Customer Segmentation Part 1

Abstract

The goal of this analysis is to segment customers into appropriate groups using their purchasing behavior in order to drive improved customer satisfaction and increase revenue. It also highlights opportunities to improve service and product lines.
The following insights and conclusions that were derived in the end of this analysis:

- Customers are segmented into four groups to better summarize their historical purchases and preferences.
- Most of the revenue comes from the group with steady relatively frequent orders (up to 50 times in a given period) and sales orders up to $4,500. This group is a great benchmark for evaluating remaining customer groups.
- Customers purchasing fewer items become potential targets for expanding company's business through the products that were not purchased before.
- Customers that buy less costly products more frequently products as well as those with rare purchases but high spending are most predictable and could be great target for optimizing the B2B operations.

Introduction

Question to answer: What information about their customers does the company need to have to drive revenue?

Data set 1 variables:

- SoldTo - Customer Accounts
- Sales (in a given period)
- NoSO - Number of Sales Orders
- Avg_So - Average spent per Sales Order

Data set 2 variables:

- SoldTo - Customer Accounts
- Category - Purchased Product Categories
- Sales - Sales per Category
- CatType - Category Class (Primary or not)

Exploratory Analysis

Like any other data set, it needs some exploration. So let's look at its nature of variables, formatting, missing values and whether it needs standardization.

First, I am going to look at first 6 rows and a structure. I need to convert 'Soldto' account into a character, since this is supposed to be a categorical variable that represents the customer account number.

```
sales$SoldTo<-as.character(sales$SoldTo)
sales<-sales[,-4]
str(sales)
```

```
## 'data.frame':    3868 obs. of  6 variables:
##  $ DstC  : chr  "US" "US" "US" "US" ...
##  $ PGroup: int  99 99 99 99 99 99 99 99 99 99 ...
##  $ SoldTo: chr  "100029" "100060" "100275" "100304" ...
##  $ Sales : num  483 1088 820 929 112 ...
##  $ NoSO  : int  1 1 1 4 1 1 1 11 1 10 ...
##  $ Avg_So: num  483 1088 820 232 112 ...
```

Next, I inspect the data set for missing values. The code says we have none, and this is expected since I've done all necessary cleaning to save the time and effort at the time of exporting the query from SQL Server earlier.

```
any(is.na(sales))
```

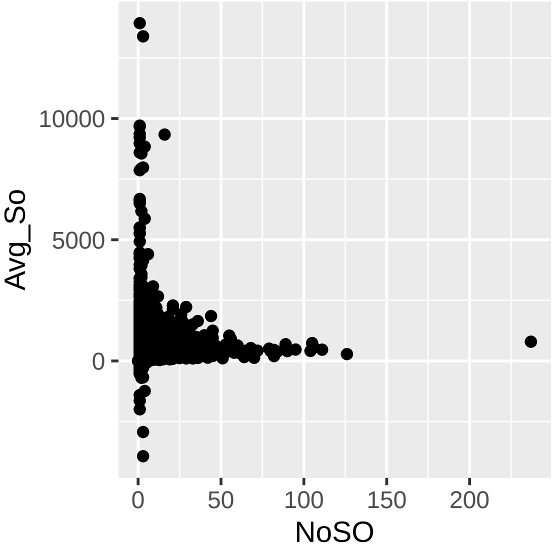
```
## [1] FALSE
```

```
#install.packages("Amelia")
#library(Amelia)
#missmap(sales, col=c('yellow', 'black'))
```

To evaluate this data visually, I made a dot plot. Obviously it is quite incomplete, however we still can derive a valuable next step:

- Values on Y and X axis are very different in amplitude. Standardization is needed to give same weight to each variable.

```
library(ggplot2)
ggplot(sales, aes(NoSO, Avg_So))+geom_point()
```



Standardization is important here, because clustering analysis is based on the calculated distances between the data points. Thus higher values (like absolute sales values) easily will pull all weights on themselves, leaving smaller values (average sales order value) with small or no weight at all.

```
library(dplyr)
rescale_sales<- sales %>%
  mutate(Sales=scale(Sales),
         NoSO= scale(NoSO),
         Avg_So=scale(Avg_So))
```

The next six rows demonstrate the standardized data set, which is now ready for data mining.

```
head(rescale_sales)
```

```
##   DstC PGroup SoldTo      Sales       NoSO      Avg_So
## 1   US     99 100029 -0.3693513 -0.4930885 -0.02857759
## 2   US     99 100060 -0.2870732 -0.4930885  0.71388252
## 3   US     99 100275 -0.3235727 -0.4930885  0.38451886
## 4   US     99 100304 -0.3087666 -0.2064252 -0.33610672
## 5   US     99 100382 -0.4198198 -0.4930885 -0.48399416
## 6   US     99 100423 -0.2953127 -0.4930885  0.63953105
```

Method: K-Means

Find Optimal k

Important to note, that k-means algorithm work only with numerical values, so the code below contains only columns in a range from 4 to 6 (Columns: Sales, NoSo, Avg_So).

K is the the number of groups I am anticipating to get. In my case I chose maximum 10. Even though I would like to arrive at smaller groups of customers (maximum 5) to have more manageable data set afterwards.

Algorithm iterates through all k values from 1 to 10 and calculates 'With-In-Sum-Of-Squares'(WSS) for each k.

WSS is a measure that explains the variation within a cluster. K is associated with the WSS that explains most homogeneous variation (longer chunk of line on an elbow chart below).

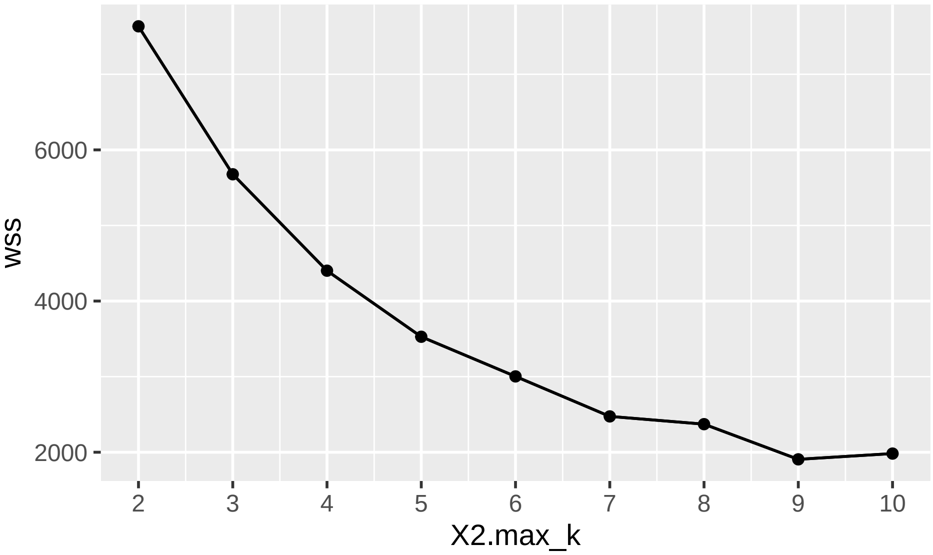
Elbow graph shows that first 4 groups would cover most of the variation (domain knowledge would be very valuable here to make a decision on optimal k). Number 4 as optimal K looks like a good starting point. If something does not make sense later, I would go back and chose a better value for k.

```
kmeans_withinss<- function(k){
  cluster<- kmeans(rescale_sales[,4:6], k)
  return(cluster$tot.withinss)
}

max_k<- 10
wss<-sapply(2:max_k,kmeans_withinss )


elbow <-data.frame(2:max_k, wss)

# Plot the graph
ggplot(elbow, aes(x = X2.max_k, y = wss)) +
    geom_point() +
    geom_line() +
    scale_x_continuous(breaks = seq(1, 10, by = 1))+
  geom_line(x2.mak_=5)
```



Now when optimal k is chosen, let's run the k-means algorithm:

```
kmeans.sales<-kmeans(rescale_sales[,4:6], 4)
```

Next we can include clusters into a new data frame and see first few records with cluster value in them:

```
##   DstC PGroup SoldTo   Sales NoSO   Avg_So Cluster
## 1   US     99 100029  482.97    1  482.970       2
## 2   US     99 100060 1088.41    1 1088.410       2
## 3   US     99 100275  819.83    1  819.830       2
## 4   US     99 100304  928.78    4  232.195       2
## 5   US     99 100382  111.60    1  111.600       2
## 6   US     99 100423 1027.78    1 1027.780       2
```

Visualizing the Clustered Data

Visual on the left (Fig 1) definitely says more now, compare to the graph on the right (Fig 2), before the clustering. Defined groups (Clusters):
- customers that buy less frequently and spend less on average per sales order, occasionally requesting refund (concentrated near (0,0) on a plot)
- customers that buy relatively often or have bigger purchases (sweet middle)
- customers that buy frequently and spend relatively less (spread more along X axis)
- customers that buy a few times and spend relatively more (spread more along Y axis)

Fig.1 After Clustering

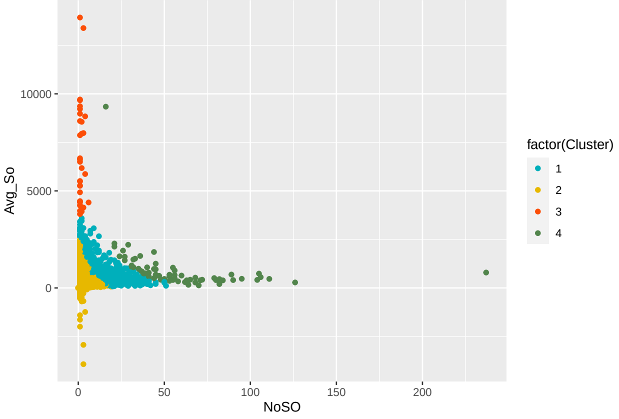
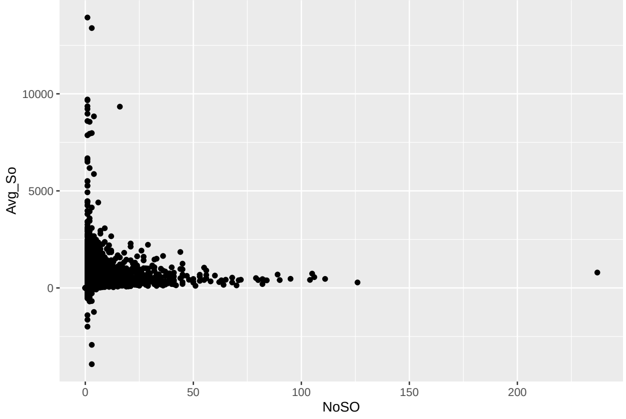


Fig.2 Before Clustering



Exploring Groups

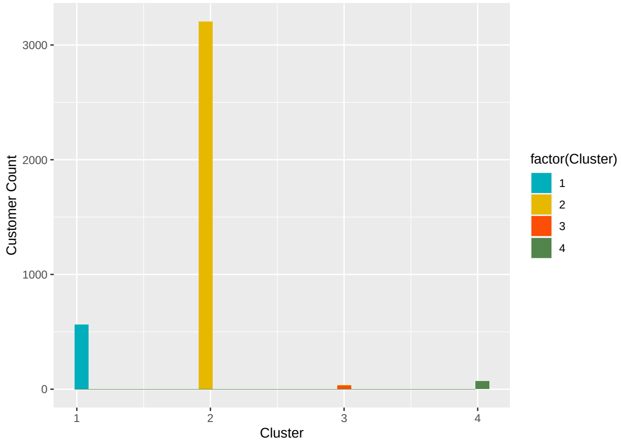
So now when we defined 4 groups of customers, let's see how the distribution of revenue looks like. Bar chart on the right shows that most of the revenue comes from the group that buys relatively often or have bigger purchases.

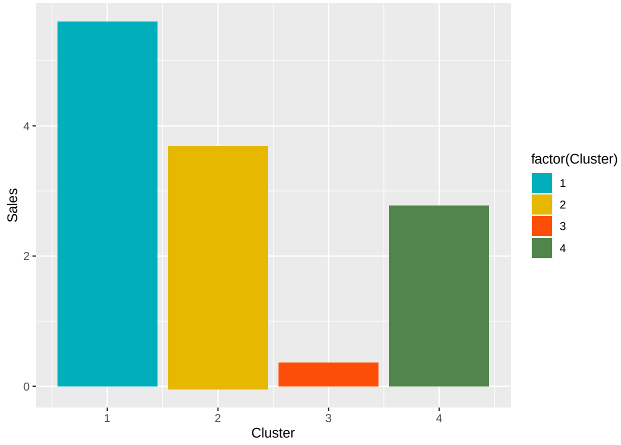
Derived insight: Keep such customers on a radar to maintain a healthy stream of the purchases and customer satisfaction.

Also, we can see that next group of customers that buys less frequent and spends less on average per sales order, occasionally requesting refund. However this group consists of lots of accounts (probably new customers with one-off sales) per chart on the left.

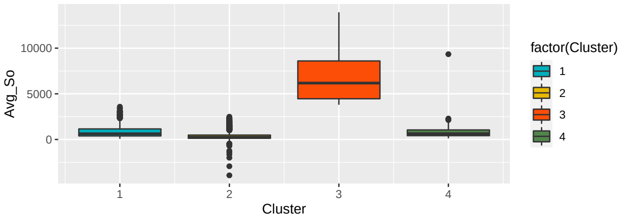
Derived insight: These customers need more engagement to tell us their needs and possible expansion of the purchases. **This is a potential group that can produce more revenue.**

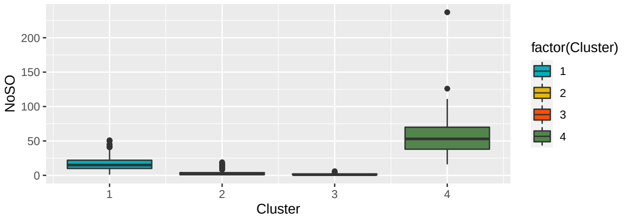
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

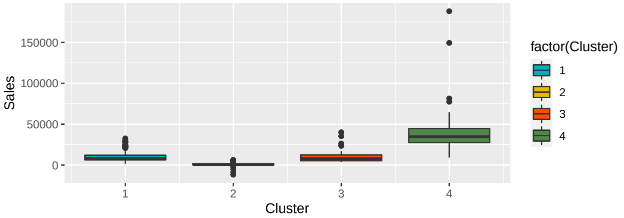




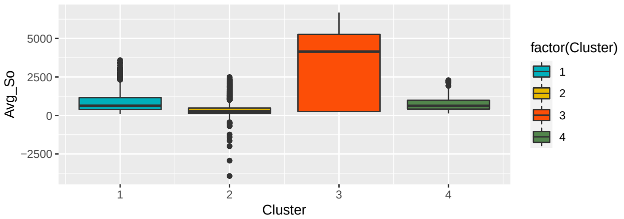
Box plots below deliver more understanding how these groups differ. First box plots are presented with the raw data, however then it made sense to exclude outliers.

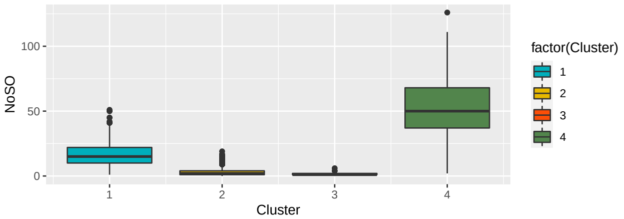


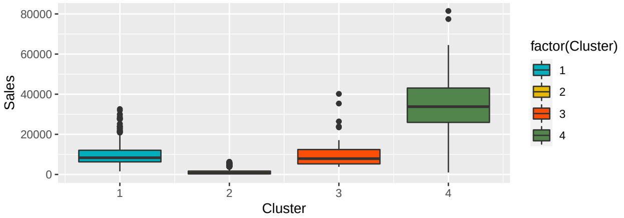




Without outliers visuals become more defined :







For a further reference I decided to store this data to excel:

```
#install.packages("xlsx")
library(xlsx)
write.xlsx(sales, "Customer Segmentation Output.xlsx", sheetName = "Sheet1",
  col.names = TRUE, row.names = TRUE, append = FALSE)
```

Customers Groups and Their Product Categories

Customer segmentation gave us the insight on the stream of revenue from the point of view of higher and lower spenders. However, to dig deeper into the groups and understand their purchasing behavior I decided to pull the product categories they purchase and analyze whether there is any valuable insight hidden.

```
## 'data.frame':    21436 obs. of  7 variables:
##  $ SoldTo  : chr  "100029" "100060" "100060" "100060" ...
##  $ Cluster : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Category: chr  "Air Supply Parts" "Detailing" "Support Equipment" "Vacuum Parts" ...
##  $ CatType : chr  "No" "No" "No" "Yes" ...
##  $ Sales   : num  483 976.2 82.2 30 320.4 ...
##  $ NoSO    : int  1 4 1 1 1 1 1 1 1 1 ...
##  $ Avg_So  : chr  "482.97" "244.04" "82.25" "30" ...
```

As usual, a little maintenance around data set is needed. So, I re-formatted a few variables in a data set and checked for missing values.

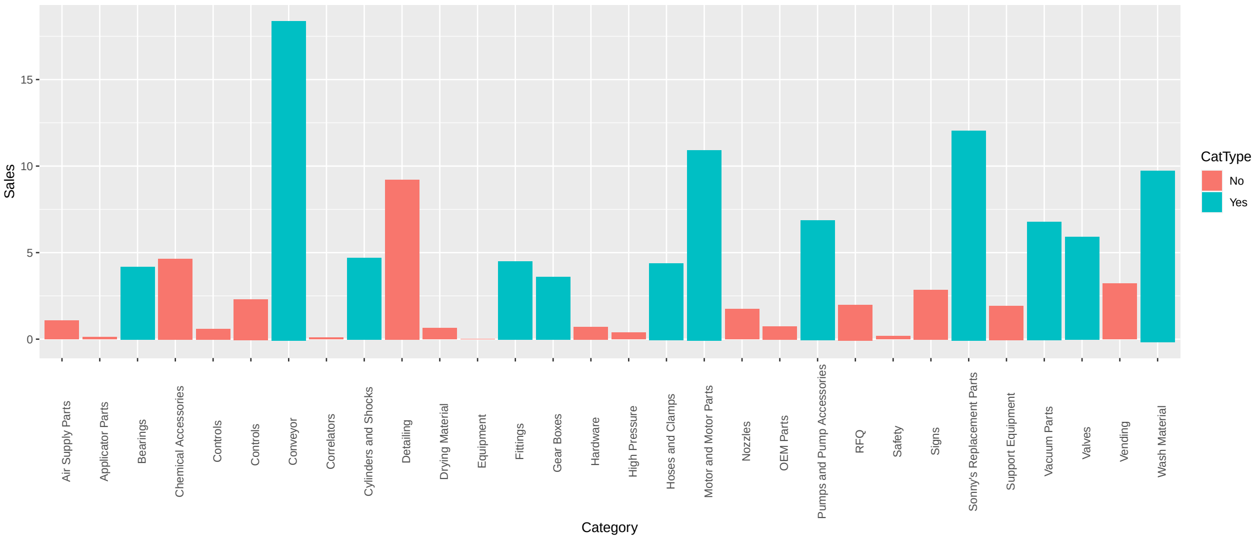
```
##  Factor w/ 4 levels "1","2","3","4": 2 2 2 2 2 2 2 2 2 2 ...

## [1] FALSE
```

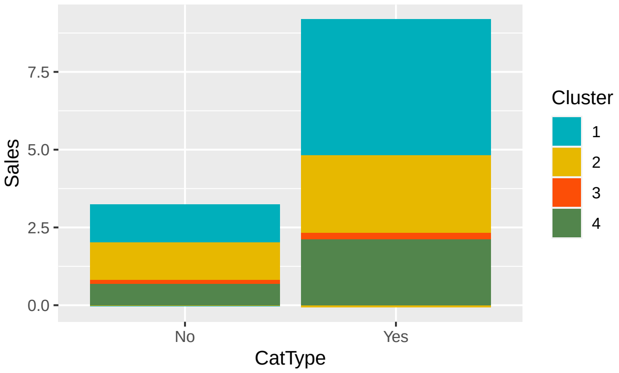
Data set 2 has two new variables:

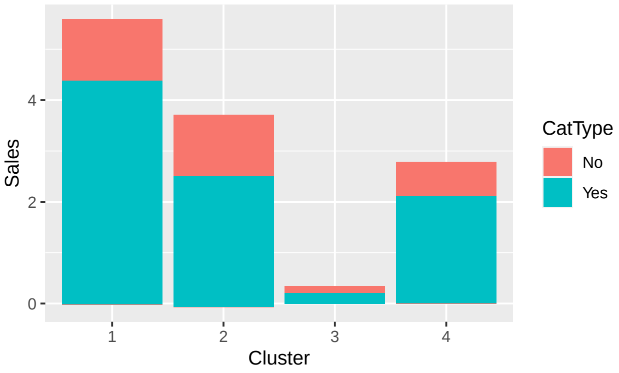
- Category - represent a product category customer purchases the product from
- CatType - tells whether this category is primary or not (internal decision defined by business based on 80-20 rule)

Bar chart below shows sales by categories:

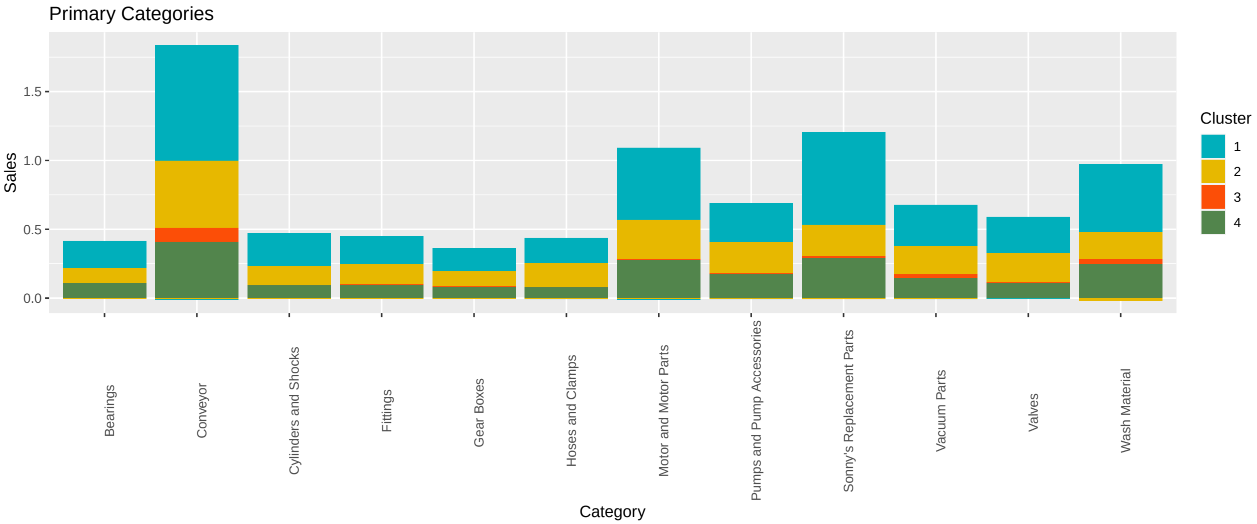


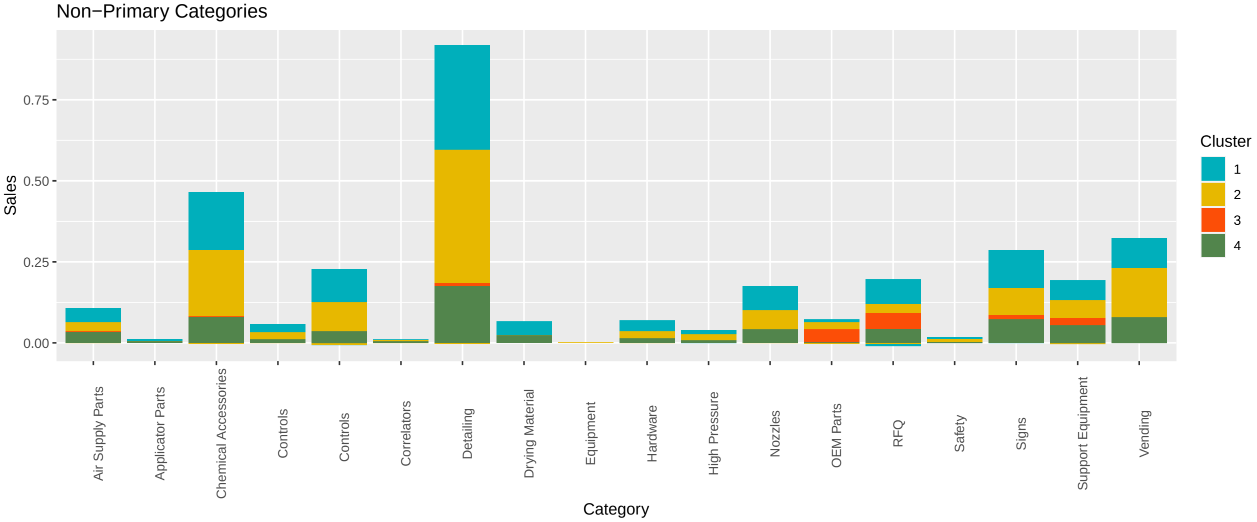
Next bar charts present sales by customer groups and category type (primary and not-primary)





Bar charts below present the purchasing behavior of customer groups by categories.





Box Plots Comparison of Group Sales and Number of Orders Placed

For a better interpretation, I am going to exclude some extreme values (outliers):

```
sales_major$Sales[sales_major$Sales > 10000 | sales_major$Sales < -3000] <- median(sales_major$Sales)
```

Even though bar charts gave us some good understanding of the sales distribution between categories and groups, it seemed a little difficult to read the information. To have a better comparison of the groups I decided to make box plots for each customer group separately.
To base the diction based on the value of sales only could be misleading also, so plots with number of orders are added to the visual grid also.

Group Insights:

1. Group that includes returns may indicate a bad experience for a customer. It is worth reviewing the credits and find the occurred issues.

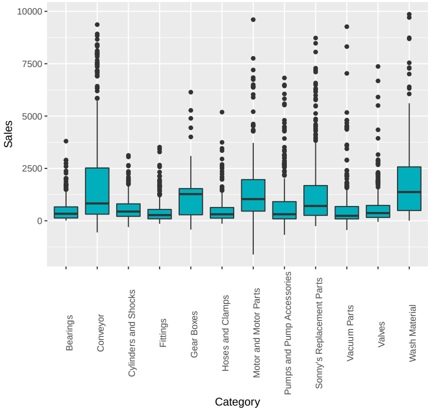
2. Categories with extreme values may represent one-off sales. It is a great reason to engage with customers and see whether they could become re-occurring clients.

3. Groups of customers with very little variation and high sales say that they prefer to purchase large

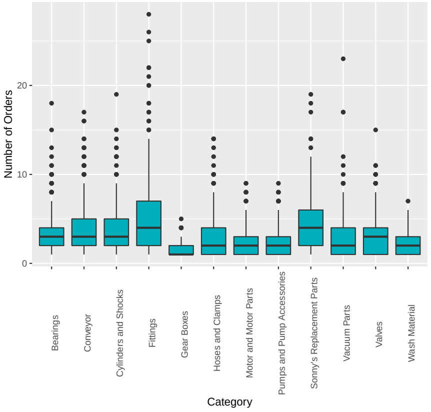
amounts but less frequently. It is worth asking how to make their experience even better and anticipate their need for optimization.

4. Groups of customers with fewer outliers may represent more re-occurring purchasing behavior. Variation is smaller in such groups and categories. This makes them more predictable and more manageable for the account executives. Such a pattern creates a room for automation and optimization.

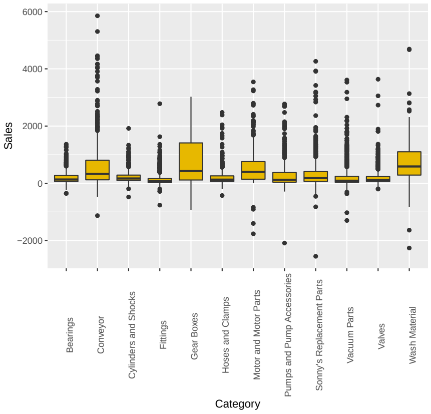
Customer Group 1



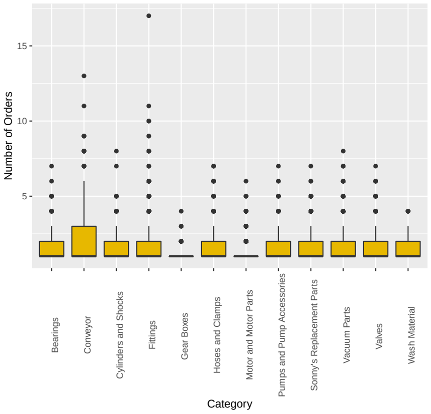
Customer Group 1



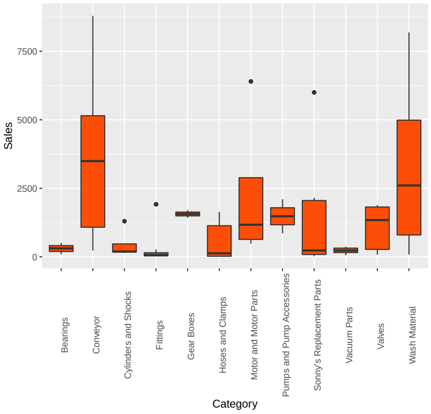
Customer Group 2



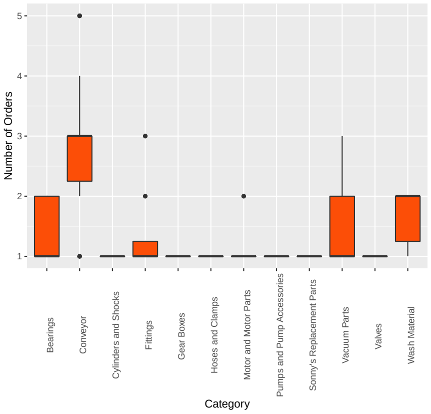
Customer Group 2



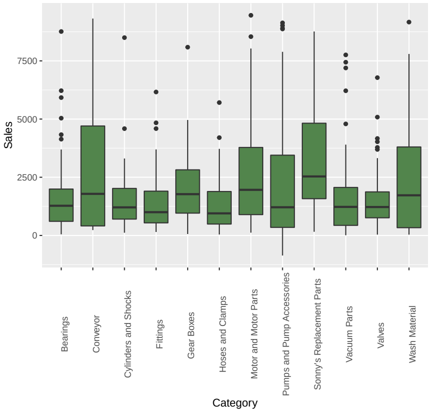
Customer Group 3



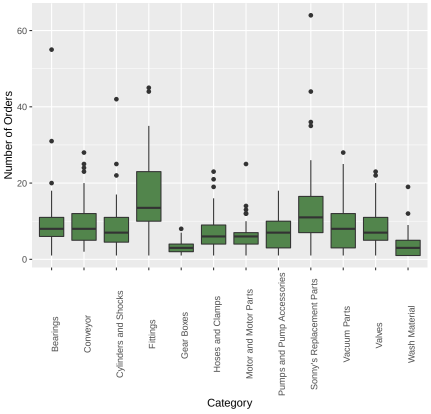
Customer Group 3



Customer Group 4



Customer Group 1


Number of Orders
Bearings
Conveyor
Cylinders and Shocks
Fittings
Gear Boxes
Hoses and Clamps
Motor and Motor Parts
Pumps and Pump Accessories
Sonny's Replacement Parts
Vacuum Parts
Valves
Wash Material
Category