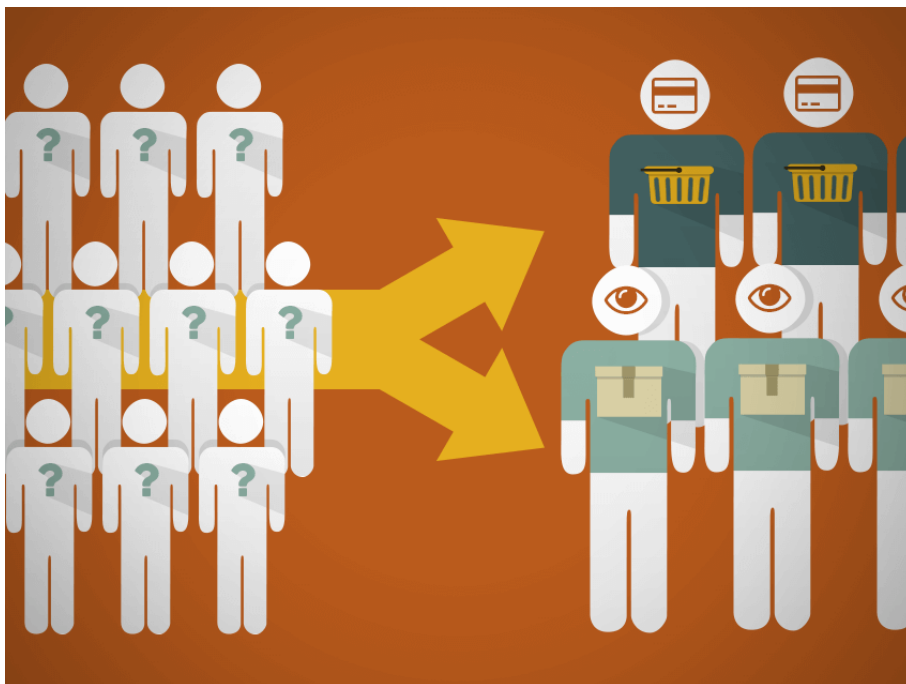# Data Science with R

Project 4:

High value customers identification for an E-Commerce company

Made by:

Siddhartha Patra

BUSINESS SCENARIO

A UK-based online retail store has captured the sales data for different products for the period of one year (Nov 2016 to Dec 2017). The organization sells gifts primarily on the online platform. The customers who make a purchase consume directly for themselves. There are small businesses that buy in bulk and sell to other customers through the retail outlet channel.

The objective of this project is to find significant customers for the business who make high purchases of their favourite products and use the clustering methodology to segment customers into groups.

EXPECTATION /GOALS

The common goal is to identify high-value and low-value customers for marketing purposes such as rolling out a loyalty program to the high-value customers after identification of segments.

For this project I have decided to calculate and work with the following metrics for each customer:

- Recency of last purchase,
- Frequency of purchase, and
- Monetary value

These three variables, collectively known as **RFM**, are often used in customer segmentation for marketing purposes. Lastly, I'm using the **k-means clustering** technique because it can efficiently handle large datasets and iterates quickly to good solutions.

CODE

```
#################################
### E-Commerce Final Project ###
#################################

#Installing necessary packages
install.packages("plyr")
install.packages("ggplot2")
install.packages("scales")
install.packages("NbClust")

#Select Working Directory
setwd(choose.dir())
getwd()

#Importing dataset
data <- read.csv("Ecommerce.csv")
View(data)
str(data)
```

```r
#################
# DATA CLEANING #
#################

#Removing redundant column X
data <- subset(data, select = -X)

#Some invoices are missing ID numbers.
#We'll be doing the analysis at the customer level, so remove any observations with missing ID
numbers.
length(unique(data$CustomerID))
sum(is.na(data$CustomerID))
data <- subset(data, !is.na(data$CustomerID))

#Customer clusters vary by geography.
#So here we'll restrict the data to one geographic unit.
table(data$Country)

#We can clearly see that, UK has significantly more number of transactions comapared to the rest of
the countries.
#Hence we'll only consider UK for our analysis.
data <- subset(data, Country == "United Kingdom")

#Let's see the number of unique invoices and unique customers.
length(unique(data$InvoiceNo))
length(unique(data$CustomerID))
#We now have a dataset of 19,857 unique invoices and 3,950 unique customers.

#To calculate the recency and frequency variables below, it will be necessary to distinguish invoices
with purchases from invoices with returns.
#Identify returns
data$item.return <- grepl("C", data$InvoiceNo, fixed=TRUE)
data$purchase.invoice <- ifelse(data$item.return=="TRUE", 0, 1)


###################################
# Creating a Customer-level dataset #
###################################
customers <- as.data.frame(unique(data$CustomerID))
names(customers) <- "CustomerID"


###########
# Recency #
###########

#Converting 'InvoiceDate' from factor to date type format
```

```r
data$InvoiceDate <- as.Date(data$InvoiceDate, "%d-%b-%y")
str(data)

# Adding a recency column by substracting the InvoiceDate from the (last InvoiceDate+1)
data$recency <- as.Date("2017-12-08") - as.Date(data$InvoiceDate)

# remove returns so only consider the data of most recent "purchase"
temp <- subset(data, purchase.invoice == 1)

# Obtain # of days since most recent purchase
recency <- aggregate(recency ~ CustomerID, data=temp, FUN=min, na.rm=TRUE)
remove(temp)

# Add recency to customer data
customers <- merge(customers, recency, by="CustomerID", all=TRUE, sort=TRUE)
remove(recency)

customers$recency <- as.numeric(customers$recency)


#############
# Frequency #
#############

customer.invoices <- subset(data, select = c("CustomerID","InvoiceNo", "purchase.invoice"))
customer.invoices <- customer.invoices[!duplicated(customer.invoices), ]
customer.invoices <- customer.invoices[order(customer.invoices$CustomerID),]
row.names(customer.invoices) <- NULL

# Number of invoices/year (purchases only)
annual.invoices <- aggregate(purchase.invoice ~ CustomerID, data=customer.invoices, FUN=sum,
na.rm=TRUE)
names(annual.invoices)[names(annual.invoices)=="purchase.invoice"] <- "frequency"

# Add # of invoices to customers data
customers <- merge(customers, annual.invoices, by="CustomerID", all=TRUE, sort=TRUE)
remove(customer.invoices, annual.invoices)

range(customers$frequency)
table(customers$frequency)

# Remove customers who have not made any purchases in the past year
customers <- subset(customers, frequency > 0)


##############################
# Monetary Value of Customers #
##############################
```

```r
# Total spent on each item on an invoice
data$Amount <- data$Quantity * data$UnitPrice

# Aggregated total sales to customer
total.sales <- aggregate(Amount ~ CustomerID, data=data, FUN=sum, na.rm=TRUE)
names(total.sales)[names(total.sales)=="Amount"] <- "monetary"

# Add monetary value to customers dataset
customers <- merge(customers, total.sales, by="CustomerID", all.x=TRUE, sort=TRUE)
remove(total.sales)

# Identify customers with negative monetary value numbers, as they were presumably returning
# purchases from the preceding year
hist(customers$monetary)
customers$monetary <- ifelse(customers$monetary < 0, 0, customers$monetary) # reset negative
numbers to zero
hist(customers$monetary)


################################
# Pareto Principle: 80/20 Rule #
################################

customers <- customers[order(-customers$monetary),]

# Apply Pareto Principle (80/20 Rule)
pareto.cutoff <- 0.8 * sum(customers$monetary)
customers$pareto <- ifelse(cumsum(customers$monetary) <= pareto.cutoff, "Top 20%", "Bottom
80%")
customers$pareto <- factor(customers$pareto, levels=c("Top 20%", "Bottom 80%"), ordered=TRUE)
levels(customers$pareto)
round(prop.table(table(customers$pareto)), 2)
remove(pareto.cutoff)

customers <- customers[order(customers$CustomerID),]


####################
# Preprocess data #
####################

# Log-transform positively-skewed variables
customers$recency.log <- log(customers$recency)
customers$frequency.log <- log(customers$frequency)
customers$monetary.log <- customers$monetary + 0.1 # can't take log(0), so add a small value to
remove zeros
customers$monetary.log <- log(customers$monetary.log)
```

```
# Z-scores
customers$recency.z <- scale(customers$recency.log, center=TRUE, scale=TRUE)
customers$frequency.z <- scale(customers$frequency.log, center=TRUE, scale=TRUE)
customers$monetary.z <- scale(customers$monetary.log, center=TRUE, scale=TRUE)


##################
# Visualize data #
##################

library(ggplot2)
library(scales)

# Original scale
scatter.1 <- ggplot(customers, aes(x = frequency, y = monetary))
scatter.1 <- scatter.1 + geom_point(aes(colour = recency, shape = pareto))
scatter.1 <- scatter.1 + scale_shape_manual(name = "80/20 Designation", values=c(17, 16))
scatter.1 <- scatter.1 + scale_colour_gradient(name="Recency\n(Days since Last Purchase))")
scatter.1 <- scatter.1 + scale_y_continuous(label=dollar)
scatter.1 <- scatter.1 + xlab("Frequency (Number of Purchases)")
scatter.1 <- scatter.1 + ylab("Monetary Value of Customer (Annual Sales)")
scatter.1

#This first graph uses the variables' original metrics and is almost completely uninterpretable.
#There's a clump of data points in the lower left-hand corner of the plot, and then a few outliers.
#This is why we log-transformed the input variables.

# Log-transformed
scatter.2 <- ggplot(customers, aes(x = frequency.log, y = monetary.log))
scatter.2 <- scatter.2 + geom_point(aes(colour = recency.log, shape = pareto))
scatter.2 <- scatter.2 + scale_shape_manual(name = "80/20 Designation", values=c(17, 16))
scatter.2 <- scatter.2 + scale_colour_gradient(name="Log-transformed Recency")
scatter.2 <- scatter.2 + xlab("Log-transformed Frequency")
scatter.2 <- scatter.2 + ylab("Log-transformed Monetary Value of Customer")
scatter.2



####################
# Handling outliers #
####################

# How many customers are represented by the two data points in the lower left-hand corner of the
plot? 19
delete <- subset(customers, monetary.log < 0)
no.value.custs <- unique(delete$CustomerID)
delete2 <- subset(data, CustomerID %in% no.value.custs)
```

```
delete2 <- delete2[order(delete2$CustomerID, delete2$InvoiceDate),]
remove(delete, delete2, no.value.custs)

# Scaled variables
scatter.3 <- ggplot(customers, aes(x = frequency.z, y = monetary.z))
scatter.3 <- scatter.3 + geom_point(aes(colour = recency.z, shape = pareto))
scatter.3 <- scatter.3 + scale_shape_manual(name = "80/20 Designation", values=c(17, 16))
scatter.3 <- scatter.3 + scale_colour_gradient(name="Z-scored Recency")
scatter.3 <- scatter.3 + xlab("Z-scored Frequency")
scatter.3 <- scatter.3 + ylab("Z-scored Monetary Value of Customer")
scatter.3

remove(scatter.1, scatter.2, scatter.3)




##################################################
# Determining number of clusters through K-Means #
##################################################

preprocessed <- customers[,9:11]
j <- 10 # specify the maximum number of clusters you want to try out

models <- data.frame(k=integer(),
            tot.withinss=numeric(),
            betweenss=numeric(),
            totss=numeric(),
            rsquared=numeric())

  for (k in 1:j ) {

    print(k)

    # Run kmeans
    # nstart = number of initial configurations; the best one is used
    # $iter will return the iteration used for the final model
    output <- kmeans(preprocessed, centers = k, nstart = 20)

    # Add cluster membership to customers dataset
    var.name <- paste("cluster", k, sep="_")
    customers[,(var.name)] <- output$cluster
    customers[,(var.name)] <- factor(customers[,(var.name)], levels = c(1:k))

    # Graph clusters
    cluster_graph <- ggplot(customers, aes(x = frequency.log, y = monetary.log))
    cluster_graph <- cluster_graph + geom_point(aes(colour = customers[,(var.name)]))
    colors <- c('red','orange','green3','deepskyblue','blue','darkorchid4','violet','pink1','tan3','black')
    cluster_graph <- cluster_graph + scale_colour_manual(name = "Cluster Group", values=colors)
```

```r
        cluster_graph <- cluster_graph + xlab("Log-transformed Frequency")
        cluster_graph <- cluster_graph + ylab("Log-transformed Monetary Value of Customer")
        title <- paste("k-means Solution with", k, sep=" ")
        title <- paste(title, "Clusters", sep=" ")
        cluster_graph <- cluster_graph + ggtitle(title)
        print(cluster_graph)

        # Cluster centers in original metrics
        library(plyr)
        print(title)
        cluster_centers <- ddply(customers, .(customers[,(var.name)]), summarize,
                       monetary=round(median(monetary),2),# use median b/c this is the raw, heavily-
skewed data
                       frequency=round(median(frequency),1),
                       recency=round(median(recency), 0))
        names(cluster_centers)[names(cluster_centers)=="customers[, (var.name)]"] <- "Cluster"
        print(cluster_centers)
        cat("\n")
        cat("\n")

        # Collect model information
        models[k,("k")] <- k
        models[k,("tot.withinss")] <- output$tot.withinss # the sum of all within sum of squares
        models[k,("betweenss")] <- output$betweenss
        models[k,("totss")] <- output$totss # betweenss + tot.withinss
        models[k,("rsquared")] <- round(output$betweenss/output$totss, 3) # percentage of variance
explained by cluster membership
        assign("models", models, envir = .GlobalEnv)

        remove(output, var.name, cluster_graph, cluster_centers, title, colors)

    }

    remove(k)

# Graph variance explained by number of clusters
r2_graph <- ggplot(models, aes(x = k, y = rsquared))
r2_graph <- r2_graph + geom_point() + geom_line()
r2_graph <- r2_graph + scale_y_continuous(labels = scales::percent)
r2_graph <- r2_graph + scale_x_continuous(breaks = 1:j)
r2_graph <- r2_graph + xlab("k (Number of Clusters)")
r2_graph <- r2_graph + ylab("Variance Explained")
r2_graph

# Graph within sums of squares by number of clusters
# Look for a "bend" in the graph, as with a scree plot
ss_graph <- ggplot(models, aes(x = k, y = tot.withinss))
ss_graph <- ss_graph + geom_point() + geom_line()
```

```
ss_graph <- ss_graph + scale_x_continuous(breaks = 1:j)
ss_graph <- ss_graph + scale_y_continuous(labels = scales::comma)
ss_graph <- ss_graph + xlab("k (Number of Clusters)")
ss_graph <- ss_graph + ylab("Total Within SS")
ss_graph

remove(j, r2_graph, ss_graph)


############################################################
# Using NbClust metrics to determine number of clusters #
############################################################

library(NbClust)
set.seed(1)
nc <- NbClust(preprocessed, min.nc=2, max.nc=7, method="kmeans")
table(nc$Best.n[1,])

nc$All.index # estimates for each number of clusters on 26 different metrics of model fit

barplot(table(nc$Best.n[1,]),
    xlab="Number of Clusters", ylab="Number of Criteria",
    main="Number of Clusters Chosen by Criteria")

remove(preprocessed)
```

ANALYSIS (along with output screenshot)

RFM Variables

The original dataset was organized long, with invoices nested within customer. To tackle this, I have created a customer-level dataset and added recency, frequency, and monetary value data to it.

- The recency variable denotes to the number of days that have elapsed since the customer last purchased something (so, smaller numbers indicate more recent activity on the customer's account).
- Frequency refers to the number of invoices with purchases during the year.
- Monetary value is the amount that the customer spent during the year. Some customers have negative monetary values. These customers probably returned something during the year that they had purchased before the year started, so I have reset their monetary value to zero.

80/20 Rule

The 80/20 Rule (also known as the Pareto Principle), is the concept that 80% of the results generally come from 20% of the causes. In this context, it implies that ~80% of sales would be produced by the

top ~20% of customers. These 20% represent the high-value, important customers a business would want to protect.

```
>
> customers <- customers[order(-customers$monetary),]
>
> # Apply Pareto Principle (80/20 Rule)
> pareto.cutoff <- 0.8 * sum(customers$monetary)
> customers$pareto <- ifelse(cumsum(customers$monetary) <= pareto.cutoff, "Top 20%", "Bottom 80%")
> customers$pareto <- factor(customers$pareto, levels=c("Top 20%", "Bottom 80%"), ordered=TRUE)
> levels(customers$pareto)
[1] "Top 20%"    "Bottom 80%"
> round(prop.table(table(customers$pareto)), 2)

   Top 20% Bottom 80%
      0.29       0.71
```

To make a point about outliers, I have created some simple segments here by looking at the top customers who produced 80% of annual sales for the year. In this dataset, 80% of the annual sales are generated by the top 29% of customers, so the percentage isn't quite 20%, but it's not that far off and it does demonstrate that there's a small segment producing the bulk of the value.

Pre-processing data

k-means clustering requires continuous variables and works best with relatively normally-distributed, standardized input variables. Standardizing the input variables is quite significant; otherwise, input variables with larger variances will have commensurately greater influence on the results. Hence, I have transformed our three input variables, using log-transform, to reduce positive skew and then standardize them as z-scores.

Visualize data

The user has to select the number of clusters with k-means clustering. Looking at the data can give a sense for what we are dealing with and how many clusters we might have. In the graphs below, the outcome we're probably most interested in, customer monetary value, is plotted on the y-axis. Frequency of purchases is on the x-axis, and the third variable, recency of purchase, is represented by color-coding the data points. Lastly, we have also included the 80/20 Rule segments using different shapes, so we could map those designations on to customer monetary value, frequency, and recency.
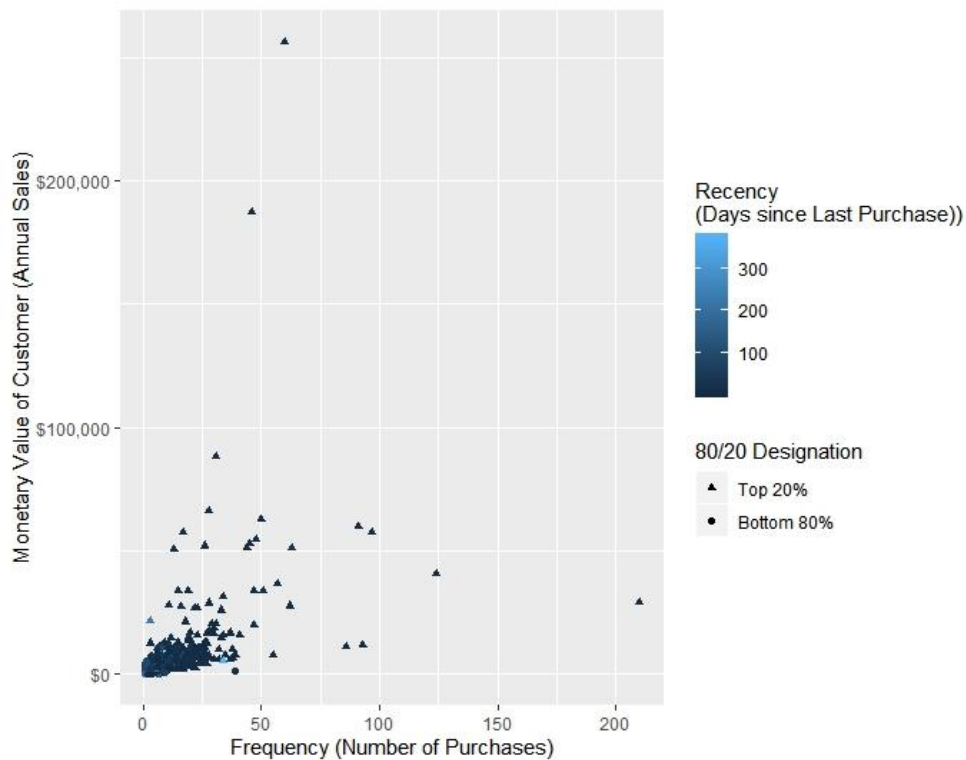
This first graph uses the variables' original metrics and as we can see, the result is almost completely uninterpretable. There's a clump of data points in the lower left-hand corner of the plot, and then a few outliers. This is reason we have log-transformed our input variables.
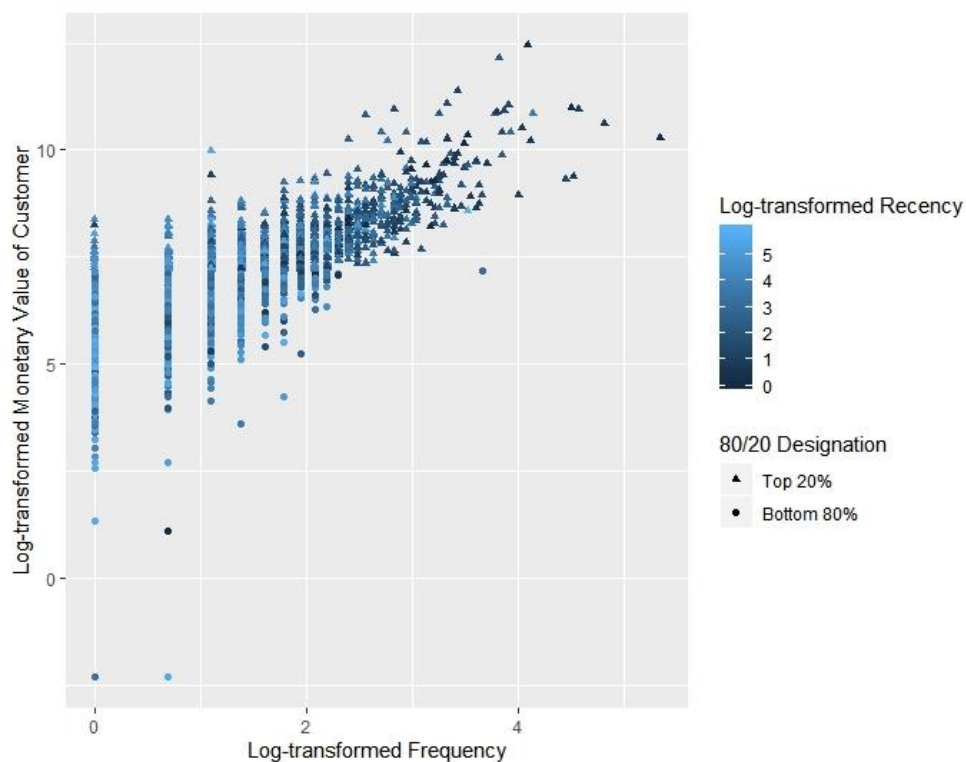
This is better. Now we can see a scattering of high-value, high-frequency customers in the top, right-hand corner of the graph. These data points are dark, indicating that they've purchased something

recently. In the bottom, left-hand corner of the plot, we can see a couple of low-value, low frequency customers who haven't purchased anything recently, with a range of values in between. Notably, we can also see that the data points are fairly continuously-distributed. There isn't really any clear clusters formations. This means that any cluster groupings we create won't exactly reflect some true, underlying group membership – they'll be somewhat arbitrary distinctions that we draw for our own purposes.
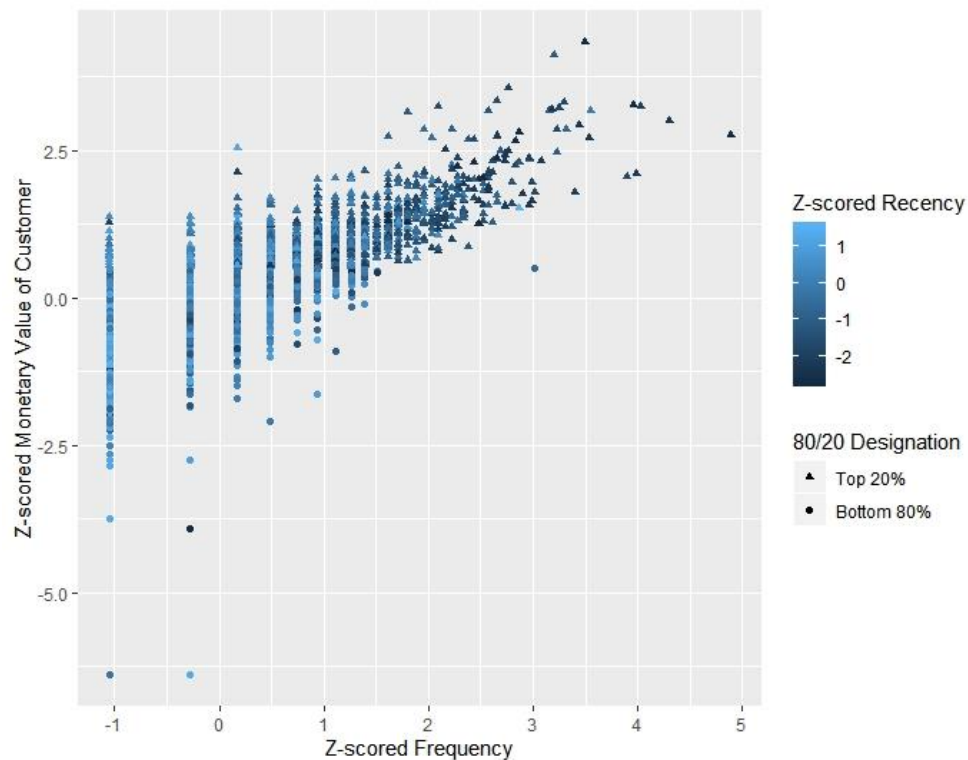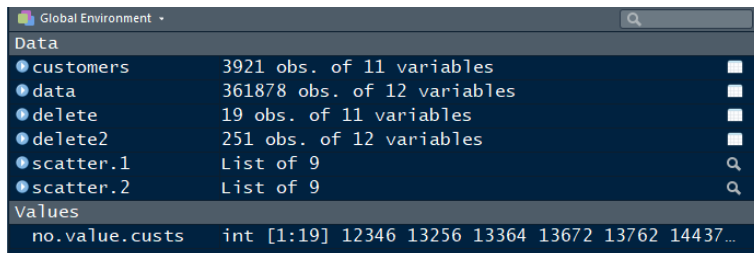
This third scatterplot is basically identical to the second – it illustrates that even though we've changed the scaling for the analysis, the shape of the distributions and the relationships among the variables remain the same.

Handling outliers

One question we might have about those dots in the bottom, left-hand corner is how many customers they represent. The following code investigates them a little more thoroughly.

```
205 ▾ #####################
206   # Handling outliers #
207 ▾ #####################
208
209   # How many customers are represented by the two data points in the lower left-hand corner of the plot? 19
210   delete <- subset(customers, monetary.log < 0)
211   no.value.custs <- unique(delete$CustomerID)
212   delete2 <- subset(data, CustomerID %in% no.value.custs)
213   delete2 <- delete2[order(delete2$CustomerID, delete2$InvoiceDate),]
214   remove(delete, delete2, no.value.custs)
```

Global Environment ▾		
Data		
customers	3921 obs. of 11 variables	■
data	361878 obs. of 12 variables	■
delete	19 obs. of 11 variables	■
delete2	251 obs. of 12 variables	■
scatter.1	List of 9	🔍
scatter.2	List of 9	🔍
Values		
no.value.custs	int [1:19] 12346 13256 13364 13672 13762 14437…	

The nineteen, no-value customers we just investigated are all customers who returned everything they bought. They highlight a vital decision point at this stage in the analysis. k-means clustering tends to be sensitive to outliers, such that outliers will sometimes end up being clustered together in their own tiny group. This is often cited as a reason to exclude them from the analysis.

In this type of customer segmentation, however, the outliers could be one of the most important customers to understand. In the top, right-hand corner, we have customers who are outliers in terms of being extraordinarily high-value, high-frequency shoppers. These data points are all represented with little triangles, indicating that they're in the top 20% category. These are important customers to understand, because they're the customers we most want.

At the other end of the continuum, we have the no-value customers in the bottom, left-hand corner. These customers, too, may be important to model and understand – they're the customers we want to minimize. Hence, I have consciously decided to include these outliers in our analysis.
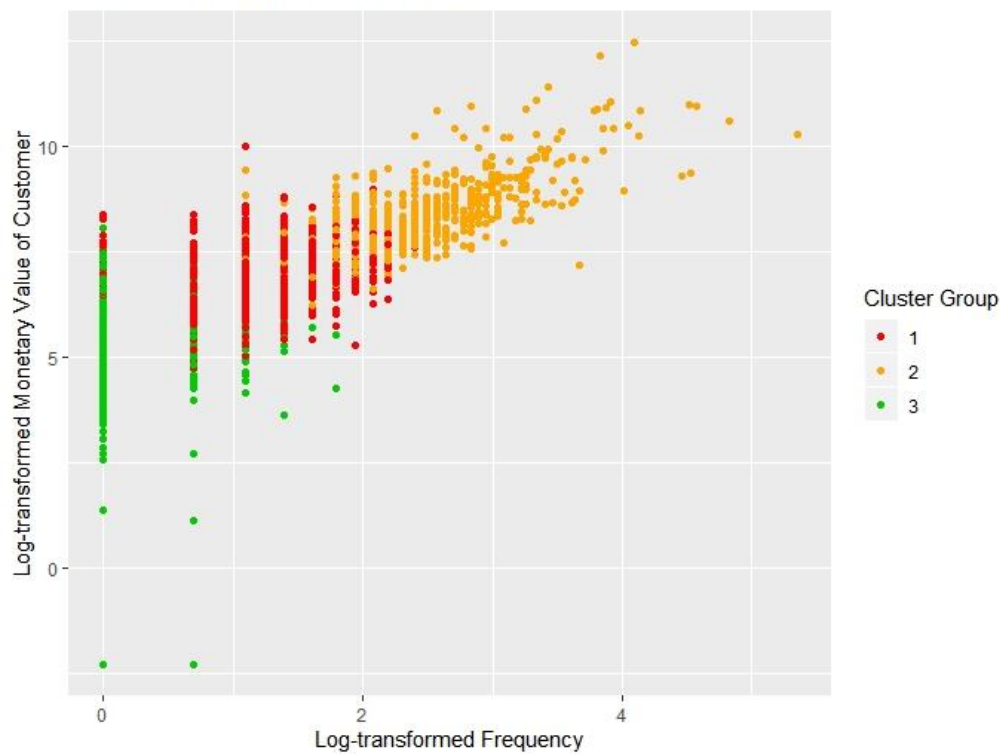
Determine number of clusters / run k-means

For simplicity, I haven't posted graphs for every model here, but somewhere from 2-5 clusters seems most interpretable.
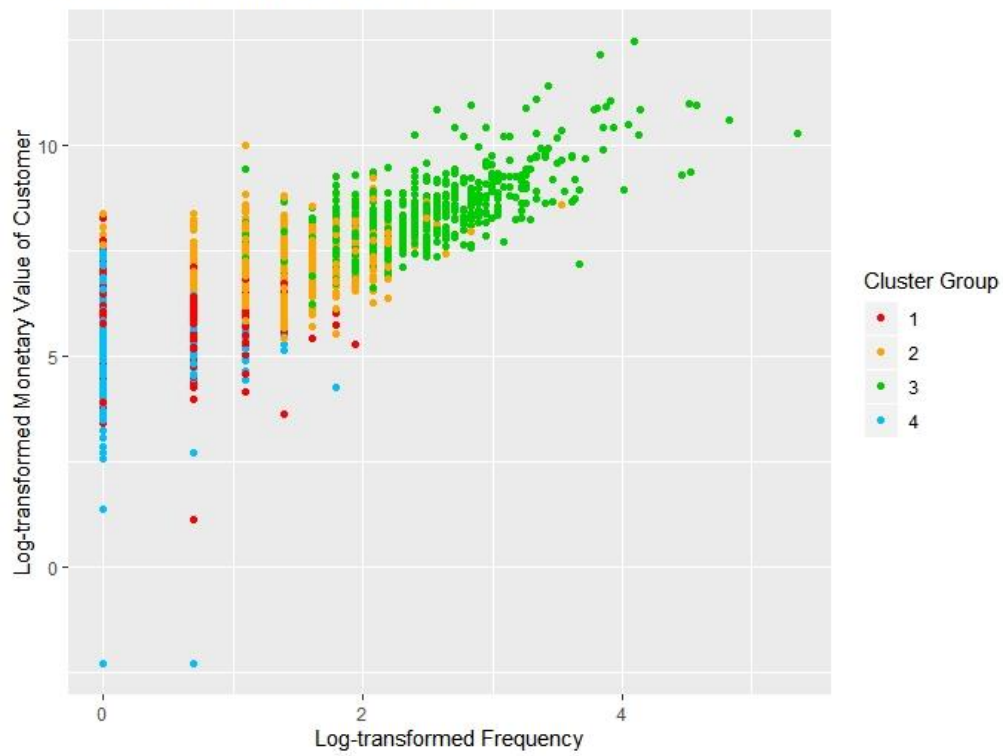

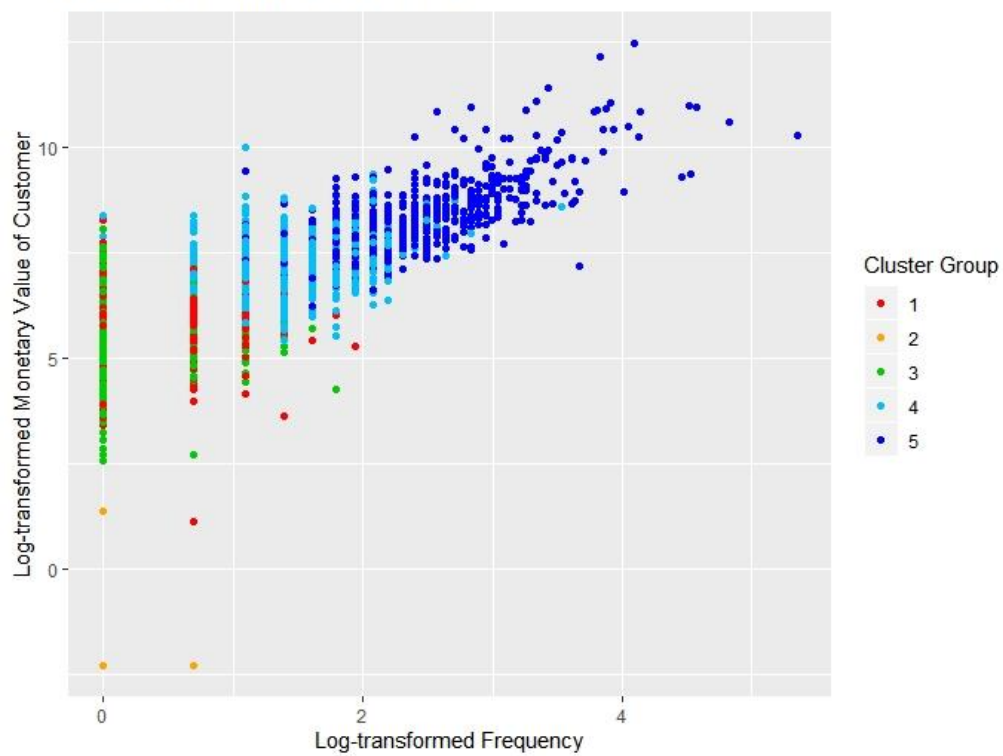### k-means Solution with 2 Clusters
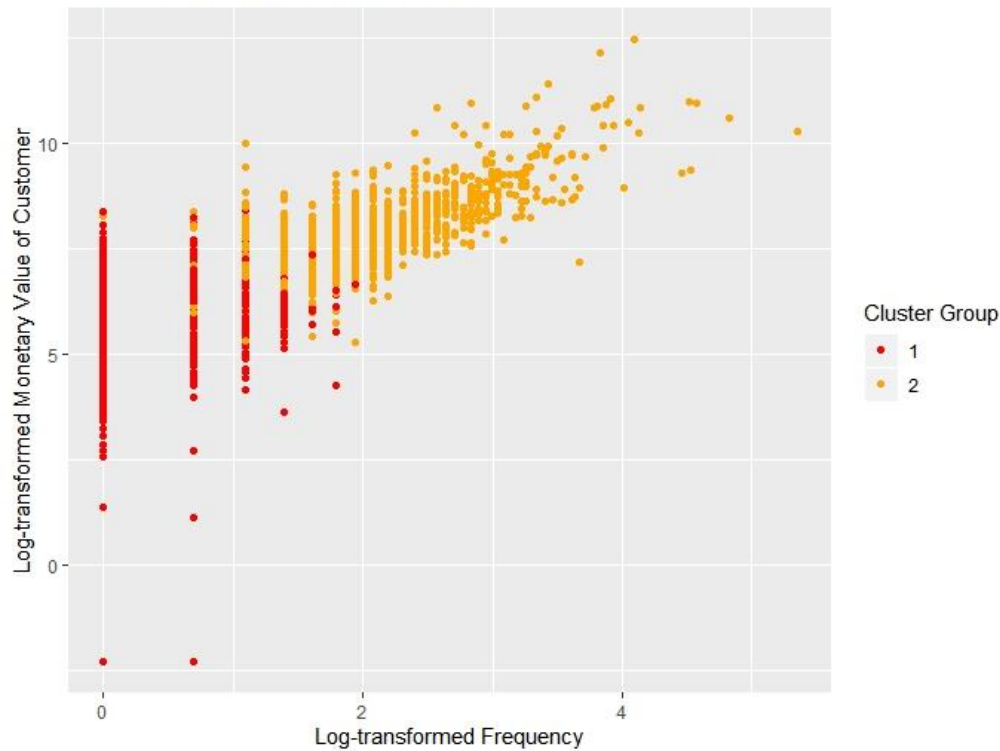
k-means Solution with 3 Clusters

k-means Solution with 4 Clusters

k-means Solution with 5 Clusters

A 2-cluster solution produces one group of high-value (median = $ 1839.07), high-frequency (median = 5 purchases) customers who have purchased recently (median = 17 days since their most recent purchase), and one group of lower value (median = $ 330.62), low frequency (median = 1 purchase) customers for whom it's been a median of 104 days since their last purchase. Although these two clusters are clear and interpretable, this may be simplifying customer behaviour too much.

As we add clusters, we gain insight into increasingly subtle distinctions between customers. I really like the 5-cluster solution. It gives us: a high-value, high-frequency, recent purchase group (cluster 5), a medium-value, medium-frequency, relatively-recent purchase group (cluster 4), two clusters of low-value, low-frequency customers broken down by whether their last purchase was recent or much earlier in the year (clusters 3 and 1, respectively), and lastly a no-value cluster whose median value to the business is $0.00 (cluster 2).

As we move beyond 5 clusters, the graphs become increasingly hard to interpret visually, and the cluster centres start to make distinctions that may not be that helpful (e.g., low-value-with-1-purchase vs. low-value-with-2-purchases customers).

```
[1] 1
[1] "k-means Solution with 1 Clusters"
  Cluster monetary frequency recency
1       1   633.66         2        51


[1] 2
[1] "k-means Solution with 2 Clusters"
  Cluster monetary frequency recency
1       1   330.62         1       104
2       2  1839.07         5        17


[1] 3
[1] "k-means Solution with 3 Clusters"
  Cluster monetary frequency recency
1       1   897.05         3        37
2       2  3127.82         9         9
3       3   258.94         1       157


[1] 4
[1] "k-means Solution with 4 Clusters"
  Cluster monetary frequency recency
1       1   389.85         2        20
2       2  1145.43         4        59
3       3  3075.04         9         9
4       4   257.70         1       186


[1] 5
[1] "k-means Solution with 5 Clusters"
  Cluster monetary frequency recency
1       1   373.24         2        20
2       2     0.00         1       139
3       3   280.73         1       186
4       4  1196.68         4        52
5       5  3181.30        10         8
```

There are a few of other things we can check to assist us in picking the optimal solution. We can see below graphs of the variance explained and within-cluster variance by number of clusters. These graphs are two different ways of visualizing the same information – in both cases, we're looking for an "elbow" or bend in the graph beyond which additional clusters add little additional explanatory power.
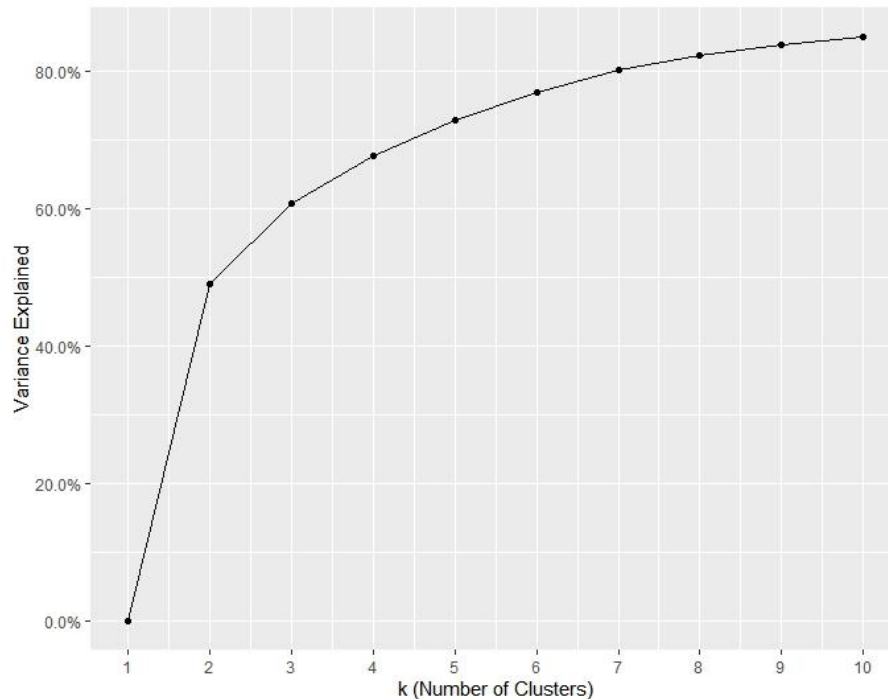

Both graphs look to have elbows at around 2 clusters, but a 2-cluster solution explains only 49% of the variance and, once again, a 2-cluster solution may be too much of a simplification to really help our business problem with targeted marketing. The 5-cluster solution explains ~73% of the variance, but there are no clear elbows in the graph at this point.
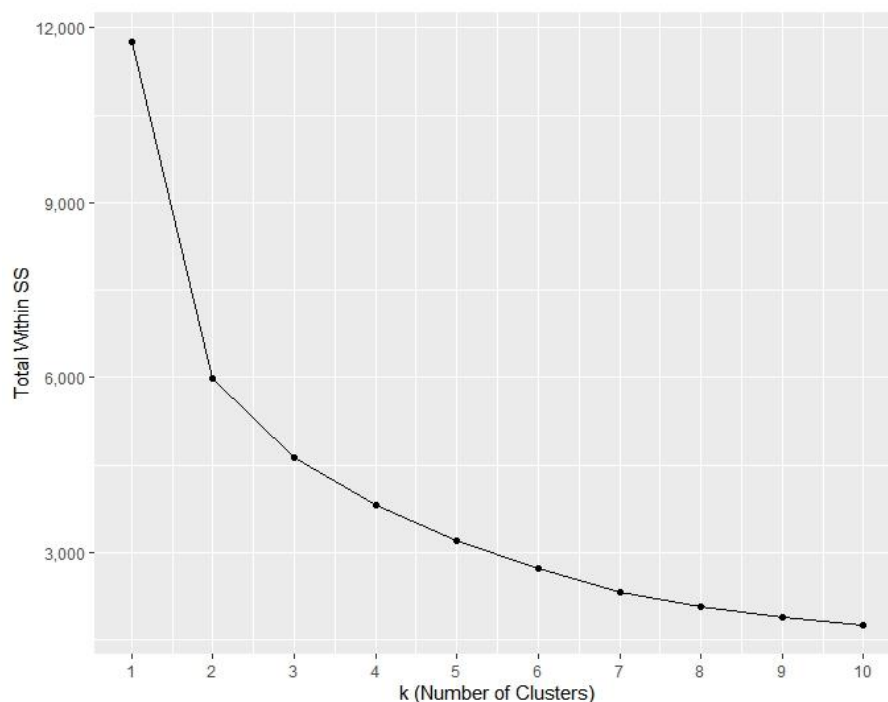
Using NbClust metrics to determine number of clusters

Lastly, there is an R package that will look at a host of different fit indices and, using majority rule, suggest the number of clusters that most indices recommend.

Number of Clusters Chosen by Crite
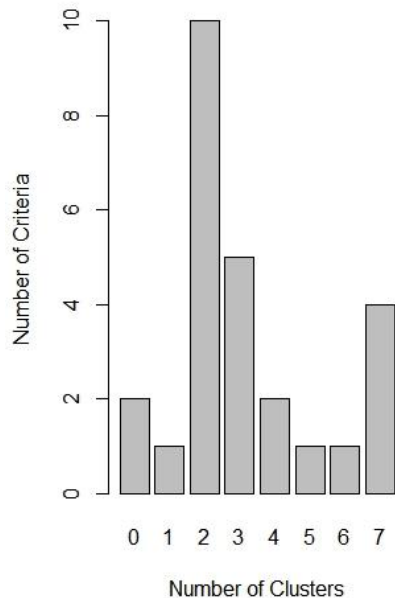
The greatest number of indices recommend the 2-cluster solution.

How to proceed at this point is murky, but authentically so. At this stage, it makes sense to show interested stakeholders the cluster solutions and get their input. The decision should be based upon how the business plans to use the results, and the level of granularity they want to see in the clusters.

CONCLUSION

If the business wants to use the results to understand a range of customer behaviour from high-to-low value customers, I'd probably recommend the 5-cluster solution. I like that it distinguishes the no-value group of customers, whom the business probably wants to eliminate as much as possible, and also separates low-value, low-frequency customers who have purchased recently from those who have not. It may be easier to encourage recently-active customers to re-engage with the business and possibly develop into medium-value customers. That said, there isn't just one correct decision here.

With regards to our business decision on whom to deploy customer loyalty programs: Cluster No. 5 is a high-monetary value, high-frequency, recent purchase group and hence can be identified as a high-valued customer segment and should be the most ideal group to roll out the loyalty program.