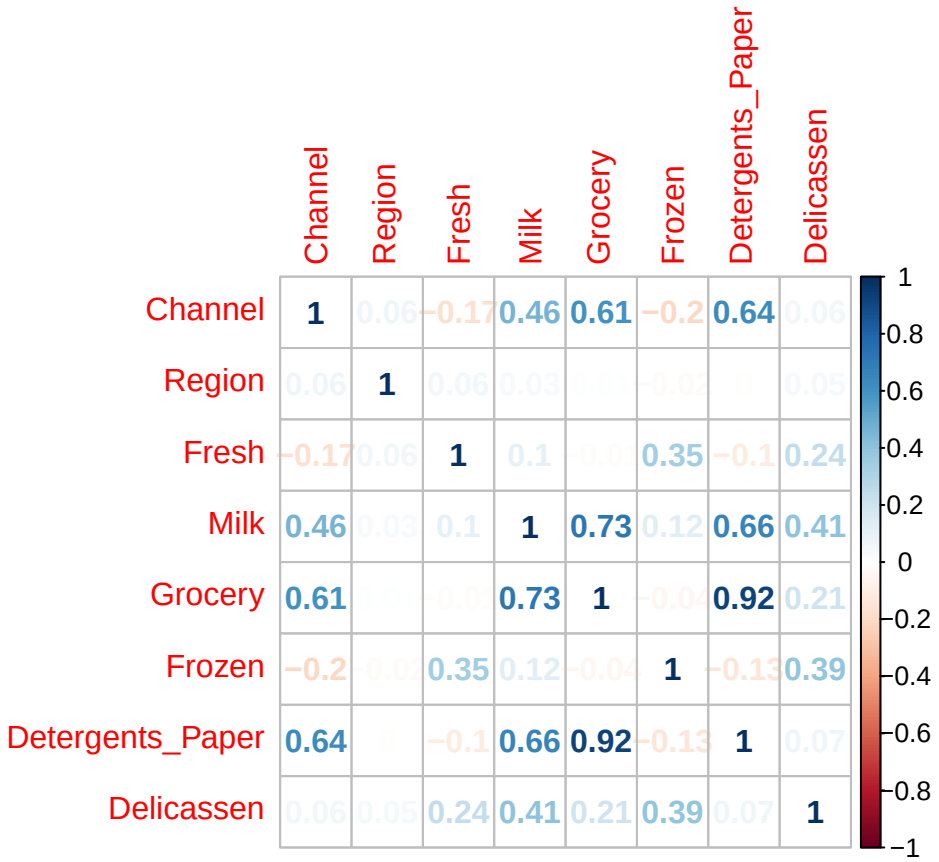
```
##
##   1   2   3
##  77  47 316
```

1 is abbreviation for "Lisbon", 2 for "Oporto" and 3 for "Other" regions.

2.1.5 Correlation between variables:

Let us further explore the dataset and check if any correlation exist between the variables.

```
library(corrplot)
ws_cor <- cor(wholesale)
corrplot(ws_cor, method = "number")
```



From the above correlation plot we can see that correlation exists between several variables but it is strongest between Detergent_Paper and Grocery.

3 Clustering Methods:

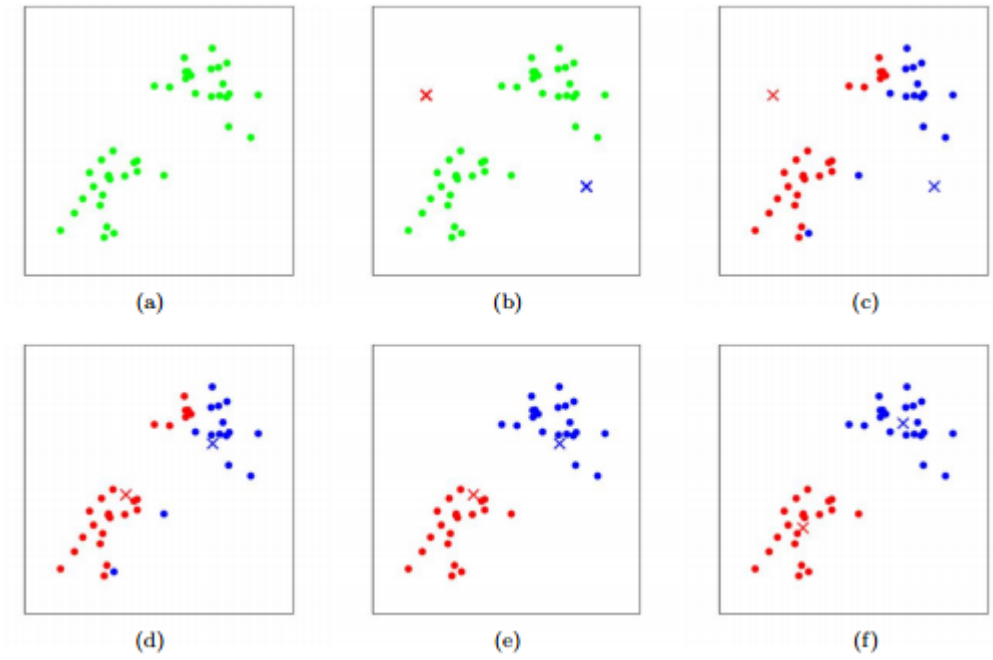
Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

There are two widely used methods used for clustering purposes.

3.1 K Means Clustering:

K-Means is one of the most popular "clustering" algorithms. K-means stores k centroids that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid.

K-Means finds the best centroids by alternating between (1) assigning data points to clusters based on the current centroids (2) chosing centroids (points which are the center of a cluster) based on the current assignment of data points to clusters.



K-means algorithm. Training examples are shown as dots, and cluster centroids are shown as crosses. (a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means. In each iteration, we assign each training example to the closest cluster centroid (shown by "painting" the training examples the same color as the cluster centroid to which is assigned); then we move each cluster centroid to the mean of the points assigned to it.

If the ith observation is in the kth cluster, then $i \in C_k$. The idea behind K-means clustering is that a *good* clustering is one for which the $within-cluster\ variation$ is as small as possible. The within-cluster variation for cluster C_k is a measure $W(Ck)$ of the amount by which the observations within a cluster differ from each other. Hence we want to solve the problem:

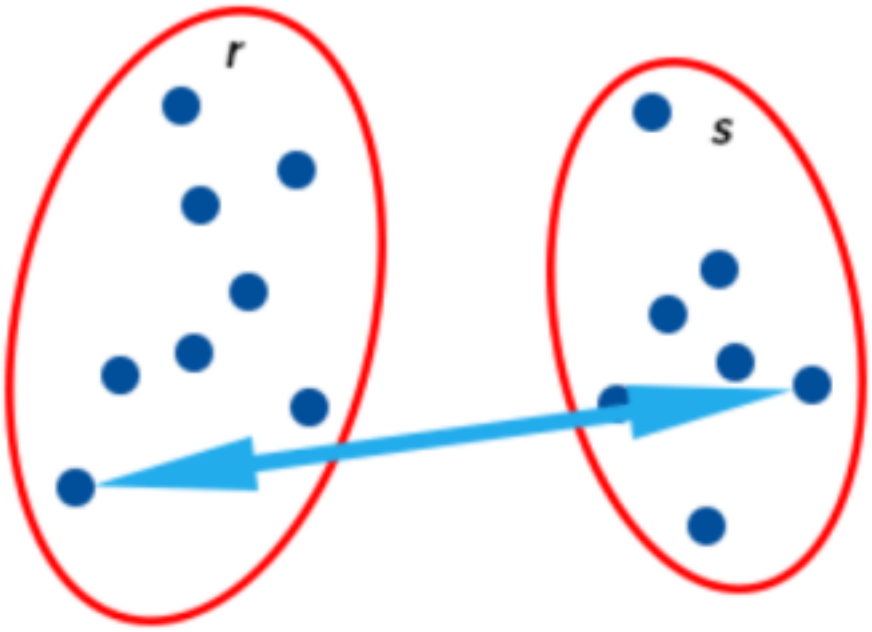
$minimize\ \{\sum_{K=1}^{K} W(C_k)\}$
$C1, ..., CK$

K Means Clustering algorithm works in five steps:
* Specify the desired number of clusters K.
* The algorithm selects k objects at random from the dataset. This object is the initial cluster or mean.

* The closest centroid obtains the assignment of a new observation. We base this assignment on the Euclidean Distance between object and the centroid.
* k clusters in the data points update the centroid through calculation of the new mean values present in all the data points of the cluster. The kth cluster's centroid has a length of p that contains means of all variables for observations in the k-th cluster. We denote the number of variables with p.
* Iterative minimization of the total within the sum of squares. Then through the iterative minimization of the total sum of the square, the assignment stop wavering when we achieve maximum iteration. The default value is 10 that the R software uses for the maximum iterations.

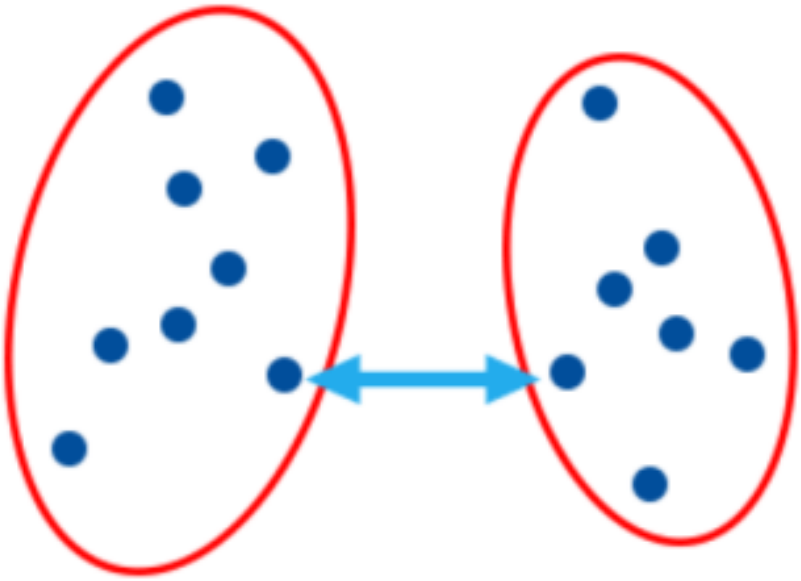
3.2 Hierarchical Clustering:

Hierarchical clustering, also known as hierarchical cluster analysis, is an algorithm that groups similar objects into groups called clusters. The endpoint is a set of clusters, where each cluster is distinct from each other cluster, and the objects within each cluster are broadly similar to each other.
Hierarchical clustering starts by treating each observation as a separate cluster. Then, it repeatedly executes the following two steps:
* Identify the two clusters that are closest together by measuring Euclidean distance. * Merge the two most similar clusters. This iterative process continues until all the clusters are merged together.
* **Computing hierarchical clustring:**
To compute the hierarchical clustering the distance matrix needs to be calculated using and put the data point to the correct cluster. There are different ways we can calculate the distance between the cluster, as given below:
1. **Complete Linkage:**
Maximum distance is calculated between clusters before merging.



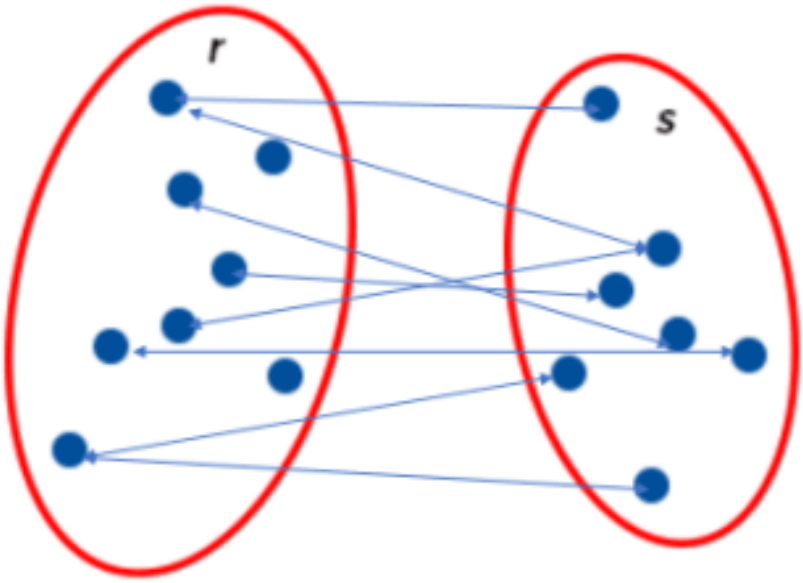
$$L(r,s) = max(D(x_{ri}, x_{sj}))$$

2. **Single Linkage:**
   Minimum distance calculates between the clusters before merging.



$$L(r,s) = min(D(x_{ri}, x_{sj}))$$

3. **Average Linkage:**
   Calculates the average distance between clusters before merging.



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

4. **Ward Linkage:**
   Instead of measuring the distance directly, it analyzes the variance of clusters. Ward's is said to be the most suitable method for quantitative variables. To implement this method, at each step find the pair of clusters that leads to minimum increase in total within-cluster variance after merging. This increase is a weighted squared distance between cluster centers.
   Ward's method says that the distance between two clusters, A and B, is how much the sum of squares will increase when we merge them:

$\Delta(A,B) = \sum_{i \in A \bigcup B} ||\overrightarrow{x_i} - \overrightarrow{m}_{A \bigcup B}||^2 - \sum_{i \in A} ||\overrightarrow{x_i} - \overrightarrow{m}_A||^2 - \sum_{i \in B} ||\overrightarrow{x_i} - \overrightarrow{m}_B||^2 = \frac{n_A n_B}{n_A + n_B} ||\overrightarrow{m}_A - \overrightarrow{m}_B||^2$
where $\overrightarrow{m}_j$ the center of cluster j, and n_j is the number of points in it. Δ is called the merging cost of combining the clusters A and B. With hierarchical clustering, the sum of squares starts out at zero (because every point is in its own cluster) and then grows as we merge clusters. Ward's method keeps this growth as small as possible.

4 Clustring Results and Analysis:

To apply the clustering model first we need to scale/standardize the data but here we have either the categorical variables channel and region in 1st and 2nd column and remaining 6 columns have the continuous

numeric data (spending in $) therefore therefore no need to scale the data but just excluding the categorical variables for clustering is enough.

```
wholesale_sc <- as.matrix(scale(wholesale[3:8]))
head(wholesale_sc)
```

```
##            Fresh        Milk     Grocery     Frozen Detergents_Paper
## [1,]  0.05287300  0.52297247 -0.04106815 -0.5886970      -0.04351919
## [2,] -0.39085706  0.54383861  0.17012470 -0.2698290       0.08630859
## [3,] -0.44652098  0.40807319 -0.02812509 -0.1373793       0.13308016
## [4,]  0.09999758 -0.62331041 -0.39253008  0.6863630      -0.49802132
## [5,]  0.83928412 -0.05233688 -0.07926595  0.1736612      -0.23165413
## [6,] -0.20457266  0.33368675 -0.29729863 -0.4955909      -0.22787885
##       Delicassen
## [1,] -0.06626363
## [2,]  0.08904969
## [3,]  2.24074190
## [4,]  0.09330484
## [5,]  1.29786952
## [6,] -0.02619421
```

4.1 K Means Clustering:

- **Selecting the number of K's:**

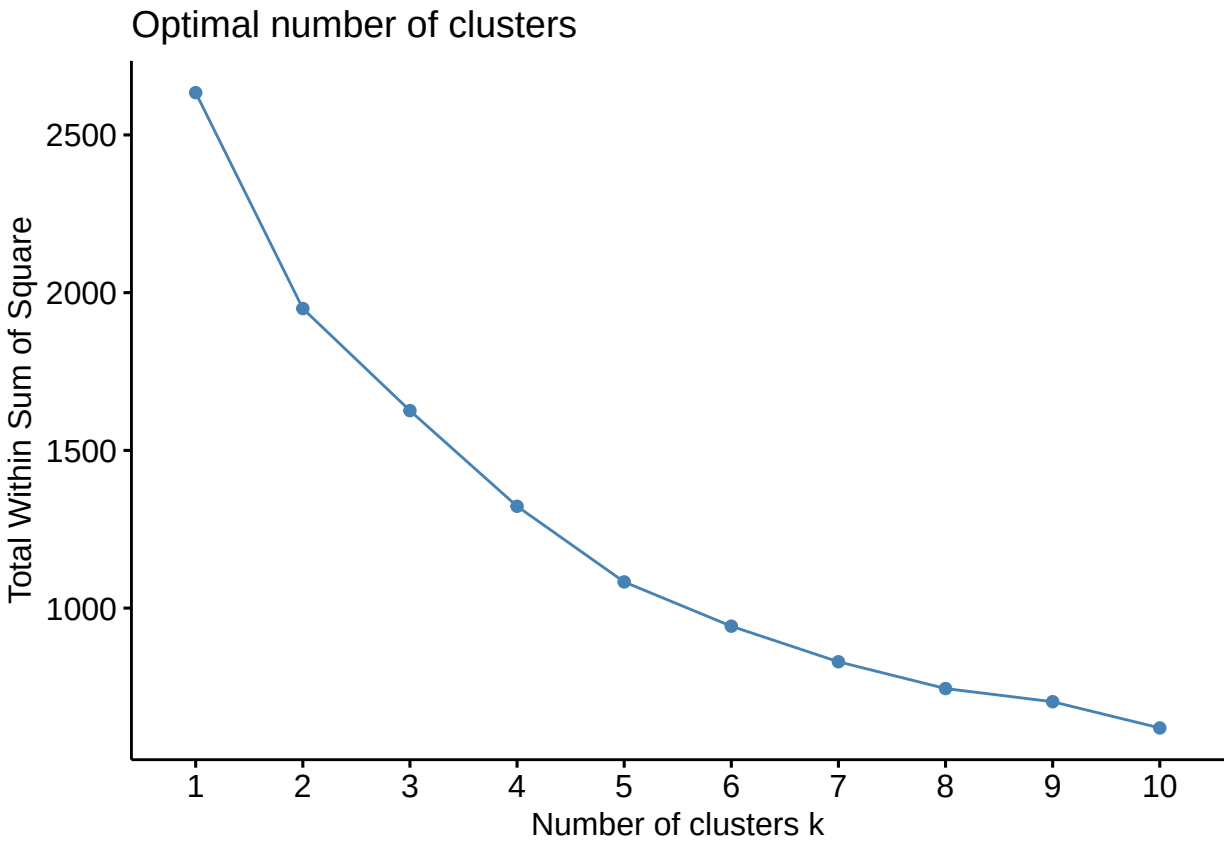
As discussed earlier first we need to find the optimal clusters,
There are three most popular methods to determine optimal clusters, which are:
* **Elbow Method**
* **Silhouette Method**
* **Gap Statistics**

Elbow Method:
We calculate the clustering algorithm for several values of k. This can be done by creating a variation within k from 1 to 10 clusters. We then calculate the total within cluster sum of square (wss). Then, we proceed to plot wss based on the number of k clusters. This plot denotes the appropriate number of clusters required in our model. In the plot, the location of a **bend** or **elbow** is the indication of the optimum number of clusters.
The process to compute the "Elbow method" has been wrapped up in a single function (fviz_nbclust):

```
library(factoextra)
library(cluster)
library(gridExtra)

set.seed(123)

fviz_nbclust(wholesale_sc , kmeans, method = "wss")
```



The elbow or bend is not sharp and any value among 2 or 5 can be selected and we select k = 5

Silhouette method:

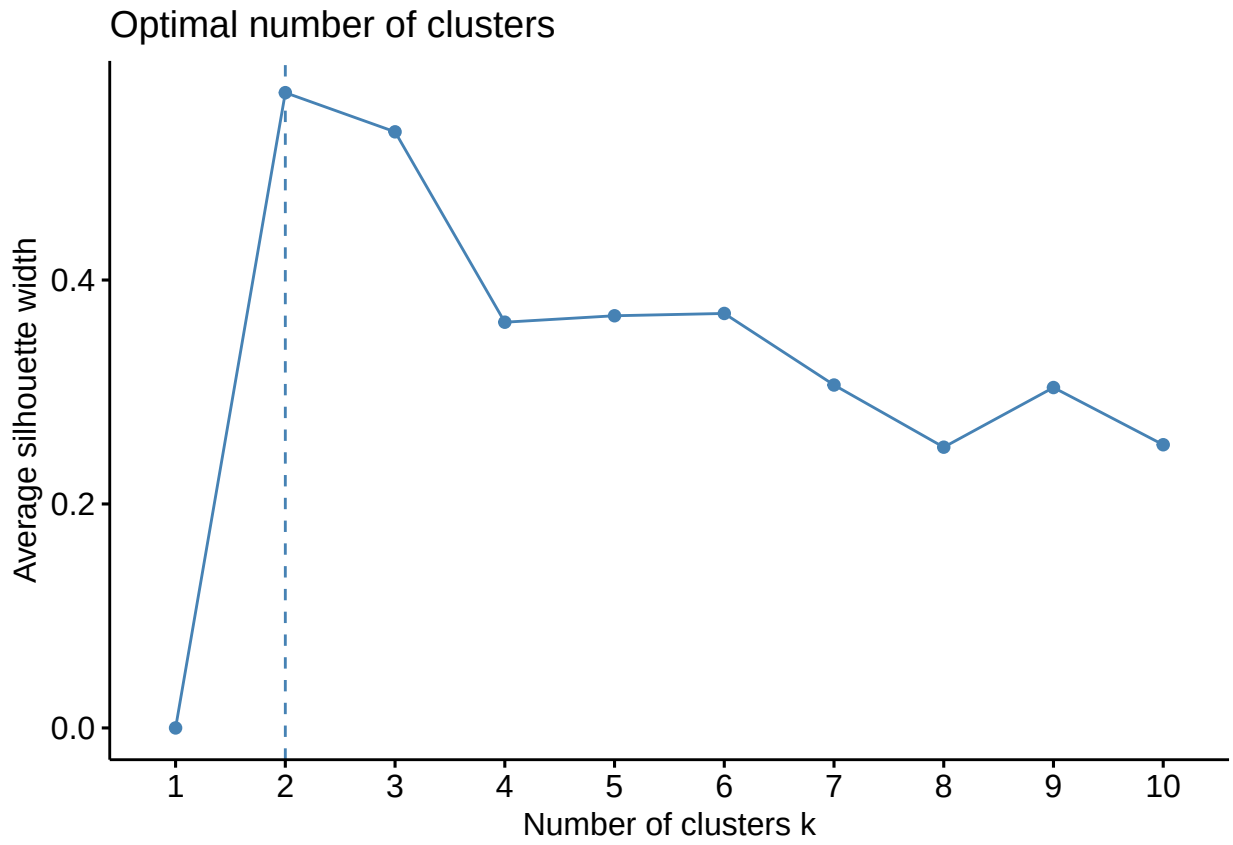
Silhouette refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object has been classified.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

The silhouette can be calculated with any distance metric, such as the Euclidean distance or the Manhattan distance. Just like elbow method, this process to compute the "average silhoutte method" has been wrapped up in a single function (fviz_nbclust):

```
set.seed(123)

fviz_nbclust(wholesale_sc, kmeans, method = "silhouette")
```



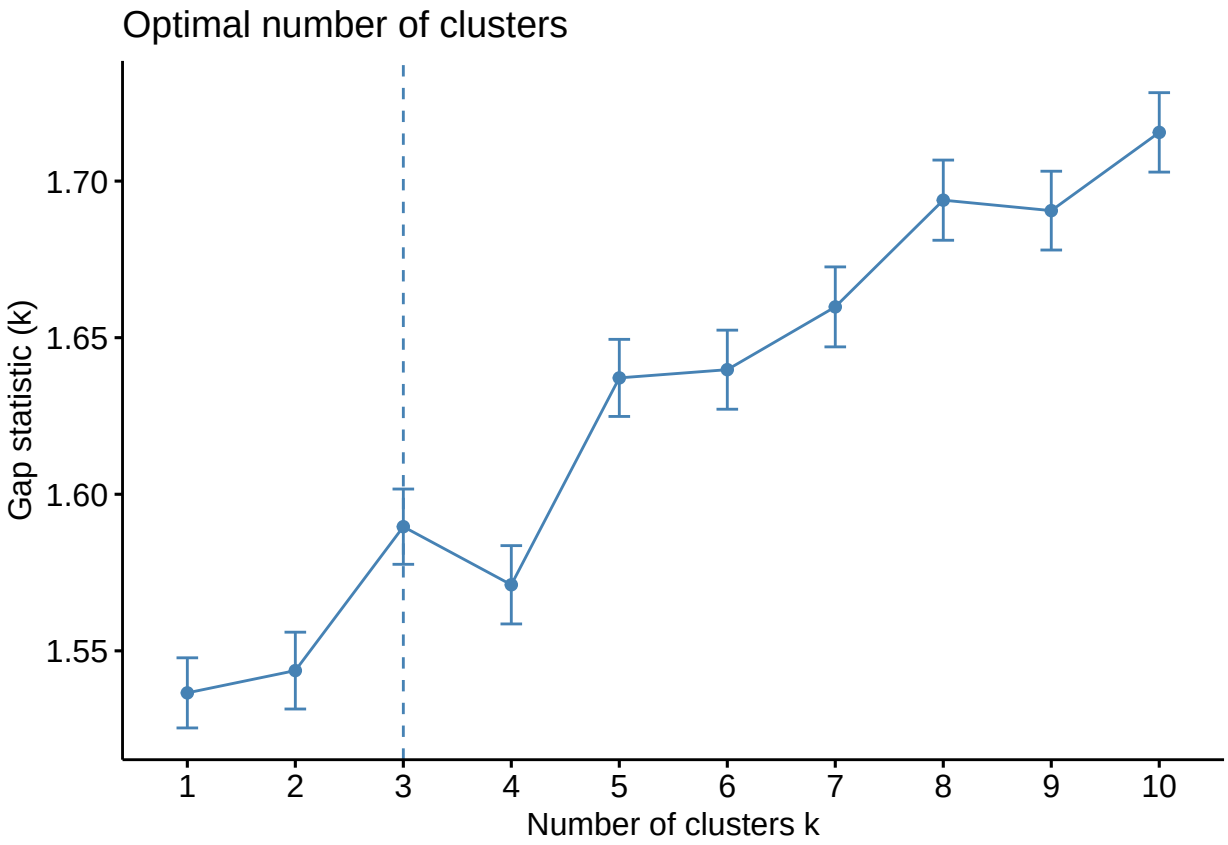
The optimal number of clusters defined by Silhouette method is 2.

Gap Statistics:

In 2001, researchers at Stanford University – R. Tibshirani, G.Walther and T. Hastie published the Gap Statistic Method. We can use this method to any of the clustering method like K-means, hierarchical clustering etc. Using the gap statistic, one can compare the total intracluster variation for different values of k along with their expected values under the null reference distribution of data. With the help of Monte Carlo simulations, one can produce the sample dataset. For each variable in the dataset, we can calculate the range between min(xi) and max (xj) through which we can produce values uniformly from interval lower bound to upper bound.
For computing the gap statistics method we can utilize the clusGap function for providing gap statistic as well as standard error for a given output.We can visualize the results with fviz_gap_stat which suggests the optimal number of clusters.

```
set.seed(123)
gap_stat_clust <- clusGap(wholesale_sc, FUN = kmeans, nstart = 25,K.max = 10, B = 50)
```

```
fviz_gap_stat(gap_stat_clust)
```

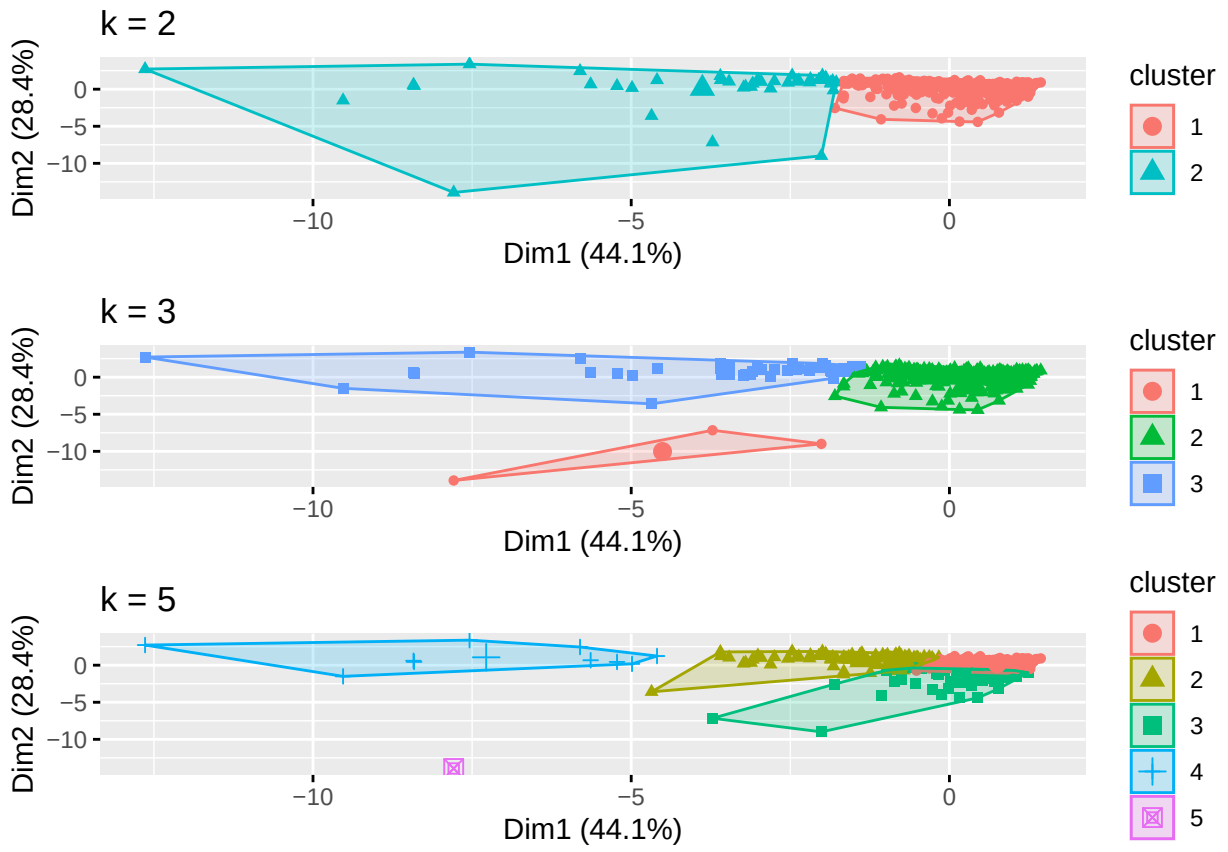


Gap statistic method shows that optimal number of clusters is 3.

- **Selecting Optimal Clusters:**
  While using the above estimates the optimal value of clusters has been 2,3 and 5. Now we have to select the most optimal number of K clusters. We can also view our results by using fviz_cluster. This provides a nice illustration of the clusters. If there are more than two dimensions (variables) fviz_cluster will perform principal component analysis (PCA) and plot the data points according to the first two principal components that explain the majority of the variance.

```
#Compute Kmeans for K=2, K=3 and K=5
k2 <- kmeans(wholesale_sc, centers = 2, nstart = 30)
k3 <- kmeans(wholesale_sc, centers = 3, nstart = 30)
k5 <- kmeans(wholesale_sc, centers = 5, nstart = 30)
#Use fviz_cluster() to compare the results
d1 <- fviz_cluster(k2, geom = "point", data = wholesale_sc) + ggtitle("k = 2")
d2 <- fviz_cluster(k3, geom = "point",  data = wholesale_sc) + ggtitle("k = 3")
d3 <- fviz_cluster(k5, geom = "point",  data = wholesale_sc) + ggtitle("k = 5")

grid.arrange(d1, d2, d3, nrow = 3)
```



From above diagram it is visible that when number of clusters is 2 and 3 it separates the data in to distinctive clusters as calculated by Silhoutte method and Gap statistic.
Now look in further detail clusters 2 and 3.

```
print(k2)
```

```
## K-means clustering with 2 clusters of sizes 399, 41
##
## Cluster means:
##         Fresh       Milk    Grocery      Frozen Detergents_Paper Delicassen
## 1 -0.00542930 -0.2122882 -0.2302493 -0.03310806       -0.2320799 -0.0826124
## 2  0.05283636  2.0659269  2.2407190  0.32219794        2.2585338  0.8039597
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 2 1 2 2 2 1 2 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1
##  [75] 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
## [149] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 2 1
## [186] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 2 1 1 1 2 1 2 1 1 1 1 2 1 1 1 1 1
## [223] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1
## [260] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [297] 1 1 1 1 1 2 1 1 2 1 2 1 1 2 1 1 2 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 2 1
## [334] 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [371] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [408] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
```

```
##
## Within cluster sum of squares by cluster:
## [1] 982.9619 966.3860
##  (between_SS / total_SS =  26.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
print(k3)
```

```
## K-means clustering with 3 clusters of sizes 3, 393, 44
##
## Cluster means:
##          Fresh       Milk    Grocery      Frozen Detergents_Paper  Delicassen
## 1  3.840444845  3.2957757  0.9852919  7.20489292       -0.1527927  6.79967230
## 2  0.004950908 -0.2277887 -0.2542638 -0.02703683       -0.2486071 -0.08005229
## 3 -0.306069120  1.8098555  2.2038591 -0.24975466        2.2309315  0.25139852
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 3 2 2 2 2 2 2 2 2
##  [38] 2 3 2 2 2 2 3 2 3 3 3 2 3 2 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2
##  [75] 2 2 2 3 2 2 2 2 2 2 2 3 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2
## [112] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2
## [149] 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 3 2 3 2 2 2 2 2 2 2 1 2 1 2
## [186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 3 2 3 2 2 2 2 3 2 2 2 2 2
## [223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2
## [260] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [297] 2 2 2 2 2 3 2 2 3 2 3 2 2 3 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 1 2 2 2 2 2 3 2
## [334] 3 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 3 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [408] 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2
##
## Within cluster sum of squares by cluster:
## [1] 214.5396 944.8291 441.0021
##  (between_SS / total_SS =  39.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

From the above details we can conclude that cluster size 3 is the most optimal number of clusters as it separates the highly variable observations in to distinctive groups. These clusters can include potentially high spending customers.

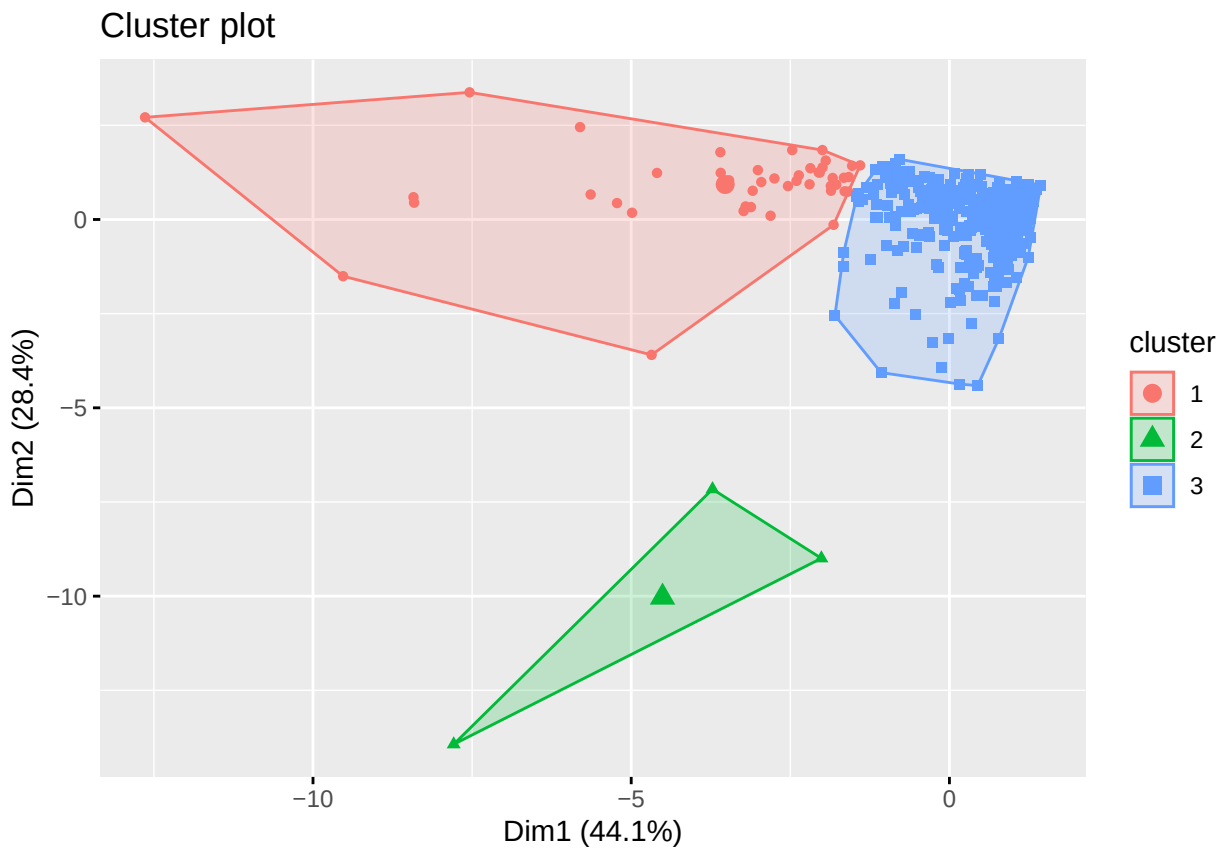
- **Clustering with K=3:**

```
# Compute K-Means Clustering with K =3
set.seed(123)
final_segments <- kmeans(wholesale_sc, centers = 3, nstart = 30)
print(final_segments)
```

```
## K-means clustering with 3 clusters of sizes 44, 3, 393
##
## Cluster means:
##          Fresh       Milk    Grocery      Frozen Detergents_Paper  Delicassen
## 1 -0.306069120  1.8098555  2.2038591 -0.24975466        2.2309315  0.25139852
## 2  3.840444845  3.2957757  0.9852919  7.20489292       -0.1527927  6.79967230
## 3  0.004950908 -0.2277887 -0.2542638 -0.02703683       -0.2486071 -0.08005229
##
## Clustering vector:
##   [1] 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 1 3 3 3 3 3 3 3 3
##  [38] 3 1 3 3 3 3 1 3 1 1 1 3 1 3 3 3 3 3 3 1 3 3 3 3 1 3 3 3 1 3 3 3 3 3 3 3 3
##  [75] 3 3 3 1 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3
## [112] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3
## [149] 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 1 3 1 3 3 3 3 3 1 3 1 3 3 3 3 3 3 3 2 3 2 3
## [186] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 3 3 3 1 3 3 3 1 3 1 3 3 3 3 1 3 3 3 3 3
## [223] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3
## [260] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [297] 3 3 3 3 3 1 3 3 1 3 1 3 3 1 3 3 1 3 3 3 3 3 3 1 3 3 3 3 3 2 3 3 3 3 3 1 3
## [334] 1 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3 1 3 1 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [371] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [408] 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3
##
## Within cluster sum of squares by cluster:
## [1] 441.0021 214.5396 944.8291
##  (between_SS / total_SS =  39.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Now visualize the final clustering when K=3

```
#Visualize the final clusters
fviz_cluster(final_segments, geom = "point",  data = wholesale_sc)
```



Now we can extract the clusters and do some descriptive statistics at cluster level.

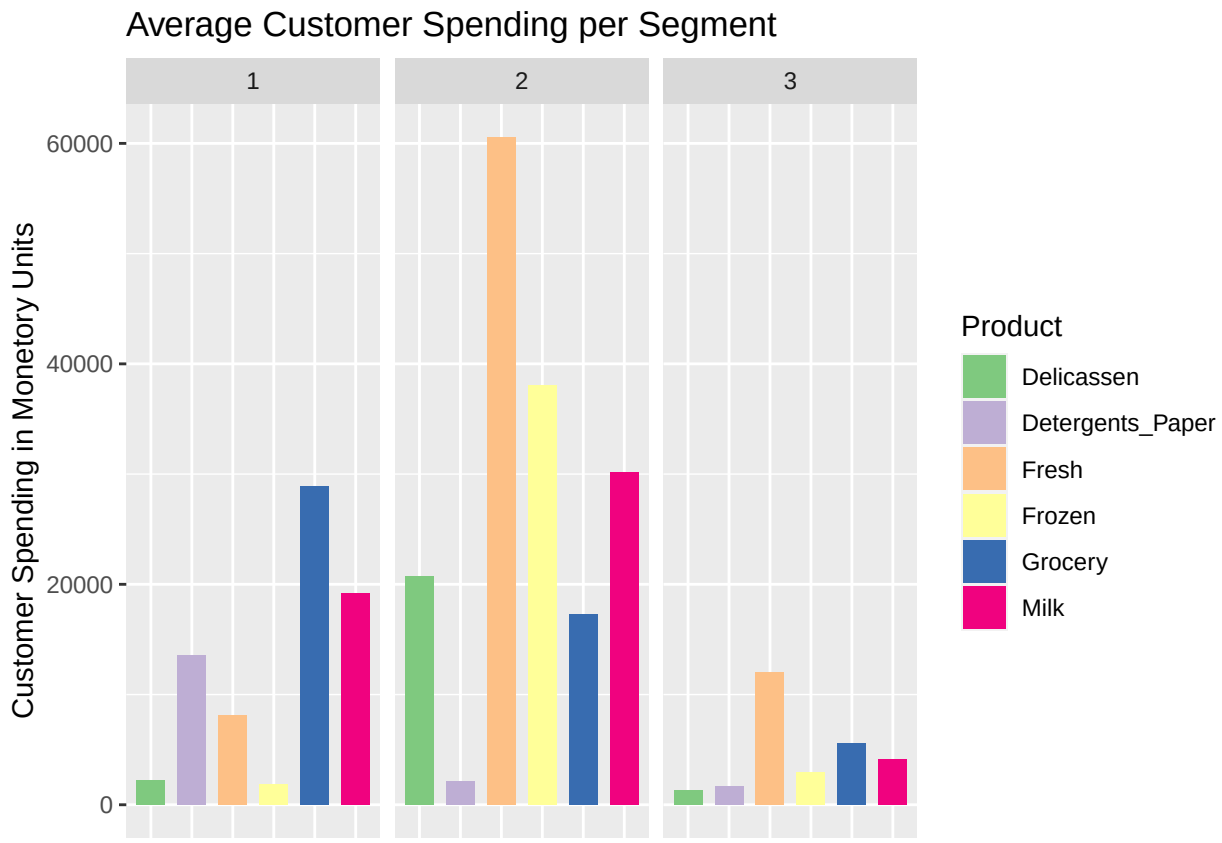
```
cluster_means <- wholesale[3:8] %>%
  mutate(Cluster = final_segments$cluster) %>%
  group_by(Cluster) %>%
  summarise_all(list(mean))
cluster_means
```

Cluster	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
1	8129.341	19153.682	28894.909	1859.455	13518.25	2233.841
2	60571.667	30120.333	17314.667	38049.333	2153.00	20700.667
3	12062.913	4115.099	5534.967	2940.677	1696.17	1299.115

- **Average customer spending per segment:**

```
cluster_means %>%
  gather(Product, MU, Fresh:Delicassen)%>%
    ggplot(aes(x=Product , y = MU, fill = Product)) + geom_col(width = 0.7) +
              facet_grid(.~ Cluster)+
                      scale_fill_brewer(palette = "Accent")+
                          ylab("Customer Spending in Monetory Units") +
                    ggtitle("Average Customer Spending per Segment")+
                                  theme(axis.text.x = element_blank(),
                                        axis.ticks.x = element_blank(),
```

```
                                                  axis.title.x = element_blank())
```



The above graph shows that customer segments vary a lot in spending patterns and prefrences.

Now lets analyze the Channel and Region distribution among the customer segments.

```
# Channel and Region in Cluster 1
wholesale[1:2] %>%
  mutate(Cluster = final_segments$cluster) %>%
  filter(Cluster == 1) %>% ungroup() %>% count(Channel, Region)
```

Channel	Region	n
2	1	7
2	2	8
2	3	29

```
# Channel and Region in Cluster 2
wholesale[1:2] %>%
  mutate(Cluster = final_segments$cluster) %>%
  filter(Cluster == 2) %>% ungroup() %>% count(Channel, Region)
```

Channel	Region	n
1	2	1
1	3	2

```
# Channel and Region in Cluster 3
wholesale[1:2] %>%
  mutate(Cluster = final_segments$cluster) %>%
  filter(Cluster == 3) %>% ungroup() %>% count(Channel, Region)
```

Channel	Region	n
1	1	59
1	2	27
1	3	209
2	1	11
2	2	11
2	3	76

The region 3 contributes the most of customers for the wholesale business.

4.1.1 K Means Clustering analysis:

From above analysis we can conclude below observations or customer spending habits of each segment identified in clustered data:

Segment 1: This segment has only Retail customers who spend more on Groceries and Milk followed by Detergents and papers and then on Fresh products.

Segment 2: This segment contains only Hotel/Restaurant/Cafe customers who spend heavily on Fresh , Frozen followed by Milk and Groceries but not on other items. Also have highest median spending on Delicassen.These customers form a well seperated group with these spending habits.

Segment 3: This segment consist of majority Hotel/Restaurant/Cafe customers along with Retail Customers who spend decently on Fresh followed by Groceries and Milk,but spend least on Detergents_Paper and Delicassen in all groups.

4.2 Hierarchical Clustering:

Hierarchical clustering builds clusters within clusters, and does not require a pre-specified number of clusters like K-means do. A hierarchical clustering can be thought of as a tree and displayed as a dendrogram; at the top there is just one cluster consisting of all the observations, and at the bottom each observation is an entire cluster. In between are varying levels of clustering.we can build the clustering with hclust. To implement hierarchical clustering first calculate the euclidean distance for wholesale data set. The 1 and 2 columns represent the categorical variables as stated earlier too therefore we can ignore these two columns for hierarchical clustering purposes.

```
wholesale_md <- wholesale[3:8]
head(wholesale_md)
```

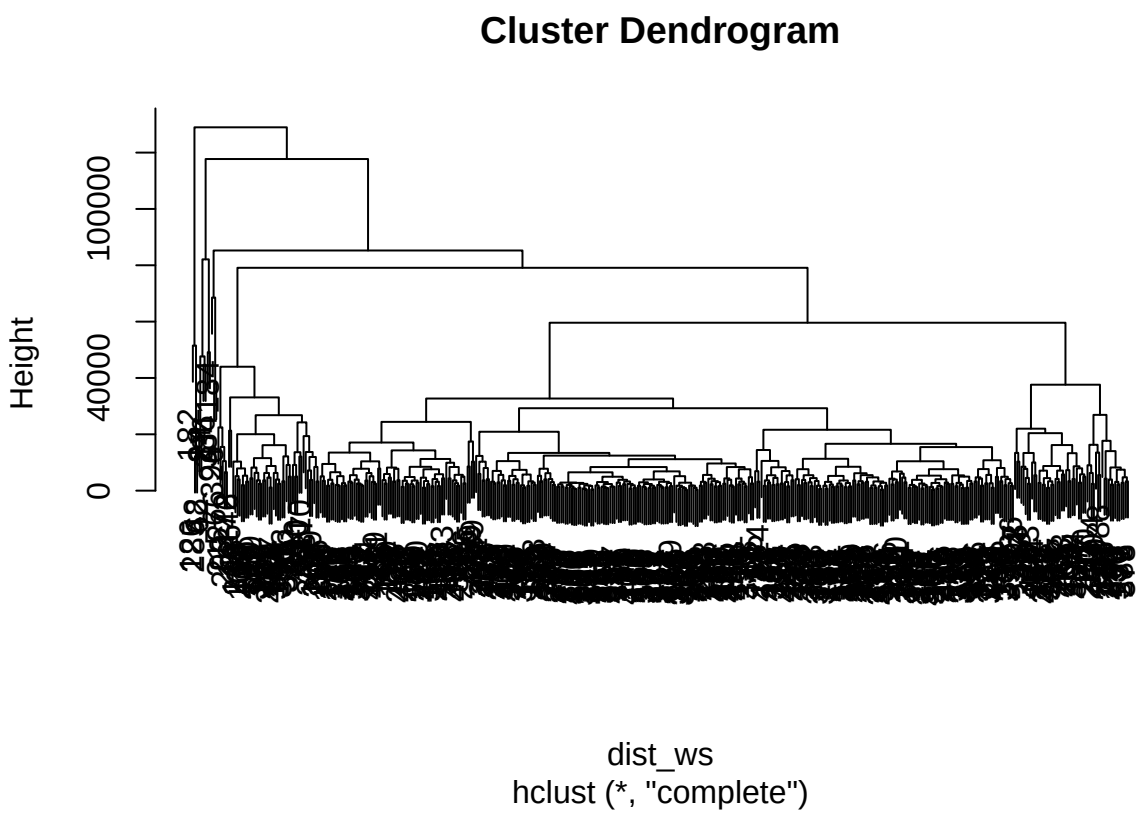
Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
12669	9656	7561	214	2674	1338
7057	9810	9568	1762	3293	1776
6353	8808	7684	2405	3516	7844
13265	1196	4221	6404	507	1788
22615	5410	7198	3915	1777	5185
9413	8259	5126	666	1795	1451

```
# Calculate Euclidean distance for scaled data wholesale_sc
dist_ws <- dist(wholesale_md, method = "euclidean")
```

- **Selecting Linkage Method:**
  First we shall have a look at dendoograms with different linkage methods to choose the most suitable linkage method.

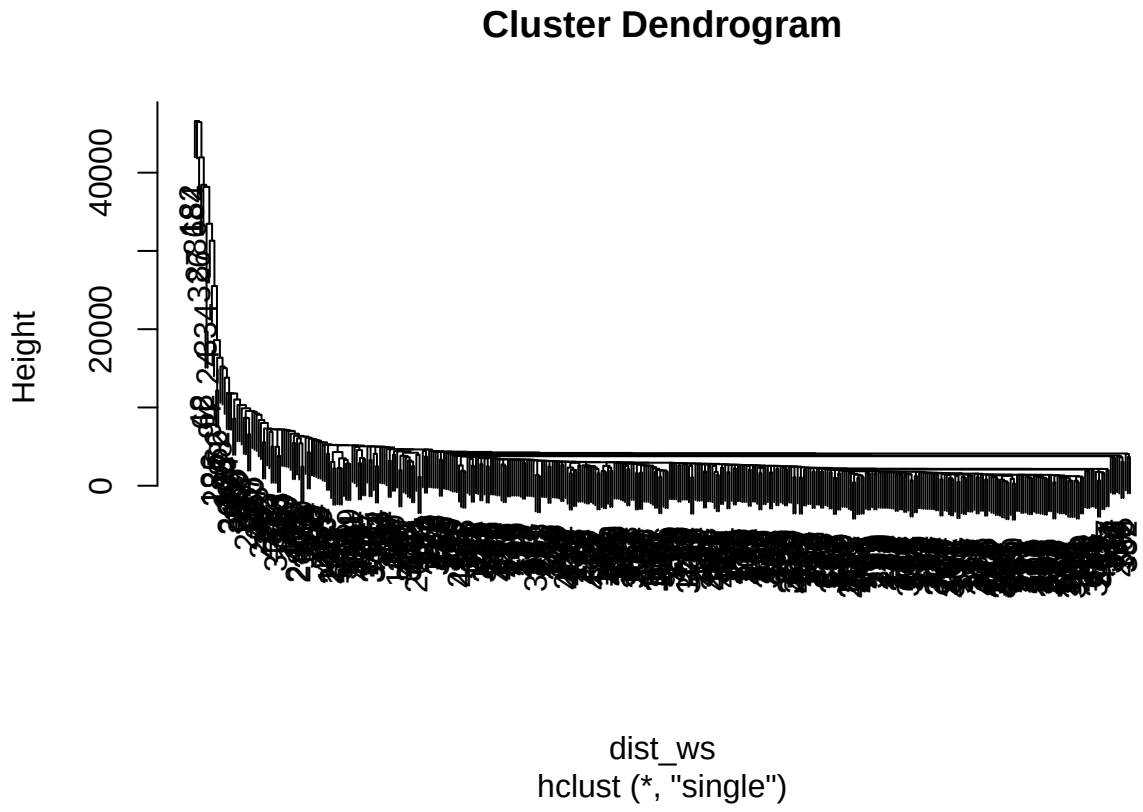
Complete Linkage:

```
# Generate a complete linkage analysis
hc_ws_complete <- hclust(dist_ws, method = "complete")
# Plot
plot(hc_ws_complete)
```



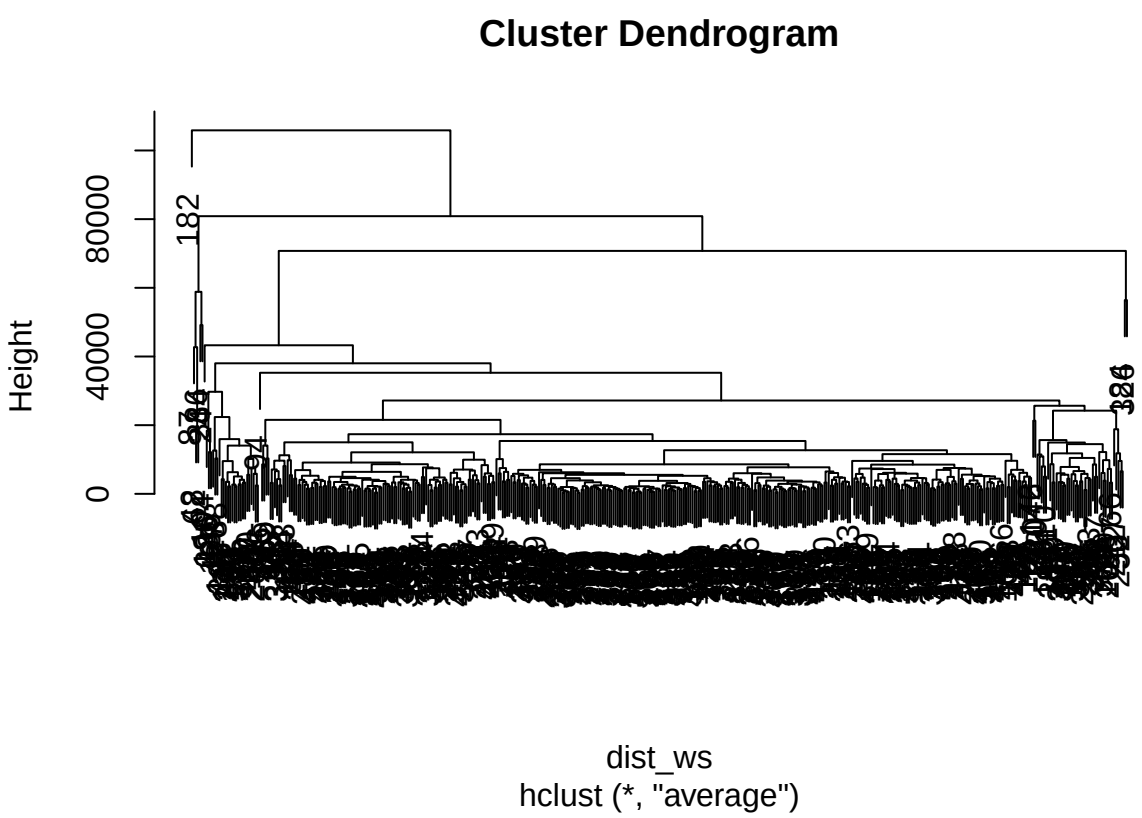
Single Linkage:

```r
# Generate a single linkage analysis
hc_ws_single <- hclust(dist_ws, method = "single")
# Plot
plot(hc_ws_single)
```



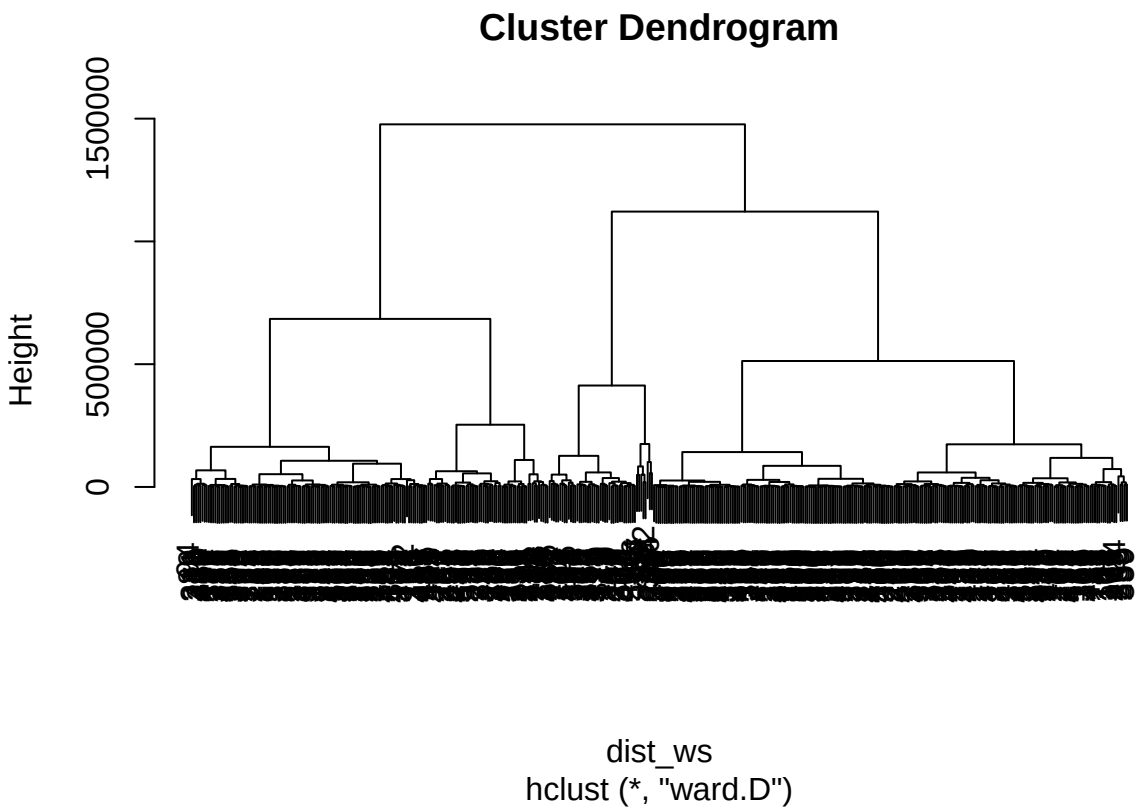
Average Linkage:

```r
# Generate a average linkage analysis
hc_ws_average <- hclust(dist_ws, method = "average")
# Plot
plot(hc_ws_average)
```



Ward Linkage:

```
# Generate a ward linkage analysis
hc_ws_ward <- hclust(dist_ws, method = "ward.D")
# Plot
plot(hc_ws_ward)
```



With a view from naked eye it is evident that **Ward Linkage** method is the most suitable method for this wholesale dataset because all other linkage methods are forming clusters even for individual observations. Also the clustering patterns formed by all other linkage methods do not make any sense. It is an integral part of the hierarchical clustering method that the best linkage method is choosen based upon the observation of dendrogram with naked eye.
Now for all the hierarchical clustering purpose we shall be using the **Ward Linkage Method**.

- **Hierarchical clustering with ward linkage method:**

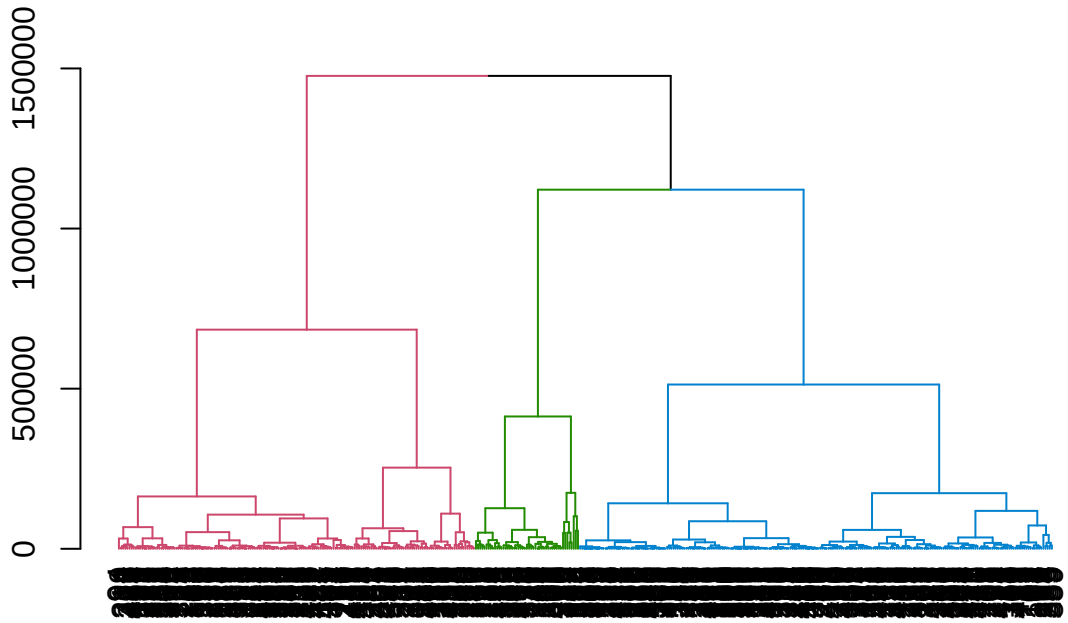
```
library(tidyverse)
hc_ws_ward <- hclust(dist_ws, method = "ward.D")
#Create a cluster assignment vector at h = 750000
cl_ws_ward <- cutree(hc_ws_ward, h = 750000)
#Generate the segmented customers data frame
Segment_ws_md <- mutate(wholesale_md, cluster = cl_ws_ward)
```

Cutting the dendogram at height of "750000" is based upon the visual observation to create k =3. Inspite of 750000 we could have cut it at 500000 and the resultant k's will be 5 but from k means clustering we know that the optimal number of clusters is 3 therefore it has been cut intentionally at a height where it creates 3 clusters.

```
# Count the number of customers that fall in to each cluster
count(Segment_ws_md, cluster)
```

cluster	n
1	168
2	223
3	49

```
# Create and Color the dendrogram based on the height of cutoff
library(dendextend)
dend_ws <- as.dendrogram(hc_ws_ward)
dend_ws_color <- color_branches(dend_ws, h = 750000)
plot(dend_ws_color)
```

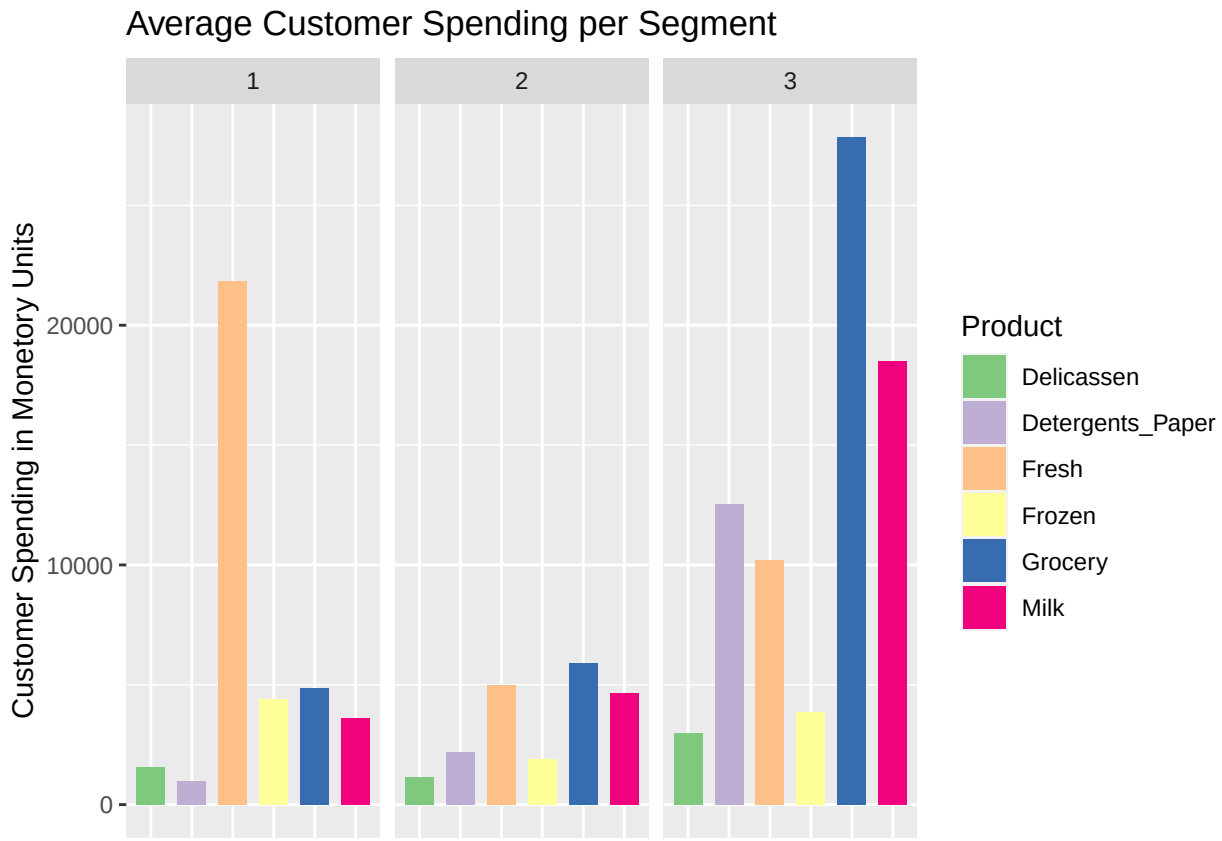



```
segmented_clusters <- Segment_ws_md %>%
  group_by(cluster) %>%
  summarise_all(list(mean))
segmented_clusters
```

cluster	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
1	21849.982	3605.851	4846.548	4392.589	991.2202	1581.512
2	4979.673	4652.511	5917.179	1900.717	2186.2242	1164.327
3	10180.959	18511.510	27853.286	3874.184	12526.6122	2971.510

- **Customer spending per cluster:**

```
segmented_clusters %>%
  gather(Product, MU, Fresh:Delicassen)%>%
    ggplot(aes(x=Product , y = MU, fill = Product)) + geom_col(width = 0.7) +
              facet_grid(.~ cluster)+
                      scale_fill_brewer(palette = "Accent")+
                          ylab("Customer Spending in Monetory Units") +
                    ggtitle("Average Customer Spending per Segment")+
                                    theme(axis.text.x = element_blank(),
                                          axis.ticks.x = element_blank(),
                                          axis.title.x = element_blank())
```



4.2.1 Hierarchical clustering analysis:

From the above analysis we can analyze the spending pattern of the wholesale customers. The three defined customer segments can be categorized as follows:

Segment 1: This segment has most of Hotel/Restaurants/Cafe customers who spend the most on Fresh items followed by spending on Grocery, Frozen and Milk whereas the spending on detergent paper and delicassan is minimum.

Segment 2: This segment does not spend much on any of items but at the same time they are the largest portion among customers. So they seem to be the small/medium sized whole customers from all business sectors.

Segment 3: This segment has most of retail customers because their are spending the most on grocery and milk followed by fresh and detergent paper while frozen and delicassan are least popular among this segment.

5 Conclusion:

Any method between K means and Hierarchical clustering can be used for segmentation purpose in unsupervised learning. However this decision is relative to the data being analysed. For this specific wholesale data set both methods have been used and number of clusters have been kept to 3 in both for comparison purposes. Results generated by both methods are different from each other however for this specific data set K means clustering method seems to be more relevant.