# ML 1000 Assignment 2

by Anupama r.k, Queenie Tsang, Crystal (Yunan) Zhu

03/03/2021

## Abstract

Anomaly detection or Outlier detection identifies data points, events or observations that deviate from dataset's normal behavior. Anomalous data indicate critical incidents or potential opportunities. In order to take advantage of opportunities or fix costly problems anomaly detection has to be done in real time. Unsupervised machine learning models can be used to automate anomaly detection. Unsupervised anomaly detection algorithms scores data based on intrinsic properties of the dataset. Distances and densities are used to give an estimation what is normal and what is an outlier. Anomaly detection monitor is a tool developed for an online retailer to check product quality issues like profit opportunities and sales glitches. The application is built using R and Shinyapp following CRISP-DM framework.

## Business Case

The Superstore is a chain of stores all over the United States with an online website where customers can view and order products. The Superstore company would like to understand more about customer preferences such as the types of products ordered, category, shipping service, quantity of items bought and whether customers use discounts. With a greater understanding of customer needs by location (city), the Superstore can then use this information to update their website. In particular, when customers select their city on the Superstore website, the site may show more products of a certain category which is commonly ordered by customers located in that area. By conducting clustering on transaction data which spans over 3 years, the Superstore company find out groupings of transactions which may share similarities. For example, if a group of transactions indicate customers from a certain location are similar in typically using discounts, the Superstore may offer more promotional offers on their site for customers at that location. Another example would be if a cluster of transactions indicate typical sales are higher than other groupings, then the Superstore may display more expensive items on their site for customers in that location.

## Objective

The objective is to detect point anomalies from superstore dataset using K-NN and clustering methods. Another objective is to use clustering methods to find commonalities in the transactions in order to offer more relevant products to customers based on their location and typical sales transactions.The clustering methods used are K-means, K-medoids and the principal components method using Factor Analysis of Mixed Data (FAMD).

## Data Understanding

US Superstore dataset is sourced from US uperstore dataset . The dataset has online orders for Superstores in U.S. from 2014-2018. Tableau community is the owner of the dataset. The dataset has 9994 records and 21 attributes.

**Import data**

```
data_superstore
```

Table 1: Dataset description

| Attribute | Data Type | Description |
|---|---|---|
| Row ID | numeric | row number |
| Order ID | character | unique order number |
| Order Date | numeric | order placed date |
| Ship Date | numeric | order shipping date |
| Ship Mode | character | shipping mode of order |
| Customer ID | character | unique customer id for order |
| Customer Name | character | name of customer |
| Segment | character | section of product |
| Country | character | country based on order |
| City | character | city based on order |
| State | character | state based on order |
| Postal Code | numeric | pin code |
| Region | character | region based on order |
| Product ID | character | product id of product |
| Category | character | category of product |
| Sub-Category | character | sub-category of product |
| Product Name | character | name of product |
| Sales | numeric | selling price of product |
| Quantity | numeric | order quantity |
| Discount | numeric | discount on product |
| Profit | numeric | profit from product |

# Exploratory Data Analysis

```
#check duplicates
#data %>% distinct()
data_nodup=distinct(data,data[,1:21], keep_all=TRUE)[,-22]
#dim did not change - no dup

#check missing values
n=c()
for (i in 1:ncol(data)) {
  n[i]=sum(is.na(data[,i]))
}
missing_values=paste0(colnames(data),rep("-",ncol(data)),n,rep(" missing values",))
#cat("The number of missing values for each variable are:")
missing_values
```

```
##  [1] "ï..Row.ID-0 missing values"     "Order.ID-0 missing values"
##  [3] "Order.Date-0 missing values"    "Ship.Date-0 missing values"
##  [5] "Ship.Mode-0 missing values"     "Customer.ID-0 missing values"
```

```
##  [7] "Customer.Name-0 missing values" "Segment-0 missing values"
##  [9] "Country-0 missing values"       "City-0 missing values"
## [11] "State-0 missing values"         "Postal.Code-0 missing values"
## [13] "Region-0 missing values"        "Product.ID-0 missing values"
## [15] "Category-0 missing values"      "Sub.Category-0 missing values"
## [17] "Product.Name-0 missing values"  "Sales-0 missing values"
## [19] "Quantity-0 missing values"      "Discount-0 missing values"
## [21] "Profit-0 missing values"        "diff_in_days-0 missing values"
```

```
#no missing values
data_miss=data[!complete.cases(data),]
```

There are no missing values in the dataset.

Get a general idea of the data set.

```
length(unique(data$Customer.ID))
```

```
## [1] 793
```

```
#793 unique customer IDs
length(unique(data$Customer.Name))
```

```
## [1] 793
```

```
#793 unique customer names - We can drop one of these two variables since they are redundant.

length(unique(data$Segment))
```

```
## [1] 3
```

```
unique(data$Segment)
```

```
## [1] "Consumer"    "Corporate"   "Home Office"
```

```
#"Consumer"    "Corporate"   "Home Office"

unique(data$Country)
```

```
## [1] "United States"
```

```
#all are from US - could drop this variable due to no-variation introduced by it

length(unique(data$City))
```

```
## [1] 531
```

```r
#531 different cities

length(unique(data$State))
```

```
## [1] 49
```

```r
#49 states

length(unique(data$Postal.Code))
```

```
## [1] 631
```

```r
#631 postal code - 793 unique customer IDs - some customers live very close!

unique(data$Region)
```

```
## [1] "South"   "West"    "Central" "East"
```

```r
#only 4 regions

unique(data$Sub.Category)
```

```
##  [1] "Bookcases"   "Chairs"      "Labels"      "Tables"      "Storage"
##  [6] "Furnishings" "Art"         "Phones"      "Binders"     "Appliances"
## [11] "Paper"       "Accessories" "Envelopes"   "Fasteners"   "Supplies"
## [16] "Machines"    "Copiers"
```

```r
#17 sub-categories

length(unique(data$Product.Name))
```

```
## [1] 1850
```

```r
#1850 product names
length(unique(data$Product.ID))
```

```
## [1] 1862
```

```r
#1862 product IDs - potential redundant variables!

par(mfrow=c(2,2))
hist(data$diff_in_days)
boxplot(data$Sales)
hist(data$Sales)
boxplot(data$Quantity)
```

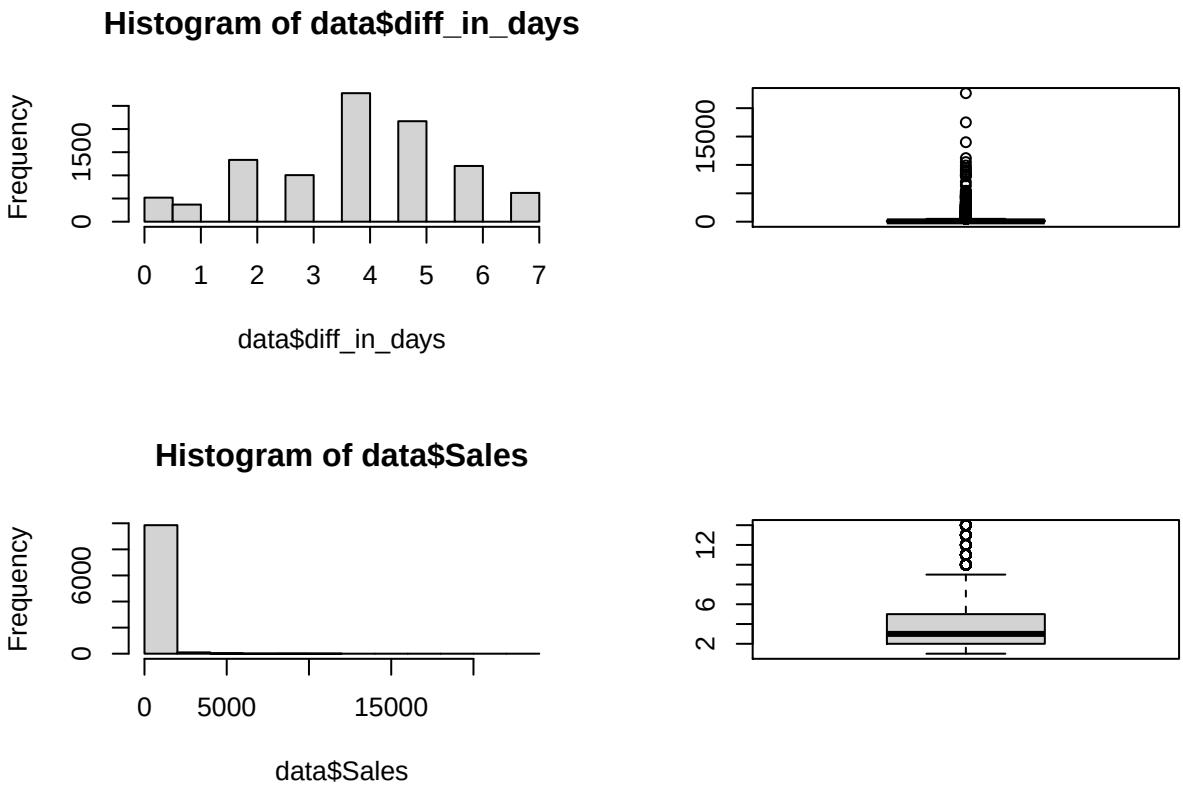



Figure 1: Clockwise from top left: a)Histogram of time difference between order date and ship date. The time difference between order date and ship date typically takes 4 days. b) Boxplot of sales showing many outliers. c) Histogram of sales values showing a large amount of orders with very small Sales!, d) Boxplot of quantity values. Many transactions involve very few items, with median value of 3 items

```r
summary(data$Sales)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##     0.444    17.280    54.490   229.858   209.940 22638.480
```

```r
ggplot(data = data, mapping = aes(x = Quantity)) +
  geom_histogram(binwidth = 0.5,fill="green4")
```

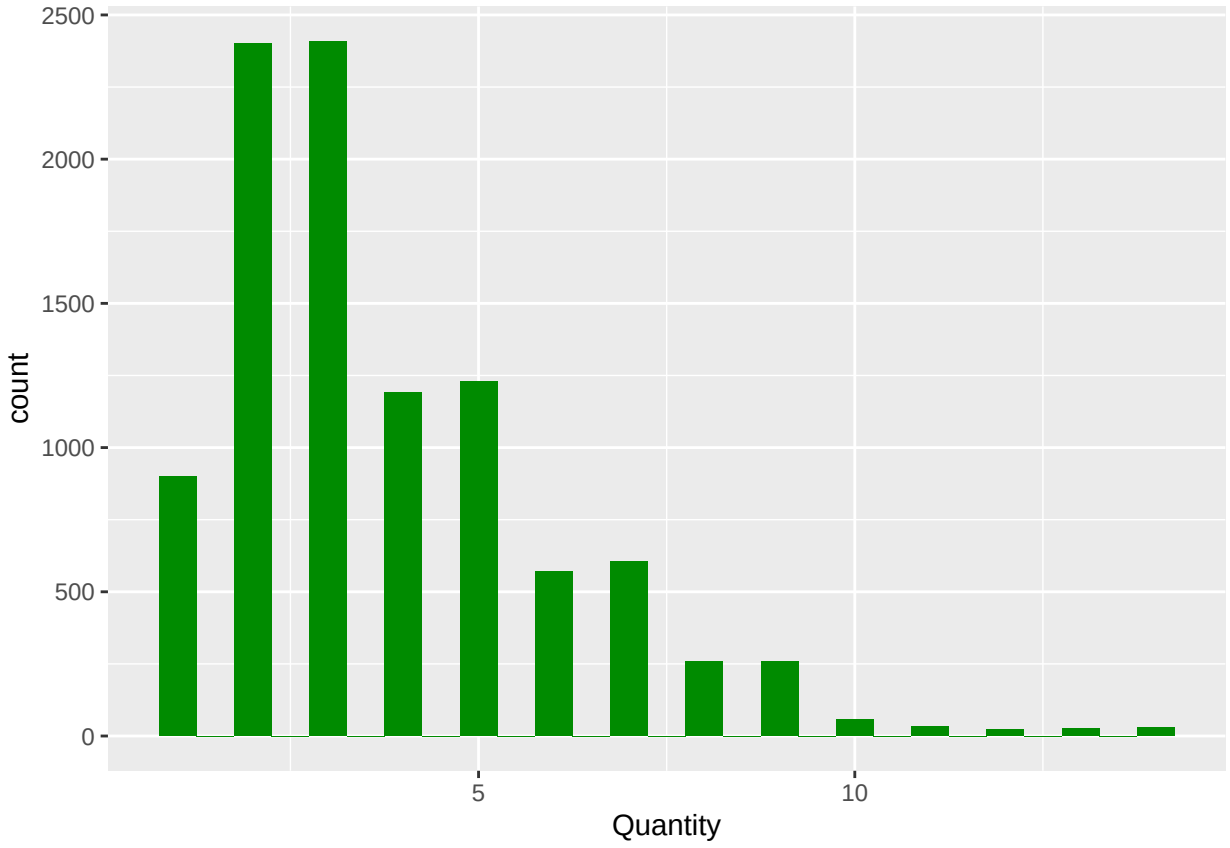



Figure 2: Plot of quantity distribution of transaction. There is a very skewed distribution where most of the orders have small number of items and there are not a lot of outliers

```r
summary(data$Quantity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    2.00    3.00    3.79    5.00   14.00
```

```r
par(mfrow=c(2,1))
boxplot(data$Discount)
boxplot(data$Profit)
```

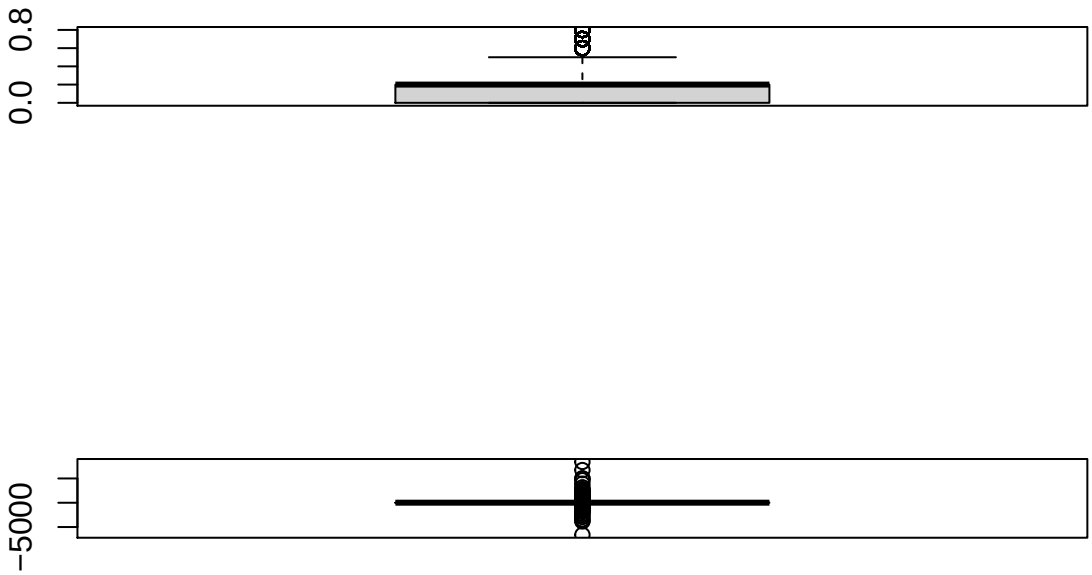



Figure 3: From top: a) Boxplot of discount values. A strange looking box dataplot - median & 3rd quantile are the same (0.2) and not many orders have high discounts. b)Boxplot of profit. Most of the profits are outside of the box - but most of them clustered close to the box(not with so extreme values

```r
summary(data$Discount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.2000  0.1562  0.2000  0.8000
```

```r
summary(data$Profit)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -6599.978     1.729     8.666    28.657    29.364  8399.976
```

```r
ggplot(data = data) +
  geom_histogram(mapping = aes(x = Discount),
                 binwidth = 0.05,
                 xlab="Discount",
                 fill="green4")
```

```
## Warning: Ignoring unknown parameters: xlab
```

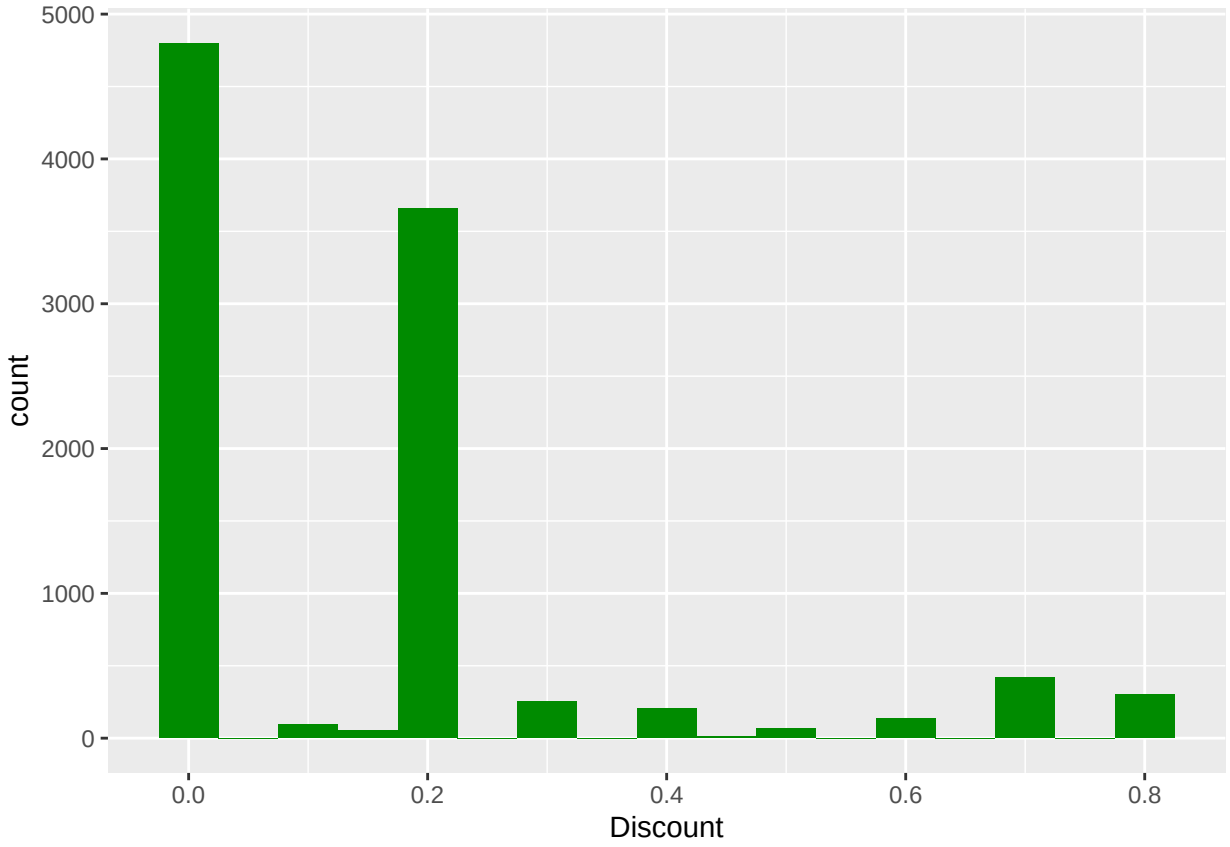



Figure 4: Plot of transactions involving discounts. Sales transactions mostly do not involve discounts.

```
ggplot(data)+
geom_histogram(mapping=aes(x=Profit),fill="green3")+
coord_cartesian(ylim = c(0, 100))+
labs(title=" Profit Distribution")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

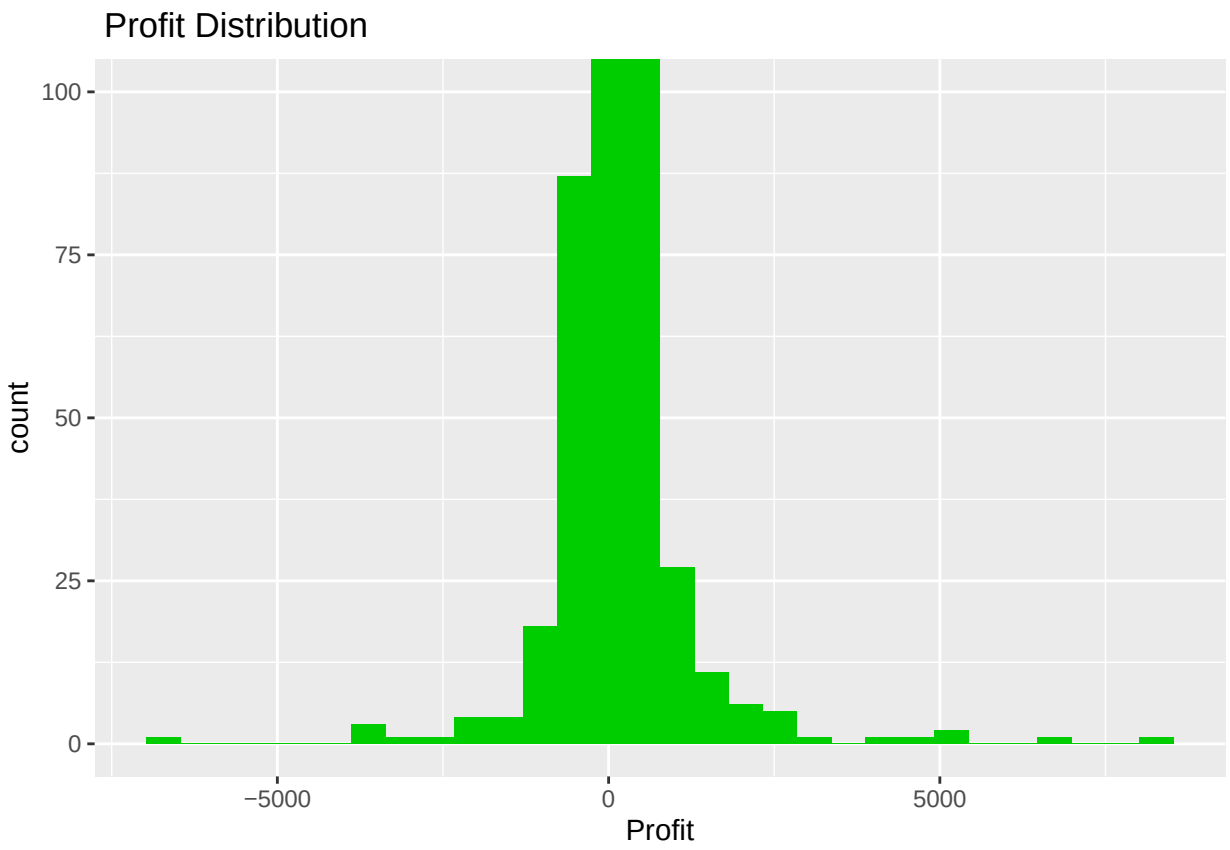



Figure 5: Profit distribution from transactions

```
#most of the orders have profits ~1000 (or ~800?), and ~ -800
```

Remove the dot in the column names and replace with "_" to make variable names easier to handle:

```
##  [1] "ï__Row_ID"     "Order_ID"      "Order_Date"    "Ship_Date"
##  [5] "Ship_Mode"     "Customer_ID"   "Customer_Name" "Segment"
##  [9] "Country"       "City"          "State"         "Postal_Code"
## [13] "Region"        "Product_ID"    "Category"      "Sub_Category"
## [17] "Product_Name"  "Sales"         "Quantity"      "Discount"
## [21] "Profit"        "diff_in_days"
```

Plot Sales in relation to Order Date:

Plot Profit in relation to Order Date:

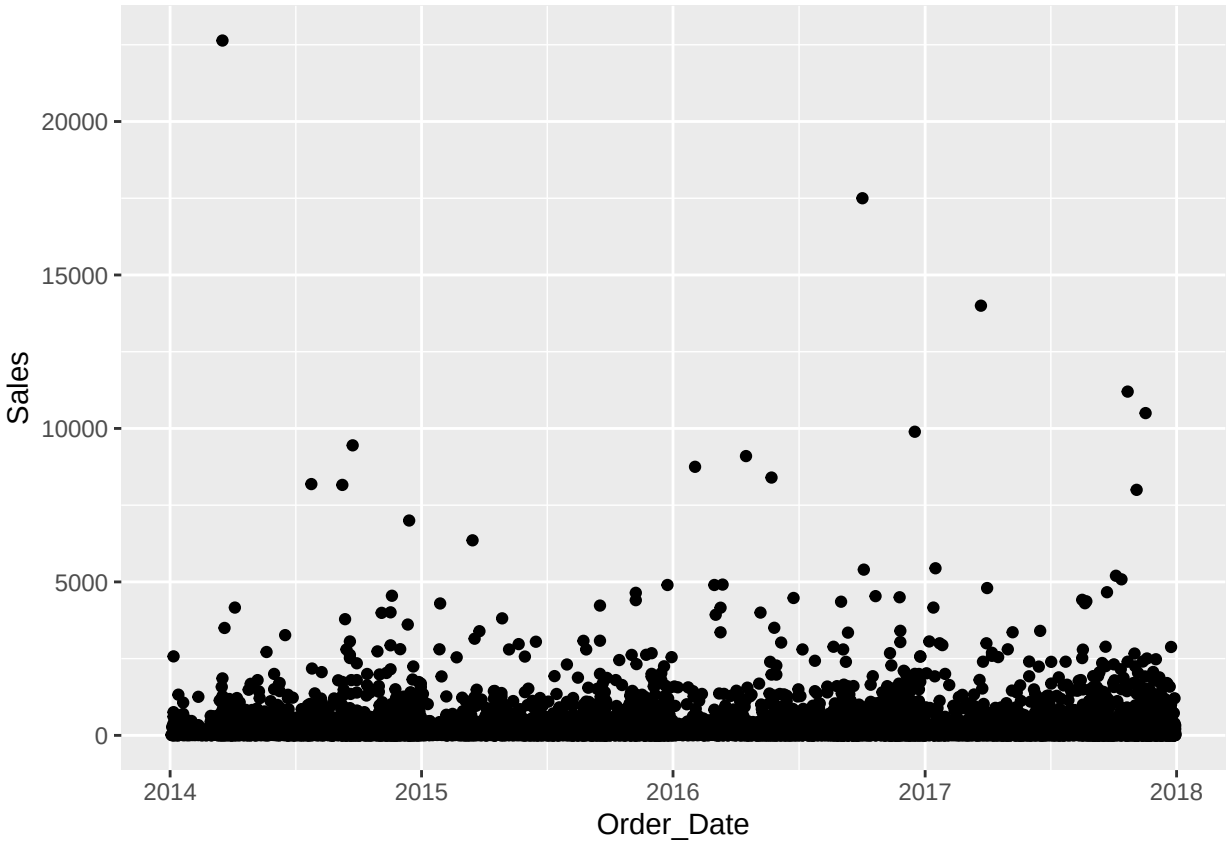



Figure 6: Sales by order date. There are some clear outliers where the sales amount is much greater

```
ggplot(data = data) +
  geom_point(mapping = aes(x = Order_Date, y = Profit), xlab="Order Date", ylab="Profit")
```

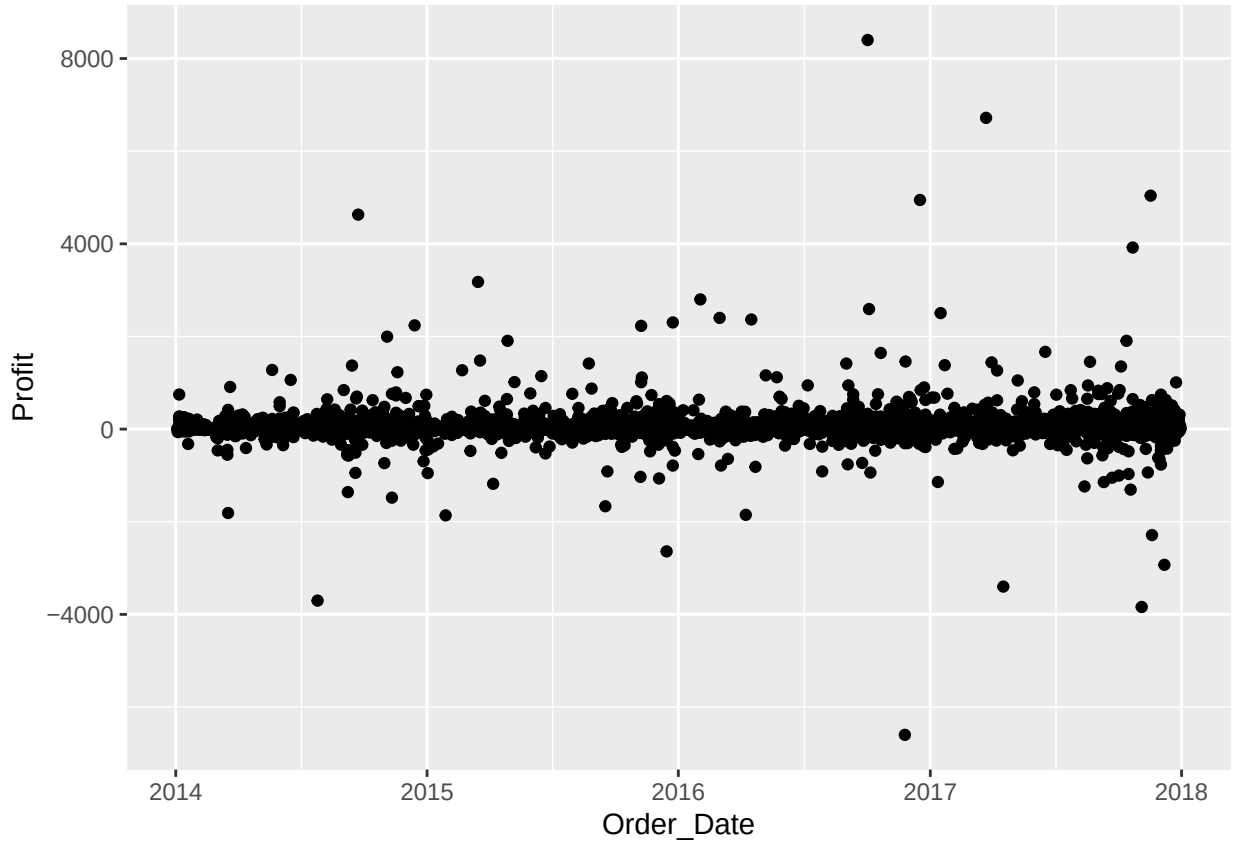



Figure 7: Profit in relation to order date. There are some outliers for certain days where profit is much lower or higher than the usual amount, such as when profit is below $2000, or above $2000.

```
table(data$`Sub_Category`)
```

```
##
## Accessories  Appliances         Art     Binders   Bookcases      Chairs
##         775         466         796        1523         228         617
##     Copiers   Envelopes   Fasteners Furnishings      Labels    Machines
##          68         254         217         957         364         115
##       Paper      Phones     Storage    Supplies      Tables
##        1370         889         846         190         319
```

look at the time range for these transactions, ie. start date for Order_Date column:

```
summary(data$Order_Date)
```

```
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "2014-01-03" "2015-05-23" "2016-06-26" "2016-04-30" "2017-05-14" "2017-12-30"
```

```
#[1] min "2014-01-03", max "2017-12-30"
```

Basically this dataset covers transactions ranging from 2014-01-03 to 2017-12-30.

```
ggplot(data = data) +
  geom_bar(mapping = aes(x = Category),fill="green4")
```

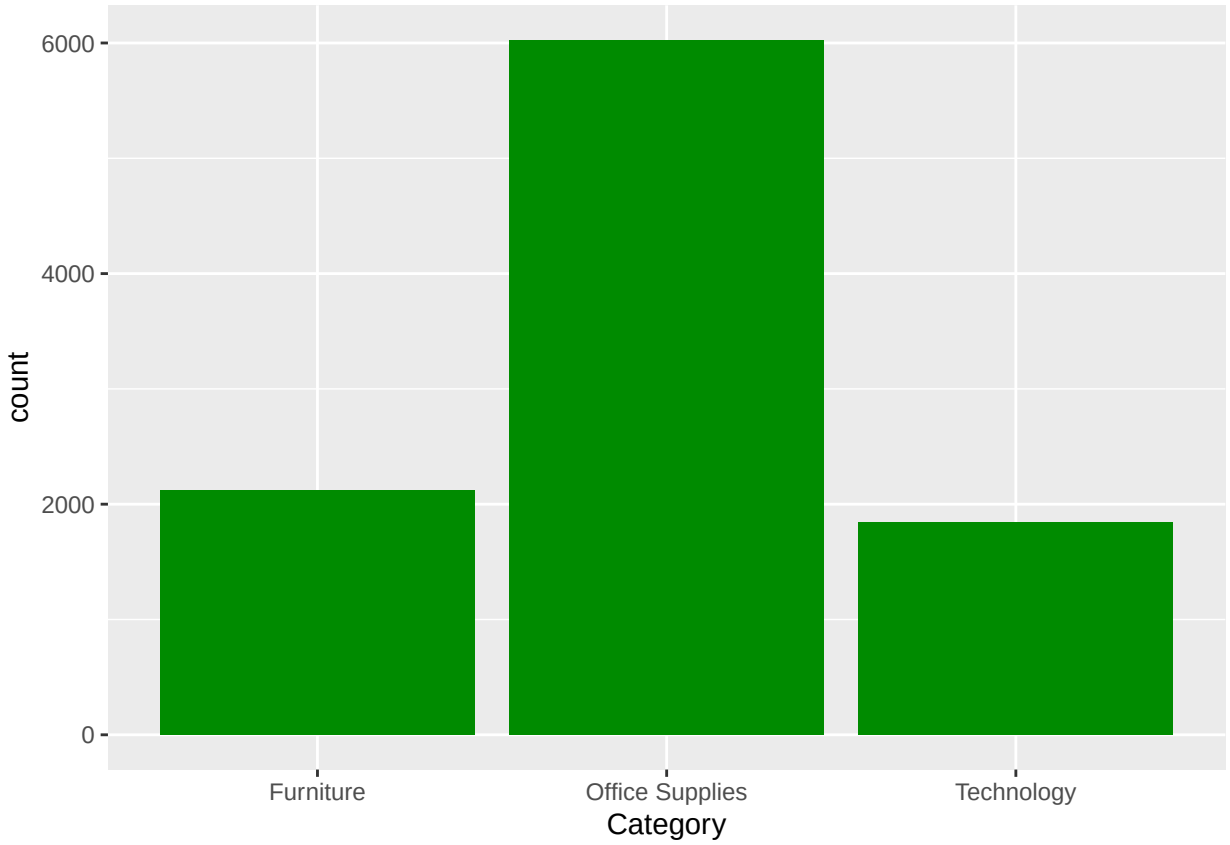



Figure 8: Barchart of product category versus count of transactions. Most type of products sold belong to the Office supplies category.

```
ggplot(data = data) +
  geom_bar(mapping = aes(y = `Sub_Category`), fill="green4")
```

```
ggplot(data = data, mapping = aes(x = Sales)) +
  xlim(0, 5000) +
  geom_histogram(binwidth = 5,fill="green4")
```

Visualise sales transactions by Region over time (order date).

```
 ggplot(data, aes(Order_Date, Sales,color=Region)) +
      geom_line()+
  ylim(0,5000)
```

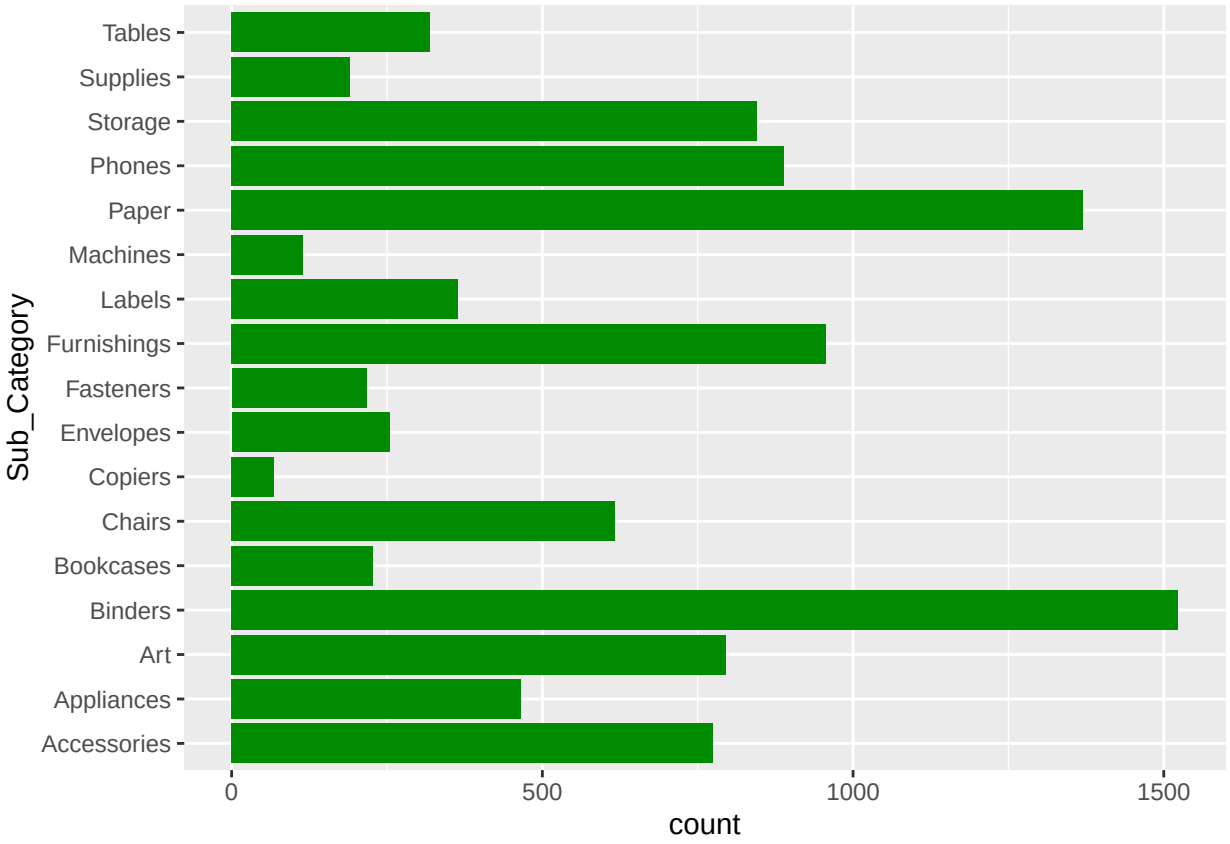



Figure 9: Barchart of product subcategory versus transaction count.

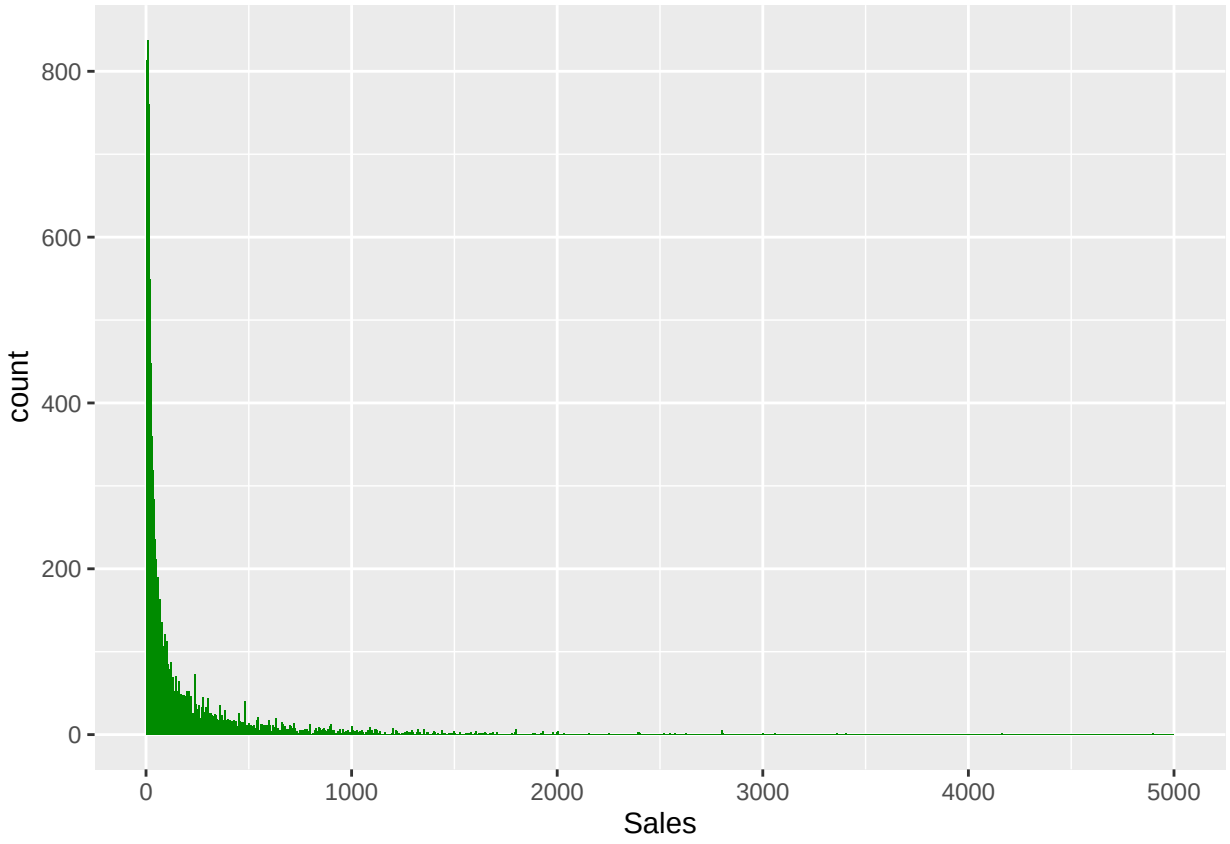
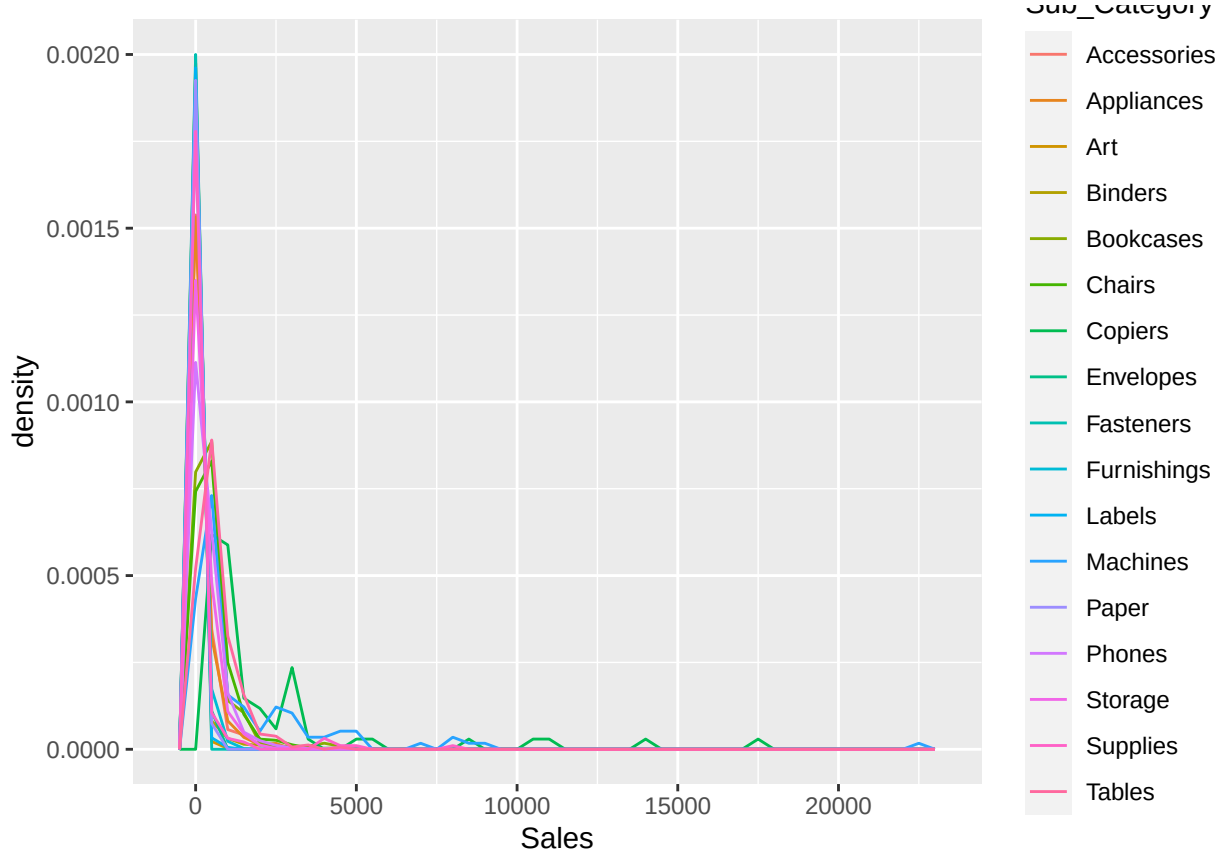



Figure 10: Histogram of sales transactions. Most sales are very few items (<500).


Figure 11: Sales over time (order date), coloured by region

How does profit change with sub-category?

```
 #density plot where the count is standardized, area under each frequency is 1
ggplot(data = data, mapping = aes(x = Sales, y = ..density..)) +
  geom_freqpoly(mapping = aes(colour = Sub_Category), binwidth = 500)
```




It looks like some categories of items ie. supplies or accessories have negative sales values.

How does sales vary across sub category?

```
ggplot(data = data, mapping = aes(x = Sales, y = `Sub_Category` )) +
  geom_boxplot()
```

```
ggplot(data =data, mapping = aes(x = Ship_Mode)) +
  geom_bar()
```

```
ggplot(data) +
geom_point(mapping = aes(x = Profit, y = Discount),colour="violetred3")+
labs(title=" Profit Discount Distribution")
```

```
#Most of the orders were placed without any discounts or with 20% off
```

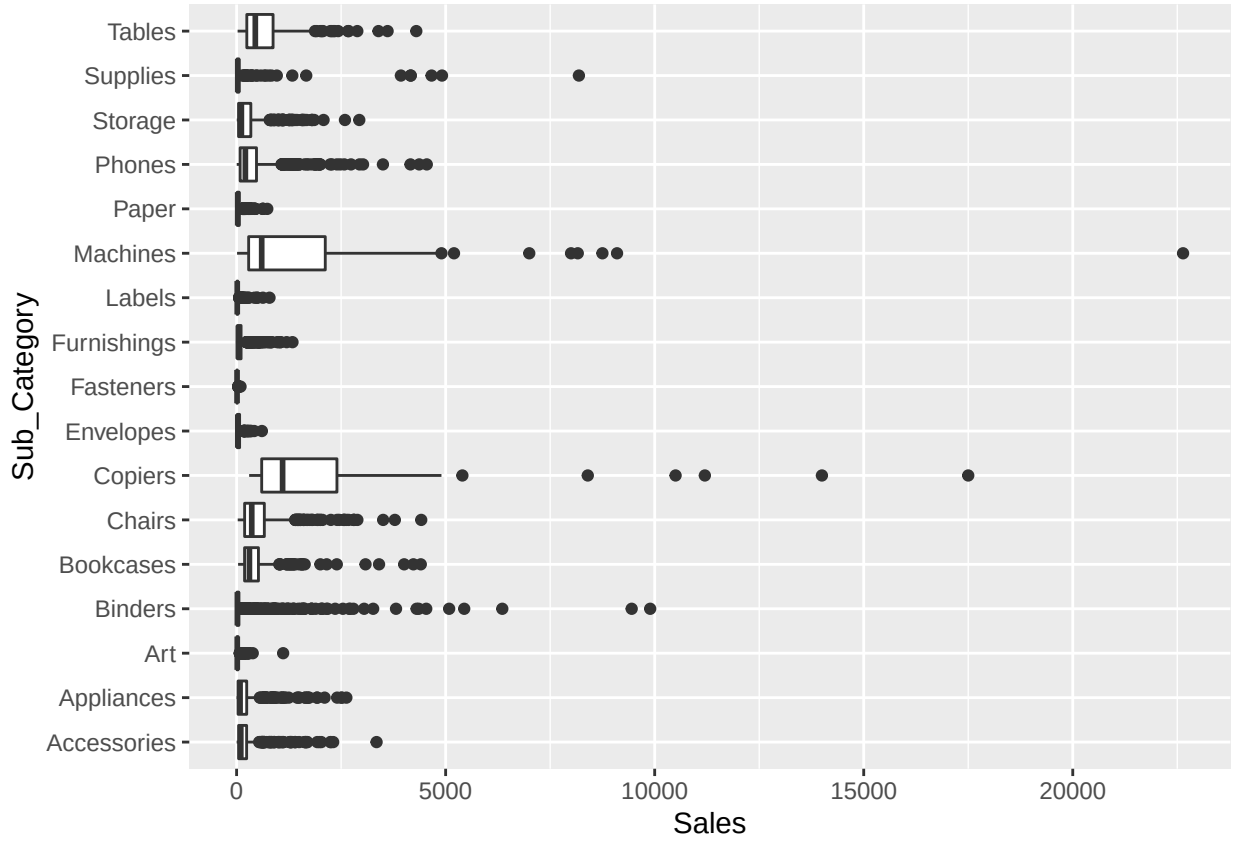



Figure 12: Boxplot of sales by sub category of products. Machines and copiers generate the highest sales amounts.

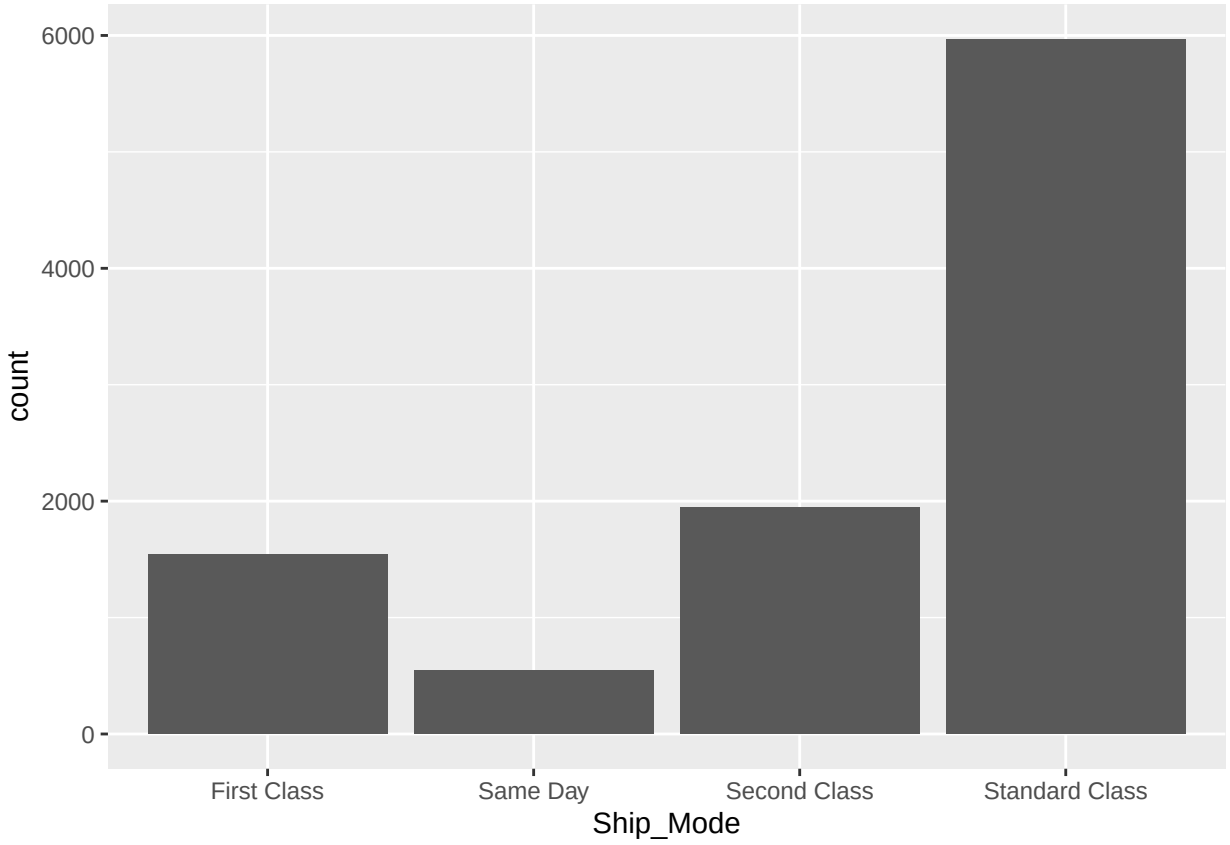



Figure 13: Bar chart of transactions by ship mode. Most transactions are shipped via Standard Class method.

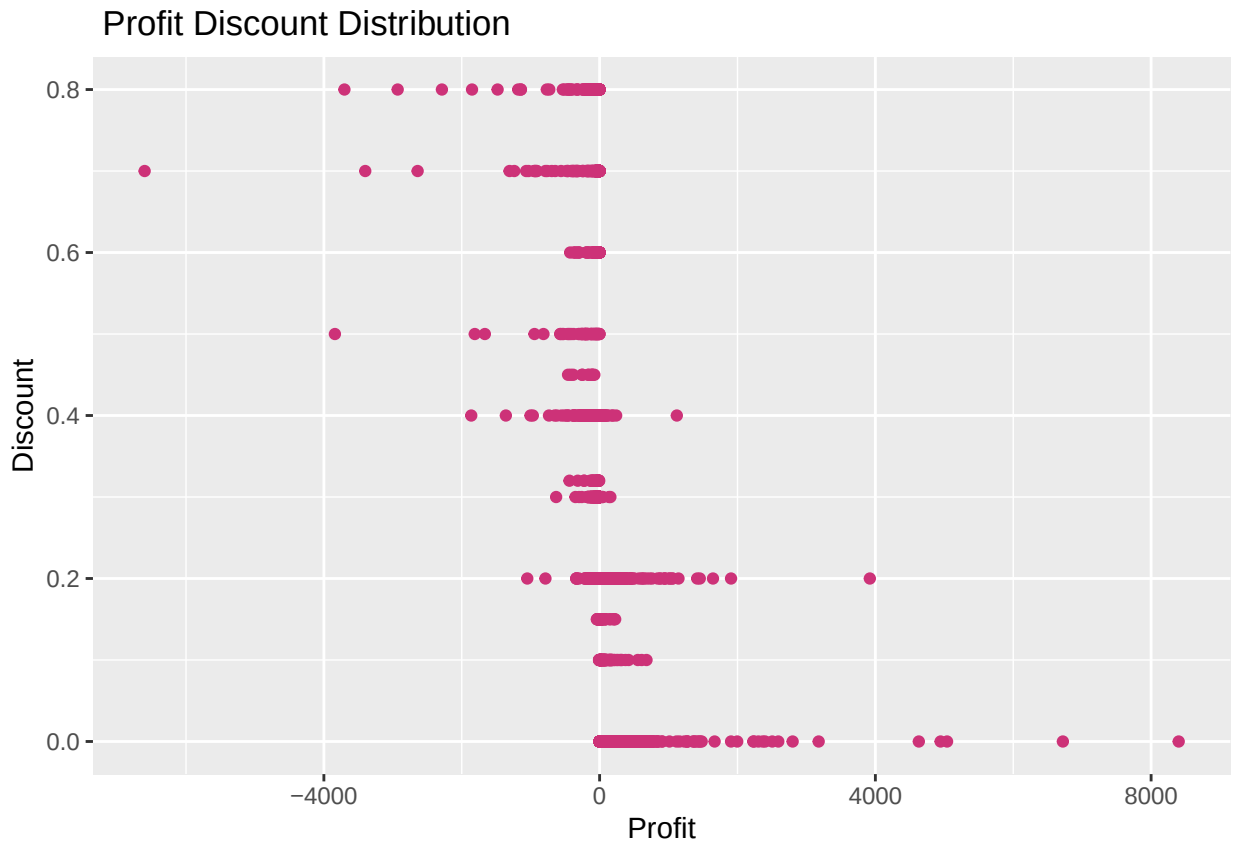



Figure 14: Plot of profit versus discount

**Sales Profit**

```
ggplot(data) +
geom_point(mapping = aes(x = Sales, y = Profit),colour="limegreen")+
labs(title=" Sales Profit Distribution")
```

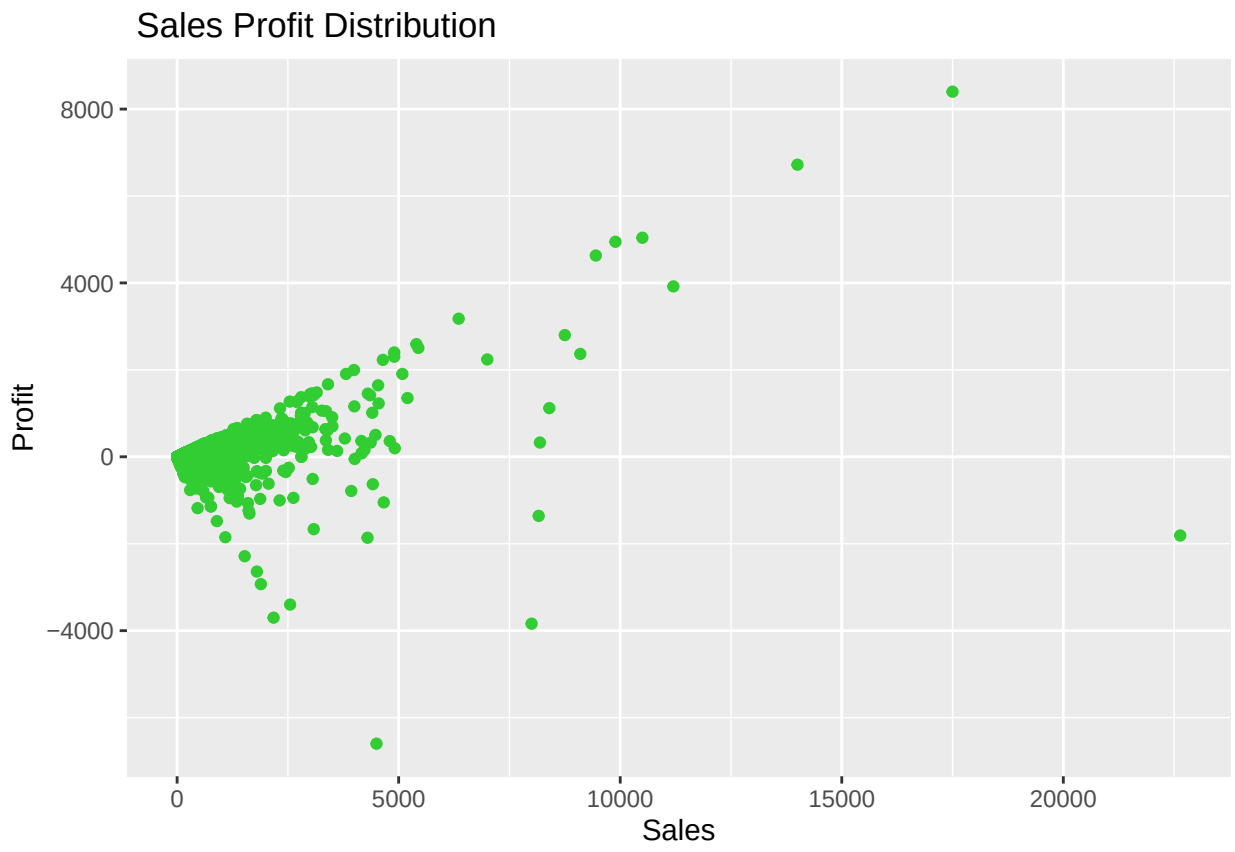



Figure 15: Scatterplot of sales vs profit.

```
#product name and product id mismatch
data %>%
  distinct(Product_Name,Product_ID) %>%
  group_by(Product_ID) %>%
  filter(n()>1) %>%
  select(Product_ID)
```

```
## # A tibble: 64 x 1
## # Groups:   Product_ID [32]
##    Product_ID
##    <chr>
##  1 FUR-FU-10004848
##  2 FUR-CH-10001146
##  3 OFF-BI-10004654
##  4 FUR-CH-10001146
```

```
##  5 OFF-PA-10002377
##  6 OFF-AR-10001149
##  7 OFF-PA-10000659
##  8 TEC-MA-10001148
##  9 FUR-FU-10004017
## 10 TEC-AC-10003832
## # ... with 54 more rows
```

```
data %>%
  distinct(Category, Sub_Category)
```

```
##           Category Sub_Category
## 1        Furniture    Bookcases
## 2        Furniture       Chairs
## 3  Office Supplies       Labels
## 4        Furniture       Tables
## 5  Office Supplies      Storage
## 6        Furniture  Furnishings
## 7  Office Supplies          Art
## 8       Technology       Phones
## 9  Office Supplies      Binders
## 10 Office Supplies   Appliances
## 11 Office Supplies        Paper
## 12      Technology  Accessories
## 13 Office Supplies    Envelopes
## 14 Office Supplies    Fasteners
## 15 Office Supplies     Supplies
## 16      Technology     Machines
## 17      Technology      Copiers
```

```
superstore_sales<-data %>%
                  select(Order_Date,Sales)


superstore_sales<-as_tibble(superstore_sales)
```

Transactions by region:

```
bar <- ggplot(data = data) +
  geom_bar(
    mapping = aes(x = Region, fill = Region),
    show.legend = FALSE,
    width = 1
  ) +
  theme(aspect.ratio = 1) +
  labs(x = NULL, y = NULL)

bar + coord_polar()
```

```
#Extracting the rows for South region, and sub-categories:
South <- data %>%
```

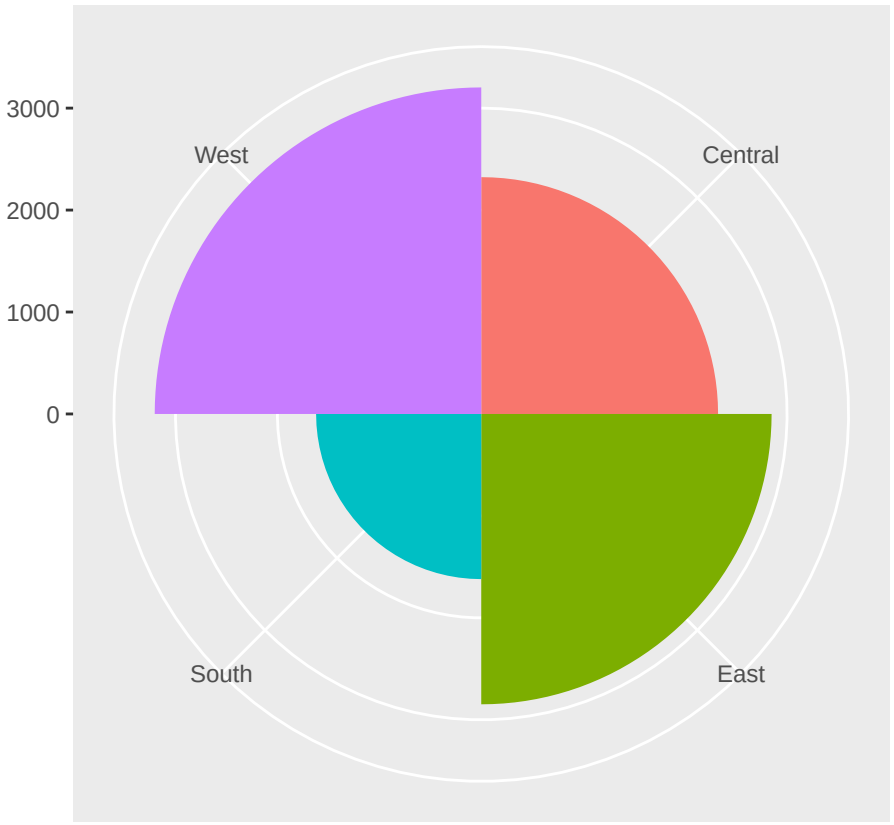
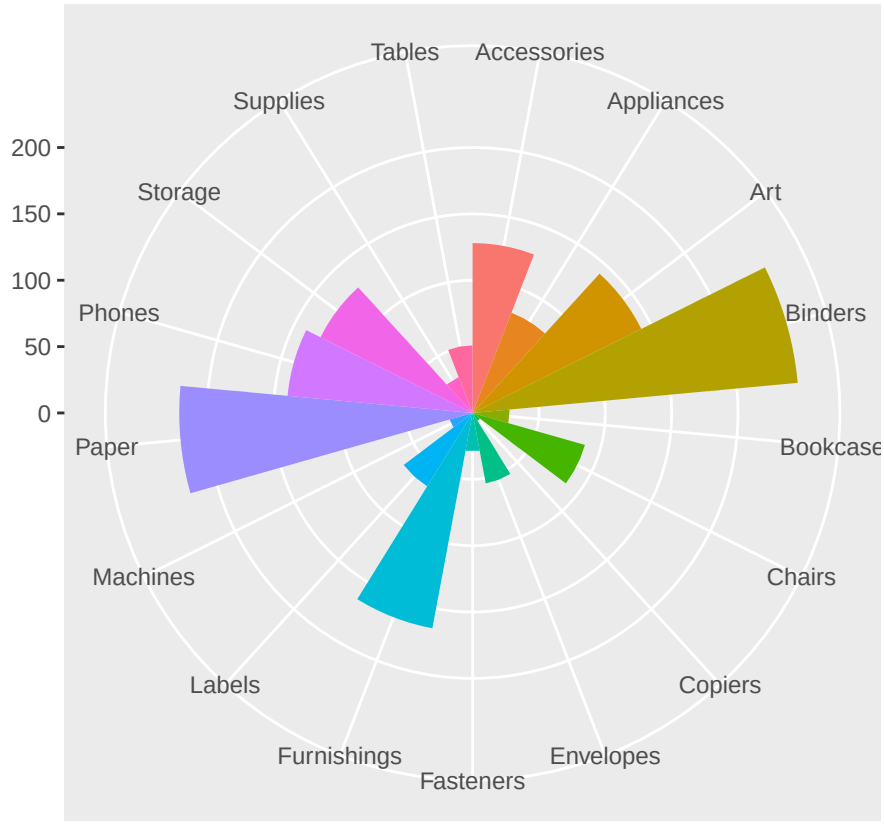



Figure 16: Pie chart which shows proportions of transactions from the different regions.

```
  select(Region, Sub_Category) %>%
  filter(Region == "South")

bar <- ggplot(data = South) +
  geom_bar(
    mapping = aes(x = Sub_Category, fill = Sub_Category),
    show.legend = FALSE,
    width = 1
  ) +
  theme(aspect.ratio = 1) +
  labs(x = NULL, y = NULL)

bar + coord_polar()
```




In the South, most transactions are Binders, Paper, or Furnishings. For the other regions, the distribution of product sub categories are very similar to the distribution of product subcategories in the south.

Look at relationship between numeric variables:

```
#subset the numeric variables:
numeric_vars<- c("Sales", "Quantity", "Discount", "Profit", "diff_in_days")
num_data <- data[numeric_vars]
```

We'll use a correlation matrix to look at the relationship between numeric variables:

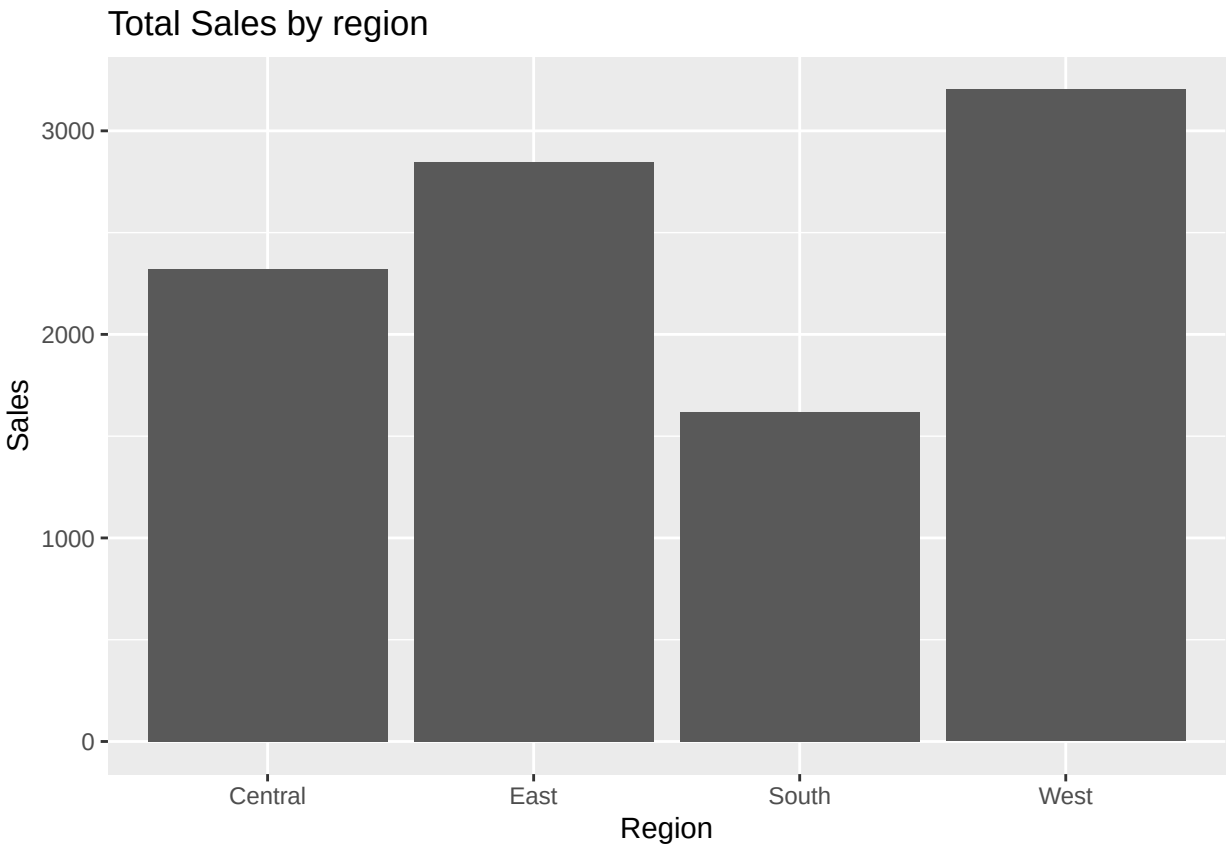



Figure 17: Total sales per region.

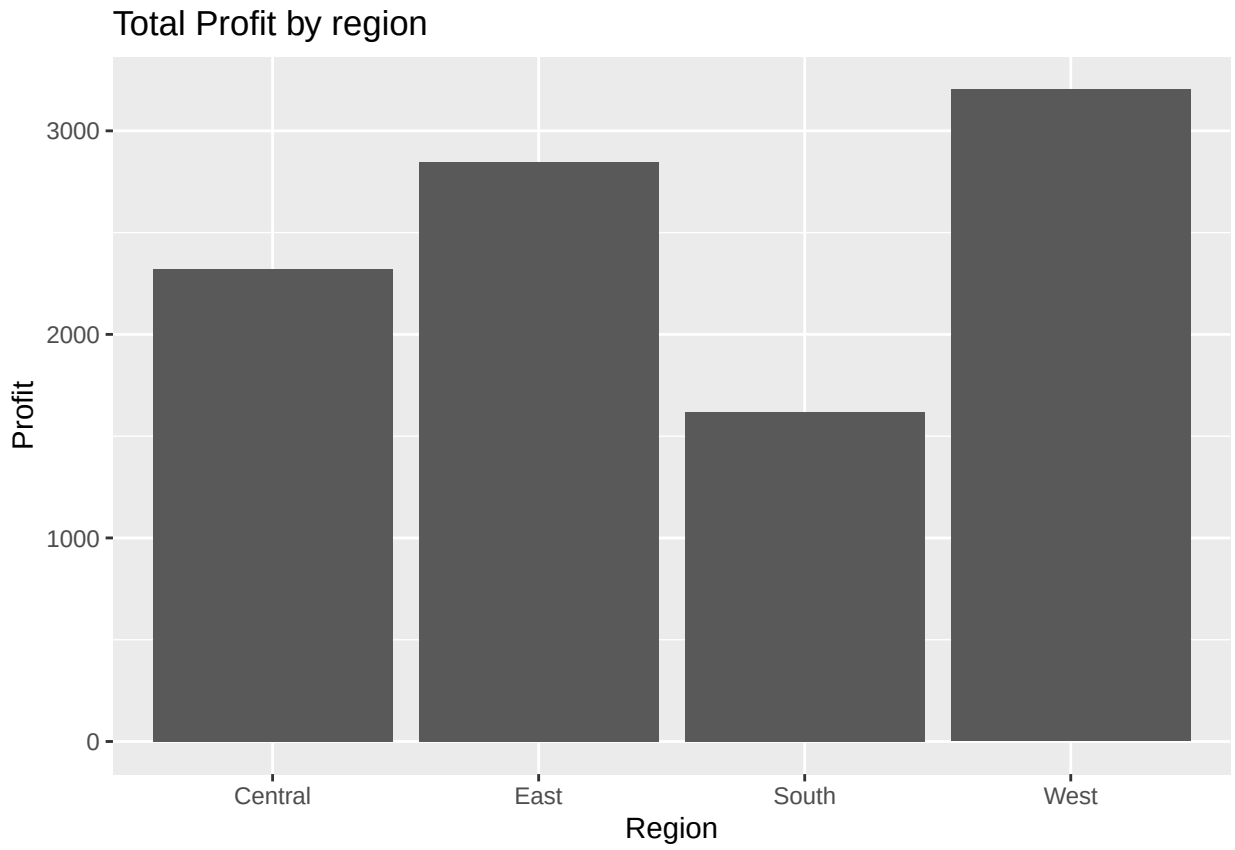



Figure 18: Total profit by region

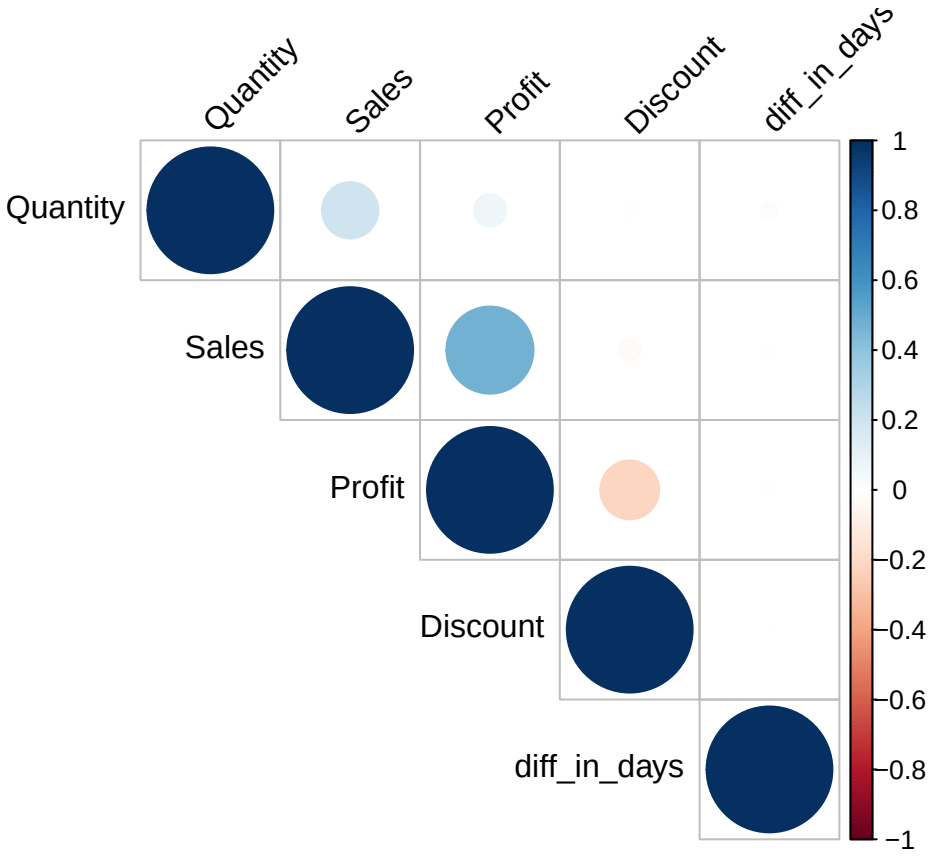
```
cor(num_data)
```

```
##                    Sales   Quantity      Discount       Profit  diff_in_days
## Sales         1.000000000 0.20079477 -0.0281901242  0.479064350 -0.0073535371
## Quantity      0.200794771 1.00000000  0.0086229703  0.066253189  0.0182984399
## Discount     -0.028190124 0.00862297  1.0000000000 -0.219487456  0.0004084856
## Profit        0.479064350 0.06625319 -0.2194874564  1.000000000 -0.0046493531
## diff_in_days -0.007353537 0.01829844  0.0004084856 -0.004649353  1.0000000000
```

```
#correlation matrix with statistical significance
cor_result=rcorr(as.matrix(num_data))

cor_result$r
```

```
##                    Sales   Quantity      Discount       Profit  diff_in_days
## Sales         1.000000000 0.20079477 -0.0281901242  0.479064350 -0.0073535371
## Quantity      0.200794771 1.00000000  0.0086229703  0.066253189  0.0182984399
## Discount     -0.028190124 0.00862297  1.0000000000 -0.219487456  0.0004084856
## Profit        0.479064350 0.06625319 -0.2194874564  1.000000000 -0.0046493531
## diff_in_days -0.007353537 0.01829844  0.0004084856 -0.004649353  1.0000000000
```

```
corrplot(cor_result$r, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)  #display only
```




Discount is negatively correlated with profit, whereas sales is positively correlated with profit. The time between order date and ship date (diff_in_days) is not correlated with sales, quantity, discount, or profit.

### Data Preparation

```
#make a copy of the original dataset and copy to data1
data1 <- data
```

drop column Row ID because it is not necessary; it is the row number from the original excel file. The country variable is also not needed because all the values are United States. Customer_Name and Customer_ID give redundant information. So we will drop the Customer_Name column and keep only the Customer_ID column.

```
data1[,c("Row_ID","ï__Row_ID", "Country", "Customer_Name")]<-NULL
```

```
head(data1)
```

```
##         Order_ID Order_Date  Ship_Date      Ship_Mode Customer_ID   Segment
## 1 CA-2016-152156 2016-11-08 2016-11-11   Second Class    CG-12520  Consumer
## 2 CA-2016-152156 2016-11-08 2016-11-11   Second Class    CG-12520  Consumer
## 3 CA-2016-138688 2016-06-12 2016-06-16   Second Class    DV-13045 Corporate
## 4 US-2015-108966 2015-10-11 2015-10-18 Standard Class    SO-20335  Consumer
## 5 US-2015-108966 2015-10-11 2015-10-18 Standard Class    SO-20335  Consumer
## 6 CA-2014-115812 2014-06-09 2014-06-14 Standard Class    BH-11710  Consumer
##              City      State Postal_Code Region      Product_ID        Category
## 1       Henderson   Kentucky       42420  South FUR-BO-10001798       Furniture
## 2       Henderson   Kentucky       42420  South FUR-CH-10000454       Furniture
## 3     Los Angeles California       90036   West OFF-LA-10000240 Office Supplies
## 4 Fort Lauderdale    Florida       33311  South FUR-TA-10000577       Furniture
## 5 Fort Lauderdale    Florida       33311  South OFF-ST-10000760 Office Supplies
## 6     Los Angeles California       90032   West FUR-FU-10001487       Furniture
##   Sub_Category                                                  Product_Name
## 1    Bookcases                                Bush Somerset Collection Bookcase
## 2       Chairs      Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back
## 3       Labels        Self-Adhesive Address Labels for Typewriters by Universal
## 4       Tables                     Bretford CR4500 Series Slim Rectangular Table
## 5      Storage                                   Eldon Fold 'N Roll Cart System
## 6  Furnishings Eldon Expressions Wood and Plastic Desk Accessories, Cherry Wood
##      Sales Quantity Discount    Profit diff_in_days
## 1 261.9600        2     0.00   41.9136            3
## 2 731.9400        3     0.00  219.5820            3
## 3  14.6200        2     0.00    6.8714            4
## 4 957.5775        5     0.45 -383.0310            7
## 5  22.3680        2     0.20    2.5164            7
## 6  48.8600        7     0.00   14.1694            5
```

# Modelling

### Method 1: Clustering with Kmeans without removing outliers

For this K-means clustering we will use the numeric variables only: which are sales, quantity, discount, profit (columns 15 - 18). K means clustering is a partitional, exclusive and complete type of clustering approach, meaning the clustering is independent, (not nested within each other), and every observation has to be placed

inside a cluster (Nwanganga and Chapple, 2020). K means clustering is affected by the starting assignment points, so we will try with 25 different starting assignments (nstart = 25), and see which ones work the best.

```
summary(results_kmeans)
```

```
##              Length Class  Mode
## cluster      9994   -none- numeric
## centers        12   -none- numeric
## totss           1   -none- numeric
## withinss        3   -none- numeric
## tot.withinss    1   -none- numeric
## betweenss       1   -none- numeric
## size            3   -none- numeric
## iter            1   -none- numeric
## ifault          1   -none- numeric
```

K-means clustering with 3 clusters of sizes 1136, 8831, 27

Cluster means: Sales Quantity Discount Profit 1 -0.05714414 0.045908572 2.3730228 -0.5928228 2 -0.02922217 -0.007823215 -0.3039892 0.0425665 3 11.96210167 0.627210176 -0.4157497 11.0200717

Within cluster sum of squares by cluster: [1] 5207.851 16648.669 3315.203 (between_SS / total_SS = 37.0 %)

Available components:

[1] “cluster” “centers” “totss” “withinss” “tot.withinss” “betweenss” “size” “iter”
[9] “ifault”

The results of this clustering indicate that the within cluster sum of squares by cluster is 37.0 % which means that the observations within a given group are not very similar to each other.

## Plot K-means

The factoextra package contains a function called fviz_cluster() which can be used to visualize kmeans clusters. The input required is the original dataset, and the kmeans results. These are used to produce plots which show points that represent observations.

```
fviz_cluster(results_kmeans, data = data1[,(15:18)],
             palette = c("#2E9FDF", "#E495A5", "#E7B800"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```

## Reduce dimensions using Principal Component Analysis.

```
results_pca <- prcomp(data1[,(15:19)], scale=TRUE)

#Coordinates of individual observations
indiv_coordinates <- as.data.frame(get_pca_ind(results_pca)$coord)
```

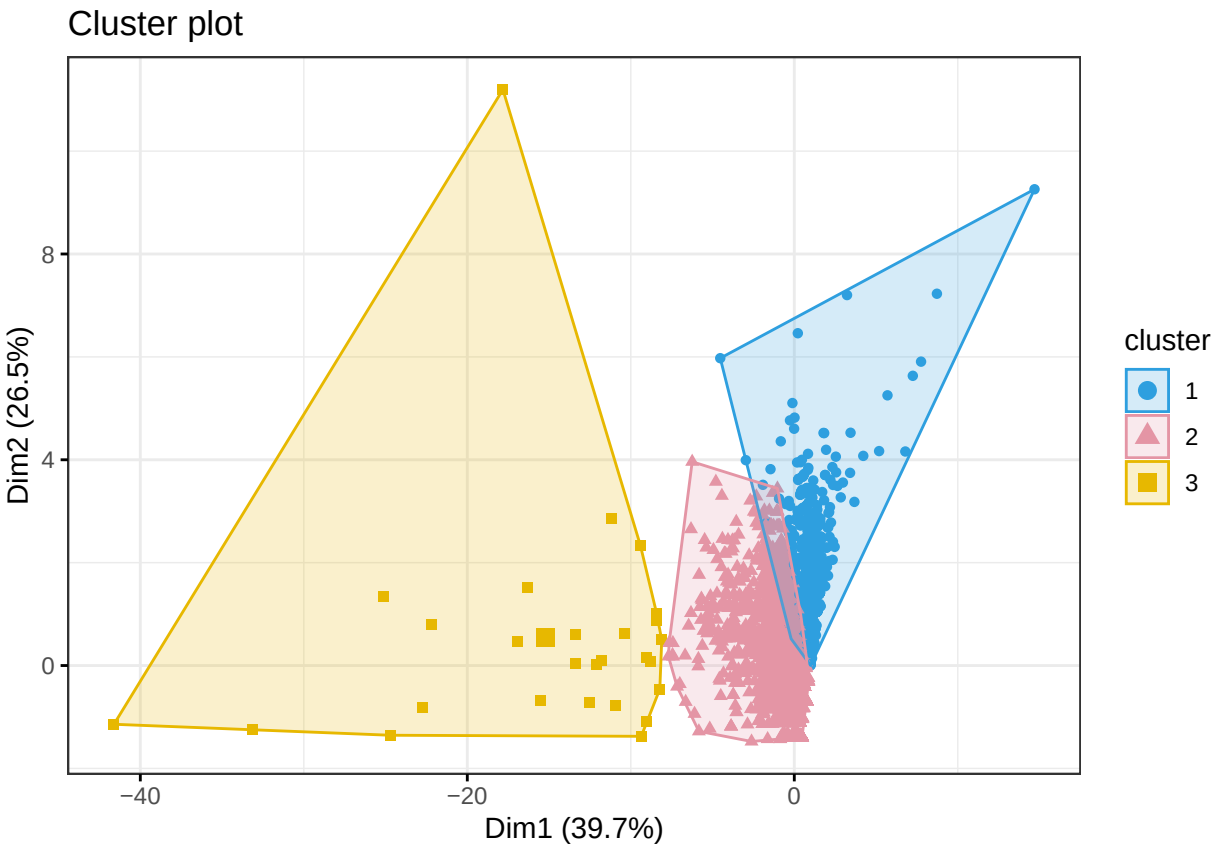



Figure 19: Plot of Kmeans clustering of numeric variables from Superstore transactions dataset, for k=3

```r
#Add clusters obtained through the Kmeans algorithm
indiv_coordinates$cluster <- factor(results_kmeans$cluster)

#Add region from the dataset
indiv_coordinates$Region <- data1$Region

#look at the first few rows of individual coordinates
head(indiv_coordinates)
```

```
##         Dim.1       Dim.2      Dim.3         Dim.4       Dim.5 cluster Region
## 1  0.04520418 -1.13666870  0.3612739  0.0807637427  0.30768311       2  South
## 2  1.15718390 -0.84275631  0.4252513  0.1851981706  0.21856052       2  South
## 3 -0.31072375 -1.08484635 -0.2274155 -0.0007838382  0.15040037       2   West
## 4 -0.63620673  2.24187253 -1.2663321  0.5805617103  1.55922466       1  South
## 5 -0.58444157 -0.08714015 -1.6947210  0.7880697439 -0.05040709       2  South
## 6  0.39080002  0.44622564 -0.7312599 -1.4739348647 -0.19665673       2   West
```

Percentage of variance explained by dimensions.

```r
eigenvalue <- round(get_eigenvalue(results_pca), 1)
variance.percent <- eigenvalue$variance.percent
head(eigenvalue)
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1        1.6             31.8                          31.8
## Dim.2        1.1             21.2                          53.0
## Dim.3        1.0             20.0                          73.0
## Dim.4        0.9             17.6                          90.6
## Dim.5        0.5              9.4                         100.0
```

Variance of a group indicates how different members of a group are. Higher variance means greater dissimilarity within a group.

```r
#To visualize the k-means clusters:

ggscatter(
  indiv_coordinates, x = "Dim.1", y = "Dim.2",
  color = "cluster", palette = "npg", ellipse = TRUE, ellipse.type = "convex",    #adding the concentra
  shape = "Region", size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 1 (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 2 (", variance.percent[2], "% )" )
) +
  stat_mean(aes(color = cluster), size = 4)      #stat_mean is used for adding the cluster centroid
```

The clustering plot shows that the groups are very close together, and overlap slightly. The clusters could be further apart with some tuning by changing the number of clusters (k).

## Evaluation of Kmeans Clustering

The within sum of squares (Withinss) is a value that represents the level of dissimilarity within a group. The higher the withinss, the greater the dissimilarity within the group (Foncesca, 2019).

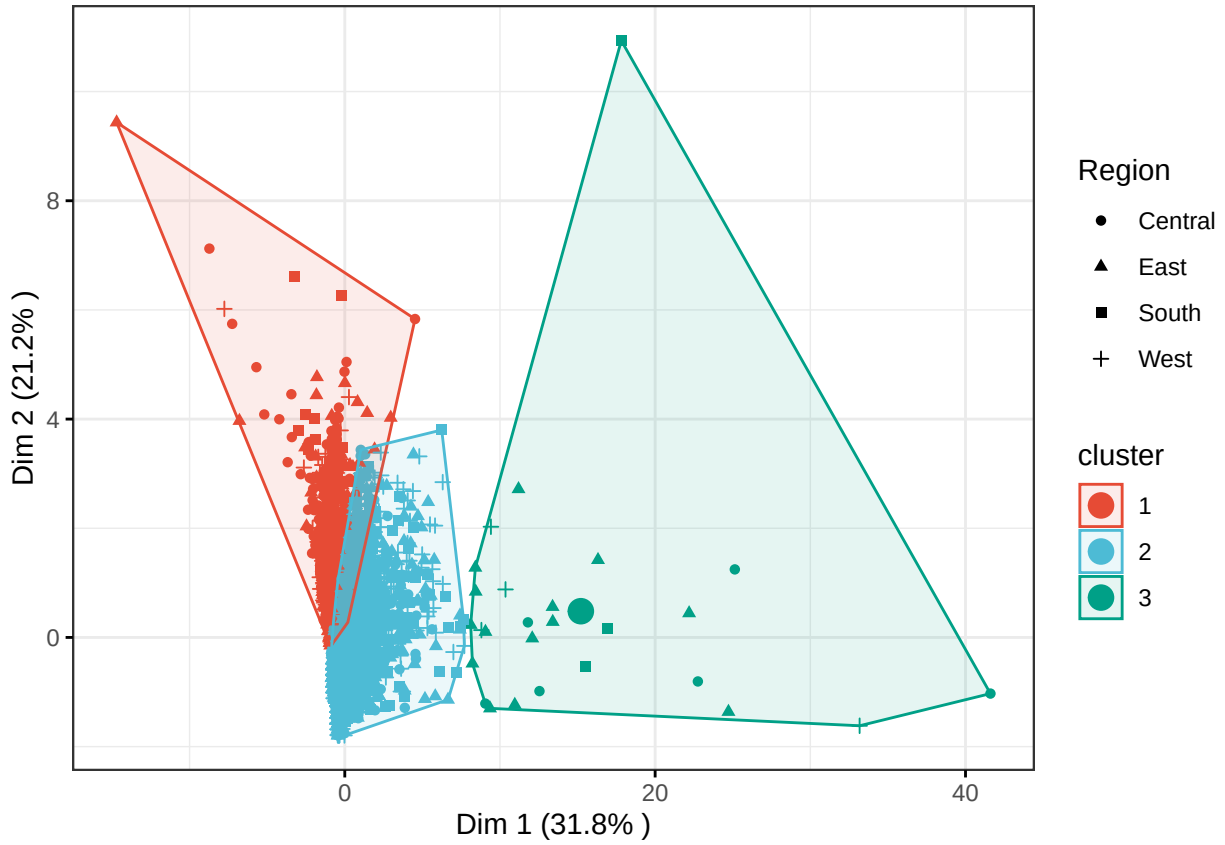



Figure 20: Alternate way to plot kmeans clustering for Superstore numeric columns, k=3, with observations labelled by region

```r
#To plot a within sum of squares plot for a range of different number of initial K-means centroids:
#This function is modified from: (https://towardsdatascience.com/clustering-analysis-in-r-using-k-means

#data is the input dataset, nc is the maximum number of initial centres

wssplot <- function(data, nc=25, seed=123){
               wss <- (nrow(data)-1)*sum(apply(data,2,var))
               for (i in 2:nc){
                    set.seed(seed)
                    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
                plot(1:nc, wss, type="b", xlab="Number of groups",
                     ylab="Sum of squares within a group")}

wssplot(data1[,(15:18)], nc = 20)
```

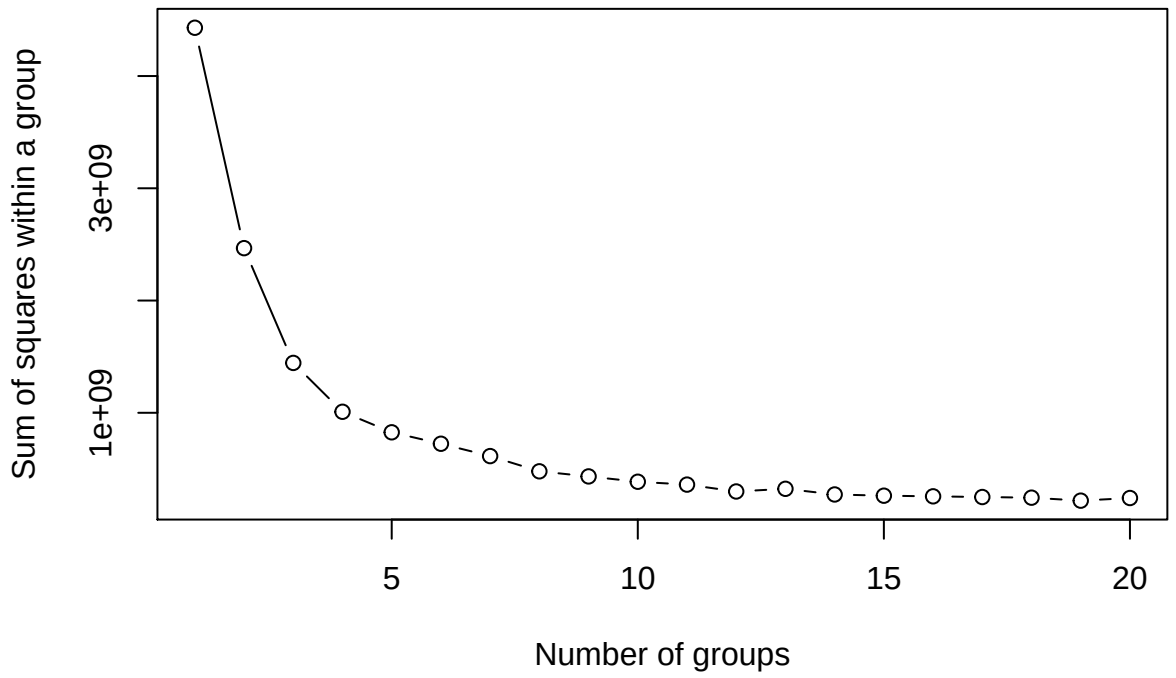



Figure 21: Within sum of squares plot for kmeans clustering of Superstore numeric data

From the Within sum of squares plot, the optimal number of clusters is around 4. When the number of groups (k) initially increases 1 to 4, the error measures (sum of squares within a group) starts to decrease. After around k=4 or k=5, the sum or sqaures within a group error value starts to mostly flatten.

The main purpose is to find a number of initial groups which achieves some fair amount of compactness (or similarity) between observations within a group. When k is too high, each cluster starts to represent individual points, whereas when k is too low, the observations may not be in the right cluster.

We can try re-running the k-means model with the number of groups, k = 4

K-means clustering with 4 clusters of sizes 27, 1044, 2838, 6085

Cluster means: Sales Quantity Discount Profit 1 11.9621017 0.62721018 -0.4157497 11.02007172 2 -0.1222964 0.02095769 2.4788189 -0.58466503 3 0.2856822 1.19688927 -0.2981228 0.17558930 4 -0.1653353 -0.56459922 -0.2844025 -0.03048054

Within cluster sum of squares by cluster: [1] 3315.203 4284.643 7571.058 3614.724 (between_SS / total_SS = 53.0 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss" "size" "iter"
[9] "ifault"

The within cluster sum of squares by cluster value is now 53.0%. This represents the compactness of the clustering, or how similar observations are to other observations within the same group.

```
# Dimension reduction using PCA
results_pca_4 <- prcomp(data1[,(15:18)],  scale = TRUE)

# Coordinates of individuals
ind.coord <- as.data.frame(get_pca_ind(results_pca_4)$coord)

# Add clusters obtained using the K-means algorithm
ind.coord$cluster <- factor(clustering_results_4$cluster)

# Add Region groups from the original data sett
ind.coord$Region <- data1$Region

#look at the first few rows to double check
head(ind.coord)
```

```
##        Dim.1      Dim.2       Dim.3       Dim.4 cluster Region
## 1 -0.0426607 -1.0508481  0.14635987  0.31244832       4  South
## 2 -1.1546701 -0.7511273  0.25351937  0.22331213       4  South
## 3  0.3107253 -1.1081474 -0.01276717  0.15000936       4   West
## 4  0.6283963  1.9637213  0.37261495  1.54367992       2  South
## 5  0.5768386 -0.4056909  0.55980463 -0.06649362       4  South
## 6 -0.3936310  0.3152509 -1.56442060 -0.20148637       3   West
```

```
# Percentage of variance explained by dimensions
eigenvalue <- round(get_eigenvalue(results_pca_4), 1)
variance.percent <- eigenvalue$variance.percent
head(eigenvalue)
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1        1.6             39.7                        39.7
## Dim.2        1.1             26.5                        66.2
## Dim.3        0.9             22.0                        88.3
## Dim.4        0.5             11.7                       100.0
```

```
ggscatter(
  ind.coord, x = "Dim.1", y = "Dim.2",
  color = "cluster", palette = brewer.pal(n = 8, name = "Dark2"), ellipse = TRUE, ellipse.type = "conve
  shape = "Region", size = 1.5,  legend = "right", ggtheme = theme_bw(),                #change point s
```

```
  xlab = paste0("Dim 1 (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 2 (", variance.percent[2], "% )" )
) +
  stat_mean(aes(color = cluster), size = 4)      #add cluster centroid using stat_mean()
```

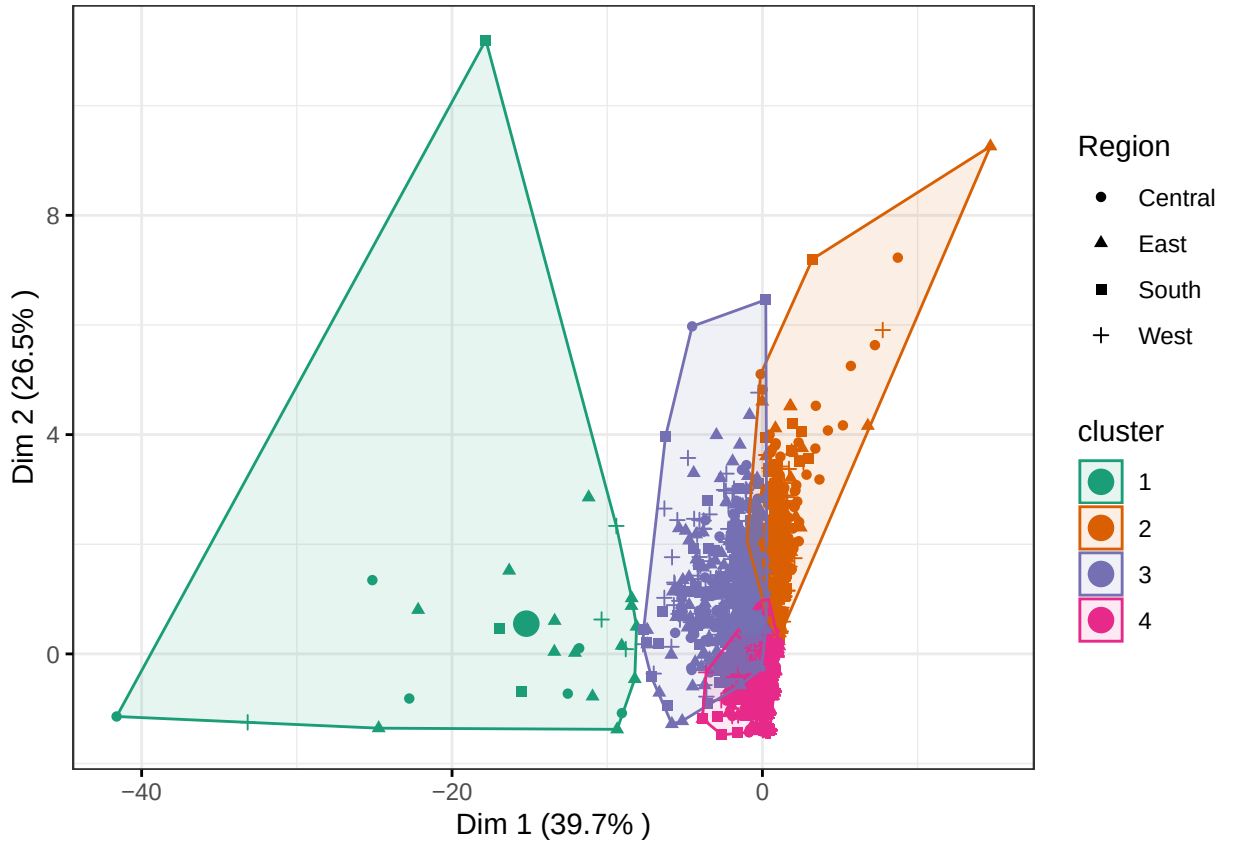



Figure 22: K means clusters of Superstore numeric data for k=4. Observations are labeled by region.

## Clustering Validation

Silhouette coefficient can be used to evaluate the goodness of the clustering. First, for each observation i, it calculates the dissimilarity between i and all the other points within the same cluster. This value is called average dissimilarity Di.

Then calculate the dissimilarity between i and all the other clusters and get the lowest value among them. Find the dissimilarity between i and the next closest cluster, called Ci.

Next find the silhouette width which is the difference between Ci and Di, divided by the maximum difference between Ci and Di. Si = (Ci - Di)/max(Di, Ci)

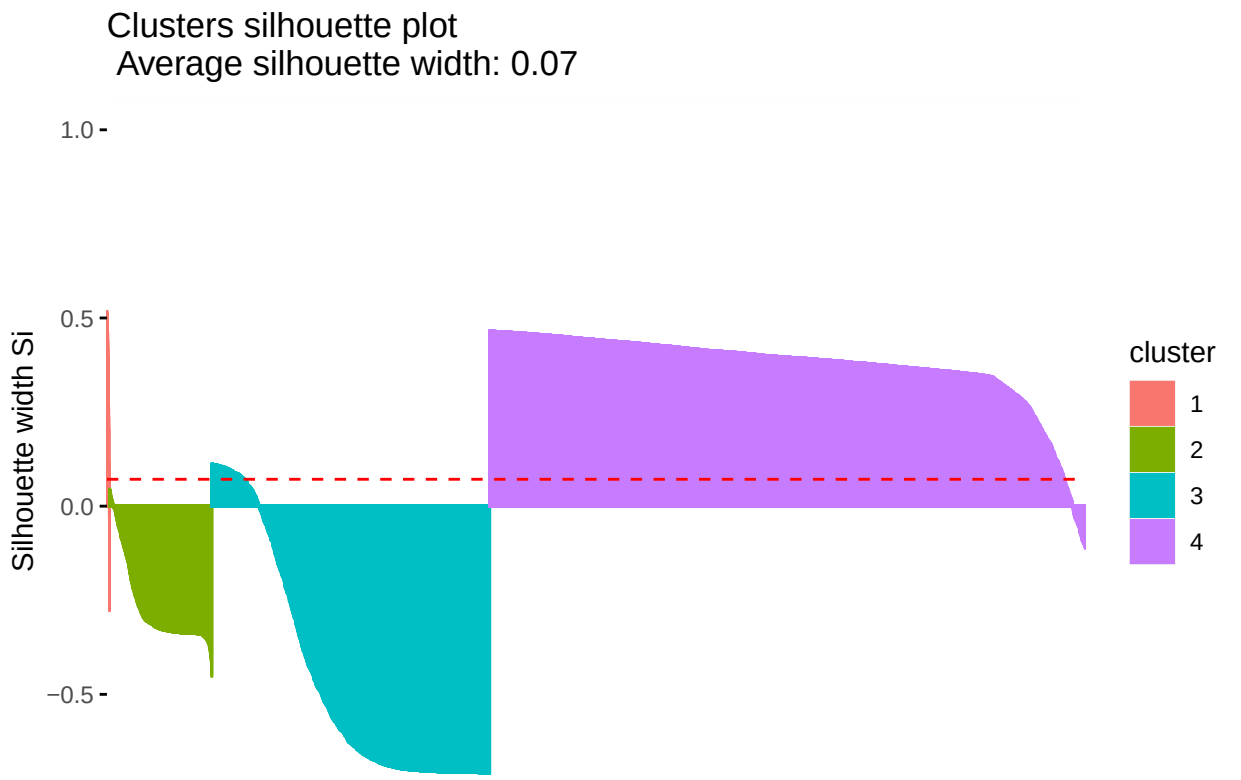
Si>0 means the observation is well clustered. The closer it is to 1 the better it is clustered.

Si <0 means the observation is wrongly clustered.

Si = 0 means the observation is between 2 clusters.

```
library(cluster)
library(factoextra)

sil <- silhouette(clustering_results_4$cluster, dist(data1[,(15:18)]))
fviz_silhouette(sil)
```

```
##   cluster size ave.sil.width
## 1       1   27          0.28
## 2       2 1044         -0.27
## 3       3 2838         -0.45
## 4       4 6085          0.37
```




From the silhouette width plot, it looks like cluster 4 is well clustered since the average silhouette width is 0.37 (a positive value). However, cluster 2 and 3 have negative average silhouette widths which may indicate some observations are wrongly clustered. Another observation is that the size of the 4 clusters are not very similar, with cluster 4 being the biggest (with 6085 observations) while cluster 1 only contains 27 observations. One of the assumptions of K-means clustering is that the size of the clusters should be similar, in order to calculate the boundaries of the clusters.

### Anomaly Detection Method 1 - using Local Outlier Factor Algorithm -Nearest neighbour method

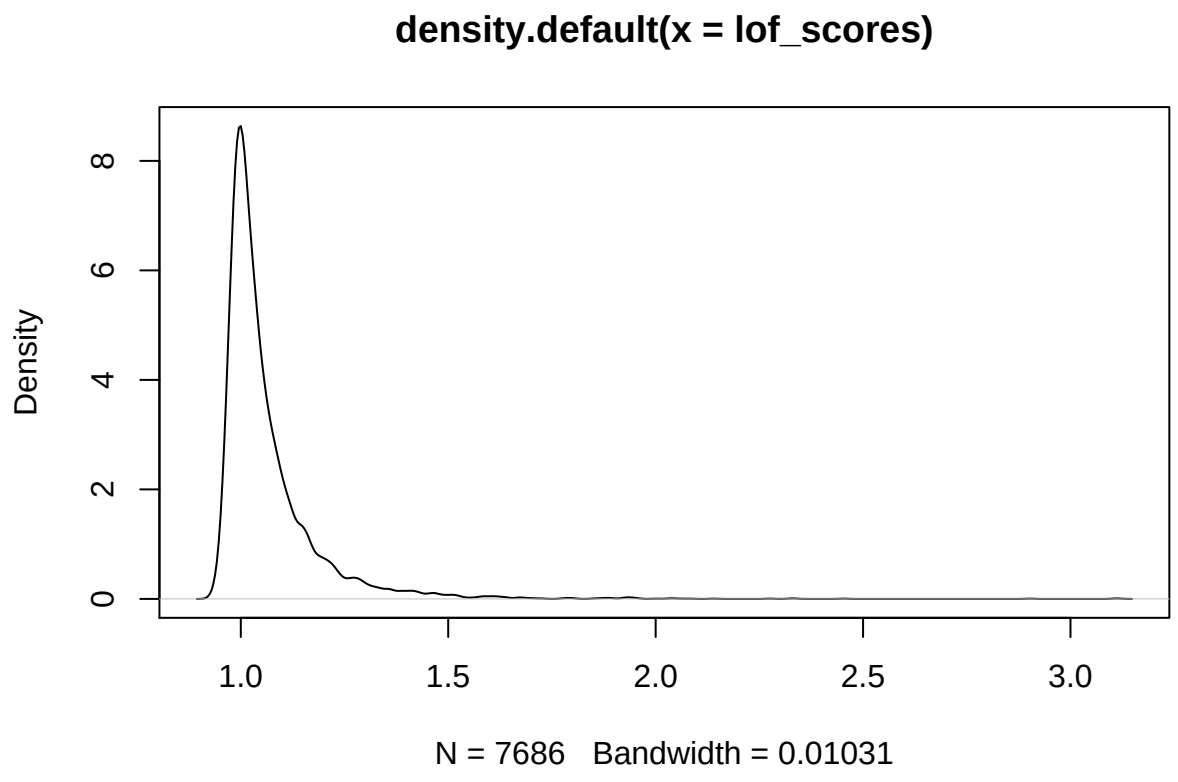
LOF uses density based methods to calculate degree of outlying. LOF is a unsupervised anomaly detection technique, every point in the dataset is assigned LOF score based on the threshold value it classify the data points as outlier or non-outlier.

Points which are much lower in density compared to its neighbours are considered outliers.

A LOF score of approximately 1 indicates that density around the point is comparable to its neighbors. Scores significantly larger than 1 indicate outliers (Hahsler).

If there are more than k duplicate points in the data, then lof can become NaN caused by an infinite local density. In this case we set lof to 1

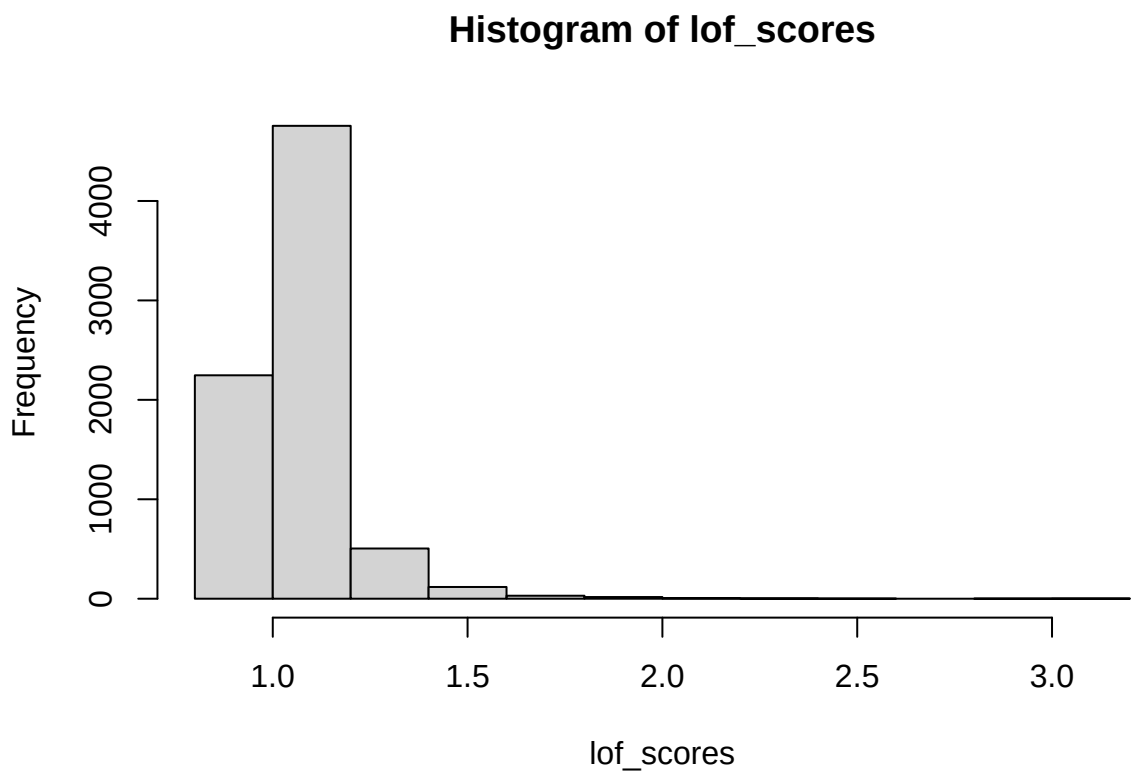
```
#remove duplicates rows
superstore_unq<-data1[!duplicated(data1[c("Sales","Profit","Quantity","Discount")]),]

#select numerical variables
superstore_lof<-superstore_unq[,c("Sales","Profit","Quantity","Discount")]

# for k=10, ,lof produces a vector of local outlier factor for each data point
lof_scores <- lofactor(superstore_lof, k=10)
plot(density(lof_scores))
```




```
# distribution of outlier factors
summary(lof_scores)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9250  0.9952  1.0267  1.0635  1.0870  3.1173
```

```r
hist(lof_scores, breaks=10)
```




**Manual Evaluation of Local Outlier Factor**

```r
#top 5 outliers transactons , the greater the lof_scores the farther the point is from the cluster. Cho
 lof_outliers <- order(lof_scores>2, decreasing=T)[1:13]
 superstore_unq[lof_outliers,]
```

```
##            Order_ID Order_Date  Ship_Date      Ship_Mode Customer_ID
## 28   US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520
## 166  CA-2014-139892 2014-09-08 2014-09-12 Standard Class    BM-11140
## 684  US-2017-168116 2017-11-04 2017-11-04       Same Day    GT-14635
## 2506 CA-2014-143917 2014-07-25 2014-07-27   Second Class    KL-16645
## 2698 CA-2014-145317 2014-03-18 2014-03-23 Standard Class    SM-20320
## 3012 CA-2017-134845 2017-04-17 2017-04-23 Standard Class    SR-20425
## 4992 US-2017-122714 2017-12-07 2017-12-13 Standard Class    HG-14965
## 5868 CA-2016-158211 2016-01-04 2016-01-08 Standard Class    BP-11185
## 6257 CA-2015-133977 2015-09-06 2015-09-08   Second Class    AT-10435
## 6827 CA-2016-118689 2016-10-02 2016-10-09 Standard Class    TC-20980
## 7773 CA-2016-108196 2016-11-25 2016-12-02 Standard Class    CS-12505
## 9410 US-2017-110149 2017-12-10 2017-12-13    First Class    WB-21850
## 9775 CA-2014-169019 2014-07-26 2014-07-30 Standard Class    LF-17185
##          Segment          City          State Postal_Code  Region
```

```
## 28      Consumer  Philadelphia   Pennsylvania       19140    East
## 166     Consumer   San Antonio          Texas       78207 Central
## 684    Corporate    Burlington North Carolina       27217   South
## 2506    Consumer San Francisco     California       94122    West
## 2698 Home Office  Jacksonville        Florida       32216   South
## 3012 Home Office    Louisville       Colorado       80027    West
## 4992   Corporate       Chicago       Illinois       60653 Central
## 5868   Corporate  Philadelphia   Pennsylvania       19143    East
## 6257 Home Office       Tamarac        Florida       33319   South
## 6827   Corporate     Lafayette        Indiana       47905 Central
## 7773    Consumer     Lancaster           Ohio       43130    East
## 9410    Consumer  Philadelphia   Pennsylvania       19143    East
## 9775    Consumer   San Antonio          Texas       78207 Central
##          Product_ID        Category Sub_Category
## 28   FUR-BO-10004834       Furniture    Bookcases
## 166  TEC-MA-10000822      Technology     Machines
## 684  TEC-MA-10004125      Technology     Machines
## 2506 OFF-SU-10000151 Office Supplies     Supplies
## 2698 TEC-MA-10002412      Technology     Machines
## 3012 TEC-MA-10000822      Technology     Machines
## 4992 OFF-BI-10001120 Office Supplies      Binders
## 5868 OFF-AR-10004078 Office Supplies          Art
## 6257 OFF-BI-10003166 Office Supplies      Binders
## 6827 TEC-CO-10004722      Technology      Copiers
## 7773 TEC-MA-10000418      Technology     Machines
## 9410 OFF-BI-10000014 Office Supplies      Binders
## 9775 OFF-BI-10004995 Office Supplies      Binders
##                                                    Product_Name     Sales
## 28   Riverside Palais Royal Lawyers Bookcase, Royale Cherry Finish  3083.430
## 166                     Lexmark MX611dhe Monochrome Laser Printer  8159.952
## 684                     Cubify CubeX 3D Printer Triple Head Print  7999.980
## 2506                  High Speed Automatic Electric Letter Opener  8187.650
## 2698         Cisco TelePresence System EX90 Videoconferencing Unit 22638.480
## 3012                    Lexmark MX611dhe Monochrome Laser Printer  2549.985
## 4992                         Ibico EPK-21 Electric Binding System  1889.990
## 5868                                                   Newell 312     4.672
## 6257                              GBC Plasticlear Binding Covers     3.444
## 6827                        Canon imageCLASS 2200 Advanced Copier 17499.950
## 7773                    Cubify CubeX 3D Printer Double Head Print  4499.985
## 9410                                      Heavy-Duty E-Z-D Binders     3.273
## 9775                    GBC DocuBind P400 Electric Binding System  2177.584
##      Quantity Discount     Profit diff_in_days
## 28          7      0.5 -1665.0522            4
## 166         8      0.4 -1359.9920            4
## 684         4      0.5 -3839.9904            0
## 2506        5      0.0   327.5060            2
## 2698        6      0.5 -1811.0784            5
## 3012        5      0.7 -3399.9800            6
## 4992        5      0.8 -2929.4845            6
## 5868        1      0.2     0.5840            4
## 6257        1      0.7    -2.5256            2
## 6827        5      0.0  8399.9760            7
## 7773        5      0.7 -6599.9780            7
## 9410        1      0.7    -2.5093            3
```

```
## 9775        8      0.8 -3701.8928            4
```

Here we are filtering for outliers which have LOF scores greater than 2.

```
#outliers for z-score >2(how far an observation is from the mean)

outlier_orderid<-superstore_unq[which(lof_scores >2),1]
vec<-as.vector(outlier_orderid)


#new column for outlier status
superstore_unq<-mutate(superstore_unq,outlier_status=ifelse(Order_ID %in% (vec) ,"Yes","No"))

x<-subset(superstore_unq,superstore_unq$outlier_status=="Yes")
head(x, 10)          #look at top 10 rows of x
```

```
##           Order_ID Order_Date  Ship_Date      Ship_Mode Customer_ID  Segment
## 28  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 29  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 30  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 31  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 32  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 33  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 34  US-2015-150630 2015-09-17 2015-09-21 Standard Class    TB-21520 Consumer
## 165 CA-2014-139892 2014-09-08 2014-09-12 Standard Class    BM-11140 Consumer
## 166 CA-2014-139892 2014-09-08 2014-09-12 Standard Class    BM-11140 Consumer
## 167 CA-2014-139892 2014-09-08 2014-09-12 Standard Class    BM-11140 Consumer
##             City        State Postal_Code  Region      Product_ID
## 28  Philadelphia Pennsylvania       19140    East FUR-BO-10004834
## 29  Philadelphia Pennsylvania       19140    East OFF-BI-10000474
## 30  Philadelphia Pennsylvania       19140    East FUR-FU-10004848
## 31  Philadelphia Pennsylvania       19140    East OFF-EN-10001509
## 32  Philadelphia Pennsylvania       19140    East OFF-AR-10004042
## 33  Philadelphia Pennsylvania       19140    East OFF-BI-10001525
## 34  Philadelphia Pennsylvania       19140    East OFF-AR-10001683
## 165  San Antonio        Texas       78207 Central OFF-AR-10004441
## 166  San Antonio        Texas       78207 Central TEC-MA-10000822
## 167  San Antonio        Texas       78207 Central OFF-ST-10000991
##            Category Sub_Category
## 28        Furniture    Bookcases
## 29  Office Supplies      Binders
## 30        Furniture  Furnishings
## 31  Office Supplies    Envelopes
## 32  Office Supplies          Art
## 33  Office Supplies      Binders
## 34  Office Supplies          Art
## 165 Office Supplies          Art
## 166      Technology     Machines
## 167 Office Supplies      Storage
##                                                         Product_Name
## 28            Riverside Palais Royal Lawyers Bookcase, Royale Cherry Finish
## 29                     Avery Recycled Flexi-View Covers for Binding Systems
## 30           Howard Miller 13-3/4" Diameter Brushed Chrome Round Wall Clock
```

```
## 31                                                  Poly String Tie Envelopes
## 32            BOSTON Model 1800 Electric Pencil Sharpeners, Putty/Woodgrain
## 33  Acco Pressboard Covers with Storage Hooks, 14 7/8" x 11", Executive Red
## 34                                                            Lumber Crayons
## 165                                          BIC Brite Liner Highlighters
## 166                               Lexmark MX611dhe Monochrome Laser Printer
## 167                          Space Solutions HD Industrial Steel Shelving.
##        Sales Quantity Discount     Profit diff_in_days outlier_status
## 28  3083.430        7      0.5 -1665.0522            4            Yes
## 29     9.618        2      0.7    -7.0532            4            Yes
## 30   124.200        3      0.2    15.5250            4            Yes
## 31     3.264        2      0.2     1.1016            4            Yes
## 32    86.304        6      0.2     9.7092            4            Yes
## 33     6.858        6      0.7    -5.7150            4            Yes
## 34    15.760        2      0.2     3.5460            4            Yes
## 165    9.936        3      0.2     2.7324            4            Yes
## 166 8159.952        8      0.4 -1359.9920            4            Yes
## 167  275.928        3      0.2   -58.6347            4            Yes
```

### Plot LOF outliers

```
pch <- rep(".", 7000)
pch[lof_outliers] <- "+"
col <- rep("black",7000)
col[lof_outliers] <- "red"
pairs(superstore_lof, pch=pch, col=col)
```

### Outlier Detection Method 2 - Random Forest Algorithm

outForest is a random forest based implementation of the method. Each numeric variable is regressed onto all other variables using a random forest. If the scaled absolute difference between observed value and out-of-bag prediction is suspiciously large (e.g. more than three times the RMSE of the out-of-bag predictions), then a value is considered an outlier. After identification of outliers, they can be replaced e.g. by predictive mean matching from the non-outliers. outForest package estimates this conditional mean by a random forest. RF algorithm works well without parameter tuning, outliers in the input variables are no issue and the out-of-bag mechanism helps to provide fair outlier scores.

```
# add day columns
superstore$orderday<-yday(superstore$`Order Date`)  #yday functions pulls out day of the year informati
superstore$shipday<-yday(superstore$`Ship Date`)

#add year column
superstore$orderyear<-year(superstore$`Order Date`)  #year function pulls out year from Order Date colu

#remove duplicates rows
superstore_unq<-superstore[!duplicated(superstore[c("Sales","Profit","Quantity","Discount","orderday","s
```

```
#dataframe for outforest
superstore_out<-superstore_unq[,c("Sales","Profit","Quantity","Discount","orderday","shipday")]
```

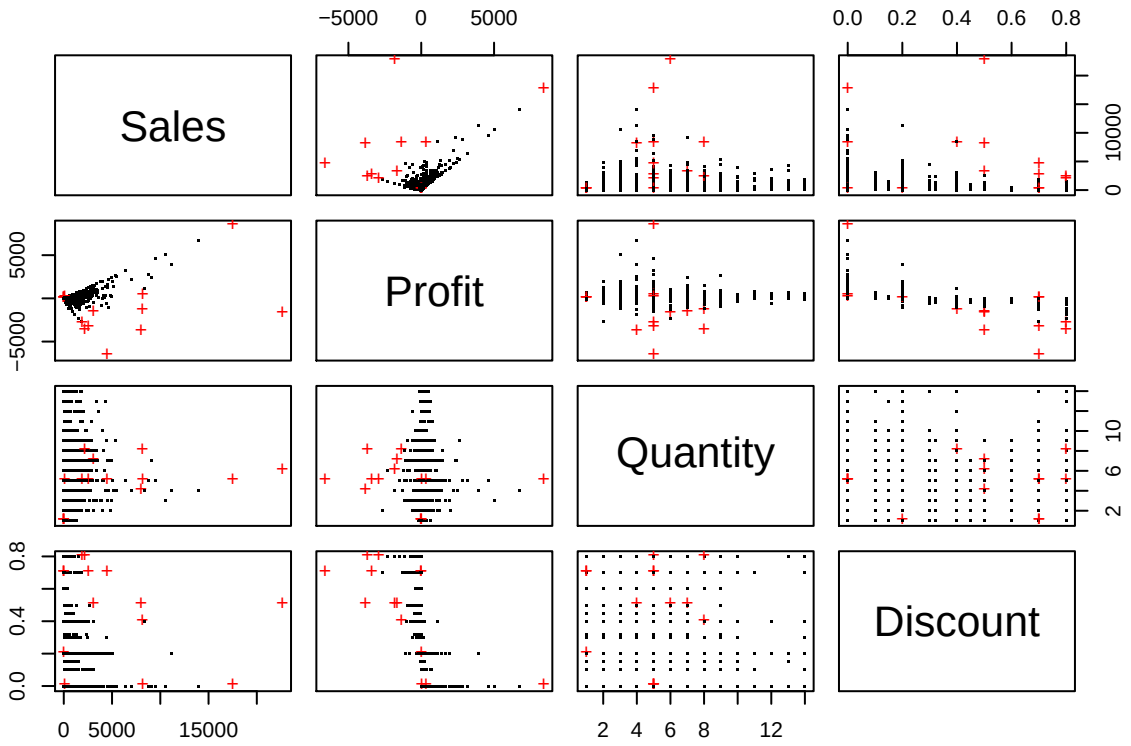



Figure 23: Plot of outliers found using Local Outlier Factor algorithm. Outliers are represented as red + signs.

```r
superstore_out<-scale(superstore_out)

#fit outforest
outforest_outlier<-outForest(superstore_out, replace = "NA" )
```

```
##
## Outlier identification by random forests
##
##   Variables to check:        Sales, Profit, Quantity, Discount, orderday, shipday
##   Variables used to check:   Sales, Profit, Quantity, Discount, orderday, shipday
##
##   Checking: Sales  Profit  Quantity  Discount  orderday  shipday
```

```r
print(outforest_outlier)
```

```
## I am an object of class(es) outForest and list
##
## The following number of outliers have been identified:
##
##          Number of outliers
## Sales                    74
## Profit                   83
## Quantity                123
## Discount                 44
## orderday                102
## shipday                 130
```

### Manual Evaluation

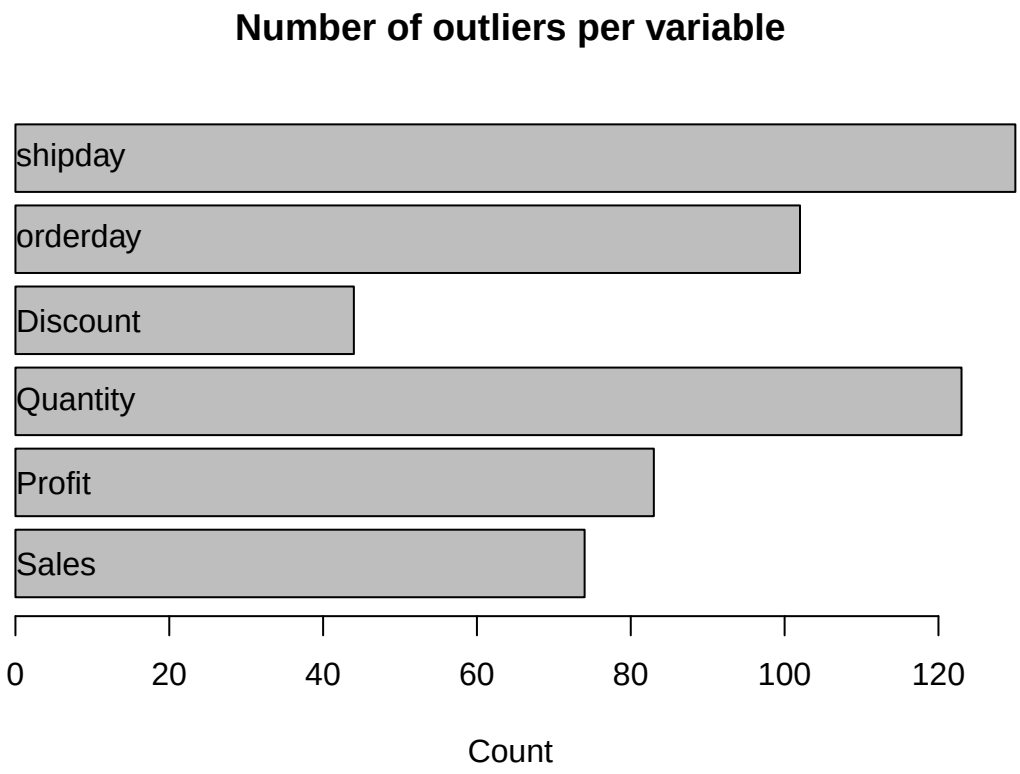
Observed same set of top outliers as LOF algorithm.

```r
#outlier rows, observed values, predicted and RMSE
y<-outliers(outforest_outlier)
row_out<-y$row
superstore_unq<-mutate(superstore_unq,outlier_outstatus=ifelse(row.names(superstore_unq) %in%row_out ,"

head(outliers(outforest_outlier),21)
```

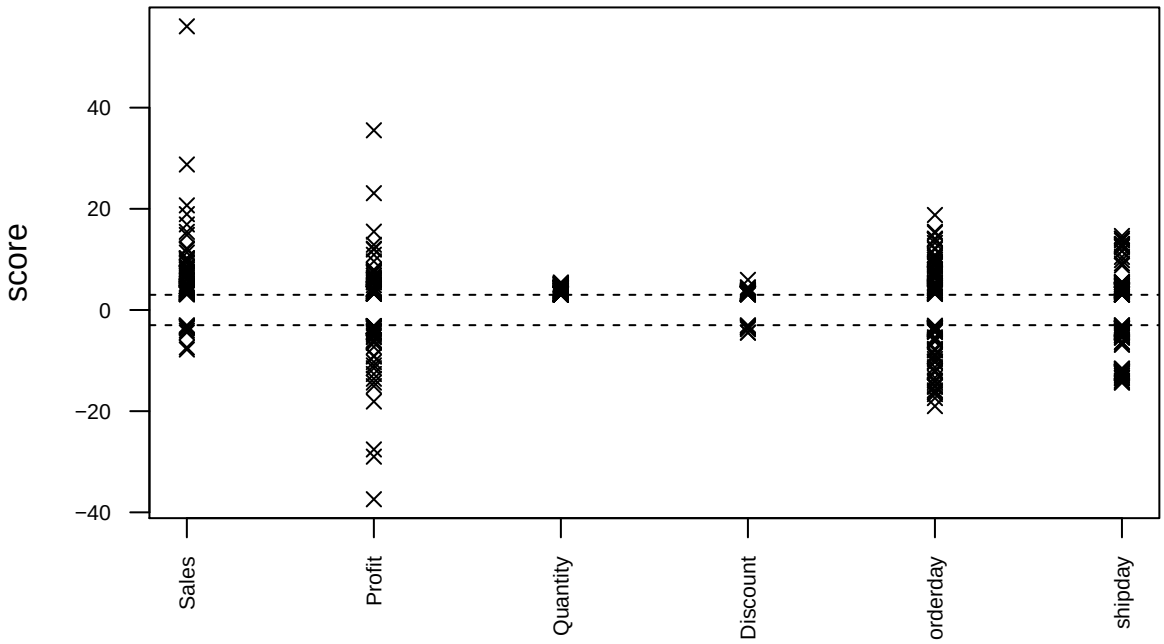
```
##      row      col   observed  predicted      rmse     score threshold
## 21  2698    Sales  35.949718  3.3566430 0.5813180  56.06755         3
## 140 7770   Profit -28.291897 -4.8619280 0.6263437 -37.40753         3
## 136 6824   Profit  35.729828 13.4837288 0.6263437  35.51740         3
## 99  2698   Profit  -7.852260 10.3056345 0.6263437 -28.99030         3
## 47  6824    Sales  27.706060 10.9879235 0.5813180  28.75902         3
## 86   684   Profit -16.511916  0.7521809 0.6263437 -27.56330         3
## 143 8151   Profit  28.559394 14.1000197 0.6263437  23.08537         3
## 56  8151    Sales  22.091085 10.0713514 0.5813180  20.67669         3
## 385 6680 orderday  -2.204847  1.0591869 0.1717897 -19.00017         3
## 18  2506    Sales  12.766493  1.7539361 0.5813180  18.94412         3
## 395 7676 orderday   1.425622 -1.8006508 0.1717897  18.78036         3
## 157 9772   Profit -15.922498 -4.5913575 0.6263437 -18.09093         3
```

```
## 353 2487 orderday  -2.194874  0.7961707 0.1717897 -17.41108         3
## 2    166    Sales  12.722058  2.8874314 0.5813180  16.91781         3
## 416 8361 orderday  -2.194874  0.6612927 0.1717897 -16.62594         3
## 355 2489 orderday  -2.194874  0.5968649 0.1717897 -16.25091         3
## 413 8358 orderday  -2.194874  0.5344678 0.1717897 -15.88769         3
## 112 4189   Profit  21.388959 11.6846075 0.6263437  15.49365         3
## 397 7678 orderday   1.425622 -1.2322866 0.1717897  15.47187         3
## 43  6423    Sales  13.107124  4.1286325 0.5813180  15.44506         3
## 394 7675 orderday   1.425622 -1.1869903 0.1717897  15.20820         3
##     replacement
## 21           NA
## 140          NA
## 136          NA
## 99           NA
## 47           NA
## 86           NA
## 143          NA
## 56           NA
## 385          NA
## 18           NA
## 395          NA
## 157          NA
## 353          NA
## 2            NA
## 416          NA
## 355          NA
## 413          NA
## 112          NA
## 397          NA
## 43           NA
## 394          NA
```

```
# Outliers per variable
plot(outforest_outlier)
```




```
# Basic plot of the scores of the outliers per variable
plot(outforest_outlier,what="scores")
```



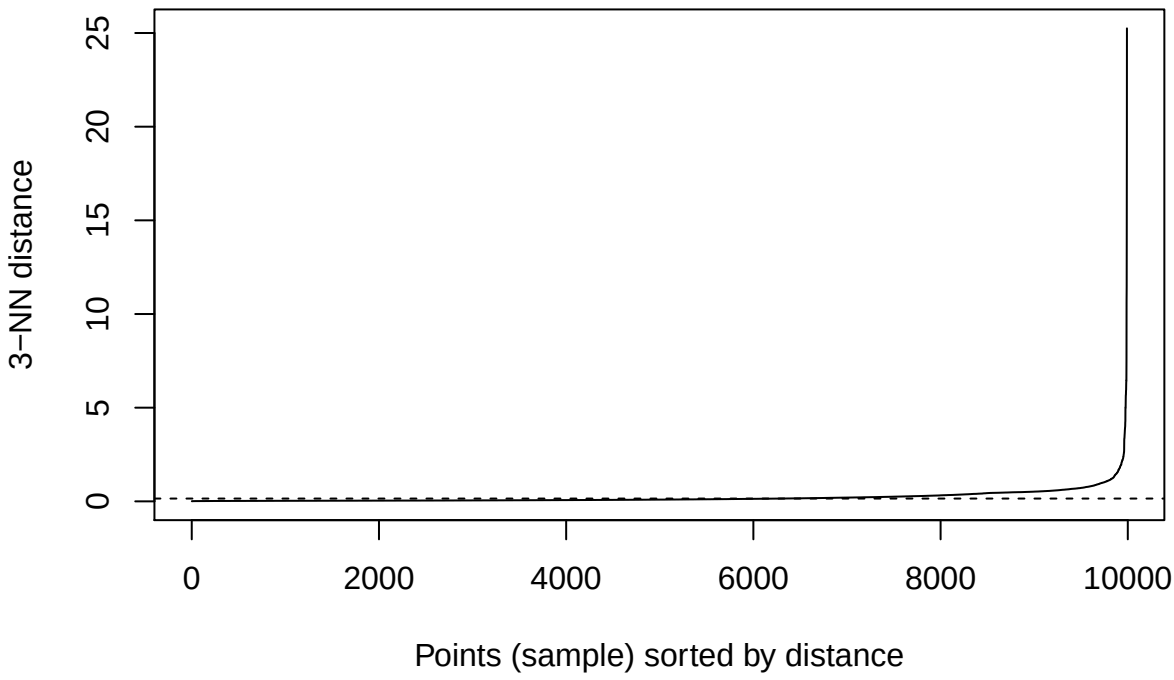

## Density Based Clustering Algorithm

DBSCAN is density-based clustering algorithm, which can be used to identify clusters of any shape in data set containing noise and outliers. DBSCAN stands for Density-Based Spatial Clustering and Application with Noise. DBSCAN does not require the user to specify the number of clusters to be generated and can identify outliers.

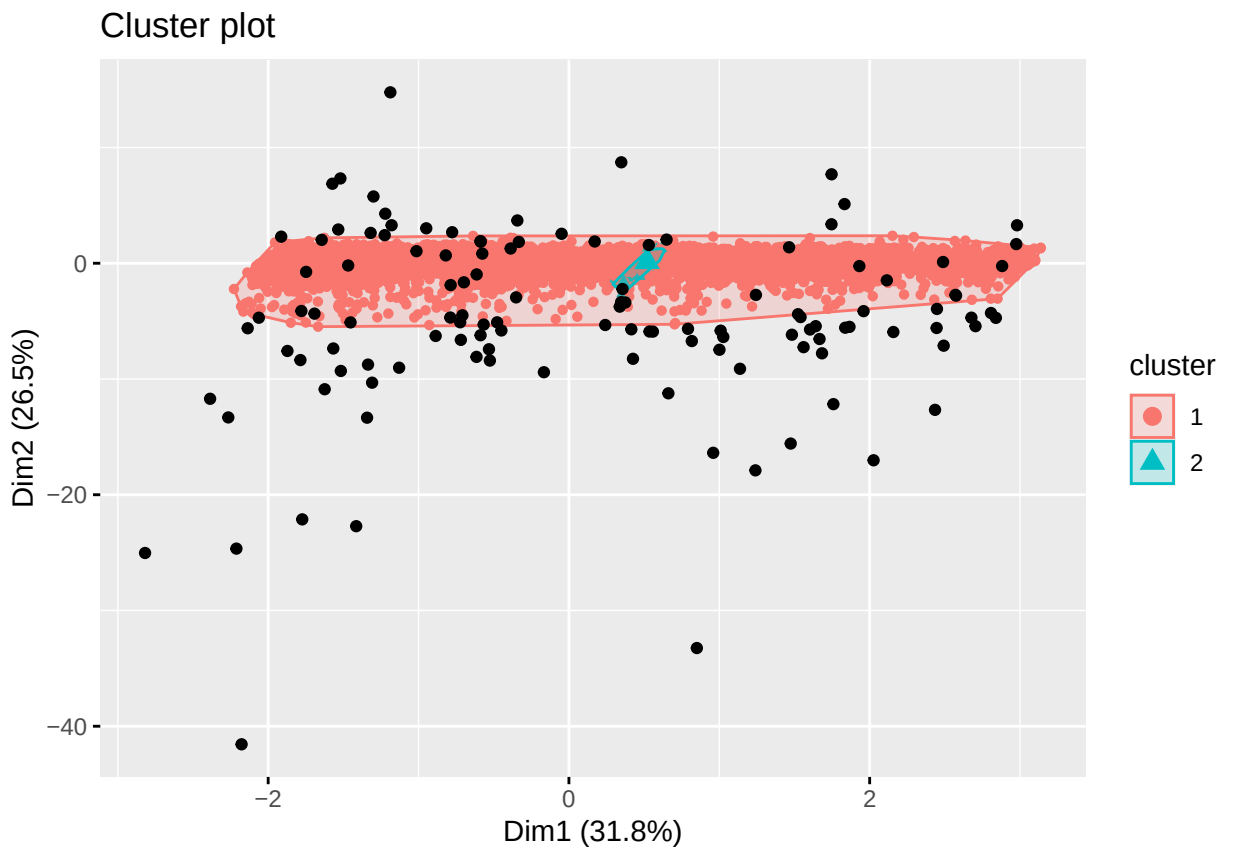
```
#dataset for dbscan, using sales and profit
superstore_dbc<-superstore_unq[,c("Sales","Profit","Quantity","Discount","orderday","shipday")]
superstore_dbc<-scale(superstore_dbc)
```

K-distance plot -find maximum reachability distance (eps)

```
#find value of eps
dbscan::kNNdistplot(superstore_dbc, k = 3)
abline(h = 0.15, lty = 2)
```




```
# cluster assignments with dbscan
dbc<-fpc::dbscan(superstore_dbc,eps=1.5, MinPts  = 10)
fviz_cluster(dbc, superstore_dbc,  geom = "point")
```




```
#hullplot(superstore_dbc, dbc$cluster)
```

The black points are outliers .

```
print(dbc)
```

```
## dbscan Pts=9991 MinPts=10 eps=1.5
##          0    1   2
## border 121   67   5
## seed     0 9657 141
## total  121 9724 146
```

```
#status<-cluster.stats(dist(superstore_dbc), dbc$cluster)
superstore_unq<-mutate(superstore_unq,outlier_dbcstatus=ifelse(dbc$cluster==0 ,"Yes","No"))
head(superstore_unq)
```

```
## # A tibble: 6 x 26
##   `Row ID` `Order ID` `Order Date`        `Ship Date`         `Ship Mode`
##      <dbl> <chr>      <dttm>              <dttm>              <chr>
## 1        1 CA-2016-1~ 2016-11-08 00:00:00 2016-11-11 00:00:00 Second Cla~
## 2        2 CA-2016-1~ 2016-11-08 00:00:00 2016-11-11 00:00:00 Second Cla~
## 3        3 CA-2016-1~ 2016-06-12 00:00:00 2016-06-16 00:00:00 Second Cla~
## 4        4 US-2015-1~ 2015-10-11 00:00:00 2015-10-18 00:00:00 Standard C~
## 5        5 US-2015-1~ 2015-10-11 00:00:00 2015-10-18 00:00:00 Standard C~
```

```
## 6        6 CA-2014-1~ 2014-06-09 00:00:00 2014-06-14 00:00:00 Standard C~
## # ... with 21 more variables: `Customer ID` <chr>, `Customer Name` <chr>,
## #   Segment <chr>, Country <chr>, City <chr>, State <chr>, `Postal Code` <dbl>,
## #   Region <chr>, `Product ID` <chr>, Category <chr>, `Sub-Category` <chr>,
## #   `Product Name` <chr>, Sales <dbl>, Quantity <dbl>, Discount <dbl>,
## #   Profit <dbl>, orderday <dbl>, shipday <dbl>, orderyear <dbl>,
## #   outlier_outstatus <chr>, outlier_dbcstatus <chr>
```

### Re-running K-means after removing outliers identified with the Local Outlier Factor

From here we can keep just the normal transactions of the superstore dataset (which are not outliers):

```
normal_transactions <-subset(superstore_unq,superstore_unq$outlier_outstatus=="No")
```

**Data Preparation** Extract just the numeric columns from the normal transactions:

```
normal_transactions_numeric <- normal_transactions %>%
                        select(Sales, Quantity, Discount, Profit)
```

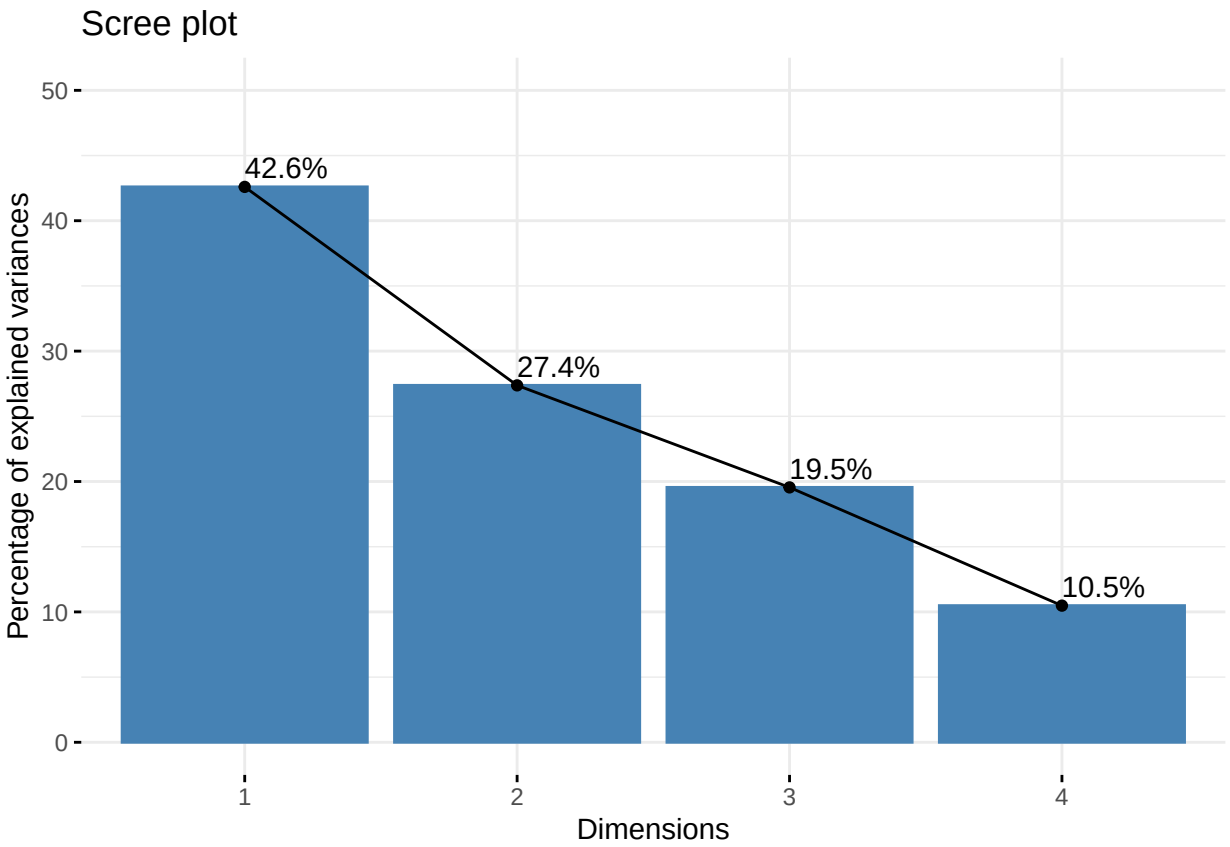
First scale the data since the variables are measured in different scales:

```
normal_transactions_scaled <- scale(normal_transactions_numeric)

rownames(normal_transactions_scaled) <- normal_transactions_numeric$name     #re-assign the rownames
```

```
## Warning: Unknown or uninitialised column: `name`.
```

Use PCA to scale the dimensions of the data:

```
res.pca <- PCA(normal_transactions_scaled, graph = FALSE)

#Visualize eigenvalues/variances
fviz_screeplot(res.pca, addlabels = TRUE, ylim = c(0, 50))
```



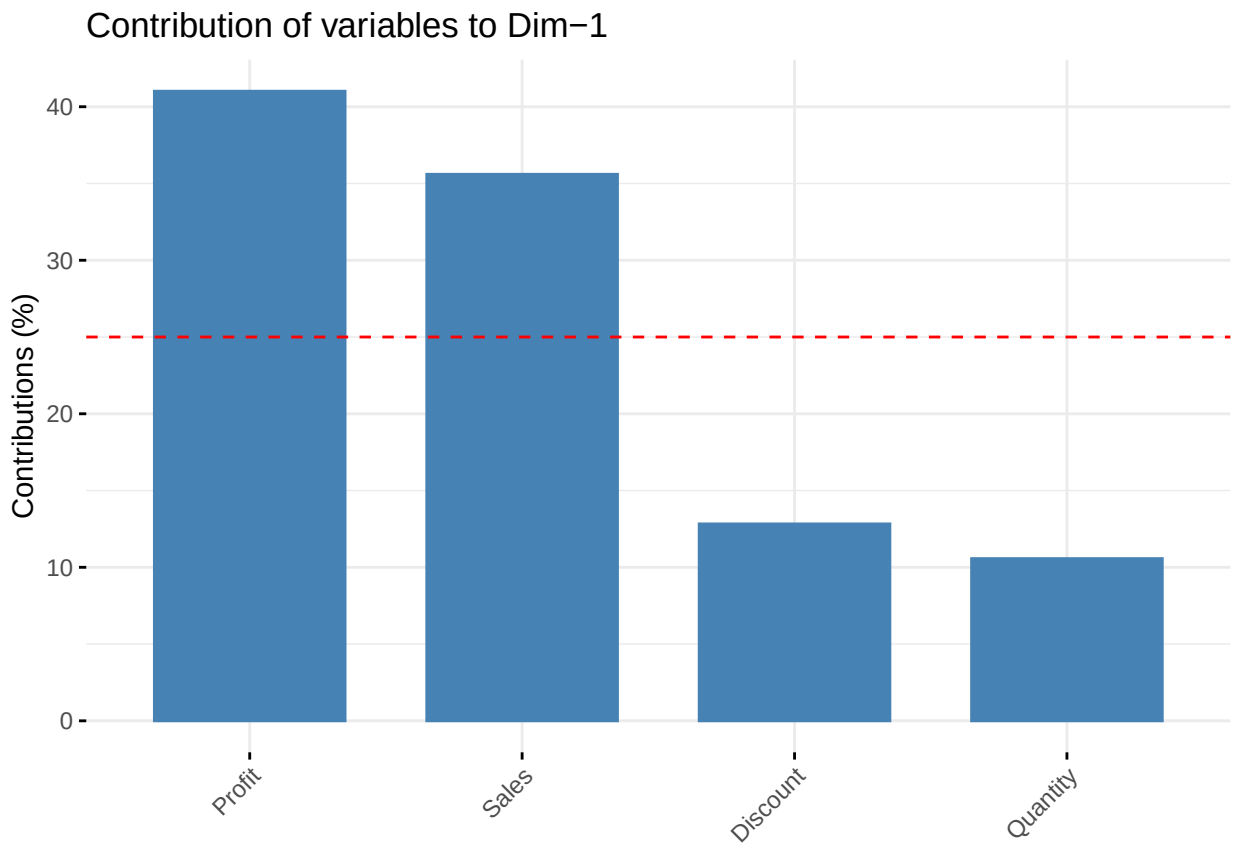

Scree plot shows dimension 1 captures 44.8% of the variance while dimension 2 captures 25.4% of the variance.

Extract the results for variables

```
var <- get_pca_var(res.pca)
```

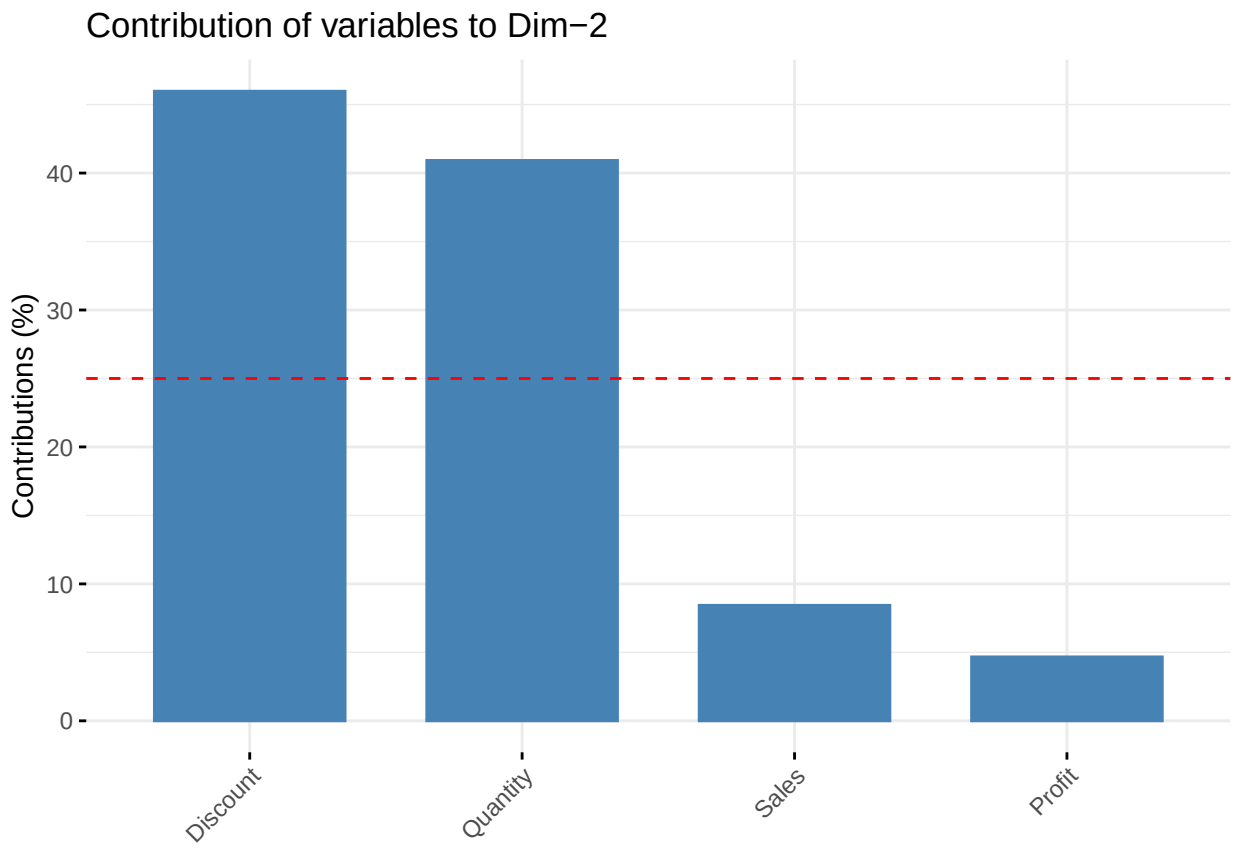
Contributions of variables to PC1

```
fviz_contrib(res.pca, choice = "var", axis = 1, top = 10)
```



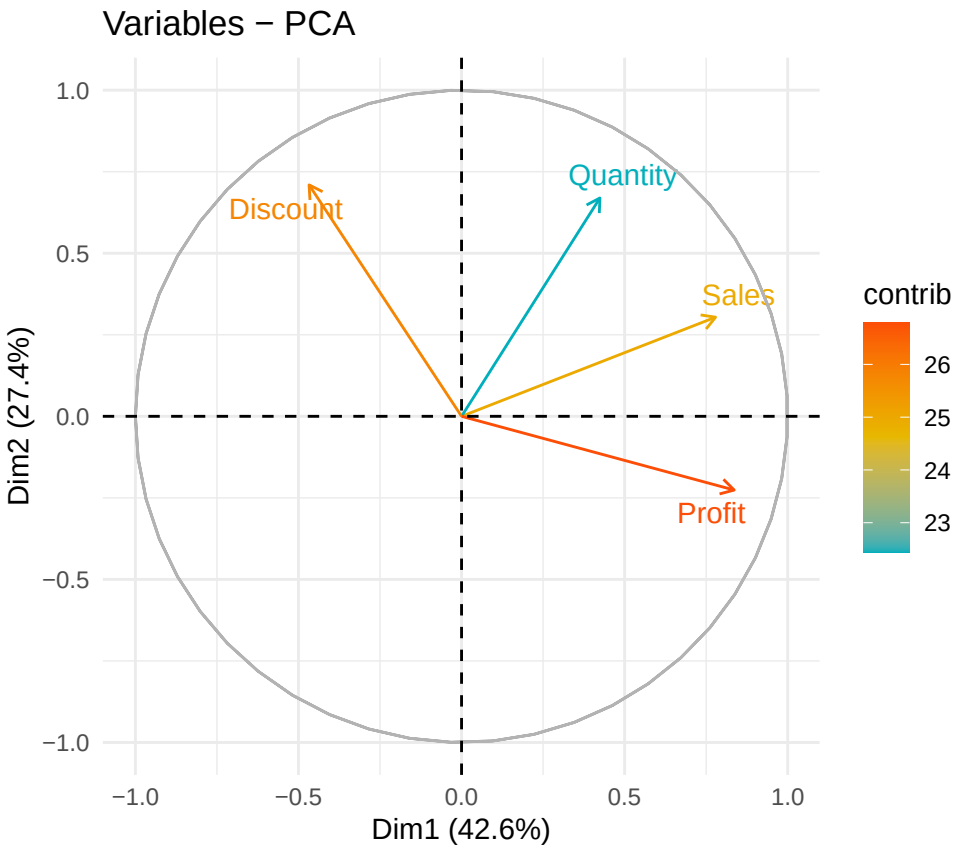

Contributions of variables to PC2

```
fviz_contrib(res.pca, choice = "var", axes = 2, top = 10)
```




#Control variable colours using their contributions to the principal axis

```
fviz_pca_var(res.pca, col.var = "contrib",
             gradient.cols =c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE #prevent text overlapping
             ) + theme_minimal() + ggtitle("Variables - PCA")
```




From the above visualization, sales and profit tend to increase together. Quantity also increases as sales and profits increase. Discount is inversely related to sales, profit and quantity.

## Determining the Optimal Number of Clusters for Kmeans Clustering

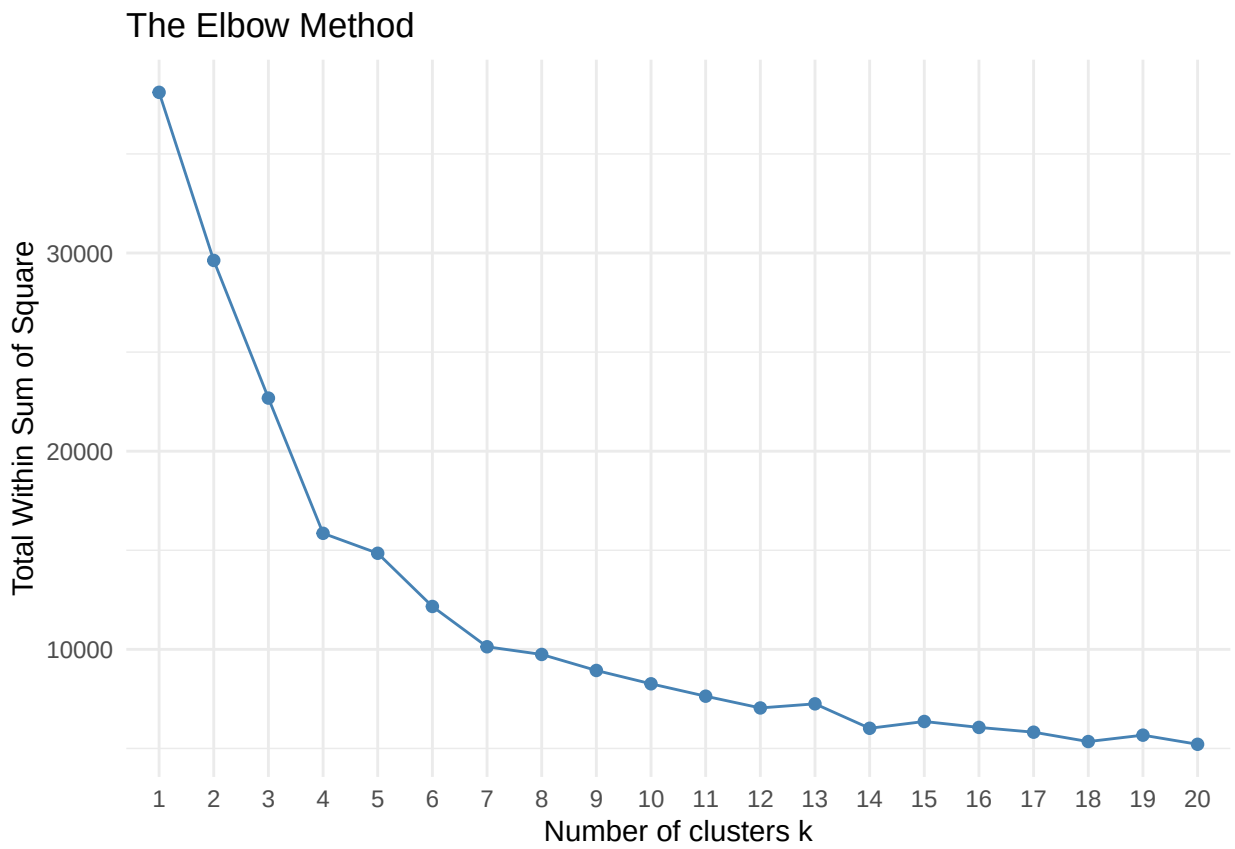
### Elbow Method

Using the Elbow method to determine the optimal number of clusters involves plotting the within cluster sum of squares. The optimal number of clusters is the one located at the "bend" on the graph, where WSS is lowered. Within-cluster sum of squares (WCSS) is the degree to which items within a cluster are similar or dissimilar (Nwanganga and Chapple, 2020).

The closer the items within a cluster are to the centroid, the lower the value of WCSS.

```
set.seed(123)

#to compute the total within-cluster sum of squares:
fviz_nbclust(normal_transactions_scaled, kmeans, method = "wss", k.max = 20) +
  theme_minimal() + ggtitle("The Elbow Method")
```

```
## Warning: did not converge in 10 iterations
```



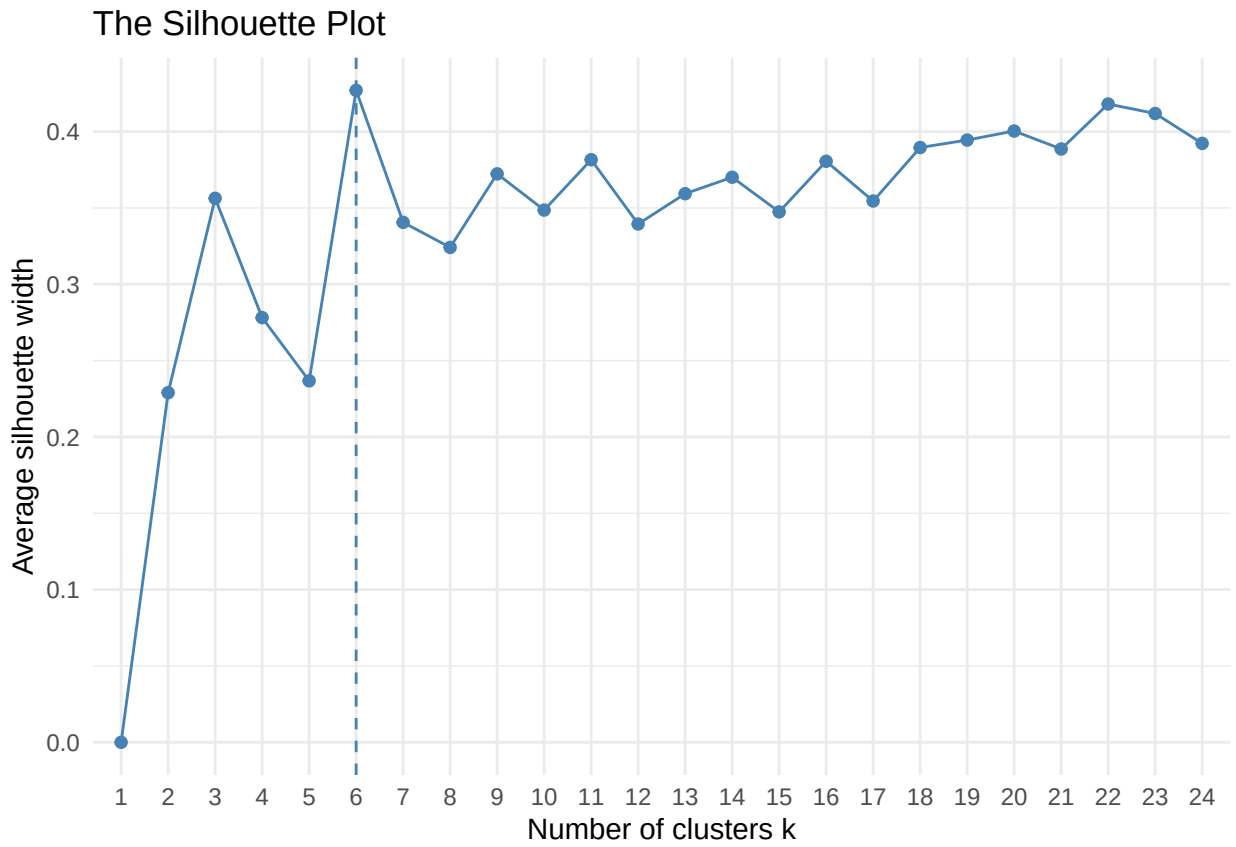

The optimal number of clusters looks to be k = 4 (the elbow point). K=4 is where the curve bends, indicating the point at which increasing the number of clusters (k) does not significantly decrease the total within cluster sum of square value.

### Silhouette Method

Another method is to look at the silhouette method. The silhouette of an observation is a value that represents how closely the item is matched with other items with the the same cluster and how loosely matched it is with observations in a different cluster.

```
fviz_nbclust(normal_transactions_scaled, kmeans, method = "silhouette", k.max = 24) + theme_minimal() +
```

```
## Warning: did not converge in 10 iterations
```




The silhouette method suggests the optimal number of clusters is 2, since k=2 corresponds to the highest average silhouette width value.

### Gap Statistic Method

A third statistical approach called the gap statistic method, compares the difference between clusters created from observed data and clusters created from a randomly generated dataset called the reference dataset. The gap statistic for a given k value is the difference in the total WCSS for the observed data and that of the reference dataset. The optimal number of clusters is denoted by the k value that yields the largest gap statistic.

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = normal_transactions_scaled, FUNcluster = kmeans,     K.max = 10, B = 50, nstart = 25)
## B=50 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
##  --> Number of clusters (method 'firstmax'): 1
##           logW    E.logW      gap      SE.sim
##  [1,] 8.601138 10.176695 1.575557 0.002853784
##  [2,] 8.490489  9.909276 1.418788 0.002372051
##  [3,] 8.326993  9.819997 1.493005 0.002311546
##  [4,] 8.146029  9.736027 1.589998 0.002592209
##  [5,] 8.087892  9.673130 1.585238 0.002343395
##  [6,] 8.049923  9.624168 1.574244 0.002352222
##  [7,] 7.919249  9.590916 1.671667 0.001975810
##  [8,] 7.876153  9.559126 1.682974 0.001935427
##  [9,] 7.809601  9.528528 1.718927 0.002035944
## [10,] 7.750613  9.502109 1.751496 0.001911606
```

Visualize the results of the gap statistic method.

```
fviz_gap_stat(gap_stat)
```

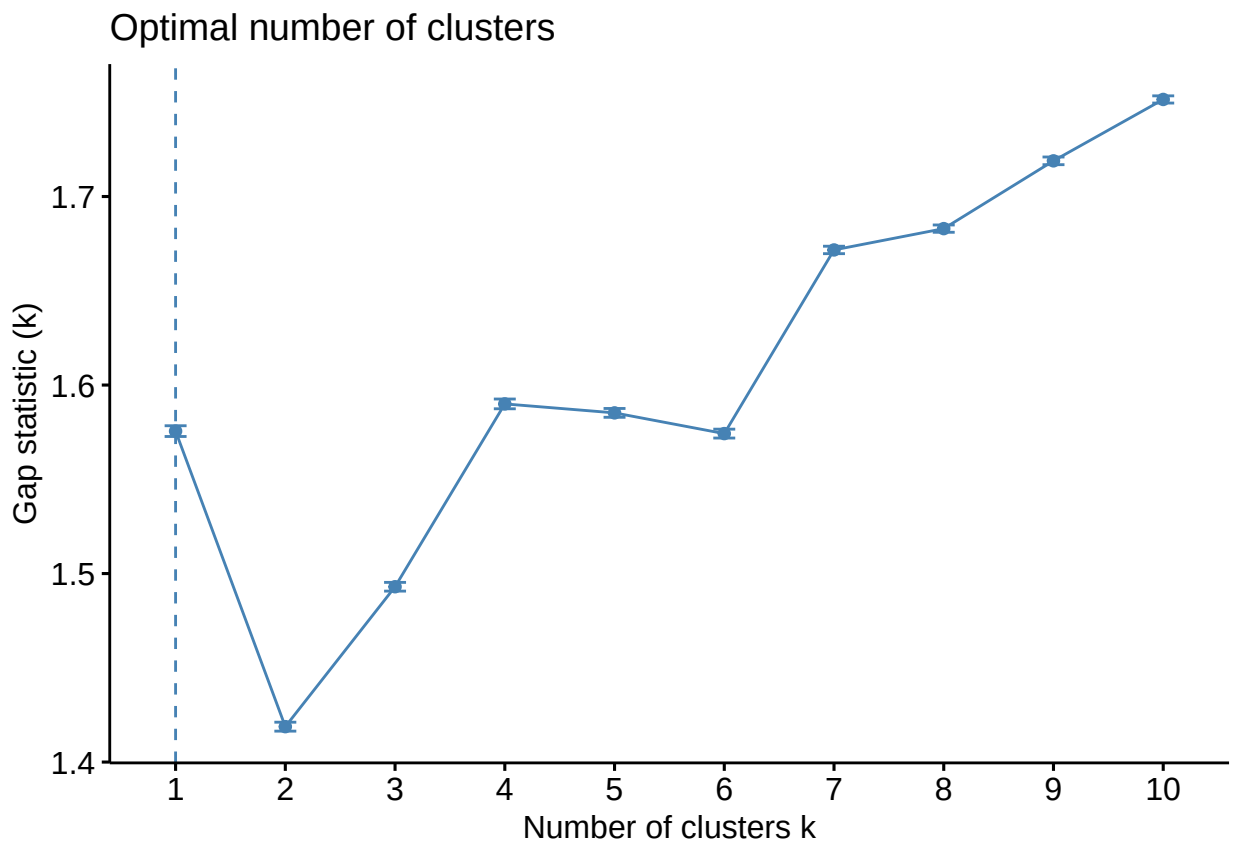
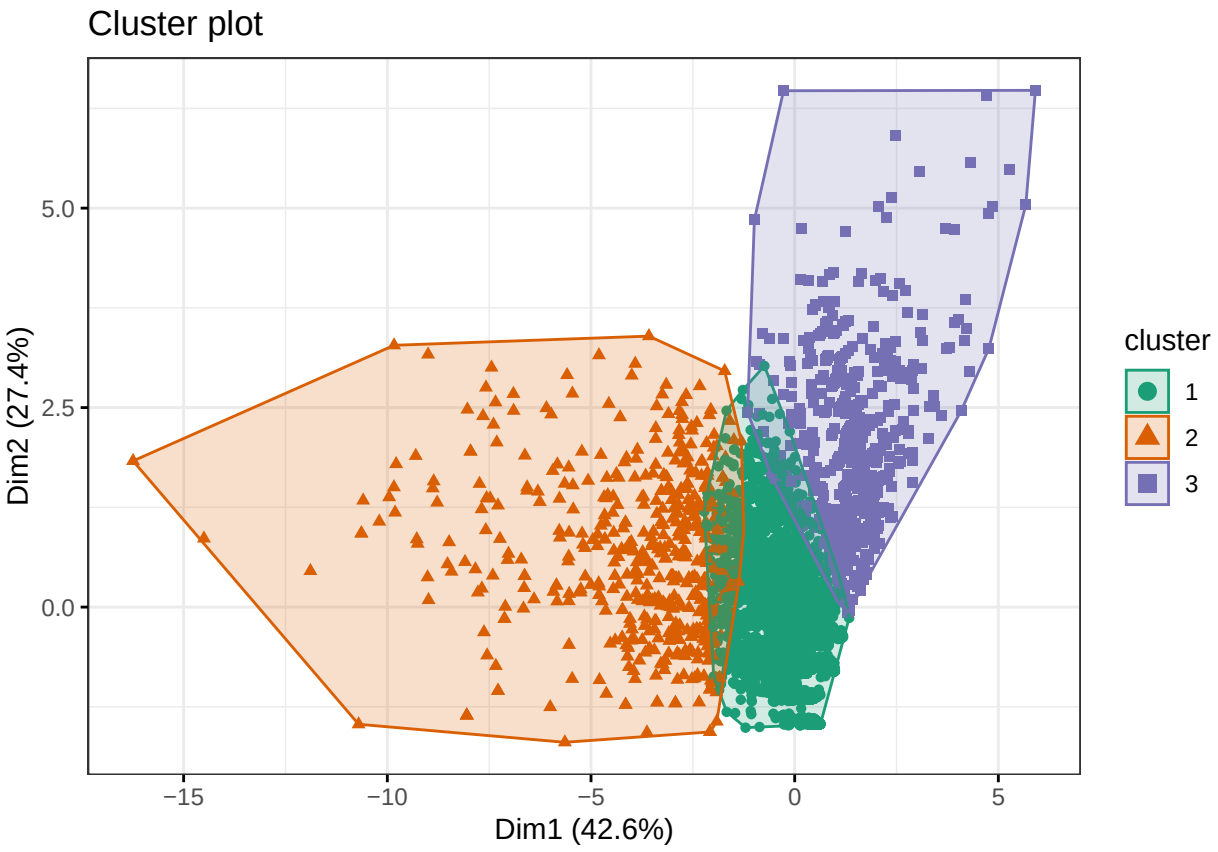



Figure 24: Finding the optimal number of clusters for k-medoids using the gap statistic method

According to the gap statistic method, the optimal number of clusters is 1 which does not make sense as all observations will be in a single cluster. The next optimal k-value is 6.

From the results of the silhouette method, elbow method and gap statistic method, the optimal value of k=4 is the most common result (from both the silhouette method and the elbow method). So we tried both k=3 and k=4 for kmeans clustering.

```
library(RColorBrewer)
fviz_cluster(results_kmeans, data = normal_transactions_scaled,
             palette = brewer.pal(n = 8, name = "Dark2"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```




To see the kmeans results:

```
#results_kmeans

#the clustering vector information has been omitted for brevity.
```

K-means clustering with 3 clusters of sizes 131, 6484, 1022

Cluster means: Sales Quantity Discount Profit 1 5.29934595 1.04564847 -0.4747453 4.5131872279 2 -0.08504992 -0.01836038 -0.3350941 0.0001464245 3 -0.13967774 -0.01754527 2.1868313 -0.5794294944

[1] 4677.307 10427.565 2891.185 (between_SS / total_SS = 41.1 %)

Available components:

[1] “cluster” “centers” “totss” “withinss” “tot.withinss” “betweenss” “size” “iter”
[9] “ifault”

The within cluster sum of squares by cluster is 41.1% for k = 3.

try clustering with kmeans using k = 4

K-means clustering with 4 clusters of sizes 1741, 4917, 941, 38

Cluster means: Sales Quantity Discount Profit 1 0.4660012 1.36028979 -0.3218513 0.2382839 2 -0.1953987 -0.47692843 -0.3185796 -0.0499454 3 -0.1892876 -0.04414503 2.2823862 -0.5695165 4 8.6207116 0.48244836 -0.5506917 9.6485345

Within cluster sum of squares by cluster: [1] 5391.896 3363.327 2252.744 2215.088 (between_SS / total_SS = 56.7 %)

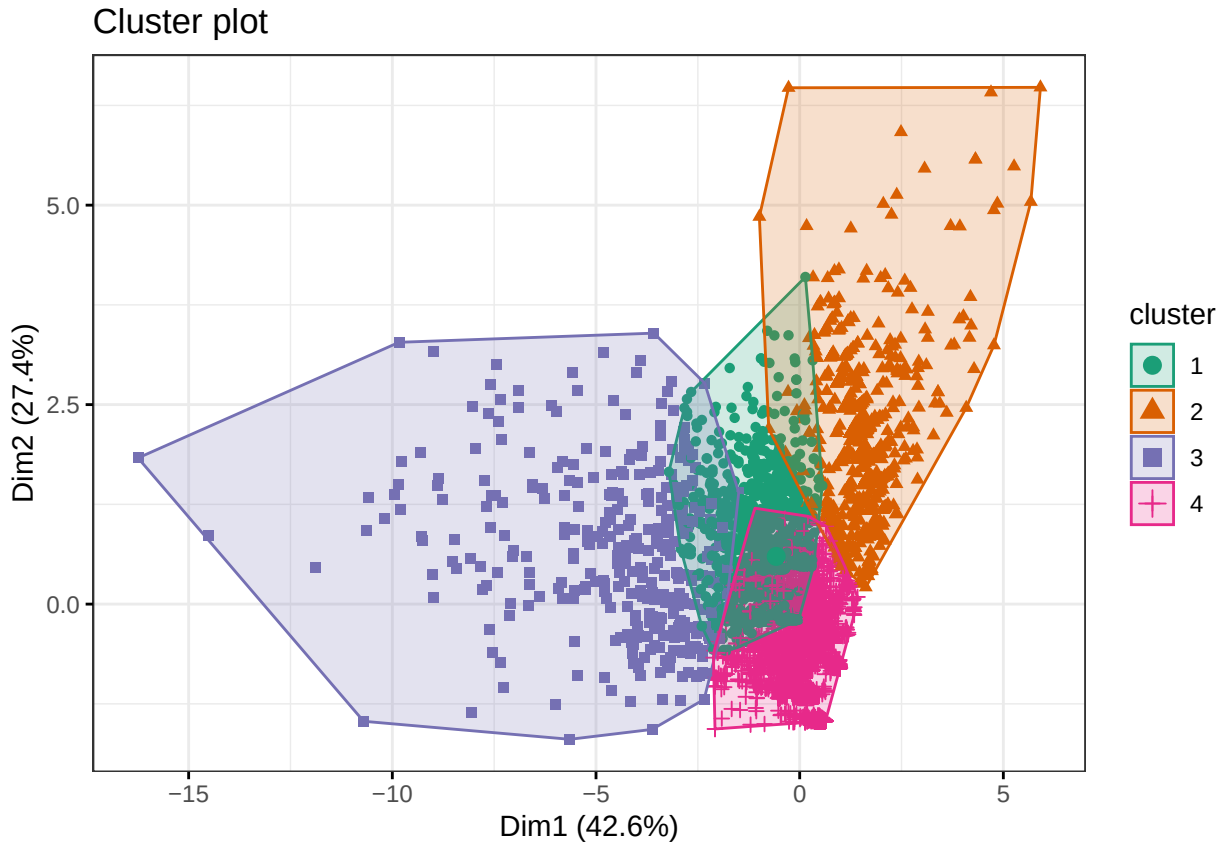
Available components:

[1] “cluster” “centers” “totss” “withinss” “tot.withinss” “betweenss” “size” “iter”
[9] “ifault”

The within sum of squares by cluster is 56.7%.

Plotting the 4 kmeans clustering results

```
library(RColorBrewer)
fviz_cluster(results_kmeans_4, data = normal_transactions_scaled,
             palette = brewer.pal(n = 8, name = "Dark2"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```




## Method 2: Clustering with K-medoids

Following this tutorial: https://towardsdatascience.com/clustering-on-mixed-type-data-8bbd0a2569c3

K-means algorithm is limited in that it can only work with numerical data, whereas our dataset contains both numeric and categorical data.

We will now try the PAM clustering algorithm (Partitioning across medoids). K-medoids is more robust to outliers and noise than K-means. The reason for this is because K-medoids uses some of the observations as the cluster centres (medoids). This algorithm uses Gower distance to measure the partial dissimilarity across

individuals, and ranges in [0 1]. Standardization is first applied to the features, and the distance between individuals represents the average of all feature specific distances.

Partial dissimilarity is different depending on the type of variable: numeric or categorical.

Numeric features - partial dissimilarity is dependent on absolute difference between 2 observations (x_i and x_j), and the maximum range observed from all individuals. d_ijˆf = |x_i - x_j|/|max_N(x) - min_N(x))| where N is the number of individuals in a dataset (Filaire, 2018).

Categorical/Qualitative features - feature dissimilarity is equal to 1 if y_i and y_j do not have the same values, otherwise it is 0.

## Data Preparation

For this K-medoids analysis we will omit the Region variable because there are only 4 values and the information is too broad.

We will try to cluster transactions according to the following features:

```
data2 <- data1 %>%
             select(Ship_Mode, Segment, City, State, Sub_Category, diff_in_days, Sales, Quantity, Disc

#convert all character data type to factor:
data2[sapply(data2, is.character)] <- lapply(data2[sapply(data2, is.character)],
                                       as.factor)
```

Compute the Gower distance

```
gower_dist <- daisy(data2, metric = "gower")

gower_mat <- as.matrix(gower_dist)
```

```
#Print most similar transactions
data2[which(gower_mat == min(gower_mat[gower_mat != min(gower_mat)]), arr.ind = TRUE)[1, ], ]
```

```
##           Ship_Mode  Segment    City State Sub_Category diff_in_days Sales
## 4875 Standard Class Consumer Houston Texas      Binders            4 1.188
## 3326 Standard Class Consumer Houston Texas      Binders            4 1.234
##      Quantity Discount  Profit
## 4875        1      0.8 -1.9602
## 3326        1      0.8 -1.9744
```

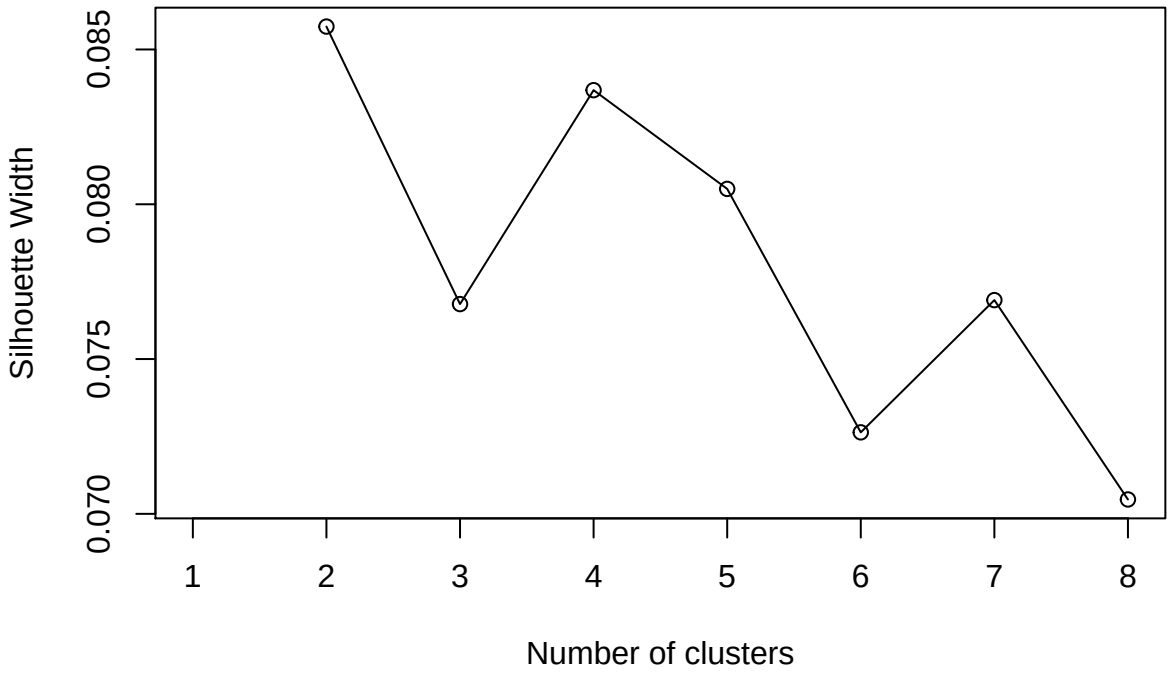
```
#Print most dissimilar transactions
data2[which(gower_mat == max(gower_mat[gower_mat != max(gower_mat)]), arr.ind = TRUE)[1, ], ]
```

```
##           Ship_Mode   Segment      City   State Sub_Category diff_in_days
## 6827 Standard Class Corporate Lafayette Indiana      Copiers            7
## 6798       Same Day  Consumer   Houston   Texas      Binders            0
##          Sales Quantity Discount    Profit
## 6827 17499.950        5      0.0 8399.9760
## 6798     3.798        1      0.8   -6.0768
```

### Determine the optimal number of clusters using the Silhouette Width Method

Try to figure out the number of clusters to use by using the silhouette coefficient. Typically the number of clusters used is between 2 and 8.

```
sil_width <- c(NA)
for (i in 2:8){
  pam_fit <- pam(gower_dist, diss = TRUE, k = i )
  sil_width[i] <- pam_fit$silinfo$avg.width
}

plot(1:8, sil_width,
     xlab = "Number of clusters",
     ylab = "Silhouette Width")
lines(1:8, sil_width)
```




2 clusters has the highest silhouette width, while 4 has the second highest. Higher silhouette width is generally better. 2 Clusters may be too simple, so we will pick k = 4.

To look at a summary of each of the clusters:

```
k <- 4
pam_fit_4 <- pam(gower_dist, diss = TRUE, k)
pam_results_4 <- data2 %>%
  mutate(cluster = pam_fit_4$clustering) %>%
  group_by(cluster) %>%
```

```
  do(the_summary = summary(.))
pam_results_4$the_summary
```

```
## [[1]]
##           Ship_Mode            Segment                City             State
##  First Class   : 528   Consumer   :1586   San Francisco: 279   California:638
##  Same Day      : 284   Corporate  : 386   Seattle      : 113   Texas     :195
##  Second Class  :1262   Home Office: 200   Los Angeles  : 100   Washington:142
##  Standard Class:  98                      Houston      :  85   Ohio      :127
##                                           Chicago      :  66   Illinois  : 95
##                                           Columbus     :  53   New York  : 90
##                                           (Other)      :1476   (Other)   :885
##       Sub_Category  diff_in_days       Sales             Quantity
##  Furnishings:407   Min.   :0.000   Min.   :    0.99   Min.   : 1.000
##  Paper      :260   1st Qu.:2.000   1st Qu.:   18.58   1st Qu.: 2.000
##  Accessories:196   Median :2.000   Median :   54.79   Median : 3.000
##  Storage    :193   Mean   :2.358   Mean   :  216.38   Mean   : 3.699
##  Art        :188   3rd Qu.:3.000   3rd Qu.:  197.91   3rd Qu.: 5.000
##  Phones     :168   Max.   :7.000   Max.   :13999.96   Max.   :14.000
##  (Other)    :760
##     Discount          Profit             cluster
##  Min.   :0.0000   Min.   :-1862.312   Min.   :1
##  1st Qu.:0.0000   1st Qu.:    2.936   1st Qu.:1
##  Median :0.0000   Median :    9.752   Median :1
##  Mean   :0.1113   Mean   :   32.005   Mean   :1
##  3rd Qu.:0.2000   3rd Qu.:   31.693   3rd Qu.:1
##  Max.   :0.8000   Max.   : 6719.981   Max.   :1
##
##
## [[2]]
##           Ship_Mode            Segment                City
##  First Class   : 309   Consumer   :2746   Los Angeles  : 615
##  Same Day      :  69   Corporate  : 475   San Francisco: 165
##  Second Class  : 159   Home Office: 237   Houston      : 152
##  Standard Class:2921                      Chicago      : 136
##                                           Seattle      : 135
##                                           San Diego    : 100
##                                           (Other)      :2155
##         State           Sub_Category   diff_in_days       Sales
##  California:1142   Binders    :1063   Min.   :0.000   Min.   :    0.444
##  Texas     : 408   Phones     : 312   1st Qu.:4.000   1st Qu.:   14.943
##  Illinois  : 221   Art        : 263   Median :5.000   Median :   49.564
##  Washington: 164   Storage    : 263   Mean   :4.711   Mean   :  225.913
##  Ohio      : 161   Accessories: 231   3rd Qu.:6.000   3rd Qu.:  217.584
##  Florida   : 135   Chairs     : 213   Max.   :7.000   Max.   :10499.970
##  (Other)   :1227   (Other)    :1113
##     Quantity         Discount          Profit             cluster
##  Min.   : 1.000   Min.   :0.0000   Min.   :-6599.978   Min.   :2
##  1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:   -0.936   1st Qu.:2
##  Median : 3.000   Median :0.2000   Median :    6.774   Median :2
##  Mean   : 3.846   Mean   :0.2148   Mean   :   19.673   Mean   :2
##  3rd Qu.: 5.000   3rd Qu.:0.2000   3rd Qu.:   25.192   3rd Qu.:2
##  Max.   :14.000   Max.   :0.8000   Max.   : 5039.986   Max.   :2
```

```
##
##
## [[3]]
##           Ship_Mode           Segment               City               State
##  First Class   : 281   Consumer   : 295   Philadelphia:506   Pennsylvania:537
##  Same Day      :  83   Corporate  : 190   Seattle     : 79   Texas       :147
##  Second Class  : 204   Home Office:1188   Houston     : 66   Illinois    : 90
##  Standard Class:1105                      Chicago     : 55   Washington  : 89
##                                           Columbus    : 34   Ohio        : 85
##                                           Springfield : 31   California  : 72
##                                           (Other)     :902   (Other)     :653
##       Sub_Category  diff_in_days       Sales             Quantity
##  Paper      :322   Min.   :0.000   Min.   :    0.852   Min.   : 1.000
##  Binders    :187   1st Qu.:3.000   1st Qu.:   15.552   1st Qu.: 2.000
##  Phones     :163   Median :4.000   Median :   47.984   Median : 3.000
##  Storage    :141   Mean   :4.014   Mean   :  216.941   Mean   : 3.727
##  Accessories:137   3rd Qu.:5.000   3rd Qu.:  203.976   3rd Qu.: 5.000
##  Furnishings:133   Max.   :7.000   Max.   :22638.480   Max.   :14.000
##  (Other)    :590
##     Discount          Profit             cluster
##  Min.   :0.0000   Min.   :-3399.980   Min.   :3
##  1st Qu.:0.0000   1st Qu.:   -2.131   1st Qu.:3
##  Median :0.2000   Median :    5.443   Median :3
##  Mean   :0.2065   Mean   :   10.078   Mean   :3
##  3rd Qu.:0.2000   3rd Qu.:   20.539   3rd Qu.:3
##  Max.   :0.8000   Max.   : 2591.957   Max.   :3
##
##
## [[4]]
##           Ship_Mode           Segment               City
##  First Class   : 420   Consumer   : 564   New York City: 789
##  Same Day      : 107   Corporate  :1969   Seattle      : 101
##  Second Class  : 320   Home Office: 158   Houston      :  74
##  Standard Class:1844                      Columbus     :  70
##                                           Springfield  :  59
##                                           Chicago      :  57
##                                           (Other)      :1541
##         State           Sub_Category  diff_in_days       Sales
##  New York  : 925   Paper      :600   Min.   :0.000   Min.   :    1.24
##  Texas     : 235   Storage    :249   1st Qu.:4.000   1st Qu.:   20.34
##  California: 149   Phones     :246   Median :4.000   Median :   62.28
##  Washington: 111   Art        :215   Mean   :4.248   Mean   :  253.84
##  Ohio      :  96   Accessories:211   3rd Qu.:5.000   3rd Qu.:  219.10
##  Florida   :  94   Furnishings:205   Max.   :7.000   Max.   :17499.95
##  (Other)   :1081   (Other)    :965
##     Quantity         Discount           Profit             cluster
##  Min.   : 1.000   Min.   :0.00000   Min.   :-3839.990   Min.   :4
##  1st Qu.: 2.000   1st Qu.:0.00000   1st Qu.:    3.972   1st Qu.:4
##  Median : 3.000   Median :0.00000   Median :   12.118   Median :4
##  Mean   : 3.828   Mean   :0.08585   Mean   :   49.049   Mean   :4
##  3rd Qu.: 5.000   3rd Qu.:0.20000   3rd Qu.:   39.660   3rd Qu.:4
##  Max.   :14.000   Max.   :0.80000   Max.   : 8399.976   Max.   :4
##
```

The first cluster has the majority values: ship mode is Second Class , segment is Consumer , City is San Francisco/other, State is California/other, sub category is furnishings. The mean difference in days between order date and ship date is 2.358 days, the mean value for Sales is $216.38, the mean quantity is 3.699, the discount mean is $0.1113, profit mean is $32.

For cluster 2, the ship mode is mostly standard class, segment is mostly consumer, city is mostly Los Angeles/other, State is mostly California/Other, subcategory is mostly Binders/other. The mean difference in days between order date and ship date is 4.7 days, the mean sales $225.91, mean quantity is 3.48, discount is 0.2148, and profit mean is $19.67.

For cluster 3, the main ship mode is First class, the segment is mostly Home Office, the most common city is Philadelphia/Other, the most common State is Pennsylvania/Other, and the most common subcategory is paper/other. For numeric variables, the mean difference in days between order date and ship date is 4.04 days, the mean sales $216.94, mean quantity is 3.72, mean discount is 0.2065, and mean profit is $10.08.

For cluster 4, the main ship mode is First class, the most common segment is Corporate, New York City/Other is the most common city, New York State/Other is the most common state, and the most common subcategory is paper. For the numeric features, the mean difference in days between order date and ship date is 4.25 days, the mean sales is $253.84, and mean quantity is 3.83, mean discount is 0.086, mean profit is $49.05.

We can now visualise the clusters in lower dimensional space with tSNE (t-Distributed Stochastic Neighbor Embedding) which can be used for dimensionality reduction.

```
#summary(pam_fit_4)
```

Medoids: ID
[1,] 560 560 [2,] 9 9 [3,] 951 951 [4,] 6943 6943

Objective function: build swap 0.3215923 0.3208374

Numerical information per cluster: size max_diss av_diss diameter separation [1,] 2172 0.5369861 0.3495248 0.7527143 0.0004132536 [2,] 3458 0.6045552 0.3040315 0.8092413 0.0142857143 [3,] 1673 0.5553794 0.3280627 0.7789958 0.0142857143 [4,] 2691 0.5881813 0.3147869 0.8127438 0.0004132536

Isolated clusters: L-clusters: character(0) L*-clusters: character(0)

Average silhouette width per cluster: [1] 0.07665593 0.11233789 0.05559724 0.07000398 Average silhouette width of total data set: [1] 0.08368581

Available components: [1] “medoids” “id.med” “clustering” “objective” “isolation” “clusinfo” “silinfo” “diss” “call”

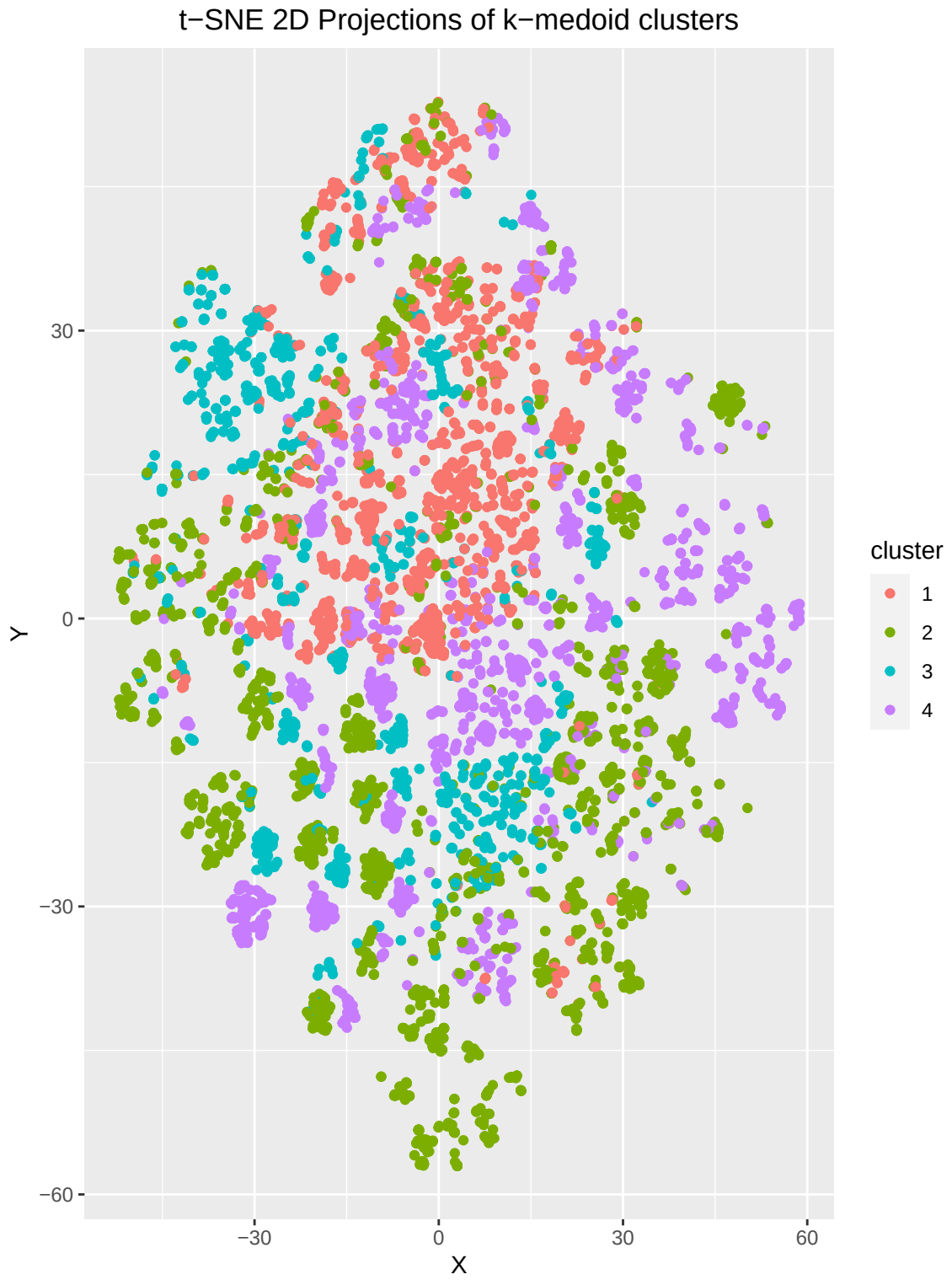
These were the observations used as the medoids for the pam fit with k=4:

```
data2[pam_fit_4$medoids, ]
```

```
##           Ship_Mode     Segment          City        State Sub_Category
## 560    Second Class    Consumer San Francisco   California  Furnishings
## 9    Standard Class    Consumer   Los Angeles   California      Binders
## 951  Standard Class Home Office  Philadelphia Pennsylvania        Paper
## 6943 Standard Class   Corporate New York City     New York        Paper
##      diff_in_days  Sales Quantity Discount  Profit
## 560             2 42.600        3      0.0 16.6140
## 9               5 18.504        3      0.2  5.7825
## 951             4 15.552        3      0.2  5.4432
## 6943            4 68.520        3      0.0 31.5192
```

Plotting the k-medoids with k=4. This method applies t-distributed stochastic neighborhood embedding (t-SNE) which enables visualization of several variables in a lower dimensional space.

```
tsne_obj <- Rtsne(gower_dist, is_distance = TRUE)

tsne_data <- tsne_obj$Y %>%
  data.frame() %>%
  setNames(c("X", "Y")) %>%
  mutate(cluster = factor(pam_fit_4$clustering))

ggplot(aes(x = X, y = Y), data = tsne_data) +
  geom_point(aes(color = cluster)) +
  labs(title = 't-SNE 2D Projections of k-medoid clusters') +
  theme(plot.title = element_text(hjust = 0.5))
```




Clusters still look like they have some overlap, but cluster 1 is pretty well defined.

```
input_data <- data2
```

We also tried using k =3 for k-medoids clustering.

Let's try using k=3:

```
gower_dist_3 = daisy(input_data, metric = "gower", type = list(logratio = 3))
                  gower_mat = as.matrix(gower_dist_3)
                  pam_fit_3 = pam(gower_mat, k=3, diss=TRUE)
```

```
#summary(pam_fit_3)
```

some information from the summary of pam fit for k=3:

Medoids: ID
[1,] "399" "399" [2,] "9" "9"
[3,] "7119" "7119"

Numerical information per cluster: size max_diss av_diss diameter separation [1,] 2361 0.6204122 0.3754422 0.7803200 0.01490751 [2,] 4281 0.6045552 0.3173436 0.8092413 0.01524512 [3,] 3352 0.5882532 0.3311972 0.8127438 0.01490751

Isolated clusters: L-clusters: character(0) L*-clusters: character(0)

Average silhouette width per cluster: [1] 0.06840880 0.09434220 0.06023282 Average silhouette width of total data set: [1] 0.07677532

Available components: [1] "medoids" "id.med" "clustering" "objective" "isolation" "clusinfo" "silinfo" "diss" "call"

The average silhouette width of the total data set is 0.0767753.

```
pam_results_3 <- data2 %>%
  mutate(cluster = pam_fit_3$clustering) %>%
  group_by(cluster) %>%
  do(the_summary = summary(.))
pam_results_3$the_summary
```

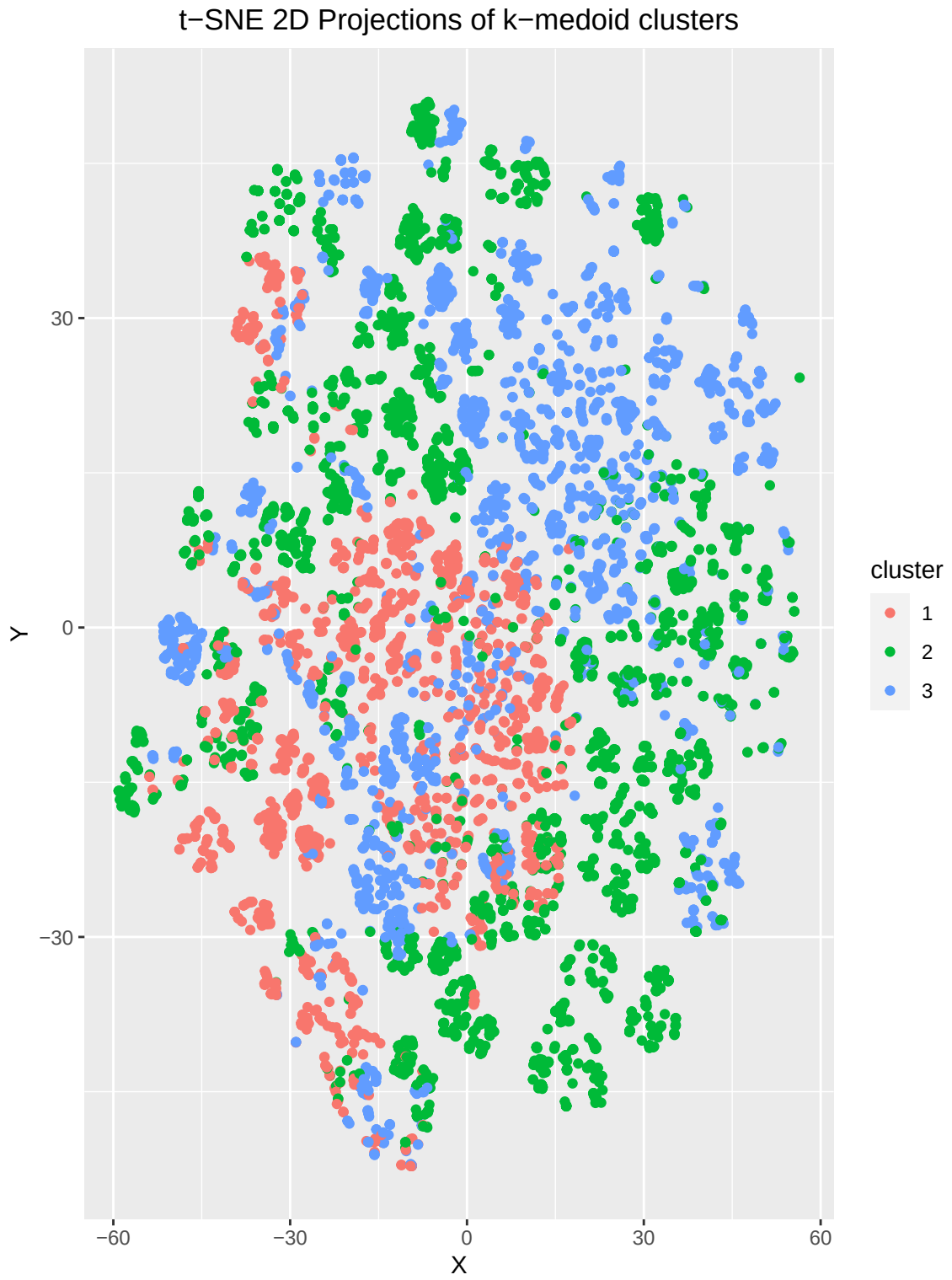
```
## [[1]]
##           Ship_Mode           Segment                City
##  First Class   : 650   Consumer   :1593   Houston      : 286
##  Same Day      : 276   Corporate  : 342   Philadelphia : 137
##  Second Class  :1277   Home Office: 426   Seattle      : 135
##  Standard Class: 158                      Chicago      :  90
##                                           Dallas       :  64
##                                           San Francisco:  63
##                                           (Other)      :1586
##           State           Sub_Category  diff_in_days       Sales
##  Texas       : 528   Storage    :340   Min.   :0.000   Min.   :    0.444
##  Ohio        : 171   Furnishings:251   1st Qu.:2.000   1st Qu.:   19.152
##  Washington  : 162   Paper      :251   Median :2.000   Median :   67.000
##  California  : 152   Phones     :238   Mean   :2.524   Mean   :  239.150
##  Pennsylvania: 150   Accessories:222   3rd Qu.:3.000   3rd Qu.:  239.980
##  Illinois    : 140   Art        :203   Max.   :7.000   Max.   :13999.960
##  (Other)     :1058   (Other)    :856
##     Quantity         Discount          Profit             cluster
##  Min.   : 1.000   Min.   :0.0000   Min.   :-2639.991   Min.   :1
##  1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:    0.691   1st Qu.:1
##  Median : 3.000   Median :0.2000   Median :    6.871   Median :1
##  Mean   : 3.713   Mean   :0.1758   Mean   :   20.726   Mean   :1
```

```
##  3rd Qu.: 5.000   3rd Qu.:0.2000   3rd Qu.:   28.310   3rd Qu.:1
##  Max.   :14.000   Max.   :0.8000   Max.   : 6719.981   Max.   :1
##
##
## [[2]]
##          Ship_Mode           Segment                City
##  First Class   : 391   Consumer   :3015   Los Angeles  : 710
##  Same Day      : 138   Corporate  : 549   San Francisco: 335
##  Second Class  : 292   Home Office: 717   Philadelphia : 257
##  Standard Class:3460                      Chicago      : 154
##                                           Seattle      : 154
##                                           San Diego    : 133
##                                           (Other)      :2538
##          State           Sub_Category   diff_in_days       Sales
##  California  :1583   Binders    :1217   Min.   :0.000   Min.   :    0.836
##  Texas       : 283   Phones     : 376   1st Qu.:4.000   1st Qu.:   14.910
##  Pennsylvania: 282   Furnishings: 354   Median :5.000   Median :   47.976
##  Illinois    : 253   Art        : 317   Mean   :4.497   Mean   :  224.406
##  Ohio        : 185   Accessories: 299   3rd Qu.:5.000   3rd Qu.:  206.962
##  Washington  : 184   Paper      : 285   Max.   :7.000   Max.   :22638.480
##  (Other)     :1511   (Other)    :1433
##     Quantity         Discount          Profit             cluster
##  Min.   : 1.000   Min.   :0.0000   Min.   :-6599.978   Min.   :2
##  1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:    0.000   1st Qu.:2
##  Median : 3.000   Median :0.2000   Median :    6.888   Median :2
##  Mean   : 3.815   Mean   :0.2051   Mean   :   19.091   Mean   :2
##  3rd Qu.: 5.000   3rd Qu.:0.2000   3rd Qu.:   24.858   3rd Qu.:2
##  Max.   :14.000   Max.   :0.8000   Max.   : 5039.986   Max.   :2
##
##
## [[3]]
##          Ship_Mode           Segment                City
##  First Class   : 497   Consumer   : 583   New York City: 796
##  Same Day      : 129   Corporate  :2129   Philadelphia : 143
##  Second Class  : 376   Home Office: 640   Seattle      : 139
##  Standard Class:2350                      San Francisco: 112
##                                           Columbus     :  87
##                                           Chicago      :  70
##                                           (Other)      :2005
##          State           Sub_Category   diff_in_days      Sales
##  New York    : 935   Paper      : 834   Min.   :0.00   Min.   :    1.24
##  California  : 266   Furnishings: 352   1st Qu.:4.00   1st Qu.:   18.96
##  Texas       : 174   Art        : 276   Median :4.00   Median :   53.28
##  Washington  : 160   Phones     : 275   Mean   :4.28   Mean   :  230.28
##  Pennsylvania: 155   Accessories: 254   3rd Qu.:5.00   3rd Qu.:  197.18
##  Ohio        : 113   Storage    : 237   Max.   :7.00   Max.   :17499.95
##  (Other)     :1549   (Other)    :1124
##     Quantity         Discount          Profit             cluster
##  Min.   : 1.000   Min.   :0.00000   Min.   :-3839.99   Min.   :3
##  1st Qu.: 2.000   1st Qu.:0.00000   1st Qu.:    4.02   1st Qu.:3
##  Median : 3.000   Median :0.00000   Median :   11.65   Median :3
##  Mean   : 3.811   Mean   :0.07989   Mean   :   46.46   Mean   :3
##  3rd Qu.: 5.000   3rd Qu.:0.20000   3rd Qu.:   37.14   3rd Qu.:3
##  Max.   :14.000   Max.   :0.80000   Max.   : 8399.98   Max.   :3
```

##

### Visualise the clusters using T-SNE

```
tsne_obj <- Rtsne(gower_dist_3, is_distance = TRUE)

tsne_data <- tsne_obj$Y %>%
  data.frame() %>%
  setNames(c("X", "Y")) %>%
  mutate(cluster = factor(pam_fit_3$clustering))

ggplot(aes(x = X, y = Y), data = tsne_data) +
  geom_point(aes(color = cluster)) +
   labs(title = 't-SNE 2D Projections of k-medoid clusters') +
  theme(plot.title = element_text(hjust = 0.5))
```




### Evaluating the consistency within Clusters of Data

Silhouette coefficient can be used to compare the average distance to observations within the same cluster, to the average distance to observations in other clusters.

High silhouette coefficient means that the observation is well clustered, while a low silhouette coefficient may indicate outliers.

We also tried k=5 for k-medoids clustering. For k= 5, The average silhouette width per cluster were: [1] 0.05349182 0.10906323 0.04114758 0.08911682 0.07800729. The average silhouette width of the total data set for k=5 is 0.08049662, which is less than the silhouette width for k=4 (0.08368581). The optimal number of clusters is k=4, since it maximises the average silhouette width. Higher silhouette width values indicate better quality clustering (Datanova, 2018). We also plotted the clustering result using the gower distance metric and t-SNE for dimensionality reduction. However, the plot shows there is still a great deal of overlap between the clusters.

## Method 3 - Clustering on Principal Components using Factor analysis of mixed data (FAMD)

Factor analysis of mixed data is a principal component method useful for analyzing datasets which contain numeric and categorical data.

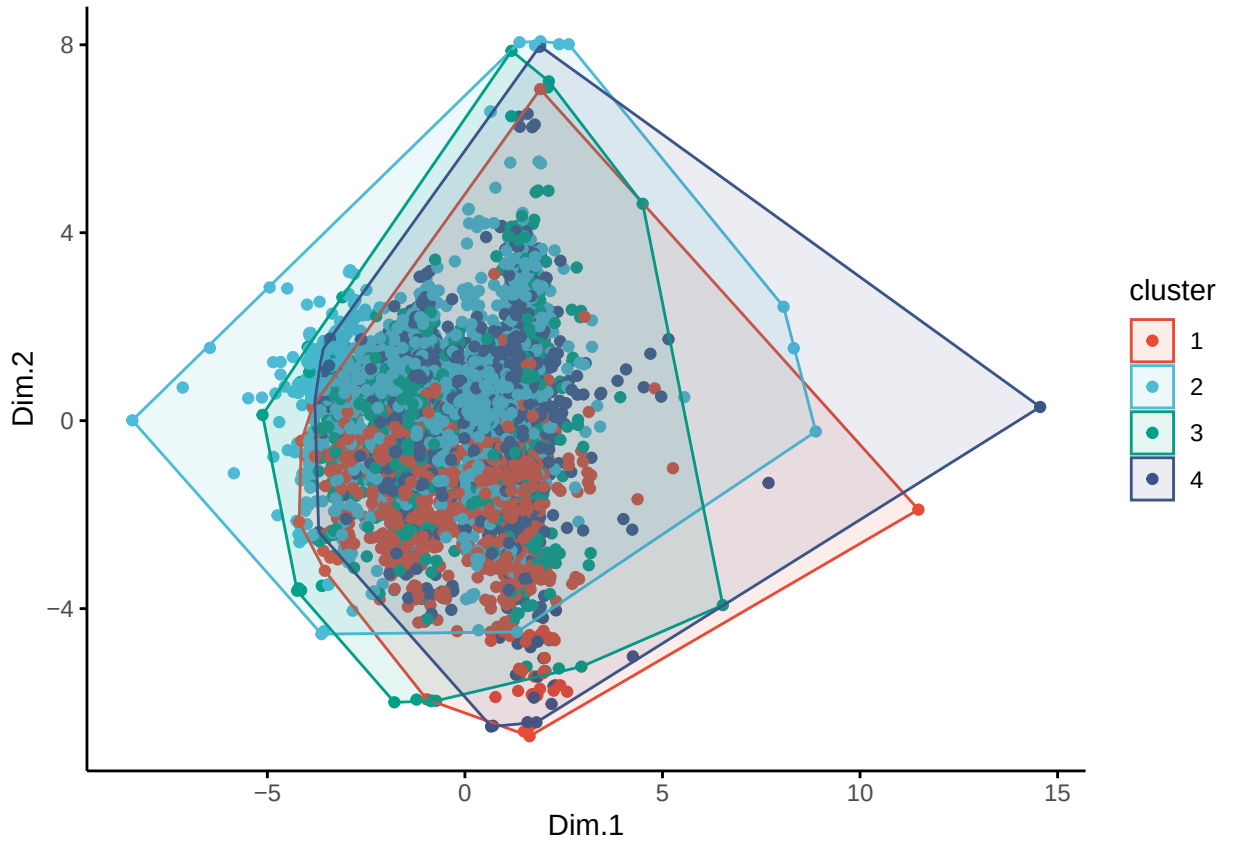
Quantitative variables and qualitative variables are normalized to obtain a balance of the different sets of variables (Kassambra, 2017).

To run the Factor Analysis of Mixed Data method:

```
set.seed(123)

famd_fca=FAMD(input_data, ncp = 5, sup.var = NULL, ind.sup = NULL, graph = FALSE)  #ncp is the number o

ind=get_famd_ind(famd_fca)$coord   #retrieve the results for the individual observations
ind=as.data.frame(ind)
ind$cluster=as.factor(pam_fit_4$clustering)
```

Plotting the clusters found using Factor Analysis of Mixed Data (FAMD).

```
  ggscatter(
    ind, x = "Dim.1", y = "Dim.2",
    color = "cluster", palette = "npg", ellipse = TRUE, ellipse.type = "convex",    #adding the concent
    size = 1.5,  legend = "right", ggtheme = theme_classic()

  )
```




## Deployment

The clustering model and anomaly detection app are deployed in shiny app. The interactive applications accepts inputs from the user and display accordingly. The url for application is below
Clustering - https://crystalzhu72.shinyapps.io/MixedDataClustering/

Anomaly Detection - https://mlgroupb.shinyapps.io/anomaly/

## Responsible ML Framework

Superstore dataset has transaction data for online orders from Superstore customers. The dataset has Personally Identifiable information like Customer Name and Postal code. The use of personal information for other purposes, subsequent to the original transaction between an individual and an organization need consent from the customer. Individuals are increasingly feeling that they are losing control over their own personal information that may reside on thousands of servers largely beyond the control of existing privacy laws. This scenario can led to privacy invasion on a scale never before possible if not handled ethically. In this project, personally identifiable attributes are removed from the data as part of data cleansing and not considered in any stages of the study. The techniques used in the project is not biased for any particular region, city or customer segment.

## Conclusion

In K means clustering, the goal is to have high similarity within each cluster, and low similarity between clusters. High similarity within a group means low variance within the cluster (within_SS), while low

similarity between the groups is the same as high variance between the clusters (between_SS). Total_SS is all the variance in the data. The goal is to maximize the between_ss/total_ss value.

For K-means clustering, for k=4 (4 clusters), when we perform clustering on the data without removing the outliers first, the Within cluster sum of squares by cluster: is 3315.203, 4284.643, 7571.058, 3614.724, and the between_SS / total_SS = 53.0 %. In comparison, when outlier transactions are first removed using the Local Outlier Factor method, for k=4, the Kmeans Within cluster sum of squares by cluster values are: 5391.896, 3363.327, 2252.744, 2215.088 and between_SS / total_SS = 56.7 %. This shows that by removing the outlier transactions first before performing K-means clustering, there is better clustering with a greater between_SS / total_SS percentage.

For clustering with K-medoids, using the gower distance metric is useful when the dataset includes both numeric and categorical data. In this project, we found 4 optimal clusters for different Superstore customers: cluster 1 is made of second class shippping, Consumer segment x San Franciso x California x furnishings. Cluster 2 is made of standard class shipping x consumer segment x Los Angeles x California x Binders. Cluster 3 is First class shipping x Home Office segment x Philadelphia x Pennsylvania x Paper. Cluster 3 has the lowest mean profit of the 4 clusters. Cluster 4 is made of First Class shipping x Corporate segment x New York City x New York State x Paper. Cluster 4 has the highest mean sales amount and mean profit, with lowest mean discount.

The Superstore can now use the commonalities for customers within these clusters to adjust product offerings on their website in order to meet customer needs. For example, for cluster 4, the customers are generally from New York and mainly buy corporate type products with the highest mean sales amount of $253.84. Using this data, the Superstore can have their website show more business related products which may be more expensive than products from other segments.

When evaluating the clustering using the t-SNE method to examine the clusters in 2 dimensions, we can see that there is still some overlap between the different clusters. We also evaluated the clustering by examining the silhouette width, by choosing the optimal number of clusters with maximal silhouette width. In our case, the optimal number of clusters of k=4 produced a silhouette width of 0.08368581.

**Ways to Improve Clustering Results in Future**

**For Kmeans clustering which uses Euclidean distance:** To improve the clustering results we may have to perform some feature engineering to reduce observations which have values of 0.5 in the similarity matrix. Similarity is distance based, where similarity measures have value in [0, 1] where 0 is no similarity and 1 is identical. TO reduce the noise due to observations having values of 0.5 on the similarity matrix, we can use the loss metric to get the 0.5 similarity values closer to either 0 or 1 (Sinclair, 2018).

$$E(w) = \frac{2}{N(N-1)} \sum_{q<p} \frac{1}{2} \left( \rho_{pq}^{(w)} \left( 1 - \rho_{pq}^{(1)} \right) + \rho_{pq}^{(1)} \right.$$

The equation for the loss metric is below:

where 1 = base weights, p = resulting fuzzy partition matrix which is a product of the weights used in the euclidean distance function between points p and q.

Afterwards, perform Gradient Descent on the loss function to get the local minimum of the function, and continuously update the weights until the maximum number of iterations are met or the function converges (Sinclair, 2018). By improving the feature weights, clusters will be easier to identify.

## References

(2017).Exploring Assumptions of K-means Clustering using R. Retrieved from https://www.r-bloggers.com/2017/08/exploring-assumptions-of-k-means-clustering-using-r/

Alboukadel .K-Means Clustering Visualization in R: Step By Step Guide. Retrieved from: https://www.datanovia.com/en/blog/k-means-clustering-visualization-in-r-step-by-step-guide/

Li, C.C., Detecting Outlier by Local Outlier Factor and Random Forest. Retrieved from: https://rstudio-pubs-static.s3.amazonaws.com/216270_ccd62999b6504c22a390bb39fee22f39.html

Dbscan. Retrieved from:https://www.rdocumentation.org/packages/dbscan/versions/1.1-5/topics/lof

Arkaghosh, N. (2020). Detect Anomalies in Time Series Using Anomalize Package In R. Retrieved from: https://www.analyticsvidhya.com/blog/2020/12/a-case-study-to-detect-anomalies-in-time-series-using-anomalize-package-in-r/

Filaire, T. (2018). Clustering on mixed type data. Retrieved from: https://towardsdatascience.com/clustering-on-mixed-type-data-8bbd0a2569c3

Fonseca, L. (2019). Clustering Analysis in R using K-means. Retrieved from: https://towardsdatascience.com/clustering-analysis-in-r-using-k-means-73eca4fb7967

Gama, K-Medoids in R: Algorithm and Practical Examples. Retrieved from:https://www.datanovia.com/en/lessons/k-medoids-in-r-algorithm-and-practical-examples/

Kassambara, Riad, & Visitor, FAMD - Factor Analysis of Mixed Data in R: Essentials. Retrieved from:http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/115-famd-factor-analysis-of-mixed-data-in-r-essentials/#comments-list (Kassambara, 2017)

K-means clustering with tidy data principles. Retrieved from: https://www.tidymodels.org/learn/statistics/k-means/

Manimaran (2019). Clustering Evaluation strategies. Retrieved from: https://towardsdatascience.com/clustering-evaluation-strategies-98a4006fcfc

Mayer, M. (2021). Using OutForest. Retrieved from: https://cran.r-project.org/web/packages/outForest/vignettes/outRanger.html

Nwanganga, F. and Chapple, M. (2020). Practical Machine Learning in R. Indianapolis, Indiana: Wiley & Sons.-

Sinclair, C. (2018). Improving Clustering Performance Using Feature Weight Learning. Retrieved from https://towardsdatascience.com/improving-clustering-performance-using-feature-weight-learning-d65d4fec77cb