IXIS Data Science Challenge Script

Isabelle Stratton

2024-04-30

---

1. Calling Relevant Libraries and Data

```
# set working directory to the current directory of the script

knitr::opts_knit$set(root.dir = getwd())
```

Set working directory data directory

```
# Calling tidyverse for transforming and representing our data
library('openxlsx') # for excel functionality
library('tidyverse') #For data cleaning and presentation
```

Load libraries

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
# Calling sessionCounts
sessionCounts <- read.csv("raw_data/DataAnalyst_Ecom_data_sessionCounts.csv")
# Calling addsToCart
addsToCart <- read.csv("raw_data/DataAnalyst_Ecom_data_addsToCart.csv")
```

Load data

2. Data Exploration and Cleaning

First we can get a sense of the data and check for null values.

```
# Session Counts:
glimpse(sessionCounts)
```

Session counts data:

```
## Rows: 7,734
## Columns: 6
## $ dim_browser        <chr> "Safari", "Internet Explorer", "Chrome", "Amazon Si~
## $ dim_deviceCategory <chr> "tablet", "desktop", "tablet", "tablet", "mobile", ~
## $ dim_date           <chr> "7/1/12", "7/1/12", "7/1/12", "7/1/12", "7/1/12", "~
## $ sessions           <int> 2928, 1106, 474, 235, 178, 120, 10, 9, 5, 4, 4, 1, ~
## $ transactions       <int> 127, 28, 3, 4, 6, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ QTY                <int> 221, 0, 13, 5, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

```
# Session Counts Summary:
summary(sessionCounts)
```

```
##  dim_browser        dim_deviceCategory   dim_date            sessions
##  Length:7734        Length:7734        Length:7734        Min.   :    0
##  Class :character   Class :character   Class :character   1st Qu.:    3
##  Mode  :character   Mode  :character   Mode  :character   Median :   23
##                                                           Mean   : 1347
##                                                           3rd Qu.:  772
##                                                           Max.   :43559
##   transactions          QTY
##  Min.   :   0.00   Min.   :   0.00
##  1st Qu.:   0.00   1st Qu.:   0.00
##  Median :   0.00   Median :   0.00
##  Mean   :  32.28   Mean   :  58.29
##  3rd Qu.:   9.00   3rd Qu.:  12.00
##  Max.   :1398.00   Max.   :2665.00
```

```
#Check for null values
if (anyNA(sessionCounts)) {
  sessionCounts <- na.omit(sessionCounts) # Removes rows with NA values
  print("NA values found and removed")
} else {
  print("No NA values found")
}
```

```
## [1] "No NA values found"
```

```
# Adds to Cart:
glimpse(addsToCart)
```

Adds to Cart data:

```
## Rows: 12
## Columns: 3
## $ dim_year   <int> 2012, 2012, 2012, 2012, 2012, 2012, 2013, 2013, 2013, 2013,~
## $ dim_month  <int> 7, 8, 9, 10, 11, 12, 1, 2, 3, 4, 5, 6
## $ addsToCart <int> 191504, 217666, 123726, 139803, 186572, 168972, 147619, 135~
```

```
# Adds to Cart Summary:
summary(addsToCart)
```

```
##     dim_year      dim_month       addsToCart
##  Min.   :2012   Min.   : 1.00   Min.   :107970
##  1st Qu.:2012   1st Qu.: 3.75   1st Qu.:132843
##  Median :2012   Median : 6.50   Median :143711
##  Mean   :2012   Mean   : 6.50   Mean   :154173
##  3rd Qu.:2013   3rd Qu.: 9.25   3rd Qu.:184525
##  Max.   :2013   Max.   :12.00   Max.   :217666
```

```
# Check for and handle NA values
if (anyNA(addsToCart)) {
  addsToCart <- na.omit(addsToCart) # Removes rows with NA values
  print("NA values found and removed")
} else {
  print("No NA values found")
}
```

```
## [1] "No NA values found"
```

```
# The Browser and Device categories are characters rather than factors
# so we will switch them for better data visualization later on.
sessionCounts$dim_browser <- factor(sessionCounts$dim_browser, levels
                                    = names(sort(table(sessionCounts$dim_browser),
                                                 decreasing = TRUE)))
sessionCounts$dim_deviceCategory <- factor(sessionCounts$dim_deviceCategory)

# The date is also listed as a character so we'll need to switch that to a date class.
sessionCounts$dim_date <- as.Date(sessionCounts$dim_date, format = "%m/%d/%y")
```

Convert Data Types

Check for Outliers We can calculate the z-score to determine the data's relationship to the mean.

```
# Calculate Z-scores for each metric
# Sessions
sessionCounts$z_scores_sessions <- (sessionCounts$sessions -
                                      mean(sessionCounts$sessions)
                                    ) / sd(sessionCounts$sessions)
# Transactions
```

```r
sessionCounts$z_scores_transactions <- (sessionCounts$transactions -
                                         mean(sessionCounts$transactions)
                                       ) / sd(sessionCounts$transactions)
# QTY
sessionCounts$z_scores_QTY <- (sessionCounts$QTY -
                                 mean(sessionCounts$QTY)
                               ) / sd(sessionCounts$QTY)

#Reveal Outliers

# This dataset will include any category row above 3.5 standard deviations from
# the mean
outliers <- sessionCounts[sessionCounts$z_scores_sessions > 3.5 |
                          sessionCounts$z_scores_transactions > 3.5 |
                          sessionCounts$z_scores_QTY > 3.5, ]

# This dataset will only include a row where ALL categories are
# above 3.5 standard deviations from the mean.
outliers_strict <- sessionCounts[sessionCounts$z_scores_sessions > 3.5 &
                                 sessionCounts$z_scores_transactions > 3.5 &
                                 sessionCounts$z_scores_QTY > 3.5, ]

summary(outliers_strict)
```

```
##             dim_browser dim_deviceCategory    dim_date             sessions
##  Safari           :43   desktop:26         Min.   :2012-07-05   Min.   :14251
##  Chrome           :17   mobile :15         1st Qu.:2013-01-11   1st Qu.:16612
##  Internet Explorer: 0   tablet :19         Median :2013-04-16   Median :21204
##  Edge             : 0                      Mean   :2013-03-20   Mean   :22981
##  Firefox          : 0                      3rd Qu.:2013-06-06   3rd Qu.:27868
##  Safari (in-app)  : 0                      Max.   :2013-06-29   Max.   :42592
##  (Other)          : 0
##   transactions         QTY         z_scores_sessions z_scores_transactions
##  Min.   : 380.0   Min.   : 712.0   Min.   : 3.561    Min.   : 3.537
##  1st Qu.: 481.0   1st Qu.: 876.8   1st Qu.: 4.213    1st Qu.: 4.565
##  Median : 615.5   Median :1103.5   Median : 5.480    Median : 5.933
##  Mean   : 680.2   Mean   :1263.0   Mean   : 5.971    Mean   : 6.592
##  3rd Qu.: 820.0   3rd Qu.:1498.0   3rd Qu.: 7.319    3rd Qu.: 8.014
##  Max.   :1398.0   Max.   :2665.0   Max.   :11.383    Max.   :13.894
##
##   z_scores_QTY
##  Min.   : 3.538
##  1st Qu.: 4.429
##  Median : 5.657
##  Mean   : 6.520
##  3rd Qu.: 7.792
##  Max.   :14.107
##
```

```r
# This data set groups session, transaction, and QTY values by browser and
# category.
aggregated_data <- sessionCounts %>%
  group_by(dim_browser, dim_deviceCategory) %>%
```

```
  summarise(
    Mean_Sessions = mean(sessions),
    Mean_Transactions = mean(transactions),
    Mean_QTY = mean(QTY)
  ) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'dim_browser'. You can override using the
## `.groups` argument.
```

```
# Finding the highest value combination of device and browser
top_sessions <- aggregated_data[which.max(aggregated_data$Mean_Sessions), ]
top_transactions <- aggregated_data[which.max(aggregated_data$Mean_Transactions), ]
top_qty <- aggregated_data[which.max(aggregated_data$Mean_QTY), ]

# Display results
top_sessions
```

```
## # A tibble: 1 x 5
##   dim_browser dim_deviceCategory Mean_Sessions Mean_Transactions Mean_QTY
##   <fct>       <fct>                      <dbl>             <dbl>    <dbl>
## 1 Safari      mobile                    12387.              168.     298.
```

```
top_transactions
```

```
## # A tibble: 1 x 5
##   dim_browser dim_deviceCategory Mean_Sessions Mean_Transactions Mean_QTY
##   <fct>       <fct>                      <dbl>             <dbl>    <dbl>
## 1 Chrome      desktop                    8689.              266.     507.
```

```
top_qty
```

```
## # A tibble: 1 x 5
##   dim_browser dim_deviceCategory Mean_Sessions Mean_Transactions Mean_QTY
##   <fct>       <fct>                      <dbl>             <dbl>    <dbl>
## 1 Chrome      desktop                    8689.              266.     507.
```

There are a few conclusions we can take from this:

- Safari and Desktop users had the highest number of rows where all categories were higher 3.5 standard deviations from the mean
- Safari and mobile were the best combination for a high number of sessions
- Chrome and Desktop were the best combination for a higher number of transactions and quantity.

3. Data Manipulation and Analysis

Now the simplest way to consolidate Session Counts and Adds to Cart is to merge them by date. The most accurate way to do this is by creating a month and year column for Session Counts to be compatible with Adds to Cart.

The mutate function will add new columns to our Session Counts We will simultaneously add an e-commerce conversion rate column for later analysis.

Note that there are some zero values under the Sessions column, which indicates a user may not have interacted with the site. We'll set any NaNs to 0 so they will not contribute to our summations later on.

```
sessionCounts$dim_year <- year(sessionCounts$dim_date)

sessionCounts <- sessionCounts %>%
  mutate(dim_month = month(dim_date, label = TRUE),
         ECR = if_else(sessions > 0, transactions / sessions, 0))

#It could be important to figure out if there are rows where transactions > 0,
#but sessions = 0, this could indicate some kind of error

error_rows <- sessionCounts %>%
  filter(transactions > 0, sessions == 0) # Filter to find errors

error_rows
```

```
##   dim_browser dim_deviceCategory   dim_date sessions transactions QTY
## 1     Maxthon            desktop 2012-08-06        0            2   0
## 2   SeaMonkey            desktop 2012-12-31        0            2   5
##   z_scores_sessions z_scores_transactions z_scores_QTY dim_year dim_month ECR
## 1        -0.3717882            -0.3080454   -0.3154742     2012       Aug   0
## 2        -0.3717882            -0.3080454   -0.2884144     2012       Dec   0
```

```
# Note that under Adds To Cart, the month is written in numerical form,
# for ease of readability, we'll convert this to words. For each numeric month
# we'll go in and switch to an abbreviated and ordered factor like in
# Session Counts

if("dim_month" %in% names(addsToCart) && is.numeric(addsToCart$dim_month)) {
  addsToCart <- addsToCart %>%
    mutate(dim_month = factor(month.abb[dim_month],
                              levels = month.abb,
                              ordered = TRUE))  # Create an ordered factor
                                                # Replace month numbers with abb.
} else {
  print("dim_month column missing or not numeric")
}
# Note that the year column in Session Counts is written as a double data class
# so we will convert the year column in Adds To Cart to match.

addsToCart <- addsToCart %>%
  mutate(dim_year = as.double(dim_year))

glimpse(addsToCart)
```

```
## Rows: 12
## Columns: 3
## $ dim_year   <dbl> 2012, 2012, 2012, 2012, 2012, 2012, 2013, 2013, 2013, 2013,~
## $ dim_month  <ord> Jul, Aug, Sep, Oct, Nov, Dec, Jan, Feb, Mar, Apr, May, Jun
## $ addsToCart <int> 191504, 217666, 123726, 139803, 186572, 168972, 147619, 135~
```

```r
glimpse(sessionCounts)
```

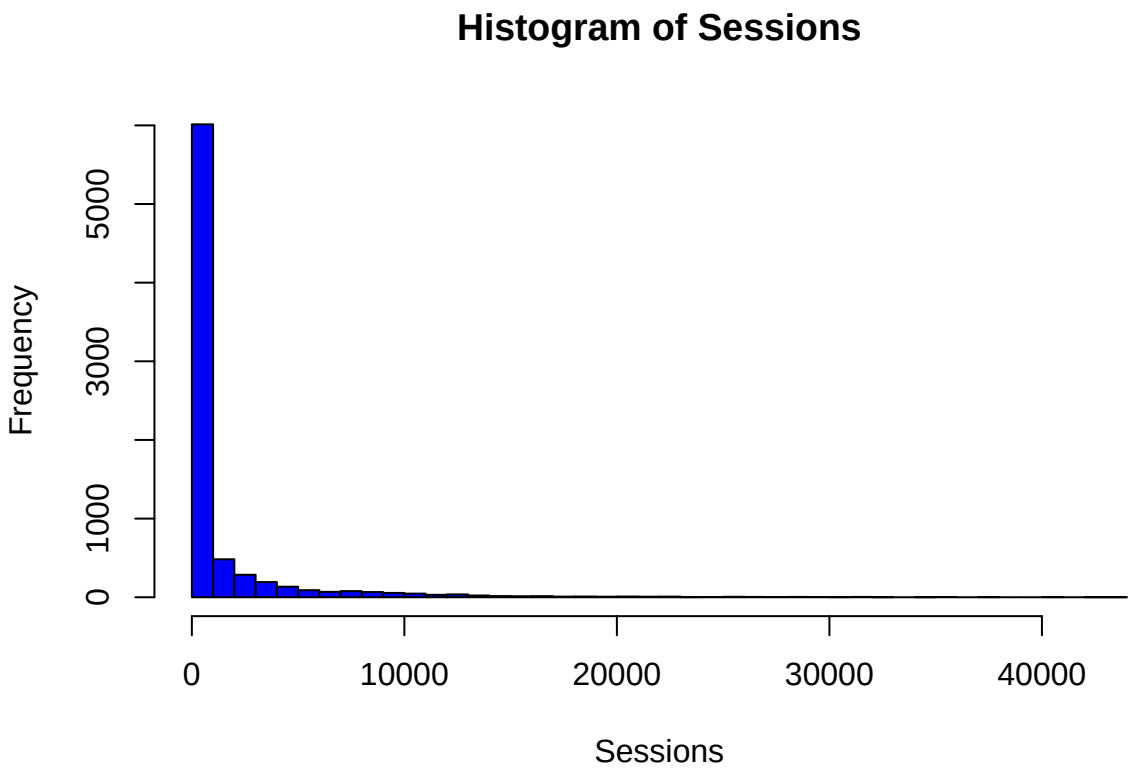
```
## Rows: 7,734
## Columns: 12
## $ dim_browser           <fct> Safari, Internet Explorer, Chrome, Amazon Silk, ~
## $ dim_deviceCategory    <fct> tablet, desktop, tablet, tablet, mobile, tablet,~
## $ dim_date              <date> 2012-07-01, 2012-07-01, 2012-07-01, 2012-07-01,~
## $ sessions              <int> 2928, 1106, 474, 235, 178, 120, 10, 9, 5, 4, 4, ~
## $ transactions          <int> 127, 28, 3, 4, 6, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ QTY                   <int> 221, 0, 13, 5, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ z_scores_sessions     <dbl> 0.43628082, -0.06655450, -0.24097378, -0.3069329~
## $ z_scores_transactions <dbl> 0.9636004, -0.0435431, -0.2978723, -0.2876991, -~
## $ z_scores_QTY          <dbl> 0.8805708, -0.3154742, -0.2451186, -0.2884144, -~
## $ dim_year              <dbl> 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, ~
## $ dim_month             <ord> Jul, Jul, Jul, Jul, Jul, Jul, Jul, Jul, Jul, Jul~
## $ ECR                   <dbl> 0.043374317, 0.025316456, 0.006329114, 0.0170212~
```

```r
# Now we can merge Session Counts and Adds to Cart

combined_data <- full_join(addsToCart, sessionCounts,
                           by = c("dim_month", "dim_year"))


# Now we can explore some relationships in the consolidated data
```

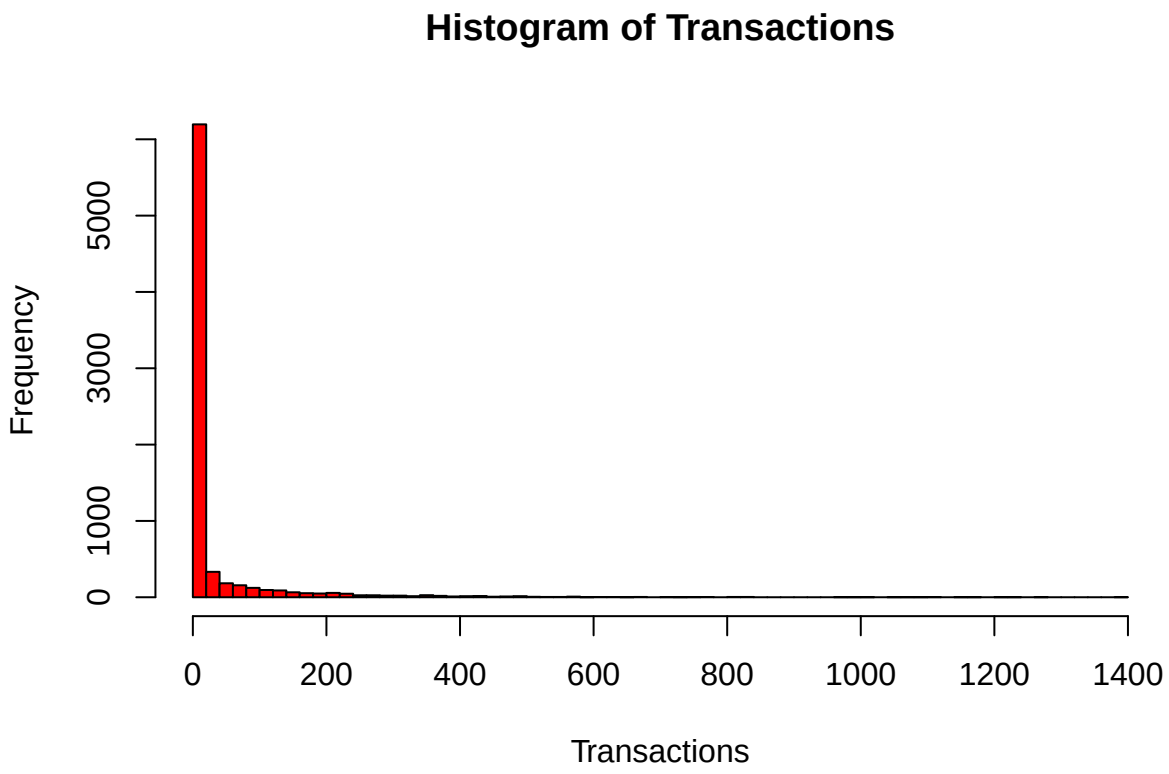
Note that browsers Maxthon and SeaMonkey may deliver unreliable data to Google Analytics because the user would need to have interacted with the sight to have followed through on a transaction.

We'll want to look at each variable individually to get a thorough understanding of the combined data.

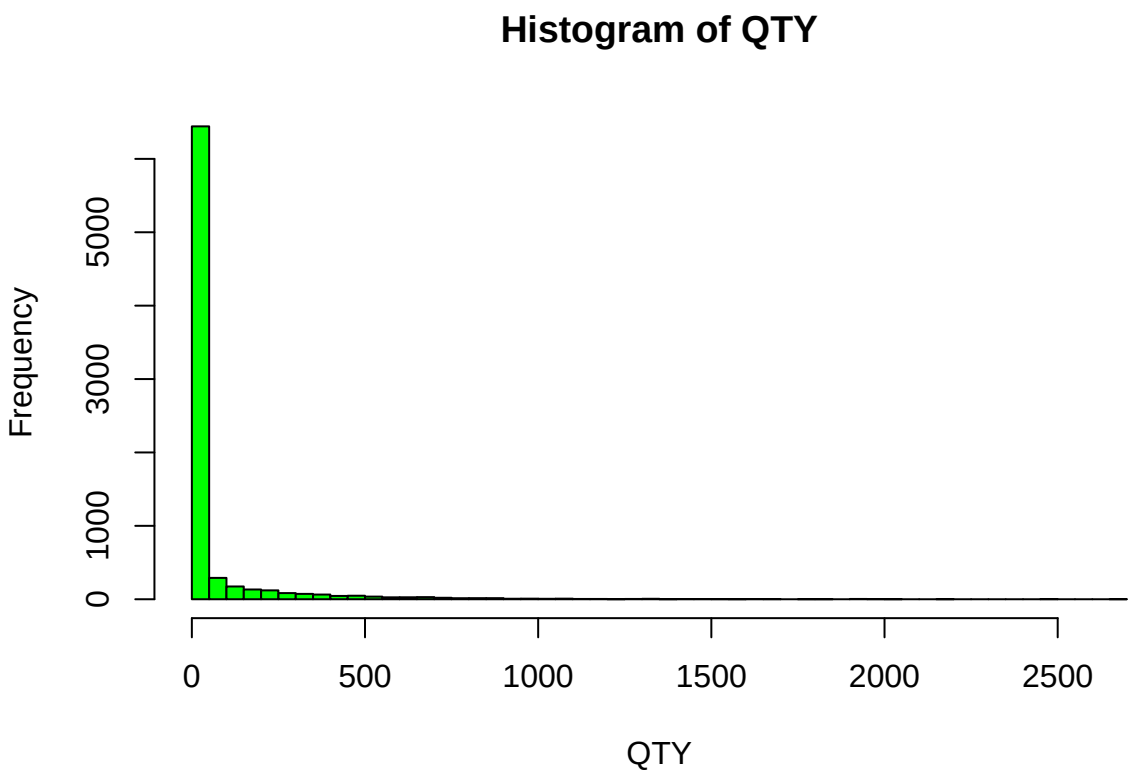
```r
# The first three variables will be depicted with histograms
# with x-axis == value and y-axis == frequency.

# Sessions histogram
hist(combined_data$sessions,
     main="Histogram of Sessions",
     xlab="Sessions",
     col="blue",
     breaks=50)
```



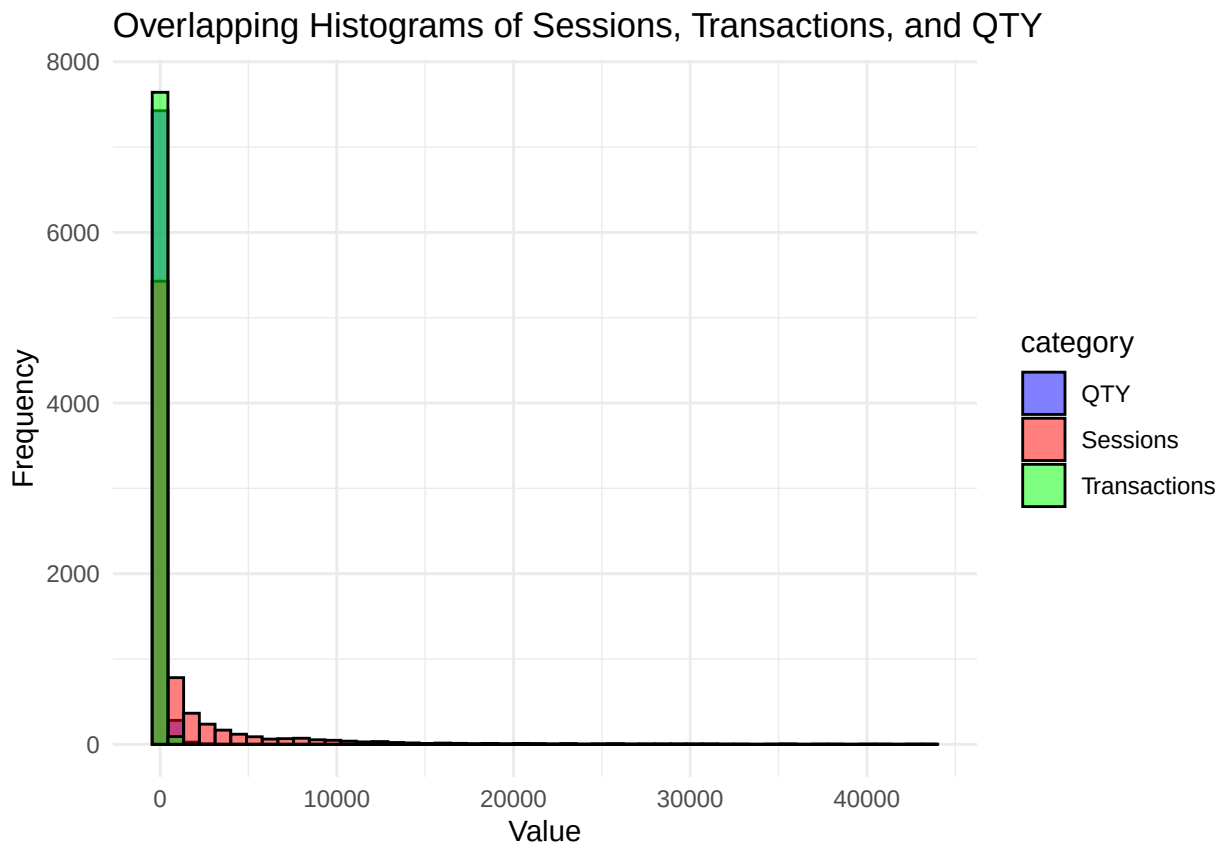

```
# Transactions histogram
hist(combined_data$transactions,
     main="Histogram of Transactions",
     xlab="Transactions",
     col="red",
     breaks=50)
```




```r
# QTY histogram
hist(combined_data$QTY,
     main="Histogram of QTY",
     xlab="QTY",
     col="green",
     breaks=50)
```

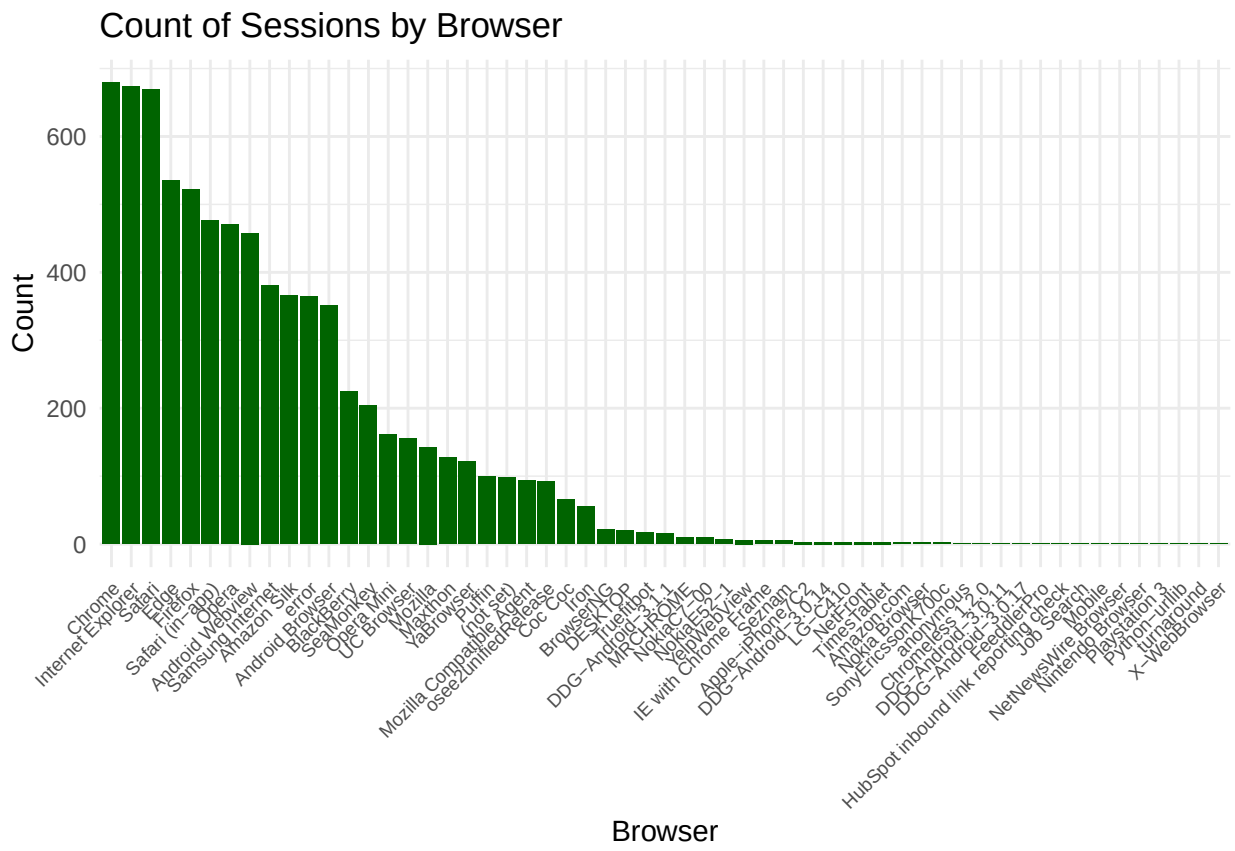



```r
# Create a data frame from the combined data for plotting
longData <- data.frame(value = c(combined_data$sessions, combined_data$transactions,
                                 combined_data$QTY),
                       category = factor(rep(c("Sessions", "Transactions", "QTY"),
                                             each = nrow(combined_data))))

# Using ggplot2 to create overlapping histograms for a better understanding of
# the distribution
ggplot(longData, aes(x = value, fill = category)) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 50, color = "black") +
  scale_fill_manual(values = c("blue", "red", "green")) +
  labs(title = "Overlapping Histograms of Sessions, Transactions, and QTY",
       x = "Value",
       y = "Frequency") +
  theme_minimal()
```



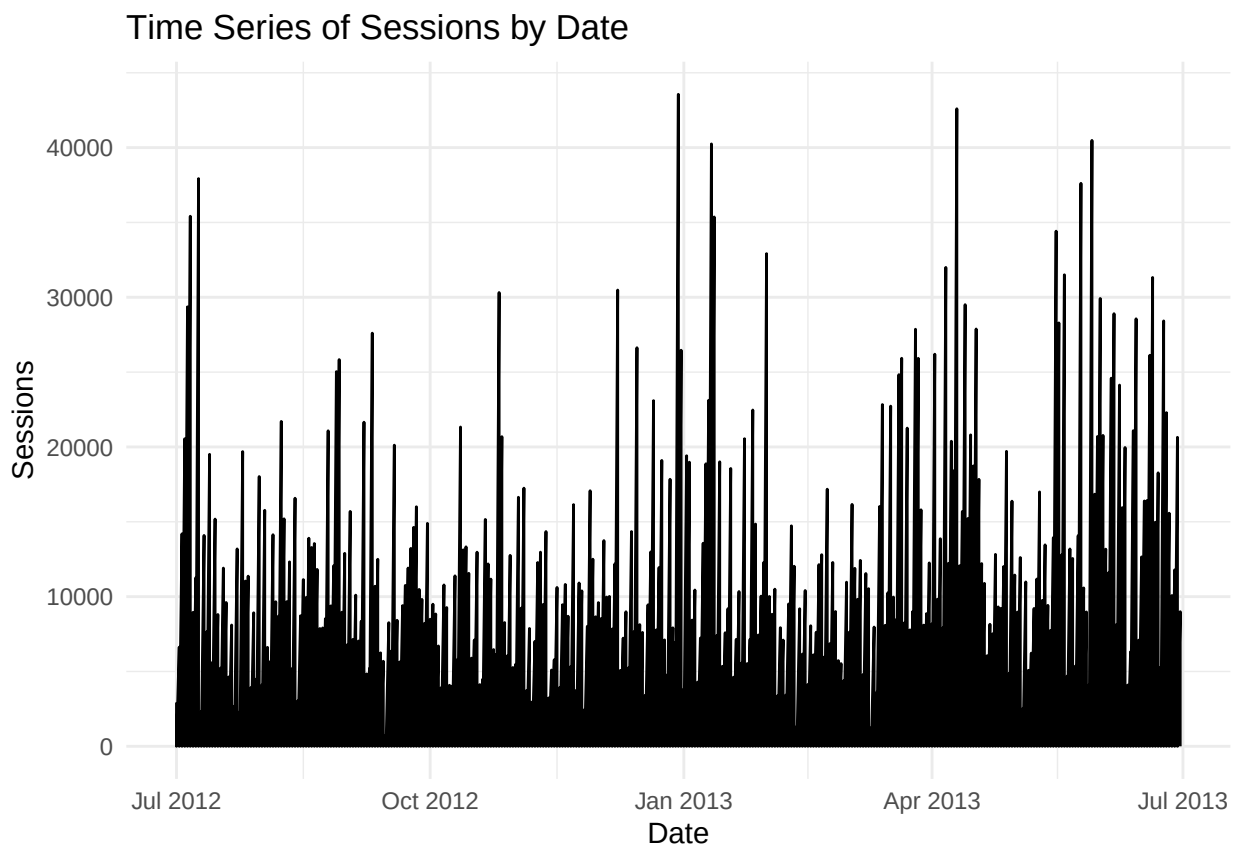
The raw data appears to follow a poisson distribution, because the data is independent and clustered near the origin.

```
# The next three variables will be depicted with histograms where
# x-axis == category and y-axis == frequency

# Bar plot for dim_browser
ggplot(combined_data, aes(x = dim_browser)) +
  geom_bar(fill = "darkgreen") +
  theme_minimal() +
  labs(title = "Count of Sessions by Browser",
       x = "Browser",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1,
                                   family = "Helvetica",
                                     size = 6.5))
```

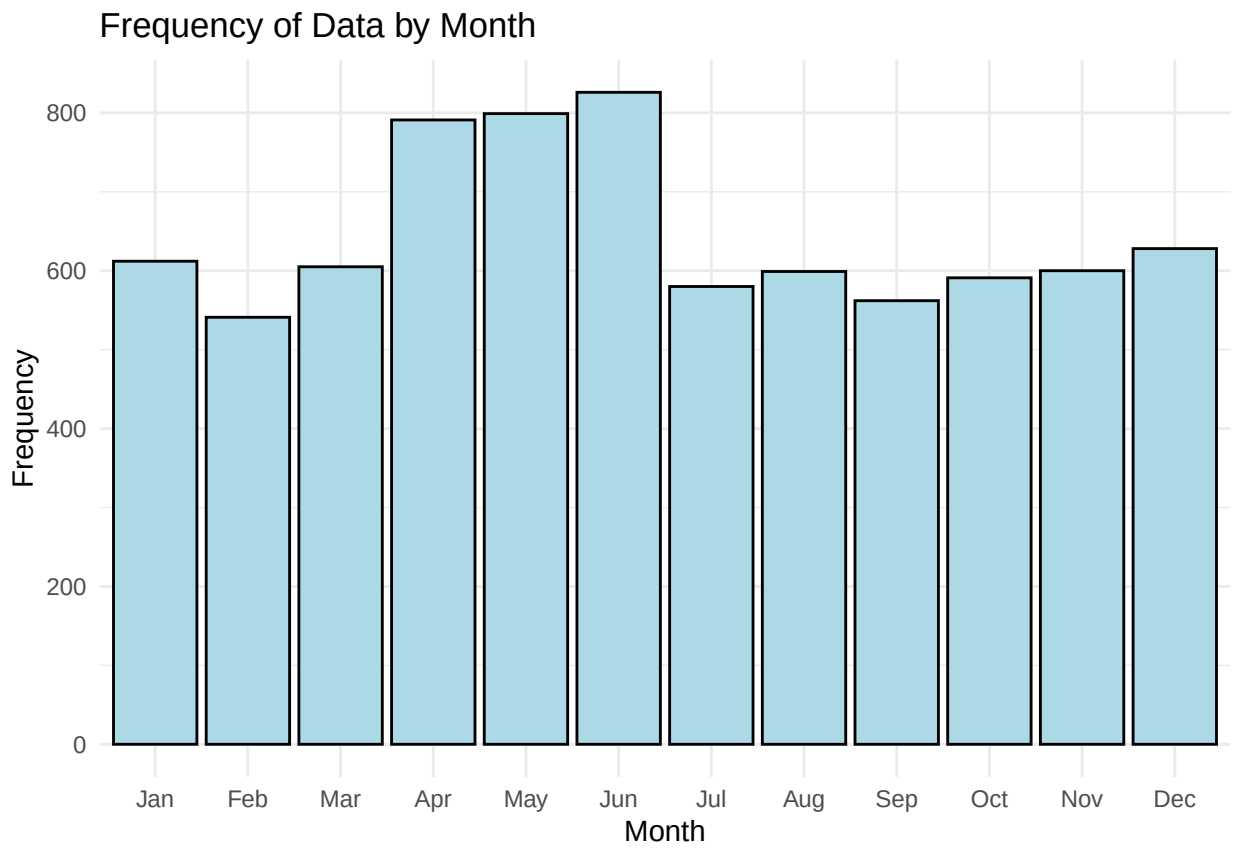



```
# Rotate x labels for better readability
```

```
# Time Series Plot for dim_date
print(ggplot(combined_data, aes(x = dim_date, y = sessions)) +
    geom_line() +
    labs(title = "Time Series of Sessions by Date",
         x = "Date",
         y = "Sessions") +
    theme_minimal())
```




```r
# Bar Plot for Ordinal Variable dim_month
print(ggplot(combined_data, aes(x = dim_month)) +
    geom_bar(fill = "lightblue", color = "black") +
    labs(title = "Frequency of Data by Month",
         x = "Month",
         y = "Frequency") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    theme_minimal())
```

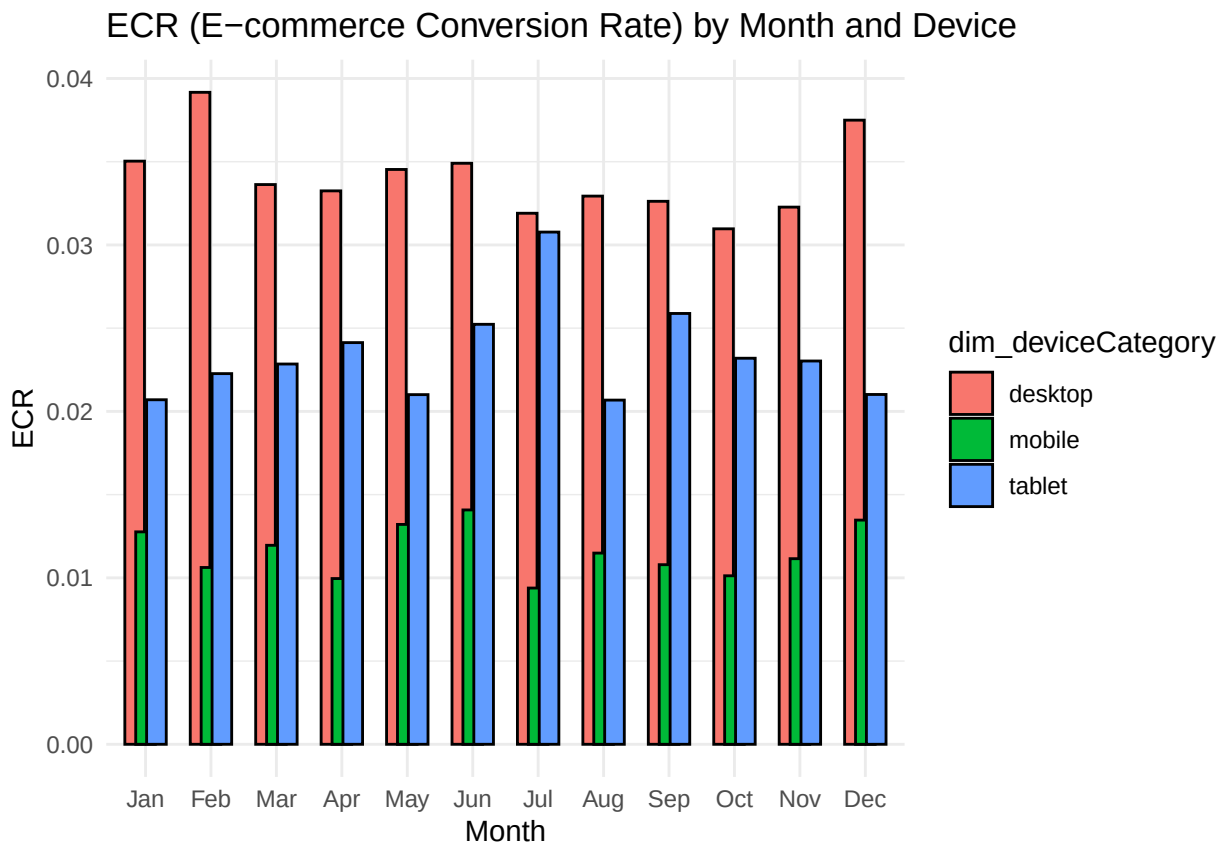


Here we see that March through June were on average ideal for website engagement and that there was an unpredictable timeline of session count through this year.

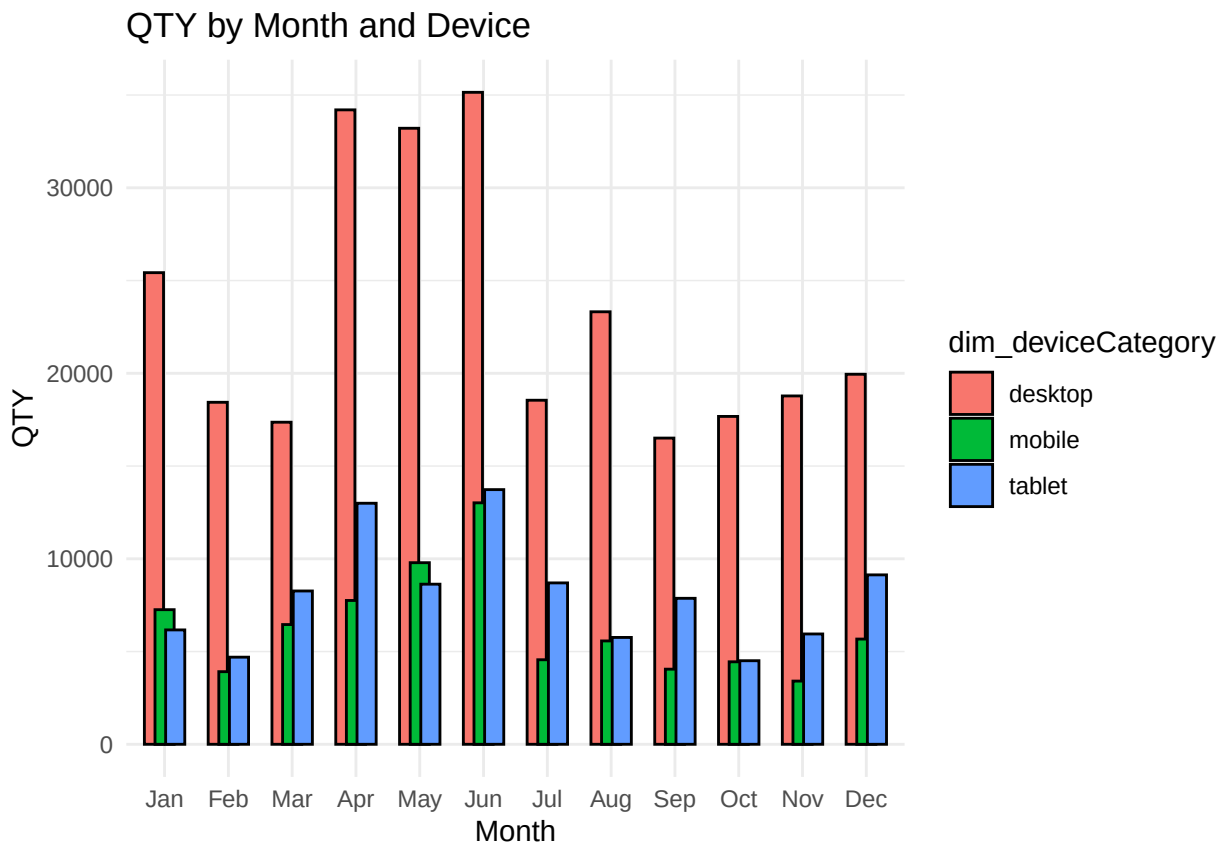
4. Aggregation

```
# Aggregate data by Month and Device for sheet one including Sessions,
# Transactions, QTY, and ECR
monthDevice <- combined_data %>%
  group_by(dim_month, dim_deviceCategory) %>%
  summarise(
    Sessions = sum(sessions), #total number of sessions for each group
    Transactions = sum(transactions), #total number of transactions for each group
    QTY = sum(QTY), #total QTY for each group
    ECR = sum(transactions) / sum(sessions),
    .groups = 'drop'  # This drops the grouping structure
  )

# Plotting Transactions by Month and Device
monDevTrans <- ggplot(monthDevice, aes(x = dim_month, y = Transactions, fill
                                       = dim_deviceCategory)) +
  geom_bar(stat = "identity",  position
           = position_dodge(width = 0.5), color = "black") +
  labs(title = "Transactions by Month and Device", x = "Month",
       y = "Transactions") +
  theme_minimal()
```

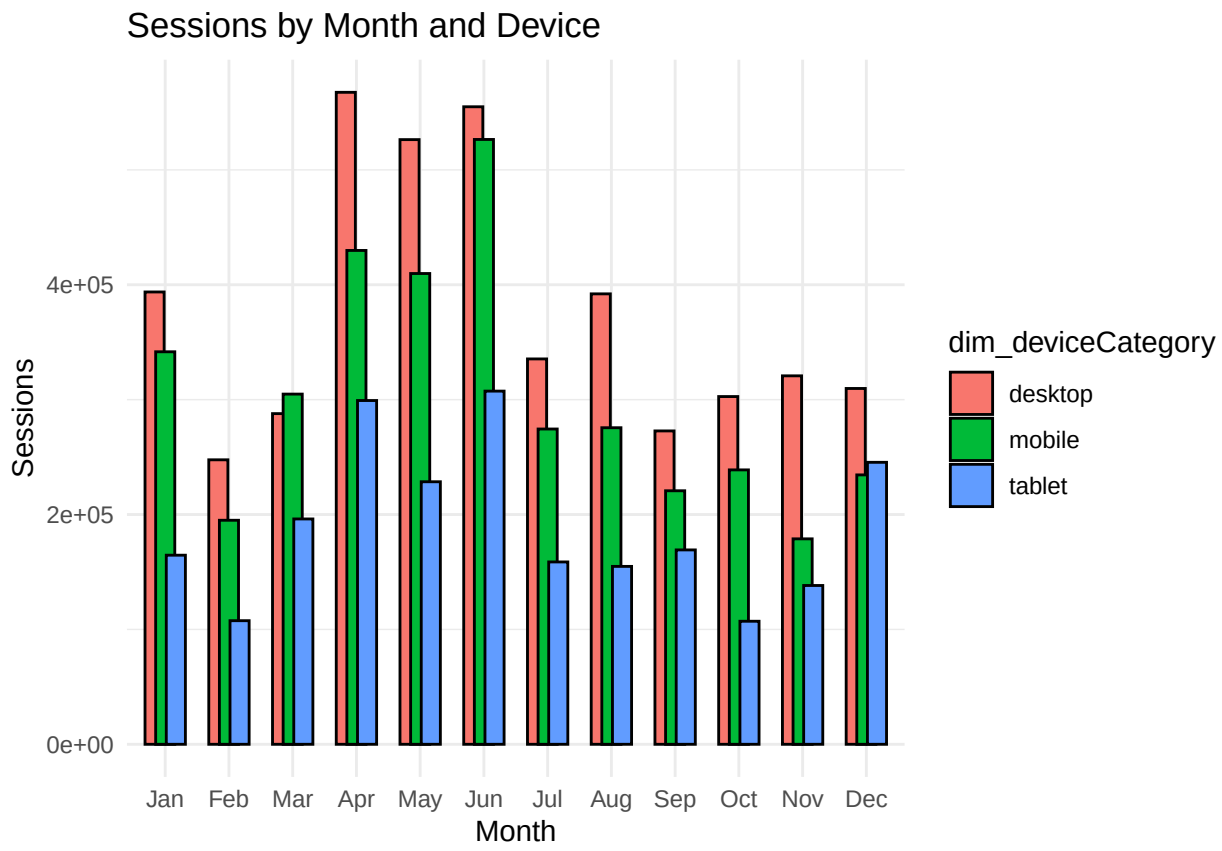
```r
# Plotting QTY by Month and Device
monDevQTY <- ggplot(monthDevice,
                    aes(x = dim_month, y = QTY, fill = dim_deviceCategory)) +
  geom_bar(stat = "identity",
           position = position_dodge(width = 0.5), color = "black") +
  labs(title = "QTY by Month and Device", x = "Month", y = "QTY") +
  theme_minimal()

# Plotting ECR by Month and Device
monDevECR <- ggplot(monthDevice, aes(x = dim_month, y = ECR, fill
                                     = dim_deviceCategory)) +
  geom_bar(stat = "identity",  position = position_dodge(width = 0.5), color
           = "black") +
  labs(title = "ECR (E-commerce Conversion Rate) by Month and Device",
       x = "Month", y = "ECR") +
  theme_minimal()

# Plotting Sessions by Month and Device
monDevSes <- ggplot(monthDevice, aes(x = dim_month, y = Sessions,
                                     fill = dim_deviceCategory)) +
  geom_bar(stat = "identity",  position = position_dodge(width = 0.5),
           color = "black") +
  labs(title = "Sessions by Month and Device", x = "Month", y = "Sessions") +
  theme_minimal()

monDevECR
```


ECR (E−commerce Conversion Rate) by Month and Device
0.04
0.03
0.02
0.01
0.00
ECR
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month
dim_deviceCategory
desktop
mobile
tablet

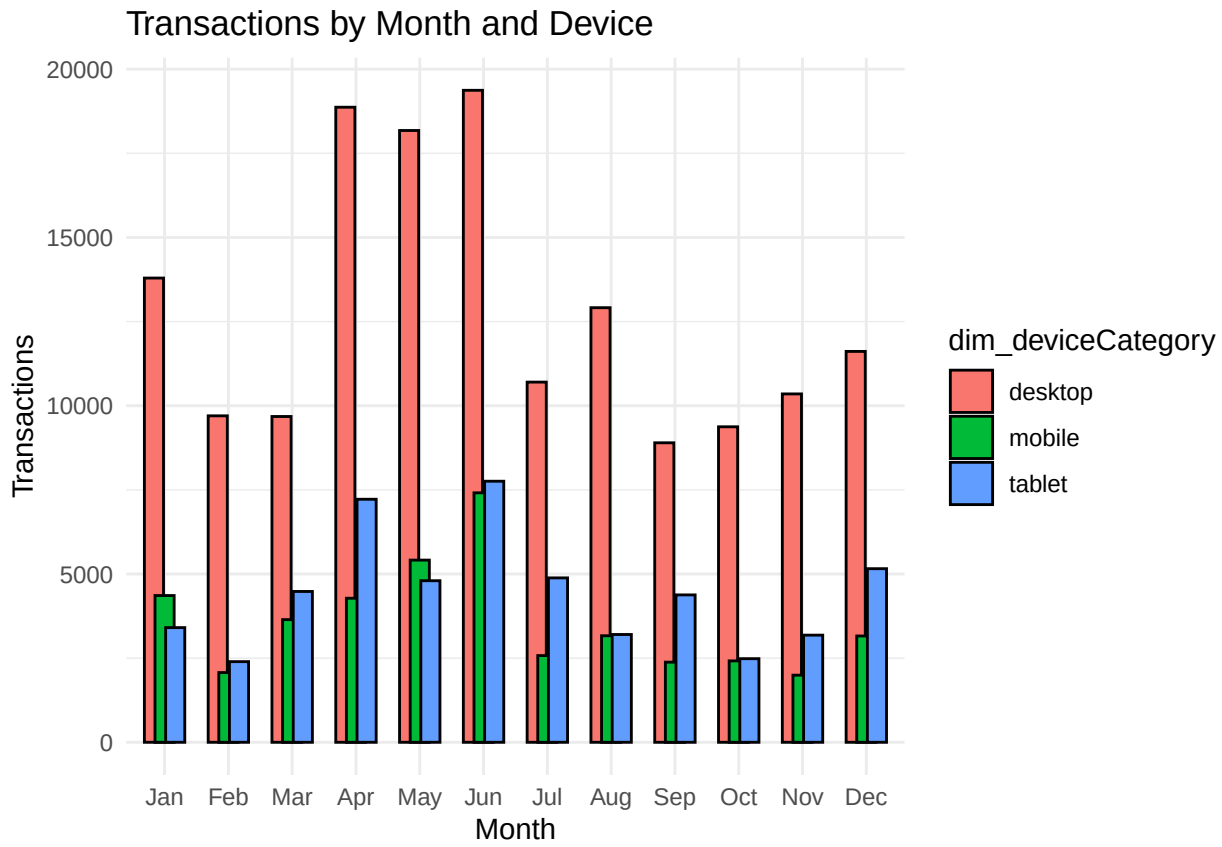
```
monDevQTY
```


QTY by Month and Device
30000
20000
10000
0
QTY
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month
dim_deviceCategory
desktop
mobile
tablet

```
monDevSes
```




```
monDevTrans
```



From this data we glean that the desktop is the most popular device across the board for website engagement. Tablet and phone fight for second place- implying a necessity for more mobile-friendly business adjustments.

```
# Create month over month comparison for second sheet over the two most
# recent months the most recent month's value, the prior month's value,
# and both the absolute and relative differences between them

combined_data <- combined_data %>%
  mutate(date = as.Date(dim_date, format = "%m/%d/%y"))  # Convert to Date object
summary(combined_data)
```

```
##     dim_year      dim_month      addsToCart                dim_browser
##  Min.   :2012   Jun    : 826   Min.   :107970   Chrome           : 679
##  1st Qu.:2012   May    : 799   1st Qu.:123726   Internet Explorer: 673
##  Median :2013   Apr    : 791   Median :139803   Safari           : 669
##  Mean   :2013   Dec    : 628   Mean   :153322   Edge             : 535
##  3rd Qu.:2013   Jan    : 612   3rd Qu.:183842   Firefox          : 522
##  Max.   :2013   Mar    : 605   Max.   :217666   Safari (in-app)  : 476
##                 (Other):3473                    (Other)          :4180
##  dim_deviceCategory    dim_date             sessions      transactions
##  desktop:2672       Min.   :2012-07-01   Min.   :    0   Min.   :   0.00
##  mobile :3013       1st Qu.:2012-10-11   1st Qu.:    3   1st Qu.:   0.00
##  tablet :2049       Median :2013-01-17   Median :   23   Median :   0.00
##                     Mean   :2013-01-11   Mean   : 1347   Mean   :  32.28
##                     3rd Qu.:2013-04-16   3rd Qu.:  772   3rd Qu.:   9.00
##                     Max.   :2013-06-30   Max.   :43559   Max.   :1398.00
##
```

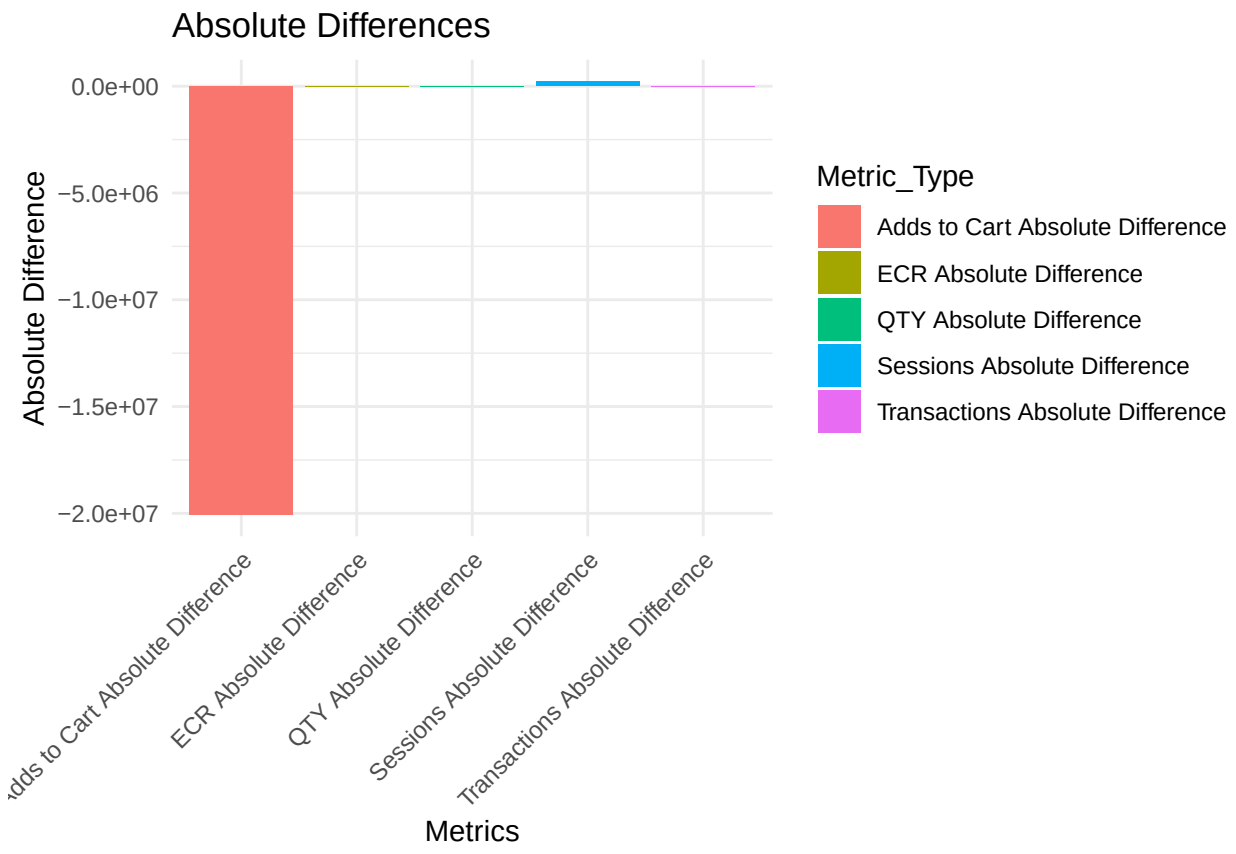
```
##       QTY          z_scores_sessions z_scores_transactions  z_scores_QTY
##  Min.   :   0.00   Min.   :-0.3718   Min.   :-0.3284       Min.   :-0.3155
##  1st Qu.:   0.00   1st Qu.:-0.3710   1st Qu.:-0.3284       1st Qu.:-0.3155
##  Median :   0.00   Median :-0.3654   Median :-0.3284       Median :-0.3155
##  Mean   :  58.29   Mean   : 0.0000   Mean   : 0.0000       Mean   : 0.0000
##  3rd Qu.:  12.00   3rd Qu.:-0.1587   3rd Qu.:-0.2368       3rd Qu.:-0.2505
##  Max.   :2665.00   Max.   :11.6496   Max.   :13.8937       Max.   :14.1074
##
##       ECR               date
##  Min.   :0.00000   Min.   :2012-07-01
##  1st Qu.:0.00000   1st Qu.:2012-10-11
##  Median :0.00000   Median :2013-01-17
##  Mean   :0.01293   Mean   :2013-01-11
##  3rd Qu.:0.01575   3rd Qu.:2013-04-16
##  Max.   :3.00000   Max.   :2013-06-30
##
```

```r
# Filter for May and June 2013, the most recent months in the dataset
momData <- combined_data %>%
  filter(format(date, "%m%Y") %in% c("052013", "062013"))

# Group the metrics by month
momDataN <- momData %>%
  group_by(dim_month, dim_year) %>%
    summarise(
      Total_Sessions = sum(sessions),
      Total_Transactions = sum(transactions),
      Total_QTY = sum(QTY),
      Total_ECR = sum(ECR),
      Total_Adds = sum(addsToCart),
      .groups = 'drop')

# Find the Absolute differences by taking the difference by month
differences <- momDataN %>%
              mutate('Sessions Absolute Difference' = Total_Sessions[2]
                                                    - Total_Sessions[1],
                     'Transactions Absolute Difference' = Total_Transactions[2]
                                                    - Total_Transactions[1],
                     'QTY Absolute Difference' = Total_QTY[2]
                                                    - Total_QTY[1],
                     'ECR Absolute Difference' = Total_ECR[2]
                                                   -Total_ECR[1],
                     'Adds to Cart Absolute Difference'=  Total_Adds[2]
                                                    - Total_Adds[1],
                     )

#Find Relative Differences
momComparison <- differences %>%
              mutate('Sessions Relative Difference' = (Total_Sessions[2]
                                                   - Total_Sessions[1])
                                                       /Total_Sessions[1],
                     'Transactions Relative Difference' = (Total_Transactions[2]
                                                      - Total_Transactions[1])
                                                       /Total_Transactions[1],
```

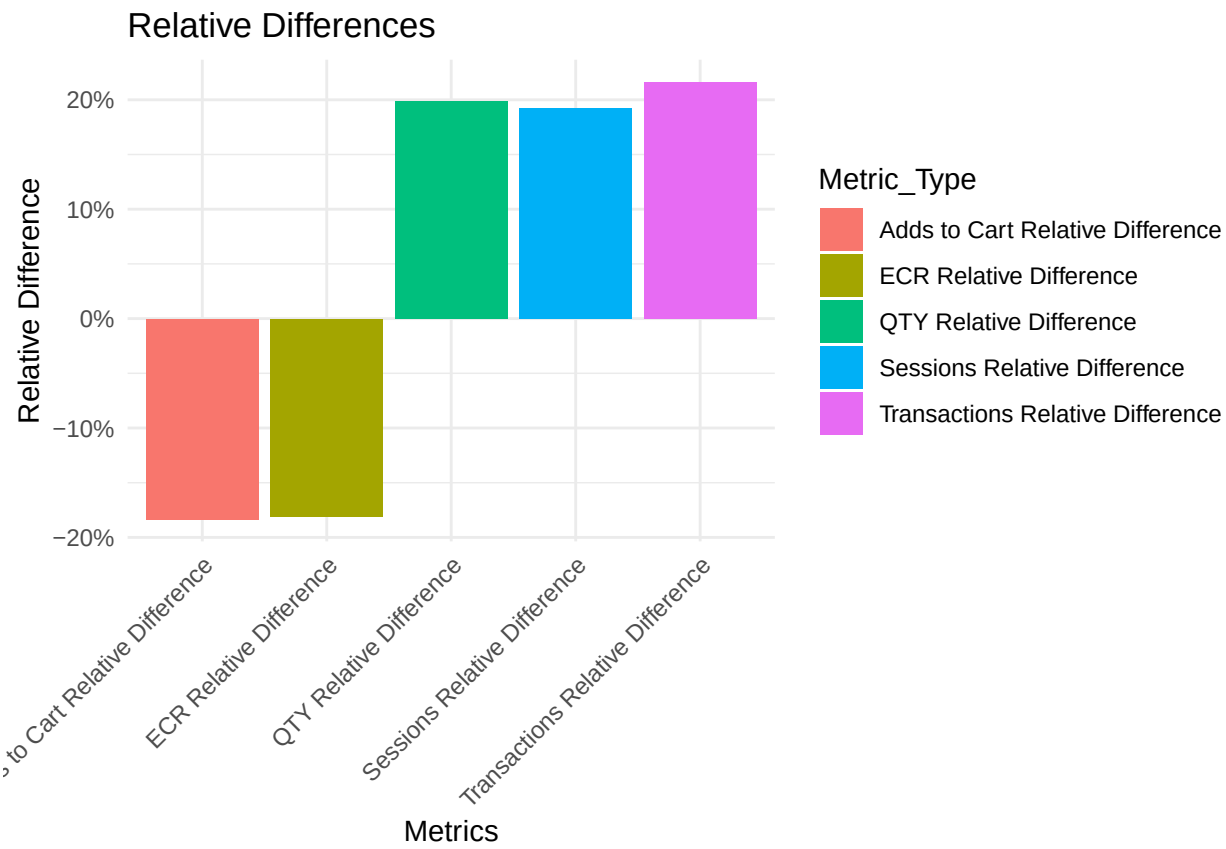
```
                     'QTY Relative Difference' = (Total_QTY[2]
                                                  - Total_QTY[1])
                                                           /Total_QTY[1],
                     'ECR Relative Difference' = (Total_ECR[2]
                                                  -Total_ECR[1])/Total_ECR[1],
                     'Adds to Cart Relative Difference'=  (Total_Adds[2]
                                                           - Total_Adds[1])
                                                                /Total_Adds[1],
                     )
momComparison
```

```
## # A tibble: 2 x 17
##   dim_month dim_year Total_Sessions Total_Transactions Total_QTY Total_ECR
##   <ord>        <dbl>          <int>              <int>     <int>     <dbl>
## 1 May           2013        1164639              28389     51629      13.5
## 2 Jun           2013        1388834              34538     61891      11.1
## # i 11 more variables: Total_Adds <int>, `Sessions Absolute Difference` <int>,
## #   `Transactions Absolute Difference` <int>, `QTY Absolute Difference` <int>,
## #   `ECR Absolute Difference` <dbl>, `Adds to Cart Absolute Difference` <int>,
## #   `Sessions Relative Difference` <dbl>,
## #   `Transactions Relative Difference` <dbl>, `QTY Relative Difference` <dbl>,
## #   `ECR Relative Difference` <dbl>, `Adds to Cart Relative Difference` <dbl>
```

```
# Reshape data using pivot_longer
data_long <- momComparison %>%
  pivot_longer(
    cols = ends_with("Difference"),
    names_to = "Metric_Type",
    values_to = "Difference_Value"
  )
# Split data into absolute and relative for easier plotting
absolute_data <- data_long %>% filter(str_detect(Metric_Type, "Absolute"))
relative_data <- data_long %>% filter(str_detect(Metric_Type, "Relative"))

# Plotting Absolute Differences
ggplot(absolute_data, aes(x = Metric_Type, y = Difference_Value, fill = Metric_Type)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(title = "Absolute Differences",
       x = "Metrics",
       y = "Absolute Difference") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```r
# Plotting Relative Differences
ggplot(relative_data, aes(x = Metric_Type, y = Difference_Value, fill = Metric_Type)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(title = "Relative Differences",
       x = "Metrics",
       y = "Relative Difference") +
  scale_y_continuous(labels = scales::percent_format()) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



We see an uptick in transactions and quantity of items bought but a decrease in adds to cart over the month. This finding is counter-intuitive but may not be a negative trait.

5. Save Output to Disk

Now it's time to create our two sheet Excel file.

```
# To create the desired worksheets for our Excel file:

# Initialize new Workbook
wb <- createWorkbook()

#Add Worksheets
addWorksheet(wb, "Month with Device")
addWorksheet(wb, "Month over Month Comparison")

# Write data to the sheets
writeData(wb, sheet = "Month with Device", monthDevice)
writeData(wb, sheet = "Month over Month Comparison", momComparison)

# Save the workbook
saveWorkbook(wb, "IXIS_Data_Science_Challenge.xlsx", overwrite = TRUE)
```

5. Conclusion

The analysis of user engagement patterns across different browsers and devices highlights several key insights.

- Desktop usage remains the most popular for website interactions, with Safari and Chrome on desktops showing strong performance in engagement and transactions, respectively.
- Seasonal trends indicate that March through June are peak months for user activity. Despite an increase in transactions, there is a notable decline in cart additions, suggesting more decisive purchasing behavior.
- The data's Poisson-like distribution points to the independence of events and their concentration near the origin, providing a basis for further predictive modeling.

These insights should inform strategic enhancements, particularly in optimizing for device-specific experiences and adapting to user behavior trends.