Customer Segmentation- Initial Exploration Model with Clustering

JGWJ

9/2/2021

0. Background

This is a demo customer segmentation model, an extension project from a customer credit evaluation and control system I've worked on previously at work. Although the system was made for customer credit evaluation, the sales team benefited in the process of building it- identifying customers with potential for further development.

In the credit system, customers are allocated credit term and limit based on factors such as company size, client base, sales, profitability, purchasing pattern, payment pattern, etc. Customers are grouped based on quantile/ brackets. For example, holding risk factors such as client base and credit rating constant, a 1AA customer would receive higher credit limit compared to a 1BB customer. The sales team used the credit system to segment customers, which in turn affected the resource allocation to the customers. Customer segmentation deserves its own model, hence this project.

This project is an exploration model to search for previously unknown insights/ customer groupings for later customer segmentation modeling.

1. Methods

We will be doing the following:

Section 2. Data Prep: find data, clean data, merge/join datasets

Section 3. Data Exploration: check for duplicates, missing, outliers, normality

Section 4. Clustering: run 4 clustering models

```
a. cluster based on sales pattern (12 monthly data)

b. cluster based on total sales (12-month total)

c. cluster based on location

d. cluster customers based on a, b, c
```

Here I have chosen clustering as this is an initial exploration model, I am searching for trends/ groupings I might have missed out. Clustering being an unsupervised learning method helps in that aspect.

2. Data prep

Firstly, read in 12 monthly sales data from SAP, do some initial exploration, and filter for the information I could use. Note that only active customers would be included. This is the main dataset.

Then use zipcode to identify customers' latitude longitude for location clustering in step 4c.

```
library(magrittr)
library(dplyr)
library(stringr)
library(tidyverse)
library(tidygeocoder)

#----------1. Sales Data----------#
salesData <- read.csv("2020SalesReport.csv", stringsAsFactors = FALSE) #read data

str(salesData) #initial data exploration
summary(salesData)
head(salesData)

salesdf <- salesData[, -c(5,6,8,9,11,12,14,15,17,18,20,21,23,24,26,27,29,30,32,33,35,36,38,39)] #remove

salesdf <- salesdf %>% #rename columns
            select(contains("Sales.Amount")) %>%
            rename(JanSales = January...2020....Sales.Amount) %>%
            rename(FebSales = February...2020....Sales.Amount) %>%
            rename(MarSales = March...2020....Sales.Amount) %>%
            rename(AprSales = April...2020....Sales.Amount) %>%
            rename(MaySales = May...2020....Sales.Amount) %>%
            rename(JunSales = June...2020....Sales.Amount) %>%
            rename(JulSales = July...2020....Sales.Amount) %>%
            rename(AugSales = August...2020....Sales.Amount) %>%
            rename(SepSales = September...2020....Sales.Amount) %>%
            rename(OctSales = October...2020....Sales.Amount) %>%
            rename(NovSales = November...2020....Sales.Amount) %>%
            rename(DecSales = December...2020....Sales.Amount)

salesdf <- salesdf %>% #data formatting
            mutate_all(function(x) gsub("MYR ","",x)) %>%
            mutate_all(function(x) gsub("[,]","",x)) %>%
            mutate_all(function(x) as.numeric(x)) %>%
            mutate_all(~replace_na(., 0))


#----------2. Customers Data----------#

#read customer data
bpData <- read.csv("customerMaster.csv", stringsAsFactors = FALSE)

#initial data exploration
str(bpData)
head(bpData)

#filter for customer location
```

```r
bpArea <- bpData %>%
            select(BP.Code, Area.Code, State, Bill.to.Zip.Code)

#join customer code with location
salescustomers <- salesData %>%
              select("Customer.Code") %>%
              rename(BP.Code = Customer.Code)

customerData <- inner_join(salescustomers, bpArea)

customerData <- customerData %>%
                  rename(BP = BP.Code) %>%
                  rename(Area = Area.Code) %>%
                  rename(ZipCode = Bill.to.Zip.Code)
str(customerData)


#----------3. Latitude Longitude Data----------#
coordinates <- data.frame()

for(i in 1:length(customerData$ZipCode)){
    latlng <- geo(postalcode = customerData$ZipCode[i],
                  country = "Malaysia",
                  method ='osm',lat = latitude, long = longitude)
    coordinates <- rbind(coordinates, latlng)
    print(i)
}

latlng <- unique(coordinates[c("postalcode","latitude","longitude")])


#----------4. Join Data----------#

#merge customer location and sales data
salesdf <- cbind(customerData, salesdf)

#merge customer location, sales data with latitude longitude data
colnames(latlng)[1] <- "ZipCode"
salesdf <- merge(salesdf, latlng, by="ZipCode")

#optional: remove customers with no location info
#salesdf <- salesdf %>% na.omit(salesdf)
#salesdf <- salesdf[-1:-4,]
```

3. Explore Data

A simple exploration of data checking for duplicates, missing data, outliers, normality. From this section I see that there are outliers and data are positively skewed. Outliers might not work well with clustering and skewed data would be troublesome for sales prediction later on.

```
library(outliers)
library(moments)

head(salesdf)

#----------1. check for duplicates----------#
nrow(distinct(salesdf)) #no duplicates


#----------2. check for missing----------#
which(is.na(salesdf), arr.ind=TRUE) #NA's in latitude, longitude cols


#----------3. check for outliers----------#
#boxplot(salesdf[,5:16])

#there are outliers

#optional
#can use Grubbs's test to test outlier
#   Null hypothesis: there is no outlier in one tail
#or histogram for visual identification of outliers
#   hist(salesdf[,i])


#----------4. check for normality----------#
#skewness
skewness(salesdf[,5:16]) #positively skewed

#or Shapiro-Wilk test
#   Null hypothesis: Data is normally distributed
cols <- colnames(salesdf[,5:16])

swdf <- data.frame()
for(i in 5:16){
  sw <- shapiro.test(salesdf[,i])
  swdf <- rbind(swdf, sw[2])
}
swdf <- cbind(cols, swdf)
swdf #null rejected for all sales data: not normally distributed

#optional
#can do qqplot for each month's sales:
#for(i in 5:16){
#   qqnorm(salesdf[,i]); qqline(salesdf[,i])
#}
```

4. Clustering Models

From data exploration, there are outliers in this dataset; clustering may not be the best method. With that in mind, we'll continue with clustering as initial exploration to see if there is any interesting result that could be helpful for later model building.

4a. Purchasing Pattern: K-Means Clustering

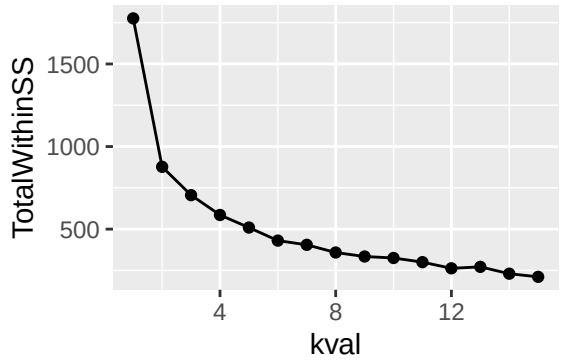
Here K-means is used to cluster sales pattern (customer purchasing pattern), a customer loyalty gauge. Some customers are consistent in their purchases while others are more erratic; this would affect quota allocation for each customer.

```
library(ggplot2)

#----------1. First Model----------#
ccdf <- data.frame()

for(k in 1:15){
  set.seed(54)
  custcluster <- kmeans(scale(salesdf[,c(5:16)]), centers = k, nstart=25)

  TotalWithinSS <- custcluster$tot.withinss
  Performance <- round((custcluster$betweenss/custcluster$totss)*100,digits=2)
  ccdf <- rbind(ccdf, c(k, TotalWithinSS, Performance))
}
colnames(ccdf) <- c("kval","TotalWithinSS","Performance")
str(ccdf)
ggplot(ccdf, aes(kval, TotalWithinSS)) + geom_line() + geom_point()
```




```
#----------2. Refine Model----------#
#Optimal k = 6
#But due to resource constraint, k = 4
#scale data
custCM <- kmeans(scale(salesdf[,c(5:16)]), centers = 4, nstart=25)

salesdf <- cbind(salesdf, custCM$cluster)
colnames(salesdf)[19] <- "SalesPatternCluster"
str(salesdf)


#----------3. Examine Results----------#

cluster1 <- salesdf %>% filter(SalesPatternCluster==1)
cluster2 <- salesdf %>% filter(SalesPatternCluster==2)
cluster3 <- salesdf %>% filter(SalesPatternCluster==3)
cluster4 <- salesdf %>% filter(SalesPatternCluster==4)
```

```
cluster1 #Varying sales pattern
#colMeans(salesdf[salesdf$SalesPatternCluster == 1, 5:16])
cluster2 #Very inconsistent sales pattern
#colMeans(salesdf[salesdf$SalesPatternCluster == 2, 5:16])
cluster3 #Strong sales pattern but no sales in April
#colMeans(salesdf[salesdf$SalesPatternCluster == 3, 5:16])
cluster4 #Strong sales pattern
#colMeans(salesdf[salesdf$SalesPatternCluster == 4, 5:16])
```

4b. Sales: K-Means Clustering

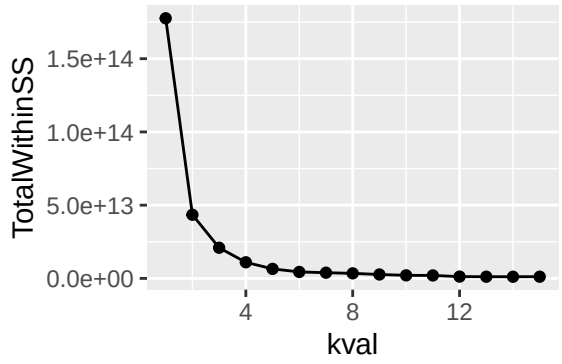
This is a more traditional gauge: how big is the customer, by revenue? Here I'm trying out clustering instead of going by quantile/ brackets.

```
#----------1. Data Prep----------#
asdf <- salesData %>%
        rename(AnnualSales = Annual.Total) %>%
        rename(BP = Customer.Code) %>%
        mutate(AnnualSales = gsub("MYR ","",AnnualSales)) %>%
        mutate(AnnualSales = gsub("[,]","",AnnualSales)) %>%
        mutate(AnnualSales = as.numeric(AnnualSales)) %>%
        select(BP, AnnualSales)


#----------2. First Model----------#
ccdf1 <- data.frame()

for(k in 1:15){
  set.seed(54)
  custcluster <- kmeans(asdf[,2], centers = k, nstart=25)

  TotalWithinSS <- custcluster$tot.withinss
  Performance <- round((custcluster$betweenss/custcluster$totss)*100,digits=2)
  ccdf1 <- rbind(ccdf1, c(k, TotalWithinSS, Performance))
}
colnames(ccdf1) <- c("kval","TotalWithinSS","Performance")
str(ccdf1)
ggplot(ccdf1, aes(kval, TotalWithinSS)) + geom_line() + geom_point()
```



```
#----------2. Refine Model----------#
#Optimal k = 4
salesCM <- kmeans(asdf[,2], centers = 4, nstart=25)

asdf <- cbind(asdf, SalesCluster = salesCM$cluster)
str(asdf)


#----------3. Examine Results----------#

cluster1 <- asdf %>% filter(SalesCluster==1)
cluster2 <- asdf %>% filter(SalesCluster==2)
cluster3 <- asdf %>% filter(SalesCluster==3)
cluster4 <- asdf %>% filter(SalesCluster==4)

cluster1 #tier 1 sales
#mean(cluster1$AnnualSales)
cluster2 #tier 2 sales
#mean(cluster2$AnnualSales)
cluster3 #tier 4 sales
#mean(cluster3$AnnualSales)
cluster4 #tier 3 sales
#mean(cluster4$AnnualSales)
```

4c. Location: Hierarchical Clustering

Location clustering is more of distance cluster. I want to try hierarchical clustering. This could be further build out with map visualization and density based clustering and with total sales as additional factor.

```
library(dplyr)
library(tidyverse)
library(fields)
library(dbscan)


#----------1. Data Prep----------#

head(salesData)

sdf <- salesData %>%
        rename(AnnualSales = Annual.Total) %>%
        rename(BP = Customer.Code) %>%
        mutate(AnnualSales = gsub("MYR ","",AnnualSales)) %>%
        mutate(AnnualSales = gsub("[,]","",AnnualSales)) %>%
        mutate(AnnualSales = as.numeric(AnnualSales)) %>%
        mutate(AnnualSales = scale(AnnualSales)) %>%
        select(BP, AnnualSales)

ll <- cbind(salesdf$latitude, salesdf$longitude)

lldf <- data.frame(ll) %>% replace(is.na(.), 0)
```

```
#distance matrix
distMatrix <- rdist.earth(lldf, miles=FALSE, R=6371)


#----------2. Model----------#

#hierarchical
hcm <- hclust(as.dist(distMatrix), method="single")
hclusters <- cutree(hcm, h=45) #45km being about 30min driving distance


#----------3. Plot Model----------#

#hierarchical
#plot(lldf[,2],lldf[,1],col=hclusters, pch=20)


#----------4. Add location cluster to dataset----------#
sdf <- cbind(sdf, lldf, HCluster = hclusters)
```

4d. Customer Segmentation: k-means clustering

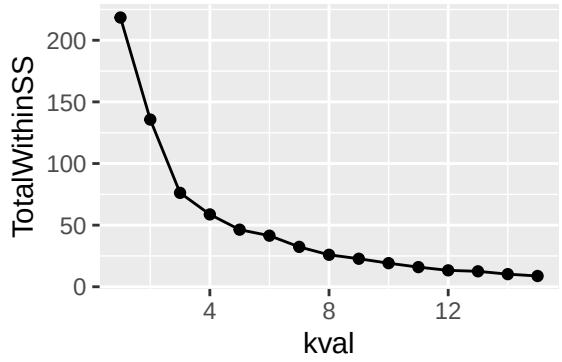
This is the last clustering, the actual segmentation model. Taking results from a, b, c let's see if there are interesting groupings to be considered in later models.

```
csdf <- cbind(LocationCluster = sdf$HCluster, SalesCluster = asdf$SalesCluster, SalesPatternCluster = sa


#----------1. First Model----------#
cscm <- data.frame()

for(k in 1:15){
  set.seed(54)
  custcluster <- kmeans(csdf, centers = k, nstart=25)

  TotalWithinSS <- custcluster$tot.withinss
  Performance <- round((custcluster$betweenss/custcluster$totss)*100,digits=2)
  cscm <- rbind(cscm, c(k, TotalWithinSS, Performance))
}
colnames(cscm) <- c("kval","TotalWithinSS","Performance")
str(cscm)
ggplot(cscm, aes(kval, TotalWithinSS)) + geom_line() + geom_point()
```




```
#----------2. Refine Model----------#
#Optimal k = 4

custSeg <- kmeans(csdf, centers = 4, nstart=25)

csdf <- cbind(BP = salesdf$BP, csdf, CustomerSegment = custSeg$cluster)

csdf <- data.frame(csdf)


#----------3. Examine Results----------#

segment1 <- csdf %>% filter(CustomerSegment==1)
segment2 <- csdf %>% filter(CustomerSegment==2)
segment3 <- csdf %>% filter(CustomerSegment==3)
segment4 <- csdf %>% filter(CustomerSegment==4)


#Here I'll export the customer segments to Excel to explore with customer master data
#segment1
#segment2
#segment3
#segment4
```

5. Conclusion

The customer purchasing pattern cluster provided good information, previously we only checked whether customers purchased monthly. With this, loyal customers with consistently strong purchasing pattern are grouped separately from those who purchased monthly but in sporadic amount. The sales/location clustering can be reworked with DBSCAN with annual sales or delivery frequency as weight.Outliers can be further examined to determine if they should be modified/ thrown out/ kept as is. The overall customer segmentation clustering then can be rerun. If we do sales prediction later on, the monthly sales data are not normally distributed and may need to be log transformed or use Holt Winter's to first smooth the data.

Thank you for reading through!