MSCI 719: Operations Analytics

Assignment 6: End to End Analytics for an Online Retailer, Rue La La: Part I (Demand Forecasting)

Student: Advait Shah

Faculty: Engineering

Department: Management Sciences

Instructor: Prof. Hossein Abouee Mehrizi

Contents

1. Demand estimation

In the class, we discussed how Kate was able to use clustering to estimate the demand for sold-out items. Considering what she did, answer the following questions.

1.1. What is the difference between total sales and demand? Why should the demand for the sold-out items be estimated?

Total sales of particular SKU of the fashion style item is the minimum of the demand and inventory available. So, sometimes demand could be higher but due to limited inventory available, item gets sold out early, and in this case actually demand and sales will not be the same. Therefore, knowing the expected demand for such sold-out items precisely could prepare us to increase our inventory or prices for such product suitably and increase our sales or revenues from such products in the future.

1.2. Why did Kate use the percentage of sales instead of the absolute value to cluster the items?

As we wanted to understand demand pattern of different items with respect to time, and compare their demand curve or distribution, we had done clustering. So, if we had done the clustering just based on absolute sales value, we would not have got the meaningful clusters, as some items might be having more inventory quantities compared to other and they might skew our clustering, as clustering based on Euclidean distance, like in K-means clustering, is highly dependent on the scales of the features. Hence, generally it is always better to normalize or standardize our data before clustering, and here we are normalizing with respect to percentage of total sales for each our, and this makes our data for each feature (hourly sales data) kind of normalized to compare and study relative demand of products over the time. So, this is the main reason Kate used the percentage of sales instead of the absolute value to cluster the items.

2. Clustering

Consider the given data set.

2.1. Perform a k-means clustering for k = 2, 3, 4, 5.

We have to create clusters for the non-stockout items. And this will later help us in measuring distance of stockout items from centroids to predict their lost demand.

We will use R programming for this assignment of creating clusters and predicting demands of stockout items. Therefore, I have divided dataset into stockout and non-stockout items in R and run clustering as shown below.

Read the data:

```
mydata = read.csv("Data_Part1.csv")
colnames(mydata)
```

```
##  [1] "Item."       "Total.sales" "hour.1"      "hour.2"      "hour.3"
##  [6] "hour.4"      "hour.5"      "hour.6"      "hour.7"      "hour.8"
## [11] "hour.9"      "hour.10"     "hour.11"     "hour.12"     "hour.13"
## [16] "hour.14"     "hour.15"     "hour.16"     "hour.17"     "hour.18"
## [21] "hour.19"     "hour.20"     "hour.21"     "hour.22"     "hour.23"
## [26] "hour.24"
```

```
nrow(mydata)
```

```
## [1] 2446
```

```
dim(mydata)
```

```
## [1] 2446   26
```

Calculate the total inventory used (share of sale)

```
Total = rowSums(mydata[,3:26])
head(Total)
```

```
## [1] 0.933 0.938 0.996 0.842 1.000 0.998
```

Adding "Total" to main table

```
mydata = cbind(mydata, Total)
head(mydata)
```

Filter to find stockouts and nonstockouts

```
stockouts = mydata[(mydata$Total>=1.00 & mydata$hour.24==0),]
nonstockouts = mydata[!(mydata$Total>=1.00 & mydata$hour.24==0),]
```

Calculate the cumulative stockouts (to help us find the stockout hour)

```
stockouts_cumulative = stockouts
for (i in 4:26) {
  stockouts_cumulative[,i] = stockouts_cumulative[,i] +
stockouts_cumulative[,i-1]
}
```

Finding the index of 1 (100%) in the stockouts cumulative table

```
stockouts_time = vector(mode="numeric")
for (i in 1:nrow(stockouts_cumulative)) {
  stockouts_time[i] = match(max(stockouts_cumulative[i,3:26]),stockouts_cu
mulative[i,3:26])
}
stockouts_time
```

Adding Stockout time to stockouts table:

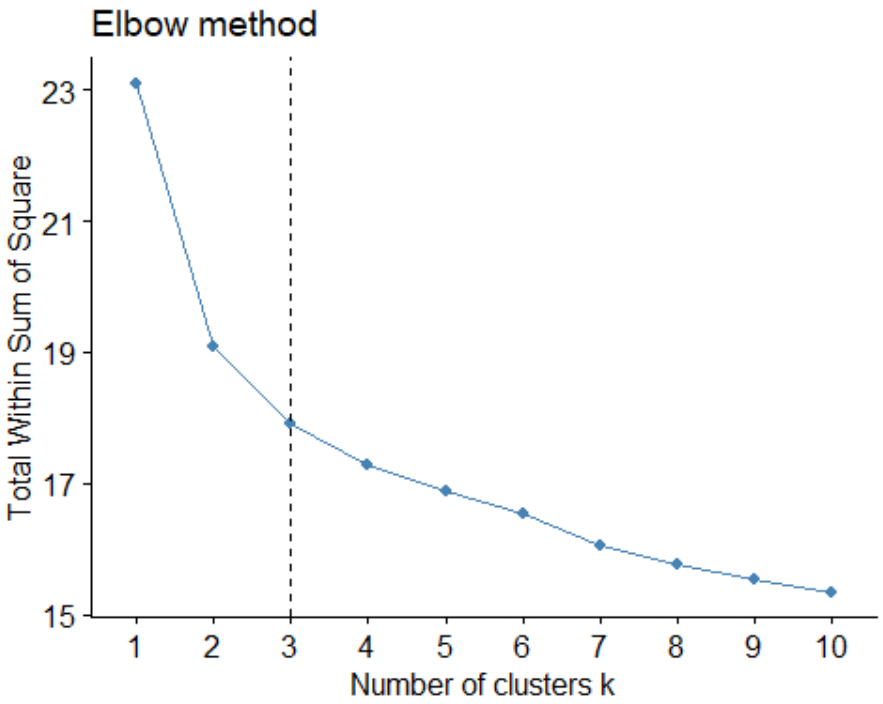
```
stockouts = cbind(stockouts, stockouts_time)
head(stockouts)
```

```
##    Item. Total.sales hour.1 hour.2 hour.3 hour.4 hour.5 hour.6 hour.7 hour.8
## 5      5        3115  0.116  0.032  0.048  0.054  0.035  0.049  0.023  0.075
## 12    12        4943  0.144  0.060  0.038  0.019  0.076  0.072  0.008  0.000
## 14    14        4783  0.132  0.033  0.002  0.083  0.033  0.068  0.057  0.006
## 17    17        6294  0.114  0.060  0.031  0.078  0.082  0.026  0.081  0.022
## 19    19        4716  0.144  0.046  0.048  0.004  0.030  0.056  0.036  0.053
## 36    36        3206  0.133  0.003  0.080  0.025  0.005  0.056  0.073  0.008
##    hour.9 hour.10 hour.11 hour.12 hour.13 hour.14 hour.15 hour.16 hour.17
## 5   0.063   0.082   0.064   0.037   0.003   0.134   0.038   0.048   0.039
## 12  0.079   0.081   0.063   0.076   0.076   0.055   0.028   0.036   0.021
## 14  0.079   0.049   0.069   0.061   0.061   0.066   0.039   0.030   0.043
## 17  0.054   0.027   0.029   0.047   0.005   0.139   0.023   0.044   0.041
## 19  0.076   0.048   0.064   0.011   0.082   0.075   0.049   0.029   0.034
## 36  0.029   0.067   0.082   0.045   0.078   0.316   0.000   0.000   0.000
##    hour.18 hour.19 hour.20 hour.21 hour.22 hour.23 hour.24 Total stockouts_time
## 5    0.044   0.012   0.004   0.000   0.000       0       0 1.000             20
## 12   0.028   0.023   0.019   0.000   0.000       0       0 1.002             20
## 14   0.043   0.004   0.011   0.031   0.000       0       0 1.000             21
## 17   0.035   0.028   0.009   0.010   0.015       0       0 1.000             22
## 19   0.039   0.013   0.028   0.036   0.000       0       0 1.001             21
## 36   0.000   0.000   0.000   0.000   0.000       0       0 1.000             14
```

Clustering:

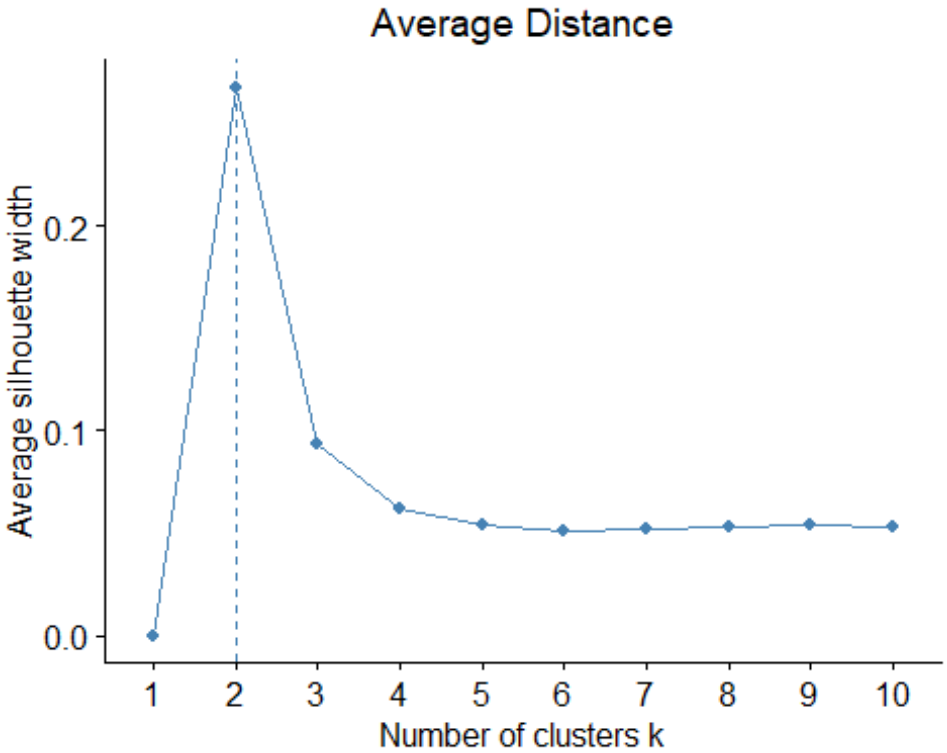
- Within Sum of Squares and Elbow Method:

```
# Install "factoextra" and "NbClust"
#install.packages("factoextra")
#install.packages("NbClust")
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.1.3

library(factoextra)

## Warning: package 'factoextra' was built under R version 4.1.3

## Welcome! Want to learn more? See two factoextra-related books at https:
//goo.gl/ve3WBa

library(NbClust)

## Warning: package 'NbClust' was built under R version 4.1.3

fviz_nbclust(nonstockouts[,3:26], kmeans, method = "wss",iter.max=50) +
  geom_vline(xintercept = 3, linetype =2) +
  labs(title = "Elbow method")
```



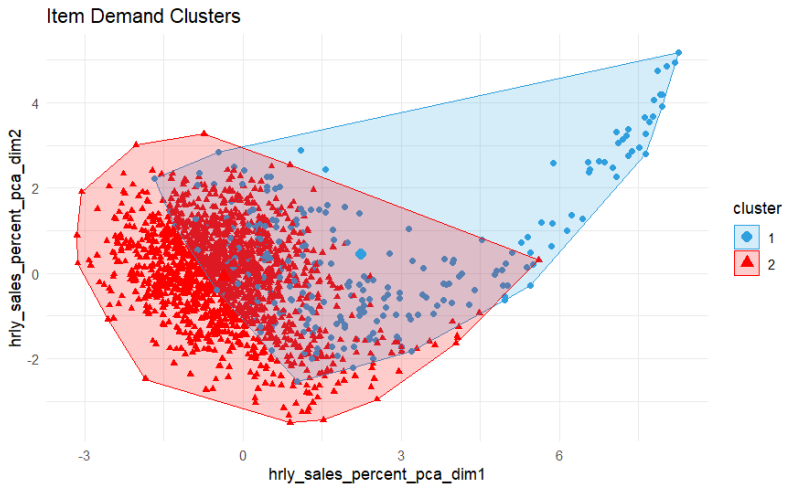
- Silhouette Method:

```
fviz_nbclust(nonstockouts[,3:26], kmeans, method = "silhouette",iter.max=50) +
  ggtitle("Average Distance") +
  theme(plot.title = element_text(hjust = 0.5))
```



We can also visually see these clusters and segregation of products based on their sales features, as below:

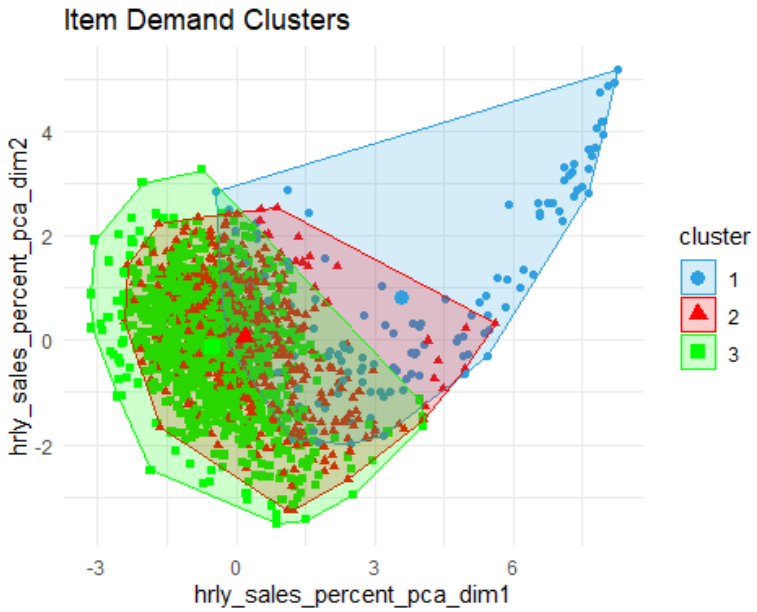
Clustering with K=2:



Clustering with K=3:

Sample code to create visualization for k=3:

```
clusters3 = kmeans(nonstockouts[,3:26], 3, nstart = 1,iter.max=50)
fviz_cluster(clusters3,data=nonstockouts[,3:26],
             palette= c("#2E9FDF","#FF0000","#00FF00"),
             geom = "point",
             ellipse.type = "convex",
             xlab= "hrly_sales_percent_pca_dim1",
             ylab= "hrly_sales_percent_pca_dim2",
             main= "Item Demand Clusters")+
  theme_minimal()
```



Clustering with K=4:

Clustering with K=5:

2.2. Plot the total within the sum of squares and the average distance of points in different clusters.

Let's now plot the total within and between clusters sum of squares errors plots by following process:

Clustering - K-means

```
totwithinss = list()
betweenss = list()
for (k in 2:5){
  clusters = kmeans(nonstockouts[,3:26], k, nstart = 1,iter.max=50)
  totwithinss[k] = clusters$tot.withinss

  betweenss[k] = clusters$betweenss

}

plot(c(2:5),totwithinss[2:5], type = "b", xlab= "no. of clusters", ylab= "
total within sum of squares")
```

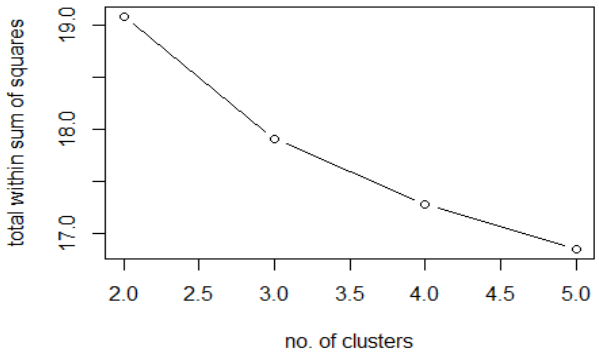


Fig: Total of within clusters sum of squares vs. no. of clusters

```
plot(c(2:5),betweenss[2:5], type = "b", xlab= "no. of clusters", ylab= "to
tal between sum of squares")
```

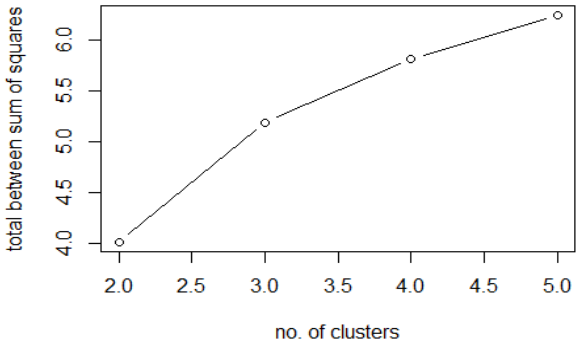


Fig: between clusters sum of squares vs. no. of clusters

2.3. Estimate the demand for sold-out items using clustering as discussed in class. Use a proper visualization method to examine the behaviour of your estimation for different values of k.

This code will help us to find the Lost Sales and Calculate True demand:

```
k = 2
true_demand_2 = vector(mode = "numeric")

clusters = kmeans(nonstockouts[,3:26], k, nstart = 1,iter.max=50)
centroids = clusters$centers

determined_cluster = vector()
lost_percentage = vector()


for (i in 1:nrow(stockouts)) {
  p = stockouts[i, 3:(stockouts[i,28]+2)]
```

```
  rownames(p) = "p"
  k_1 = centroids[1,1:stockouts[i,28]]
  k_2 = centroids[2,1:stockouts[i,28]]
  mat = rbind(k_1, k_2, p)

  dis_mat = as.matrix(dist(mat, method = "euclidean"))
  determined_cluster[i] = match(min(dis_mat[3,1:2]), dis_mat[3,1:2])
  lost_percentage[i] = sum(centroids[determined_cluster[i], (stockouts[i,2
8] + 1):24])

  true_demand_2[i] = stockouts[i,2]/(1-lost_percentage[i])
}

stockouts=cbind(stockouts,true_demand_2)

k = 3
true_demand_3 = vector(mode = "numeric")

clusters = kmeans(nonstockouts[,3:26], k, nstart = 1,iter.max=50)
centroids = clusters$centers

determined_cluster = vector()
lost_percentage = vector()


for (i in 1:nrow(stockouts)) {
  p = stockouts[i, 3:(stockouts[i,28]+2)]
  rownames(p) = "p"
  k_1 = centroids[1,1:stockouts[i,28]]
  k_2 = centroids[2,1:stockouts[i,28]]
  k_3 = centroids[3,1:stockouts[i,28]]
  mat = rbind(k_1, k_2, k_3, p)
  dis_mat = as.matrix(dist(mat, method = "euclidean"))
  determined_cluster[i] = match(min(dis_mat[4,1:3]), dis_mat[4,1:3])
  lost_percentage[i] = sum(centroids[determined_cluster[i], (stockouts[i,2
8] + 1):24])
  true_demand_3[i] = stockouts[i,2]/(1-lost_percentage[i])
}

stockouts=cbind(stockouts,true_demand_3)

k = 4
true_demand_4 = vector(mode = "numeric")

clusters = kmeans(nonstockouts[,3:26], k, nstart = 1,iter.max=50)
centroids = clusters$centers

determined_cluster = vector()
lost_percentage = vector()


for (i in 1:nrow(stockouts)) {
  p = stockouts[i, 3:(stockouts[i,28]+2)]
  rownames(p) = "p"
```

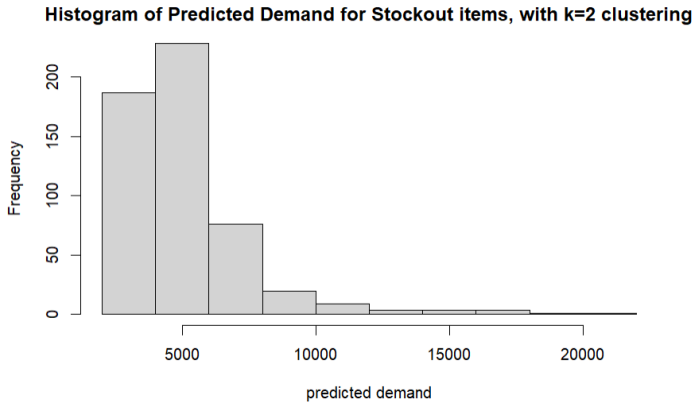
```
  k_1 = centroids[1,1:stockouts[i,28]]
  k_2 = centroids[2,1:stockouts[i,28]]
  k_3 = centroids[3,1:stockouts[i,28]]
  k_4 = centroids[4,1:stockouts[i,28]]
  mat = rbind(k_1, k_2, k_3,k_4, p)
  dis_mat = as.matrix(dist(mat, method = "euclidean"))
  determined_cluster[i] = match(min(dis_mat[5,1:4]), dis_mat[5,1:4])
  lost_percentage[i] = sum(centroids[determined_cluster[i], (stockouts[i,2
8] + 1):24])
  true_demand_4[i] = stockouts[i,2]/(1-lost_percentage[i])
}

stockouts=cbind(stockouts,true_demand_4)

k = 5
true_demand_5 = vector(mode = "numeric")

clusters = kmeans(nonstockouts[,3:26], k, nstart = 1,iter.max=50)
centroids = clusters$centers

determined_cluster = vector()
lost_percentage = vector()

for (i in 1:nrow(stockouts)) {
  p = stockouts[i, 3:(stockouts[i,28]+2)]
  rownames(p) = "p"
  k_1 = centroids[1,1:stockouts[i,28]]
  k_2 = centroids[2,1:stockouts[i,28]]
  k_3 = centroids[3,1:stockouts[i,28]]
  k_4 = centroids[4,1:stockouts[i,28]]
  k_5 = centroids[5,1:stockouts[i,28]]
  mat = rbind(k_1, k_2, k_3,k_4,k_5, p)
  dis_mat = as.matrix(dist(mat, method = "euclidean"))
  determined_cluster[i] = match(min(dis_mat[6,1:5]), dis_mat[6,1:5])
  lost_percentage[i] = sum(centroids[determined_cluster[i], (stockouts[i,2
8] + 1):24])
  true_demand_5[i] = stockouts[i,2]/(1-lost_percentage[i])
}

stockouts=cbind(stockouts,true_demand_5)

head(stockouts)
```

```
##    Item. Total.sales hour.1 hour.2 hour.3 hour.4 hour.5 hour.6 hour.7 hour.8
## 5      5        3115  0.116  0.032  0.048  0.054  0.035  0.049  0.023  0.075
## 12    12        4943  0.144  0.060  0.038  0.019  0.076  0.072  0.008  0.000
## 14    14        4783  0.132  0.033  0.002  0.083  0.033  0.068  0.057  0.006
## 17    17        6294  0.114  0.060  0.031  0.078  0.082  0.026  0.081  0.022
## 19    19        4716  0.144  0.046  0.048  0.004  0.030  0.056  0.036  0.053
## 36    36        3206  0.133  0.003  0.080  0.025  0.005  0.056  0.073  0.008
##    hour.9 hour.10 hour.11 hour.12 hour.13 hour.14 hour.15 hour.16 hour.17
## 5   0.063   0.082   0.064   0.037   0.003   0.134   0.038   0.048   0.039
## 12  0.079   0.081   0.063   0.076   0.076   0.055   0.028   0.036   0.021
## 14  0.079   0.049   0.069   0.061   0.061   0.066   0.039   0.030   0.043
## 17  0.054   0.027   0.029   0.047   0.005   0.139   0.023   0.044   0.041
## 19  0.076   0.048   0.064   0.011   0.082   0.075   0.049   0.029   0.034
```

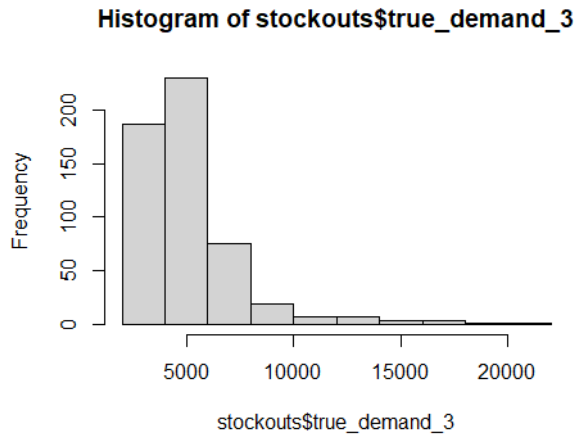
```
## 36  0.029   0.067   0.082   0.045   0.078   0.316   0.000   0.000   0.000
##    hour.18 hour.19 hour.20 hour.21 hour.22 hour.23 hour.24 Total stockouts_time
## 5    0.044   0.012   0.004   0.000   0.000       0       0 1.000             20
## 12   0.028   0.023   0.019   0.000   0.000       0       0 1.002             20
## 14   0.043   0.004   0.011   0.031   0.000       0       0 1.000             21
## 17   0.035   0.028   0.009   0.010   0.015       0       0 1.000             22
## 19   0.039   0.013   0.028   0.036   0.000       0       0 1.001             21
## 36   0.000   0.000   0.000   0.000   0.000       0       0 1.000             14
##    true_demand_2 true_demand_3 true_demand_4 true_demand_5
## 5       3260.693      3384.083      3361.922      3362.471
## 12      5418.381      5428.229      5370.943      5431.621
## 14      5112.274      5071.518      5074.634      5144.143
## 17      6410.982      6532.938      6508.759      6509.616
## 19      5040.661      5000.476      5091.887      5072.084
## 36      3984.741      3765.380      3707.046      3704.176
```

Now let's visualize the demands we have predicted using different clustering k.

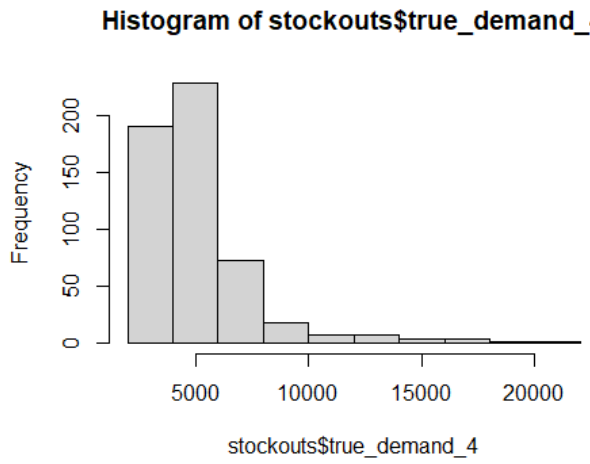
```
hist(stockouts$true_demand_2,xlab= "predicted demand", main="Histogram of
Predicted Demand for Stockout items, with k=2 clustering")
```



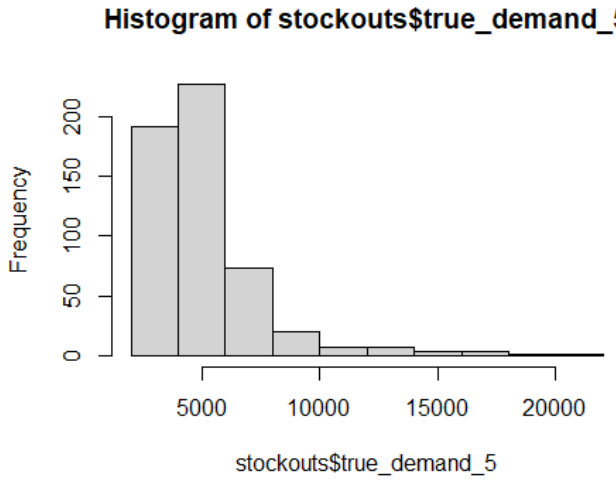

```
hist(stockouts$true_demand_3)
```




```
hist(stockouts$true_demand_4)
```

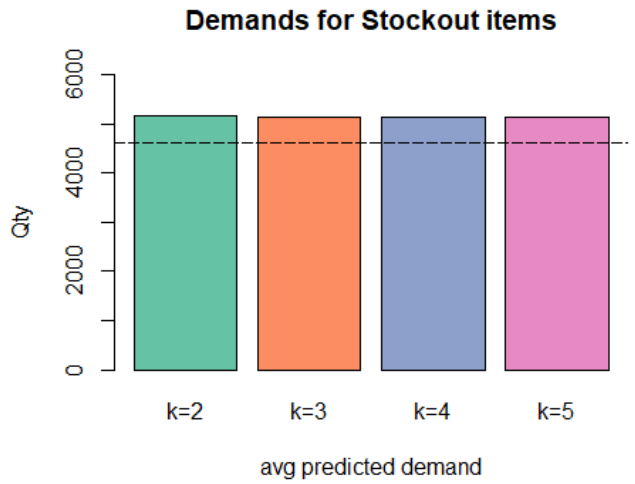



```
hist(stockouts$true_demand_5)
```



We can also plot the average of demands we have predicted with different k and then we will compare it with the average of non-stock out items demand:

```
library(RColorBrewer)
coul <- brewer.pal(4, "Set2")
M= c("k=2","k=3","k=4","k=5")
H = c(mean(stockouts$true_demand_2),
      mean(stockouts$true_demand_3),
      mean(stockouts$true_demand_4),
      mean(stockouts$true_demand_5))
barplot(H,xlab="avg predicted demand",ylab="Qty",main="Demands for Stockou
t items", names.arg= M,col=coul,
         ylim=c(0000,6000))
abline( h = mean(nonstockouts$Total.sales),lty = 5)
```



The horizontal dotted line in above plot shows the non-stockout items actual average demand. This suggests that average demand of stockout items is predicted to be greater than non-stock out items, with any k value of 2 to 5.

2.4. What would you recommend for the number of clusters, k? Why?

I would recommend k=3 for clustering the non-stockout items mainly due to following two reasons.

- Firstly, from 'Within SS' error plot, we can say that k=3 point lies at the elbow.
- Also, average predicted demand of stockout items from k=2 to 5 based clusters comes out to be almost the same.
- Moreover, distribution (Histogram) of these stockout items predicted demands with k=3 to 5 is consistent with the original distribution of total sales of these items, whereas it differs significantly for k=2.

In a nutshell, I would say that k=2 is not accurate enough for demand prediction; whereas, there is no significant benefit in going for extra clusters as in k=4 and 5, and actually k=3 suffices our requirement of accurately predicting stockout items demand. Due to these reasons, I recommend that non-stockout products shall be divided into three clusters for our application.

References:

[1] H A Mehrizi, eBook: MSCI 719 Winter 2023 Cases Multiple (ID: 9723713) Accessed: Jan. 22, 2023. [Online].

Available: https://www.campusebookstore.com/integration/AccessCodes/default.aspx?permalinkId=ee044bf2-fe82-4db0-ad22-088e81954eef&frame=YES&t=permalink&sid=4u2faw45zyslbp45bbqlpc55