Group Project Report

Evan Canfield

7/28/2019

# Data Processing

## R Library Packages

Data processing was primarily done through R. The following libraries were used to process the provided data.

if(!require(pacman)){install.packages("pacman")}  
library(pacman)  
  
p\_load(   
 arules,  
 arulesViz,  
 caTools,  
 dplyr,  
 tidyr,  
 geosphere,  
 lubridate,  
 noncensus,  
 purrr,  
 randomForest,  
 readxl,  
 ROSE,  
 ROCR,  
 tidycensus,  
 stringr  
)

## Data Import

The following were the provided files for Option 2 of the DSBA 6211 Group Project:

* QVC Data 1.xlsx
* QVC Data 2.xlsx
* QVC Data 3.xlsx
* QVCdist\_ctr.xlsx
* QVCorderstatustype.xlsx
* QVC Data Dictionary.xlsx

Before upload and processing in R an error was noticed. The rows listed below had blank value inserted at column Size\_DESC. The remaining data in that row was then shifted over by one column. This data was readjusted manually.

* QVC Data 1: 220464 (#Sales\_Order\_Nbr: 649344509406)
* QVC Data 1: 259651 (#Sales\_Order\_Nbr: 449320937874)
* QVC Data 2: 111684 (#Sales\_Order\_Nbr: 649288839226)
* QVC Data 2: 462659 (#Sales\_Order\_Nbr: 849460177698)
* QVC Data 3: 271093 (#Sales\_Order\_Nbr: 149349466481)
* QVC Data 3: 303069 (#Sales\_Order\_Nbr: 649356909726)

During upload all blank values were treated as NA.

qvc\_data\_1 <- read\_xlsx("./data\_R/input/QVC Data 1.xlsx", na = c("NA", ""))  
  
qvc\_data\_2 <- read\_xlsx("./data\_R/input/QVC Data 2.xlsx", na = c("NA", ""))  
  
qvc\_data\_3 <- read\_xlsx("./data\_R/input/QVC Data 3.xlsx", na = c("NA", ""))  
  
qvc\_distctr <- read\_xlsx("./data\_R/input/QVCdist\_ctr.xlsx", na = c("NA", ""))   
  
qvc\_orderstatustype <- read\_xlsx("./data\_R/input/QVCorderstatustype.xlsx", na = c("NA", ""))

State population data was provided by 2018 National and State Population Estimates from the US Census Bureau. <https://www.census.gov/newsroom/press-kits/2018/pop-estimates-national-state.html>. The data set form was then processed so that only the state code, state (and state equivalents), and 2018 population estimate were in the file used in this analysis.

us\_census\_2018 <- read.csv("./data\_R/input/us\_census\_population\_change\_2018\_reformatted.csv")

A future step involves calculating the distance from shipping origin to destination. To expidite this process, a database of all origin and destination pairs, and corresponding distances between the two, was generated. Further discussion of the calcaultion of distance and the development of this database is occurs at a later step.

zip\_code\_dist\_database <- readRDS("./data\_R/output/Group Project Dataset/zip\_code\_dist\_database.RDS")

The **zip\_codes** data set from the **noncensus** package was used to provide city, state, latitude, and longitude data for the QVC warehouse and the package destination based on the provided zip codes. Additionally, the **zip\_codes** data set provided the Federal Information Processing Standard (FIPS) code for each destination zip code. FIPS codes are unique identifiers for US counties (and county equivalents).

data("zip\_codes")

The **fips\_codes** data set from the **tidycensus** package was used to provide state information in relation to each unique FIPS code.

data("fips\_codes")

## Processing The Data

### Creating The Raw Data File

Once uploaded the QVC sales data was combined into a single data frame.

qvc\_data <- qvc\_data\_1 %>%   
 bind\_rows(qvc\_data\_2) %>% #Bind QVC Data 1 to QVC Data 2  
 bind\_rows(qvc\_data\_3) #Bind QVC Data 3

Note, that for ease moving forward, the **#Sales\_Order\_Number** variable was renamed to **Sales\_Order\_Number**. Furthermore, **SHIP\_TO-ZIP** was renamed **zip\_dest** to be consistent with data that will be added to the data frame during a later step.

qvc\_data <- qvc\_data %>%   
 rename(Sales\_Order\_Nbr = '#Sales\_Order\_Nbr',  
 zip\_dest = SHIP\_TO\_ZIP)

After studying the project questions and the provided data, the following variables were determined to not be relevant to the analysis were then dropped from the data frame. Note that the information from **SHIP\_TO\_STATE** is important to the analyzing the data, but will be repopulated through a different data source during a later step in the process.

drop\_variables = c(   
 "Sales\_Order\_Line\_Nbr",  
 "Order\_Type\_Cd",  
 "Shipping\_Priority\_Ind",  
 "Line\_Status\_Dt",  
 "Skn\_Id", "Sku\_Id",  
 "Color\_Desc","Size\_Desc",  
 "Assigned\_Dc\_Id",  
 "Cancelled\_Qty",  
 "Merchandise\_Div\_Desc",  
 "Carrier\_Used\_Tracking\_Id",  
 "Shipment\_Status\_Dt",  
 "Pickup\_Dt",  
 "Scheduled\_Delivery\_Dt",  
 "Package\_Scan\_Dttm",  
 "Package\_Cnt",  
 "SHIP\_TO\_CITY",  
 "SHIP\_TO\_STATE"  
 )  
qvc\_data <- qvc\_data %>%   
 select(-drop\_variables)

### Modifying the Data

As the provided data was in *.XLSX* form not all variables had the correct data types after upload. The following modifications were necessary.

#### Changing Data Types

The variable **Merchandise\_Dept\_Desc** was imported as a character, but functions as a factor. **Sales\_Order\_Nbr** , **Party\_Id** , **Product\_Id** , and **Package\_Id**  are all identification codes and although are comprised a numbers, these variables function as a character string.

qvc\_data <- qvc\_data %>%   
 mutate(  
 Merchandise\_Dept\_Desc = factor(Merchandise\_Dept\_Desc),  
 Sales\_Order\_Nbr = as.character(Sales\_Order\_Nbr),  
 Party\_Id = as.character(Party\_Id),  
 Product\_Id = as.character(Product\_Id),  
 Package\_Id = as.character(Package\_Id)  
 )

#### Leading Zeros

In several instances the leading zeros for data employing unique numerical codes were dropped during upload. The variables zip\_dest and Source\_Ship\_Warehouse\_Nbr are both numeric identifiers, zip\_dest five digit, and Source\_Ship\_Warehouse\_Nbr four digit. During uploaded from the *.XLSX* file, these variables were considered numeric and any leading zeros in the code were dropped. To properly use these variables the leading zeros need to be reinserted, and the variables converted to character type, in both the QVC Data and QVC Distribution Center data sets. Additionally, leading zeros for FIPS code from the **zip\_codes** were also dropped, and therefore needed to be reinserted.

# Source\_Ship\_Warehouse\_Nbr - QVC Data  
qvc\_data$Source\_Ship\_Warehouse\_Nbr <- str\_pad(string = qvc\_data$Source\_Ship\_Warehouse\_Nbr,   
 width = 4,   
 side = "left",   
 pad = "0")  
  
# Source\_Ship\_Warehouse\_Nbr - QVC Disttribution Center  
qvc\_distctr$Source\_Ship\_Warehouse\_Nbr <- str\_pad(string = qvc\_distctr$Source\_Ship\_Warehouse\_Nbr,   
 width = 4,   
 side = "left",   
 pad = "0")  
  
  
# zip\_dest  
qvc\_data$zip\_dest <- str\_pad(string = qvc\_data$zip\_dest,   
 width = 5,   
 side = "left",   
 pad = "0")  
  
# FIPS  
zip\_codes$fips <- str\_pad(string = zip\_codes$fips,   
 width = 5,   
 side = "left",   
 pad = "0")

### Connecting Data Sets

The data set was then combined with two other data sets, the QVC Distribution Center data and the **zip\_codes** data set. Joining with the QVC Distribution Center data provided further distribution center data for each sales item.

qvc\_data <- qvc\_data %>%   
 left\_join(  
 select(qvc\_distctr,Source\_Ship\_Warehouse\_Nbr,POSTL\_CD),  
 by = "Source\_Ship\_Warehouse\_Nbr"  
 ) %>%   
 rename(zip\_distctr = POSTL\_CD)

The **zip\_codes** data set was joined to provide city, state, latitude, and longitude data for the QVC warehouse and the package destination based on the provided zip codes, as well as FIPS codes based on the destination zip codes. Variables were renamed to provide a clean and consistent naming convention.

qvc\_data <- qvc\_data %>%  
   
 #Join zip\_codes to Distribution Center zip code  
 left\_join(  
 select(zip\_codes, zip:longitude),  
 by = c("zip\_distctr" = "zip")  
 ) %>%   
   
 #Rename Distribution Center Variables  
 rename(  
 city\_distctr = city,  
 state\_distctr = state,  
 lat\_distctr = latitude,  
 lon\_distctr = longitude  
 ) %>%   
   
 #Join zip\_codes to Desitnation zip code  
 left\_join(zip\_codes  
 , by = c("zip\_dest" = "zip")) %>%   
   
 #Rename Destination Columns  
 rename(  
 city\_dest = city,  
 state\_dest = state,  
 lat\_dest = latitude,  
 lon\_dest = longitude,  
 fips\_dest = fips  
 )

The data set variables were then reordered to a more logical arrangement.

qvc\_data <- qvc\_data %>%   
 select(Sales\_Order\_Nbr, Package\_Id,Party\_Id:Product\_Id, Ordered\_Qty, Shipped\_Qty,   
 Actual\_Total\_Package\_Qty, Merchandise\_Dept\_Desc, Order\_Dt,  
 Shipped\_Dt, Rescheduled\_Delivery\_Dt, Delivery\_Confirmation\_Dt, Source\_Ship\_Warehouse\_Nbr,  
 city\_distctr, state\_distctr, zip\_distctr, lat\_distctr, lon\_distctr, zip\_distctr,  
 city\_dest, state\_dest, zip\_dest, lat\_dest, lon\_dest, fips\_dest  
 )

## New Variables

Several new variables were developed for use in analyzing the provided QVC data.

### Rescheduled

The **Rescheduled\_Delivery\_Dt** variable has a high missing data rate, with 40% of the data observations recorded as NA. The description of this variable indicates that a value would only be recorded if the package was rescheduled, so a shipment that was not rescheduled would be indicated by NA. It was assumed that all of the NA values were shipments that were not rescheduled. Using this assumption, a new binary variable was developed, **Rescheduled**. The binary indicates whether a shipment was rescheduled (1) or was not (0). The **Rescheduled\_Delivery\_Dt** variable was then dropped.

qvc\_data <- qvc\_data %>%   
 mutate(Rescheduled = if\_else(is.na(Rescheduled\_Delivery\_Dt), 0, 1)) %>%   
 select(-Rescheduled\_Delivery\_Dt)

### Fulfillment\_Days

The variable **Fulfillment\_Days** is the length of time, in days, between **Order\_Dt** and **Delivery\_Confirmation\_Dt** . The span between **Shipped\_Dt** and **Delivery\_Confirmation\_Dt** was initially considered as well, but after inspection, quality issues regarding the **Shipped\_Dt** values lead to the decision not to use **Shipped\_Dt**. The span of time between dates was calculated, in part, using the **lubridate** package.

qvc\_data <- qvc\_data %>%   
 mutate(Fulfillment\_Days = as.double(difftime(time1 = ymd(Delivery\_Confirmation\_Dt)  
 , time2 = ymd(Order\_Dt)  
 , units = "days"))  
 )

### Distance

The distance between distribution center and shipping destination was calculated for each observation. The location of the distribution center and shipping destination was approximated as the central latitude and longitude of the zip codes of those locations, as provide by the **zip\_codes** data set. The **geosphere** package was used to calculate the distance, using the latitude and longitude of the distribution center and shipping destination as the starting and end point, and using the Haversine method to account for the curvature of the earth. The function distm() returns the distance in meters, so a conversion factor of 0.000621371 miles/meter was also used in order to express the distance in miles.

The calculation of distance is a computationally taxing process. To minimize the process time of this step on possible future runs of the code, a database of origin and destination zip codes was developed, with the distance calculated between the points.

# #Zip Code Distance Database  
# zip\_code\_dist\_database <- qvc\_data %>%  
# select(zip\_distctr:lon\_distctr, zip\_dest:lon\_dest) %>%  
# distinct() %>%  
# rowwise() %>%  
# mutate(Distance = round((distm(x = c(lon\_distctr, lat\_distctr)  
# , y = c(lon\_dest, lat\_dest)  
# , fun = distHaversine) \* 0.000621371), 0)  
# ) %>%  
# ungroup() %>%  
# select(zip\_distctr, zip\_dest, Distance)  
#   
# saveRDS(object = zip\_code\_dist\_database  
# , file = "./data\_R/output/Group Project Dataset/zip\_code\_dist\_database.RDS")

qvc\_data <- qvc\_data %>%   
 left\_join(zip\_code\_dist\_database  
 , by = c("zip\_distctr", "zip\_dest"))

## Data Cleaning

There are currently missing values in the following variables:

* **Shipped\_Dt**
* **Delivery\_Confirmation\_Dt**
* **lat\_distctr**
* **lon\_distctr**
* **lat\_dest**
* **lon\_dest**
* **Fulfillment\_Days**
* **Distance**

None of above variables are have a rate of missing data high enough to imply poor quality of the data, and therefore require dropping the variable. Therefore, the missing values must be imputed or the associated observations dropped.

**Delivery\_Confirmation\_Dt** and **Fulfillment\_Days** have the largest number of missing values. As **Fulfillment\_Days** is a calculated variable based on **Delivery\_Confirmation\_Dt**, all of the missing values in the former are due to the missing values in the latter. **Delivery\_Confirmation\_Dt** has a missing value rate of 4.6%. This missing rate is considered acceptable with such a large data set.These missing values are dropped.

qvc\_data <- qvc\_data %>%   
 drop\_na(Delivery\_Confirmation\_Dt)

Dropping the **Delivery\_Confirmation\_Dt** also drops the missing **Shipped\_Dt** values. The only renaming missing values are the missing latitude and longitude values, as well as the corresponding missing distance values, which is a calculated value based on latitude and longitude. The missing data is due to the **zip\_codes** data set being based on zip codes from the 2010 census. This was the most current data documenting all US zip codes, with corresponding geographical data, that could be found. New zip codes have been established since 2010. Distribution centers and shipping destinations from these zip codes are what is causing the missing values. As the missing **Distance** values are only 0.78% of the data set, these missing values are dropped.

qvc\_data <- qvc\_data %>%   
 drop\_na(Distance)

Inspection of the data shows that there are some observations which have a negative **Fulfillment\_Days** value. This would me the **Delivery\_Confirmation\_Dt** pre-dates the **Order\_Dt**. This is considered a quality issue with the data. The number of observations with **Fulfillment\_Days** less that 1 day constitutes 0.05% of the data set. All observations with a **Fulfillment\_Days** of less than one are dropped.

**Note**: All but one instance of a **Fulfillment\_Days** value of less than one originated at Distribution Center 0540. There may be a systematic problem related to this location and should be investigated.

qvc\_data <- qvc\_data %>%   
 filter(Fulfillment\_Days >= 1)

With the data processed and cleaned, the resulting data set is starting input for use in the analysis used for the questions posed in this project.

## Additional Data

### Shape File

For Question 1, one of the visuals created in Tableau was a county choropleth based on total sales in each county within California, Texas, and Florida. While Tableau is able to plot state based data by default, additional information is required to plot county level data. The county level shape files were provided by the US Census Bureau through the 2018 Tiger/Lines Shape File [<https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.2018.html>] (<https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.2018.html>).

### State Information

For additional contextual information, a data set of information on the US states was developed based on available R data sets, with some modification.

# Develop data frame of state name, abbreviations, regions, and dvisions  
state\_df <- data.frame(State = state.name  
 , Abb = state.abb  
 , Region = state.region  
 )  
  
#Re-define Regions from 4 to 5  
  
Southwest <- c("Arizona", "New Mexico", "Texas", "Oklahoma")  
  
state\_df <- state\_df %>%   
 mutate(Region = as.character(Region),   
 Region = if\_else(Region == "North Central", "Midwest", Region),  
 Region = if\_else(Region == "South", "Southeast", Region),   
 Region = if\_else(State %in% Southwest, "Southwest", Region))  
  
# Washington DC  
state\_df\_DC <- data.frame( State = "Washington, D.C.", Abb = "DC", Region = "Southeast")  
  
# Puerto Rico  
state\_df\_PR <- data.frame( State = "Puerto Rico", Abb = "PR", Region = "Southeast")  
  
#Bind Washington DC Dataframe into Main State DF  
state\_df <- rbind(state\_df, state\_df\_DC, state\_df\_PR)

## Question 1

*Does the current distribution network maximize customer penetration (spend)? If not, what should QVC do to increase customer penetration with the current distribution network?*

Not all of the variables included in the main data set are required for Question 1. Therefore, the unnecessary variables are dropped. The following data set was used for all state based Tableau visualizations.

qvc\_data\_Q1\_state <- qvc\_data %>%   
 select(Sales\_Order\_Nbr,  
 Party\_Id,  
 Total\_Line\_Amt,  
 Merchandise\_Dept\_Desc,  
 Source\_Ship\_Warehouse\_Nbr:Distance,  
 -Rescheduled )  
  
write.csv(x = qvc\_data\_Q1\_state, file = "./data\_R/output/q1\_tableau/qvc\_data\_q1.csv", row.names = FALSE)  
  
saveRDS(object = qvc\_data\_Q1\_state, file = "./data\_R/output/q1\_tableau/qvc\_data\_q1.rds")

### County Level Visualizations

For county level visualizations, the calculation of total sales by FIPS code was necessary. Currently the QVC data set only has FIPS code information for areas where a purchase was shipped to (**fips\_dest**). If a county visualization was done only using these FIPS values, any county where no purchases were made would not be present in the data, and any subsequent visualization would should a blank space on the map.

In order to avoid a map with blank spaces, a full list of US FIPS codes is right-joined to the QVC sales data set, ensuring all FIPS code are included in the resultant data set. All FIPS codes in the full list of FIPS codes which do not have a corresponding FIPS code in the sales data set will result in NAs. All NAs are then replaced by 0, indicating no sales were made in those FIPS regions.

The state of each FIPS code is included in the full FIPS list for to act as a filter while creating visuals in Tableau

# Create List of Every5-digit FIPS code  
fips\_codes\_full <- fips\_codes %>%   
 mutate(fips\_code = paste(state\_code, county\_code, sep = "")) %>%   
 distinct() %>%   
 select(fips\_code, state)  
  
# Create Data Frame of QVC Sales and FIPS Code  
qvc\_sales\_by\_fips <- qvc\_data %>%   
 select(Total\_Line\_Amt, fips\_dest)  
  
# Join Total FIPS List to Sales Data and Replace NAs with zero  
qvc\_data\_Q1\_FIPS <- qvc\_sales\_by\_fips %>%   
 right\_join(fips\_codes\_full,   
 by = c("fips\_dest" = "fips\_code")  
 ) %>%   
 mutate(Total\_Line\_Amt = coalesce(Total\_Line\_Amt, 0))

### Sales / Population Linear Regression

In analyzing the relationship between sales within a state and the state’s population, a linear regression analysis was performed. A new data set focused on sales by state was necessary to perform the regression. The data set used to perform the regression needed to include each state’s population and total sales.

First, it was required to join the US Census data set to the state information data set. This was to ensure the census data had the correct information to successfully be joined to the sales data. Additionally, the population variable was renamed to **Population\_2018** for easier comprehension.

us\_census\_2018 <- us\_census\_2018 %>%   
 left\_join(select(state\_df, State, Abb)  
 , by = c("NAME" = "State")  
) %>%   
 rename(Population\_2018 = POPESTIMATE2018)

With the census data prepared, it was then joined to the QVC sales data aggregated by state.

qvc\_data\_stat\_pop <- qvc\_data %>%   
 # Calculate Sales By State  
 group\_by(state\_dest) %>%   
 summarise(Sales\_Per\_State = sum(Total\_Line\_Amt)  
 ) %>%   
   
 #Join State Population Data  
left\_join(us\_census\_2018  
 , by = c("state\_dest" = "Abb")) %>%   
 select(-STATE)

Linear regression was then performed with the new data, comparing **Sales\_Per\_State** to **Population\_2018**.

lm(formula = Sales\_Per\_State ~ Population\_2018  
 , data = qvc\_data\_stat\_pop)

##   
## Call:  
## lm(formula = Sales\_Per\_State ~ Population\_2018, data = qvc\_data\_stat\_pop)  
##   
## Coefficients:  
## (Intercept) Population\_2018   
## 3.371e+05 4.212e-01

## Question 2

*Are there specific products or product categories that should be located in specific distribution centers?*

Analysis of product categories was performed using the final, complete data set, **qvc\_data**, in Tableau.

For specific product analysis, Association Rule mining was performed. First, a new data frame was developed listing only **Sales\_Order\_Nbr** and **Product\_Id** .

qvc\_data\_am <- qvc\_data %>%   
 select(Sales\_Order\_Nbr,  
 Product\_Id)

In order to perform Association Mining with the **Arules** package, the input data needs to be in the form of a transactions object, not a standard data frame. To convert the data frame to a transaction object, the data frame is exported as a *.csv* file and then re-imported, using the read.transactions function.

write.csv(x = qvc\_data\_am  
 , file = "./data\_R/output/q2\_association\_mining/qvc\_data\_sales-and-prod.csv"  
 , row.names = FALSE)  
  
qvc\_sales\_trans <- read.transactions(file = "./data\_R/output/q2\_association\_mining/qvc\_data\_sales-and-prod.csv"  
 , format = "single"  
 , cols = c(1,2)  
 , sep = ","  
 , rm.duplicates = TRUE  
 , skip = 1)

With the data now in a transactions object form, frequent sets analysis and rules mining analysis are performed.

### Frequent Sets

# Frequent Sets  
frqsets.qvc <- apriori(data = qvc\_sales\_trans  
 , parameter=list(minlen=2  
 , supp=1e-5  
 , conf=0.5  
 , target="frequent itemsets")  
 , control = list(verbose = FALSE))

# Rules Mining  
rules.qvc <- apriori(data = qvc\_sales\_trans  
 , control = list(verbose=FALSE)  
 , parameter = list(minlen=2  
 , supp = 1e-7  
 , conf=0.5))   
# Determine and Prune Redundant Rules  
redundant <- which (colSums(is.subset(rules.qvc, rules.qvc)) > 1)  
  
rules.qvc.pruned <- rules.qvc[-redundant]

## Question 3

*Do customers that receive their product sooner purchase more than customers with longer delivery times?*

# Model idea

’’’ group by party id label customers as repeat or not repeat group by party id and take the min of the order date - representing their first order train a model on this, using the repeat labels as a target’’’

glimpse(qvc\_data)

## Observations: 2,411,751  
## Variables: 29  
## $ Sales\_Order\_Nbr <chr> "949055704509", "349033174553", "549...  
## $ Package\_Id <chr> "2896512278", "2894108123", "2908986...  
## $ Party\_Id <chr> "15560150", "35929840", "34759160", ...  
## $ Total\_Line\_Amt <dbl> 32.22, 15.96, 32.42, 44.25, 45.52, 3...  
## $ Unit\_Price\_Amt <dbl> 26.36, 12.97, 30.92, 36.24, 39.84, 2...  
## $ Line\_Status\_Cd <chr> "VS", "VS", "VS", "VS", "VS", "VS", ...  
## $ Product\_Id <chr> "39200038", "9227492", "272300789", ...  
## $ Ordered\_Qty <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
## $ Shipped\_Qty <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...  
## $ Actual\_Total\_Package\_Qty <dbl> 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, ...  
## $ Merchandise\_Dept\_Desc <fct> Health/Beauty, Apparel, Apparel, Hom...  
## $ Order\_Dt <dttm> 2016-08-13, 2016-08-05, 2016-09-26,...  
## $ Shipped\_Dt <dttm> 2016-08-14, 2016-08-06, 2016-09-30,...  
## $ Delivery\_Confirmation\_Dt <dttm> 2016-08-19, 2016-08-11, 2016-10-03,...  
## $ Source\_Ship\_Warehouse\_Nbr <chr> "0540", "0530", "0530", "0200", "053...  
## $ city\_distctr <chr> "Rocky Mount", "Lancaster", "Lancast...  
## $ state\_distctr <chr> "NC", "PA", "PA", "VA", "PA", "VA", ...  
## $ zip\_distctr <chr> "27801", "17604", "17604", "23434", ...  
## $ lat\_distctr <dbl> 35.93512, 40.06510, 40.06510, 36.735...  
## $ lon\_distctr <dbl> -77.77076, -76.33563, -76.33563, -76...  
## $ city\_dest <chr> "Gatlinburg", "Islamorada", "Media",...  
## $ state\_dest <chr> "TN", "FL", "PA", "FL", "IL", "OH", ...  
## $ zip\_dest <chr> "37738", "33036", "19063", "32539", ...  
## $ lat\_dest <dbl> 35.73363, 24.91828, 39.91608, 30.757...  
## $ lon\_dest <dbl> -83.47156, -80.63676, -75.40251, -86...  
## $ fips\_dest <chr> "47155", "12087", "42045", "12091", ...  
## $ Rescheduled <dbl> 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, ...  
## $ Fulfillment\_Days <dbl> 6, 6, 7, 7, 6, 5, 6, 6, 5, 14, 7, 5,...  
## $ Distance <dbl> 320, 1077, 51, 704, 600, 330, 367, 6...

# Upload Data

q3\_model <- qvc\_data

# Sort by Date

q3\_model$Order\_Dt <- as.Date(q3\_model$Order\_Dt)

Create a Data Frame of initial orders

q3\_orders\_init <- q3\_model %>%  
 group\_by(Party\_Id) %>%  
 slice(which.min(Order\_Dt))

# Create Repeat Customer Dataset

qvc\_customer\_repeat <- qvc\_data %>%  
 distinct() %>%   
 select(Party\_Id, Sales\_Order\_Nbr) %>%   
 group\_by(Party\_Id) %>%  
 count() %>%   
 mutate(Repeat = if\_else(n > 1, 1, 0)) %>%   
 select(-n)

Creat Data Frame of repeat customers. Join Repeat customer dataframe to init sales

q3\_model\_df <- q3\_orders\_init %>%  
 inner\_join(qvc\_customer\_repeat)

## Joining, by = "Party\_Id"

# Logistic Regression

# #Mean / Median  
# mean(qvc\_data$Fulfillment\_Days)  
#   
# median(qvc\_data$Fulfillment\_Days)  
#   
# # Create New Variable  
# x <- 8  
#   
# q3\_model\_df  
#   
# q3\_model\_df\_log <- q3\_model\_df %>%   
# mutate(Late = if\_else(Fulfillment\_Days >= x, 1, 0))  
#   
# glimpse(q3\_model\_df\_log)

# Random Forest

Convert character to factor

model\_df\_trim <- q3\_model\_df %>%  
 ungroup() %>%   
 select(-c(Shipped\_Dt, Order\_Dt, Delivery\_Confirmation\_Dt, Sales\_Order\_Nbr,Party\_Id, Package\_Id,  
 Product\_Id, Source\_Ship\_Warehouse\_Nbr, zip\_dest, city\_distctr,   
 city\_dest, zip\_distctr, fips\_dest, lat\_distctr, lon\_distctr))  
  
model\_df\_trim <- model\_df\_trim %>%   
 mutate(Line\_Status\_Cd = factor(Line\_Status\_Cd),  
 state\_distctr = factor(state\_distctr),  
 state\_dest = factor(state\_dest)  
 )

# Subset

sample = sample.split(model\_df\_trim$Repeat, SplitRatio = .70)  
df.train = subset(model\_df\_trim, sample == TRUE)  
df.test = subset(model\_df\_trim, sample == FALSE)

#Random Forest Analysis

# rf <- randomForest(as.factor(Repeat)~., data=df.train, importance=TRUE, ntree=1000, do.trace = 50)  
#   
# varImpPlot(rf)  
#   
#   
# mtry <- tuneRF(df.train[-15], as.factor(df.train$Repeat), ntreeTry=1000,  
# stepFactor=0.5, improve=0.01, trace=TRUE, plot=TRUE, do.trace = TRUE)  
# best.m <- mtry[mtry[, 2] == min(mtry[, 2]), 1]  
#   
#   
# rf\_tuned <- randomForest(as.factor(Repeat)~., data=df.train, mtry=best.m,  
# importance=TRUE, ntree=1000, do.trace = 50)  
# varImpPlot(rf\_tuned)  
#   
#   
# pred = predict(rf\_tuned, df.test)  
# print(table(pred, df.test$Repeat))  
#   
# roc.curve(df.test$Repeat, pred, plotit = TRUE,add =TRUE)  
#   
#   
# perf <- performance(pred, "tpr", "fpr")  
# plot(perf)  
# abline(a=0,b=1)