Kira Plastina, Online Shopping Clothing Analysis

Naomi Chebet

9/10/2020

1. Business Understanding

1.1 Defining the Question

Main Research Question: Kira Plastina's brand sales and marketing team is interested in understanding customer behavior. They have collected data for a year. From this data, the team wants to know — what are the characteristics of customer groups that shop at Kira Platina?

1.2 Providing the Context

Kira Plastina is an online clothing shop that sells ready-to-wear designer clothes. According to it's website, Kira Plastina brand targets young girls from the age of 18-25 years and older. The brand first appeared in Russia in 2007 with Kira being the youngest of the Russia high fashion. In 2008, the brand received a "Successful Debut at Milan, Italy, Fashion Week". The successful debut was awarded by Milan Boselli, the President of the Italian National Fashion Chamber.

Some of Kira Platina's collection include: dresses, coats, trousers, tops, and blouses, which are all produced in Russia.

The objective of this project is to analyze the given data and provide the results to the brand and marketing team on features/characteristics of customers that shop at Kira Plastina.

Understanding the traits of your customers is important for any business. In the case of Kira Plastina, the analysis will help inform the team on how to best formulate the marketing and sales strategies of the brand. Further, the insights from the analysis will help the advertisement and other marketing efforts be better targeted to the right customer.

1.1 Metric for Success

a.) Performing intense EDA to get insights on the traits of customers that shop at Kira Plastina.

b.) Successful implementation of K-Means with optimal number of clusters.

b.) Successful implementation of Hierachical Clustering.

d.) Compile a list of the kinds of customers that shop at Kira Plastina, the list will be presented to the brand and marketing team of Kira Plastina.

1.3 Experimental Design

The approach for the project will include:

1. Business Understanding
2. Importing Libraries
3. Loading and Checking the Data
4. Cleaning the Data

5. EDA using Univariate and Bivariate Analysis
6. Implementing the Solution with K-Means and Hierachical Clustering
7. Challenging the Solution
8. Conclusion and Recommendation

1.4 Data Relevance

Here are some of the information provided about the dataset:

a.) The dataset consists of 10 numerical and 8 categorical attributes. The 'Revenue' attribute can be used as the class label.

b.) "Administrative", "Administrative Duration", "Informational", "Informational Duration", "Product Related" and "Product Related Duration" — represents the number of different types of pages visited by the visitor in that session and total time spent in each of these page categories. The values of these features are derived from the URL information of the pages visited by the user and updated in real-time when a user takes an action, e.g. moving from one page to another.

c.) The "Bounce Rate", "Exit Rate" and "Page Value" features represent the metrics measured by "Google Analytics" for each page in the e-commerce site.

Note: it's likely that we won't be using these columns in our analysis as it's specific to Google's Metric of measurement when someone visits a page online.

d.) The value of the "Bounce Rate" feature for a web page refers to the percentage of visitors who enter the site from that page and then leave ("bounce") without triggering any other requests to the analytics server during that session.

e.) The value of the "Exit Rate" feature for a specific web page is calculated as for all page views to the page, the percentage that was the last in the session.

Note: this seems to be Google specific as well

f.) The "Page Value" feature represents the average value for a web page that a user visited before completing an e-commerce transaction.

g.) The "Special Day" feature indicates the closeness of the site visiting time to a specific special day (e.g. Mother's Day, Valentine's Day) in which the sessions are more likely to be finalized with the transaction. The value of this attribute is determined by considering the dynamics of e-commerce such as the duration between the order date and delivery date. For example, for Valentine's day, this value takes a nonzero value between February 2 and February 12, zero before and after this date unless it is close to another special day, and its maximum value of 1 on February 8.

h.) The dataset also includes the operating system, browser, region, traffic type, visitor type as returning or new visitor, a Boolean value indicating whether the date of the visit is weekend, and month of the year.

We will make the assumption that the duration time is in seconds.

2. Importing Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts -----------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(ggcorrplot)
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(factoextra)
```

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

```
library(cluster)
```

3. Loading and Checking the Data

```
#loading the data
shoppers <- read.csv("~/Moringa School/R Programming/R datasets/online_shoppers_intention.csv")
```

```
#previewing the top of the data
head(shoppers)
```

```
##   Administrative Administrative_Duration Informational Informational_Duration
## 1              0                       0             0                      0
## 2              0                       0             0                      0
## 3              0                      -1             0                     -1
## 4              0                       0             0                      0
## 5              0                       0             0                      0
## 6              0                       0             0                      0
##   ProductRelated ProductRelated_Duration BounceRates ExitRates PageValues
## 1              1                0.000000  0.20000000 0.2000000          0
## 2              2               64.000000  0.00000000 0.1000000          0
## 3              1               -1.000000  0.20000000 0.2000000          0
## 4              2                2.666667  0.05000000 0.1400000          0
## 5             10              627.500000  0.02000000 0.0500000          0
## 6             19              154.216667  0.01578947 0.0245614          0
##   SpecialDay Month OperatingSystems Browser Region TrafficType
## 1          0   Feb                1       1      1           1
## 2          0   Feb                2       2      1           2
## 3          0   Feb                4       1      9           3
## 4          0   Feb                3       2      2           4
## 5          0   Feb                3       3      1           4
## 6          0   Feb                2       2      1           3
##         VisitorType Weekend Revenue
## 1 Returning_Visitor   FALSE   FALSE
## 2 Returning_Visitor   FALSE   FALSE
```

```
## 3 Returning_Visitor   FALSE   FALSE
## 4 Returning_Visitor   FALSE   FALSE
## 5 Returning_Visitor    TRUE   FALSE
## 6 Returning_Visitor   FALSE   FALSE
```

```
#previewing the bottom of the data
tail(shoppers)
```

```
##       Administrative Administrative_Duration Informational
## 12325              0                      0             1
## 12326              3                    145             0
## 12327              0                      0             0
## 12328              0                      0             0
## 12329              4                     75             0
## 12330              0                      0             0
##       Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 12325                      0             16                503.000 0.000000000
## 12326                      0             53               1783.792 0.007142857
## 12327                      0              5                465.750 0.000000000
## 12328                      0              6                184.250 0.083333333
## 12329                      0             15                346.000 0.000000000
## 12330                      0              3                 21.250 0.000000000
##        ExitRates PageValues SpecialDay Month OperatingSystems Browser Region
## 12325 0.03764706    0.00000          0   Nov                2       2      1
## 12326 0.02903061   12.24172          0   Dec                4       6      1
## 12327 0.02133333    0.00000          0   Nov                3       2      1
## 12328 0.08666667    0.00000          0   Nov                3       2      1
## 12329 0.02105263    0.00000          0   Nov                2       2      3
## 12330 0.06666667    0.00000          0   Nov                3       2      1
##       TrafficType       VisitorType Weekend Revenue
## 12325           1 Returning_Visitor   FALSE   FALSE
## 12326           1 Returning_Visitor    TRUE   FALSE
## 12327           8 Returning_Visitor    TRUE   FALSE
## 12328          13 Returning_Visitor    TRUE   FALSE
## 12329          11 Returning_Visitor   FALSE   FALSE
## 12330           2       New_Visitor    TRUE   FALSE
```

```
#checking how the data structure looks like using glimpse

glimpse(shoppers)
```

```
## Rows: 12,330
## Columns: 18
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 1, 2, 10, 19, 1, 1, 2, 3, 3, 16, 7, 6...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, -1.000000, 2.666667, 6...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.05...
## $ ExitRates               <dbl> 0.200000000, 0.100000000, 0.200000000, 0.14...
## $ PageValues              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ SpecialDay              <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.0, 0.8...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <int> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3...
## $ Browser                 <int> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2...
```

```
## $ Region                 <int> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3...
## $ TrafficType            <int> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3...
## $ VisitorType            <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FA...
## $ Revenue                <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

The dataset has 12,330 records and 18 columns. The columns have numerical and categorical data types. We will further explore on the unique values in each column below and fix incorrect data types during cleaning.

```
#checking for unique values in all the columns
sapply(shoppers, function(x)length(unique(x)))
```

```
##          Administrative Administrative_Duration           Informational
##                      28                    3337                      18
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                    1260                     312                    9553
##             BounceRates               ExitRates              PageValues
##                    1873                    4778                    2704
##              SpecialDay                   Month        OperatingSystems
##                       6                      10                       8
##                 Browser                  Region             TrafficType
##                      13                       9                      20
##             VisitorType                 Weekend                 Revenue
##                       3                       2                       2
```

```
#checking for unique values in columns of interest

unique(shoppers$Administrative)
```

```
##  [1]  0  1  2  4 12  3 10  6  5  9  8 16 13 11  7 18 14 17 19 15 NA 24 22 21 20
## [26] 23 27 26
```

```
unique(shoppers$Informational)
```

```
##  [1]  0  1  2  4 16  5  3 14  6 12  7 NA  9 10  8 11 24 13
```

```
unique (shoppers$ProductRelated)
```

```
##   [1]   1   2  10  19   3  16   7   6  23  13  20   8   5  32   4  45  14  52
##  [19]   9  46  15  22  11  12  36  42  27  90  18  38  17 128  25  30  21  51
##  [37]  26  28  31  24  50  96  49  68  98  67  55  35  37  29  34  71  63  87
##  [55]  40  33  54  64  75  39 111  81  61  47  44  88 149  41   0  79  66  43
##  [73] 258  80  62  83 173  48  58  57  56  69  82  59 109 287  53  84  78 137
##  [91] 113  89  65  60  NA 104 129  77  74  93  76  72 194 140 110 132 115  73
## [109] 328 160  86 150  95 130 151 117 124 127 125 116 105  92 157 154 220 187
## [127] 112 131 159  94 204 142 206 102 313 145  85  97 198 181 126 106 101 108
## [145] 119  70 122  91 276 100 291 114 172 217 141 133 156 136 180 135 195  99
## [163] 362 179 118 175 148 440 103 178 184 705 134 176 146 189 120 193 222 121
## [181] 107 305 199 439 223 230 280 377 310 158 486 153 139 182 221 229 216 170
## [199] 202 346 274 240 162 123 211 227 168 161 429 686 167 518 256 255 358 213
## [217] 191 282 155 138 246 237 271 171 414 219 262 409 243 241 197 449 143 188
## [235] 391 238 152 165 293 174 584 164 311 340 250 200 385 292 232 251 517 225
## [253] 169 309 235 501 224 275 318 144 397 343 245 186 337 351 166 349 423 359
## [271] 163 147 264 312 226 324 266 260 338 272 534 470 207 218 326 190 304 205
## [289] 233 401 177 330 286 247 357 315 231 339 283 374 248 279 281 234 261 290
## [307] 336 378 254 183 210 192
```

```
unique(shoppers$SpecialDay)
```

```
## [1] 0.0 0.4 0.8 1.0 0.2 0.6
```

```
unique(shoppers$Month)
```

```
##  [1] "Feb"  "Mar"  "May"  "Oct"  "June" "Jul"  "Aug"  "Nov"  "Sep"  "Dec"
```

```
unique(shoppers$OperatingSystems)
```

```
## [1] 1 2 4 3 7 6 8 5
```

```
unique(shoppers$Browser)
```

```
##  [1]  1  2  3  4  5  6  7 10  8  9 12 13 11
```

```
unique(shoppers$Region)
```

```
## [1] 1 9 2 3 4 5 6 7 8
```

```
unique(shoppers$TrafficType)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 18 19 16 17 20
```

```
unique(shoppers$VisitorType)
```

```
## [1] "Returning_Visitor" "New_Visitor"       "Other"
```

```
unique(shoppers$Weekend)
```

```
## [1] FALSE  TRUE
```

```
unique(shoppers$Revenue)
```

```
## [1] FALSE  TRUE
```

a.) In the administrative column we see the total number of administrative pages visited per visitor. The total number ranges from 0 to 19 with some missing values available. We will take care of the missing values during cleaning.

b.) The total number of informational pages visited by a visitor ranges from 0-16. With some missing values present, we will take care of this during cleaning.

c.) Product related pages visited ranges from 1 - 486, with missing values present.

d.) The closeness to a special day ranges from 0-1.

e.) We have 10 unique months when the visit to the site happened. No January and April. Notice how the month of June is written in full, but the rest of the months are 3 letter characters. We need to make sure the months are all uniform. We will convert June to Jun later on.

f.) The type of browser used are from 1-8

g.) The data also includes the region from which the visitor was visiting from. We have 10 unique regions.

h.) The type of customer visiting Kira Plastina were: returning_visitor, New_visitor, and other.

i.) We have information to indicate if the page was visited on the weekend or non-weekend.

j.) The revenue column gives us information on whether a visit to the site resulted in revenue or not.

Note: for the remaining columns with a lot of unique values, we will utilize summary statistics to get some insights on the column's data.

4. Data Cleaning

4.1 Missing values

```
#checking for total number of missing values in all the columns

colSums(is.na(shoppers))
```

```
##          Administrative Administrative_Duration           Informational
##                      14                      14                      14
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                      14                      14                      14
##             BounceRates               ExitRates             PageValues
##                      14                      14                       0
##              SpecialDay                   Month        OperatingSystems
##                       0                       0                       0
##                 Browser                  Region             TrafficType
##                       0                       0                       0
##             VisitorType                 Weekend                 Revenue
##                       0                       0                       0
```

There are 14 missing values in columns 1 to 8. Since it's a very small number, we will drop the missing values from our dataset.

```
new.shoppers <- na.omit(shoppers) #creating a dataset with no missing values
colSums(is.na(new.shoppers)) #confirming that the missing values have been dropped
```

```
##          Administrative Administrative_Duration           Informational
##                       0                       0                       0
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                       0                       0                       0
##             BounceRates               ExitRates             PageValues
##                       0                       0                       0
##              SpecialDay                   Month        OperatingSystems
##                       0                       0                       0
##                 Browser                  Region             TrafficType
##                       0                       0                       0
##             VisitorType                 Weekend                 Revenue
##                       0                       0                       0
```

The missing values have been dropped

```
#let's check the new shape of our dataset
glimpse(new.shoppers)
```

```
## Rows: 12,316
## Columns: 18
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 1, 2, 10, 19, 1, 1, 2, 3, 3, 16, 7, 6...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, -1.000000, 2.666667, 6...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.05...
## $ ExitRates               <dbl> 0.200000000, 0.100000000, 0.200000000, 0.14...
## $ PageValues              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ SpecialDay              <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.0, 0.8...
```

```
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <int> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3...
## $ Browser                 <int> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2...
## $ Region                  <int> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3...
## $ TrafficType             <int> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                 <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FA...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

We are left with 12,316 total number of records.

4.2 Duplicates

```
#checking for duplicates
anyDuplicated(new.shoppers)
```

```
## [1] 159
```

```
#new.shoppers[duplicated(new.shoppers),] # checking for the total number of records with duplicates.
```

Yes we do have some duplicates. The total number of duplicates observed is 117. I cleared the output to prevent the long list of 117 records showing. These duplicates will be dropped in the next chunk.

```
#creating a new data frame without the duplicates
shoppers.notdup <- new.shoppers[!duplicated(new.shoppers),]
glimpse(shoppers.notdup) #checking how our dataset looks like without the duplicates
```

```
## Rows: 12,199
## Columns: 18
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 1, 2, 10, 19, 1, 1, 2, 3, 3, 16, 7, 6...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, -1.000000, 2.666667, 6...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.05...
## $ ExitRates               <dbl> 0.200000000, 0.100000000, 0.200000000, 0.14...
## $ PageValues              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ SpecialDay              <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.0, 0.8...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <int> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3...
## $ Browser                 <int> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2...
## $ Region                  <int> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3...
## $ TrafficType             <int> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                 <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FA...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

The duplicates have been dropped. The remaining dataset has 12,199 total number of records.

```
colnames(shoppers.notdup)
```

```
##  [1] "Administrative"          "Administrative_Duration"
##  [3] "Informational"           "Informational_Duration"
##  [5] "ProductRelated"          "ProductRelated_Duration"
##  [7] "BounceRates"             "ExitRates"
##  [9] "PageValues"              "SpecialDay"
```

```
## [11] "Month"                    "OperatingSystems"
## [13] "Browser"                  "Region"
## [15] "TrafficType"              "VisitorType"
## [17] "Weekend"                  "Revenue"
```

4.3 Dropping unnecessary columns

```
#we establish that Exit Rates, PageValues are metrics specifically for Google, we won't be needing thes

shoppers.notdup$ExitRates <- NULL
shoppers.notdup$PageValues <- NULL
glimpse(shoppers.notdup) #confirming that the columns have been dropped
```

```
## Rows: 12,199
## Columns: 16
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 1, 2, 10, 19, 1, 1, 2, 3, 3, 16, 7, 6...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, -1.000000, 2.666667, 6...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.05...
## $ SpecialDay              <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 0.0, 0.8...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <int> 1, 2, 4, 3, 3, 2, 2, 1, 2, 2, 1, 1, 1, 2, 3...
## $ Browser                 <int> 1, 2, 1, 2, 3, 2, 4, 2, 2, 4, 1, 1, 1, 5, 2...
## $ Region                  <int> 1, 1, 9, 2, 1, 1, 3, 1, 2, 1, 3, 4, 1, 1, 3...
## $ TrafficType             <int> 1, 2, 3, 4, 4, 3, 3, 5, 3, 2, 3, 3, 3, 3, 3...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                 <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FA...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

Yes the unnecessary columns have been dropped. We have 16 columns remaining

4.4 Fixing Data Types

```
#special day, operating systems, browser, region, traffic type are all categorical variables. Let's con

shoppers.notdup$SpecialDay <- as.character(shoppers.notdup$SpecialDay)
shoppers.notdup$OperatingSystems <- as.character(shoppers.notdup$OperatingSystems)
shoppers.notdup$Browser <- as.character(shoppers.notdup$Browser)
shoppers.notdup$Region <- as.character(shoppers.notdup$Region)
shoppers.notdup$TrafficType <- as.character(shoppers.notdup$TrafficType)

glimpse(shoppers.notdup) #confirming that the columns have been converted to the right data types
```

```
## Rows: 12,199
## Columns: 16
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 1, 2, 10, 19, 1, 1, 2, 3, 3, 16, 7, 6...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, -1.000000, 2.666667, 6...
```

```
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.200000000, 0.05...
## $ SpecialDay              <chr> "0", "0", "0", "0", "0", "0", "0.4", "0", "...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <chr> "1", "2", "4", "3", "3", "2", "2", "1", "2"...
## $ Browser                 <chr> "1", "2", "1", "2", "3", "2", "4", "2", "2"...
## $ Region                  <chr> "1", "1", "9", "2", "1", "1", "3", "1", "2"...
## $ TrafficType             <chr> "1", "2", "3", "4", "4", "3", "3", "5", "3"...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                 <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FA...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

4.4 Anomaly and Outliers detection

4.4.1 Anomaly detection

```
#creating a data frame of numerical variables
num_cols <- select(shoppers.notdup, Administrative, Administrative_Duration, Informational, Informationa

##let's check on the summary statistics, this will help us know which variables have similar scale. Var

summary(num_cols)
```

```
##  Administrative  Administrative_Duration Informational
##  Min.   : 0.00   Min.   :  -1.00         Min.   : 0.0000
##  1st Qu.: 0.00   1st Qu.:   0.00         1st Qu.: 0.0000
##  Median : 1.00   Median :   9.00         Median : 0.0000
##  Mean   : 2.34   Mean   :  81.68         Mean   : 0.5088
##  3rd Qu.: 4.00   3rd Qu.:  94.75         3rd Qu.: 0.0000
##  Max.   :27.00   Max.   :3398.75         Max.   :24.0000
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :  -1.00        Min.   :  0.00   Min.   :   -1.0
##  1st Qu.:   0.00        1st Qu.:  8.00   1st Qu.:  193.6
##  Median :   0.00        Median : 18.00   Median :  609.5
##  Mean   :  34.84        Mean   : 32.06   Mean   : 1207.5
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1477.6
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##   BounceRates
##  Min.   :0.00000
##  1st Qu.:0.00000
##  Median :0.00293
##  Mean   :0.02045
##  3rd Qu.:0.01667
##  Max.   :0.20000
```

All numerical columns have different scales. We will plot most of them individually below. But first we need to drop all rows with negative duration times. There is no negative time, this is an anomaly.

```
#checking for columns with negative duration
anomaly <- shoppers.notdup[shoppers.notdup$Administrative_Duration ==-1.0,]
glimpse(anomaly)
```

```
## Rows: 33
## Columns: 16
## $ Administrative          <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Administrative_Duration <dbl> -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

```
## $ Informational_Duration  <dbl> -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,...
## $ ProductRelated          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ ProductRelated_Duration <dbl> -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,...
## $ BounceRates             <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 0.2...
## $ SpecialDay              <chr> "0", "0.4", "0", "0", "0", "0", "0.6", "0",...
## $ Month                   <chr> "Feb", "Feb", "Feb", "Feb", "Feb", "Feb", "...
## $ OperatingSystems        <chr> "4", "2", "1", "1", "3", "2", "2", "1", "2"...
## $ Browser                 <chr> "1", "4", "2", "1", "3", "2", "2", "1", "2"...
## $ Region                  <chr> "9", "3", "1", "4", "1", "4", "3", "3", "4"...
## $ TrafficType             <chr> "3", "3", "5", "3", "3", "1", "2", "4", "3"...
## $ VisitorType             <chr> "Returning_Visitor", "Returning_Visitor", "...
## $ Weekend                 <lgl> FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, FAL...
## $ Revenue                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F...
```

There are 33 records that have negative duration. We will drop them as shown below.

```
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$Administrative_Duration ==-1.0,] #getting rid of ro
summary(shoppers.notdup) #confirming that there are no negative values
```

```
##  Administrative   Administrative_Duration Informational
##  Min.   : 0.000   Min.   :   0.00         Min.   : 0.0000
##  1st Qu.: 0.000   1st Qu.:   0.00         1st Qu.: 0.0000
##  Median : 1.000   Median :  10.00         Median : 0.0000
##  Mean   : 2.346   Mean   :  81.91         Mean   : 0.5102
##  3rd Qu.: 4.000   3rd Qu.:  95.00         3rd Qu.: 0.0000
##  Max.   :27.000   Max.   :3398.75         Max.   :24.0000
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :   0.00        Min.   :  0.00   Min.   :    0.0
##  1st Qu.:   0.00        1st Qu.:  8.00   1st Qu.:  196.5
##  Median :   0.00        Median : 18.00   Median :  612.8
##  Mean   :  34.93        Mean   : 32.14   Mean   : 1210.8
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1481.8
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##   BounceRates        SpecialDay           Month           OperatingSystems
##  Min.   :0.000000   Length:12166       Length:12166       Length:12166
##  1st Qu.:0.000000   Class :character   Class :character   Class :character
##  Median :0.002865   Mode  :character   Mode  :character   Mode  :character
##  Mean   :0.020009
##  3rd Qu.:0.016324
##  Max.   :0.200000
##    Browser             Region          TrafficType        VisitorType
##  Length:12166       Length:12166       Length:12166       Length:12166
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Weekend         Revenue
##  Mode :logical   Mode :logical
##  FALSE:9313      FALSE:10258
##  TRUE :2853      TRUE :1908
##
##
##
```

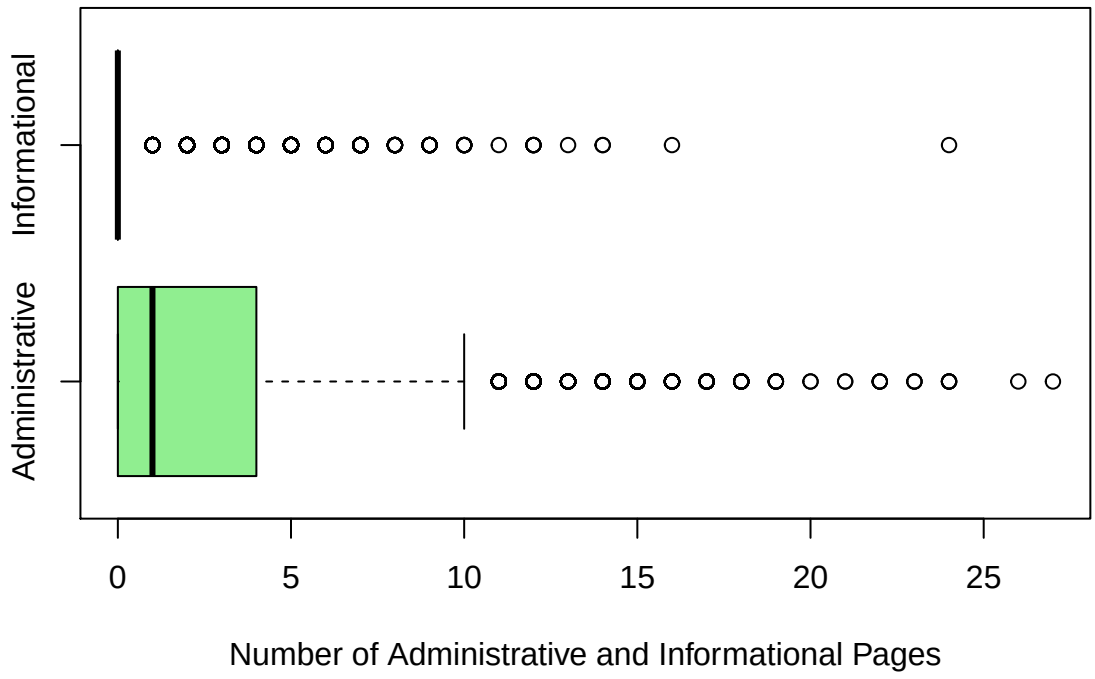
All the records with negative values have been dropped. From the statistical summary, we see that the

minimums start from zero and not -1. We will proceed to check for outliers.

4.4.2 Outliers detection using boxplots

1. Administrative and Informational

```
boxplot(select(shoppers.notdup, Administrative, Informational), col = 'light green', horizontal = TRUE,
```



The two columns represent the total number of pages visited during a session. Information column has most of it's values as 0 hence the box plot is very tiny. The column has outliers, most of them makes sense because it's very possible for someone to visit up to 25 informational pages during a session. But we have a single point that is too far from the rest, we will delete this outlier to minimize bias.

The administrative column has some outliers as well which makes sense. But the two points above 25 will be considered extremes. These points will be dropped to minimize bias.

```
#extracting the records
print(shoppers.notdup[shoppers.notdup$Administrative > 25,])
```

```
##       Administrative Administrative_Duration Informational
## 8309              27                853.7359             2
## 12179             26               1561.7176             9
##       Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 8309                126.5000            584                24844.156  0.00209938
## 12179               503.7222            183                 9676.093  0.01105483
##       SpecialDay Month OperatingSystems Browser Region TrafficType
## 8309           0   Nov                2       4      3           8
## 12179          0   Nov                3       2      2          13
##             VisitorType Weekend Revenue
```

```
## 8309  Returning_Visitor   FALSE   FALSE
## 12179 Returning_Visitor   FALSE    TRUE
```

```
print(shoppers.notdup[shoppers.notdup$Informational > 20,])
```

```
##      Administrative Administrative_Duration Informational
## 5153             17                2629.254            24
##      Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 5153                2050.433            705                43171.23 0.004851285
##      SpecialDay Month OperatingSystems Browser Region TrafficType
## 5153          0   May                2       2      1          14
##            VisitorType Weekend Revenue
## 5153 Returning_Visitor    TRUE   FALSE
```

```
#removing outliers in informational and administrative columns
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$Administrative > 25,]
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$Informational > 20,]
#confirming that the extreme outliers have been removed
print(shoppers.notdup[shoppers.notdup$Administrative > 25,])
```

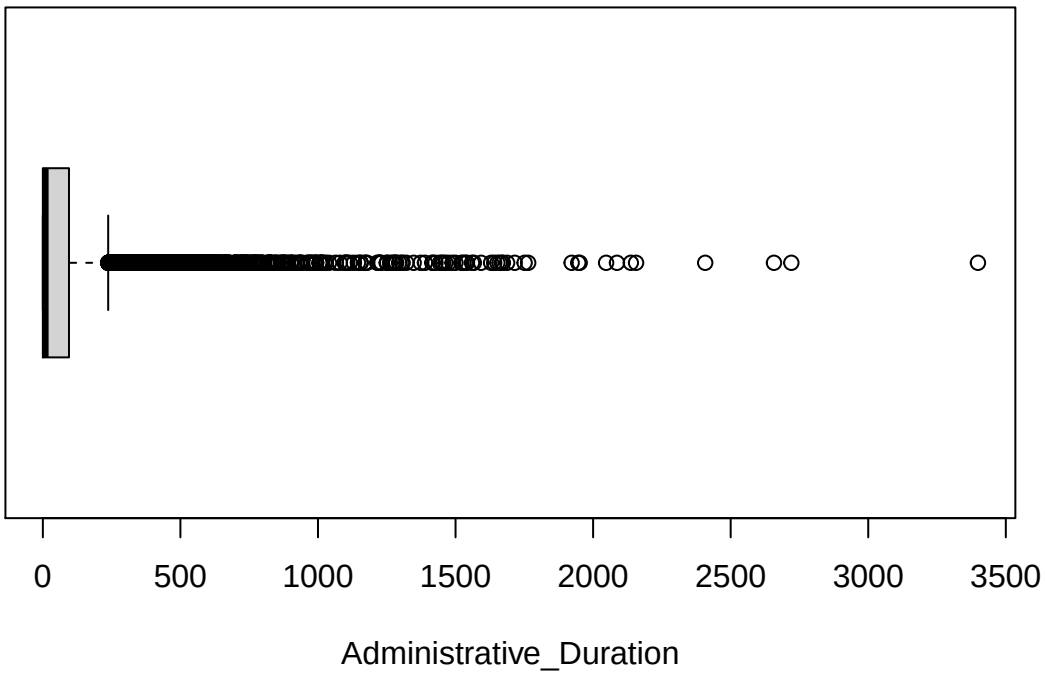
```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

```
print(shoppers.notdup[shoppers.notdup$Informational > 20,])
```

```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

2. Administrative_Duration

```
boxplot(shoppers.notdup$Administrative_Duration, color = "green", horizontal = TRUE, xlab = "Administra
```



We see a lot of outliers which are mostly true observations but from 1900 seconds, the points appear further than the rest. We will delete these outliers that are far from the rest.

```
#extracting the records
print(shoppers.notdup[shoppers.notdup$Administrative_Duration > 1900,])
```

```
##       Administrative Administrative_Duration Informational
## 1573              11                2047.235             9
## 4510              22                1951.279             1
## 4885               5                2156.167             2
## 5777               7                2720.500             3
## 6166              10                2407.424             3
## 8072               5                3398.750             6
## 9239              15                2657.318            13
## 9663               4                1946.000             0
## 9737               8                2086.750             1
## 11713             14                2137.113             0
## 12018              6                1922.000             0
##       Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 1573                1146.667             45                3641.2132 0.002636535
## 4510                  99.000             55                3373.0159 0.016438356
## 4885                  92.000             15                 463.0000 0.036363636
## 5777                 353.400             68                5943.5476 0.032236842
## 6166                 434.300            486               23050.1041 0.000323719
## 8072                2549.375            449               63973.5222 0.000764406
## 9239                1949.167            343               29970.4660 0.005315857
## 9663                   0.000             31                 731.0000 0.000000000
```

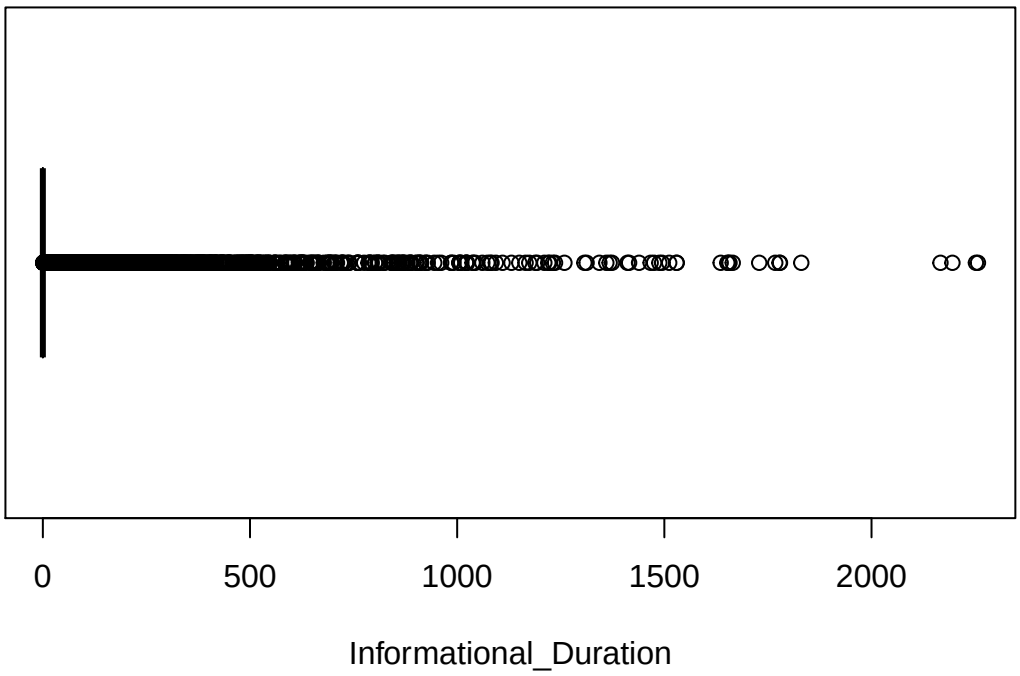
```
## 9737                  46.500             81               5546.0000 0.002325581
## 11713                  0.000             53               4223.4098 0.008771930
## 12018                  0.000             30                941.6726 0.018181818
##       SpecialDay Month OperatingSystems Browser Region TrafficType
## 1573           0   Mar                2       2      1           2
## 4510           0   May                2       2      3           1
## 4885           0   May                1       2      4           4
## 5777           0   Jul                3       2      1          13
## 6166           0   Jul                2       2      1           3
## 8072           0   Dec                2       2      1           2
## 9239           0   Dec                2       2      1           2
## 9663           0   Dec                2       2      4           2
## 9737           0   Nov                2       2      1           2
## 11713          0   Nov                3       2      2           2
## 12018          0   Nov                3       2      2           1
##             VisitorType Weekend Revenue
## 1573  Returning_Visitor   FALSE    TRUE
## 4510  Returning_Visitor   FALSE   FALSE
## 4885  Returning_Visitor    TRUE   FALSE
## 5777  Returning_Visitor   FALSE   FALSE
## 6166  Returning_Visitor   FALSE   FALSE
## 8072  Returning_Visitor   FALSE   FALSE
## 9239  Returning_Visitor   FALSE   FALSE
## 9663        New_Visitor    TRUE   FALSE
## 9737  Returning_Visitor   FALSE    TRUE
## 11713 Returning_Visitor   FALSE   FALSE
## 12018 Returning_Visitor   FALSE   FALSE
```

```
#removing outliers in Administrative_Duration column
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$Administrative_Duration > 1900,]
#confirming that the extreme outliers have been removed
print(shoppers.notdup[shoppers.notdup$Administrative > 1900,])
```

```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

3. Informational Duration

```
boxplot(shoppers.notdup$Informational_Duration, color = 'green', horizontal = TRUE, xlab = "Informationa
```



Most of the entries for Informational_Duration are zero hence the boxplot is at zero. We see a lot of outliers. We will consider points from ~1600 as too far from the rest. Let's drop the values below.

```
#extracting the records
print(shoppers.notdup[shoppers.notdup$Informational_Duration > 1600,])
```

```
##       Administrative Administrative_Duration Informational
## 2302               3                44.00000             4
## 3575               0                 0.00000             9
## 4254               3               270.00000             1
## 5513               3                28.80000             5
## 6021               8               116.97222             9
## 6064               4                69.23333             3
## 6414              11               251.30909             2
## 7926              12               504.25333             2
## 8847               0                 0.00000             3
## 9515               0                 0.00000             1
## 10295             11               202.42544             5
## 10303              4               367.00000             5
## 11471              8               145.10000             6
## 11542              8               279.25000             2
##       Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 2302                1636.000             49                4945.083 0.009259259
## 3575                1779.167             12                1886.500 0.007017544
## 4254                1778.000            362               13259.294 0.002883379
## 5513                2195.300             21                 378.400 0.029629630
## 6021                2252.033             19                1135.881 0.009677419
```

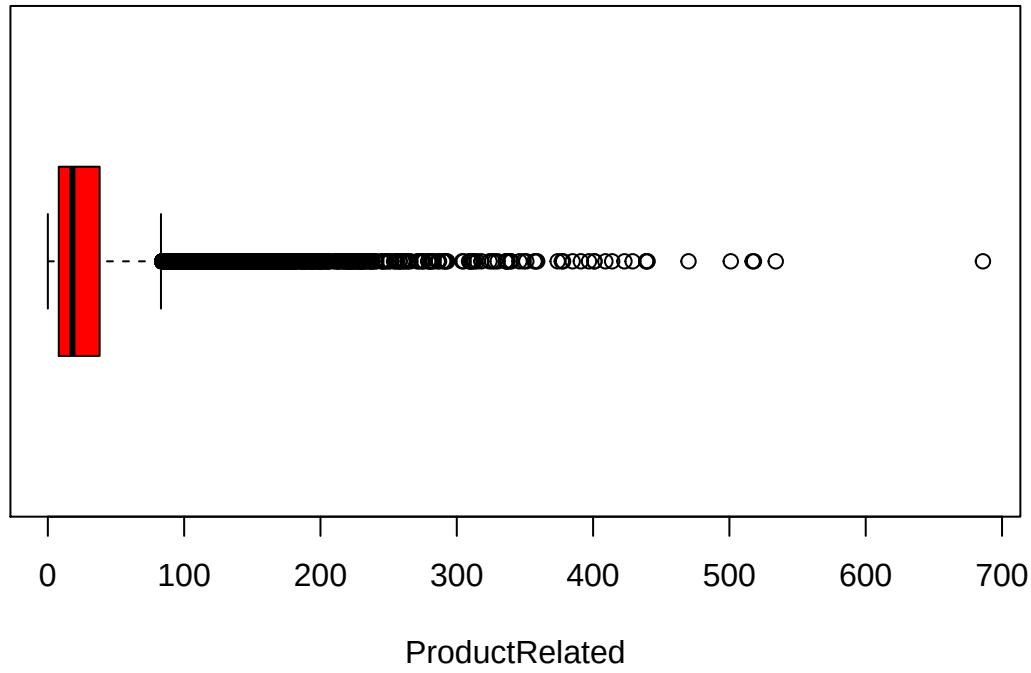
```
## 6064                1657.300             26                2306.048 0.046666667
## 6414                1655.400             80                1249.048 0.003571429
## 7926                1665.067            243                8768.477 0.014137484
## 8847                1830.500             81                5000.739 0.003294118
## 9515                1729.000             19                1401.083 0.020000000
## 10295               1767.667            338               13265.356 0.007520348
## 10303               2256.917             74                8981.580 0.002988954
## 11471               2166.500             14                1805.425 0.000000000
## 11542               1652.000             43                1717.740 0.000000000
##       SpecialDay Month OperatingSystems Browser Region TrafficType
## 2302         0.8   May                2       2      2           4
## 3575         0.6   May                2       2      6           3
## 4254           0   May                2       2      1          19
## 5513           0   Jul                1       1      3           1
## 6021           0   Jul                3       2      4           2
## 6064           0   Oct                1       1      2           3
## 6414           0   Sep                3       2      4           2
## 7926           0   Jul                2       2      1           1
## 8847           0   Nov                2       2      3           2
## 9515           0   Dec                2       2      1           3
## 10295          0   Nov                2       2      3           2
## 10303          0   Dec                2       2      1           8
## 11471          0   Dec                4       5      1           2
## 11542          0   Dec                3       2      5           2
##             VisitorType Weekend Revenue
## 2302  Returning_Visitor   FALSE    TRUE
## 3575        New_Visitor   FALSE   FALSE
## 4254  Returning_Visitor    TRUE   FALSE
## 5513  Returning_Visitor   FALSE   FALSE
## 6021  Returning_Visitor    TRUE   FALSE
## 6064  Returning_Visitor   FALSE   FALSE
## 6414  Returning_Visitor    TRUE   FALSE
## 7926  Returning_Visitor   FALSE    TRUE
## 8847  Returning_Visitor    TRUE   FALSE
## 9515  Returning_Visitor   FALSE   FALSE
## 10295 Returning_Visitor   FALSE    TRUE
## 10303 Returning_Visitor   FALSE   FALSE
## 11471 Returning_Visitor   FALSE   FALSE
## 11542 Returning_Visitor   FALSE    TRUE
```

```r
#removing outliers in Informational_Duration column
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$Informational_Duration > 1600,]
#confirming that the extreme outliers have been removed
print(shoppers.notdup[shoppers.notdup$Informational_Duration > 1600,])
```

```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

4. Product Related

```r
boxplot(shoppers.notdup$ProductRelated, col = 'red', horizontal = TRUE, xlab = "ProductRelated")
```



The points above ~480 appears too far from the rest, we will delete these points and leave the rest of outliers as true observations

```r
#extracting the records
print(shoppers.notdup[shoppers.notdup$ProductRelated > 480,])
```

```
##       Administrative Administrative_Duration Informational
## 6685              20                199.4563             7
## 6788               8                161.6686             0
## 8785              20               1307.6750             3
## 8973              11                631.4167             5
## 10319              9                444.2847             0
##       Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 6685                299.0333            686                23342.08 0.009853301
## 6788                  0.0000            518                11976.72 0.000038300
## 8785                132.6667            517                27009.86 0.004385217
## 8973               1037.1500            501                21672.24 0.003964723
## 10319                 0.0000            534                18504.13 0.010856514
##       SpecialDay Month OperatingSystems Browser Region TrafficType
## 6685           0   Aug                2       2      1           1
## 6788           0   Oct                4       2      9           2
## 8785           0   Nov                1       1      1           2
## 8973           0   Nov                2       2      1           2
## 10319          0   Nov                2       2      3           2
##             VisitorType Weekend Revenue
```

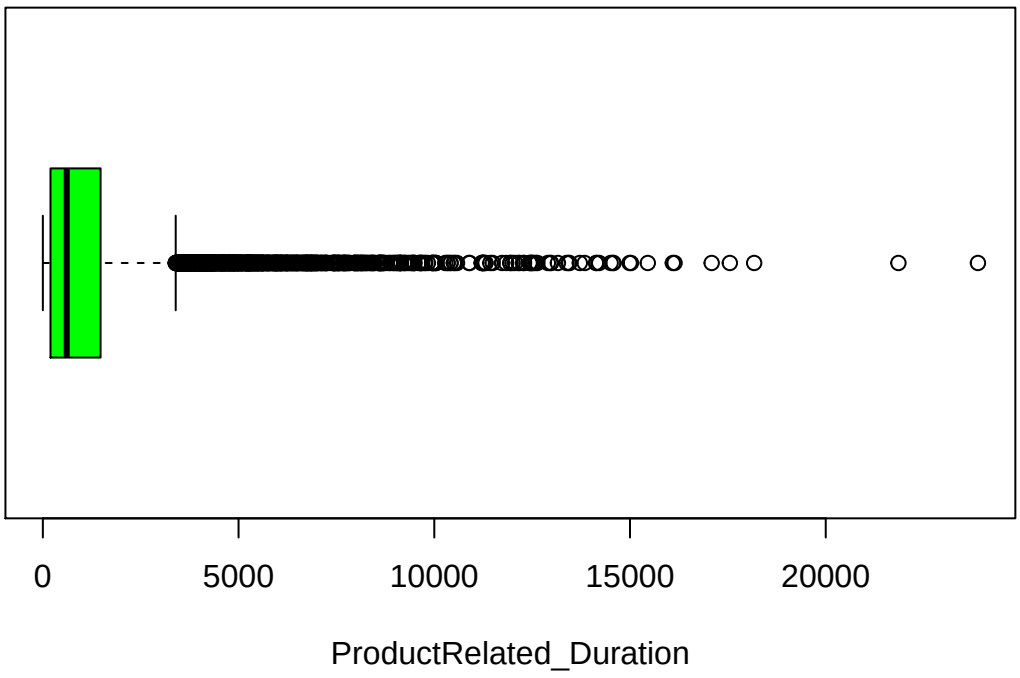
```
## 6685  Returning_Visitor   FALSE   FALSE
## 6788  Returning_Visitor   FALSE   FALSE
## 8785  Returning_Visitor   FALSE    TRUE
## 8973  Returning_Visitor   FALSE    TRUE
## 10319 Returning_Visitor    TRUE    TRUE
```

```
#removing outliers in ProductRelated column
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$ProductRelated > 480,]
#confirming that the extreme outliers have been removed
print(shoppers.notdup[shoppers.notdup$ProductRelated > 480,])
```

```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

5. ProductRelated_Duration

```
boxplot(shoppers.notdup$ProductRelated_Duration, col = 'green', horizontal = TRUE, xlab = "ProductRelat
```



Points above 20000 are too far from the test. We will delete them. And leave the rest of the outliers as true observations.

```
#extracting the records
print(shoppers.notdup[shoppers.notdup$ProductRelated_Duration > 20000,])
```

```
##      Administrative Administrative_Duration Informational
```

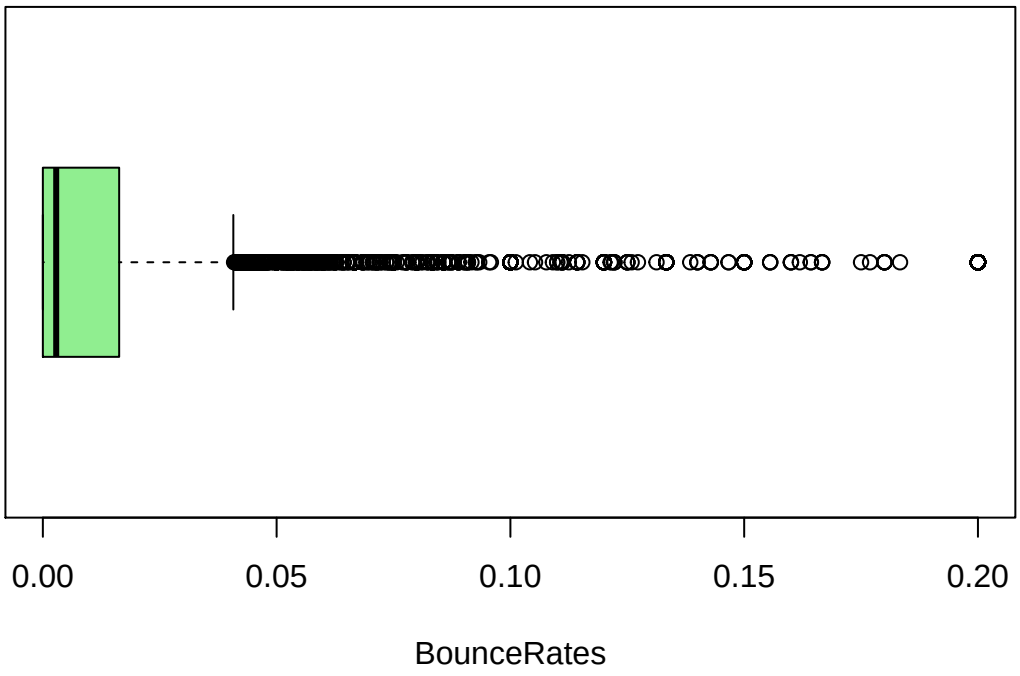
```
## 5917             12                245.7333             4
## 7663             10               1251.2000             7
##      Informational_Duration ProductRelated ProductRelated_Duration BounceRates
## 5917                 1511.7            439                21857.05 0.003588626
## 7663                  250.0            414                23888.81 0.009900332
##      SpecialDay Month OperatingSystems Browser Region TrafficType
## 5917          0   Sep                2       2      1          13
## 7663          0   Sep                2       2      4          13
##            VisitorType Weekend Revenue
## 5917 Returning_Visitor   FALSE   FALSE
## 7663 Returning_Visitor   FALSE   FALSE
```

```
#removing outliers in ProductRelated_Duration column
shoppers.notdup <- shoppers.notdup[!shoppers.notdup$ProductRelated_Duration > 20000,]
#confirming that the extreme outliers have been removed
print(shoppers.notdup[shoppers.notdup$ProductRelated_Duration > 20000,])
```

```
##  [1] Administrative          Administrative_Duration Informational
##  [4] Informational_Duration  ProductRelated          ProductRelated_Duration
##  [7] BounceRates             SpecialDay              Month
## [10] OperatingSystems        Browser                 Region
## [13] TrafficType             VisitorType             Weekend
## [16] Revenue
## <0 rows> (or 0-length row.names)
```

6. BounceRates

```
boxplot(shoppers.notdup$BounceRates, col = 'light green', horizontal = TRUE, xlab = "BounceRates")
```



We see a lot of outlier observations here. We will keep all the outliers as true observations for now.

5. EDA

5.1 Univariate Analysis

5.1.1 Measures of Central Tendency and Measures of Dispersion

```
#we will use summary statistics to get the measures of central tendency for numerical variables
#creating a data frame of numerical variables first
num_cols <- select(shoppers.notdup, Administrative, Administrative_Duration, Informational, Informationa
summary(num_cols)
```

```
##  Administrative   Administrative_Duration Informational
##  Min.   : 0.000   Min.   :   0.00         Min.   : 0.0000
##  1st Qu.: 0.000   1st Qu.:   0.00         1st Qu.: 0.0000
##  Median : 1.000   Median :   9.00         Median : 0.0000
##  Mean   : 2.325   Mean   :  79.09         Mean   : 0.4988
##  3rd Qu.: 4.000   3rd Qu.:  94.00         3rd Qu.: 0.0000
##  Max.   :24.000   Max.   :1764.00         Max.   :16.0000
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :   0.00        Min.   :  0.00   Min.   :    0.0
##  1st Qu.:   0.00        1st Qu.:  8.00   1st Qu.:  195.5
##  Median :   0.00        Median : 18.00   Median :  610.0
##  Mean   :  31.85        Mean   : 31.57   Mean   : 1178.5
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1474.5
##  Max.   :1530.00        Max.   :470.00   Max.   :18171.8
##   BounceRates
##  Min.   :0.000000
##  1st Qu.:0.000000
##  Median :0.002857
##  Mean   :0.020039
##  3rd Qu.:0.016327
##  Max.   :0.200000
```

The general observations is that, the maximum values for all the numerical columns are relatively far away from the means and medians indicating a non-normal distribution. We see a skewness to the right. This is not unusual because the data contained a lot of outliers that we kept as true observations.

```
#variance
lapply(num_cols, var)
```

```
## $Administrative
## [1] 10.84768
##
## $Administrative_Duration
## [1] 25912.06
##
## $Informational
## [1] 1.529656
##
## $Informational_Duration
## [1] 14632.99
##
## $ProductRelated
```

```
## [1] 1733.775
##
## $ProductRelated_Duration
## [1] 2789583
##
## $BounceRates
## [1] 0.001992256
```

We see large variances in Administrative_Duration, Information_Duration and ProductRelated_Duration columns. It goes back to the presence of many outliers in our dataset. This might introduce biases during modeling, so we need to scale the data before modeling to minimize the bias.

```
#standard deviation
lapply(num_cols,sd)
```

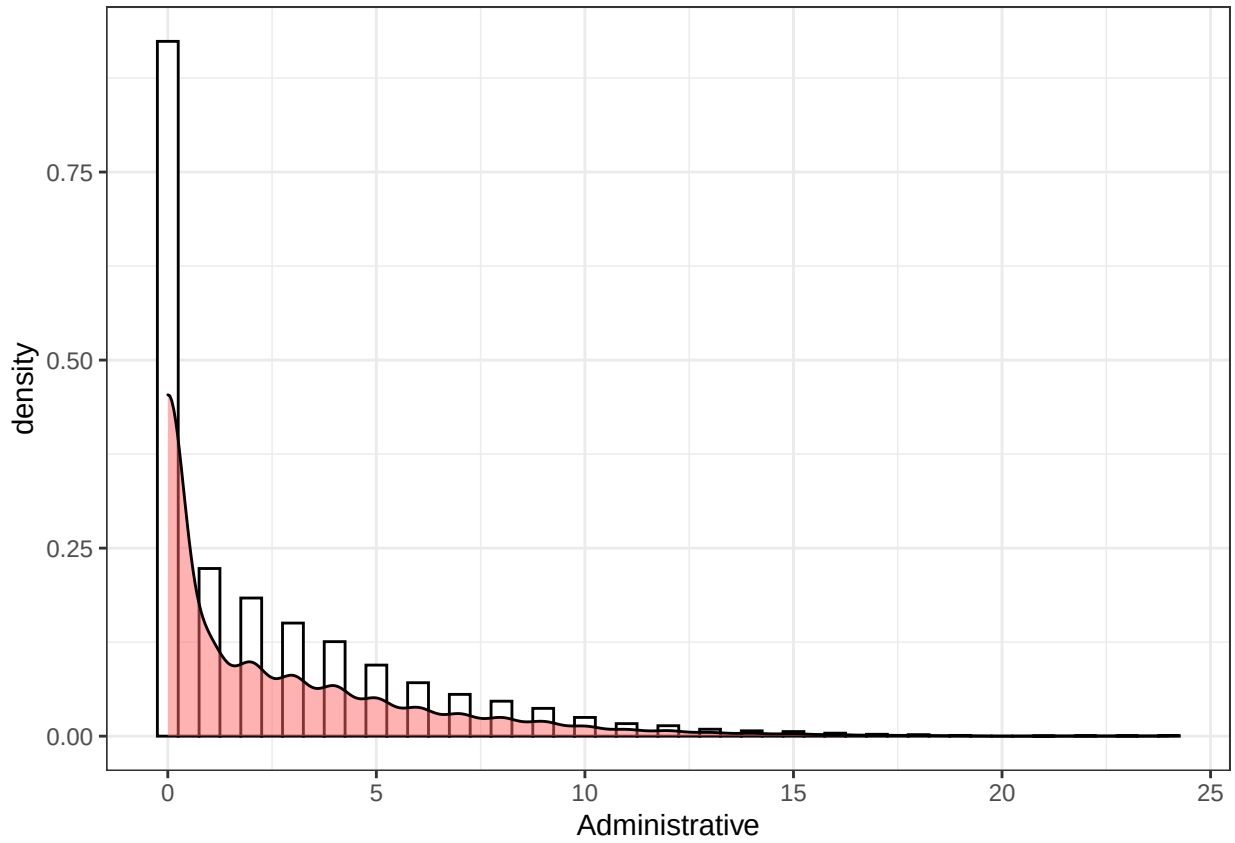
```
## $Administrative
## [1] 3.293581
##
## $Administrative_Duration
## [1] 160.9722
##
## $Informational
## [1] 1.236793
##
## $Informational_Duration
## [1] 120.9669
##
## $ProductRelated
## [1] 41.63862
##
## $ProductRelated_Duration
## [1] 1670.205
##
## $BounceRates
## [1] 0.0446347
```

Similar observations as with variances. The largest standard deviations belong to the duration columns.

5.1.2 Histograms and Density Plots

1. Administrative

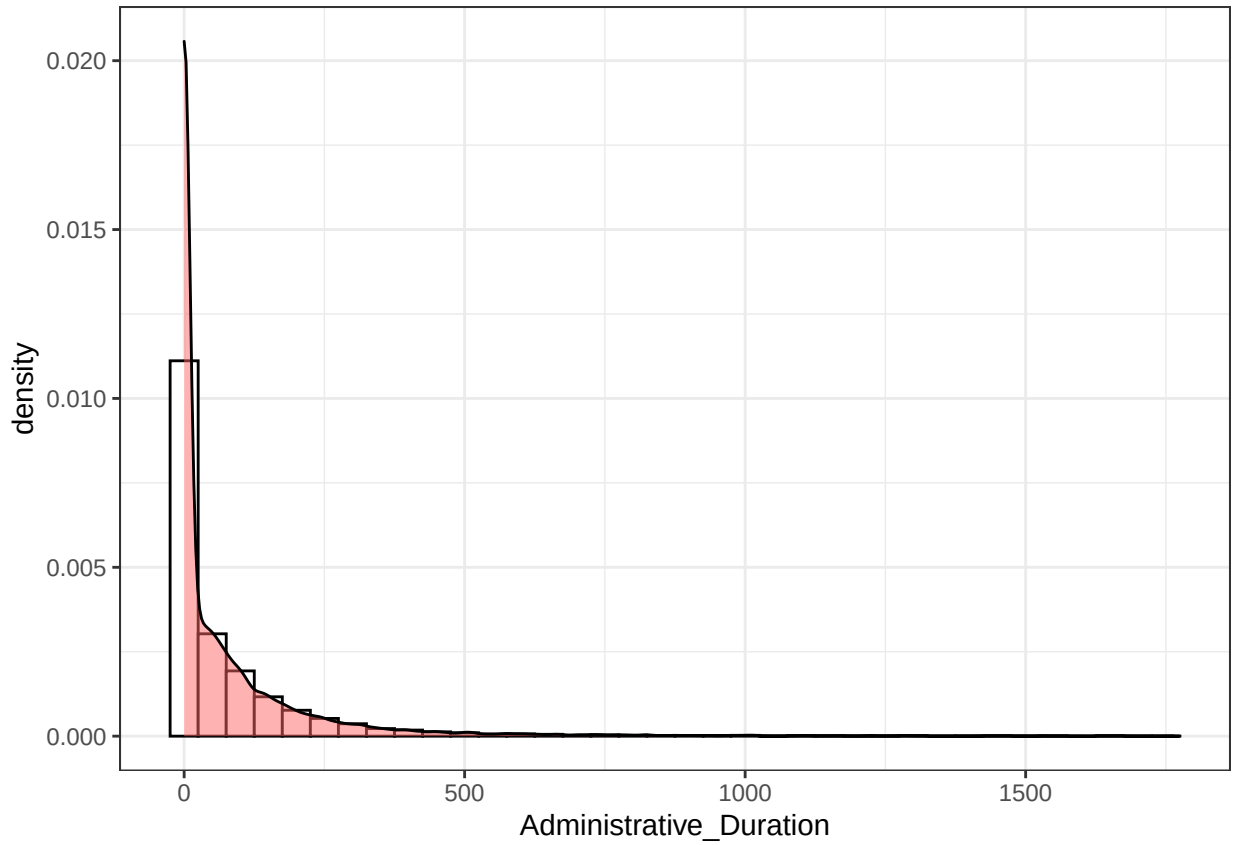
```
ggplot(num_cols, aes(x = Administrative)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 0.5, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



Administrative column is not normally distributed and has an obvious right skew.

2. Administrative Duration

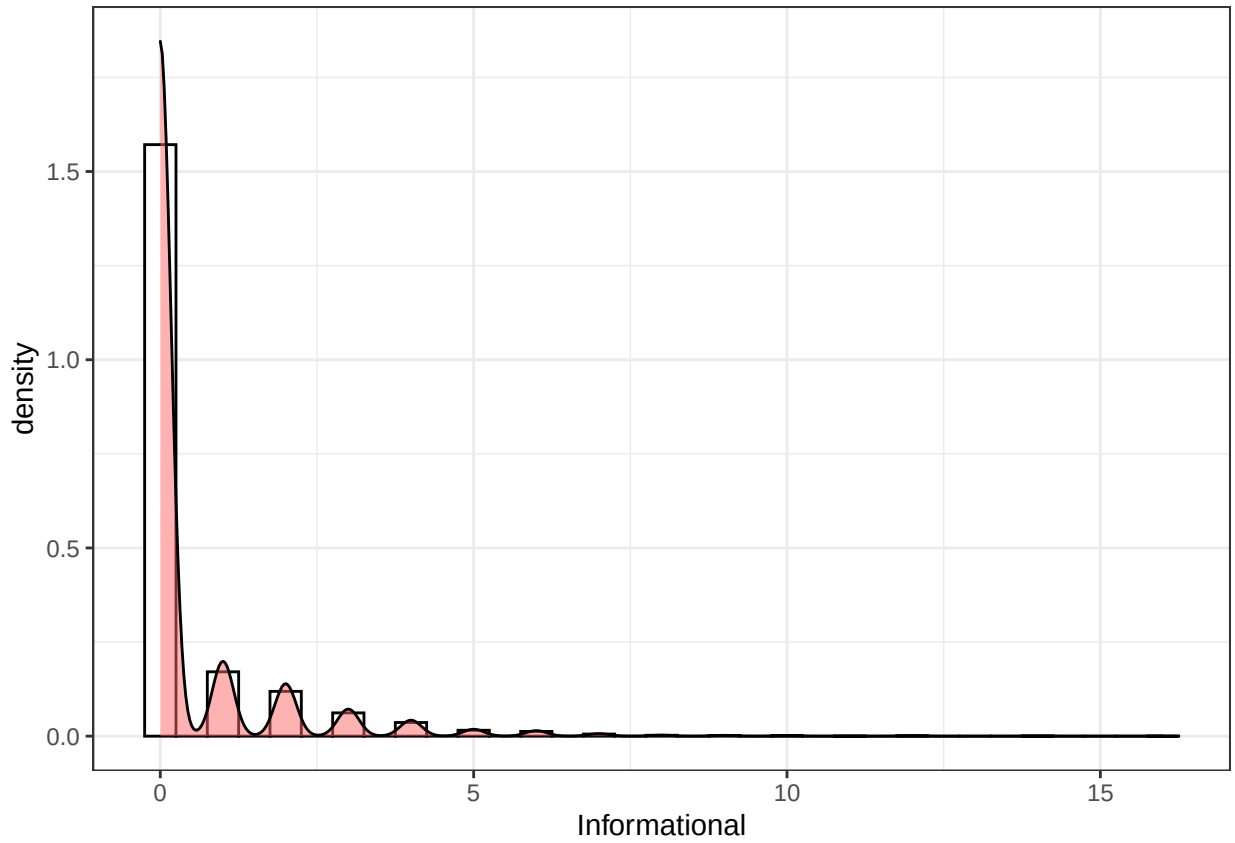
```
ggplot(num_cols, aes(x = Administrative_Duration)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 50, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



Most of the administrative duration data is located at 0. There is an obvious skewness to the right.

3. Informational

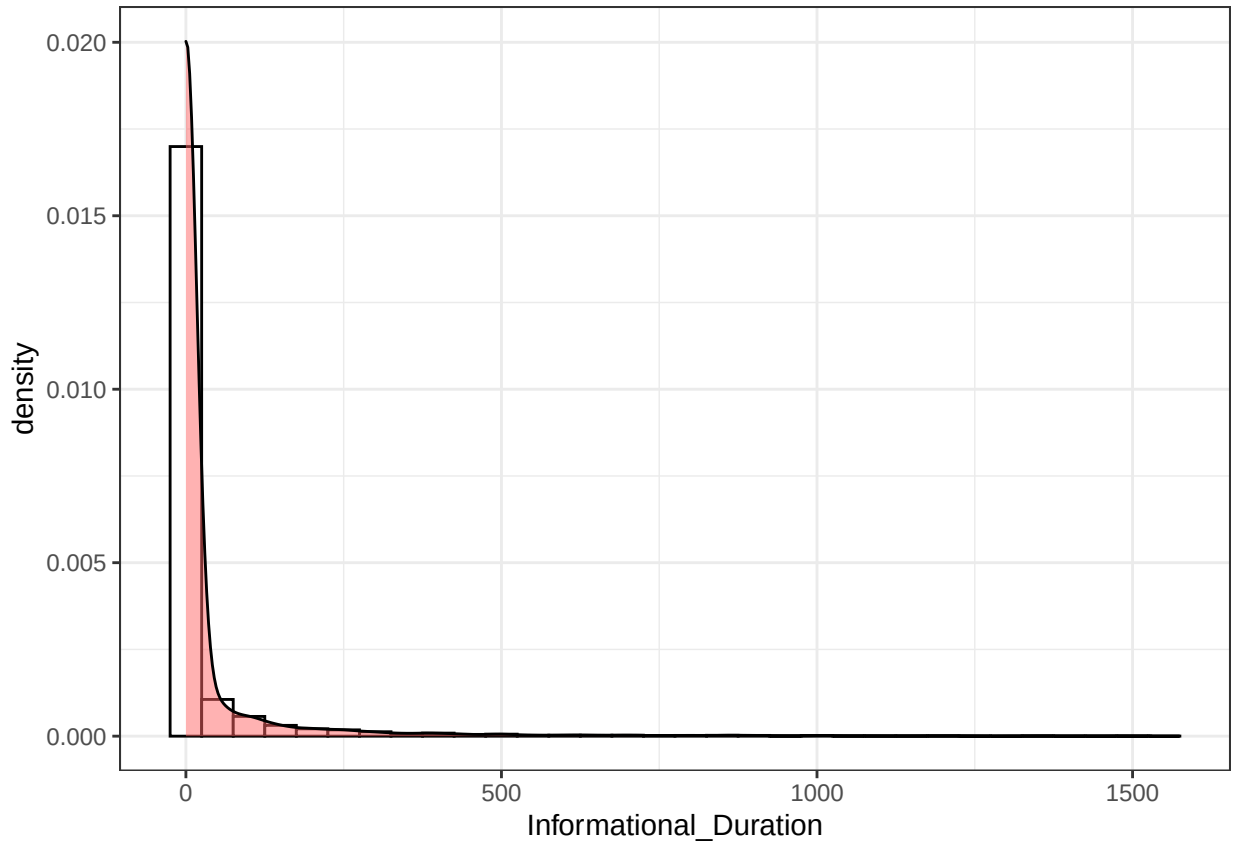
```
ggplot(num_cols, aes(x = Informational)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 0.5, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



Informational data is multi-nomial ditributed with right skewness

4. Informational Duration

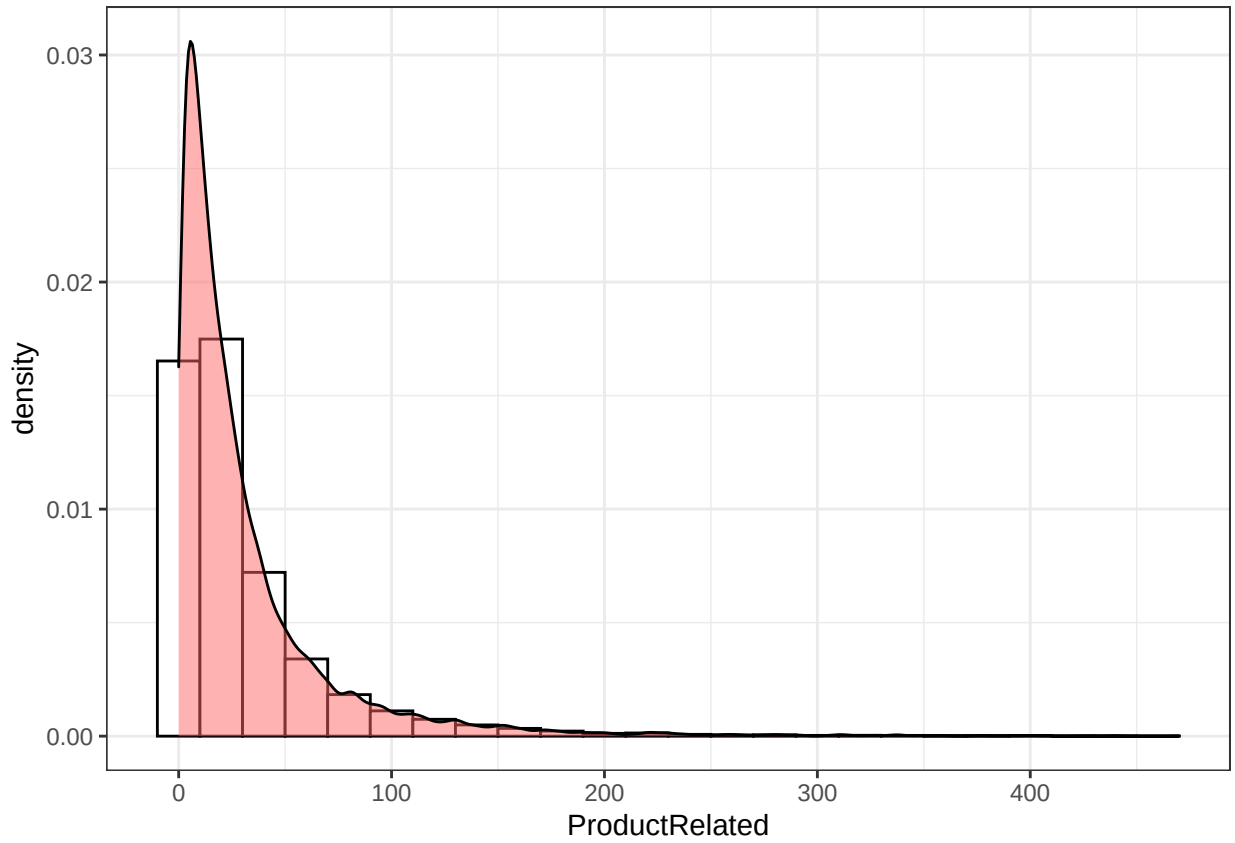
```
ggplot(num_cols, aes(x = Informational_Duration)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 50, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



Informational Duration is rightly skewed and with a lot of the data points at zero

5. Product Related

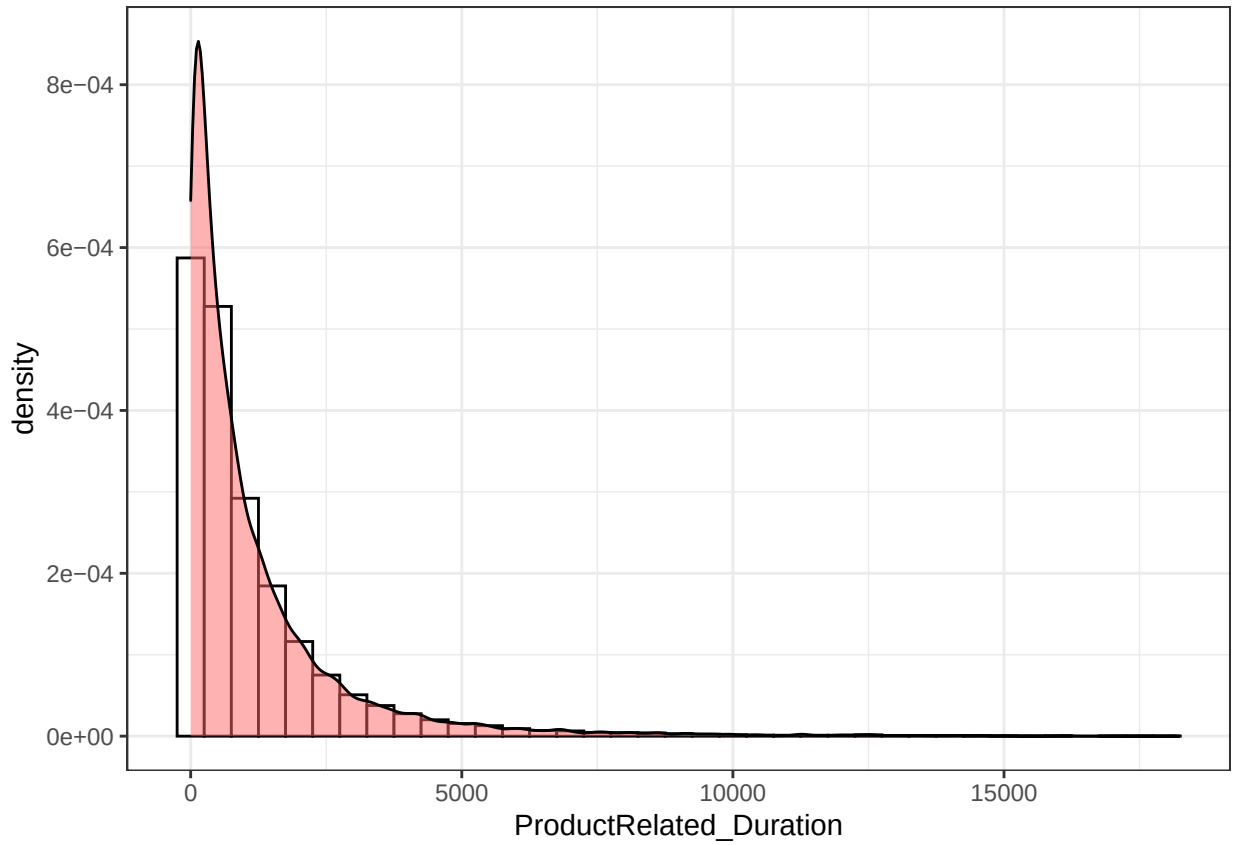
```
ggplot(num_cols, aes(x = ProductRelated)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 20, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



We see a lot of the product related data point are between 0 and 30. The distribution is not normal. Right skewness is observed.

6. Product Related Duration

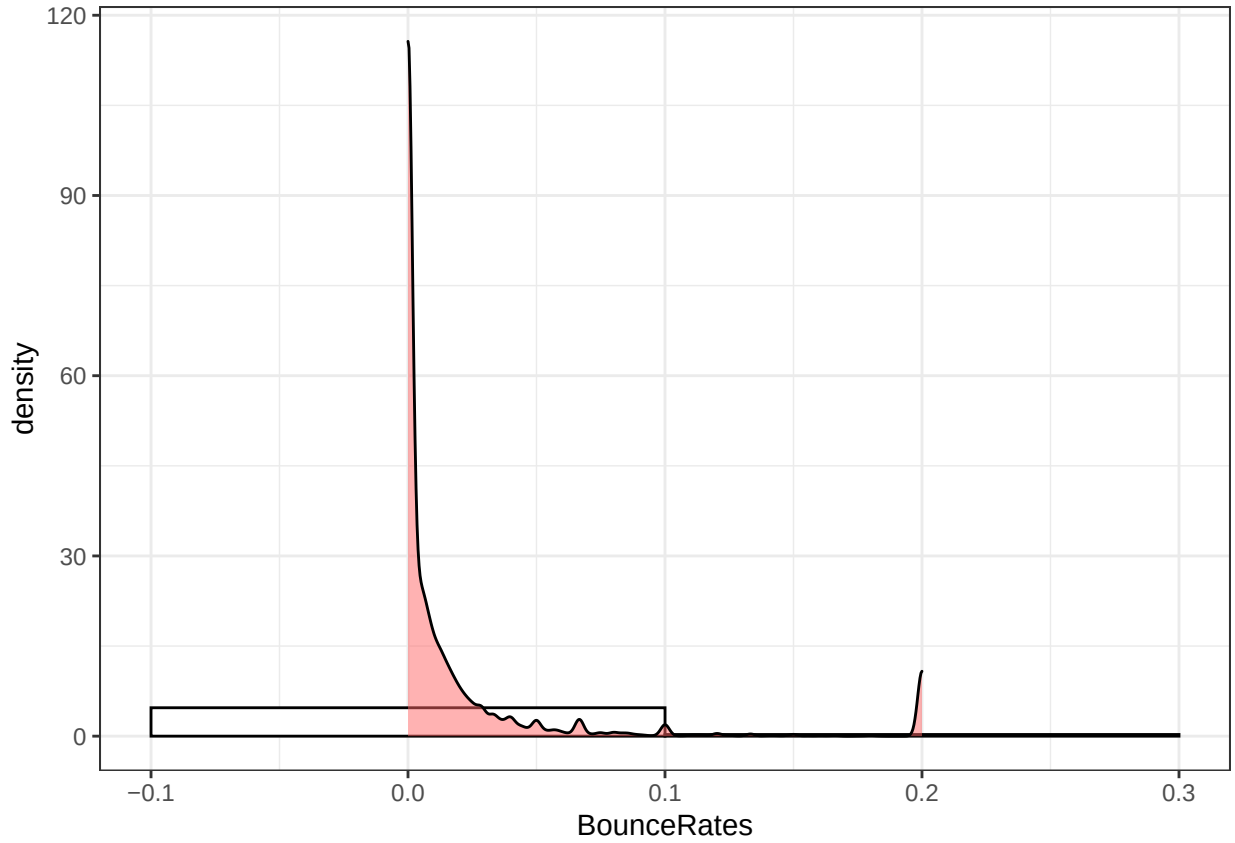
```
ggplot(num_cols, aes(x = ProductRelated_Duration)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 500, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



The data distribution is mostly from 0-1000. Right skewness is observed

7. Bounce Rates

```
ggplot(num_cols, aes(x = BounceRates)) +
  theme_bw() +
  geom_histogram(aes(y = ..density..,), binwidth = 0.2, color = "black", fill = "white") +
  geom_density(alpha = 0.5, fill = "#FF6666")
```



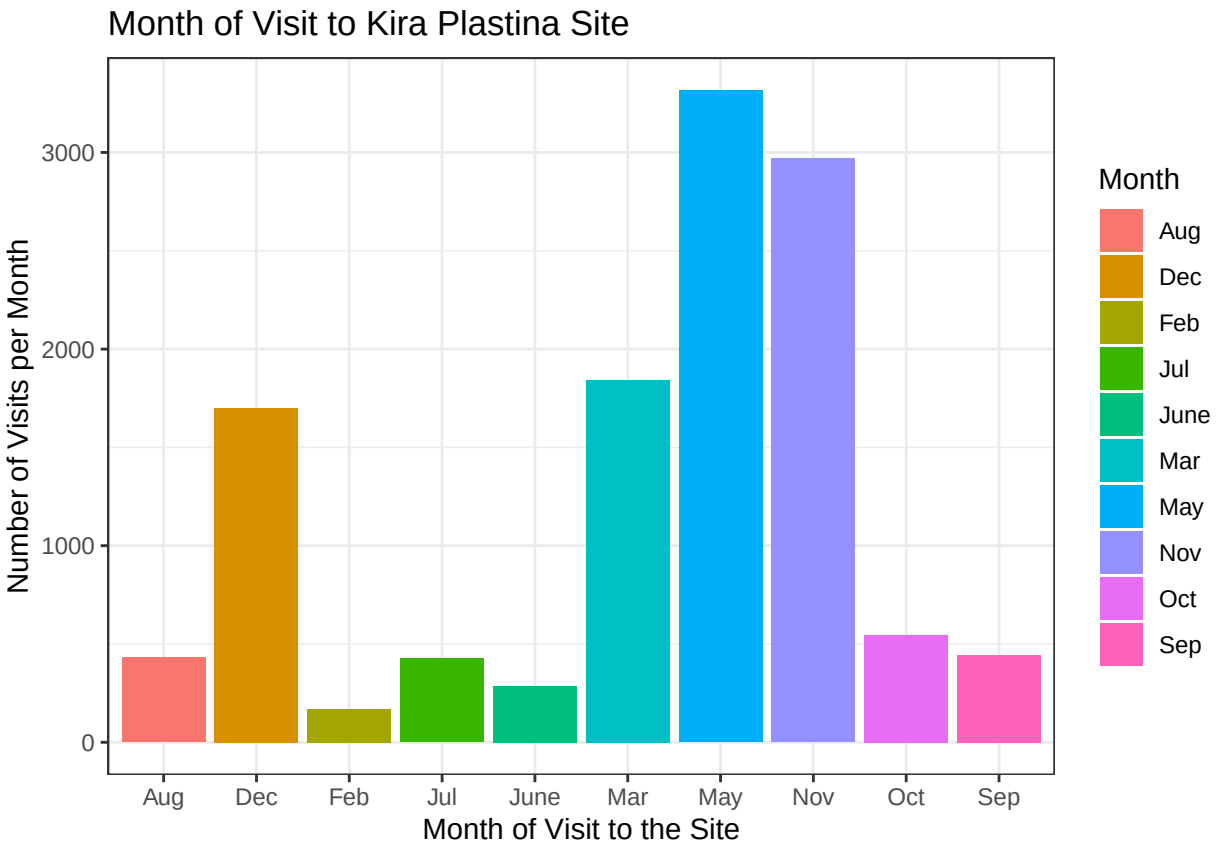
Between 0.1 and 0.2, we see no distribution.

Conclusions from histograms overlaid with density plots: all numerical variables show non-normal distributions. All the plots have an obvious right skewness with a long tail. Some columns such as Informational column show multi-nomial distribution.

5.1.3 Bar Plots for Categorical Variables

1. Month

```
ggplot(shoppers.notdup, aes(x = Month, fill = Month)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Month of Visit to the Site", y = "Number of Visits per Month", title = "Month of Visit to K
```



The month of May had the highest number of visits to the site followed by November, March and closely by December, February, June had the lowest number of visits to the site. Let's convert June to Jun to match the 3 character letter that the rest of the months have.

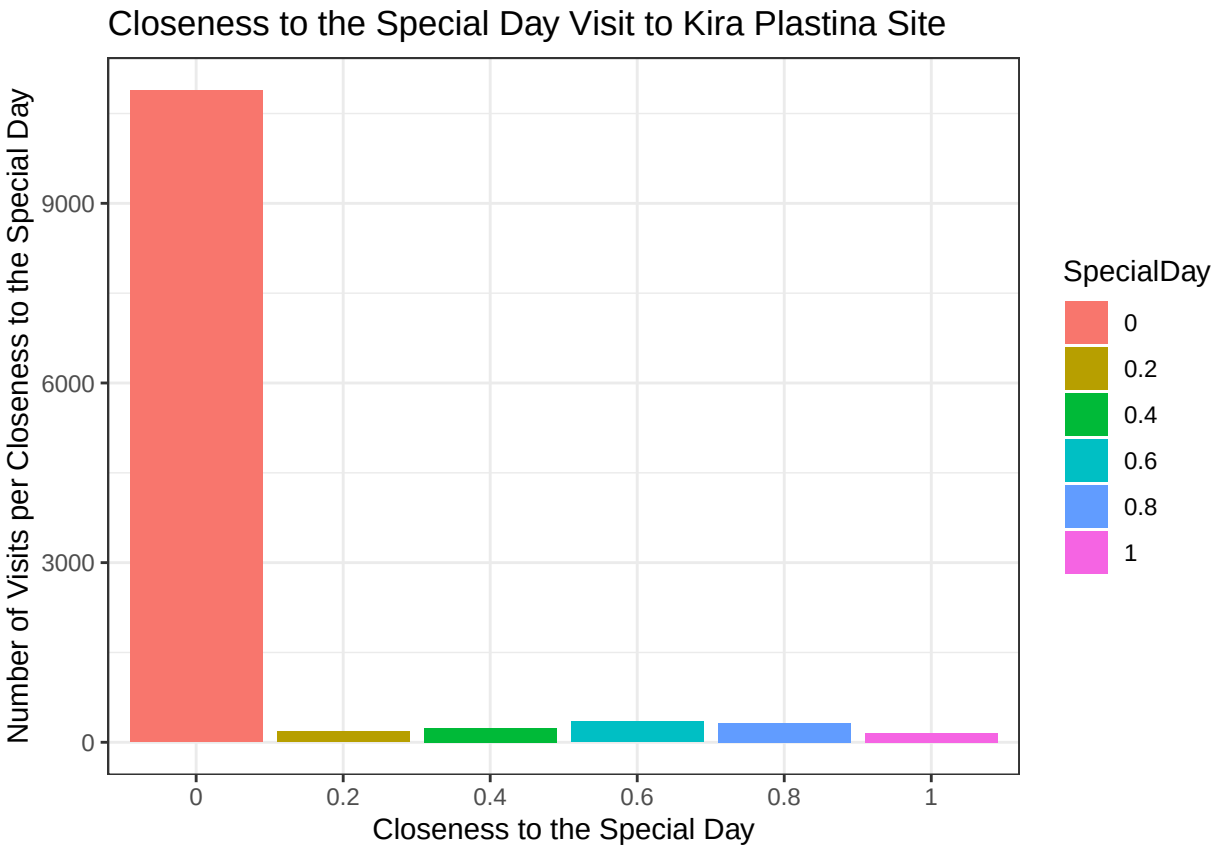
```
shoppers.notdup$Month[shoppers.notdup$Month == "June"] <- "Jun"
```

```
unique(shoppers.notdup$Month) #confirming tha t June has been converted to Jun
```

```
##  [1] "Feb" "Mar" "May" "Oct" "Jun" "Jul" "Aug" "Nov" "Sep" "Dec"
```

2. SpecialDay

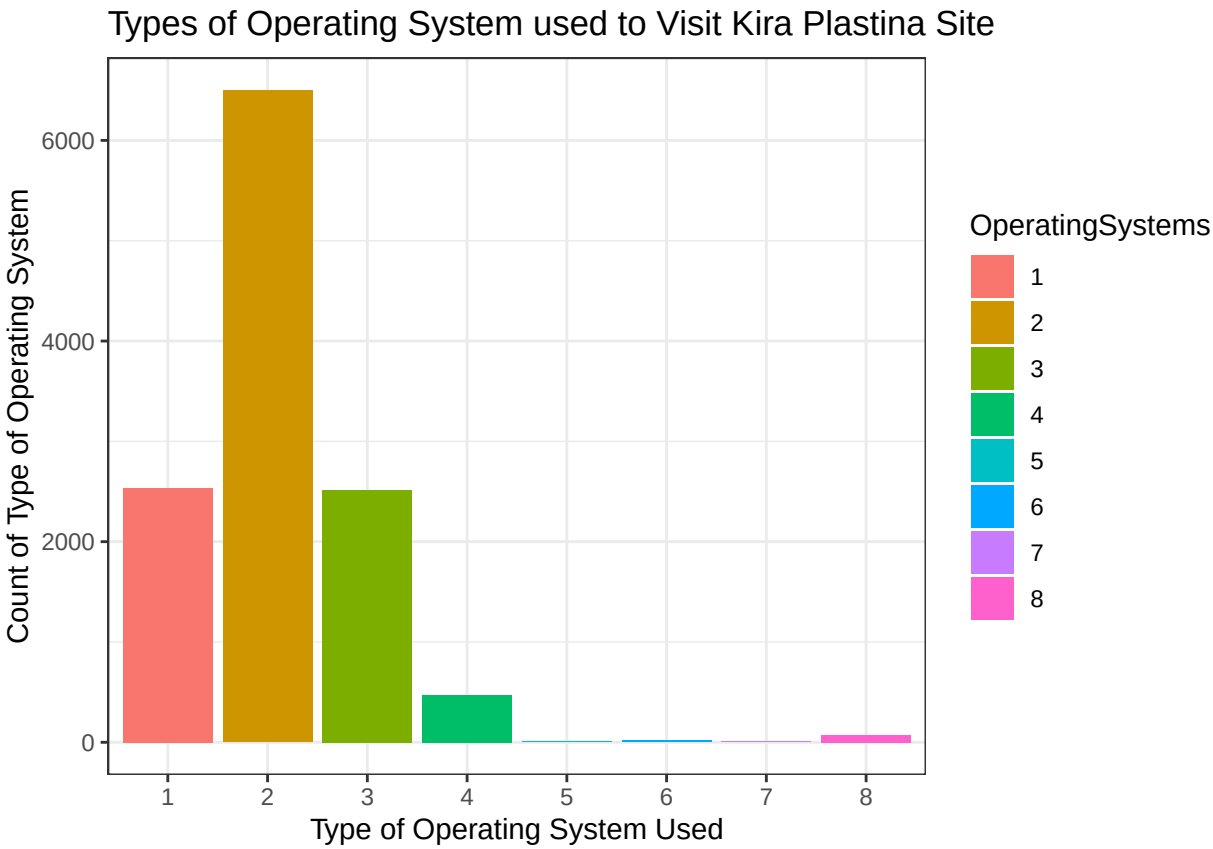
```
ggplot(shoppers.notdup, aes(x = SpecialDay, fill = SpecialDay)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Closeness to the Special Day", y = "Number of Visits per Closeness to the Special Day", tit
```



Majority of the visits to the site took place on a day not close to a special day. If a visit took place on a day not close to any special day, it took a value of zero. Our plot clearly shows that most visits happened on a day not close to any special day.

3. Operating Systems

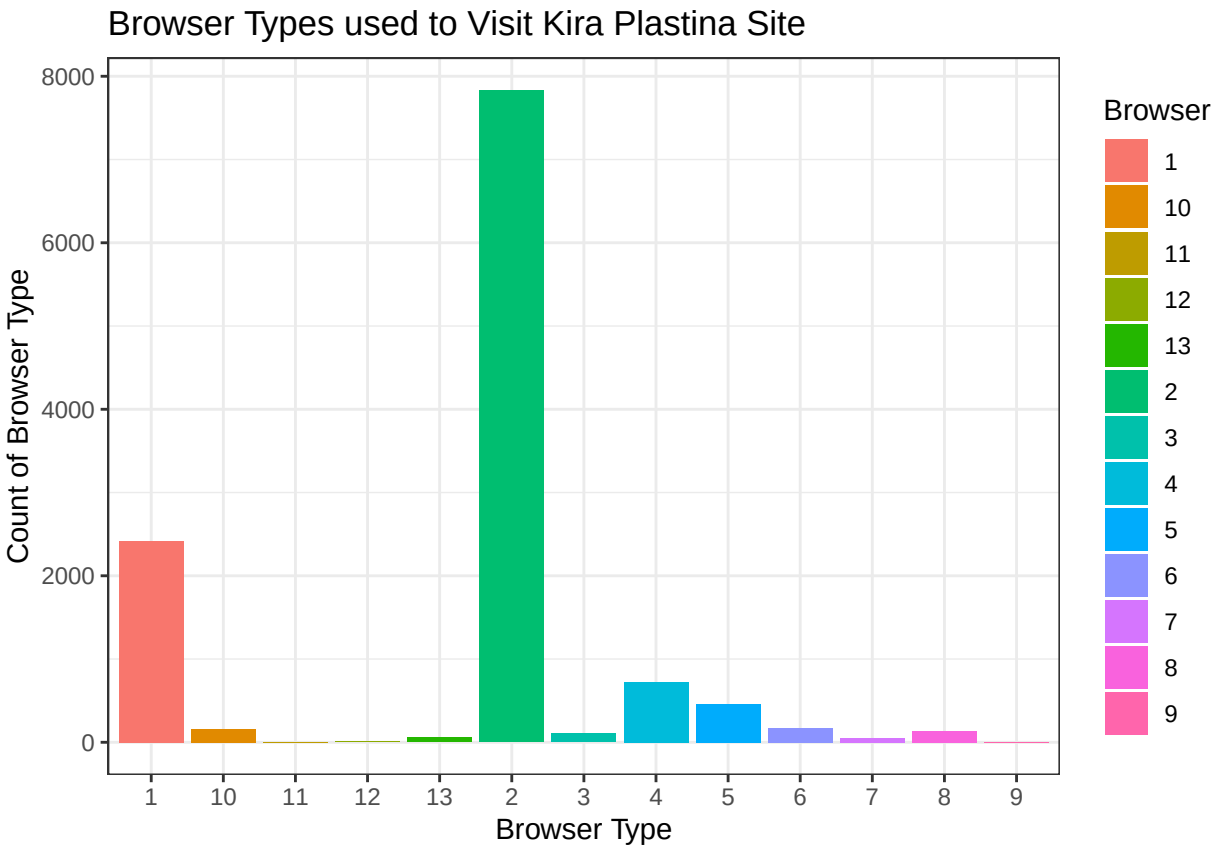
```
ggplot(shoppers.notdup, aes(x = OperatingSystems, fill = OperatingSystems)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Type of Operating System Used", y = "Count of Type of Operating System", title = "Types of
```



The most popular type of Operating system used was operating system 2 followed by 3 and 1. Operating System 5 was the least popular

4. Browser

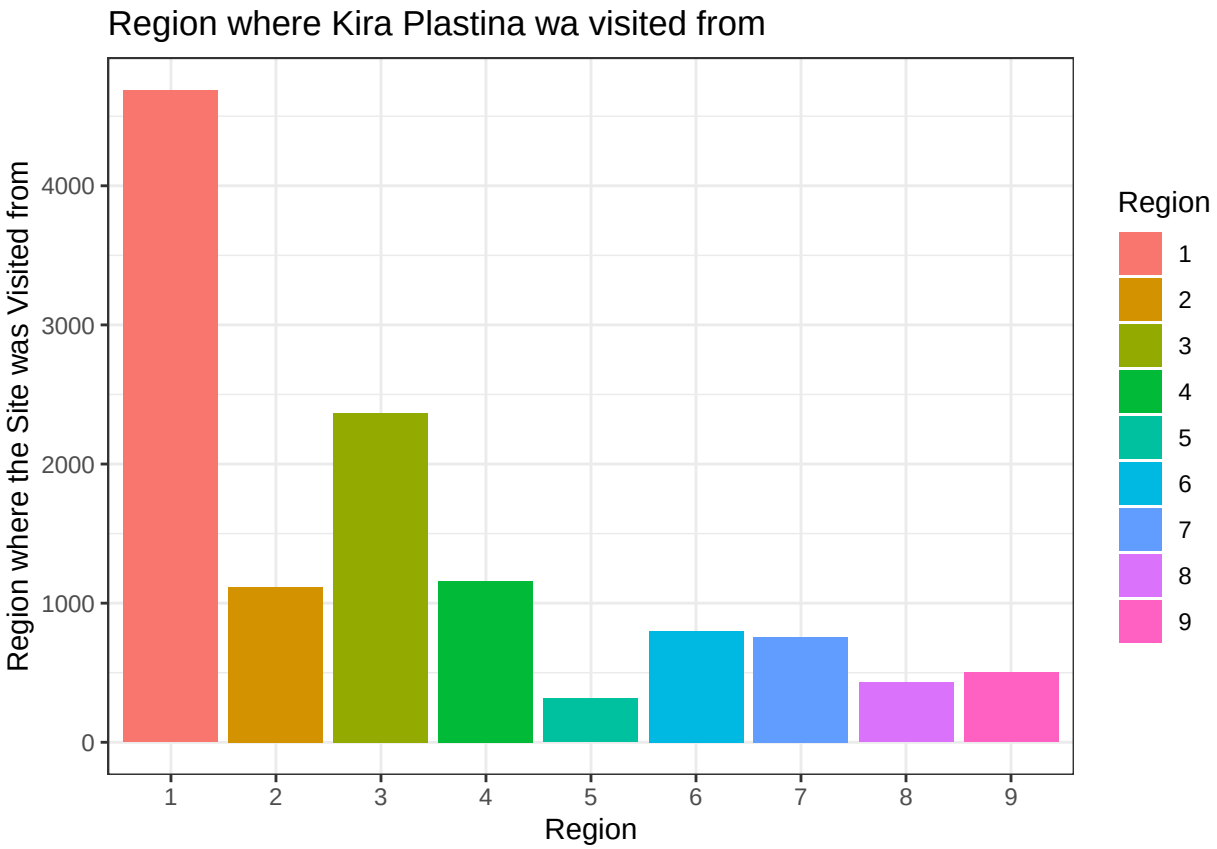
```
ggplot(shoppers.notdup, aes(x = Browser, fill = Browser)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Browser Type", y = "Count of Browser Type", title = "Browser Types used to Visit Kira Plasti
```



Browser 2 was the most commonly used to visit Kira Plastina followed by 1. 9 and 11 were the least popular.

5. Region

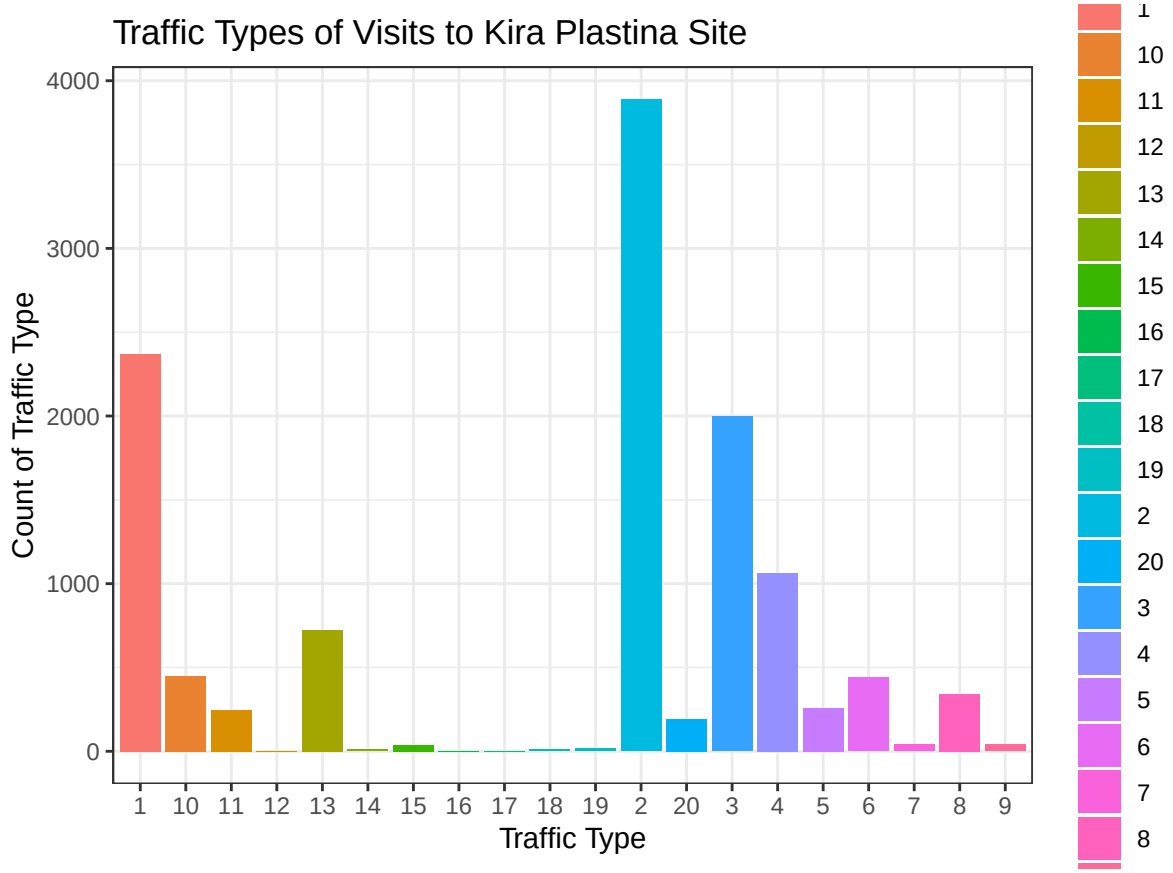
```
ggplot(shoppers.notdup, aes(x = Region, fill = Region)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Region", y = "Region where the Site was Visited from", title = "Region where Kira Plastina
```



In total, there were 9 regions. Region 1 and 3 had the most number of visit to Kira plastina. Region 5 had the least visits.

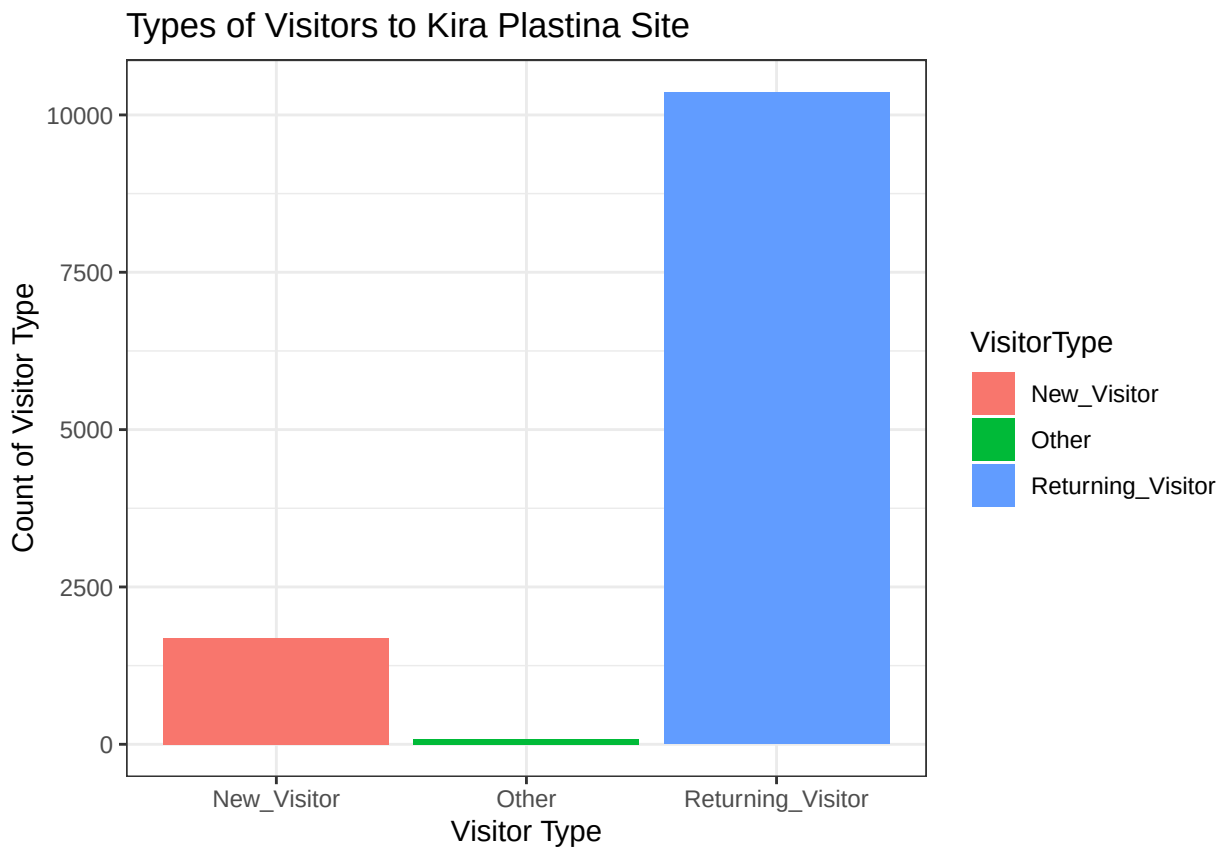
6. Traffic Type

```
ggplot(shoppers.notdup, aes(x = TrafficType, fill = TrafficType)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Traffic Type", y = "Count of Traffic Type", title = "Traffic Types of Visits to Kira Plastin
```



7. Visitor Type

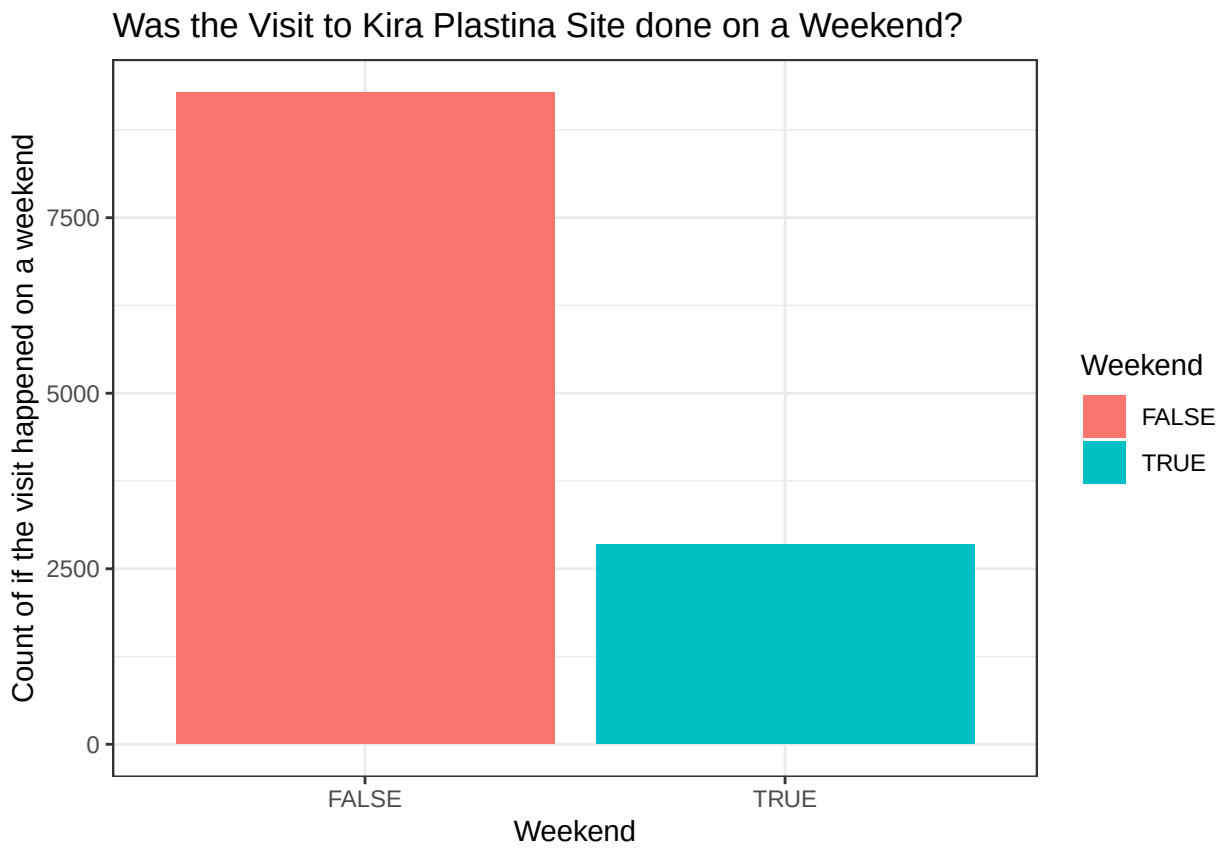
```
ggplot(shoppers.notdup, aes(x = VisitorType, fill = VisitorType)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Visitor Type", y = "Count of Visitor Type", title = "Types of Visitors to Kira Plastina Sit
```



Returning visitors were the most popular type of visitors to Kira Plastina site. This is really good for Kira Plastina. They need to maintain and continue improving this number as they work on bringing in new visitors who can end up being returning visitors.

8. Weekend

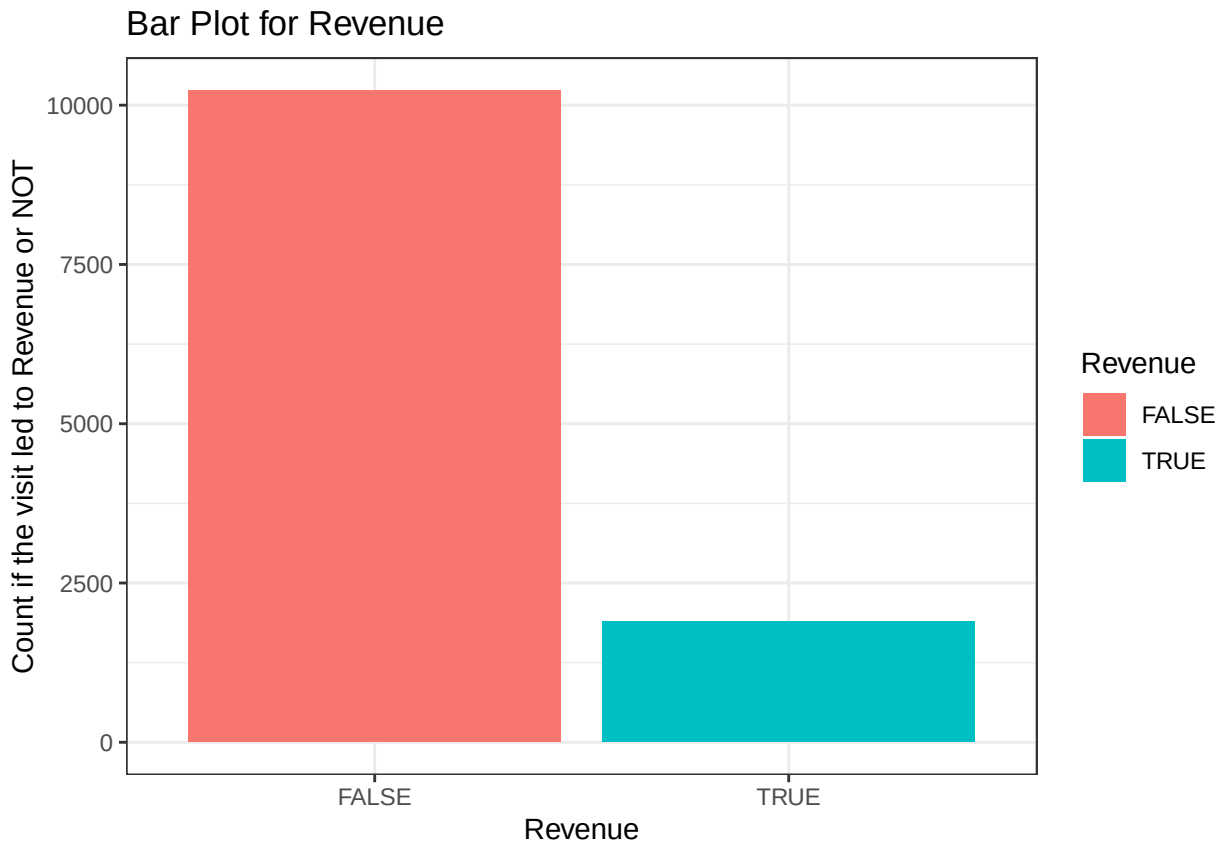
```
ggplot(shoppers.notdup, aes(x = Weekend, fill = Weekend)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Weekend", y = "Count of if the visit happened on a weekend", title = "Was the Visit to Kira
```



Most visits to Kira plastina site were done on weekdays. ~2500 visits to the site were done on the weekends compared to ~8500 visits done on weekdays. Given that weekdays have 5 days while Weekends are only 2 days, the proportional number of visits on the weekends and weekdays are about close to each.

9. Revenue

```
ggplot(shoppers.notdup, aes(x = Revenue, fill = Revenue)) +
  geom_bar() +
  theme_bw() +
  labs(x = "Revenue", y = "Count if the visit led to Revenue or NOT", title = "Bar Plot for Revenue")
```

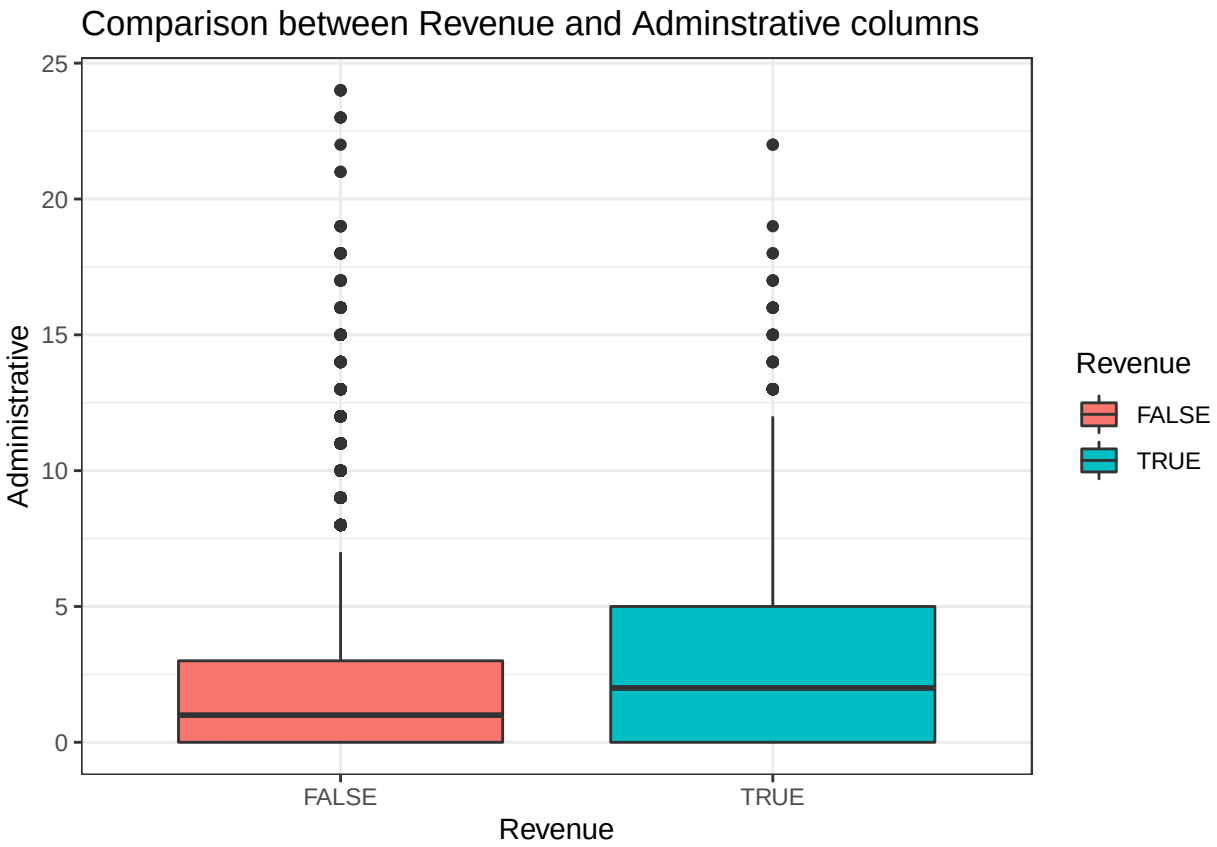


We see that majority of the visits did not lead to any revenue. This Revenue column is our target variable. We can check on how other attributes compare to this column and draw some insights on which kinds of groups bring revenue to Kira Plastina. It is important to understand how the different attributes contribute to Kira Plastina's revenue because this will inform the sales and marketing team on their strategies.

5.2 Bivariate Analysis

1. Revenue Vs Administrative

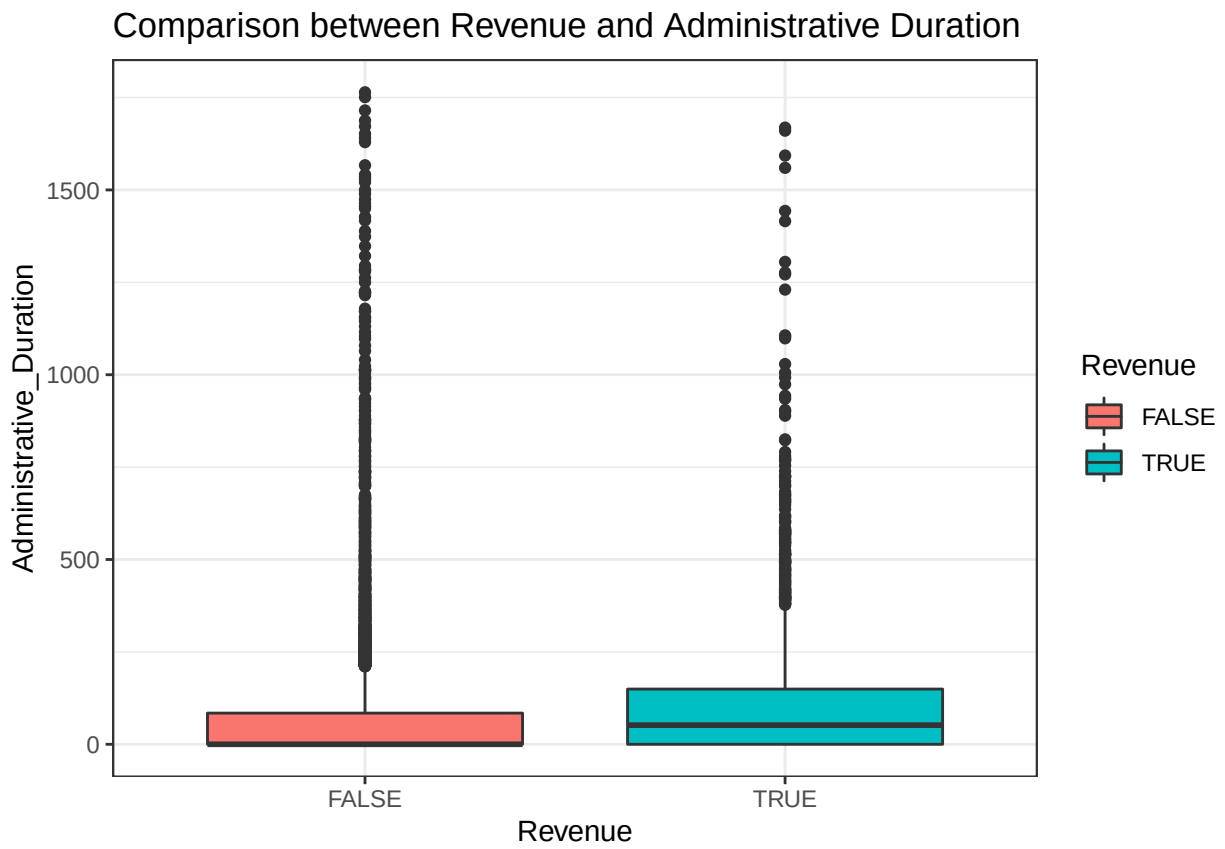
```
ggplot(shoppers.notdup, aes(x= Revenue, y = Administrative, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and Adminstrative columns")
```



From the graphs we can deduct that visitors who visited more administrative pages were likely to bring in revenue to Kira Plastina — the visitors bought some items from the site.

2. Revenue Vs Administrative_Duration

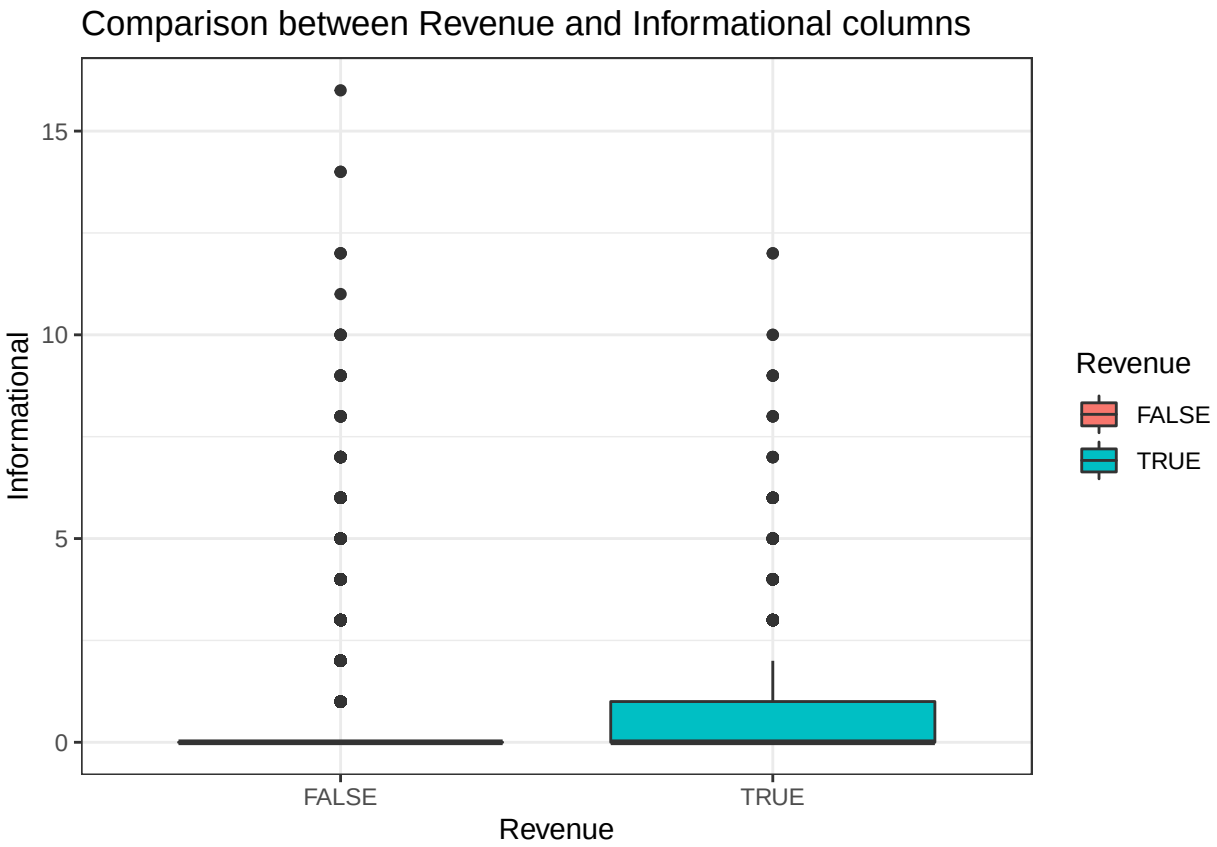
```
ggplot(shoppers.notdup, aes(x= Revenue, y = Administrative_Duration, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and Administrative Duration")
```



Some visitors who stayed longer in administrative pages brought more revenue to Kira Plastina. We also notice a lot of outliers on both bar plots.

3. Revenue Vs Informational

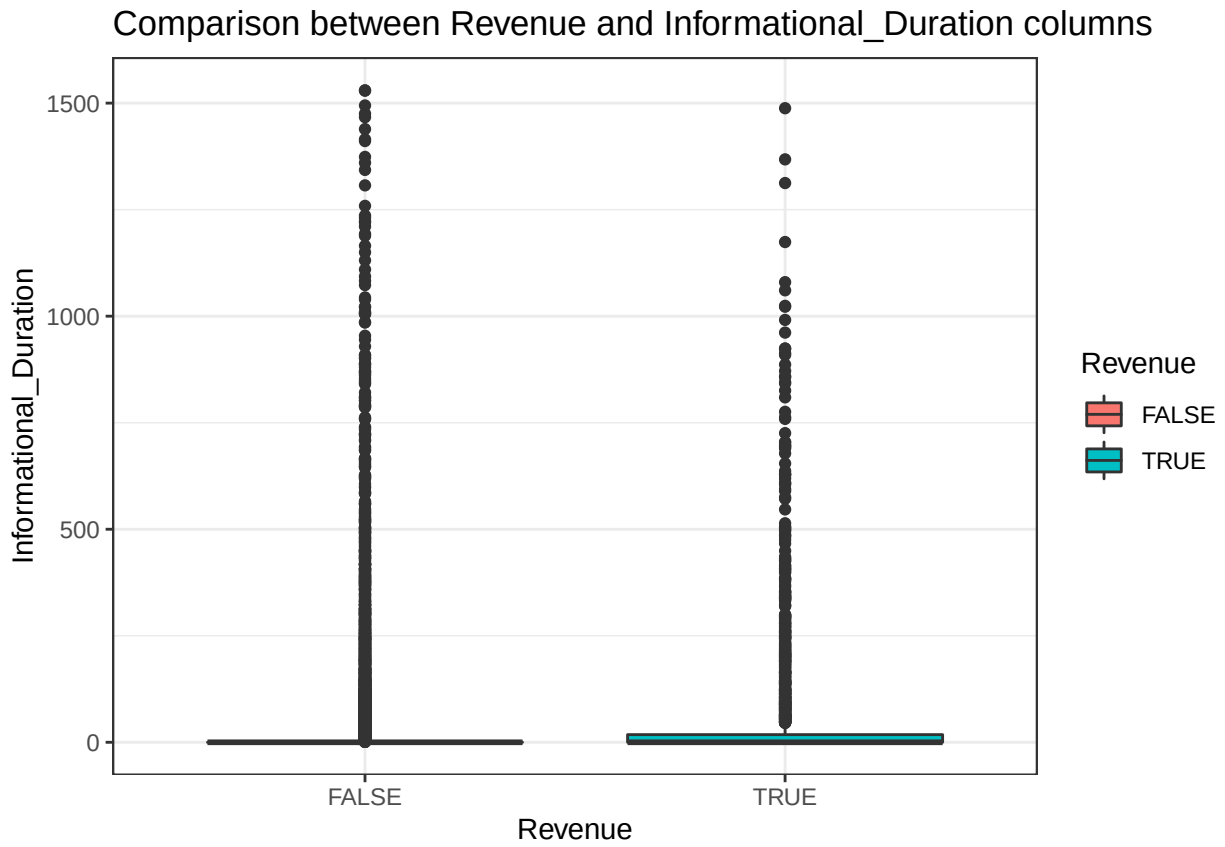
```
ggplot(shoppers.notdup, aes(x= Revenue, y = Informational, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and Informational columns")
```



Majority of the visitors who visited informational type pages contributed to Revenue in Kira Plastina.

4. Revenue Vs Informational Duration

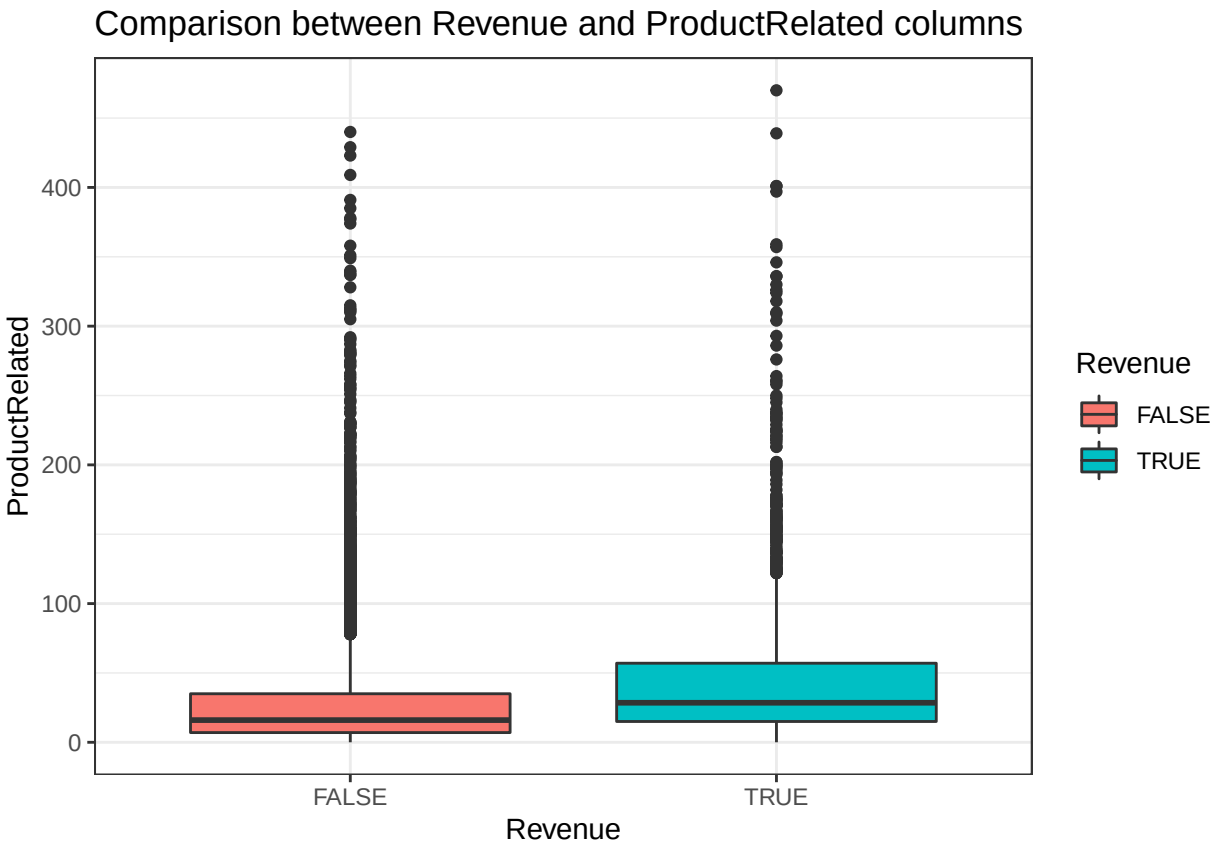
```
ggplot(shoppers.notdup, aes(x= Revenue, y = Informational_Duration, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and Informational_Duration columns")
```



We have a lot of outliers in informational duration column, but we see that visitors who stayed for a few seconds in informational pages contributed to revenue at Kira Plastina unlike visitors who did not spend time at the site.

5. Revenue Vs Product Related

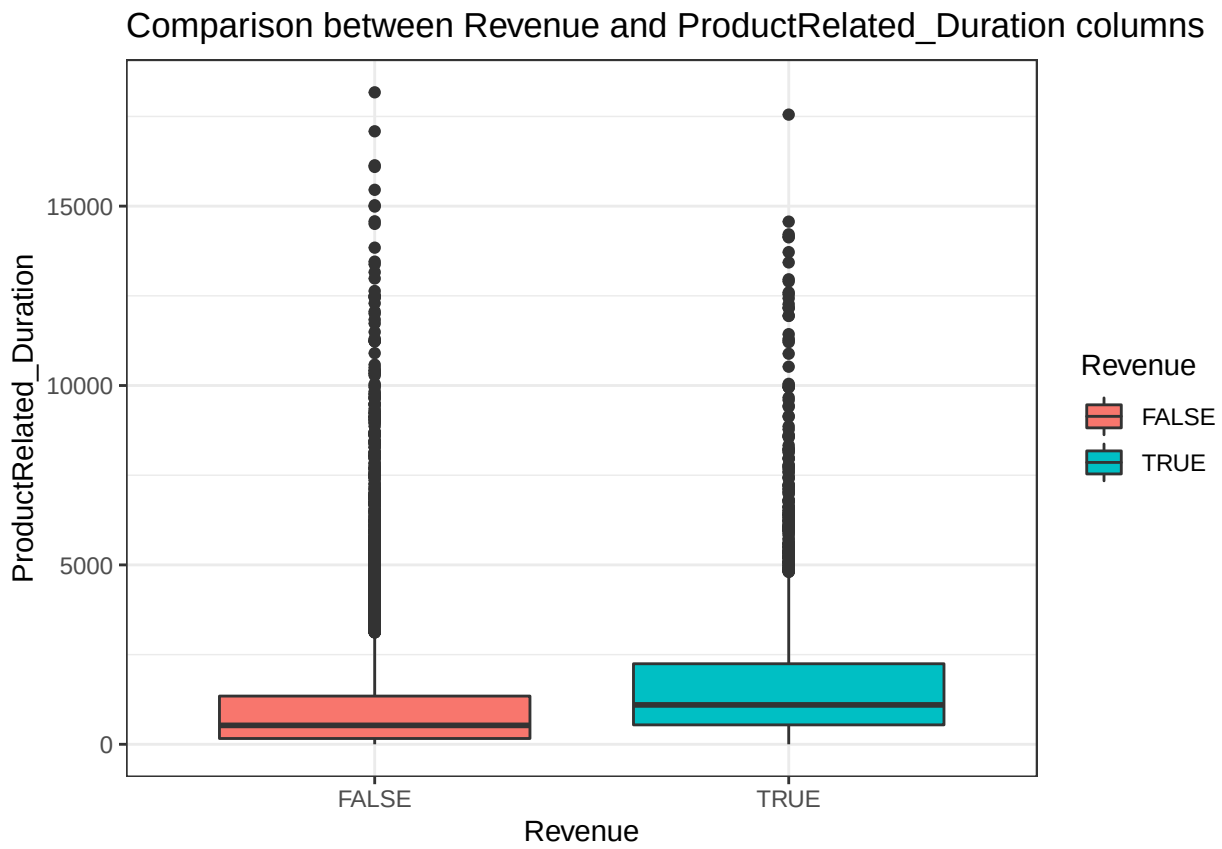
```
ggplot(shoppers.notdup, aes(x= Revenue, y = ProductRelated, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and ProductRelated columns")
```



Visitors who visited more product related pages contributed more revenue to Kira than visitors who visited less product related pages. Here we see a major issue with outliers as well.

6. Revenue Vs ProductRelated_Duration

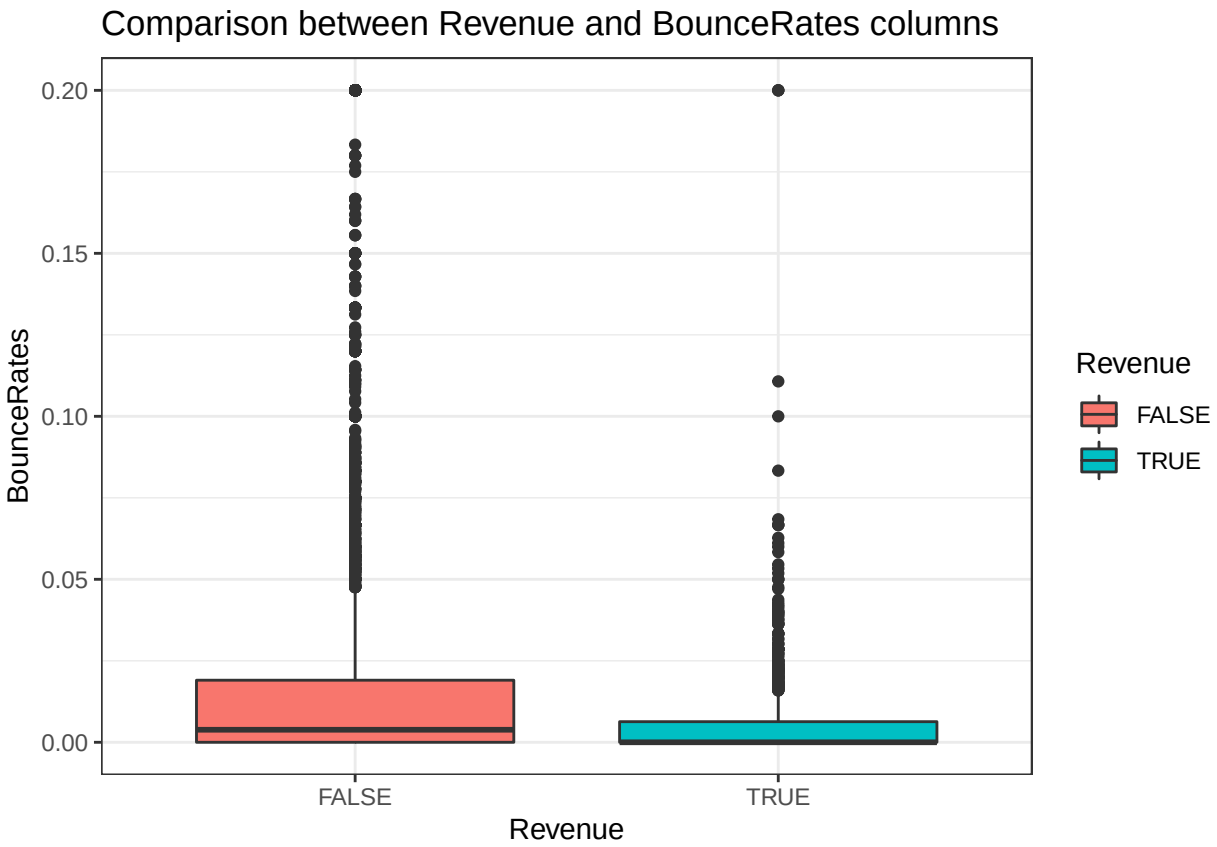
```
ggplot(shoppers.notdup, aes(x= Revenue, y = ProductRelated_Duration, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and ProductRelated_Duration columns")
```



The longer a customer visited product related pages, the more revenue they will contribute to Kira. Maybe these were customers who were comparing price and quality of related products to Kira Plastina, and ended up preferring products from Kira.

7. Revenue Vs BounceRates

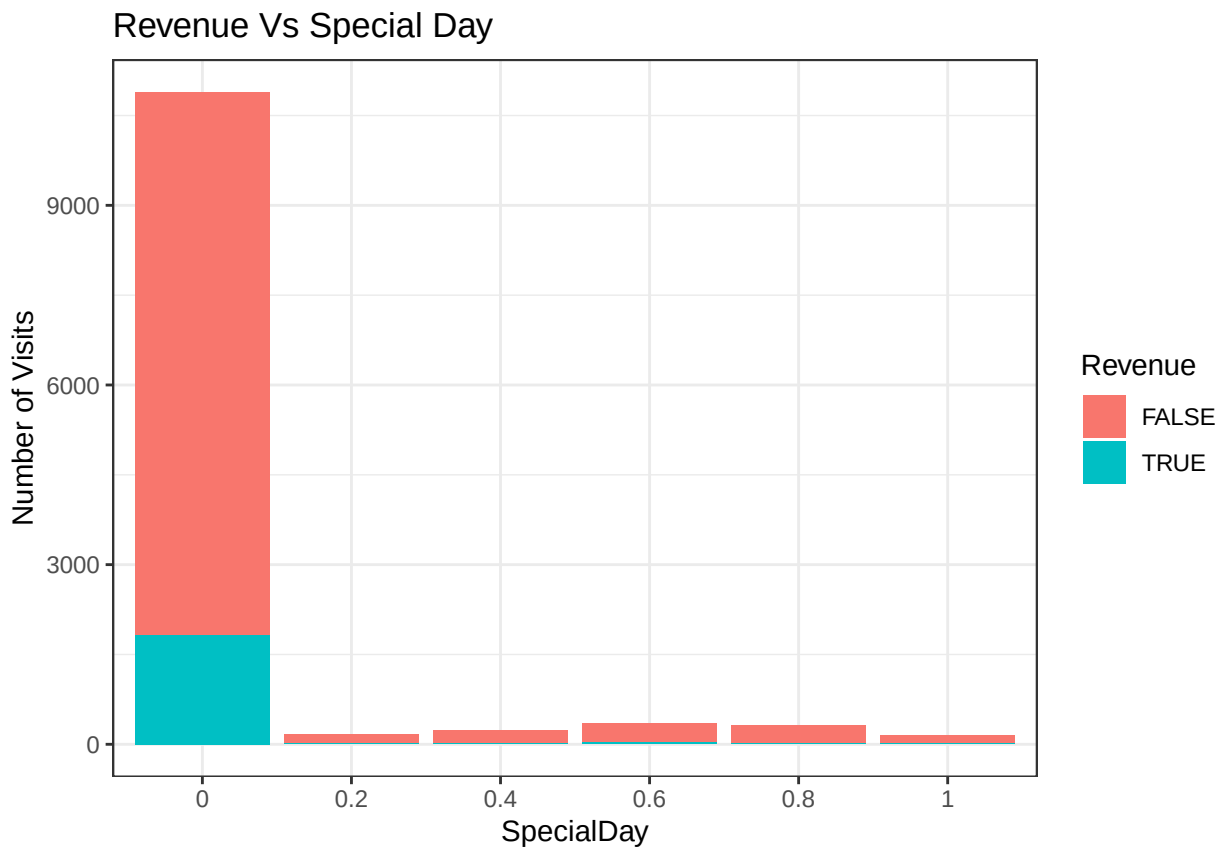
```
ggplot(shoppers.notdup, aes(x= Revenue, y = BounceRates, fill = Revenue)) +
  theme_bw() +
  geom_boxplot()+
  labs(title = "Comparison between Revenue and BounceRates columns")
```



Visitors with lower bounce rates contributed some revenue, but as the bounce rates increased, Kira did not get revenue from the visitors with higher bounce rates.

8. Revenue Vs SpecialDay

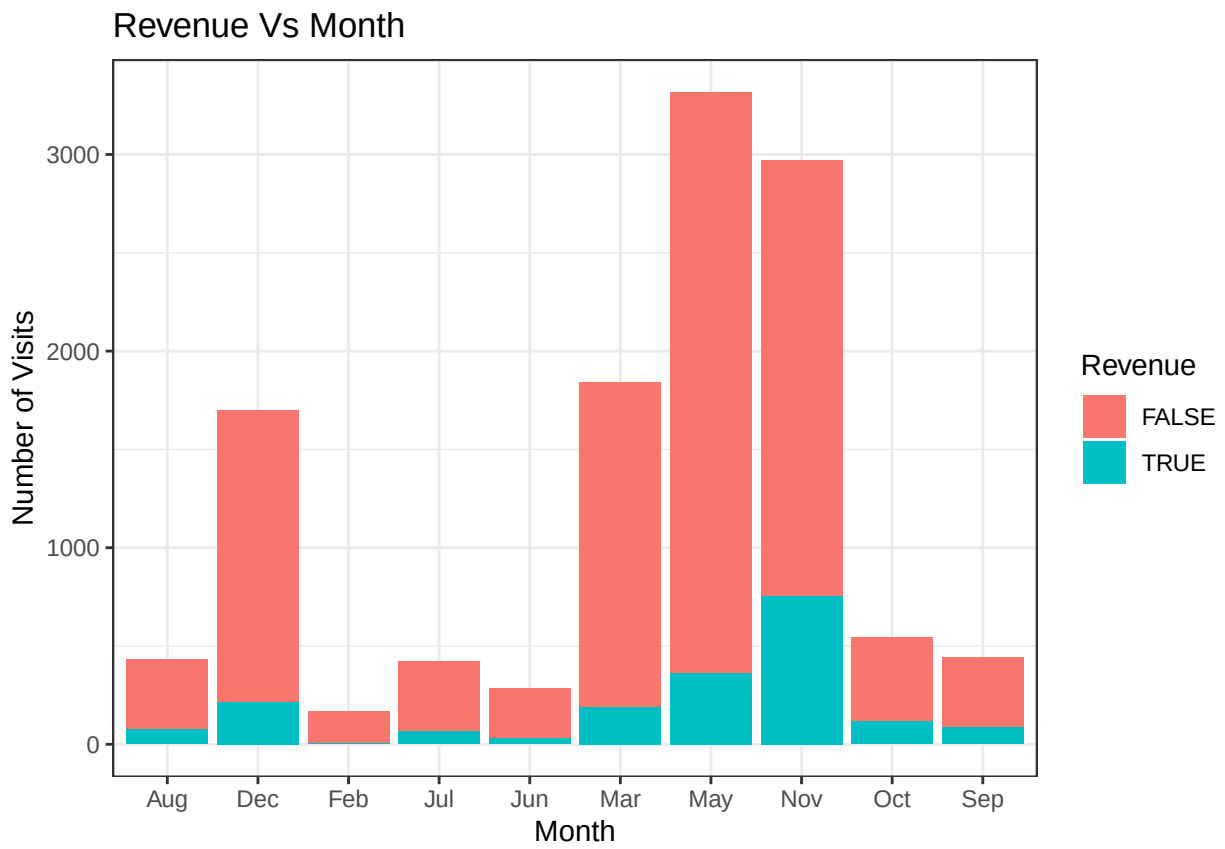
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$SpecialDay <- as.factor(shoppers.notdup$SpecialDay)
ggplot(shoppers.notdup, aes(x=SpecialDay, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Special Day")
```



A lot of visits to Kira plastina took place not close to any special day. From this visit(barplot 0), only about 25% led to revenue.

9. Revenue Vs Month

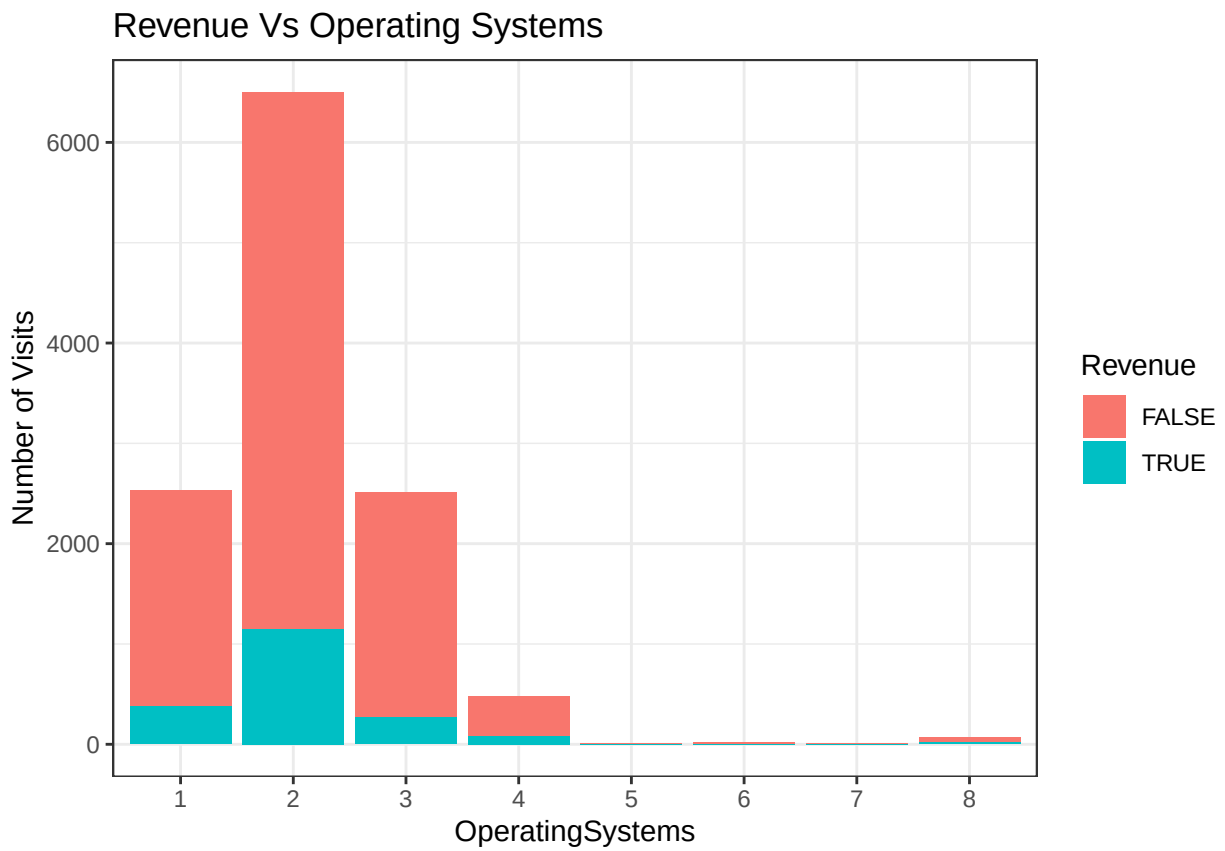
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$Month <- as.factor(shoppers.notdup$Month)
ggplot(shoppers.notdup, aes(x=Month, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Month")
```



November which has the 2nd largest number of visits to the site had the biggest revenue collection compared to the rest of the months. November was followed by May, and Dec in revenue collected

10. Revenue Vs OperatingSystems

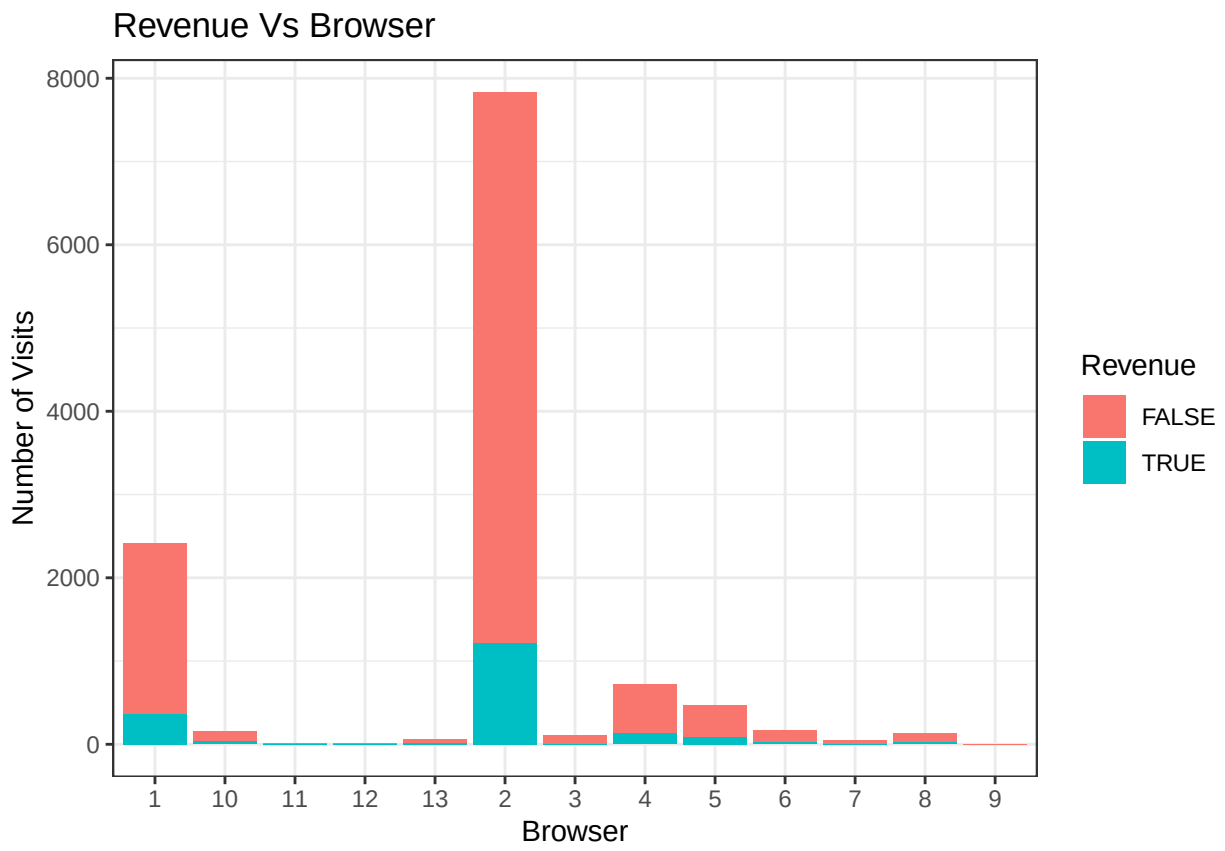
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$OperatingSystems <- as.factor(shoppers.notdup$OperatingSystems)
ggplot(shoppers.notdup, aes(x=OperatingSystems, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Operating Systems")
```



Number 2 operating system with the most visits has the most revenue collection, followed by 1 and 3 respectively. With this insight, we can advise the team to focus on operating system 2 because of the increased revenue it contributes for Kira Plastina.

11. Revenue Vs Browser

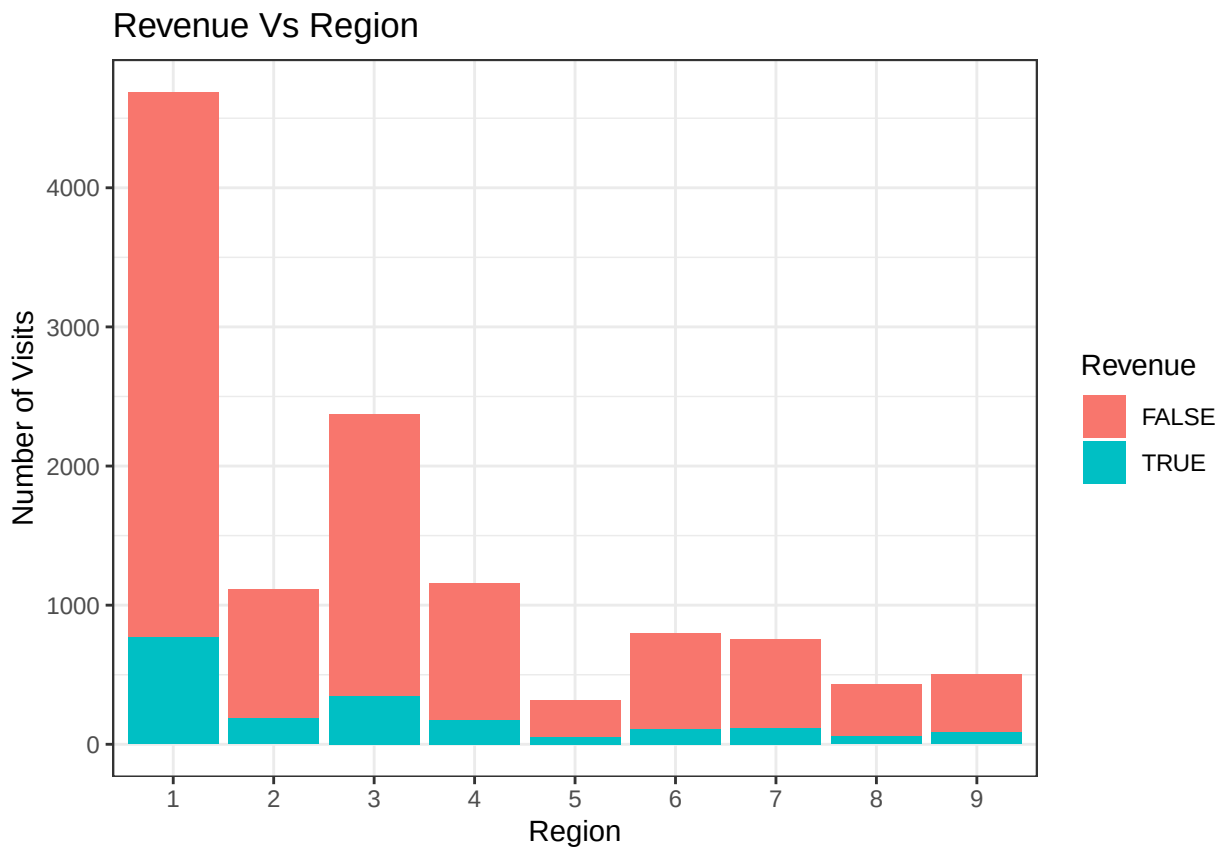
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$Browser <- as.factor(shoppers.notdup$Browser)
ggplot(shoppers.notdup, aes(x=Browser, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Browser")
```



Number 2 browser with the most visits had most revenue collected here for Kira Plastina

12. Revenue Vs Region

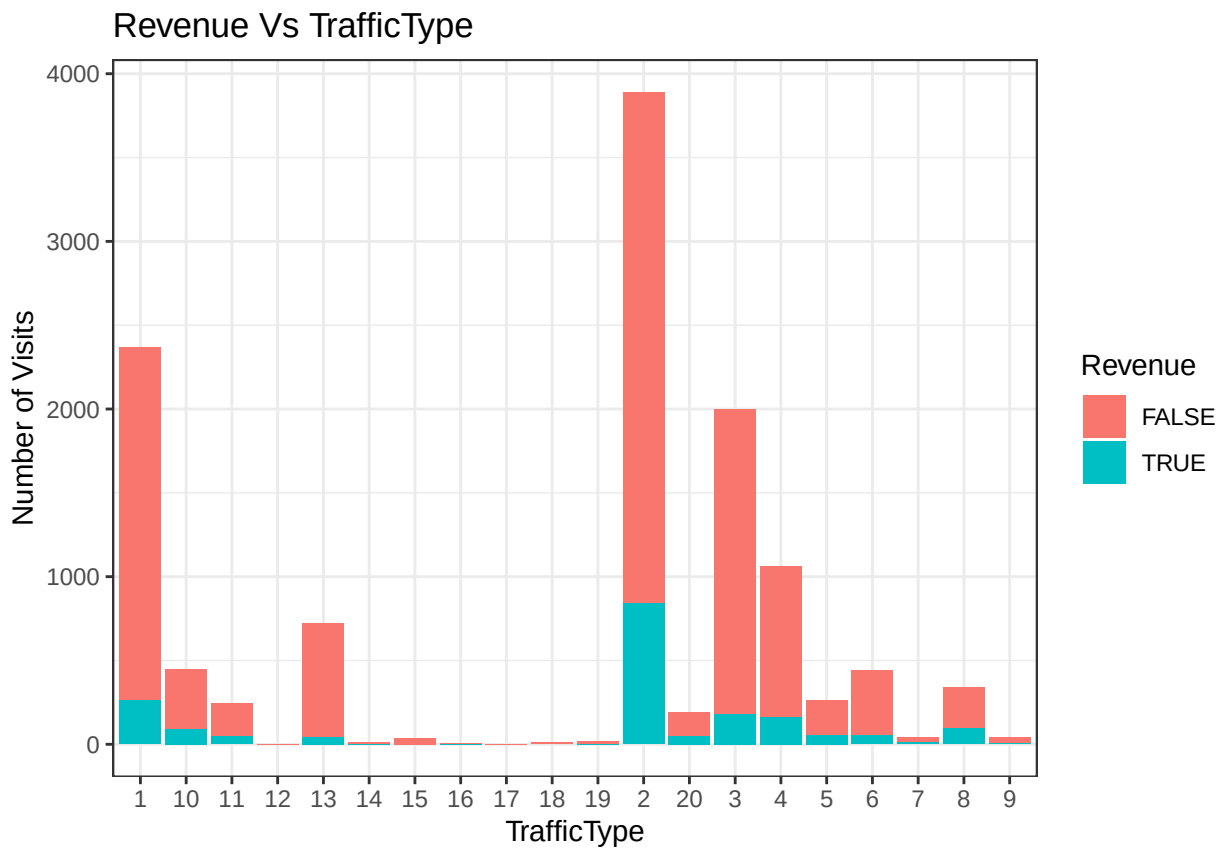
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$Region <- as.factor(shoppers.notdup$Region)
ggplot(shoppers.notdup, aes(x=Region, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Region")
```



Region 1 and 3 contributed the most revenue to Kira Plastina.

13. Revenue Vs Traffic Type

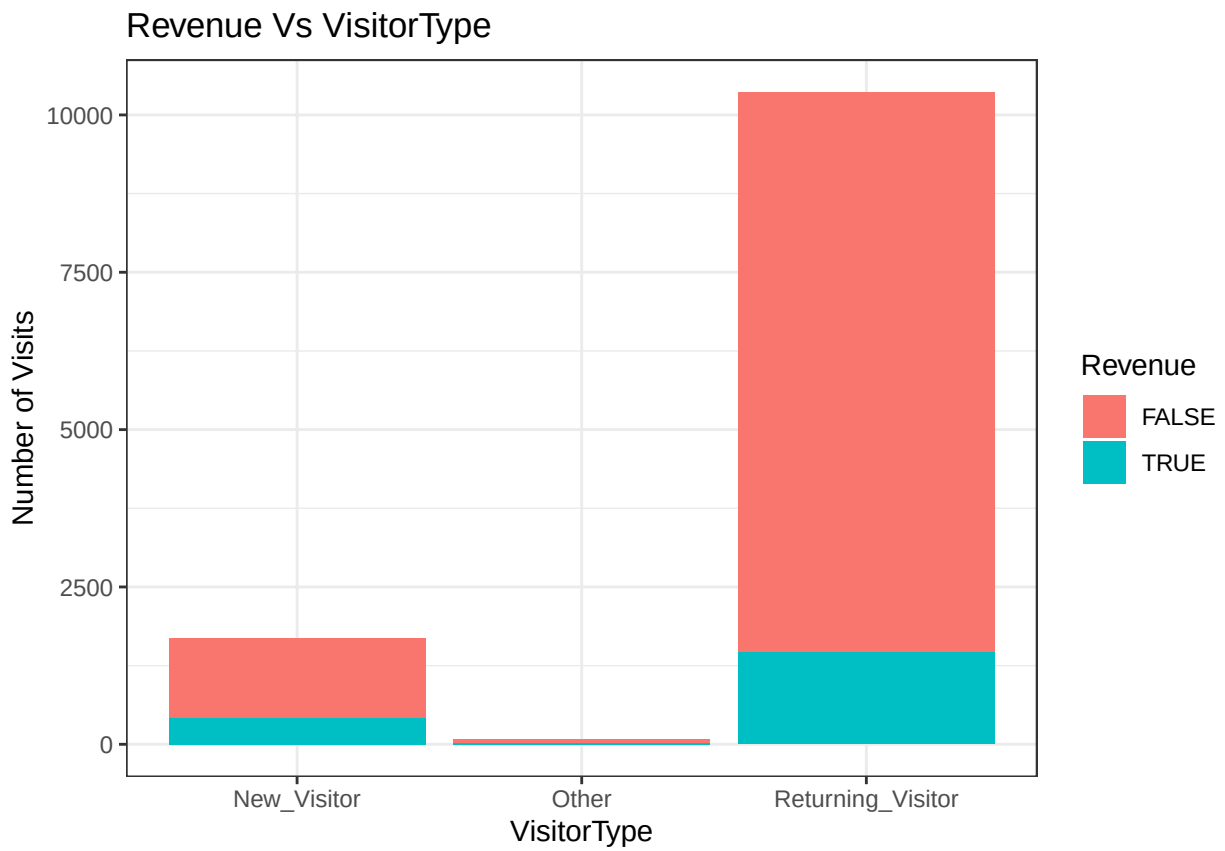
```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$TrafficType <- as.factor(shoppers.notdup$TrafficType)
ggplot(shoppers.notdup, aes(x=TrafficType, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs TrafficType")
```



Traffic type 2 brought in the most revenue followed closely by traffic type 1.

14. Revenue Vs Visitor Type

```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$VisitorType <- as.factor(shoppers.notdup$VisitorType)
ggplot(shoppers.notdup, aes(x=VisitorType, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs VisitorType")
```



Returning Visitor contributed the most revenue to Kira Plastina. This is an area that the brand sales and marketing team can leverage to retain these customers.

15. Revenue Vs Weekend

```
shoppers.notdup$Revenue <- as.factor(shoppers.notdup$Revenue)
shoppers.notdup$Weekend <- as.factor(shoppers.notdup$Weekend)
ggplot(shoppers.notdup, aes(x=Weekend, fill = Revenue))+
  theme_bw()+
  geom_bar()+
  labs(y="Number of Visits", title = "Revenue Vs Weekend")
```

The non-weekend visits to Kira are bringing in more revenue compared to weekends.

6. Implementing the Solution

6.1 K-Means Clustering

6.1.1 Preprocessing data

```
#since K-Means and Hierachical clustering is a type of Unsupervised Learning,
# we would not require the target variable (Revenue) during execution of our algorithms.
# We will, therefore, remove revenue column and store it in another variable.

shoppers.new<- shoppers.notdup[, c(1:15)]
head(shoppers.new)
```

```
##   Administrative Administrative_Duration Informational Informational_Duration
## 1              0                       0             0                      0
## 2              0                       0             0                      0
## 4              0                       0             0                      0
## 5              0                       0             0                      0
## 6              0                       0             0                      0
## 9              0                       0             0                      0
##   ProductRelated ProductRelated_Duration BounceRates SpecialDay Month
## 1              1                 0.000000  0.20000000          0   Feb
## 2              2                64.000000  0.00000000          0   Feb
## 4              2                 2.666667  0.05000000          0   Feb
## 5             10               627.500000  0.02000000          0   Feb
```

```
## 6             19              154.216667  0.01578947          0   Feb
## 9              2               37.000000  0.00000000        0.8   Feb
##   OperatingSystems Browser Region TrafficType       VisitorType Weekend
## 1                1       1      1           1 Returning_Visitor   FALSE
## 2                2       2      1           2 Returning_Visitor   FALSE
## 4                3       2      2           4 Returning_Visitor   FALSE
## 5                3       3      1           4 Returning_Visitor    TRUE
## 6                2       2      1           3 Returning_Visitor   FALSE
## 9                2       2      2           3 Returning_Visitor   FALSE
```

```
shoppers.revenue <- shoppers.notdup[, "Revenue"]
head(shoppers.revenue)
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
## Levels: FALSE TRUE
```

```
glimpse(shoppers.new) #checking the shape of our dataset before we begin modeling
```

```
## Rows: 12,131
## Columns: 15
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0...
## $ Administrative_Duration <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 2, 10, 19, 2, 3, 3, 16, 7, 6, 2, 23, ...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, 2.666667, 627.500000, ...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.050000000, 0.02...
## $ SpecialDay              <fct> 0, 0, 0, 0, 0, 0.8, 0.4, 0, 0.4, 0, 0, 0, 0...
## $ Month                   <fct> Feb, Feb, Feb, Feb, Feb, Feb, Feb, Feb, Feb...
## $ OperatingSystems        <fct> 1, 2, 3, 3, 2, 2, 2, 1, 1, 1, 2, 3, 1, 1, 2...
## $ Browser                 <fct> 1, 2, 2, 3, 2, 2, 4, 1, 1, 1, 5, 2, 1, 1, 2...
## $ Region                  <fct> 1, 1, 2, 1, 1, 2, 1, 3, 4, 1, 1, 3, 9, 1, 1...
## $ TrafficType             <fct> 1, 2, 4, 4, 3, 3, 2, 3, 3, 3, 3, 3, 3, 4, 3...
## $ VisitorType             <fct> Returning_Visitor, Returning_Visitor, Retur...
## $ Weekend                 <fct> FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FA...
```

```
#encoding categorical variables. We will use one hot encode for unordered categorical columns e.g the w
shoppers.new$Month <- match(shoppers.new$Month, month.abb)
unique(shoppers.new$Month)
```

```
##  [1]  2  3  5 10  6  7  8 11  9 12
```

```
shoppers.new$Month <- as.factor(shoppers.new$Month)
glimpse(shoppers.new)
```

```
## Rows: 12,131
## Columns: 15
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0...
## $ Administrative_Duration <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 2, 10, 19, 2, 3, 3, 16, 7, 6, 2, 23, ...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, 2.666667, 627.500000, ...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.050000000, 0.02...
## $ SpecialDay              <fct> 0, 0, 0, 0, 0, 0.8, 0.4, 0, 0.4, 0, 0, 0, 0...
## $ Month                   <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2...
## $ OperatingSystems        <fct> 1, 2, 3, 3, 2, 2, 2, 1, 1, 1, 2, 3, 1, 1, 2...
```

```
## $ Browser                 <fct> 1, 2, 2, 3, 2, 2, 4, 1, 1, 1, 5, 2, 1, 1, 2...
## $ Region                  <fct> 1, 1, 2, 1, 1, 2, 1, 3, 4, 1, 1, 3, 9, 1, 1...
## $ TrafficType             <fct> 1, 2, 4, 4, 3, 3, 2, 3, 3, 3, 3, 3, 3, 4, 3...
## $ VisitorType             <fct> Returning_Visitor, Returning_Visitor, Retur...
## $ Weekend                 <fct> FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FA...
```

```
#encoding visitor type column
#first convert visitor type and weekend to character
shoppers.new$VisitorType <- as.character(shoppers.new$VisitorType)
shoppers.new$Weekend <- as.character(shoppers.new$Weekend)
#then convert the two categorical variable to numerical using factors
shoppers.new$Weekend <- as.integer(as.factor(shoppers.new$Weekend))
shoppers.new$VisitorType <- as.integer(as.factor(shoppers.new$VisitorType))
glimpse(shoppers.new)
```

```
## Rows: 12,131
## Columns: 15
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0...
## $ Administrative_Duration <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 2, 10, 19, 2, 3, 3, 16, 7, 6, 2, 23, ...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, 2.666667, 627.500000, ...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.050000000, 0.02...
## $ SpecialDay              <fct> 0, 0, 0, 0, 0, 0.8, 0.4, 0, 0.4, 0, 0, 0, 0...
## $ Month                   <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2...
## $ OperatingSystems        <fct> 1, 2, 3, 3, 2, 2, 2, 1, 1, 1, 2, 3, 1, 1, 2...
## $ Browser                 <fct> 1, 2, 2, 3, 2, 2, 4, 1, 1, 1, 5, 2, 1, 1, 2...
## $ Region                  <fct> 1, 1, 2, 1, 1, 2, 1, 3, 4, 1, 1, 3, 9, 1, 1...
## $ TrafficType             <fct> 1, 2, 4, 4, 3, 3, 2, 3, 3, 3, 3, 3, 3, 4, 3...
## $ VisitorType             <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3...
## $ Weekend                 <int> 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1...
```

```
#converting all the factors to integers
shoppers.new$SpecialDay <- as.integer(shoppers.new$SpecialDay)
shoppers.new$Month <- as.integer(shoppers.new$Month)
shoppers.new$OperatingSystems <- as.integer(shoppers.new$OperatingSystems)
shoppers.new$Browser <- as.integer(shoppers.new$Browser)
shoppers.new$Region <- as.integer(shoppers.new$Region)
shoppers.new$TrafficType <- as.integer(shoppers.new$TrafficType)
glimpse(shoppers.new) #confirming that all the factors have been converted to integers
```

```
## Rows: 12,131
## Columns: 15
## $ Administrative          <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0...
## $ Administrative_Duration <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0...
## $ Informational           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <int> 1, 2, 2, 10, 19, 2, 3, 3, 16, 7, 6, 2, 23, ...
## $ ProductRelated_Duration <dbl> 0.000000, 64.000000, 2.666667, 627.500000, ...
## $ BounceRates             <dbl> 0.200000000, 0.000000000, 0.050000000, 0.02...
## $ SpecialDay              <int> 1, 1, 1, 1, 1, 5, 3, 1, 3, 1, 1, 1, 1, 1, 1...
## $ Month                   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ OperatingSystems        <int> 1, 2, 3, 3, 2, 2, 2, 1, 1, 1, 2, 3, 1, 1, 2...
## $ Browser                 <int> 1, 6, 6, 7, 6, 6, 8, 1, 1, 1, 9, 6, 1, 1, 6...
## $ Region                  <int> 1, 1, 2, 1, 1, 2, 1, 3, 4, 1, 1, 3, 9, 1, 1...
```

```
## $ TrafficType             <int> 1, 12, 15, 15, 14, 14, 12, 14, 14, 14, 14, ...
## $ VisitorType             <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3...
## $ Weekend                 <int> 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1...
```

Our data is now in the right form for modeling.

6.1.2 Normalizing the Data

```
#Normalizing data using min-max scaler
normalize.shoppers <- function(x) {return((x-min(x)) / (max(x) - min(x)))}
shoppers.normalized <- shoppers.new %>%
  select(Administrative, Administrative_Duration,Informational, Informational_Duration, ProductRelated,
         ProductRelated_Duration, BounceRates, Month, OperatingSystems, Browser,
         Region, TrafficType, VisitorType,
         Weekend) %>%
  normalize.shoppers() %>%
  glimpse()
```

```
## Rows: 12,131
## Columns: 14
## $ Administrative          <dbl> 0.000000e+00, 0.000000e+00, 0.000000e+00, 0...
## $ Administrative_Duration <dbl> 0.000000000, 0.000000000, 0.000000000, 0.00...
## $ Informational           <dbl> 0.000000e+00, 0.000000e+00, 0.000000e+00, 0...
## $ Informational_Duration  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ ProductRelated          <dbl> 5.503034e-05, 1.100607e-04, 1.100607e-04, 5...
## $ ProductRelated_Duration <dbl> 0.0000000000, 0.0035219416, 0.0001467476, 0...
## $ BounceRates             <dbl> 1.100607e-05, 0.000000e+00, 2.751517e-06, 1...
## $ Month                   <dbl> 5.503034e-05, 5.503034e-05, 5.503034e-05, 5...
## $ OperatingSystems        <dbl> 5.503034e-05, 1.100607e-04, 1.650910e-04, 1...
## $ Browser                 <dbl> 5.503034e-05, 3.301820e-04, 3.301820e-04, 3...
## $ Region                  <dbl> 5.503034e-05, 5.503034e-05, 1.100607e-04, 5...
## $ TrafficType             <dbl> 5.503034e-05, 6.603641e-04, 8.254551e-04, 8...
## $ VisitorType             <dbl> 0.000165091, 0.000165091, 0.000165091, 0.00...
## $ Weekend                 <dbl> 5.503034e-05, 5.503034e-05, 5.503034e-05, 1...
```

6.1.3 Applying K-Means with k=2

```
#applying K-Means with k=2. The k = 2 was randomly selected. Later on, we will perfom optimization usin
set.seed(123)
shoppers.normalizedk2 <- kmeans(shoppers.normalized, 2, nstart=25)
shoppers.normalizedk2$size
```

```
## [1] 11012  1119
```

```
shoppers.normalizedk2$betweenss
```

```
## [1] 63.92219
```

```
shoppers.normalizedk2$totss
```

```
## [1] 104.0274
```

```
shoppers.normalizedk2$betweenss/shoppers.normalizedk2$totss *100
```

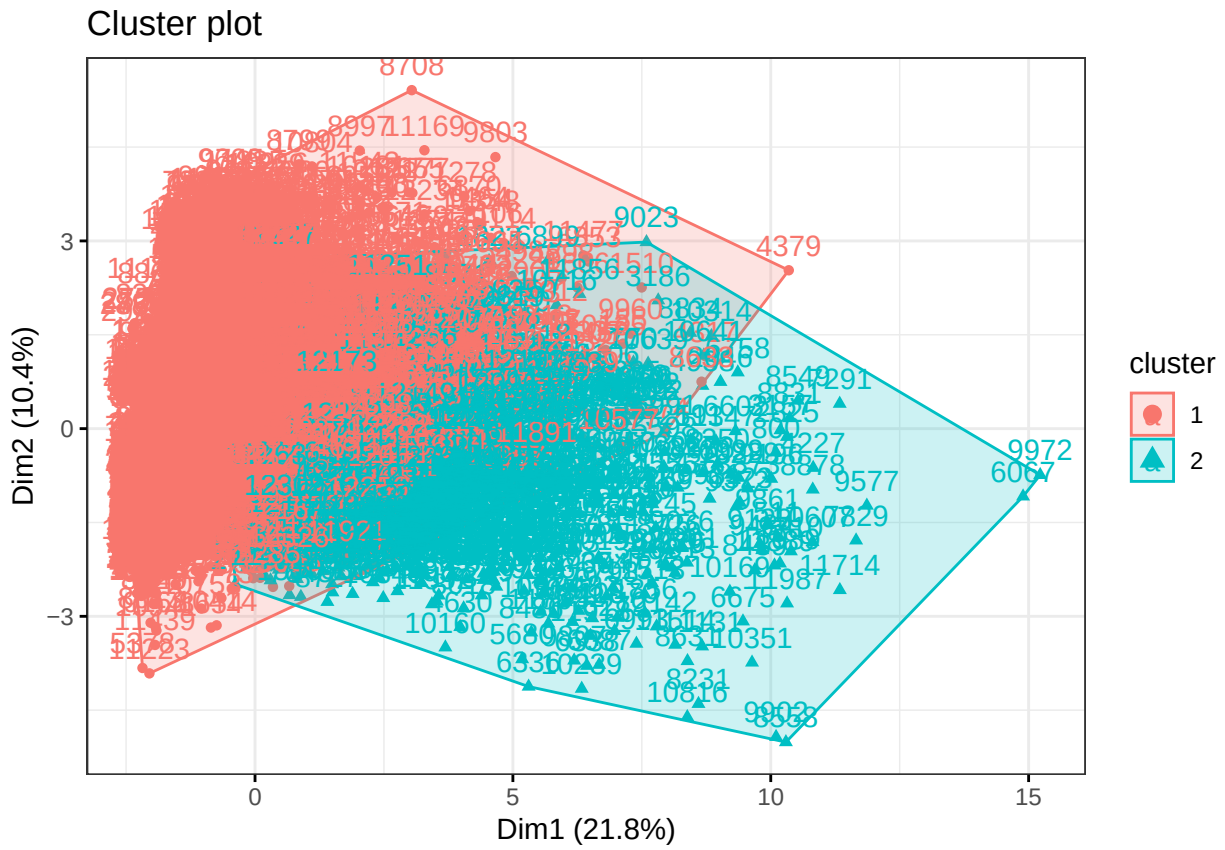
```
## [1] 61.44746
```

We got two clusters of size 11012 and 1119, it's quite imbalanced. The percentage ratio of between_SS and

total_SS is 61.4%. Usually the lower this value is the better. Let's visualize the two clusters and see what they look like.

6.1.4 Visualizing the two clusters

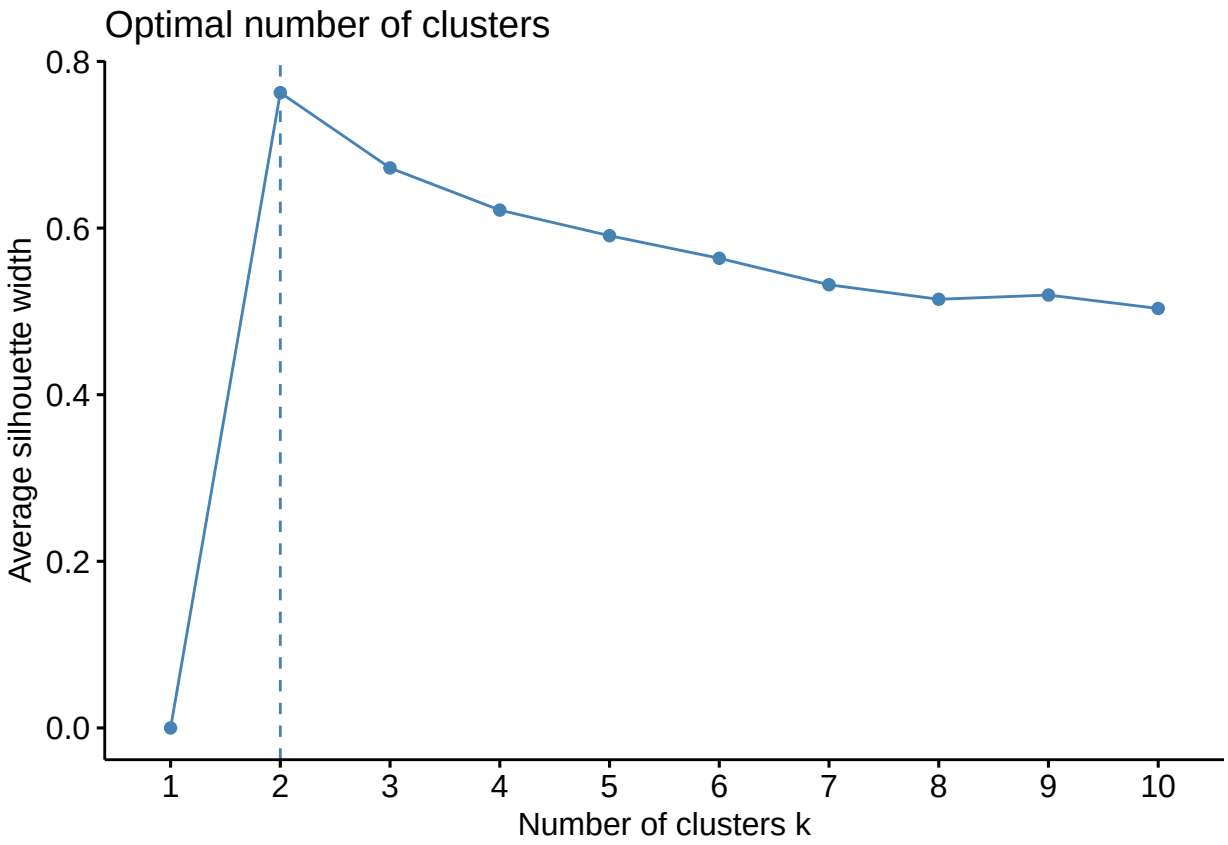
```
#visualization
fviz_cluster(shoppers.normalizedk2, data=shoppers.normalized, ggtheme = theme_bw())
```



There is a lot of overlap in our dataset which is not good. The two clusters have not been clearly distinguished. Let's perform optimization using silhouette to find out the optimal number of clusters to use.

6.1.5 Finding Optimal k using Silhoutte

```
fviz_nbclust(x=shoppers.normalized, FUNcluster =kmeans, method = 'silhouette') # two clusters are the o
```



Our initial guess was right. Two clusters is the optimal number of clusters needed for this problem according to silhouette plot above.

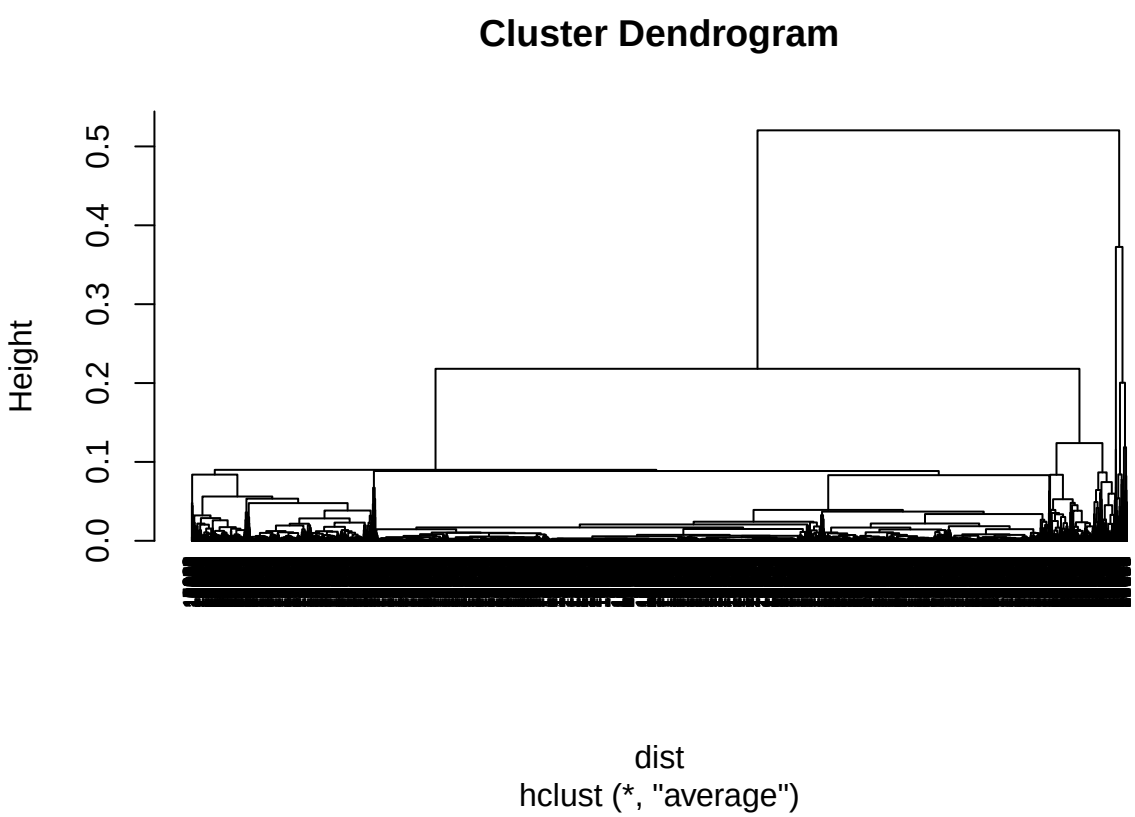
6.2 Hierachical Clustering

6.2.1 Euclidean Distance Calculation

```
#the data has already been scaled so we begin with calculating the euclidean distance between observati
shoppers.normalized.hierachical <- shoppers.normalized # creating a copy of the normalized data
dist <- dist(shoppers.normalized.hierachical, method = "euclidean")
```

6.2.2 Application of Hierachical clustering

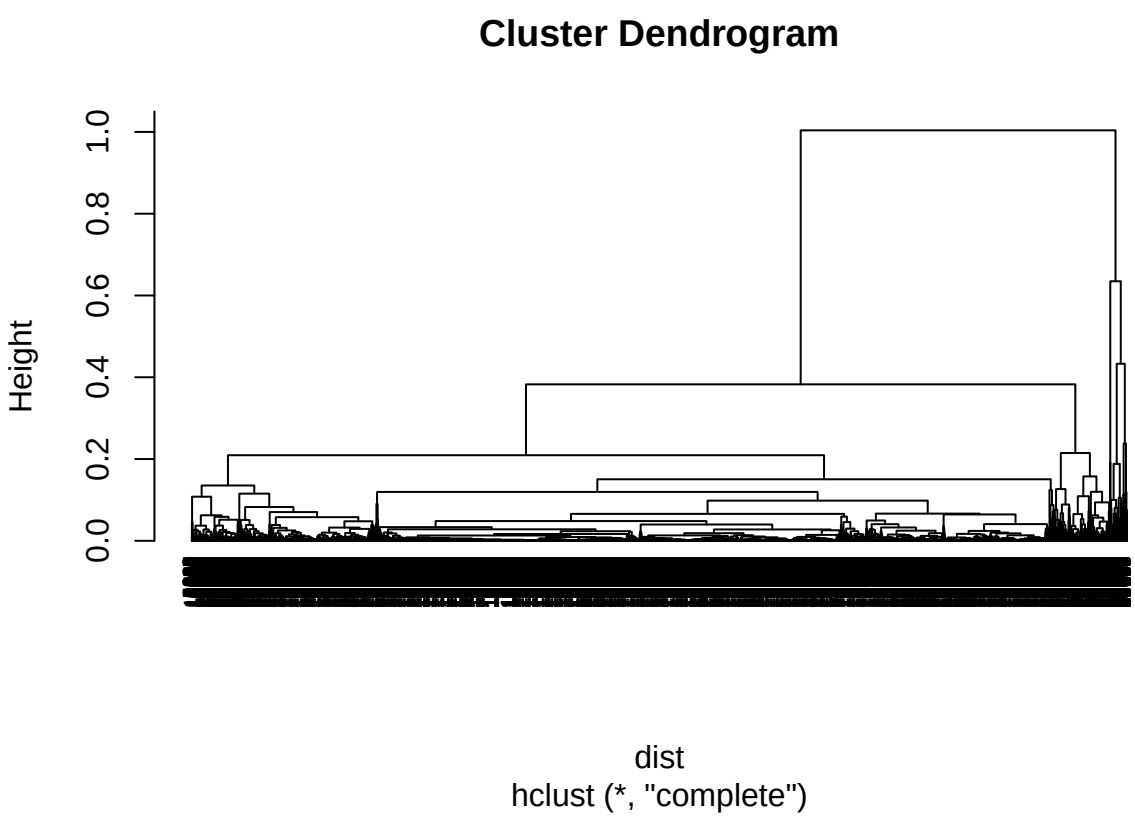
```
#applying hierachical clustering using average linkage method
shoppers.hierachical <- hclust(dist, method = "average")
plot(shoppers.hierachical, cex = 0.6, hang = -1) #plotting dendogram
```



We see a lot of overlaps, let's try with complete linkage method.

6.2.3 Complete linkage method

```
shoppers.hierachical.comp <- hclust(dist, method = "complete")
plot(shoppers.hierachical.comp, cex = 0.6, hang = -1) #plotting dendogram
```



We still observe a lot of overlaps. With this overlaps, we can't get good insights. Let's move on to challenging the solution to see if our results improve.

7. Challenging the Solution

Our modeling has not yielded good intuitive results. We think that it's because we did not select the appropriate attributes to use for the problem. To challenge the solution and hopefully get better results, we will select variables that we think are the most important in identifying the difference among groups within our data.

```
head(shoppers.new)
```

```
##   Administrative Administrative_Duration Informational Informational_Duration
## 1              0                       0             0                      0
## 2              0                       0             0                      0
## 4              0                       0             0                      0
## 5              0                       0             0                      0
## 6              0                       0             0                      0
## 9              0                       0             0                      0
##   ProductRelated ProductRelated_Duration BounceRates SpecialDay Month
## 1              1                0.000000  0.20000000          1     1
## 2              2               64.000000  0.00000000          1     1
## 4              2                2.666667  0.05000000          1     1
## 5             10              627.500000  0.02000000          1     1
## 6             19              154.216667  0.01578947          1     1
## 9              2               37.000000  0.00000000          5     1
##   OperatingSystems Browser Region TrafficType VisitorType Weekend
```

```
## 1                1       1       1           1           3       1
## 2                2       6       1          12           3       1
## 4                3       6       2          15           3       1
## 5                3       7       1          15           3       2
## 6                2       6       1          14           3       1
## 9                2       6       2          14           3       1
```

Selecting important variables. 1. Page types — Administrative, Informational (we will leave out the duration columns because they had too many outliers and we think that using the page types will be sufficient) 2. Time of the year — Month and weekend (we will leave out special day, because from our EDA most sales done were not during special day) 3. We think that Region is important. It is important because it will help the marketing team know what regions to focus on 4. Visitor Type is also important. 5. Operating system and Browser showed similar results so we don't need both, we can just pick one. We will go with Operating system.

The final variables selected are: Administrative, Informational, Month, Weekend, Region, Visitor Type, Operating systems.

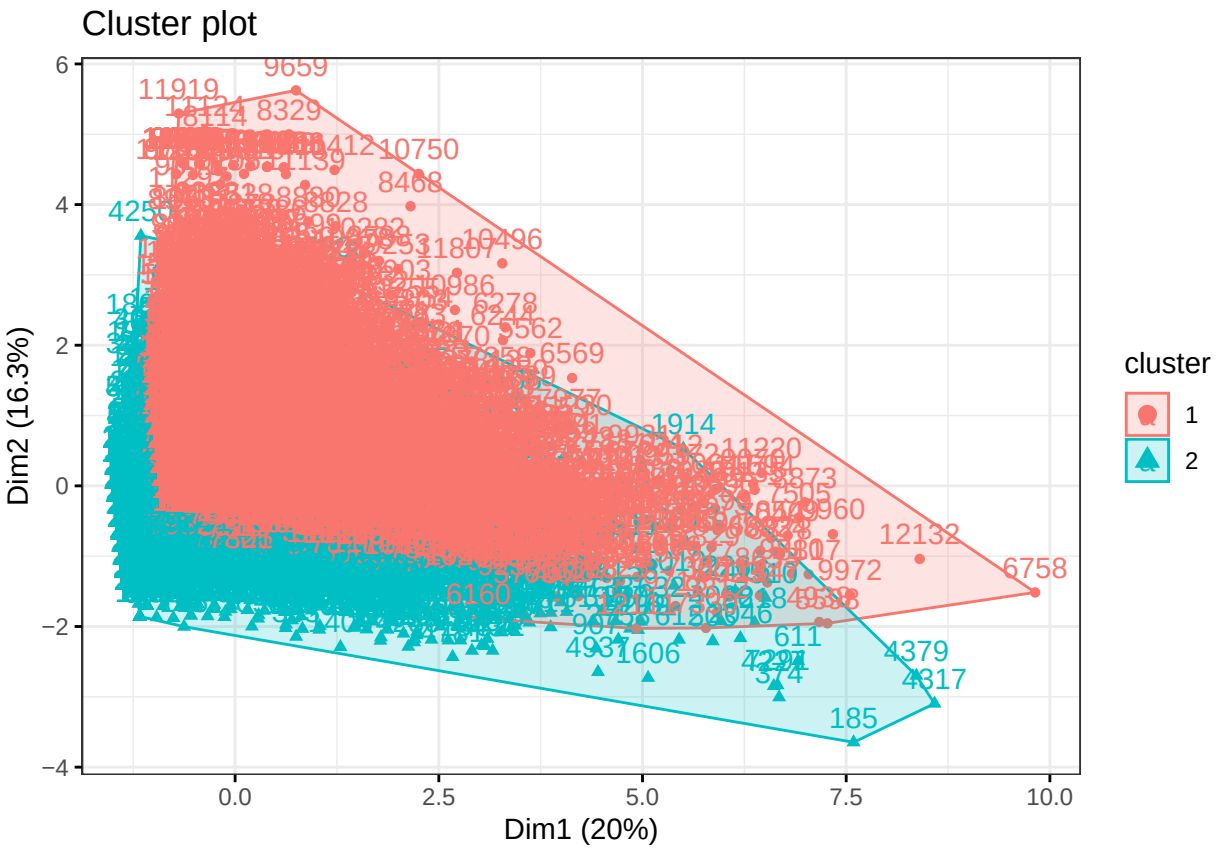
```
#creating a data frame with the selected columns
important.features <- select(shoppers.normalized, Administrative, Informational, Month, Weekend, Region
glimpse(important.features)
```

```
## Rows: 12,131
## Columns: 7
## $ Administrative   <dbl> 0.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000...
## $ Informational    <dbl> 0.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000...
## $ Month            <dbl> 5.503034e-05, 5.503034e-05, 5.503034e-05, 5.503034...
## $ Weekend          <dbl> 5.503034e-05, 5.503034e-05, 5.503034e-05, 1.100607...
## $ Region           <dbl> 5.503034e-05, 5.503034e-05, 1.100607e-04, 5.503034...
## $ VisitorType      <dbl> 0.000165091, 0.000165091, 0.000165091, 0.000165091...
## $ OperatingSystems <dbl> 5.503034e-05, 1.100607e-04, 1.650910e-04, 1.650910...
```

7.1 K-Means with selected features

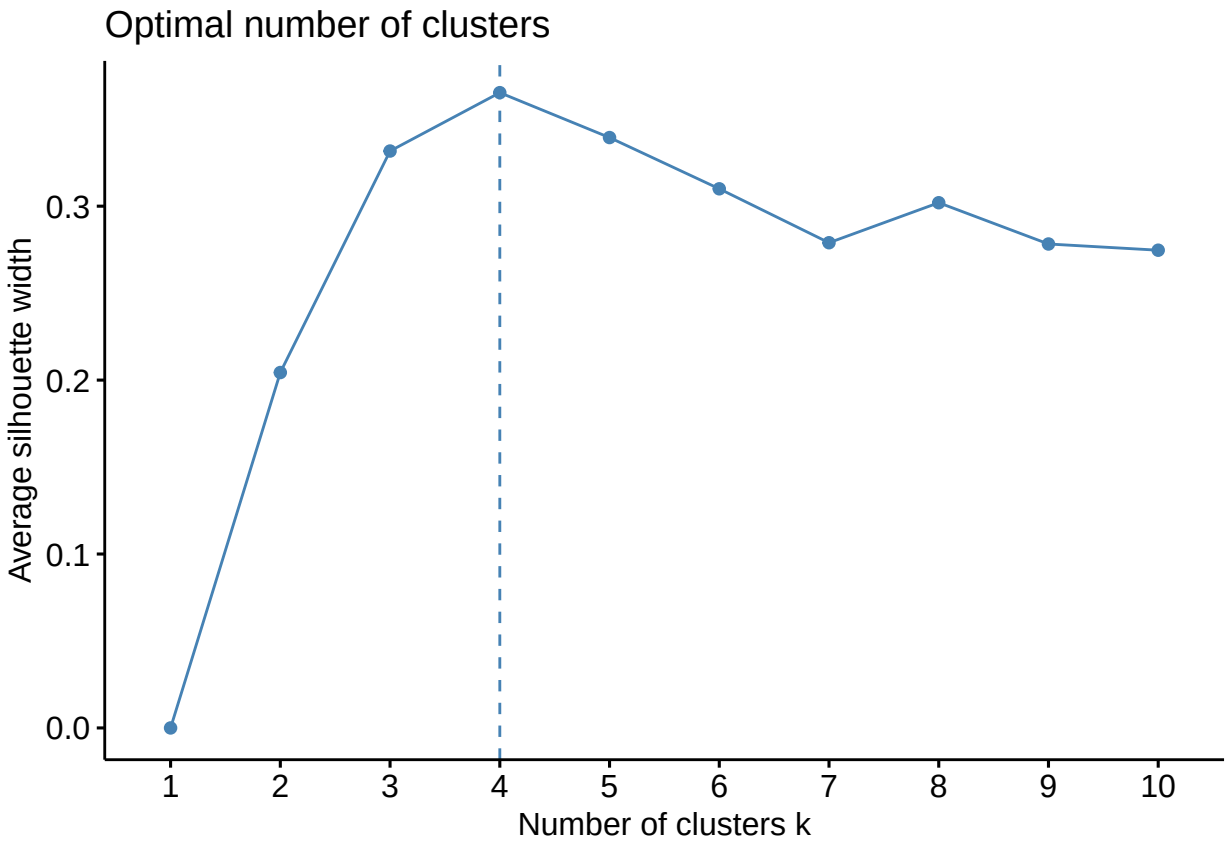
```
set.seed(123)
shoppers.importan.featk2 <- kmeans(important.features, 2, nstart=25)

#visualization
fviz_cluster(shoppers.importan.featk2, data=important.features, ggtheme = theme_bw())
```



We still see overlapping of the two clusters. Will optimization of clusters help? Let's find out below.

```
fviz_nbclust(x=important.features, FUNcluster =kmeans, method = 'silhouette')
```



From the graph, optimal number of clusters for this dataset is 3. let's rerun the algorithm and see what we find.

```
set.seed(123)
shoppers.importan.featk3 <- kmeans(important.features, 3, nstart=25)
shoppers.importan.featk3$size
```

```
## [1] 4935 5335 1861
```

```
shoppers.importan.featk3$betweenss
```

```
## [1] 0.0005662411
```

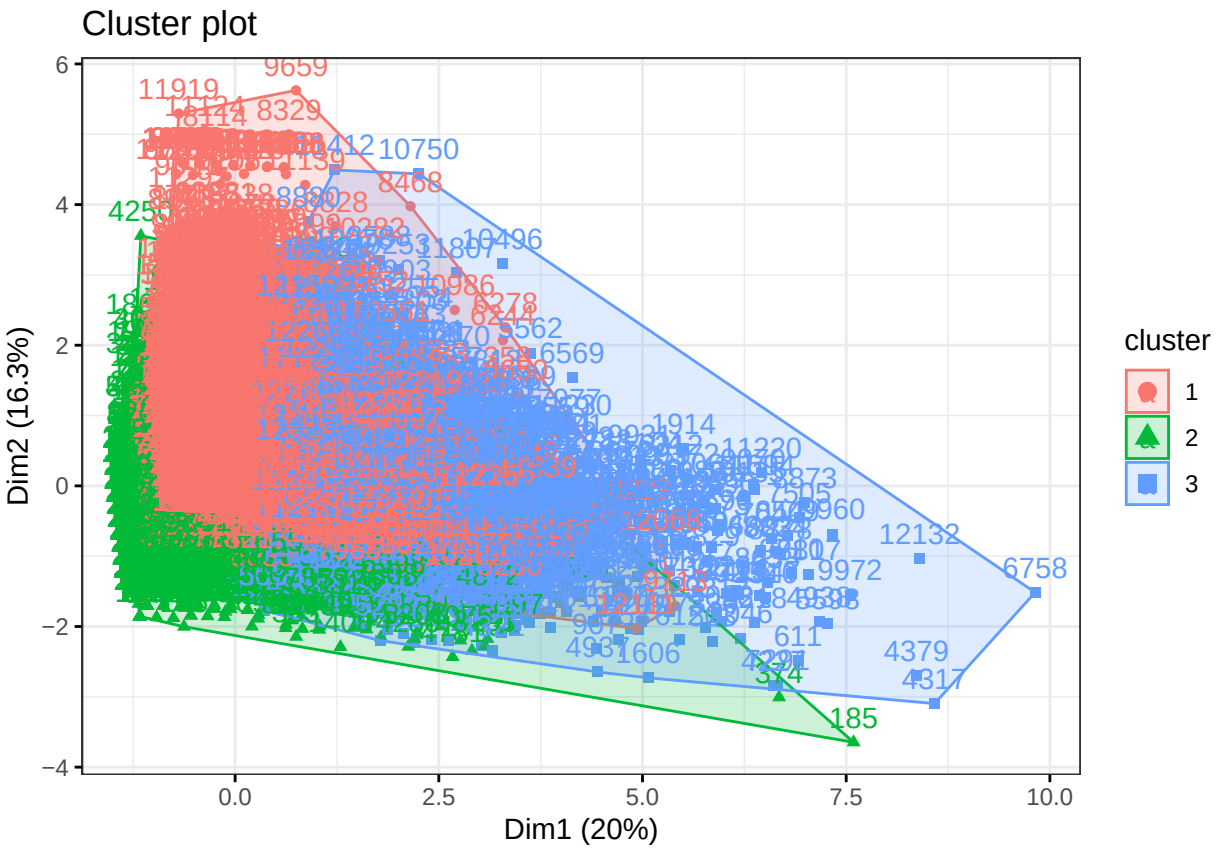
```
shoppers.importan.featk3$totss
```

```
## [1] 0.001088947
```

```
shoppers.importan.featk3$betweenss/shoppers.normalizedk2$totss *100
```

```
## [1] 0.0005443193
```

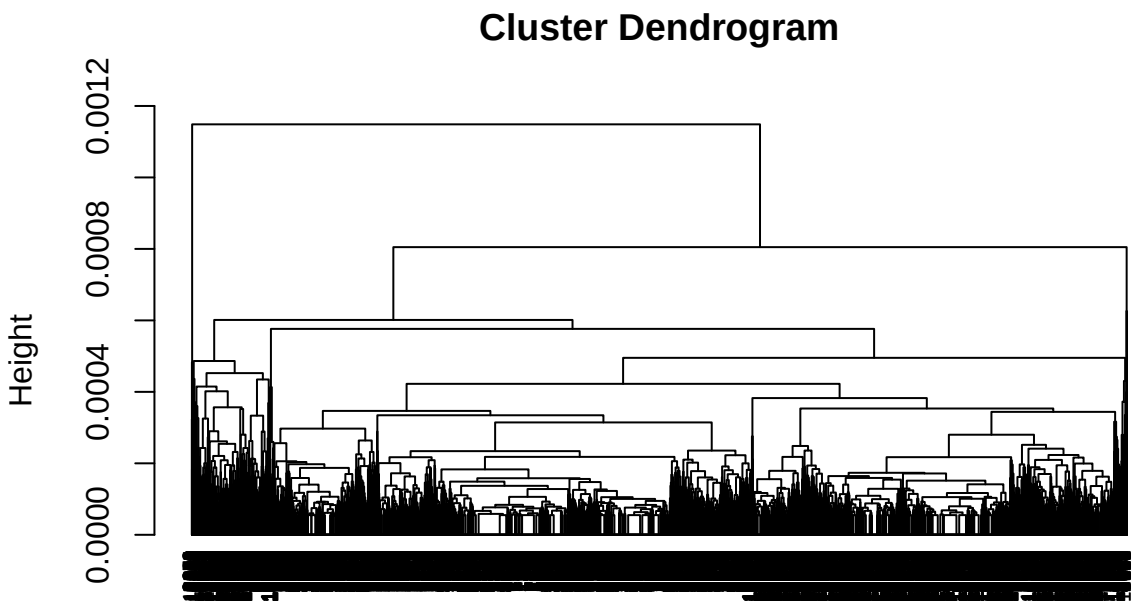
```
#visualization
fviz_cluster(shoppers.importan.featk3, data=important.features, ggtheme = theme_bw())
```



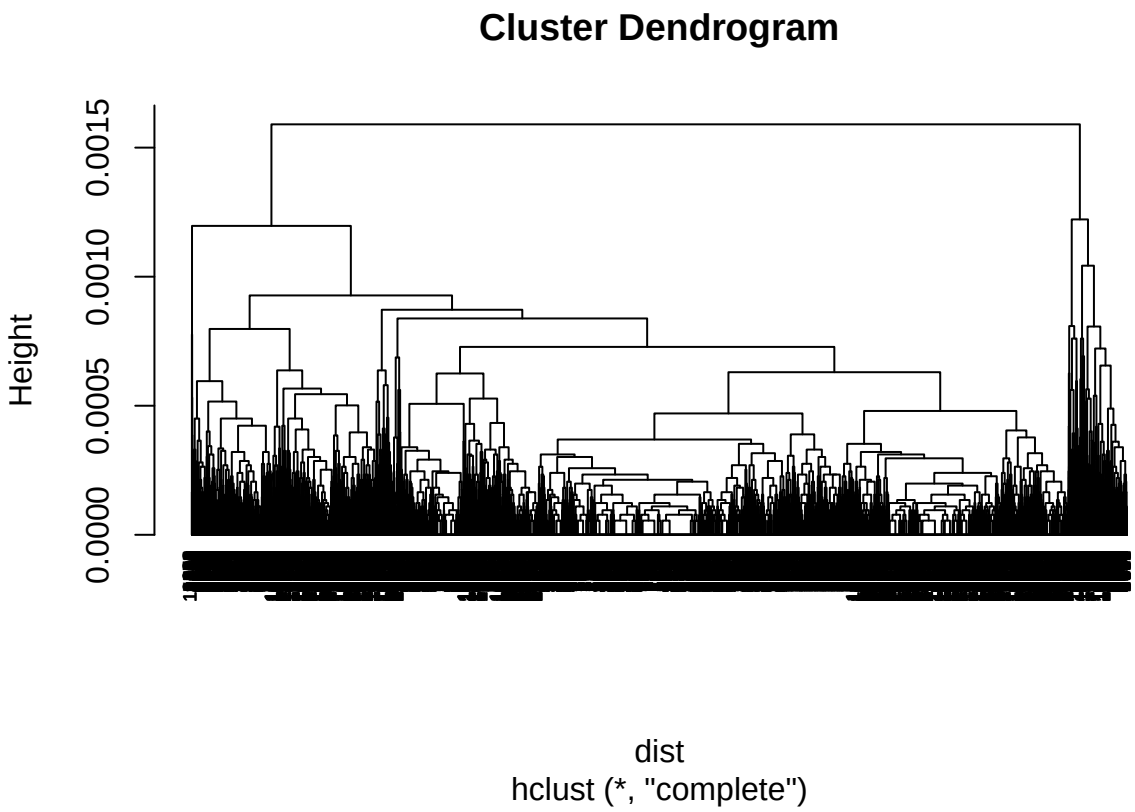
We still see overlapping clusters. But the ratio between between_ss and total_SS is almost zero. Does that suggest that we had a perfect clustering? Maybe.

7.2 Hierachical Clustering with selected features

```
features.important <- important.features # creating a copy of the normalized dataframe of selected feat
dist <- dist(features.important, method = "euclidean")
#applying hierachical clustering using average linkage method
hierachical.important.features <- hclust(dist, method = "average")
plot(hierachical.important.features, cex = 0.6, hang = -1) #plotting dendogram
```




```
dist <- dist(features.important, method = "euclidean")
#applying hierachical clustering using completelinkage method
hierachical.important.features.complete <- hclust(dist, method = "complete")
plot(hierachical.important.features.complete, cex = 0.6, hang = -1) #plotting dendogram
```



8. Conclusion and Recommendations

There is still a lot of overlaps seen by the dendrograms making it impossible to derive any insights. The overlaps seen is not surprising. From research done, we've learned that Hierachical clustering works best with less data (150 observations or less). To improve our models, next time, we should consider sampling the data before modeling. For conclusions and recommendations, we will utilize our EDA analysis.

1. Customers who visited Administrative and Informational pages also visited Kira Plastina, where some of the customers purchased Kira Plastina's products.
2. Special days did not result in increased traffic in Kira Plastina's site. From the Bivariate Analysis we saw that most visits to Kira happened before any special day and some of these visits led to purchases at Kira. Therefore, these special days could be a target for brand and marketing teams to improve on their sales.
3. November with the 2nd largest number of visits to the site had the biggest revenue collection compared to the rest of the months. November was followed by May, and Dec in revenue collection. We recommend that the brand and sales team find out why these months bring in more revenue and leverage the information to other months.
4. Region 1 and 3 contributed the most revenue to Kira Plastina. Why is that? The brand and sales team needs to find out why and see how they can leverage what is working in these regions to other regions.
5. Kira Plastina had the most returning visitors which translated to more revenue for them. Return customers ensure continuous cash flow, the brand and marketing team should conduct research to find ways of improving return customer experience and work on more marketing strategies to increase inflow of new customers.

6. Comparing the proportional revenue collected during the week and the weekends, it was about the same with weekdays bringing in slightly more. The brand and marketing team can look into weekdays and figure out what improvements to make. These improvements could translate into more revenue, since, weekdays had the most visits to the site.