K-means & K-medoid clustering in product segmentation: ASDS 6303 Final Project

Submitted by Utkarsh Pant (1002170893)

2024-05-08

Note regarding seed: please note that my student ID ends with "0893", which should be the seed wherever required. However, leading 0's are not allowed in the seed (0893 = 893, in essence). Hence, the first 2 digits have been swapped to get the seed "8093".

Loading the dataset

```
product_data = read_excel('./dataset/sku_data.xlsx')
kable(head(product_data),
      booktabs = TRUE,
      format = "latex",
      caption = "Dataset head") %>% kable_styling(latex_options = "hold_position")
```

Table 1: Dataset head

ID	Unitprice	Expire date	Outbound number	Total outbound	Pal grossweight	Pal height	Units per pal
1	0.058	547	9	2441	105.60	1.56	1920
2	0.954	547	0	0	207.68	1.00	384
3	2.385	547	12	23	165.78	1.02	108
4	5.100	547	0	0	221.04	1.05	72
5	0.000	547	0	0	0.00	0.00	0
6	1.110	547	1	1	207.68	1.00	384

```
summary(product_data)
```

```
##        ID           Unitprice         Expire date    Outbound number
##  Min.   :   1.0   Min.   :  0.000   Min.   :  0.0   Min.   :   0
##  1st Qu.: 570.5   1st Qu.:  0.000   1st Qu.:365.0   1st Qu.:   0
##  Median :1140.0   Median :  1.294   Median :547.0   Median :   1
##  Mean   :1140.0   Mean   :  4.269   Mean   :410.4   Mean   : 236
##  3rd Qu.:1709.5   3rd Qu.:  4.545   3rd Qu.:547.0   3rd Qu.:  45
##  Max.   :2279.0   Max.   :518.592   Max.   :734.0   Max.   :6325
##  Total outbound     Pal grossweight   Pal height       Units per pal
##  Min.   :    0.0   Min.   :  0.0   Min.   :0.0000   Min.   :     0.0
##  1st Qu.:    0.0   1st Qu.: 60.0   1st Qu.:0.0000   1st Qu.:    32.0
##  Median :    3.0   Median :167.7   Median :0.8400   Median :   108.0
##  Mean   :  731.7   Mean   :192.9   Mean   :0.6728   Mean   :   755.6
##  3rd Qu.:  419.5   3rd Qu.:277.6   3rd Qu.:1.0200   3rd Qu.:   384.0
##  Max.   :26411.0   Max.   :907.2   Max.   :2.1600   Max.   :200000.0
```

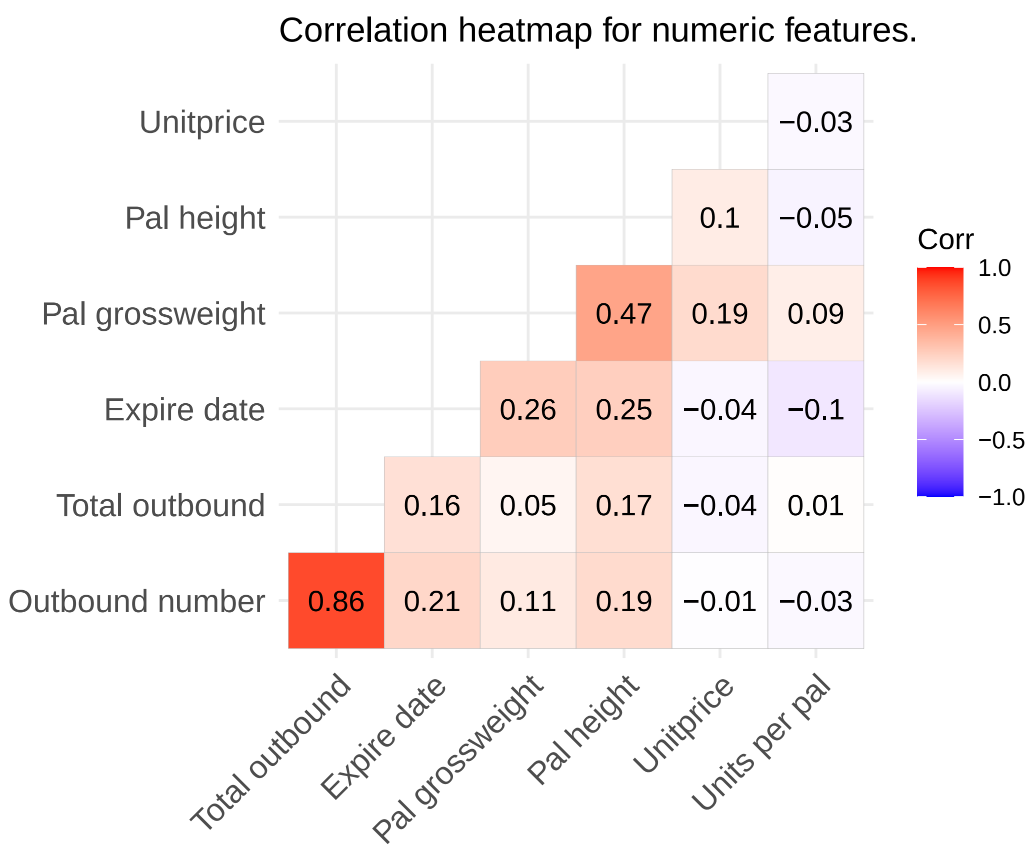
```r
product_data <- select(product_data, -c("ID"))
```

Checking correlation

```r
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 4.3.2
```

```r
correlation = cor(product_data)
ggcorrplot(correlation, hc.order = TRUE, type = "lower",
   lab = TRUE, title = "Correlation heatmap for numeric features.")
```



Let's only consider the `Outbound number` and `Total outbound` features in our dataset to perform the clustering, due to high correlation among them.

```r
product_subset <- select(product_data, c("Outbound number", "Total outbound"))
```

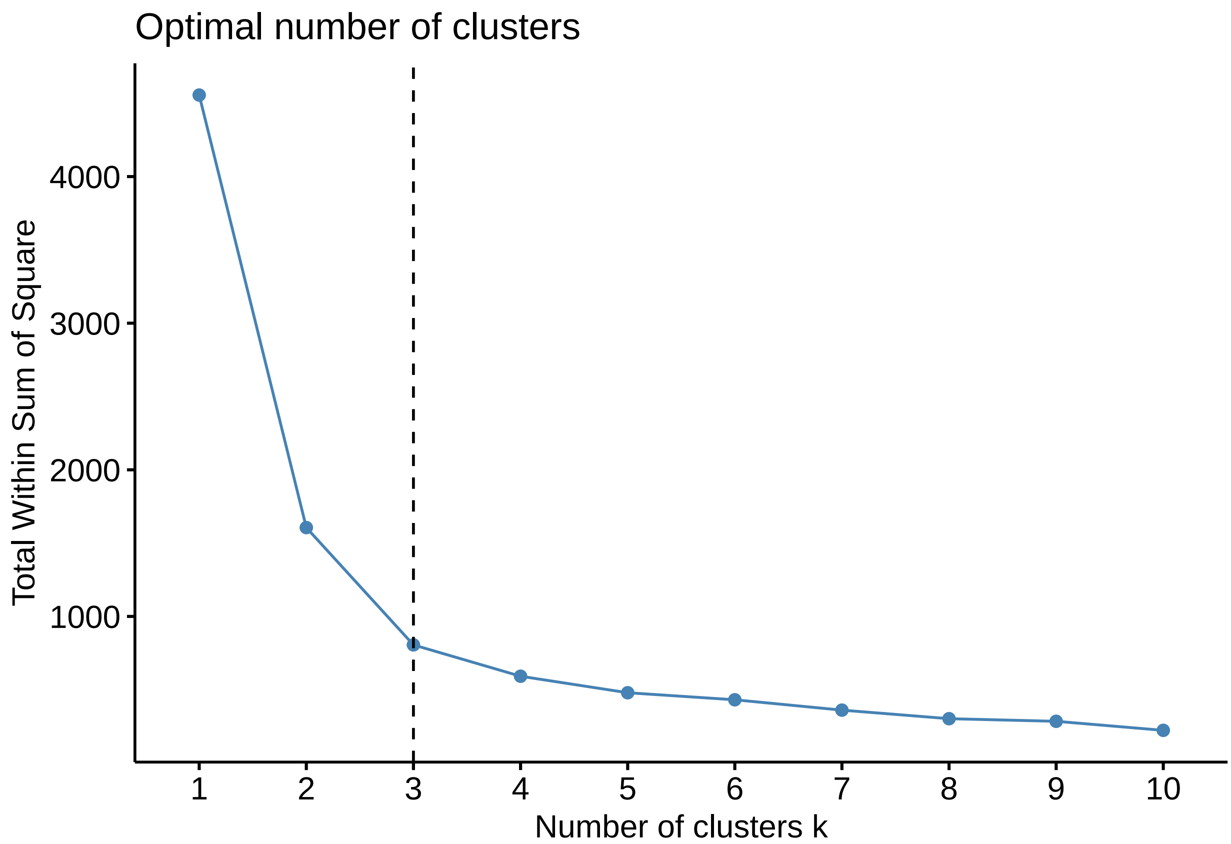
Scaling data

```r
product_subset_scaled = scale(product_subset)
```

K-means clustering

Checking a scree-plot for the ideal number of clusters, we see:

```r
fviz_nbclust(product_subset_scaled, kmeans, method = "wss") +
geom_vline(xintercept = 3, linetype = 2)
```




```r
set.seed(8093)
model.kmeans <- kmeans(product_subset_scaled, nstart = 20, centers = 3)
print(model.kmeans)
```

```
## K-means clustering with 3 clusters of sizes 34, 173, 2072
##
## Cluster means:
##   Outbound number Total outbound
## 1        5.679717      6.3389854
## 2        1.971275      1.4644906
## 3       -0.257790     -0.2262946
##
## Clustering vector:
##    [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##   [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##   [75] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [112] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [149] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [186] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [223] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [260] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [297] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [334] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

```
##  [371] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [408] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [445] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [482] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [519] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [556] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [593] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [630] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [667] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [704] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [741] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [778] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [815] 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [852] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [889] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [926] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [963] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1000] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1037] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1074] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1111] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1148] 3 3 3 3 3 3 3 3 3 3 3 3 3 1 2 3 3 3 1 3 2 3 2 3 3 3 2 3 3 3 3 3 3 1 2 2 2
## [1185] 2 1 2 2 2 1 1 2 1 1 1 1 1 2 3 2 2 2 1 1 1 2 3 2 1 3 2 1 2 2 1 2 3 2 2 3 3
## [1222] 3 3 3 3 3 2 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 3 2 3 3 3 3 2 3 3 3 3 3 3 2 3 2
## [1259] 3 3 3 3 3 3 3 3 3 3 3 3 1 2 3 3 2 3 3 3 3 2 1 2 3 3 3 3 3 3 2 2 3 3 3 3 3
## [1296] 3 3 3 3 3 3 1 3 3 3 3 3 3 1 3 2 2 3 3 3 1 2 2 2 3 1 2 3 3 3 3 3 3 3 3 3 3
## [1333] 3 3 3 3 3 3 2 2 3 3 2 3 2 3 3 3 3 2 3 1 2 1 2 2 3 2 2 2 2 3 2 2 2 1 2 2 2
## [1370] 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 2 3 3 3 2 3 3 2 3 3 2 1 2 2 2
## [1407] 2 2 2 3 3 3 3 2 3 2 3 3 3 2 2 3 3 3 2 3 3 3 3 2 2 1 2 2 3 2 2 3 3 3 2 2 3
## [1444] 3 3 3 2 3 2 2 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 2 3 3 3 3 2 3 3 3 2 2 3 2 2 3
## [1481] 3 3 2 1 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 2 3 3 3 3 3
## [1518] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1555] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 3 3 3 3 3 2 2
## [1592] 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 1 2 2
## [1629] 3 2 2 2 2 2 2 2 1 2 3 3 2 2 3 2 2 2 2 3 3 2 2 2 1 1 2 2 2 2 3 2 3 2 2 3 3
## [1666] 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3
## [1703] 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1740] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3
## [1777] 3 3 3 3 3 3 3 3 3 2 3 3 2 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 2 3 3 3 3 2 3 3 3
## [1814] 3 3 3 2 3 3 2 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3
## [1851] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 1 3 3 3 2 3 3 3 3
## [1888] 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1925] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1962] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [1999] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 2 3 3 3 3 3 3 2 2 3 2 3 3 3
## [2036] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3
## [2073] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [2110] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [2147] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [2184] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [2221] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [2258] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
```

Table 2: Average cluster characteristics for K-Means clustering

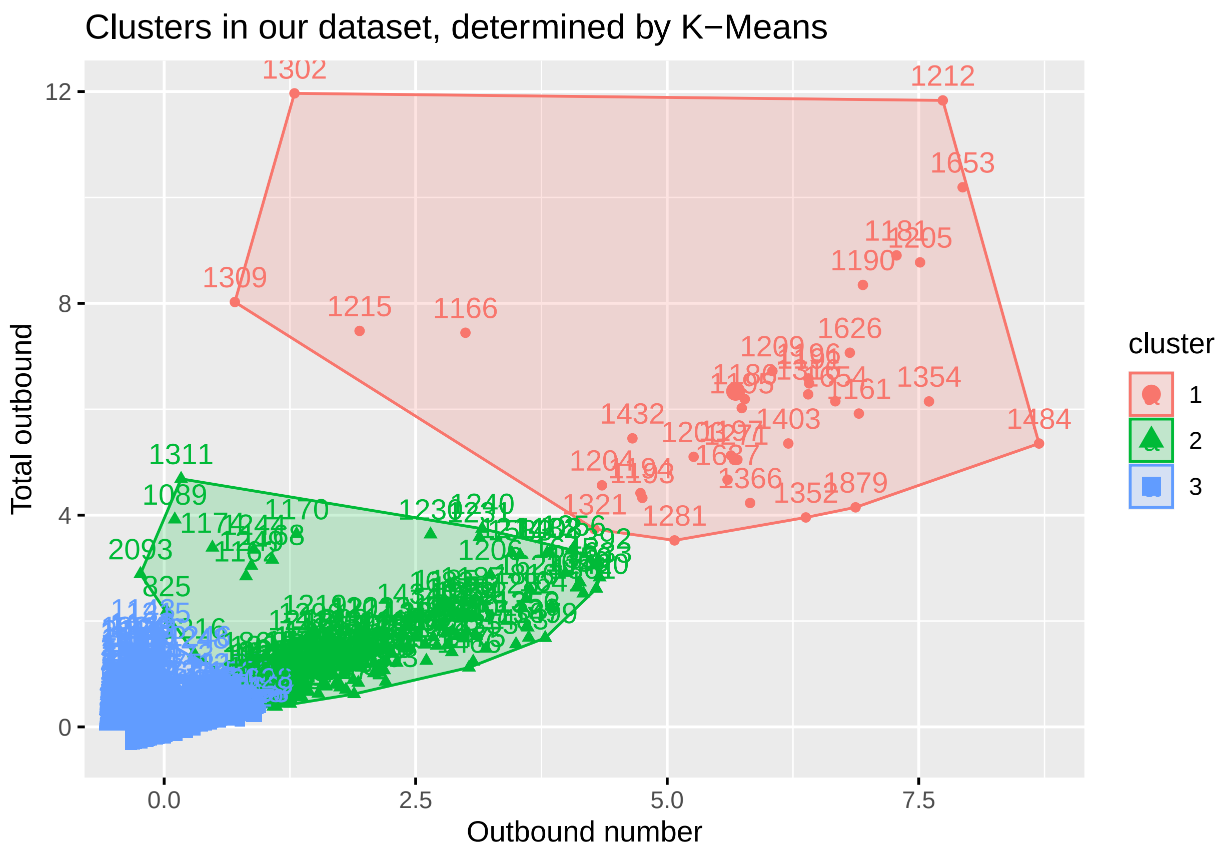
cluster	Unitprice	Expire date	Outbound number	Total outbound	Pal grossweight	Pal height	Units per pal
1	1.903088	568.5882	4213.0882	14335.3529	207.2529	0.9700000	310.3529
2	3.590896	561.8902	1616.3237	3874.5416	248.1189	1.0087283	250.9480
3	4.364883	395.1245	55.4638	246.0661	188.0976	0.6398726	805.0014

```
## [1] 260.0769 306.0841 239.7081
##  (between_SS / total_SS =  82.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Aggregating cluster characteristics

```
kable(aggregate(product_data, by=list(cluster=model.kmeans$cluster), mean),
      format = "latex",
            caption = "Average cluster characteristics for K-Means clustering",
      booktabs = TRUE) %>% kable_styling(position="center")
```

```
fviz_cluster(model.kmeans, product_subset_scaled, main = "Clusters in our dataset, determined by K-Means
```



Silhouette scores

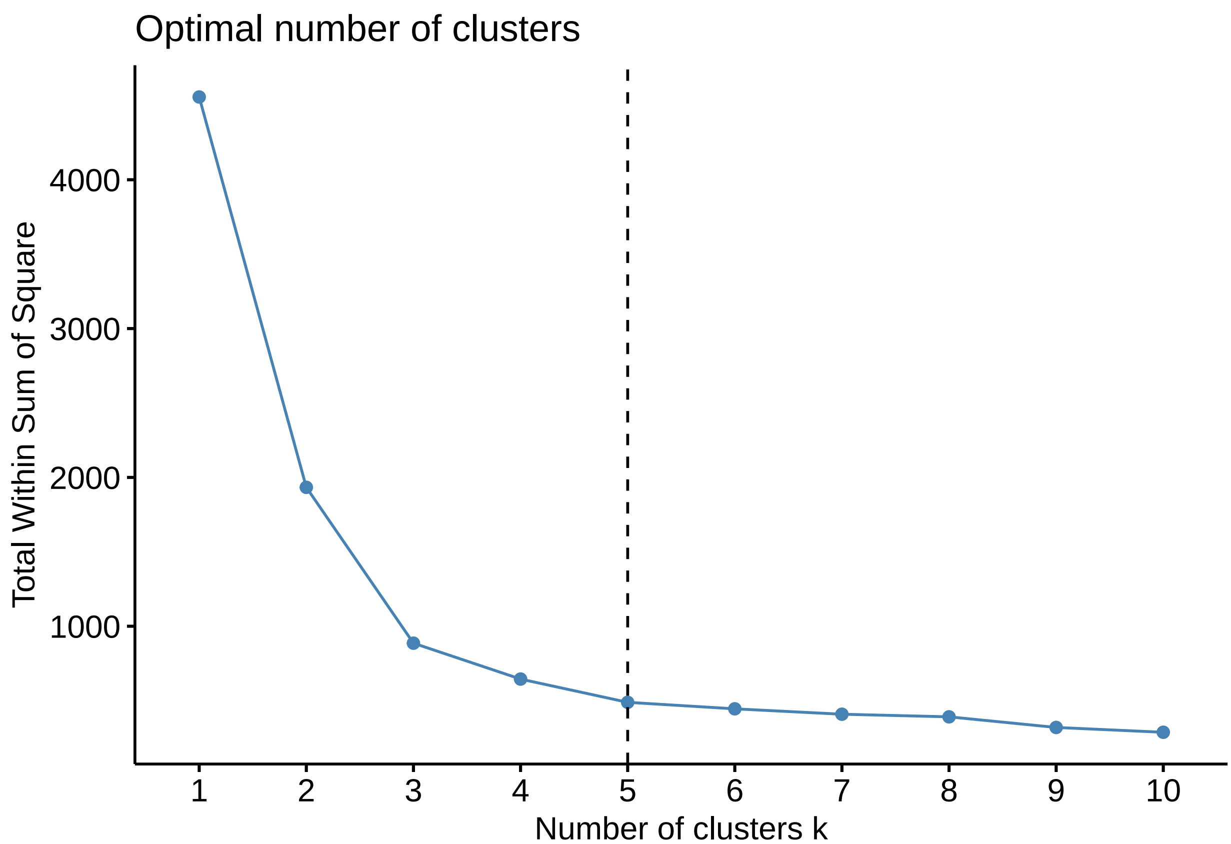
```
silhouette_score.k_means <- silhouette(model.kmeans$cluster, dist(product_data))
silhouette_score.k_means <- mean(silhouette_score.k_means[, 'sil_width'])
silhouette_score.k_means
```

```
## [1] 0.6174801
```

K-medoid clustering

Visualizing the ideal number of clusters for `pam` (k-medoid clustering) using `fvz_nbclust`:

```
fviz_nbclust(product_subset_scaled, pam, method = "wss") +
  geom_vline(xintercept = 5, linetype = 2)
```



It appears that k-medoid clustering for the same product-data does best with 5 clusters. Performing clustering with 5 clusters:

```
set.seed(8093)
model.k_medoid <- pam(product_subset_scaled, k = 5)
model.k_medoid
```

```
## Medoids:
##        ID Outbound number Total outbound
## [1,] 1325       0.1499844      0.1320107
## [2,]    2      -0.3369980     -0.3409557
## [3,] 1980       1.2710435      0.7722628
```

```
## [4,] 1195       5.7409990      6.0210248
## [5,] 1364       3.0133265      2.0867832
## Clustering vector:
##    [1] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2
##   [38] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2
##   [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [112] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [149] 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [186] 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [223] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [260] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [408] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [445] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [482] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [519] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [556] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [593] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [630] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [667] 2 2 2 1 2 2 2 1 1 2 2 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2
##  [704] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [741] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [778] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [815] 2 2 2 1 2 2 2 2 2 1 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [852] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [889] 2 2 2 2 2 2 2 1 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [926] 2 2 2 2 1 2 2 1 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 1 1 2 2 2
##  [963] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2
## [1000] 1 2 2 2 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [1037] 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [1074] 2 2 2 2 1 1 2 2 1 2 2 2 1 1 2 3 2 1 2 1 1 2 2 1 2 2 2 2 1 2 2 2 2 2 1 2 1
## [1111] 2 1 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 1 1 2 1 2 1 1 2 2 1 2 1 2 1 1 2
## [1148] 1 2 2 2 2 1 2 1 2 2 2 2 1 4 3 2 2 2 4 2 5 1 5 2 1 1 3 1 2 2 1 2 2 4 5 3 3
## [1185] 5 4 5 5 3 4 4 5 4 4 4 4 4 3 3 3 3 5 4 4 4 5 1 3 4 3 5 4 5 5 4 3 2 3 3 2 3
## [1222] 3 2 2 1 1 3 1 1 5 5 1 1 2 1 3 1 3 1 5 2 2 1 5 2 3 2 3 3 2 2 2 1 2 2 5 2 3
## [1259] 1 1 2 3 2 1 2 2 2 1 2 1 4 5 2 2 3 2 2 2 2 5 5 5 2 1 2 2 1 1 3 3 1 2 2 2 2
## [1296] 2 2 2 1 2 2 4 1 2 1 1 2 1 4 2 5 3 2 2 1 4 3 3 5 2 5 5 2 2 1 2 2 2 2 2 2 1
## [1333] 2 2 2 2 2 2 3 3 2 2 3 2 3 2 2 2 2 3 3 4 5 4 5 5 1 3 5 5 5 2 3 5 3 4 5 3 5
## [1370] 2 2 1 1 2 1 2 2 2 2 1 2 1 3 2 2 2 2 2 2 5 2 5 1 3 1 3 1 3 5 2 1 3 4 3 3 3
## [1407] 3 5 3 3 2 1 1 3 2 3 1 2 1 5 5 2 2 2 3 3 2 2 2 3 3 4 3 5 2 3 3 2 1 1 3 3 1
## [1444] 1 1 1 3 1 3 3 2 2 1 1 1 1 1 2 2 2 2 2 3 2 1 5 2 2 2 1 3 1 1 2 5 3 2 3 5 1
## [1481] 1 2 5 4 1 1 2 2 2 2 1 3 2 2 2 2 2 2 2 1 2 2 1 2 2 2 1 3 2 3 2 3 1 1 2 2 2
## [1518] 2 2 2 2 1 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2
## [1555] 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 2 2 3 2 3 5 3 1 1 2 2 2 3 3
## [1592] 3 2 1 2 2 2 2 2 1 2 2 2 2 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 5 3 1 4 3 3
## [1629] 1 3 3 3 5 5 5 5 4 5 1 1 5 3 1 3 5 3 3 2 2 5 3 3 4 4 5 5 3 3 1 3 2 3 3 2 3
## [1666] 2 3 2 1 1 2 2 2 1 5 5 1 1 2 2 1 5 2 2 2 2 3 1 1 2 2 2 2 2 2 2 3 1 2 2 1 2
## [1703] 2 1 1 2 2 3 2 2 1 2 3 2 2 1 1 2 2 2 2 1 2 3 1 2 2 2 2 2 2 2 1 2 1 2 2 2 2
## [1740] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 1 3 2 1 1 2 2 1 1 3 2 1 1 2 2 1 1
## [1777] 2 2 2 2 2 1 3 1 2 3 2 2 3 1 3 2 2 3 1 1 3 1 1 2 2 1 3 1 3 2 1 2 3 5 2 2 1
## [1814] 1 2 2 3 1 2 3 2 1 2 2 1 2 2 5 2 1 2 2 1 2 2 2 2 2 2 2 2 2 1 3 3 2 1 2 2 2
## [1851] 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 3 2 1 2 2 2 2 2 2 1 4 1 2 1 3 2 2 1 2
```

Table 3: Average cluster characteristics for K-Medoid clustering

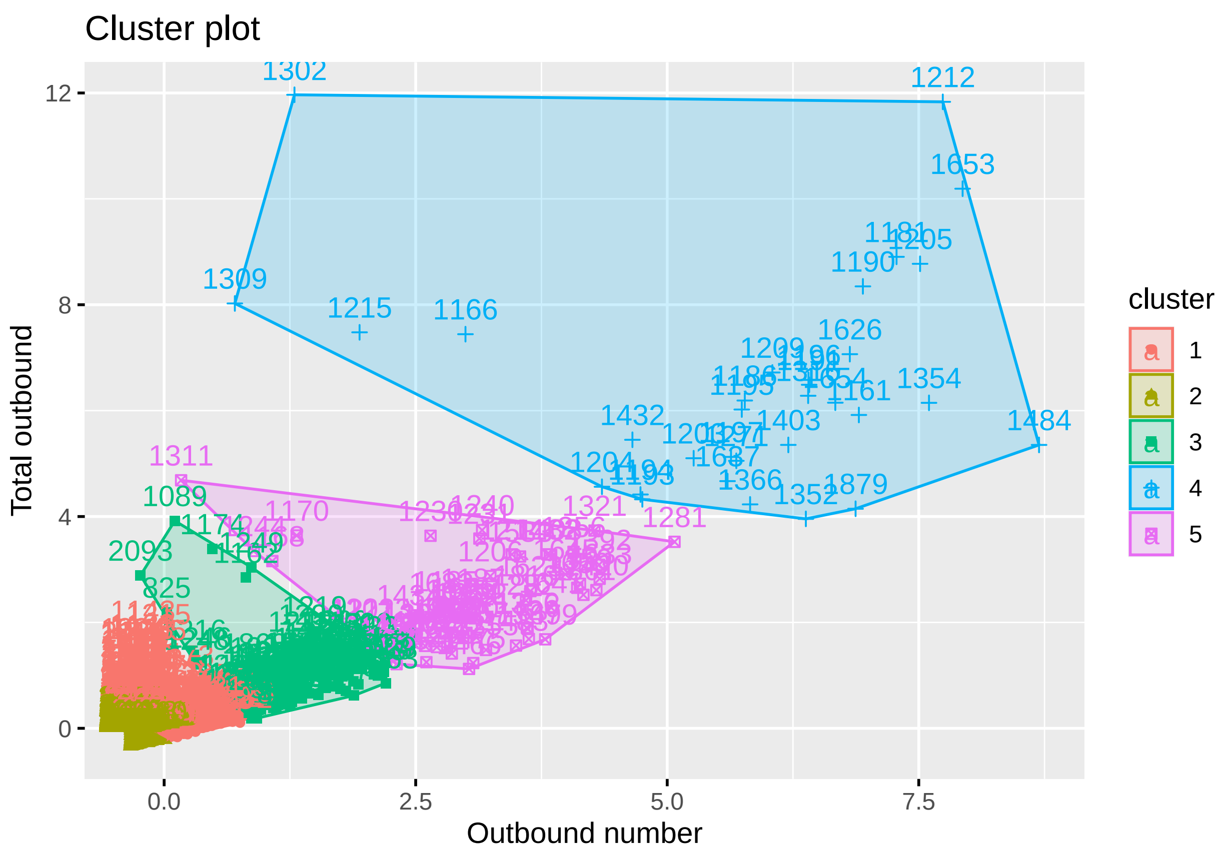
cluster	Unitprice	Expire date	Outbound number	Total outbound	Pal grossweight	Pal height	Units per pal
1	4.079924	468.8511	273.70922	1317.3794	228.9536	0.9068972	1633.4681
2	4.402425	381.2756	11.01358	55.2077	180.2633	0.5913531	680.7040
3	4.216886	552.6591	1129.59091	2706.6818	258.6896	1.0346970	237.7652
4	1.944063	569.9375	4257.06250	14698.8750	209.1037	0.9628125	312.1250
5	2.750076	577.5455	2360.77273	5618.9045	239.1029	0.9886364	259.3030

```
## [1888] 1 1 2 2 1 2 2 3 1 2 1 2 1 1 1 2 1 1 1 1 2 2 2 1 1 1 2 2 1 2 2 2 2 2 2 1 2
## [1925] 2 1 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2
## [1962] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 5 5 2 3 1 1 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2
## [1999] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 1 2 5 2 2 2 2 1 3 3 5 1 3 2 1 2
## [2036] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 1 2 2 2 1 2 1 2 2
## [2073] 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 3 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2
## [2110] 2 2 2 1 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2
## [2147] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [2184] 1 2 2 2 2 1 2 2 2 2 2 2 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [2221] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 1 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2
## [2258] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
## Objective function:
##     build      swap
## 0.1805800 0.1718847
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

Aggregating cluster characteristics

```r
kable(aggregate(product_data, by=list(cluster=model.k_medoid$cluster), mean),
      format = "latex",
      caption = "Average cluster characteristics for K-Medoid clustering",
      booktabs = TRUE) %>% kable_styling(position="center")
```

```r
fviz_cluster(model.k_medoid, product_subset_scaled)
```



Silhouette scores

```
silhouette_score.k_medoid <- silhouette(model.k_medoid$cluster, dist(product_data))
silhouette_score.k_medoid <- mean(silhouette_score.k_medoid[, 'sil_width'])
silhouette_score.k_medoid
```

```
## [1] 0.4574614
```

Comparing clusters

From our analysis and calculation of silhouette scores, we can see that K-means performs better with a silhouette score of 0.6174801, while K-Medoid performs rather poor clustering with an average silhouette score of 0.4574614.

Since these are clustering algorithms, the quality of clustering is measured using metrics like the silhouette scores, gap statistics, etc. Metrics like accuracy, AUC, etc. cannot be determined since there is no "prediction" of target classes being performed in unsupervised clustering!

Conclusion

In conclusion, we can see that the quality of clustering has room to improve. For this, we could consider better feature selection to cluster on the basis of; we might select such features based on high correlation, domain-knowledge and the "relevance" of features to the problem statement and the aspects of the data we are interested in.