Foundation of Financial Data Science (FE 582)

(Homework 1)

Prof. Dragos Bozdog

Student Name: **Paras Garg**

Course Section: **FE 582 A**

Problem 1 –

Explore realdirect.com thinking about how buyers and sellers would navigate, and how the website is organized. Use the datasets provided for Bronx, Brooklyn, Manhattan, Queens, and Staten Island. Do the following:

- Load in and clean the data
- Conduct exploratory data analysis in order to find out where there are outliers or missing values, decide how you will treat them, make sure the dates are formatted correctly, make sure values you think are numerical are being treated as such, etc.
- Conduct exploratory data analysis to visualize and make comparisons for residential building category classes across boroughs and across time (1, 2, 3 family homes, coops, and condos). Use histograms, box plots, scatter plots, or other visual graphs. Provide summary statistics along with your conclusions.

Analysis 1 –

```r
### CLEANING ENVIRONMENT AND SETTING WORK DIRECTORY
rm(list=ls())
setwd("C:/Users/Paras Garg/Documents/R/FE Assignments/ ")
### INCLUING USEFUL PACKAGES
library('gdata')
library('ggplot2')
library('plyr')
library("doBy")

### LOADING DATASETS
bronx <- read.xls("rollingsales_bronx.xls", perl = "C:\\Perl64\\bin\\perl.exe", pattern="BOROUGH")
brooklyn <- read.xls("rollingsales_brooklyn.xls", perl = "C:\\Perl64\\bin\\perl.exe", pattern="BOROUGH")
manhattan <- read.xls("rollingsales_manhattan.xls", perl = "C:\\Perl64\\bin\\perl.exe", pattern="BOROUGH")
queens <- read.xls("rollingsales_queens.xls", perl = "C:\\Perl64\\bin\\perl.exe", pattern="BOROUGH")
staten <- read.xls("rollingsales_statenisland.xls", perl = "C:\\Perl64\\bin\\perl.exe", pattern="BOROUGH"
```

```r
### DATA FORMATTING FUNCTION
format_data <- function (df) {
  df$GROSS.SQUARE.FEET.N <- as.numeric((gsub("[^[:digit:]]","", df$GROSS.SQUARE.FEET)))
  df$LAND.SQUARE.FEET.N <- as.numeric((gsub("[^[:digit:]]","", df$LAND.SQUARE.FEET)))
  df$SALE.PRICE.N <- as.numeric(gsub("[^[:digit:]]","",df$SALE.PRICE))
  df$SALE.DATE <- as.Date(df$SALE.DATE)
  df$YEAR.BUILT <- as.numeric(as.character(df$YEAR.BUILT))
  return (df)
}
```

```r
### DATA CLEANING FUNCTION
clean_data <- function (df) {
  #Removing NA values
  df <- df[!is.na(df$GROSS.SQUARE.FEET.N), ]
  df <- df[!is.na(df$LAND.SQUARE.FEET.N), ]
  df <- df[!is.na(df$SALE.PRICE.N), ]
  #Removing outliners
  df$SALE.PRICE.LOG <- log(df$SALE.PRICE.N)
  df <- df[df$SALE.PRICE.LOG > 5, ]
  #Categorize sales buildings
  family_category <- grepl("FAMILY", df$BUILDING.CLASS.CATEGORY) * 1
  condos_category <- grepl("CONDOS", df$BUILDING.CLASS.CATEGORY) * 2
  coops_category <- grepl("COOPS", df$BUILDING.CLASS.CATEGORY) * 3

  category <- as.character(family_category + condos_category + coops_category)
  category[category == "1"] <- "FAMILY"
  category[category == "2"] <- "CONDOS"
  category[category == "3"] <- "COOPS"
  category[category == "0"] <- "OTHERS"

  df$BUILDING.CLASS.CATEGORY.N <- factor(category)
  return (df)
}
```

```
### DATA FRAMES FORMATTING AND CLEANING
bronxDf <- clean_data(format_data(bronx))
brooklynDf <- clean_data(format_data(brooklyn))
manhattanDf <- clean_data(format_data(manhattan))
queensDf <- clean_data(format_data(queens))
statenDf <- clean_data(format_data(staten))
```

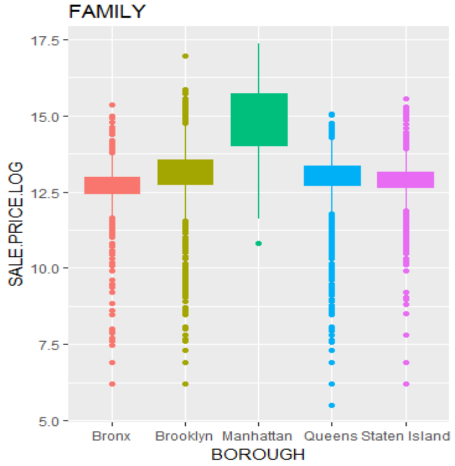
Analysis across boroughs

#Functions

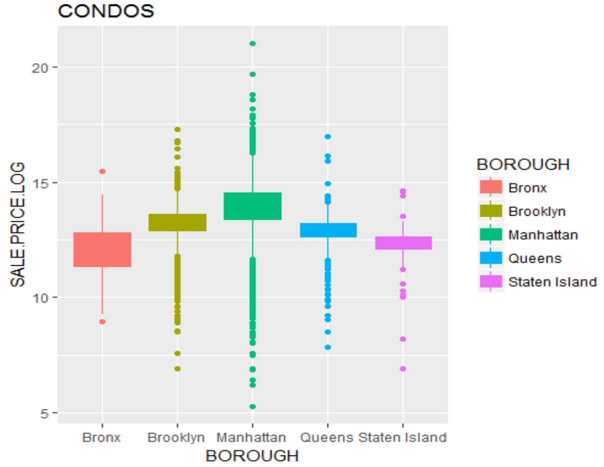
```
  # FUNCTION: Borough v/s Sale Price for particular Building Class
  borough_sp <- function (class) {
      outliers <- bronxDf$SALE.PRICE.LOG[bronxDf$BUILDING.CLASS.CATEGORY.N == class]
      borough_1 <- data.frame(BOROUGH = rep("Bronx", length(outliers)), SALE.PRICE.LOG = outliers)
      outliers <- brooklynDf$SALE.PRICE.LOG[brooklynDf$BUILDING.CLASS.CATEGORY.N == class]
      borough_2 <- data.frame(BOROUGH = rep("Brooklyn", length(outliers)), SALE.PRICE.LOG = outliers)
      outliers <- manhattanDf$SALE.PRICE.LOG[manhattanDf$BUILDING.CLASS.CATEGORY.N == class]
      borough_3 <- data.frame(BOROUGH = rep("Manhattan", length(outliers)), SALE.PRICE.LOG = outliers)
      outliers <- queensDf$SALE.PRICE.LOG[queensDf$BUILDING.CLASS.CATEGORY.N == class]
      borough_4 <- data.frame(BOROUGH = rep("Queens", length(outliers)), SALE.PRICE.LOG = outliers)
      outliers <- statenDf$SALE.PRICE.LOG[statenDf$BUILDING.CLASS.CATEGORY.N == class]
      borough_5 <- data.frame(BOROUGH = rep("Staten Island", length(outliers)), SALE.PRICE.LOG = outliers)

      finalDf <- rbind(borough_1, borough_2, borough_3, borough_4, borough_5)
      ggplot(finalDf, aes(x=BOROUGH, y=SALE.PRICE.LOG, fill=BOROUGH, colour=BOROUGH, group=BOROUGH)) +
        geom_boxplot() + ggtitle(class)
  }

  # FUNCTION: Building Class v/s Sale Price for particular Borough
  building_sp <- function (df, borough) {
    BUILDING.CLASS <- df$BUILDING.CLASS.CATEGORY.N
    SALE.PRICE.LOG <- df$SALE.PRICE.LOG

    ggplot(df, aes(x=BUILDING.CLASS, y=SALE.PRICE.LOG, fill=BUILDING.CLASS,
               colour=BUILDING.CLASS, group=BUILDING.CLASS)) + geom_boxplot() + ggtitle(borough)
  }

  # FUNCTION: Sale Price v/s Gross Square feet for particular Borough
  gross_sp <- function (df, borough) {
    SALE.PRICE.LOG <- df$SALE.PRICE.LOG
    GROSS.SQFT.LOG <- log(df$GROSS.SQUARE.FEET.N)
    BUILDING.CLASS <- df$BUILDING.CLASS.CATEGORY.N

    ggplot(df, aes(x=SALE.PRICE.LOG, y=GROSS.SQFT.LOG, fill=BUILDING.CLASS, colour=BUILDING.CLASS)) +
      geom_point() + ggtitle(borough)
  }

  # FUNCTION: Sale Price v/s Frequency for particular Borough
  sp_freq <- function (df, borough) {
    par(mfrow=c(2,2))
    hist(log(df[df$BUILDING.CLASS.CATEGORY.N=="FAMILY",]$SALE.PRICE.N), col="#8daeb4",
         main="Family Homes", xlab= "SALE.PRICE.LOG", cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
    hist(log(df[df$BUILDING.CLASS.CATEGORY.N=="CONDOS",]$SALE.PRICE.N), col="#0d447a",
         main="CONDOS", xlab= "SALE.PRICE.LOG", cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
    hist(log(df[df$BUILDING.CLASS.CATEGORY.N=="COOPS",]$SALE.PRICE.N), col="#ffbe4c",
         main="COOPS", xlab= "SALE.PRICE.LOG", cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
    hist(log(df[df$BUILDING.CLASS.CATEGORY.N=="OTHERS",]$SALE.PRICE.N), col="#fddf5f",
         main="Others", xlab= "SALE.PRICE.LOG", cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
    title(main = borough, outer = TRUE, cex.main=1.0, line=-1)
}
```

```
#Plots
# PLOT: Borough v/s Sale Price for particular Building Class
```

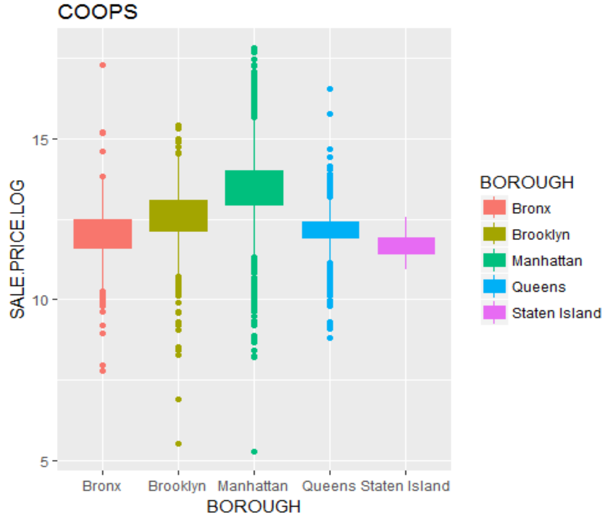
```
borough_sp("FAMILY")
```



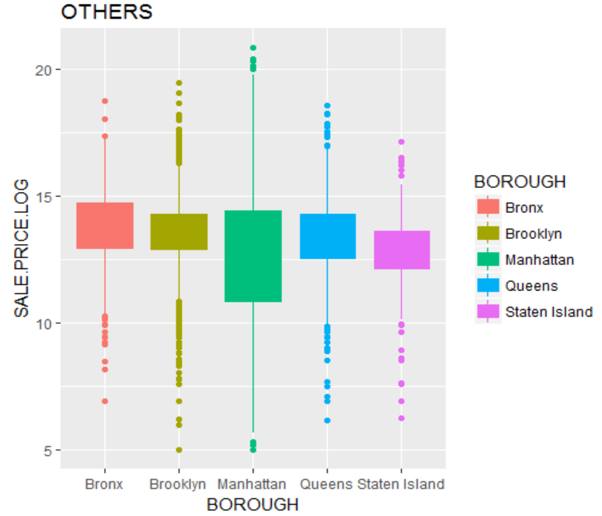
```
borough_sp("CONDOS")
```



```
borough_sp("COOPS")
```

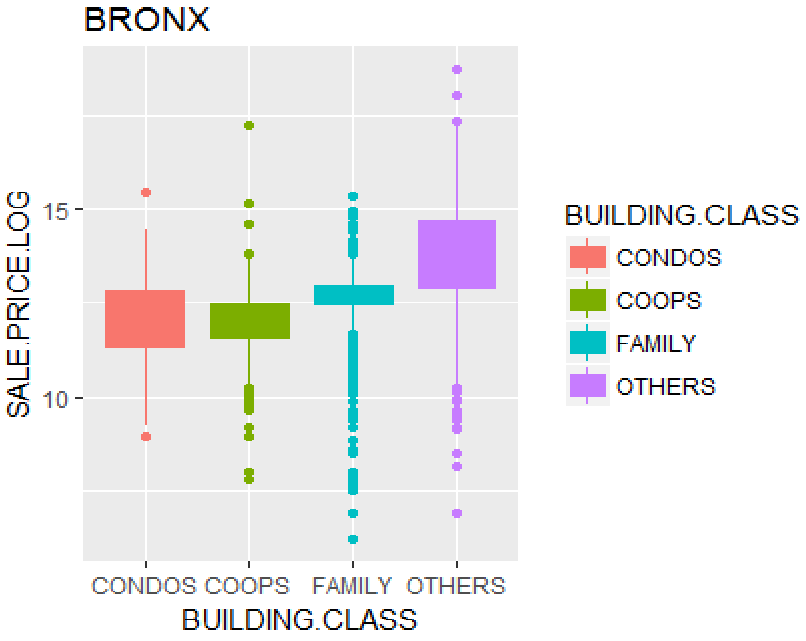


```
borough_sp("OTHERS")
```

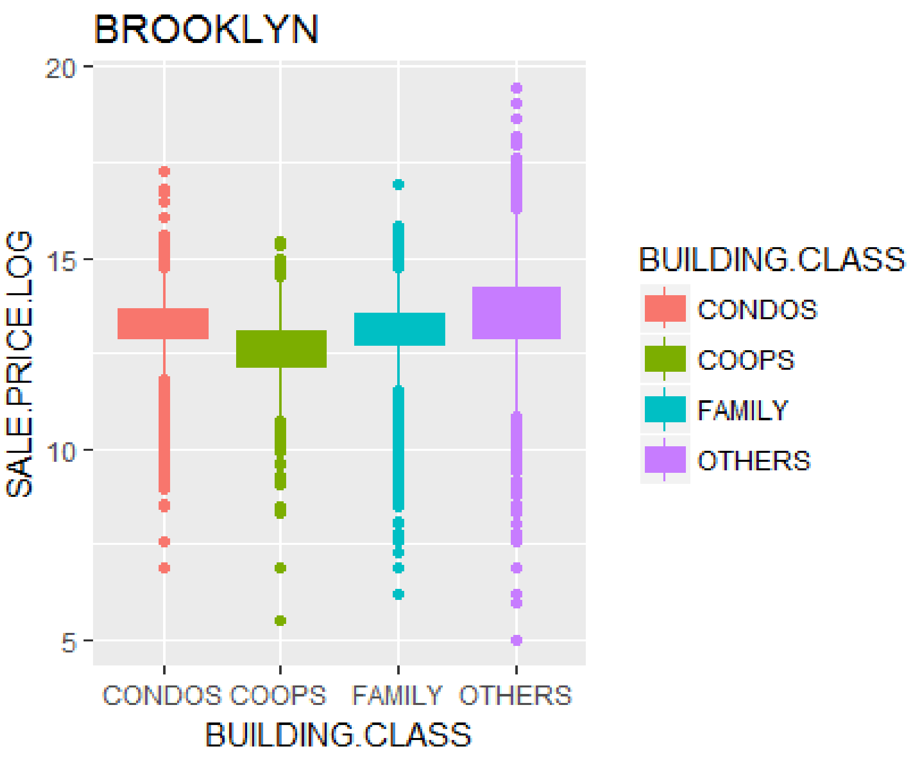


```
#PLOT: Building Class v/s Sale Price for particular Borough
```

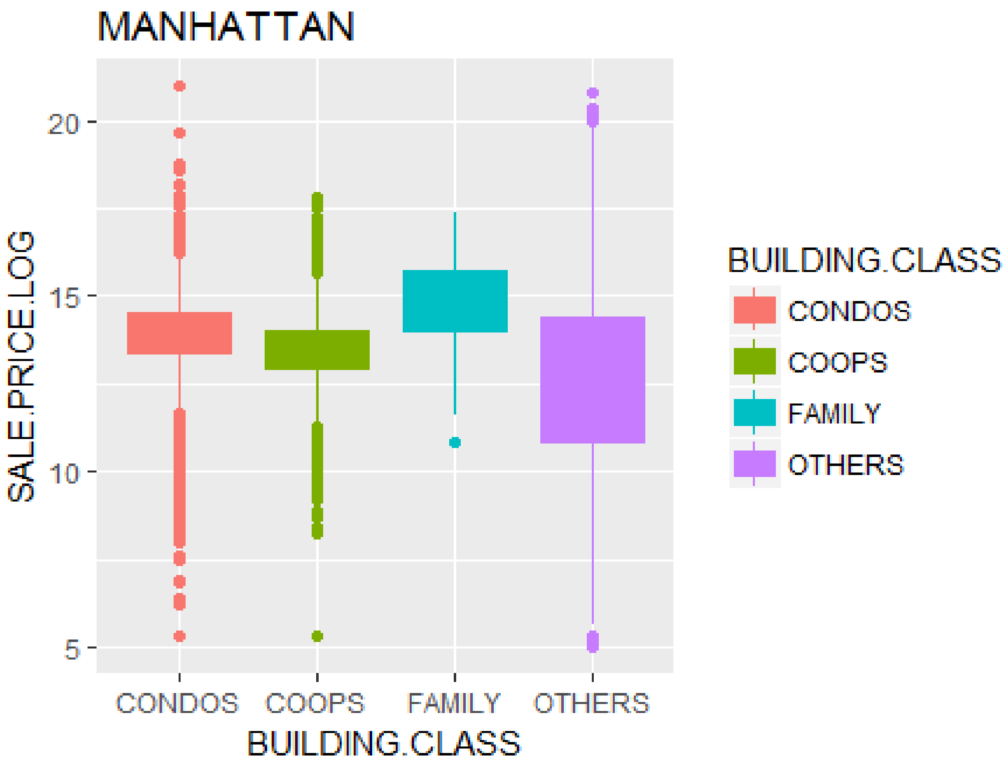
```
building_sp(bronxDf, "BRONX")
```



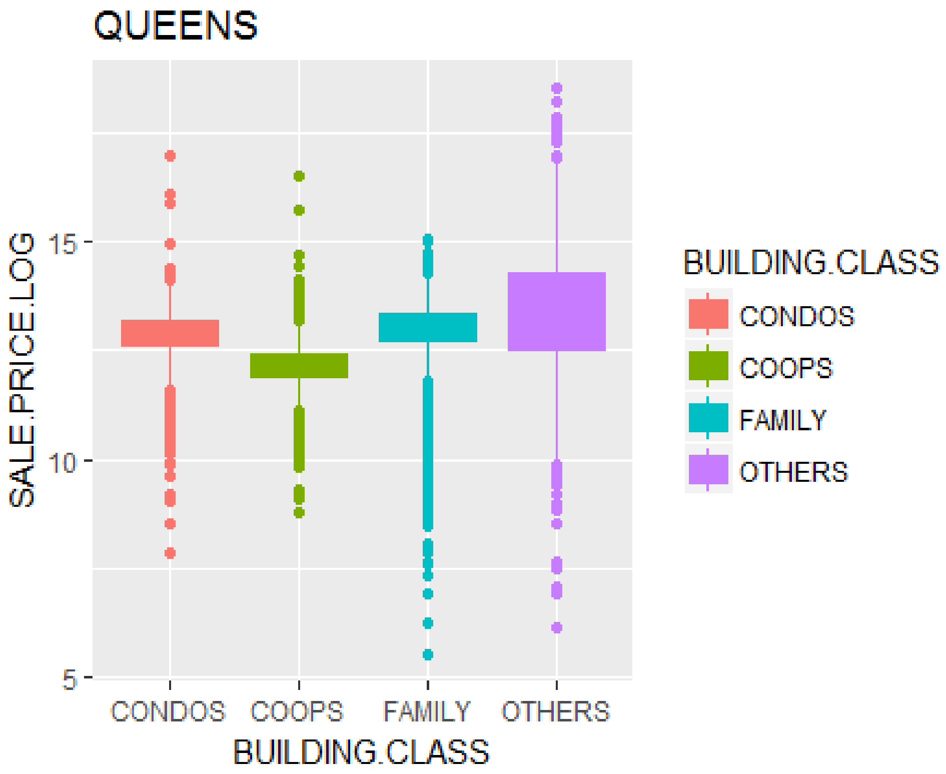
```
building_sp(brooklynDf, "BROOKLYN")
```



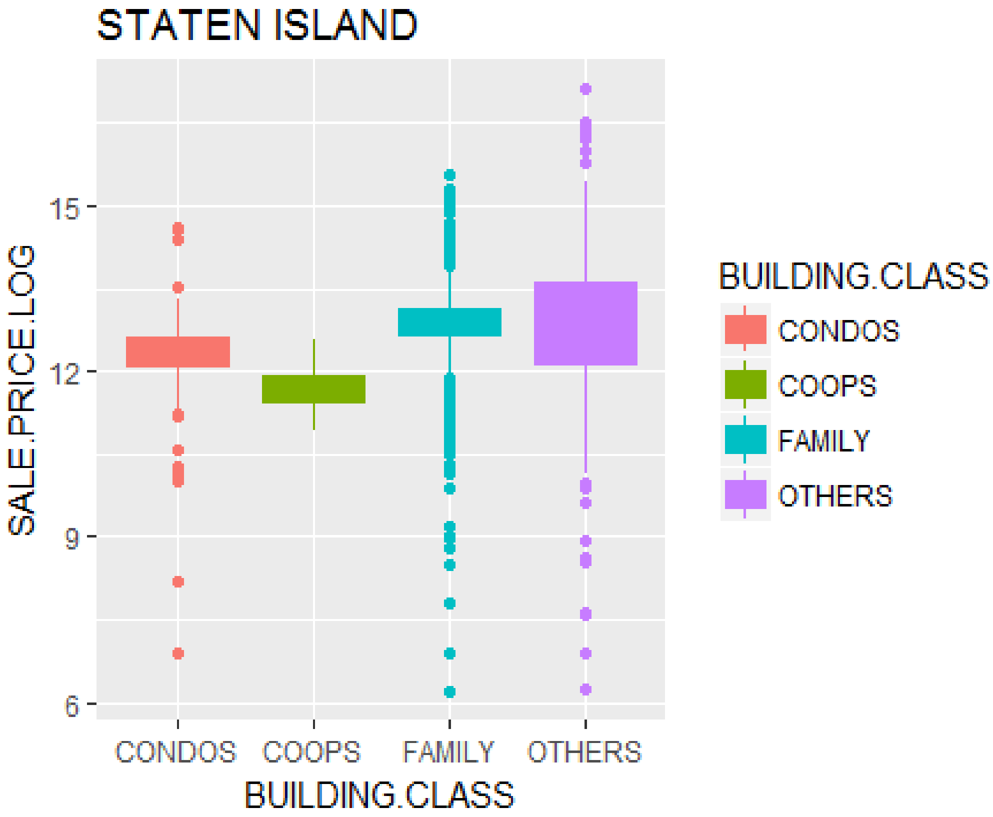
```
building_sp(manhattanDf, "MANHATTAN")
```



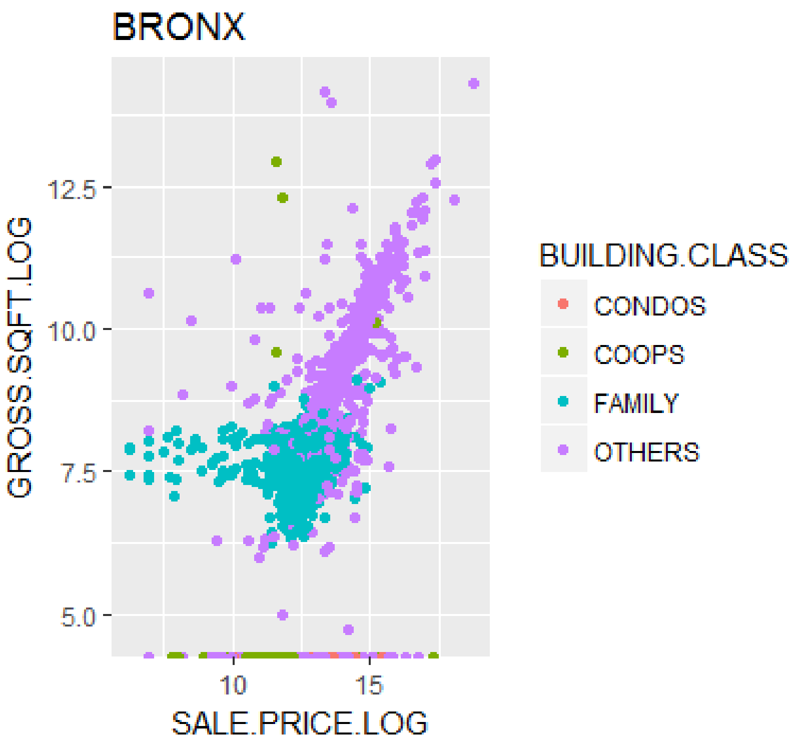
```
building_sp(queensDf, "QUEENS")
```



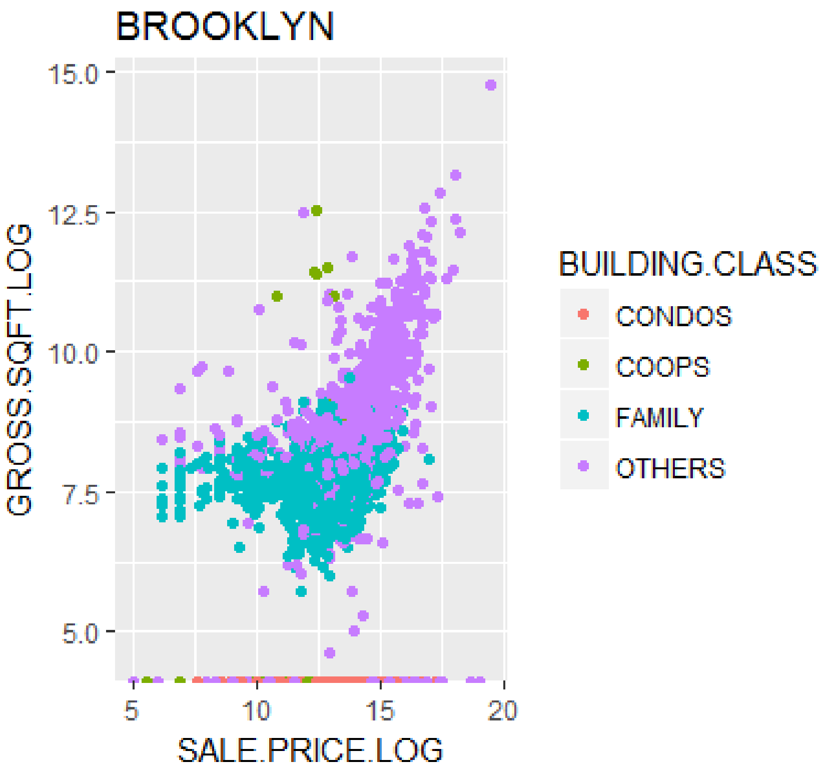
```
building_sp(statenDf, "STATEN ISLAND")
```



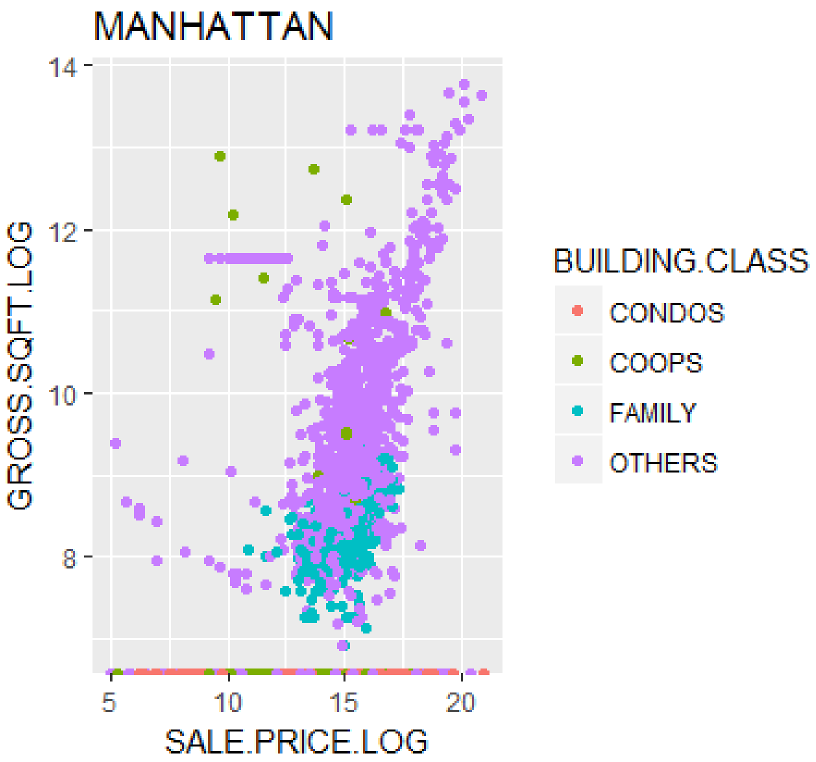
```
# PLOT: Sale Price v/s Gross Square feet for particular Borough
gross_sp(bronxDf, "BRONX")
```



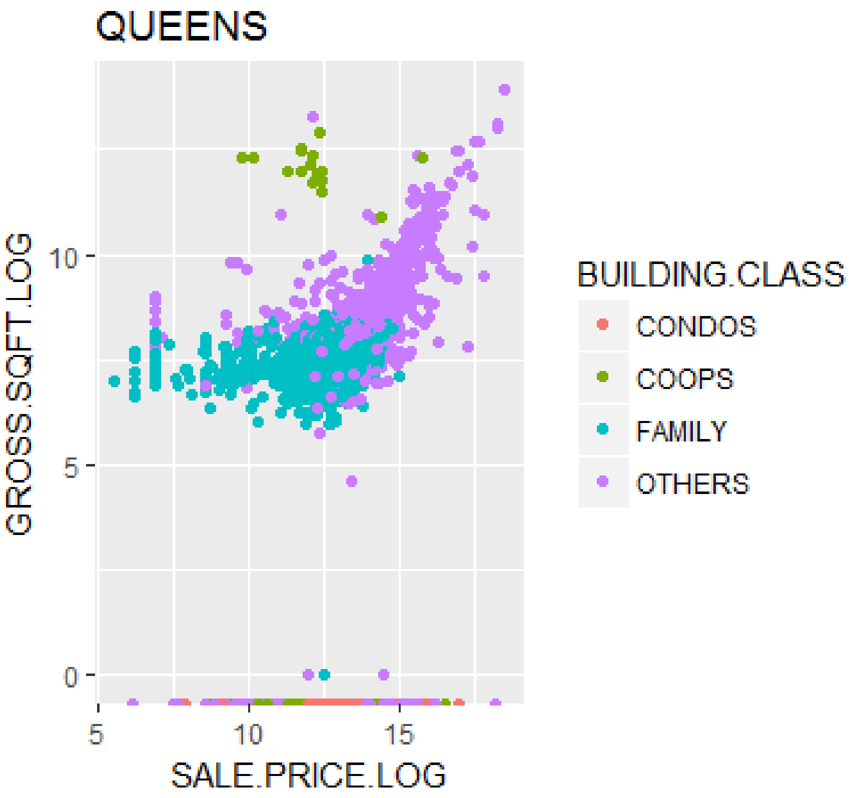
```
gross_sp(brooklynDf, "BROOKLYN")
```



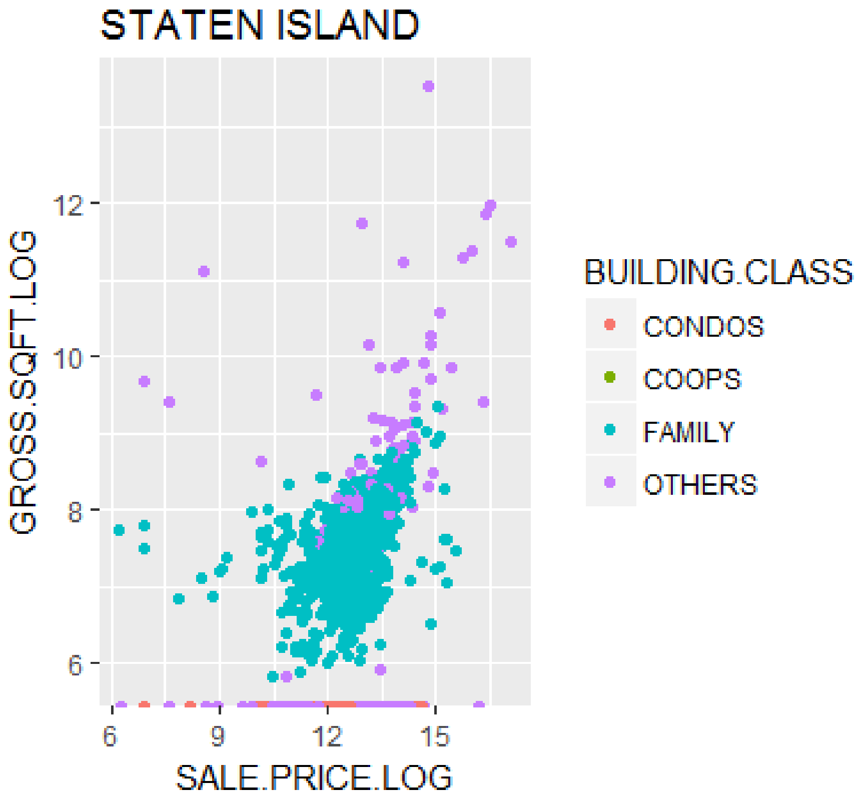
```
gross_sp(manhattanDf, "MANHATTAN")
```



```
gross_sp(queensDf, "QUEENS")
```

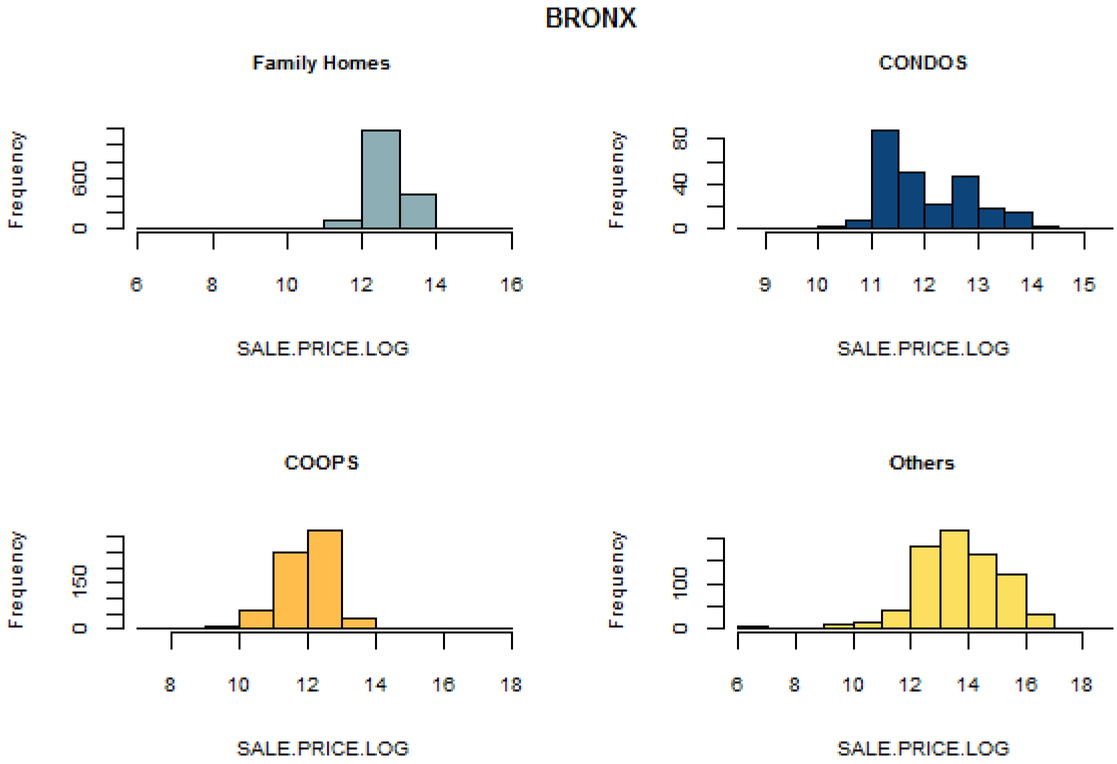


```
gross_sp(statenDf, "STATEN ISLAND")
```

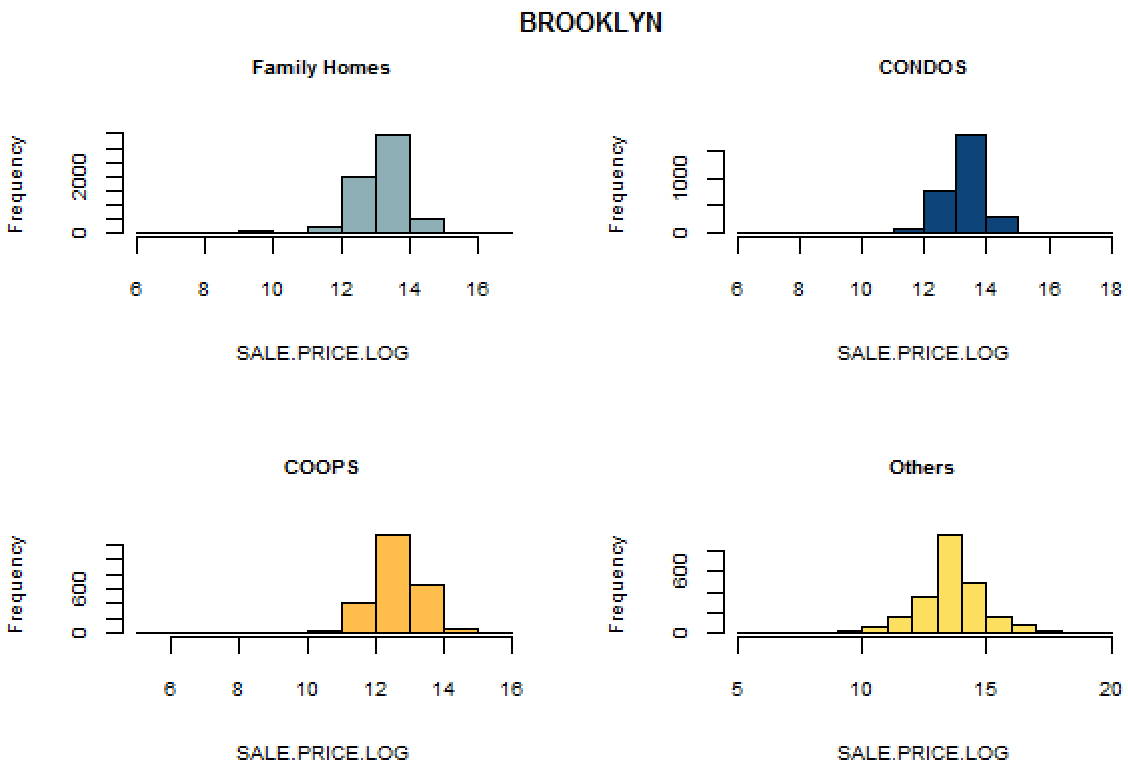


```
# PLOT: Sale Price v/s Frequency for particular Borough
```

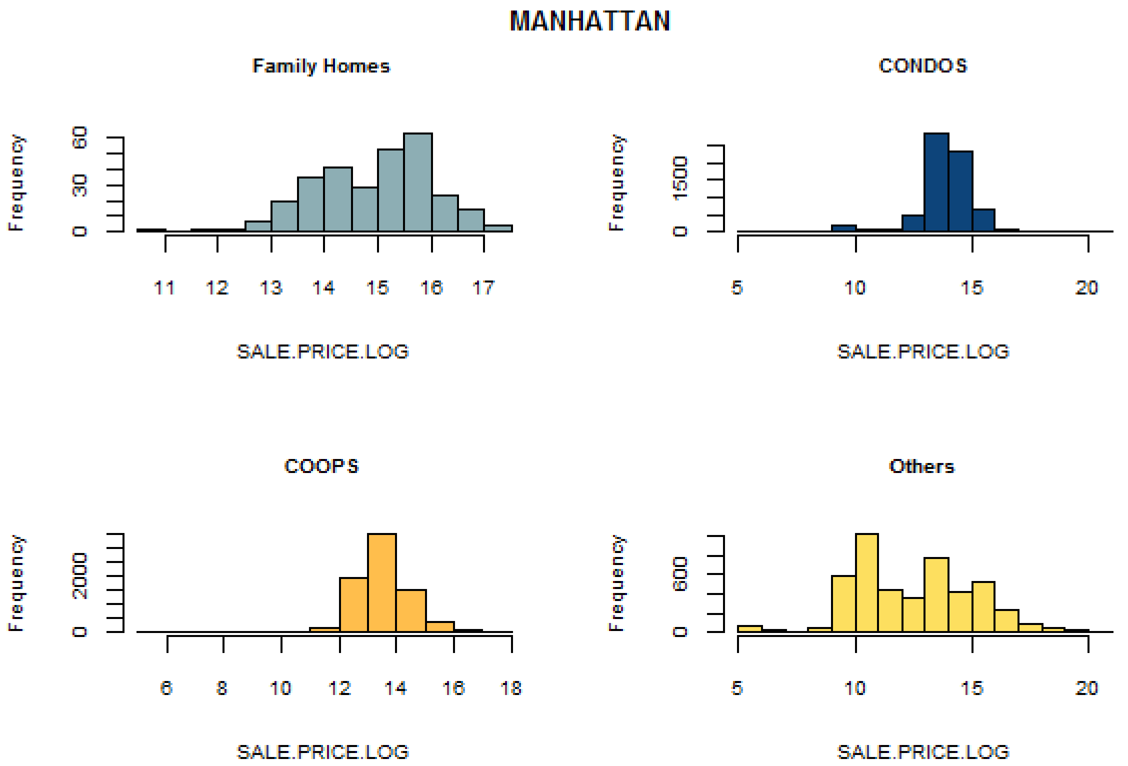
```
sp_freq(bronxDf, "BRONX")
```



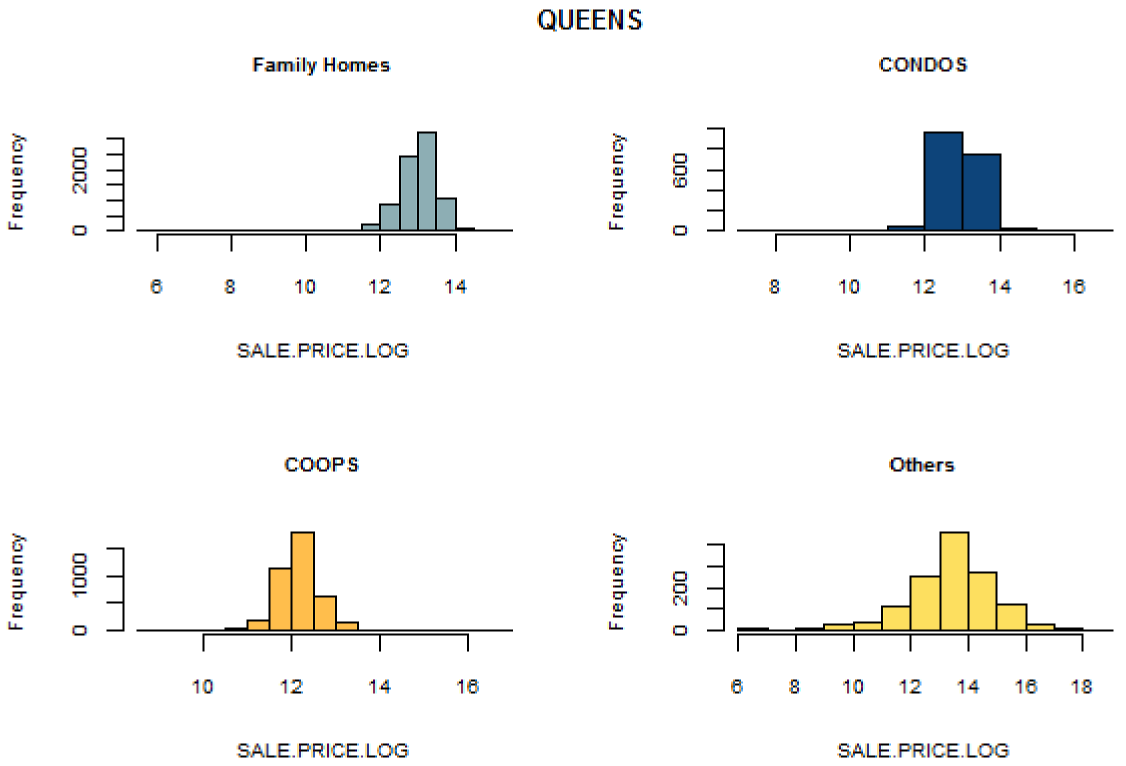
```r
sp_freq(brooklynDf, "BROOKLYN")
```



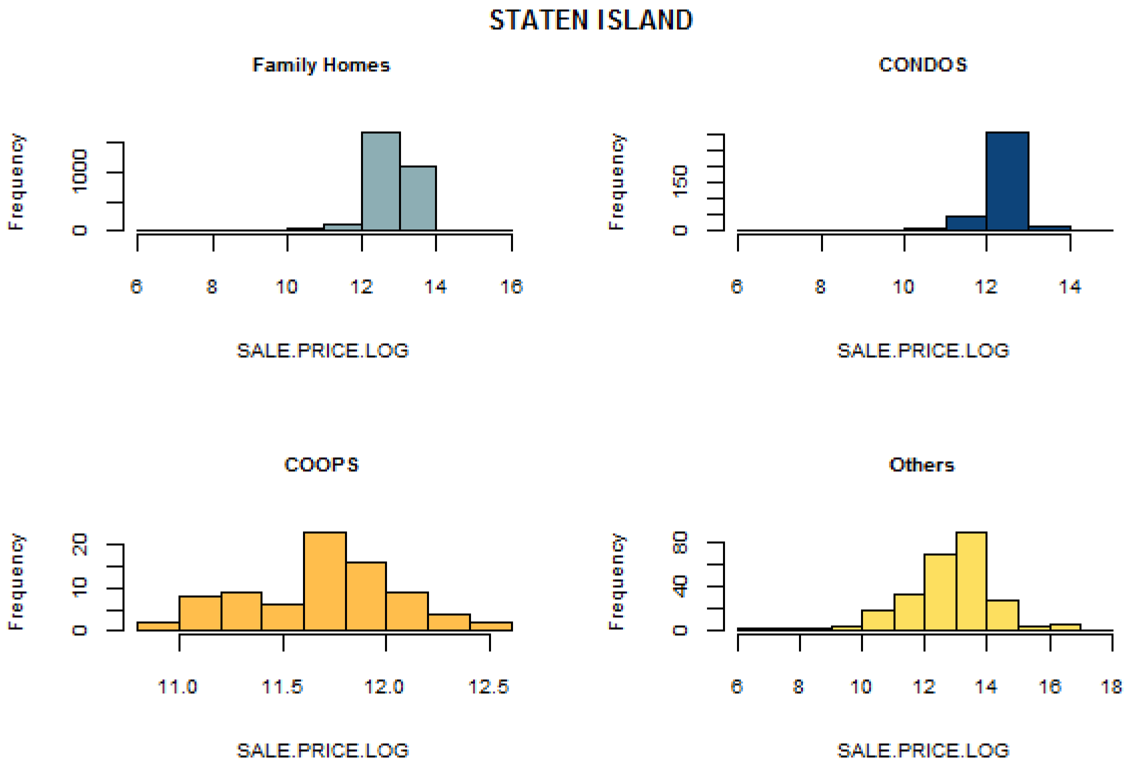
```r
sp_freq(manhattanDf, "MANHATTAN")
```

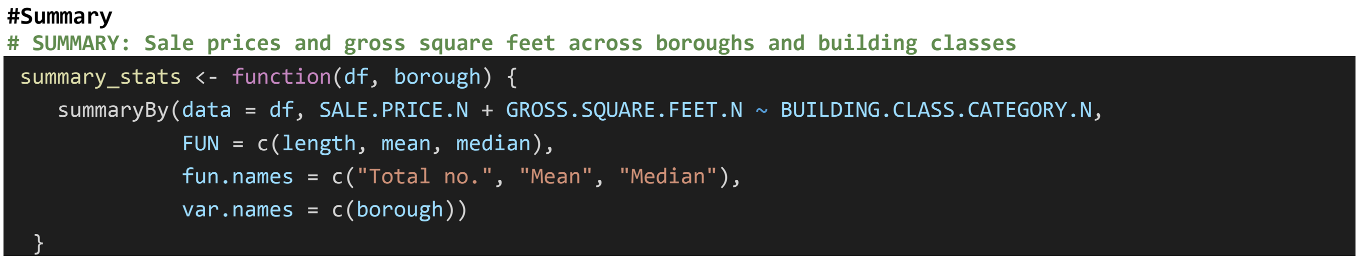


```
sp_freq(queensDf, "QUEENS")
```

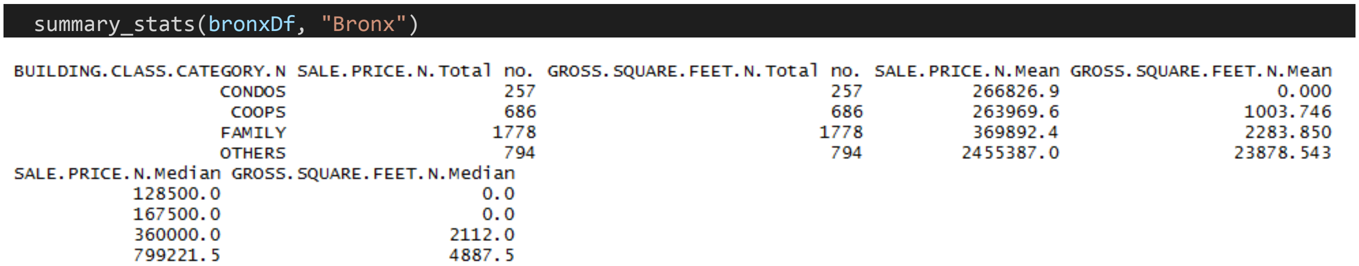


```
sp_freq(statenDf, "STATEN ISLAND")
```



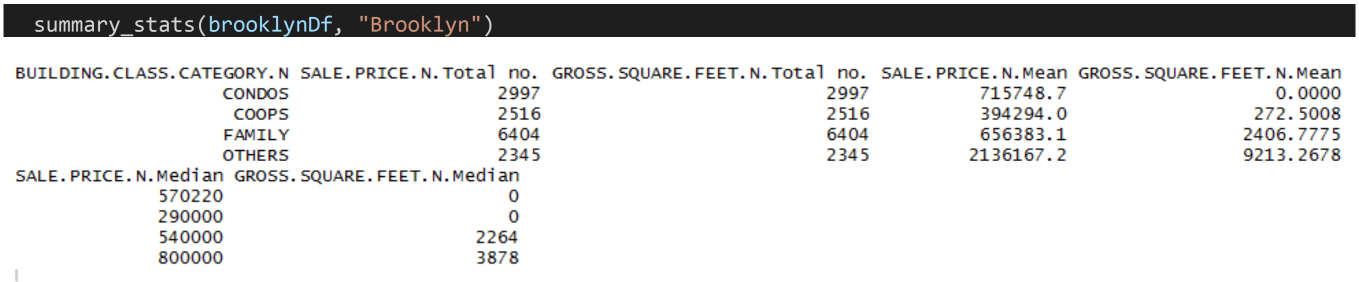


```
#Summary
# SUMMARY: Sale prices and gross square feet across boroughs and building classes
summary_stats <- function(df, borough) {
   summaryBy(data = df, SALE.PRICE.N + GROSS.SQUARE.FEET.N ~ BUILDING.CLASS.CATEGORY.N,
             FUN = c(length, mean, median),
             fun.names = c("Total no.", "Mean", "Median"),
             var.names = c(borough))
 }
```



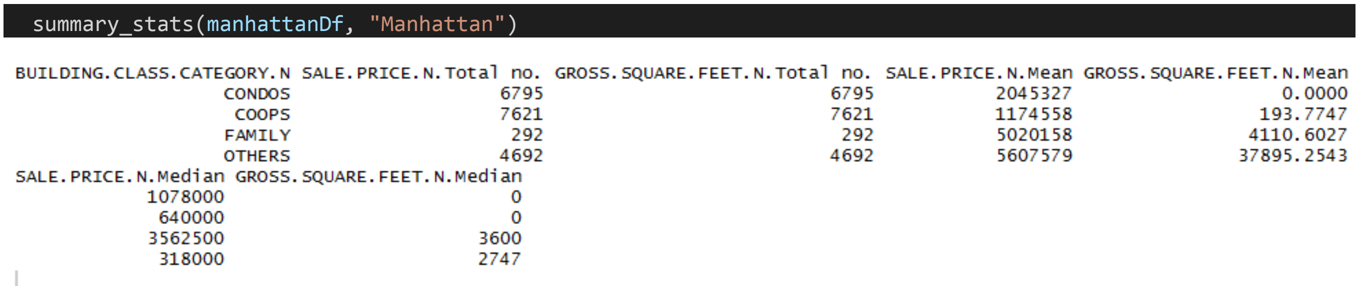
```
summary_stats(bronxDf, "Bronx")
```

```
BUILDING.CLASS.CATEGORY.N SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
                   CONDOS                    257                           257          266826.9                    0.000
                    COOPS                    686                           686          263969.6                 1003.746
                   FAMILY                   1778                          1778          369892.4                 2283.850
                   OTHERS                    794                           794         2455387.0                23878.543
SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
           128500.0                        0.0
           167500.0                        0.0
           360000.0                     2112.0
           799221.5                     4887.5
```



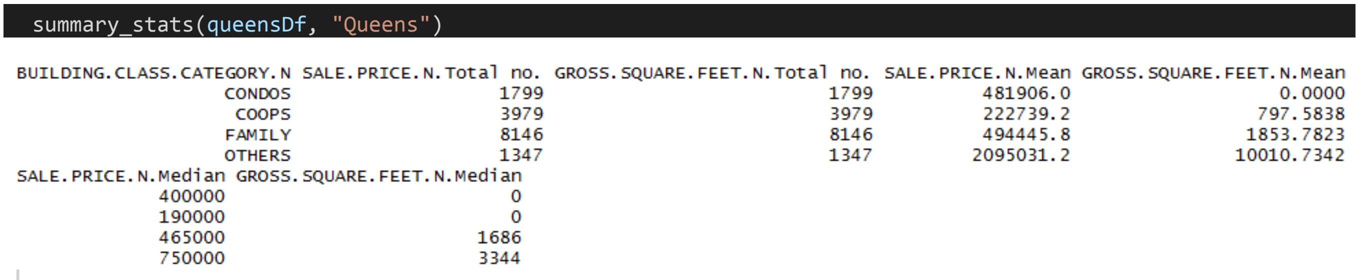
```
summary_stats(brooklynDf, "Brooklyn")
```

```
BUILDING.CLASS.CATEGORY.N SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
                   CONDOS                   2997                          2997          715748.7                   0.0000
                    COOPS                   2516                          2516          394294.0                 272.5008
                   FAMILY                   6404                          6404          656383.1                2406.7775
                   OTHERS                   2345                          2345         2136167.2                9213.2678
SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
             570220                          0
             290000                          0
             540000                       2264
             800000                       3878
```



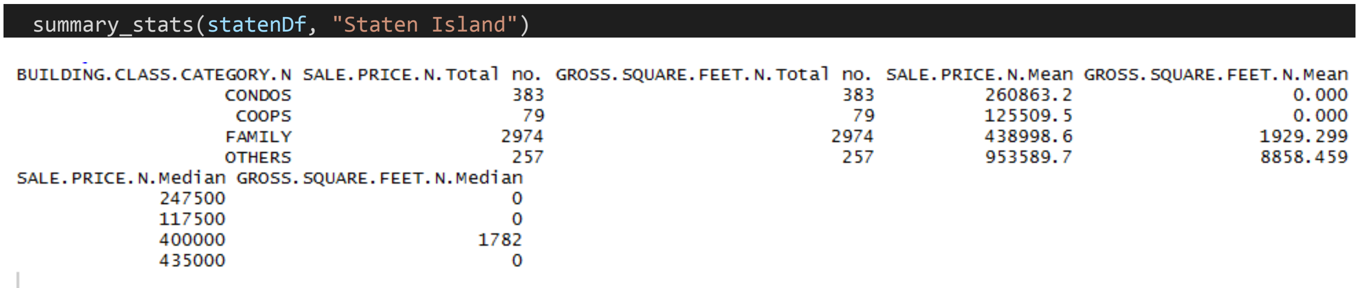
```
summary_stats(manhattanDf, "Manhattan")
```

```
BUILDING.CLASS.CATEGORY.N SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
                   CONDOS                   6795                          6795           2045327                   0.0000
                    COOPS                   7621                          7621           1174558                 193.7747
                   FAMILY                    292                           292           5020158                4110.6027
                   OTHERS                   4692                          4692           5607579               37895.2543
SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
            1078000                          0
             640000                          0
            3562500                       3600
             318000                       2747
```



```
summary_stats(queensDf, "Queens")
```

```
BUILDING.CLASS.CATEGORY.N SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
                   CONDOS                   1799                          1799          481906.0                   0.0000
                    COOPS                   3979                          3979          222739.2                 797.5838
                   FAMILY                   8146                          8146          494445.8                1853.7823
                   OTHERS                   1347                          1347         2095031.2               10010.7342
SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
             400000                          0
             190000                          0
             465000                       1686
             750000                       3344
```



```
summary_stats(statenDf, "Staten Island")
```

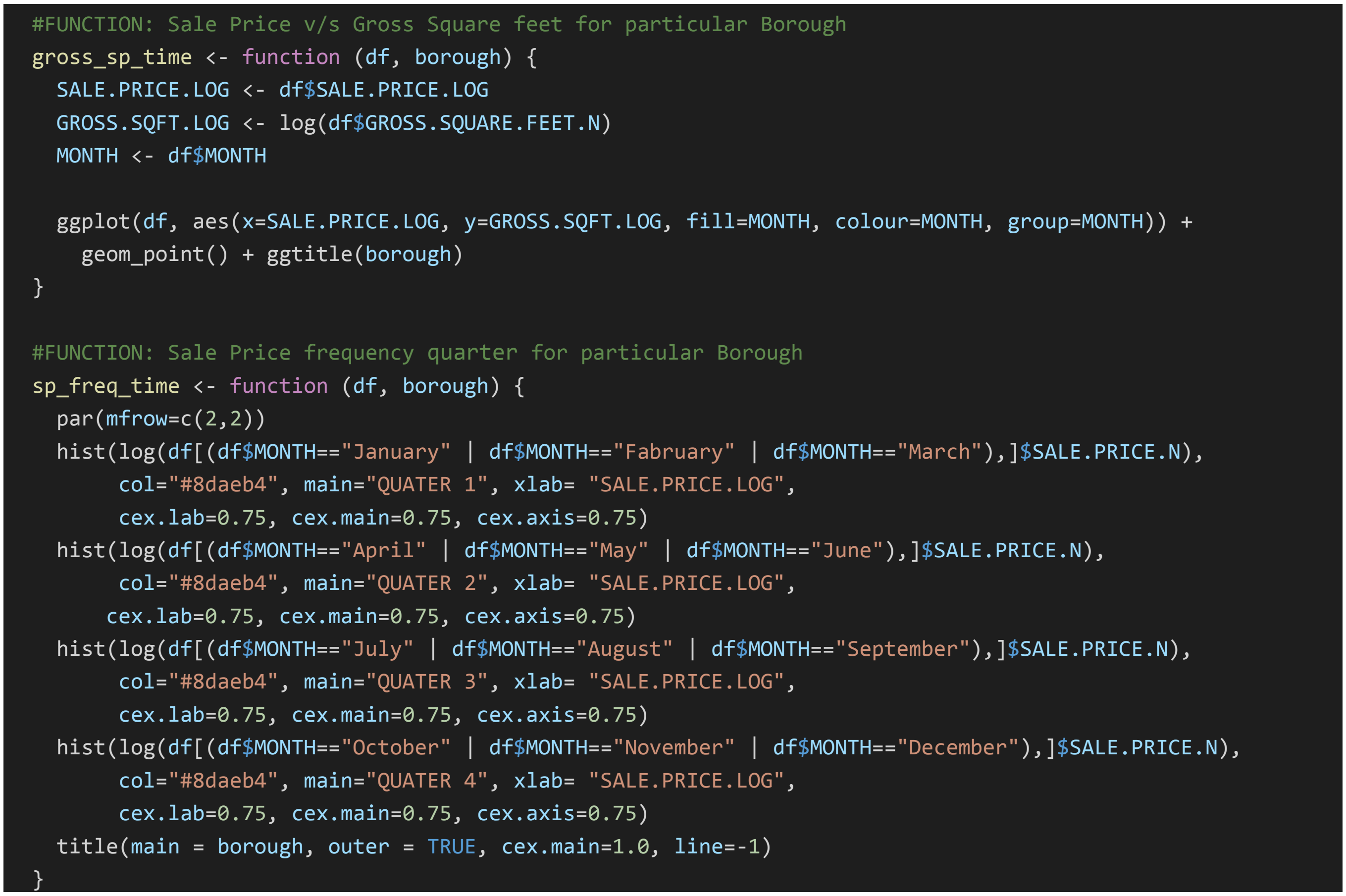
```
BUILDING.CLASS.CATEGORY.N SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
                   CONDOS                    383                           383          260863.2                    0.000
                    COOPS                     79                            79          125509.5                    0.000
                   FAMILY                   2974                          2974          438998.6                 1929.299
                   OTHERS                    257                           257          953589.7                 8858.459
SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
             247500                          0
             117500                          0
             400000                       1782
             435000                          0
```

Analysis across time

```
# DAY AND MONTH FETCHER FUNCTION
day_month <- function (df) {
  df$DAY <- format(df$SALE.DATE, "%A")
  df$DAY <- factor(df$DAY, levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday"))

  df$MONTH <- format(df$SALE.DATE, "%B")
  df$MONTH <- factor(df$MONTH, levels = c("January", "February", "March", "April", "May", "June",
                                          "July", "August", "September", "October", "November",
                                          "December"))

  return(df)
}

#Building data frames
bronxTime <- day_month(bronxDf)
brooklynTime <- day_month(brooklynDf)
manhattanTime <- day_month(manhattanDf)
queensTime <- day_month(queensDf)
statenTime <- day_month(statenDf)
```

#Functions

```
#FUNCTION: Total sale of Building Class in Months
monthly_sale <- function (class) {
  filter <- bronxTime$SALE.PRICE.LOG[bronxTime$BUILDING.CLASS.CATEGORY.N == class]
  month <- bronxTime$MONTH[bronxTime$BUILDING.CLASS.CATEGORY.N == class]
  borough_1 <- data.frame(BOROUGH = rep("Bronx", length(filter)), SALE.PRICE.LOG=filter, MONTH=month)
  filter <- brooklynTime$SALE.PRICE.LOG[brooklynTime$BUILDING.CLASS.CATEGORY.N == class]
  month <- brooklynTime$MONTH[brooklynTime$BUILDING.CLASS.CATEGORY.N == class]
  borough_2 <- data.frame(BOROUGH = rep("Brooklyn", length(filter)), SALE.PRICE.LOG=filter, MONTH=month)

  filter <- manhattanTime$SALE.PRICE.LOG[manhattanTime$BUILDING.CLASS.CATEGORY.N == class]
  month <- manhattanTime$MONTH[manhattanTime$BUILDING.CLASS.CATEGORY.N == class]
  borough_3 <- data.frame(BOROUGH = rep("Manhattan", length(filter)), SALE.PRICE.LOG=filter, MONTH=month)

  filter <- queensTime$SALE.PRICE.LOG[queensTime$BUILDING.CLASS.CATEGORY.N == class]
  month <- queensTime$MONTH[queensTime$BUILDING.CLASS.CATEGORY.N == class]
  borough_4 <- data.frame(BOROUGH = rep("Queens", length(filter)), SALE.PRICE.LOG=filter, MONTH=month)

  filter <- statenTime$SALE.PRICE.LOG[statenTime$BUILDING.CLASS.CATEGORY.N == class]
  month <- statenTime$MONTH[statenTime$BUILDING.CLASS.CATEGORY.N == class]
borough_5 <- data.frame(BOROUGH=rep("Staten Island", length(filter)), SALE.PRICE.LOG=filter,MONTH=month)

  finalDf <- rbind(borough_1, borough_2, borough_3, borough_4, borough_5)
  ggplot(finalDf, aes(x=MONTH, y=SALE.PRICE.LOG, fill=MONTH, colour=MONTH, group=MONTH)) +
   theme(axis.text.x=element_text(angle = 90, vjust = 0.5)) +
    geom_boxplot() + ggtitle(class)
}

#FUNCTION: Number of sale of Building Class by Month for particular Borough
building_sale_month <- function (df, class, borough) {
  SALE.PRICE.LOG <- df$SALE.PRICE.LOG[df$BUILDING.CLASS.CATEGORY.N == class]
  ggplot(df, aes(x=MONTH, y=SALE.PRICE.LOG, fill=MONTH, colour=MONTH, group=MONTH)) +
    geom_boxplot() + theme(axis.text.x=element_text(angle = 90, vjust = 0.5)) +
    xlab(borough) + ggtitle(class)
}
```



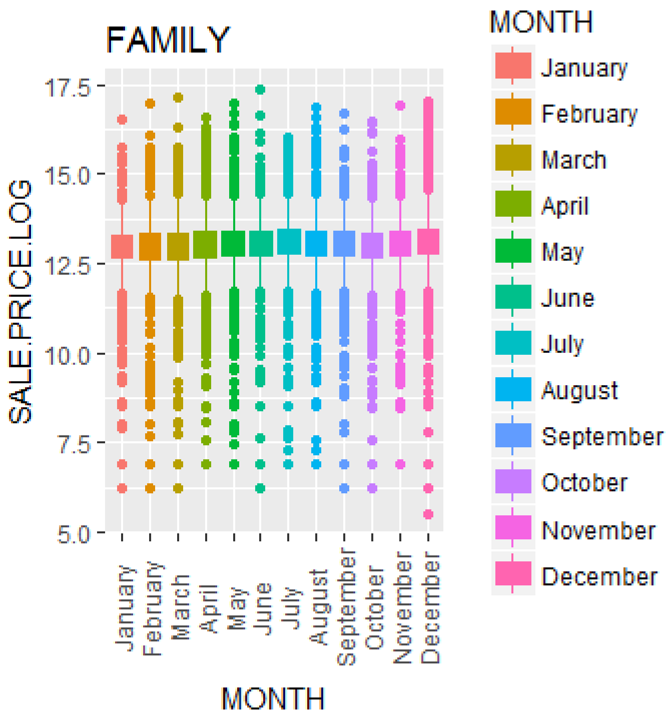
```
#FUNCTION: Sale Price v/s Gross Square feet for particular Borough
gross_sp_time <- function (df, borough) {
  SALE.PRICE.LOG <- df$SALE.PRICE.LOG
  GROSS.SQFT.LOG <- log(df$GROSS.SQUARE.FEET.N)
  MONTH <- df$MONTH

  ggplot(df, aes(x=SALE.PRICE.LOG, y=GROSS.SQFT.LOG, fill=MONTH, colour=MONTH, group=MONTH)) +
    geom_point() + ggtitle(borough)
}

#FUNCTION: Sale Price frequency quarter for particular Borough
sp_freq_time <- function (df, borough) {
  par(mfrow=c(2,2))
  hist(log(df[(df$MONTH=="January" | df$MONTH=="Fabruary" | df$MONTH=="March"),]$SALE.PRICE.N),
       col="#8daeb4", main="QUATER 1", xlab= "SALE.PRICE.LOG",
       cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
  hist(log(df[(df$MONTH=="April" | df$MONTH=="May" | df$MONTH=="June"),]$SALE.PRICE.N),
       col="#8daeb4", main="QUATER 2", xlab= "SALE.PRICE.LOG",
      cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
  hist(log(df[(df$MONTH=="July" | df$MONTH=="August" | df$MONTH=="September"),]$SALE.PRICE.N),
       col="#8daeb4", main="QUATER 3", xlab= "SALE.PRICE.LOG",
       cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
  hist(log(df[(df$MONTH=="October" | df$MONTH=="November" | df$MONTH=="December"),]$SALE.PRICE.N),
       col="#8daeb4", main="QUATER 4", xlab= "SALE.PRICE.LOG",
       cex.lab=0.75, cex.main=0.75, cex.axis=0.75)
  title(main = borough, outer = TRUE, cex.main=1.0, line=-1)
}
```

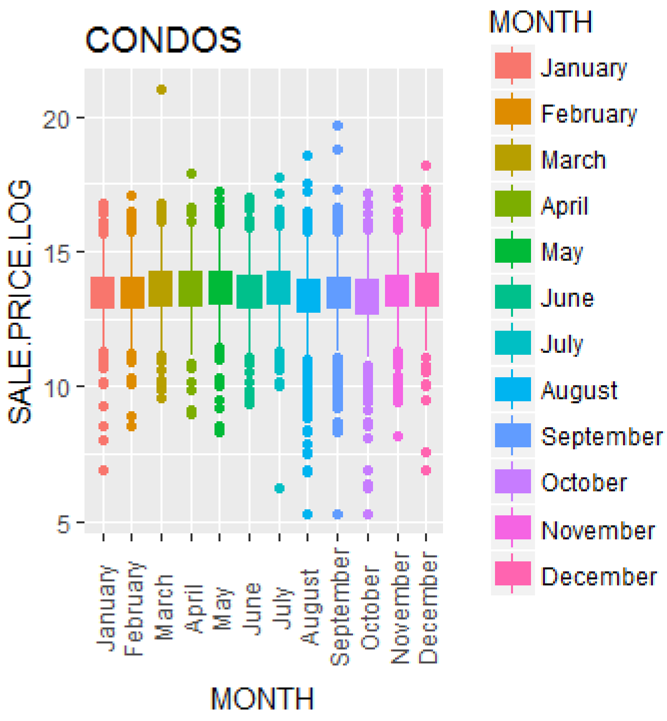
#Plot

```
#PLOT: Total sale of Building Class in Months
```

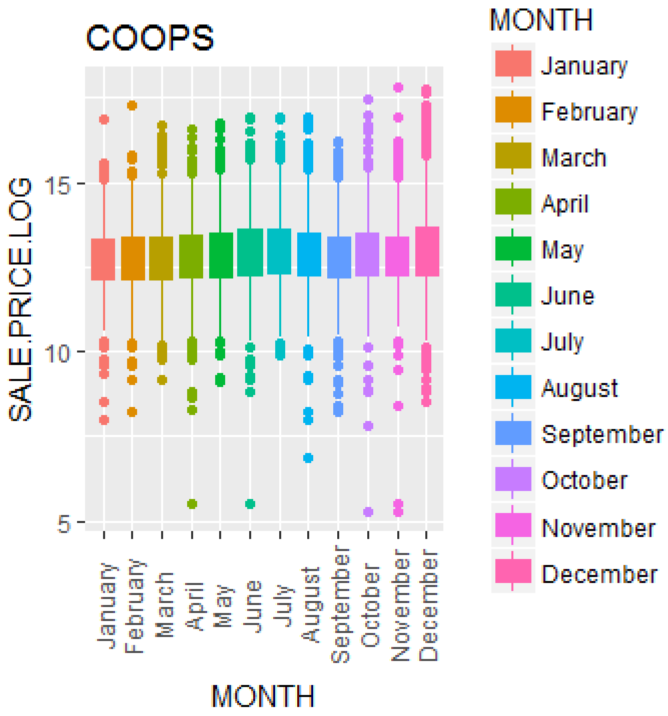
```
monthly_sale("FAMILY")
```

```
monthly_sale("CONDOS")
```

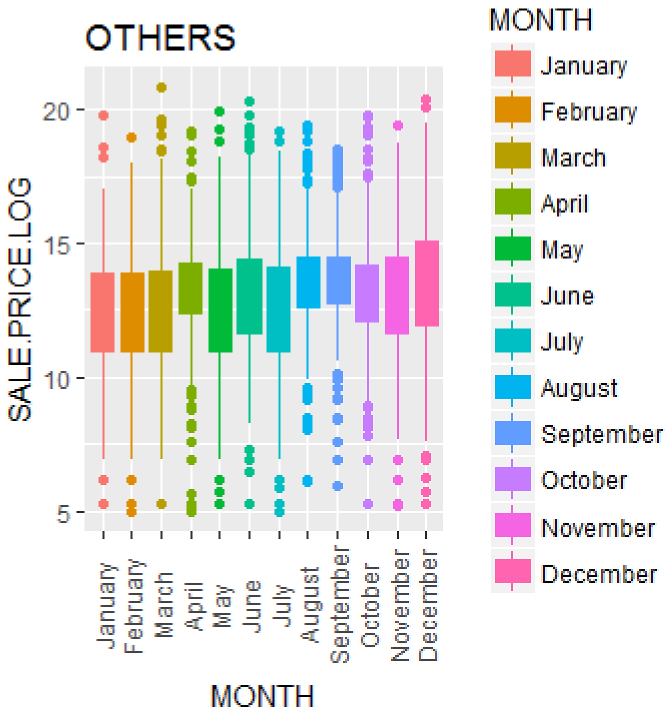




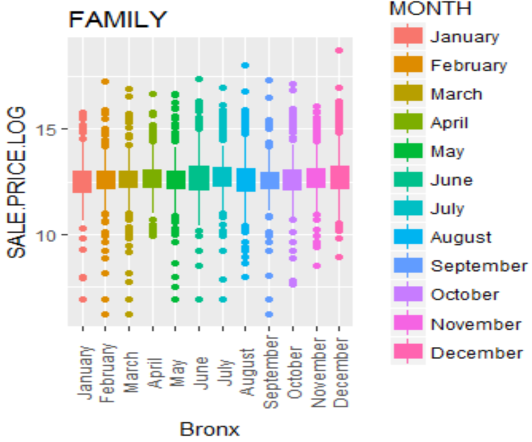
```
monthly_sale("COOPS")
```



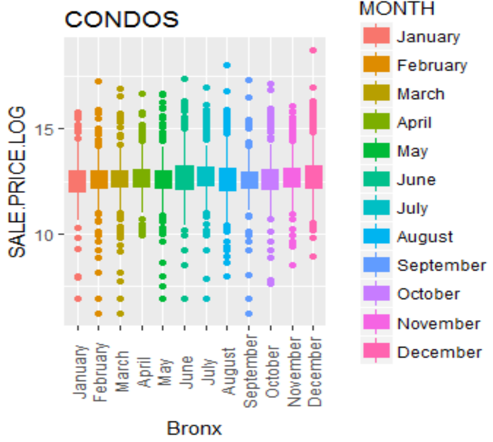
```
monthly_sale("OTHERS")
```



```
#PLOT: Number of sale of Building Class by Month for particular Borough
building_sale_month(bronxTime, "FAMILY", "Bronx")
```

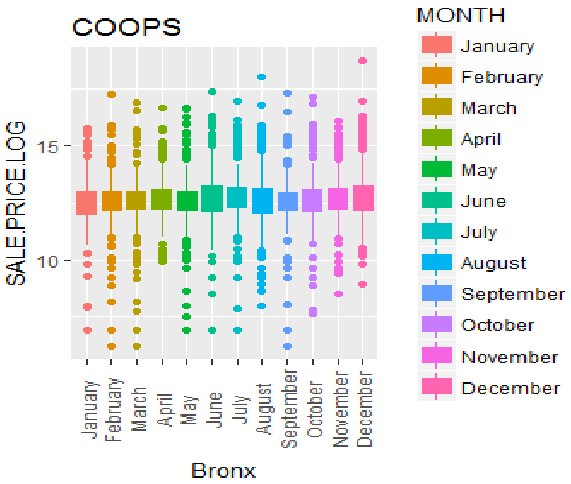


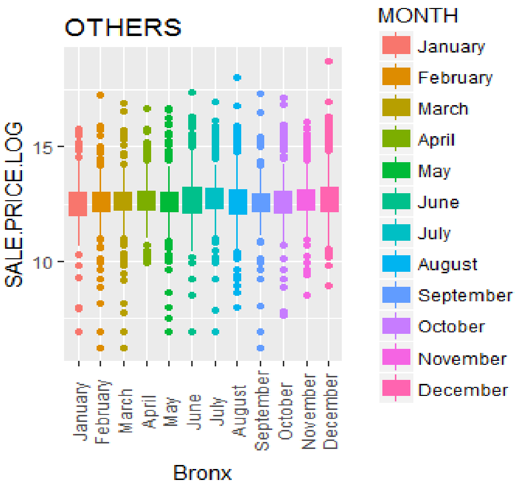
```
building_sale_month(bronxTime, "CONDOS", "Bronx")
```



```
building_sale_month(bronxTime, "COOPS", "Bronx")
```

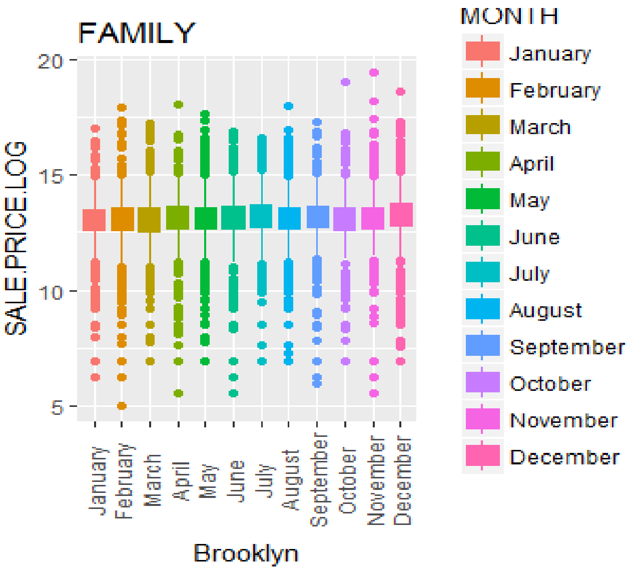
```
building_sale_month(bronxTime, "OTHERS", "Bronx")
```

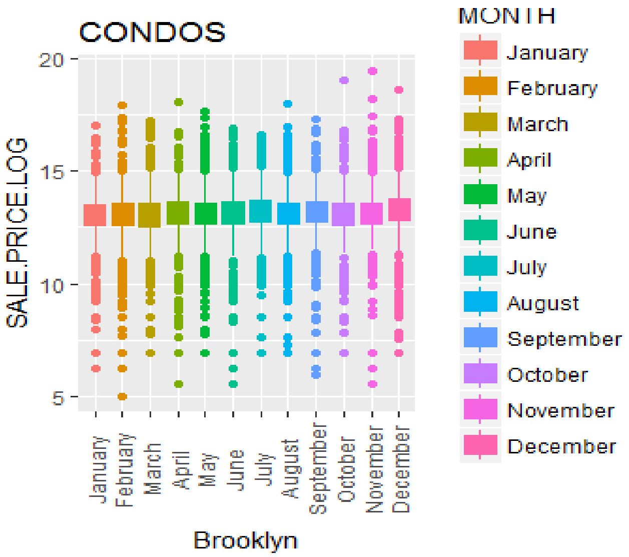




```
building_sale_month(brooklynTime, "FAMILY",
"Brooklyn")
```

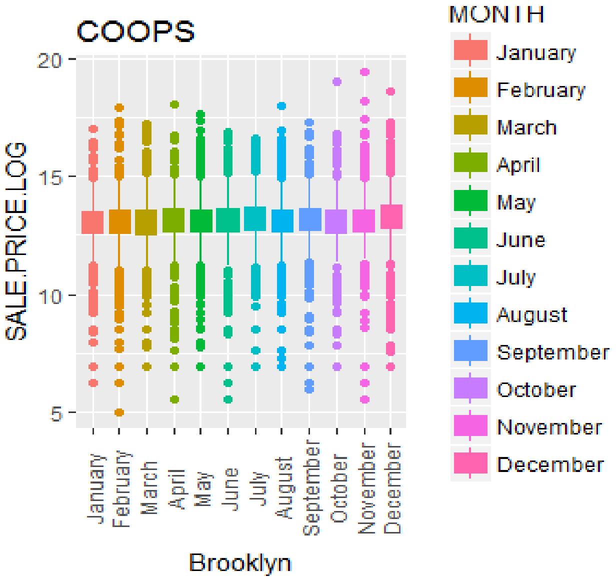
```
building_sale_month(brooklynTime, "CONDOS",
"Brooklyn")
```

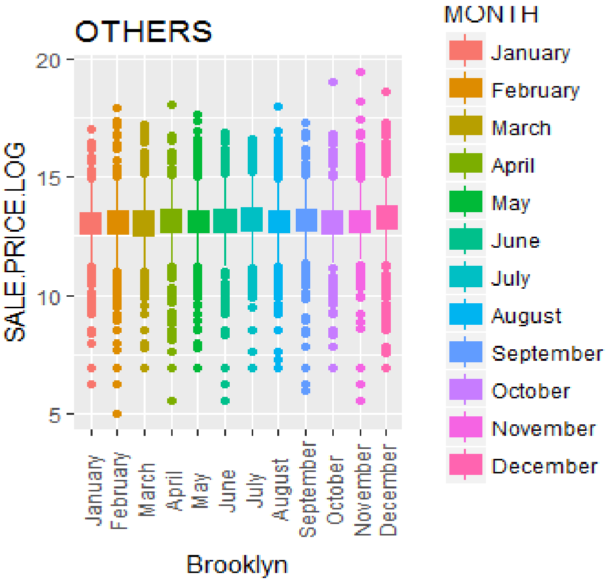




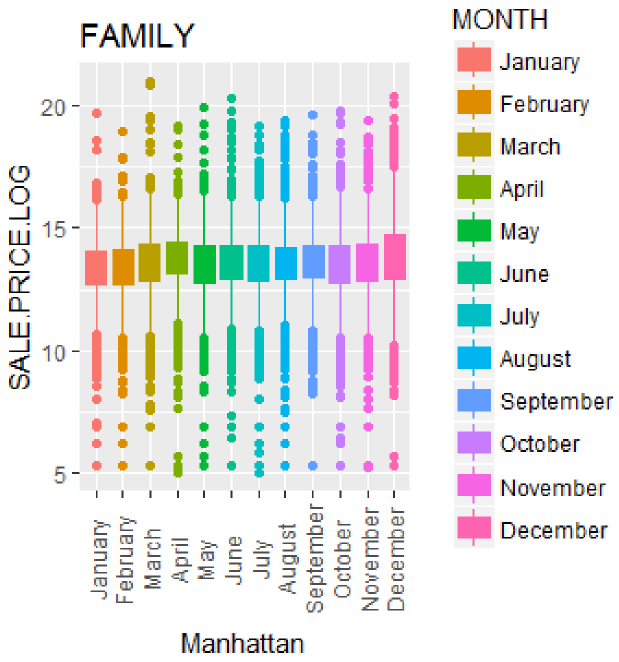
```
building_sale_month(brooklynTime, "COOPS",
"Brooklyn")
```

```
building_sale_month(brooklynTime, "OTHERS",
"Brooklyn")
```

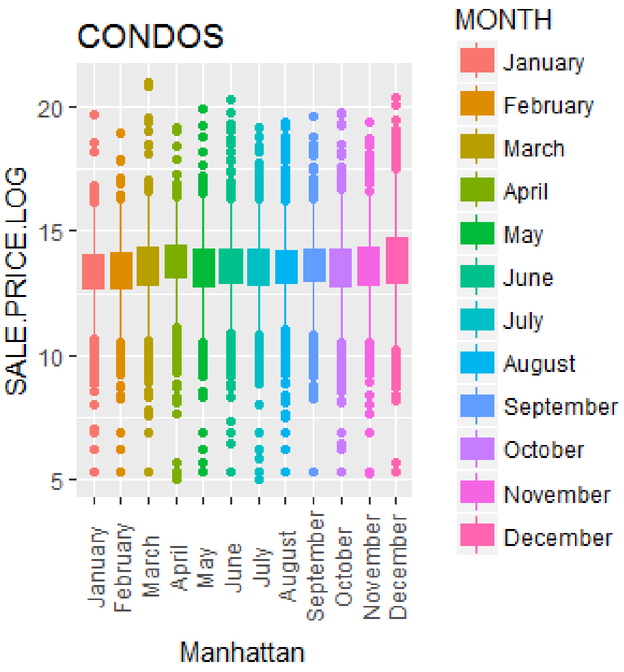




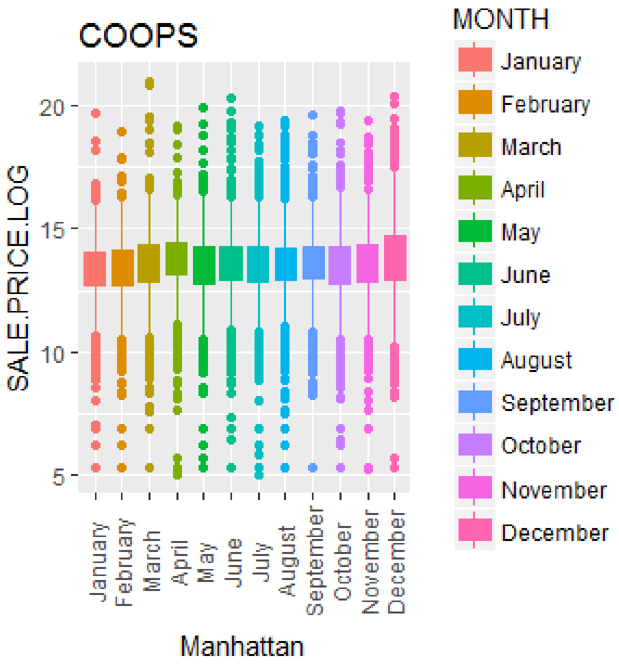
```
building_sale_month(manhattanTime, "FAMILY",
"Manhattan")
```



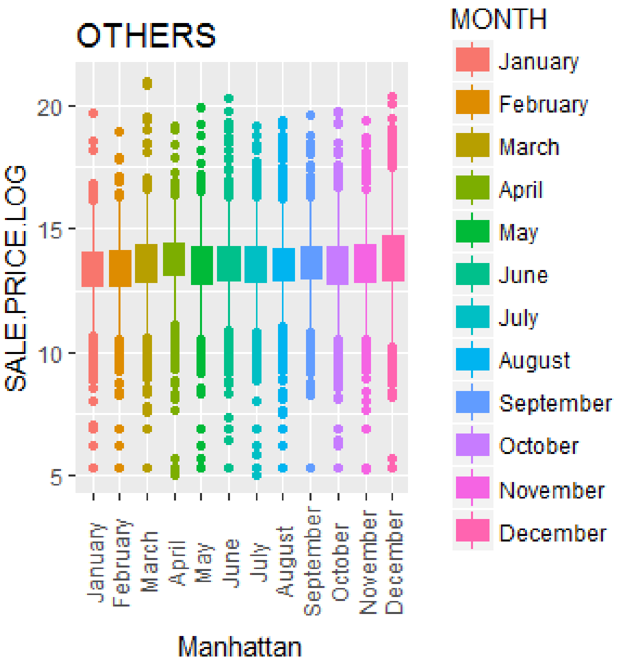
```
building_sale_month(manhattanTime, "CONDOS",
"Manhattan")
```



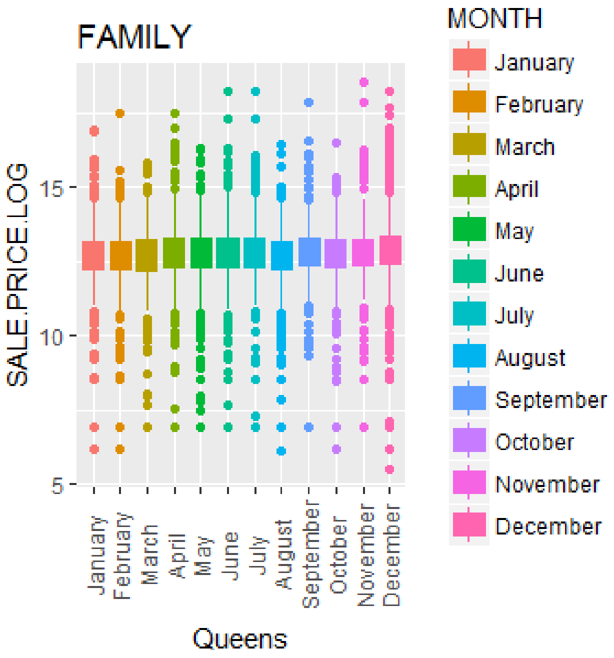
```
building_sale_month(manhattanTime, "COOPS",
"Manhattan")
```



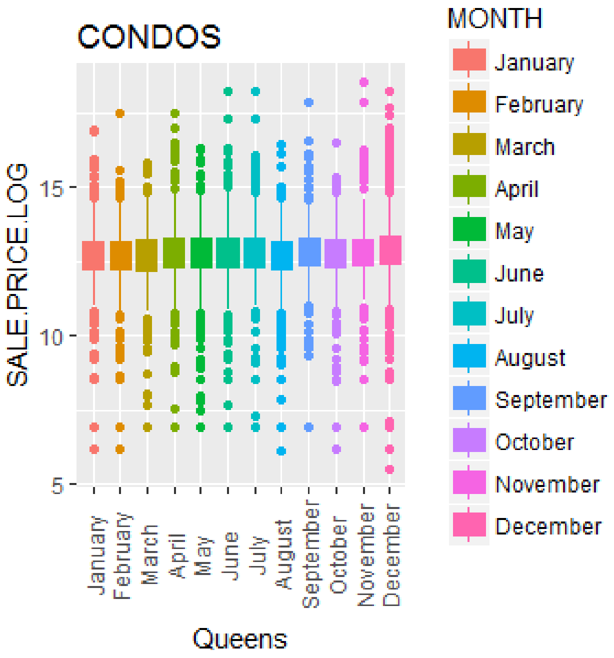
```
building_sale_month(manhattanTime, "OTHERS",
"Manhattan")
```



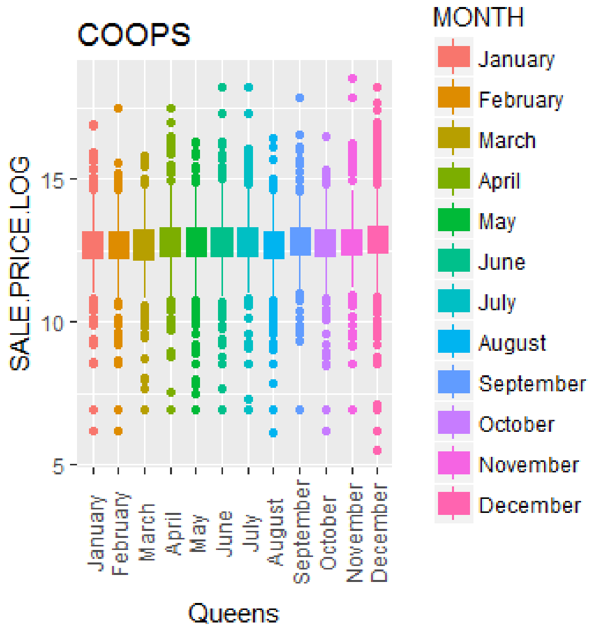
```
building_sale_month(queensTime, "FAMILY", "Queens")
```



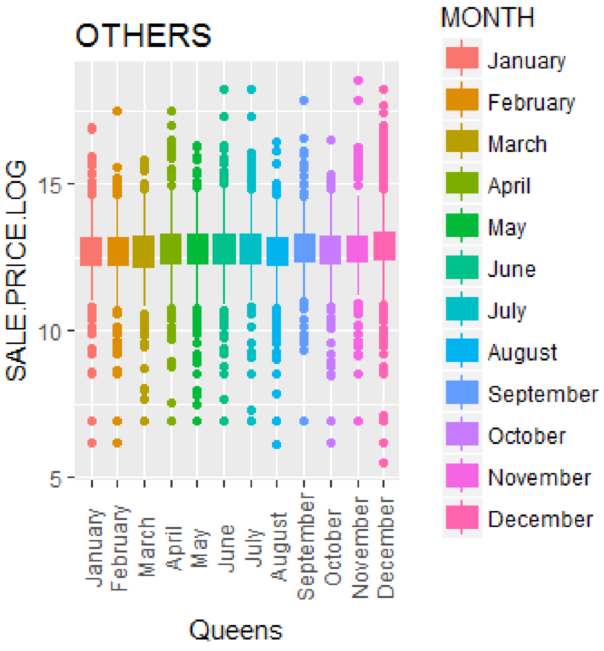
```
building_sale_month(queensTime, "CONDOS", "Queens")
```



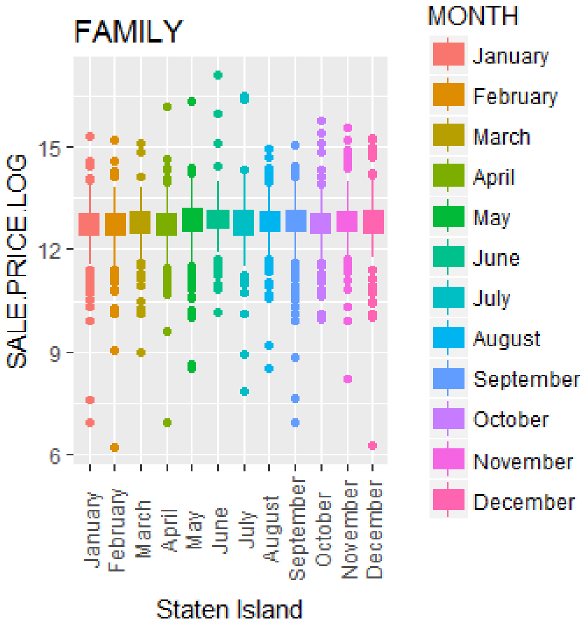
```
building_sale_month(queensTime, "COOPS", "Queens")
```



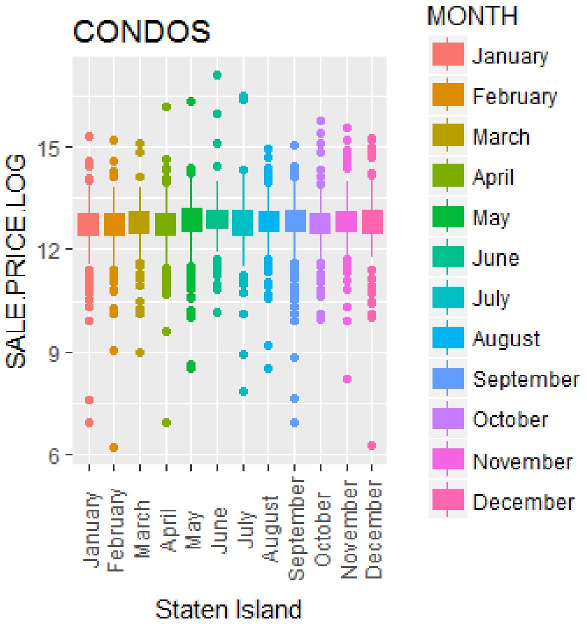
```
building_sale_month(queensTime, "OTHERS", "Queens")
```



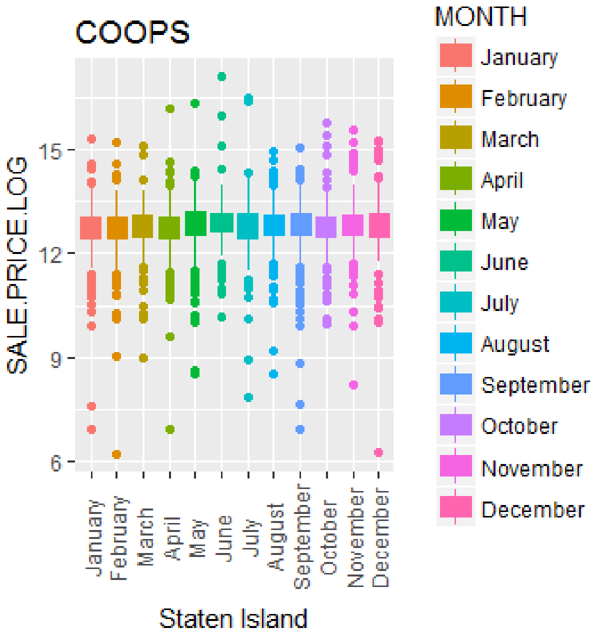
```
building_sale_month(statenTime, "FAMILY", "Staten
Island")
```



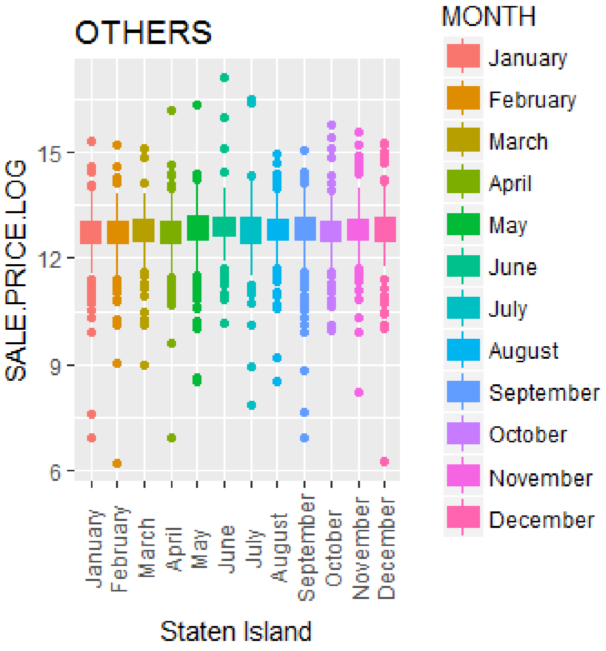
```
building_sale_month(statenTime, "CONDOS", "Staten
Island")
```



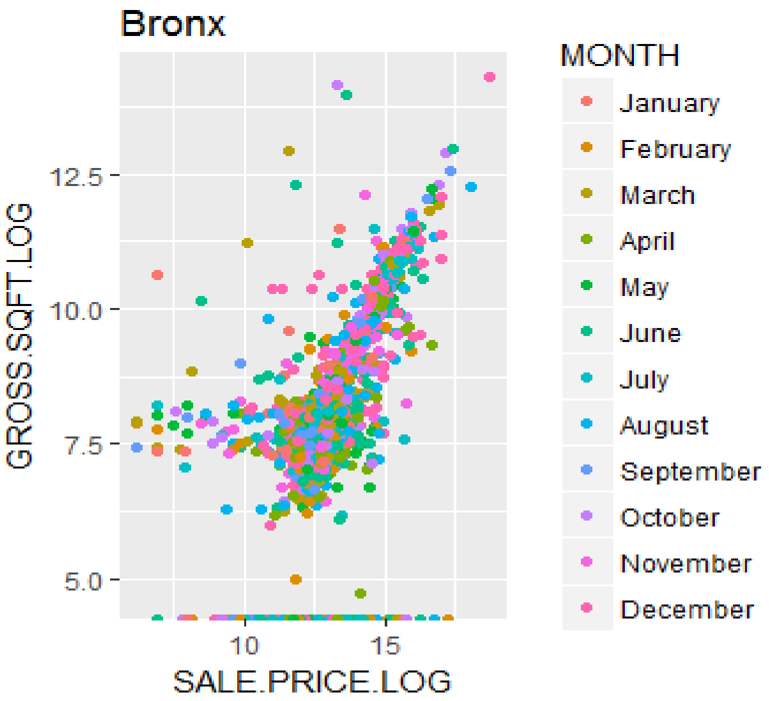
```
building_sale_month(statenTime, "COOPS", "Staten
Island")
```



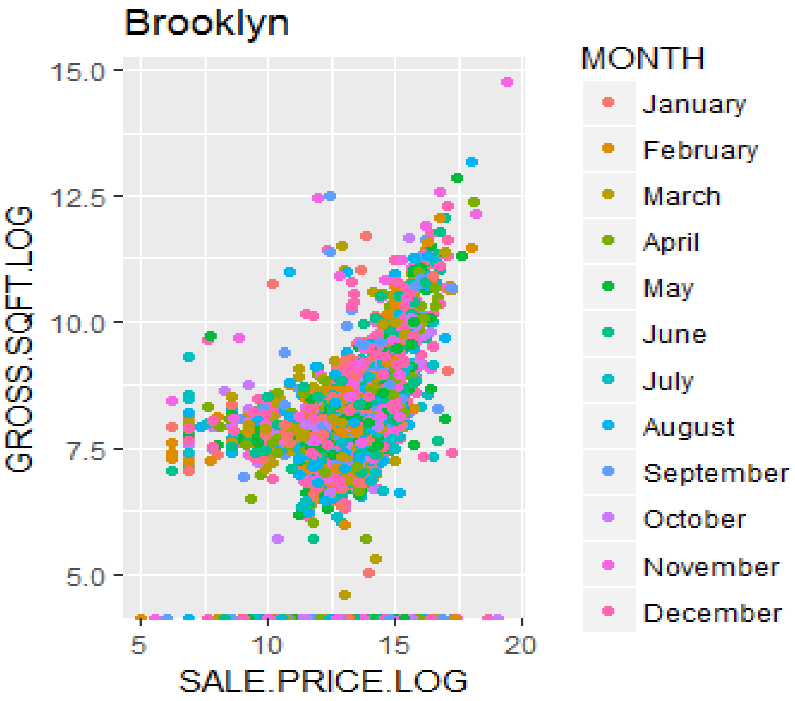
```
building_sale_month(statenTime, "OTHERS", "Staten
Island")
```



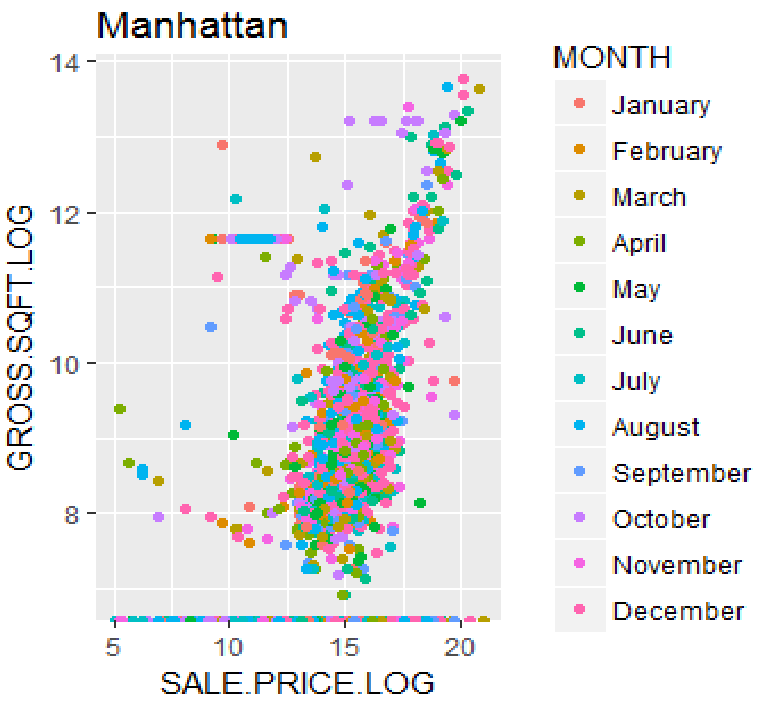
```
# PLOT: Sale Price v/s Gross Square feet for particular Borough
gross_sp_time(bronxTime, "Bronx")
```



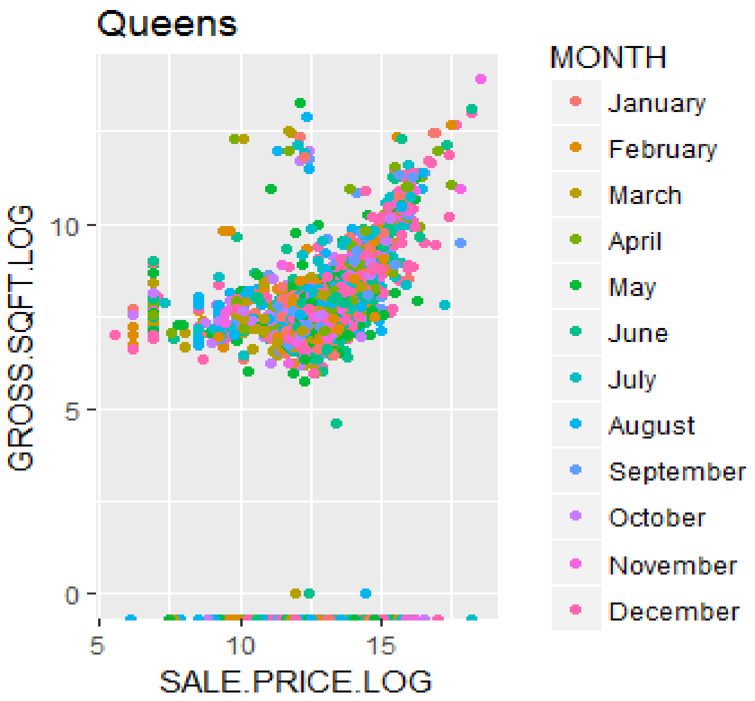
```
gross_sp_time(brooklynTime, "Brooklyn")
```



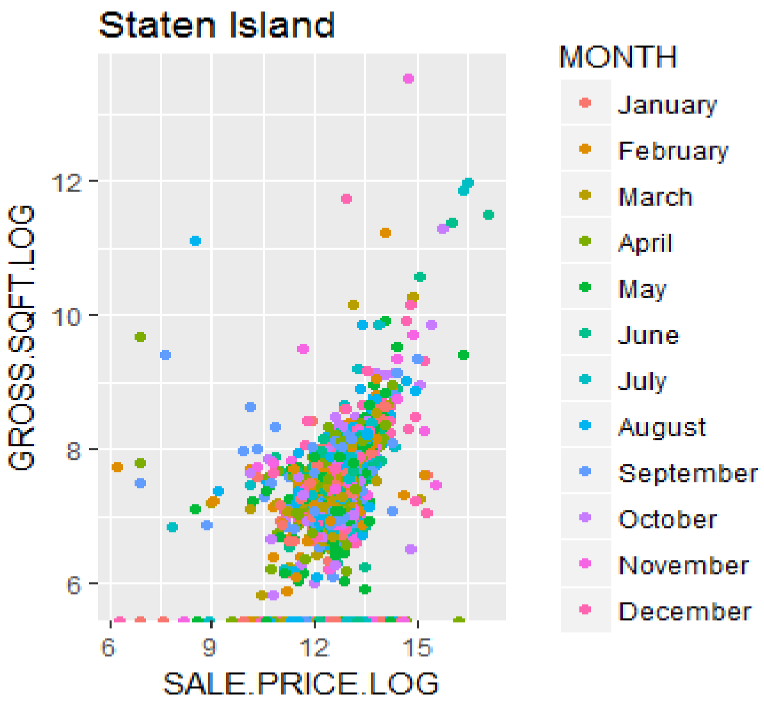
```
gross_sp_time(manhattanTime, "Manhattan")
```



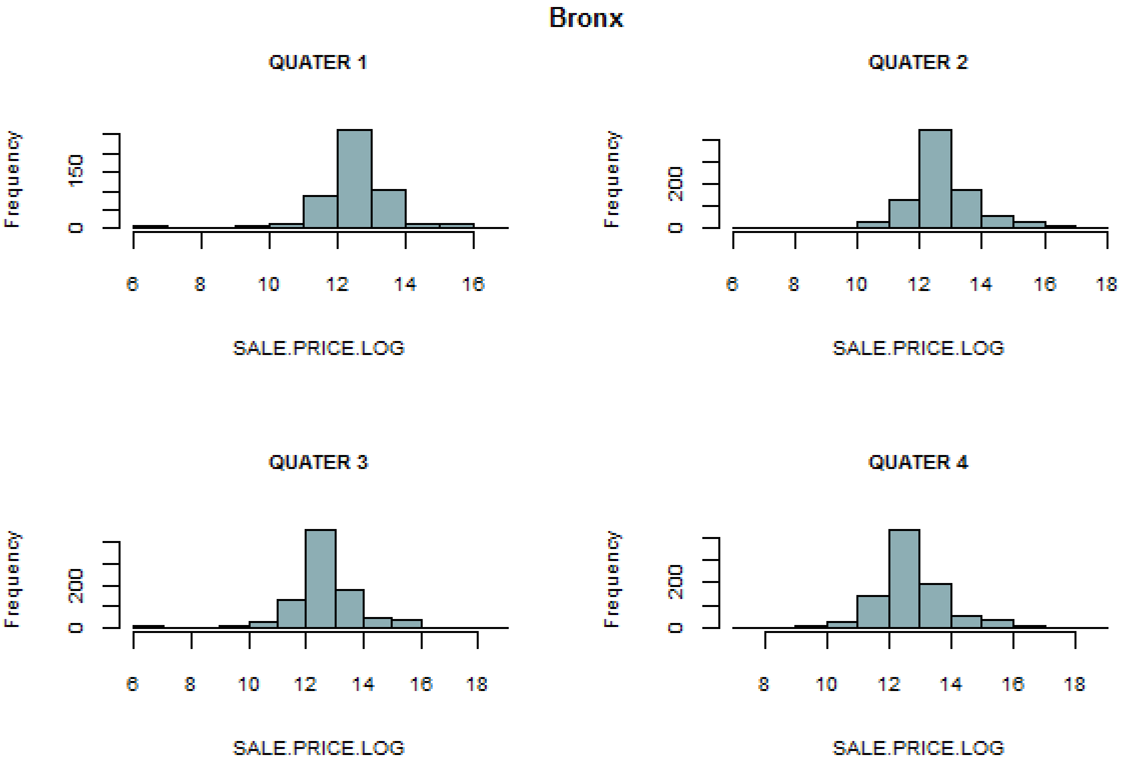
```
gross_sp_time(queensTime, "Queens")
```



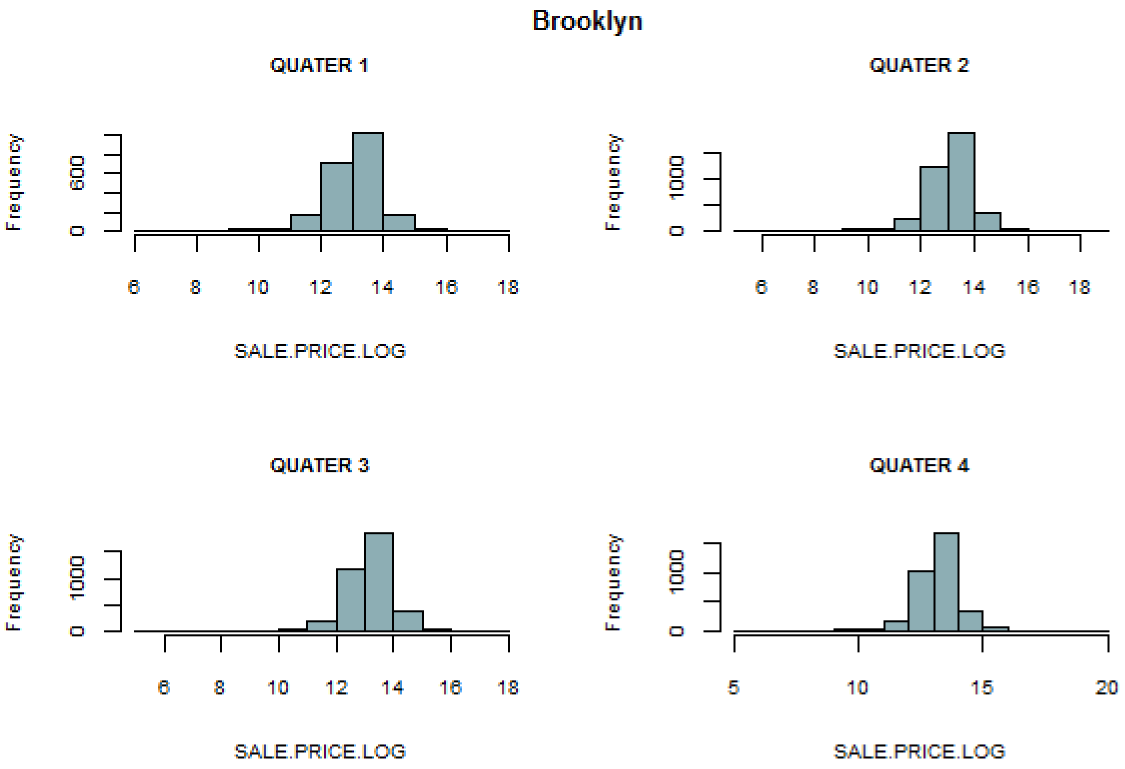
```
gross_sp_time(statenTime, "Staten Island")
```



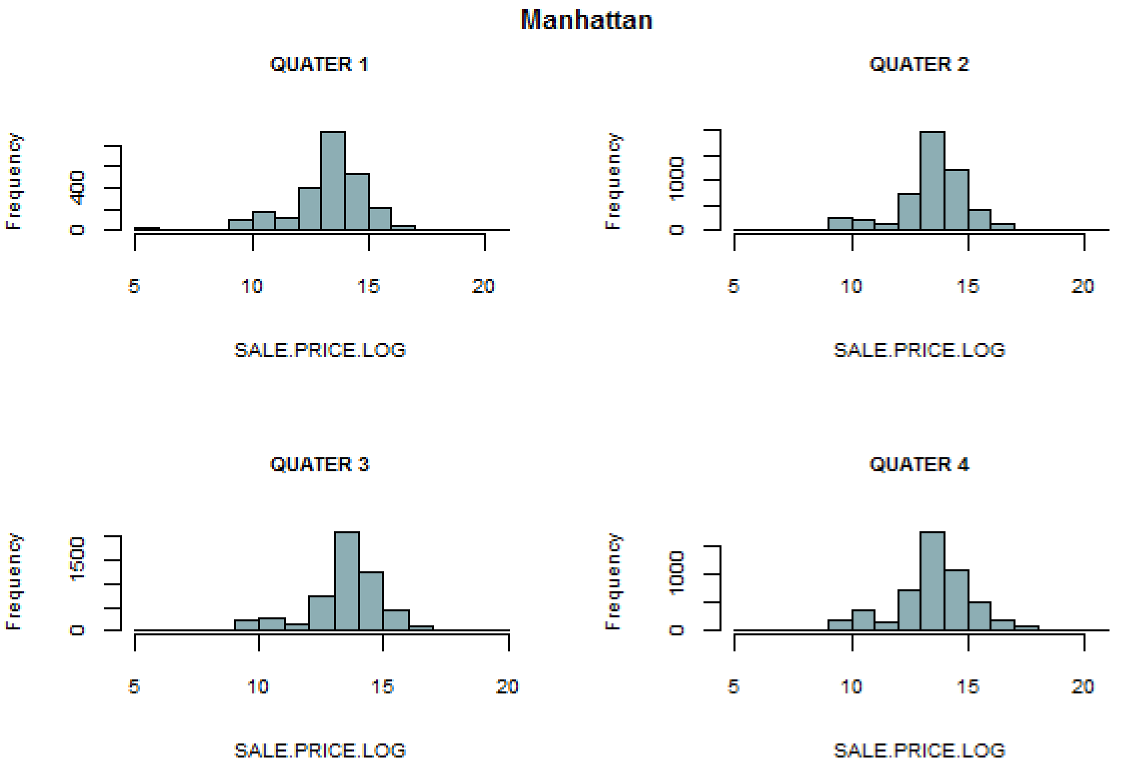
```
#PLOT: Sale Price frequency per quarter for particular Borough
sp_freq_time(bronxTime, "Bronx")
```



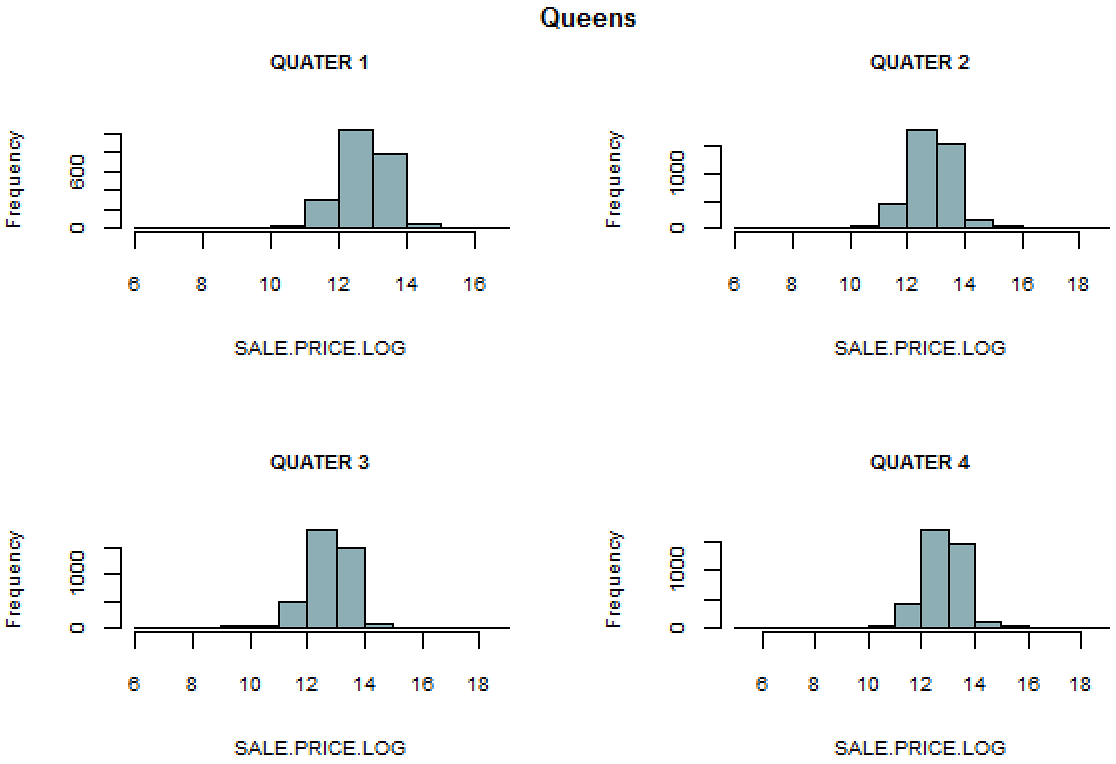
```
sp_freq_time(brooklynTime, "Brooklyn")
```



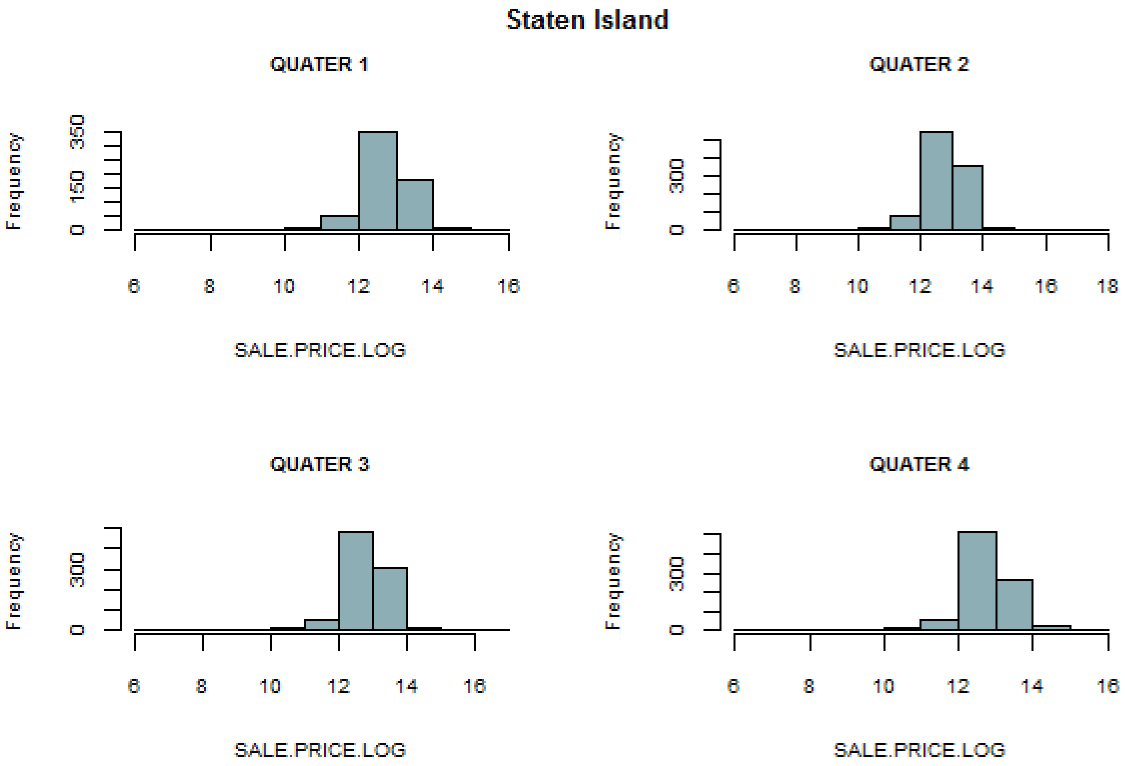
```
sp_freq_time(manhattanTime, "Manhattan")
```



```
sp_freq_time(queensTime, "Queens")
```

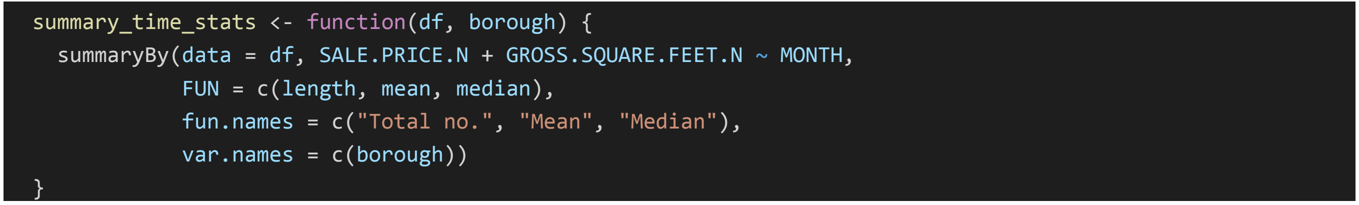


```
sp_freq_time(statenTime, "Staten Island")
```

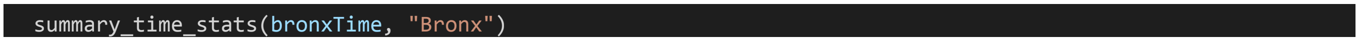


#Summary

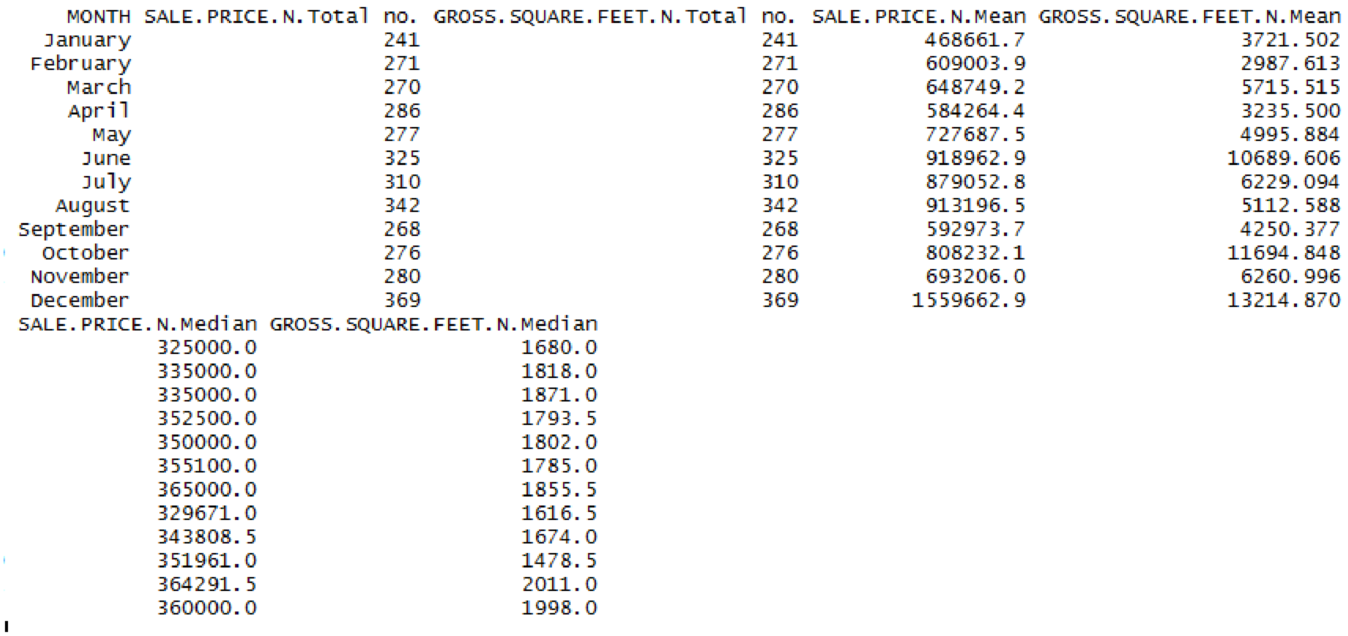
# SUMMARY: Sale prices and gross square feet across boroughs and time



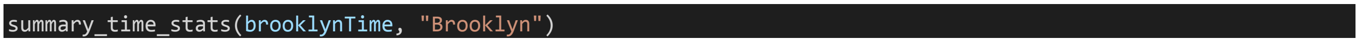
```
summary_time_stats <- function(df, borough) {
  summaryBy(data = df, SALE.PRICE.N + GROSS.SQUARE.FEET.N ~ MONTH,
            FUN = c(length, mean, median),
            fun.names = c("Total no.", "Mean", "Median"),
            var.names = c(borough))
}
```



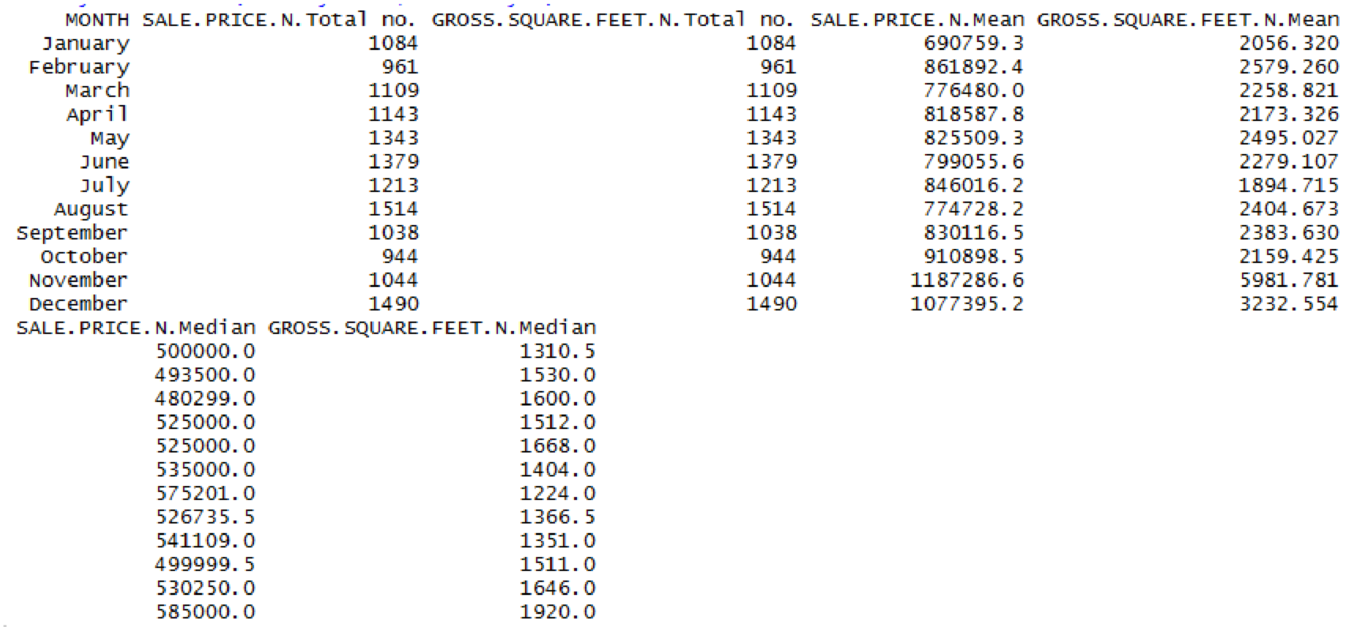
```
summary_time_stats(bronxTime, "Bronx")
```



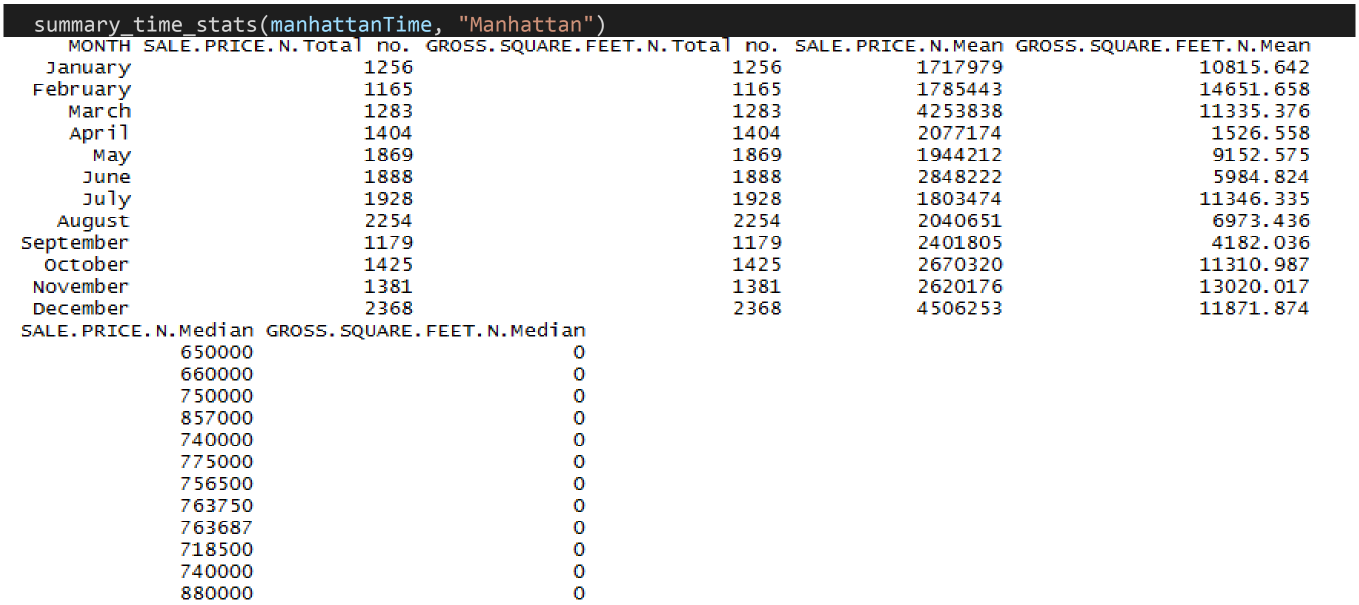
```
     MONTH SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
   January                    241                           241          468661.7                 3721.502
  February                    271                           271          609003.9                 2987.613
     March                    270                           270          648749.2                 5715.515
     April                    286                           286          584264.4                 3235.500
       May                    277                           277          727687.5                 4995.884
      June                    325                           325          918962.9                10689.606
      July                    310                           310          879052.8                 6229.094
    August                    342                           342          913196.5                 5112.588
 September                    268                           268          592973.7                 4250.377
   October                    276                           276          808232.1                11694.848
  November                    280                           280          693206.0                 6260.996
  December                    369                           369         1559662.9                13214.870
 SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
            325000.0                     1680.0
            335000.0                     1818.0
            335000.0                     1871.0
            352500.0                     1793.5
            350000.0                     1802.0
            355100.0                     1785.0
            365000.0                     1855.5
            329671.0                     1616.5
            343808.5                     1674.0
            351961.0                     1478.5
            364291.5                     2011.0
            360000.0                     1998.0
```



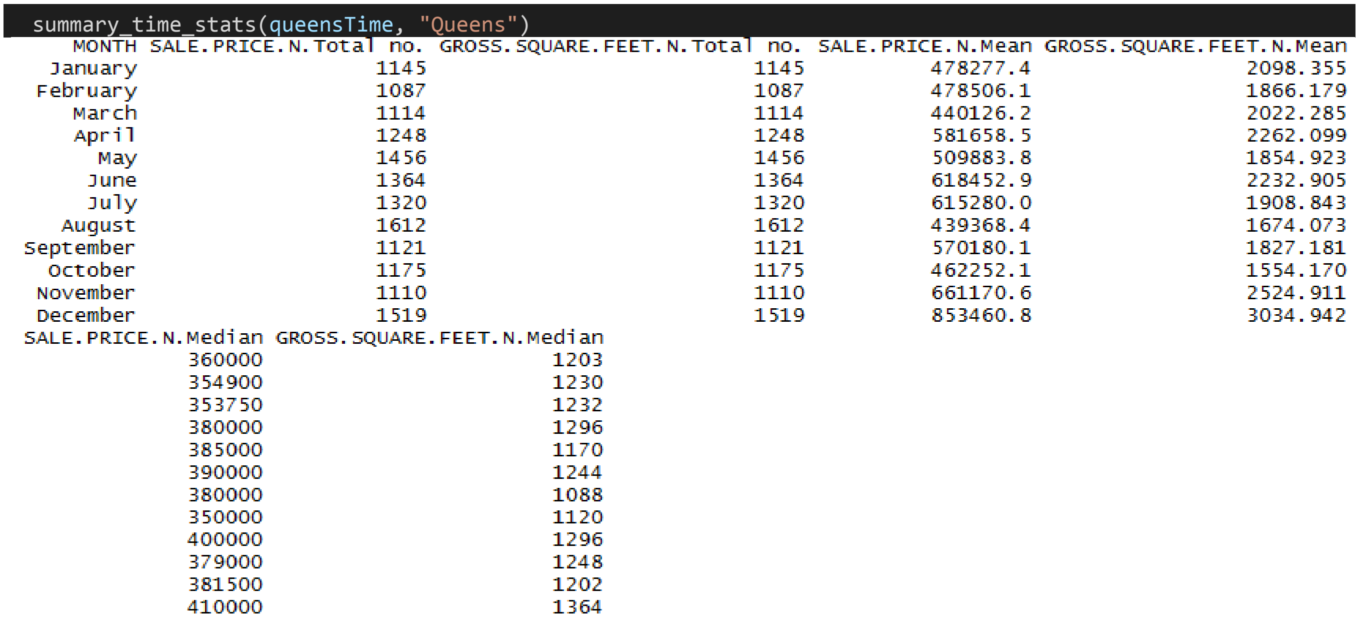
```
summary_time_stats(brooklynTime, "Brooklyn")
```



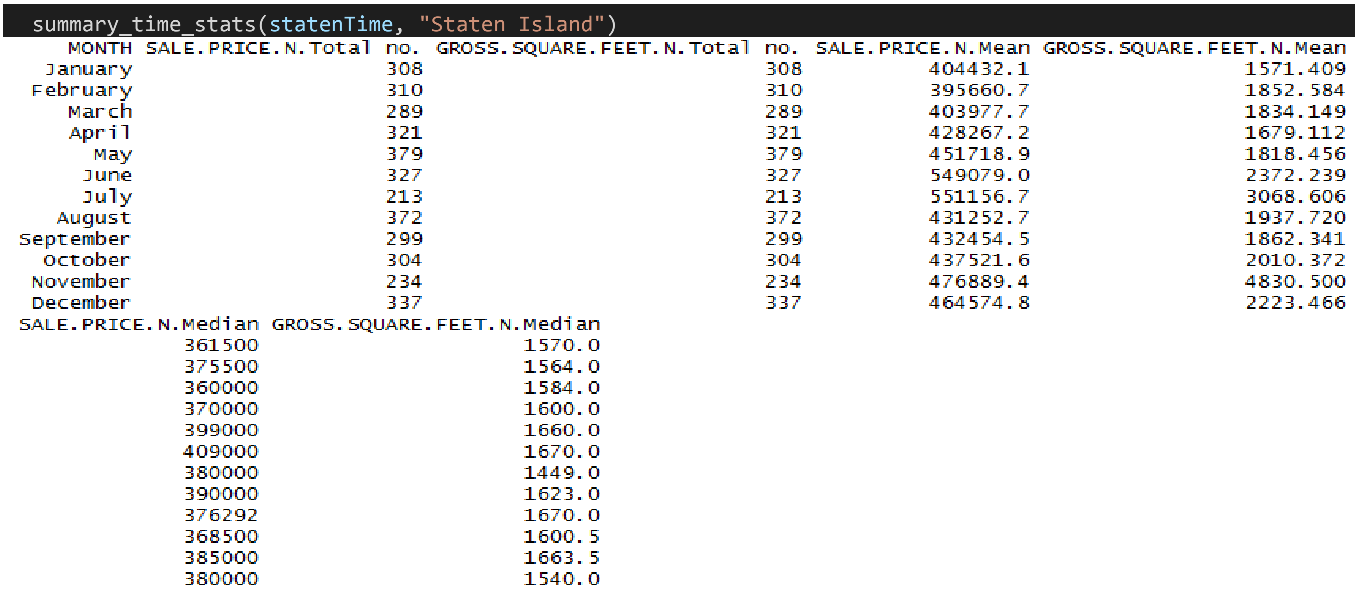
```
     MONTH SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
   January                   1084                          1084          690759.3                 2056.320
  February                    961                           961          861892.4                 2579.260
     March                   1109                          1109          776480.0                 2258.821
     April                   1143                          1143          818587.8                 2173.326
       May                   1343                          1343          825509.3                 2495.027
      June                   1379                          1379          799055.6                 2279.107
      July                   1213                          1213          846016.2                 1894.715
    August                   1514                          1514          774728.2                 2404.673
 September                   1038                          1038          830116.5                 2383.630
   October                    944                           944          910898.5                 2159.425
  November                   1044                          1044         1187286.6                 5981.781
  December                   1490                          1490         1077395.2                 3232.554
 SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
            500000.0                     1310.5
            493500.0                     1530.0
            480299.0                     1600.0
            525000.0                     1512.0
            525000.0                     1668.0
            535000.0                     1404.0
            575201.0                     1224.0
            526735.5                     1366.5
            541109.0                     1351.0
            499999.5                     1511.0
            530250.0                     1646.0
            585000.0                     1920.0
```



```
summary_time_stats(manhattanTime, "Manhattan")
     MONTH SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
   January                 1256                          1256           1717979                10815.642
  February                 1165                          1165           1785443                14651.658
     March                 1283                          1283           4253838                11335.376
     April                 1404                          1404           2077174                 1526.558
       May                 1869                          1869           1944212                 9152.575
      June                 1888                          1888           2848222                 5984.824
      July                 1928                          1928           1803474                11346.335
    August                 2254                          2254           2040651                 6973.436
 September                 1179                          1179           2401805                 4182.036
   October                 1425                          1425           2670320                11310.987
  November                 1381                          1381           2620176                13020.017
  December                 2368                          2368           4506253                11871.874
 SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
              650000                          0
              660000                          0
              750000                          0
              857000                          0
              740000                          0
              775000                          0
              756500                          0
              763750                          0
              763687                          0
              718500                          0
              740000                          0
              880000                          0
```



```
summary_time_stats(queensTime, "Queens")
     MONTH SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
   January                   1145                        1145          478277.4                 2098.355
  February                   1087                        1087          478506.1                 1866.179
     March                   1114                        1114          440126.2                 2022.285
     April                   1248                        1248          581658.5                 2262.099
       May                   1456                        1456          509883.8                 1854.923
      June                   1364                        1364          618452.9                 2232.905
      July                   1320                        1320          615280.0                 1908.843
    August                   1612                        1612          439368.4                 1674.073
 September                   1121                        1121          570180.1                 1827.181
   October                   1175                        1175          462252.1                 1554.170
  November                   1110                        1110          661170.6                 2524.911
  December                   1519                        1519          853460.8                 3034.942
 SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
              360000                       1203
              354900                       1230
              353750                       1232
              380000                       1296
              385000                       1170
              390000                       1244
              380000                       1088
              350000                       1120
              400000                       1296
              379000                       1248
              381500                       1202
              410000                       1364
```



```
summary_time_stats(statenTime, "Staten Island")
     MONTH SALE.PRICE.N.Total no. GROSS.SQUARE.FEET.N.Total no. SALE.PRICE.N.Mean GROSS.SQUARE.FEET.N.Mean
   January                    308                         308          404432.1                 1571.409
  February                    310                         310          395660.7                 1852.584
     March                    289                         289          403977.7                 1834.149
     April                    321                         321          428267.2                 1679.112
       May                    379                         379          451718.9                 1818.456
      June                    327                         327          549079.0                 2372.239
      July                    213                         213          551156.7                 3068.606
    August                    372                         372          431252.7                 1937.720
 September                    299                         299          432454.5                 1862.341
   October                    304                         304          437521.6                 2010.372
  November                    234                         234          476889.4                 4830.500
  December                    337                         337          464574.8                 2223.466
 SALE.PRICE.N.Median GROSS.SQUARE.FEET.N.Median
              361500                     1570.0
              375500                     1564.0
              360000                     1584.0
              370000                     1600.0
              399000                     1660.0
              409000                     1670.0
              380000                     1449.0
              390000                     1623.0
              376292                     1670.0
              368500                     1600.5
              385000                     1663.5
              380000                     1540.0
```

Problem 2 -

The datasets provided nyt1.csv, nyt2.csv, and nyt3.csv represents three (simulated) days of ads shown and clicks recorded on the New York Times homepage. Each row represents a single user. There are 5 columns: age, gender (0=female, 1=male), number impressions, number clicks, and logged-in. Use R to handle this data. Perform some exploratory data analysis:

- Create a new variable, age_group, that categorizes users as “<20”, “20-29”, “30-39”, “40-49”, “50-59”, “60-69”, and “70+”.
- For each day:
    - Plot the distribution of numbers of impressions and click-through-rate (CTR = #clicks/#impressions) for these age categories
    - Define a new variable to segment or categorize users based on their click behavior.
    - Explore the data and make visual and quantitative comparisons across user segments/demographics (<20-year-old males versus <20-year-old females or logged-in versus not, for example).
- Extend your analysis across days. Visualize some metrics and distributions over time.

Analysis 2 –

```
# CLEAR WORK DIRECTORY
rm(list=ls())
setwd("C:/Users/Paras Garg/Documents/R/FE Assignments")

# USING PACKAGES
library('doBy')
library('ggplot2')
library('plyr')

### LOADING DATASETS
day_1 <- data.frame(read.csv("nyt1.csv"))
day_2 <- data.frame(read.csv("nyt2.csv"))
day_3 <- data.frame(read.csv("nyt3.csv"))
```

Functions

```
  # FUNCTION: Create new variable ‘AGE_GROUP’
  age_var <- function (df) {
    df$Age_Group[df$Age < 20] <- '<20'
    df$Age_Group[df$Age >= 20 & df$Age < 30] <- '20-29'
    df$Age_Group[df$Age >= 30 & df$Age < 40] <- '30-39'
    df$Age_Group[df$Age >= 40 & df$Age < 50] <- '40-49'
    df$Age_Group[df$Age >= 50 & df$Age < 60] <- '50-59'
    df$Age_Group[df$Age >= 60 & df$Age < 70] <- '60-69'
    df$Age_Group[df$Age >= 70] <- '70+'
    return (df)
  }

  # FUNCTION: Data preparation by categorizing values
  data_prep <- function (df) {
    df$Gender[df$Gender == 1] <- "Male"
    df$Gender[df$Gender == 0] <- "Female"
    df$Signed_In[df$Signed_In == 1] <- "Signed In"
    df$Signed_In[df$Signed_In == 0] <- "Not Signed In"
    return(df)
  }
```

```
# FUNCTION: Create new variable according to clicks
click_var <- function (df) {
  df$Clicks_Cat <- cut(df$Clicks, c(-Inf,0,1,2,3,4,Inf), include.lowest = FALSE)
  return (df)
}
```

Data preparation

```
### Data Preparation
day_1 <- data_prep(age_var(day_1))
day_2 <- data_prep(age_var(day_2))
day_3 <- data_prep(age_var(day_3))
```

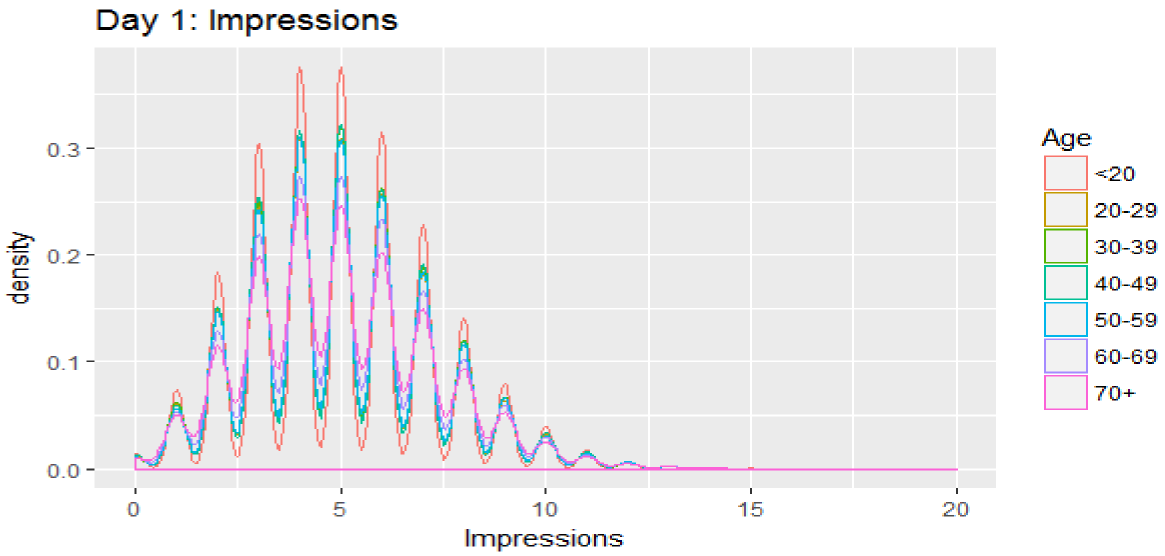
Impressions and CTR distribution for each day

#Functions

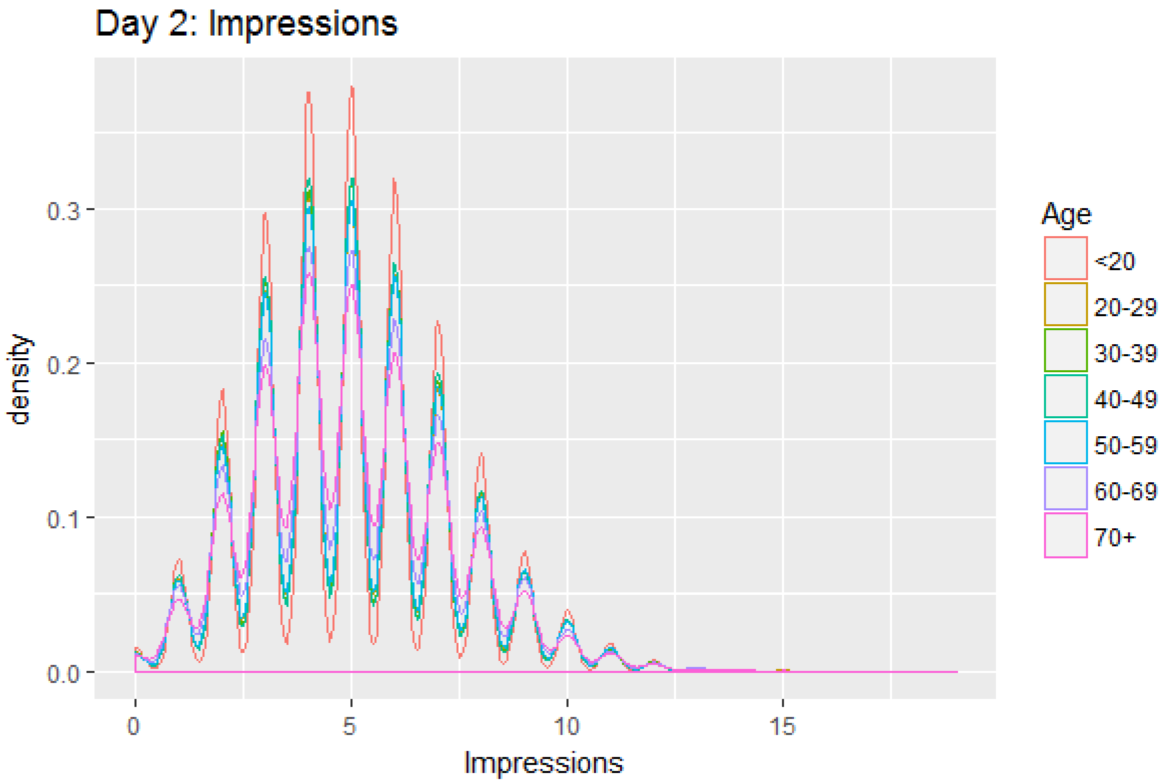
```
# FUNCTION: Plots for impressions
imp <- function(df, day) {
  ggplot(df, aes(x=Impressions, color=Age_Group)) + geom_density() +
    xlab("Impressions") + labs(color="Age") + ggtitle(paste(day, "Impressions", sep=": "))
}

# FUNCTION: Plots for CTR
ctr <- function(df, day, ref) {
  if (ref == "Imp") {
    ref <- "(Impressions > 0)"
  } else if (ref == "Clks") {
    ref <- "(Clicks > 0)"
  }
  ggplot(df, aes(x=Clicks/Impressions, color=Age_Group)) + geom_density() +
    xlab("Click-through-rate (CTR)") + labs(color="Age") +
    ggtitle(paste(day, "CTR", ref, sep=" : "))
}
```

#Plots

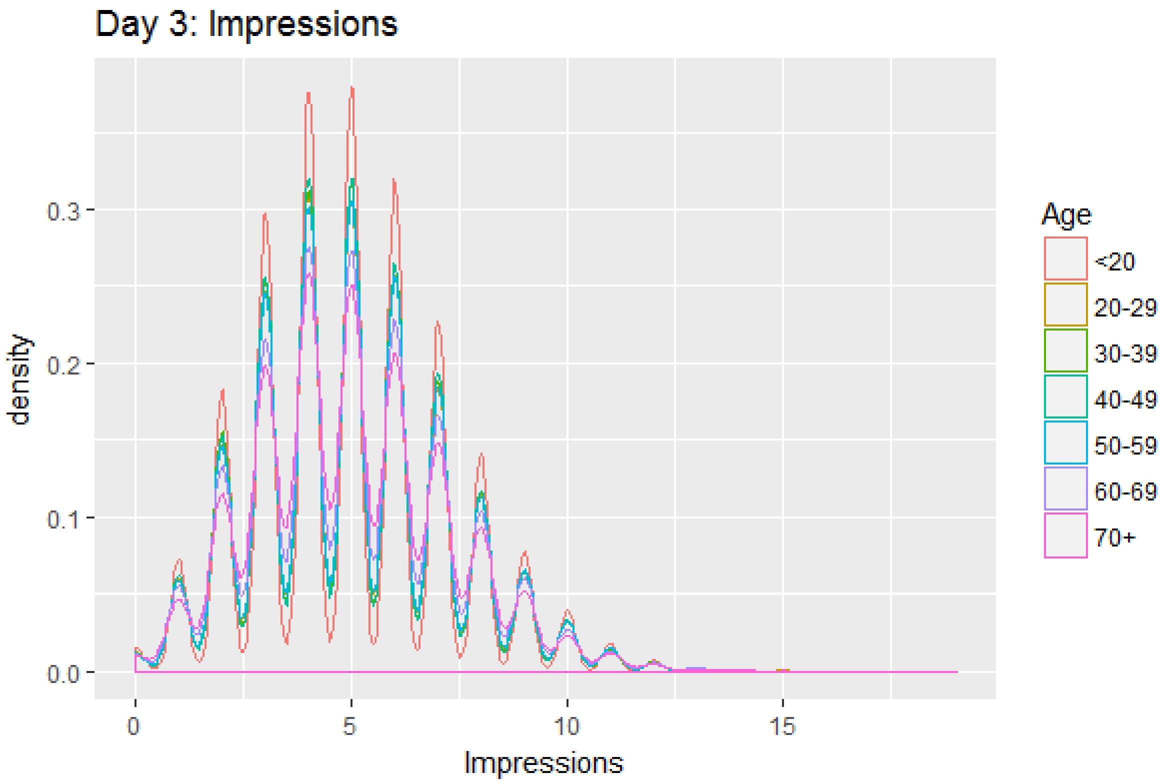
```
imp(day_1, "Day 1")
```



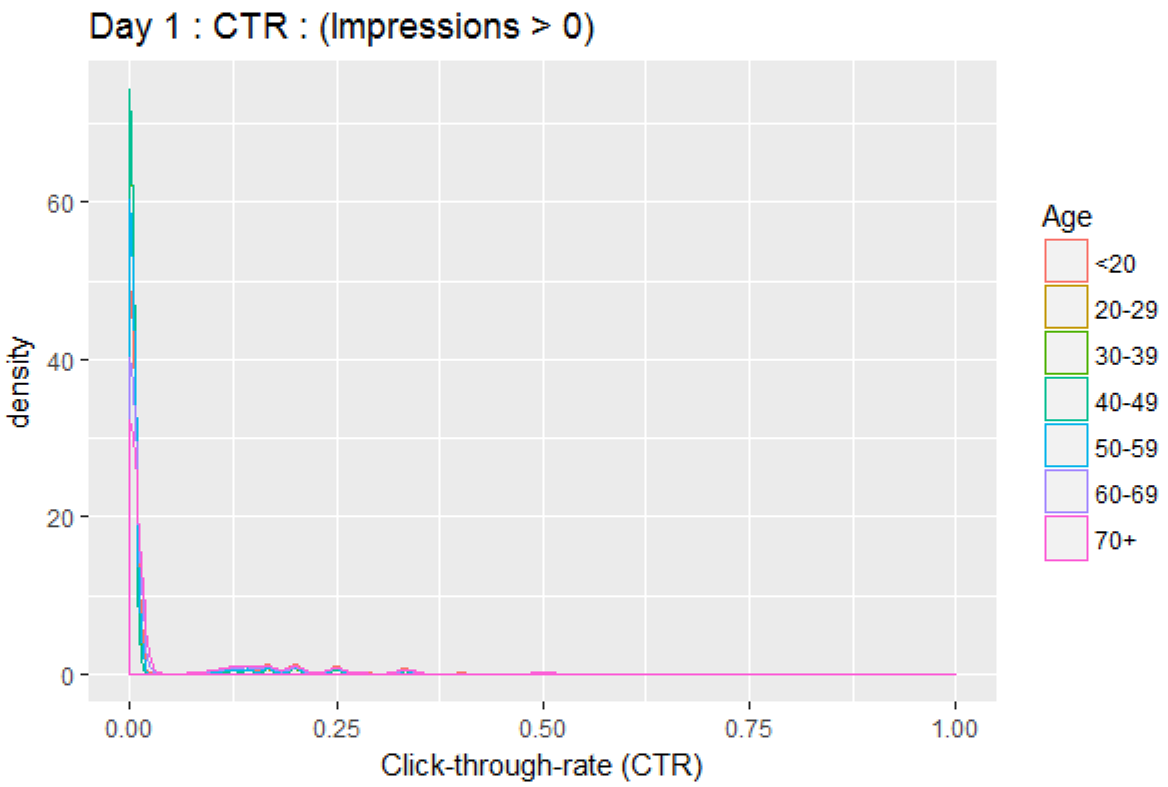
```r
imp(day_2, "Day 2")
```



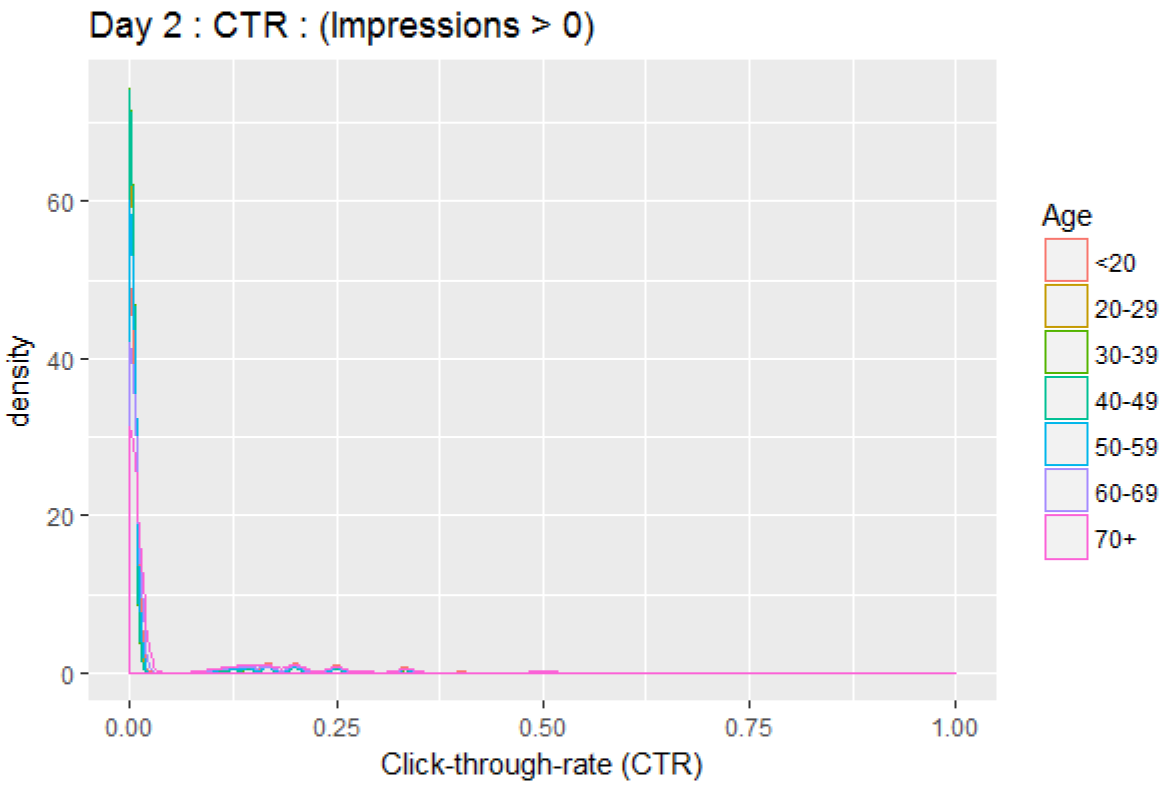
```r
imp(day_2, "Day 3")
```



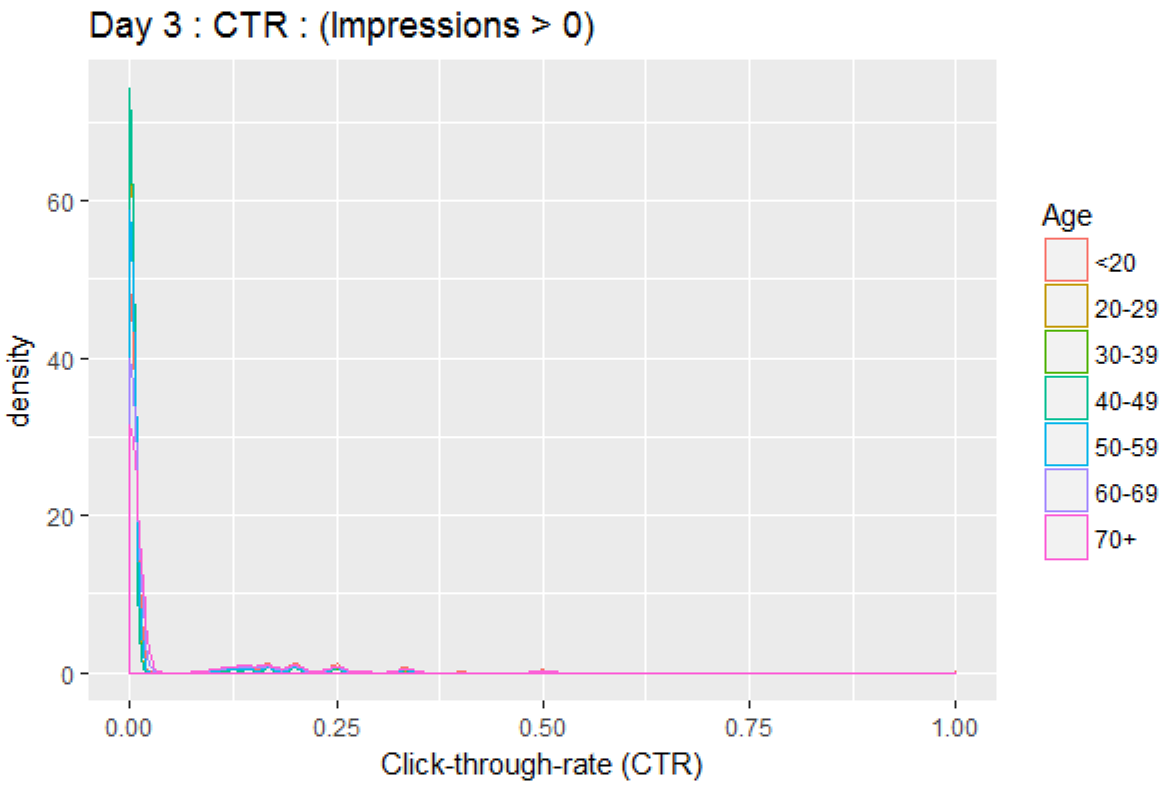
```
ctr(subset(day_1, Impressions > 0), "Day 1", "Imp")
```



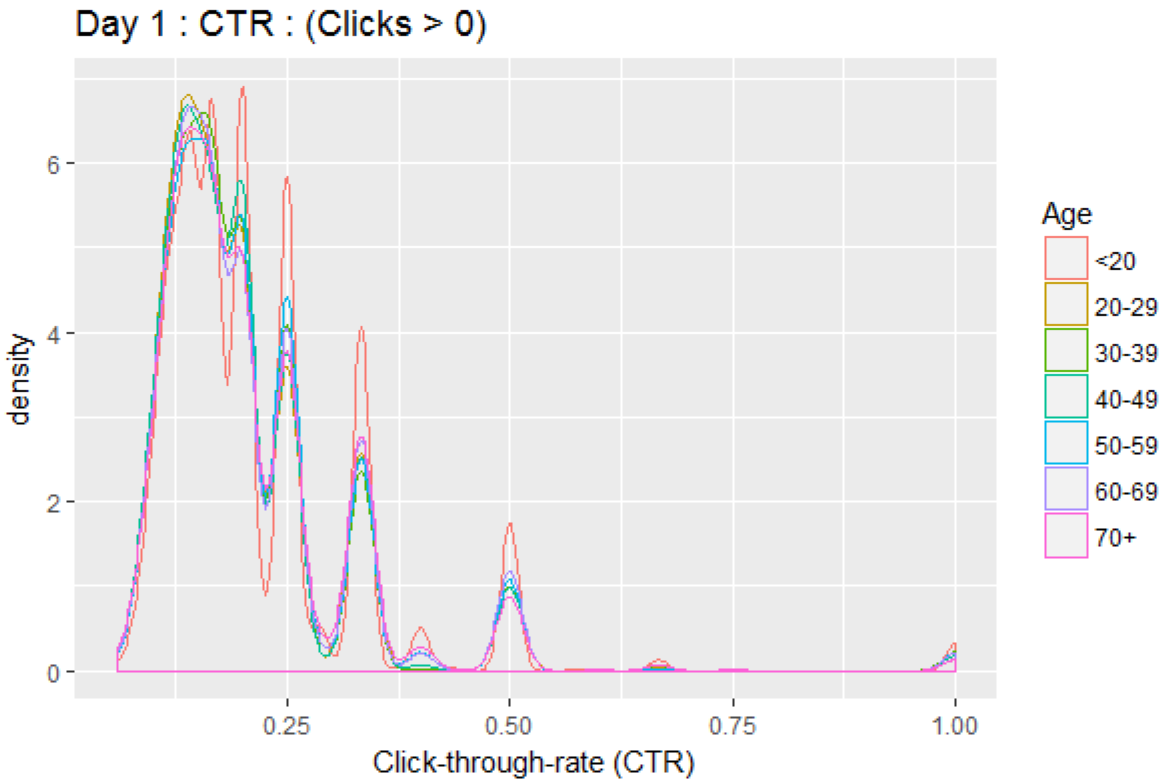
```
ctr(subset(day_2, Impressions > 0), "Day 2", "Imp")
```



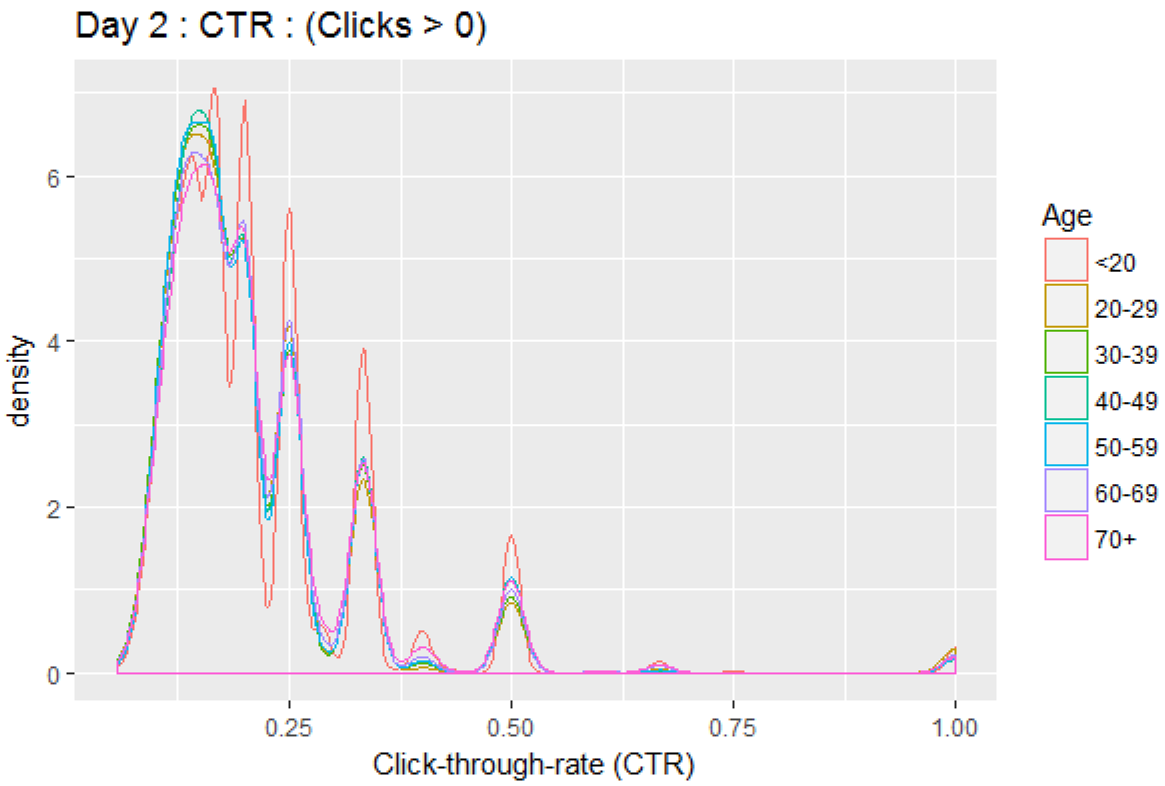
```
ctr(subset(day_3, Impressions > 0), "Day 3", "Imp")
```



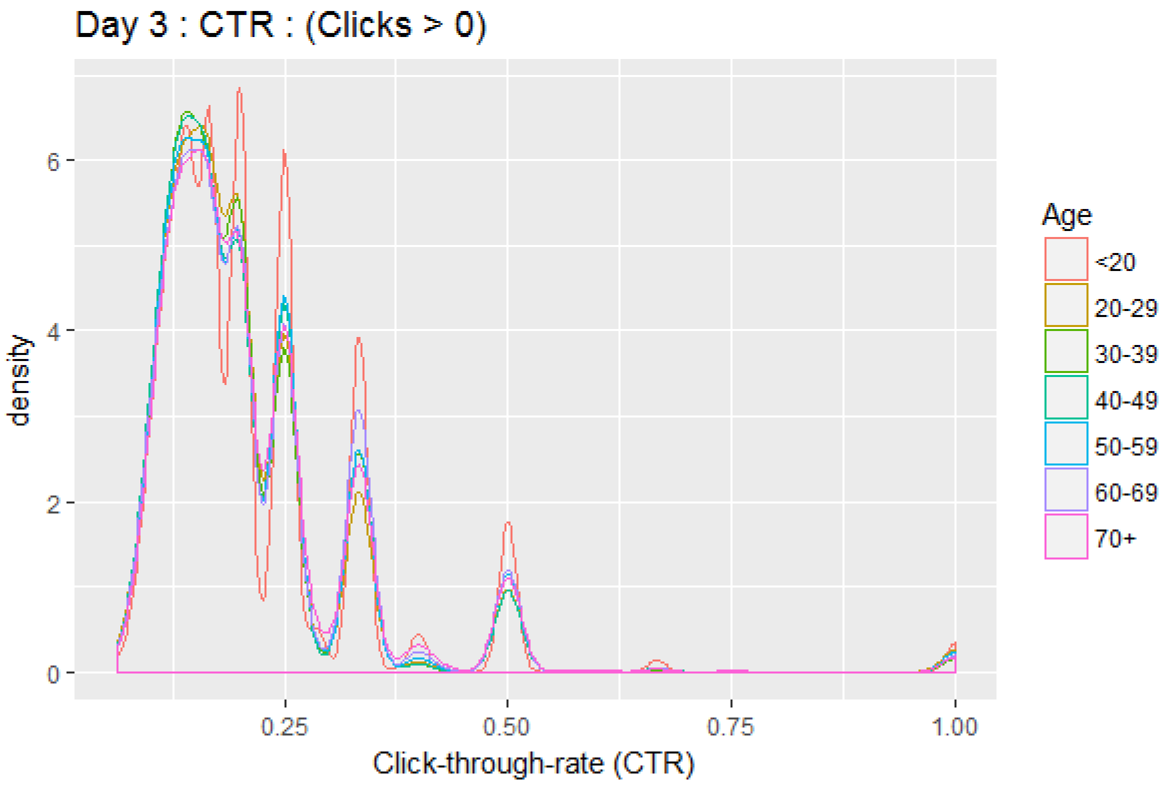
```
ctr(subset(day_1, Clicks > 0), "Day 1", "Clks")
```



```
ctr(subset(day_2, Clicks > 0), "Day 2", "Clks")
```



```
ctr(subset(day_3, Clicks > 0), "Day 3", "Clks")
```



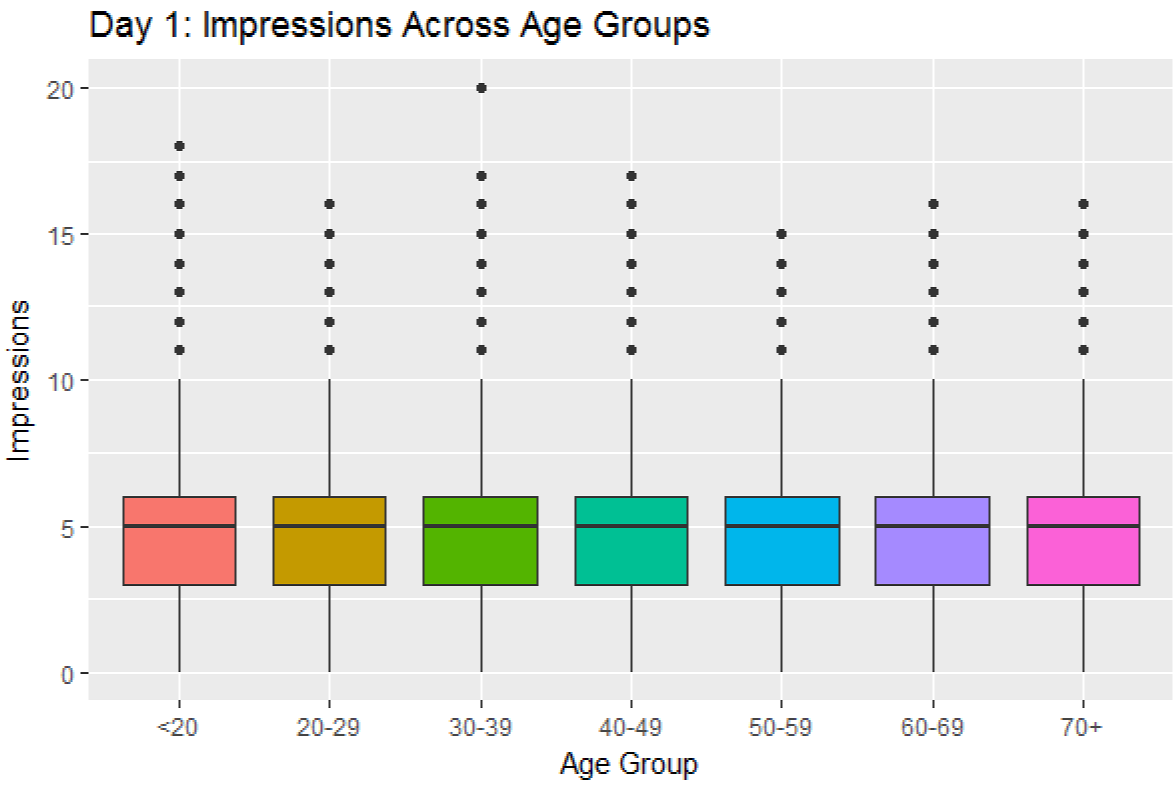
Impressions across Age Groups for each day

#Functions

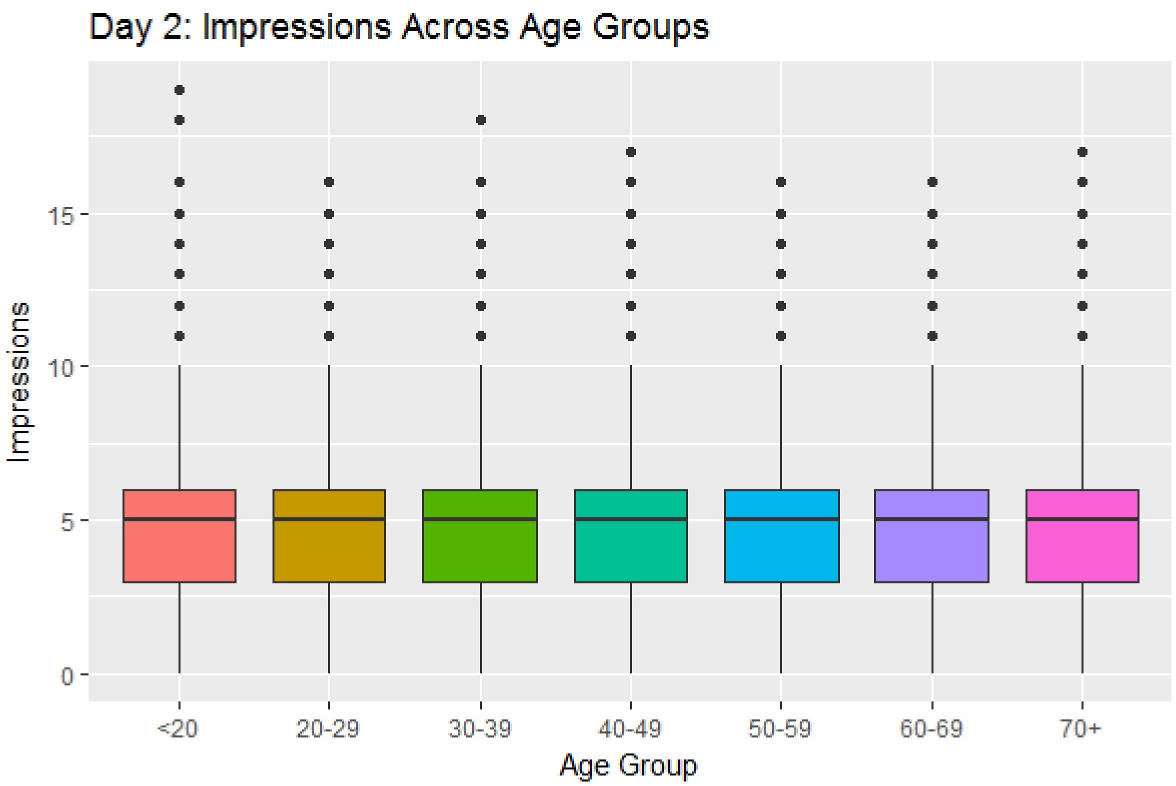
```
age_imp_boxplot <- function (df, day) {
  ggplot(df, aes(x = Age_Group, y = Impressions, fill=Age_Group)) +
    geom_boxplot() +
    xlab("Age Group") + guides(fill=FALSE) +
    ggtitle(paste(day, "Impressions Across Age Groups", sep=": "))
}

age_imp_hist <- function (df, day) {
  ggplot(df, aes(x = Impressions, fill = Age_Group)) +
    geom_histogram(bins = 15) + ggtitle(paste(day, "Impressions Across Age Groups", sep=": "))
}
```

#Plots

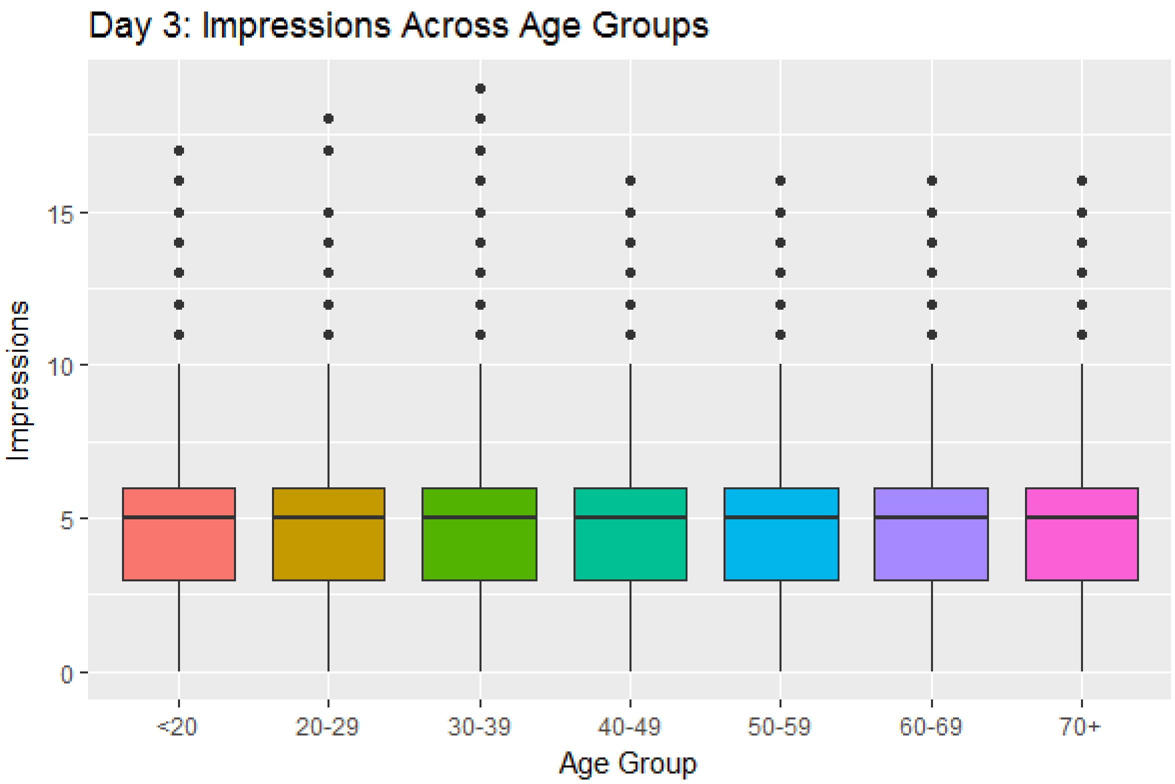
```
age_imp_boxplot(day_1, "Day 1")
```



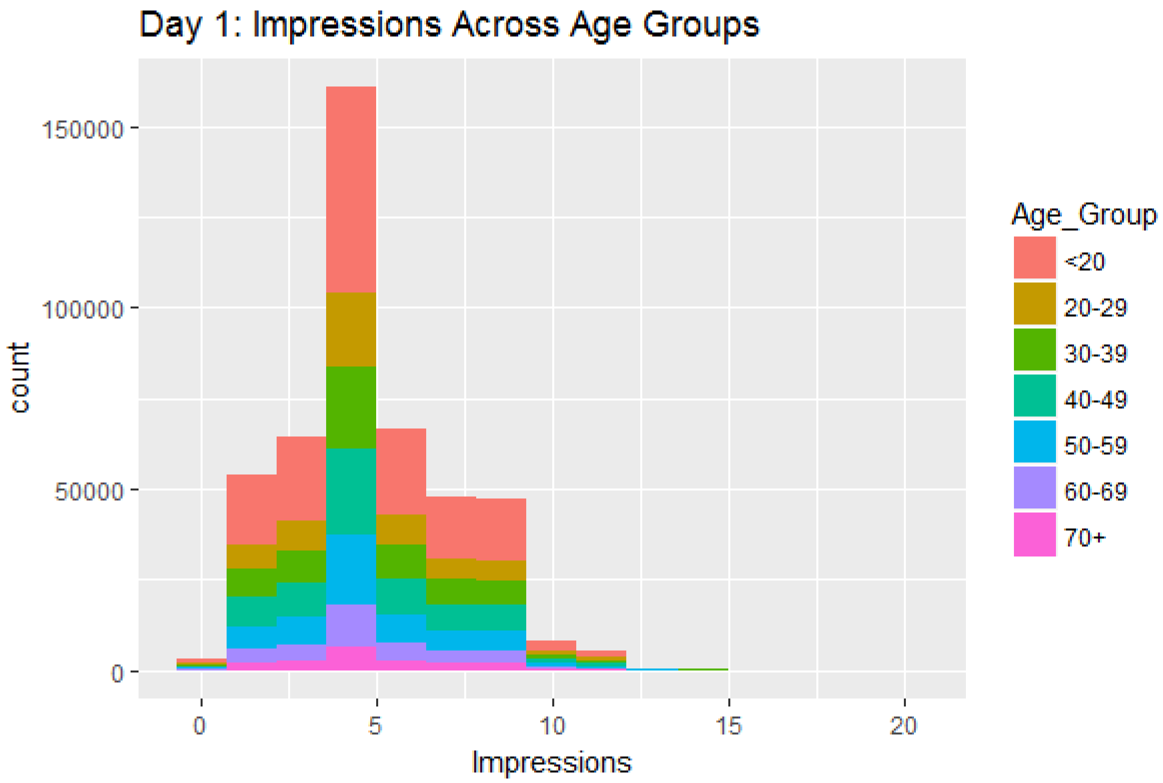
```r
age_imp_boxplot(day_2, "Day 2")
```



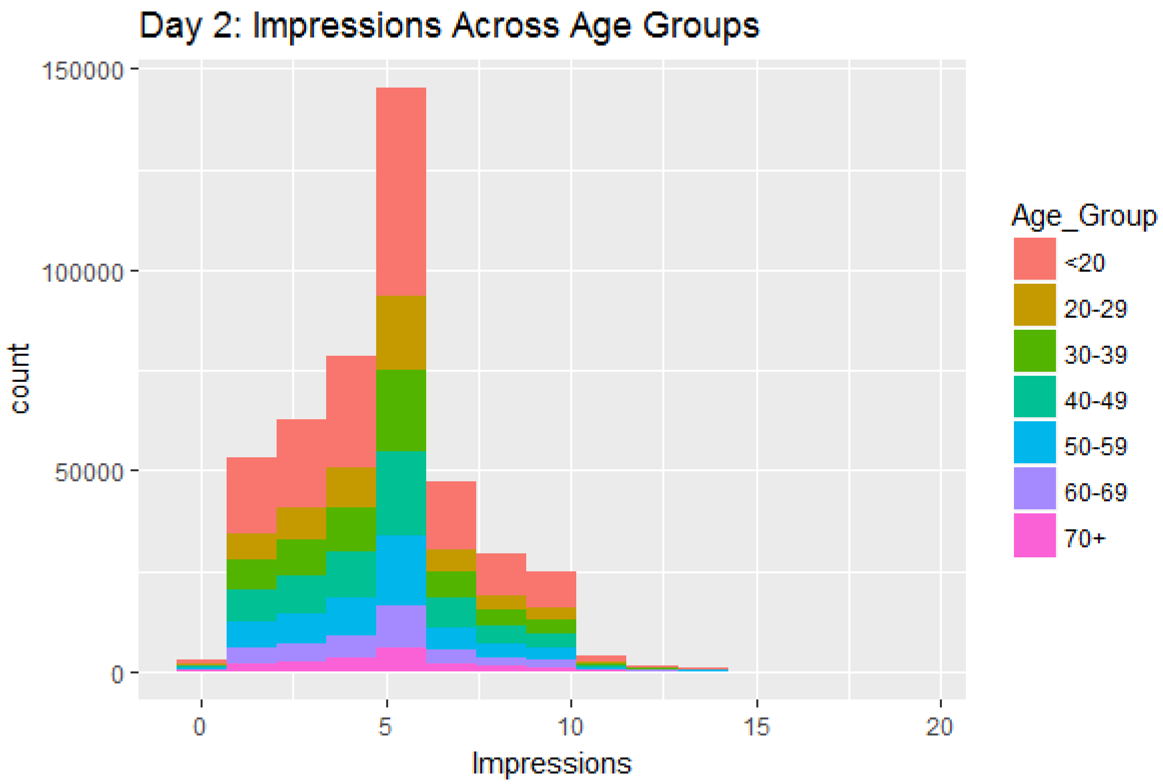
```r
age_imp_boxplot(day_3, "Day 3")
```



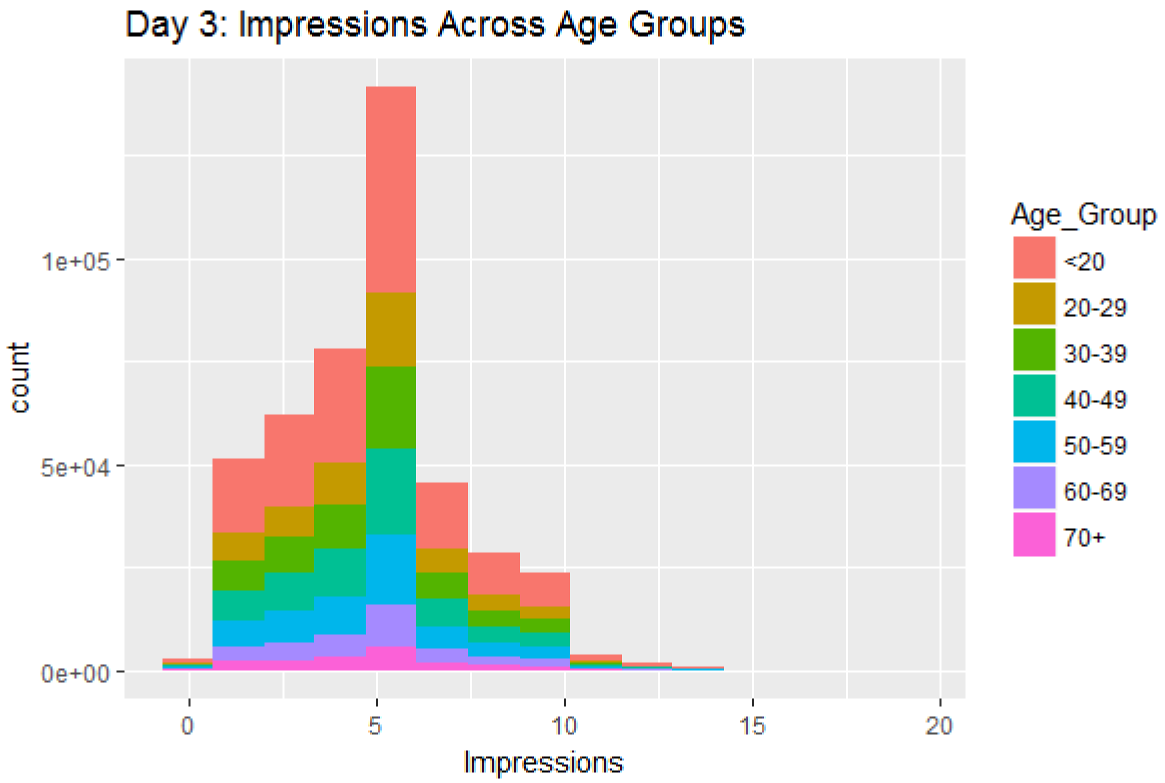
```
age_imp_hist(day_1, "Day 1")
```



```
age_imp_hist(day_2, "Day 2")
```



```r
age_imp_hist(day_3, "Day 3")
```



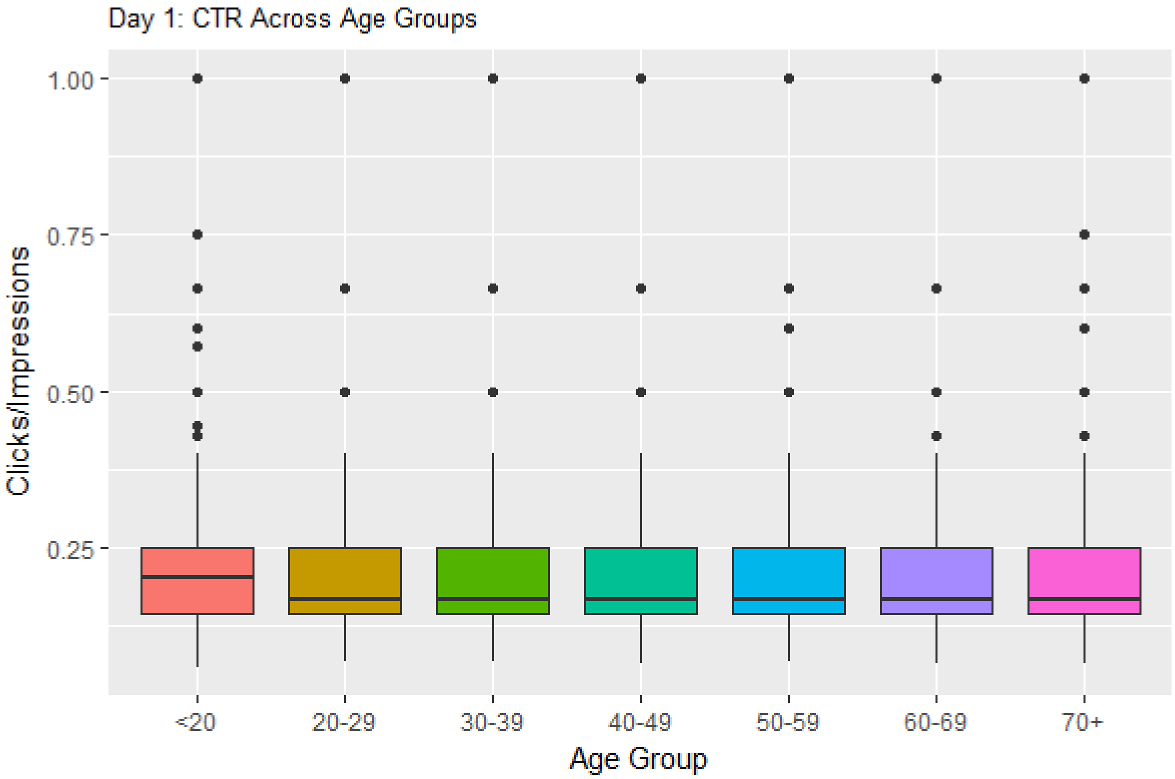
CTR across Age Groups for each day

#Functions

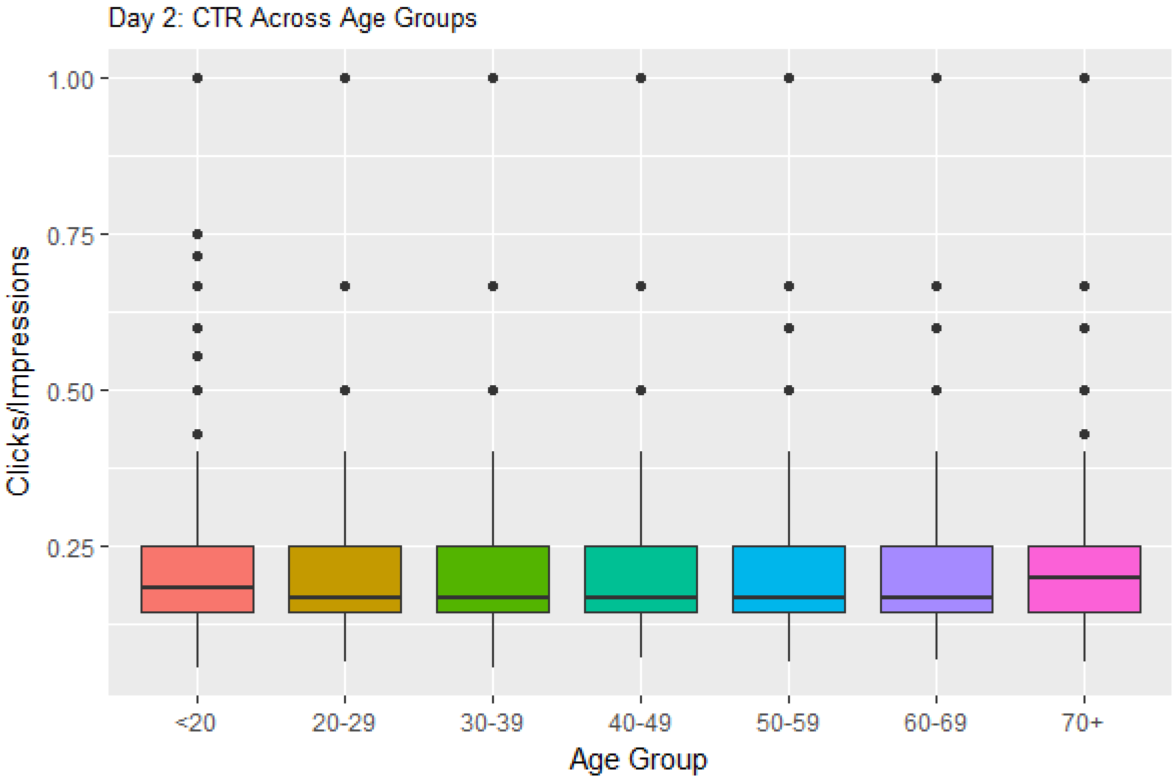
```r
# Click Through Rate across Age Group
age_ctr_boxplot <- function (df, day) {
  ggplot(df, aes(x = Age_Group, y = Clicks/Impressions, fill=Age_Group)) +
    geom_boxplot() +
    xlab("Age Group") + guides(fill=FALSE) +
    ggtitle(paste(day, "CTR Across Age Groups", sep=": "))
}

age_ctr_hist <- function (df, day) {
  ggplot(df, aes(x = Clicks/Impressions, fill = Age_Group)) +
    geom_histogram(bins = 15) + ggtitle(paste(day, "CTR Across Age Groups", sep=": "))
}
```

#Plots

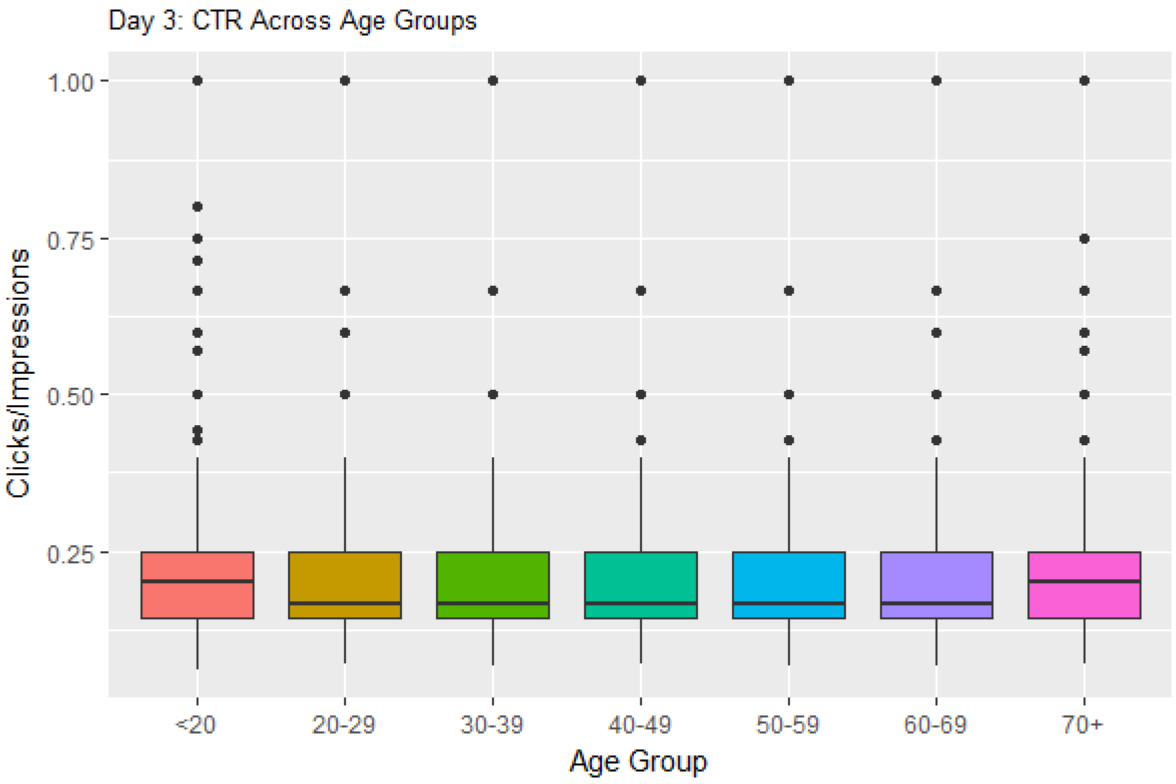
```
age_ctr_boxplot(subset(day_1, Clicks > 0 & Impressions > 0), "Day 1")
```



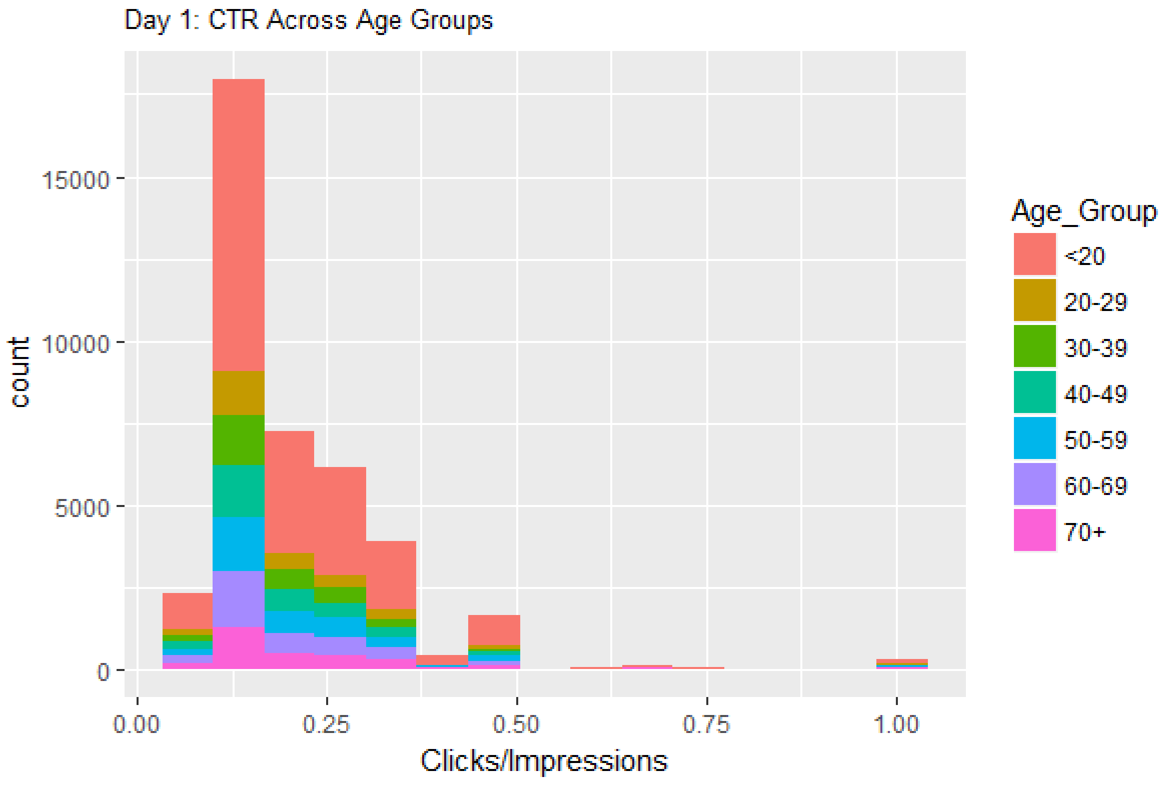
```
age_ctr_boxplot(subset(day_2, Clicks > 0 & Impressions > 0), "Day 2")
```



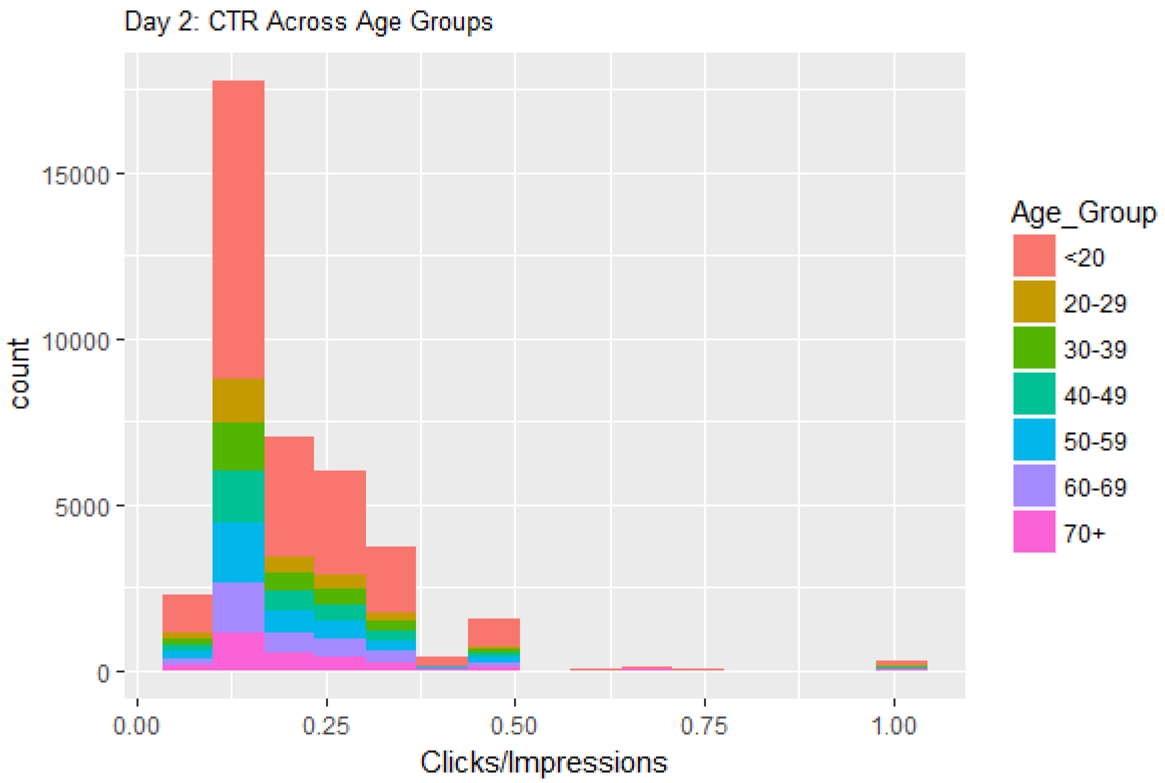
```
age_ctr_boxplot(subset(day_3, Clicks > 0 & Impressions > 0), "Day 3")
```



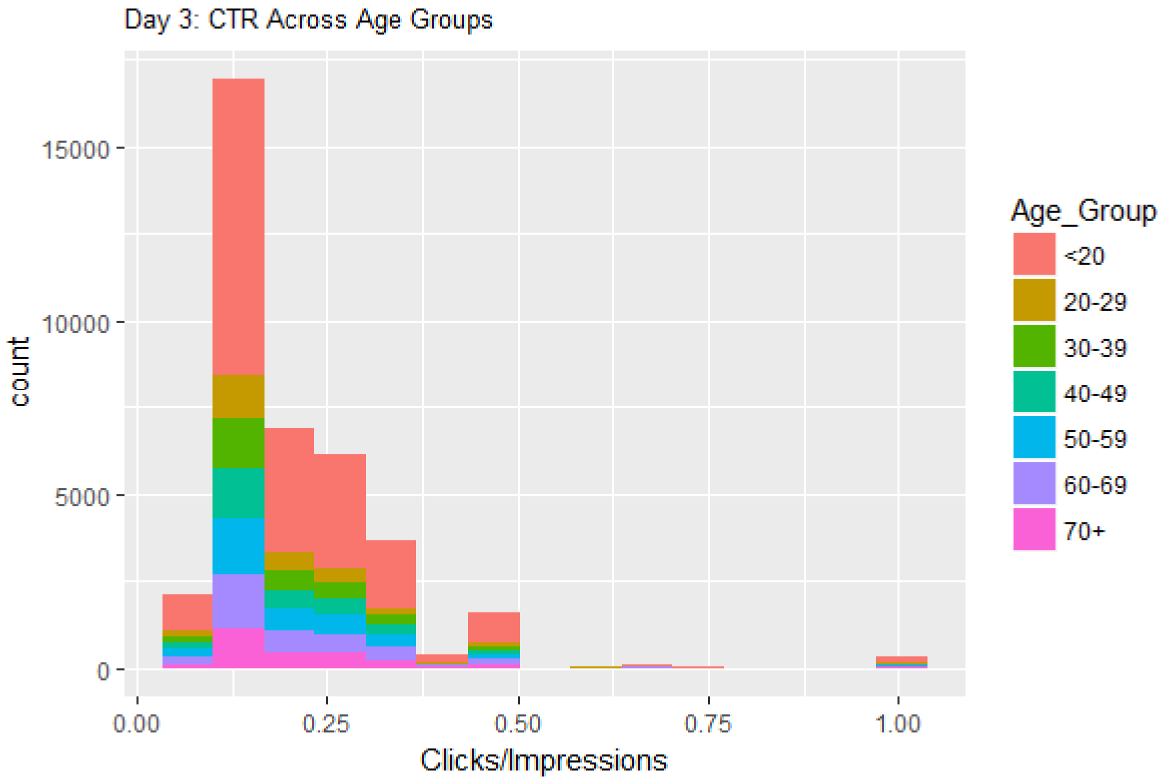
```
age_ctr_hist(subset(day_1, Clicks > 0 & Impressions > 0), "Day 1")
```



```r
age_ctr_hist(subset(day_2, Clicks > 0 & Impressions > 0), "Day 2")
```



```r
age_ctr_hist(subset(day_3, Clicks > 0 & Impressions > 0), "Day 3")
```

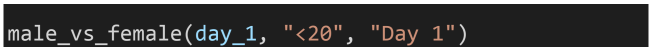


Comparisons across user segments/demographics

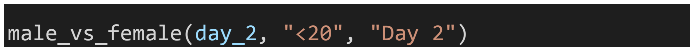
#Functions

```
# Male v/s Female
male_vs_female <- function (df, age_grp, day) {
  df <- subset(df, Age_Group == age_grp)

  ggplot(df, aes(x = Gender, y = ..count..)) +
    geom_bar(aes(colour = "black", fill = Gender)) +
    geom_text(stat='count', aes(label=..count..),vjust=2) +
    xlab("Gender") +
    ggtitle(paste(day, "Age Group (<20 Age)", sep = " : "))
}

# SignedIn v/s Not SignedIn
loggedin_vs_not <- function (df, day) {
  ggplot(df, aes(x = Signed_In, y = ..count../1000)) +
    geom_bar(aes(colour = "black", fill = Signed_In)) +
    geom_text(stat='count', aes(label=..count..), vjust=2) +
    ylab("Total (in thousands)") +
    xlab("Day 1: Logged In Stats") +
    ggtitle(paste(day, "Signed In Status", sep = " : "))
}

# Clicked v/s not clicked
clicked_vs_not <- function (df, day) {
  df$Have_Clicked[df$Clicks > 0] <- "Clicked"
  df$Have_Clicked[df$Clicks == 0] <- "Not Clicked"

  ggplot(df, aes(x = Have_Clicked, y = ..count../1000)) +
    geom_bar(aes(colour = "black", fill = Have_Clicked)) +
    geom_text(stat='count', aes(label=..count..),vjust=1.5) +
    ylab("Total (in thousands)") +
    xlab("Clicks") +
    ggtitle(paste(day, "Clicks Status", sep = " : "))
}
```

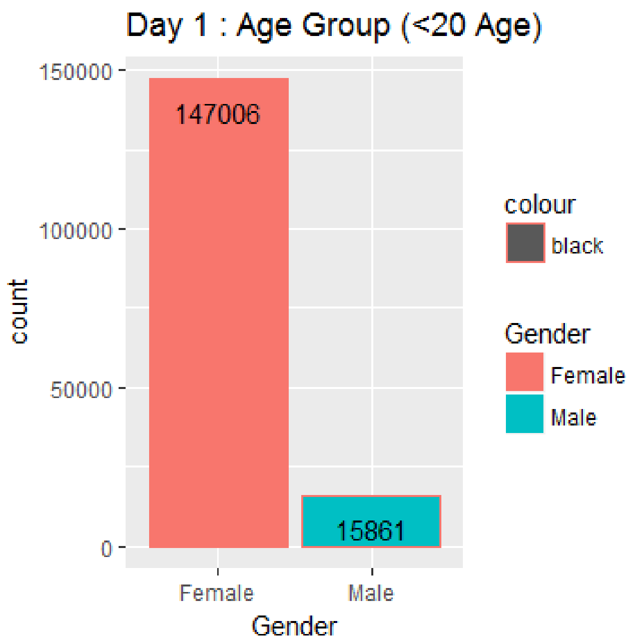
#Plots

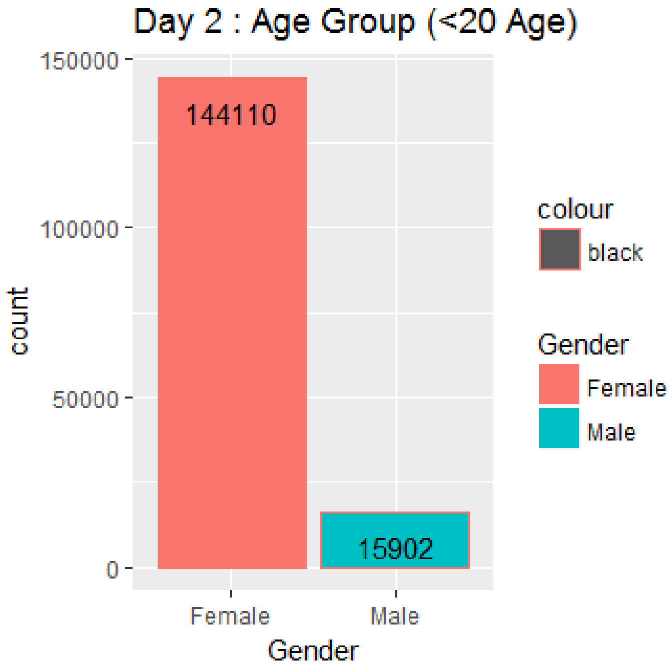


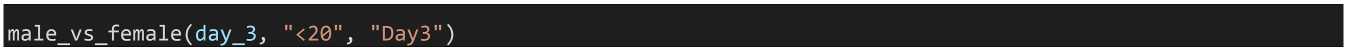
```
male_vs_female(day_1, "<20", "Day 1")
```



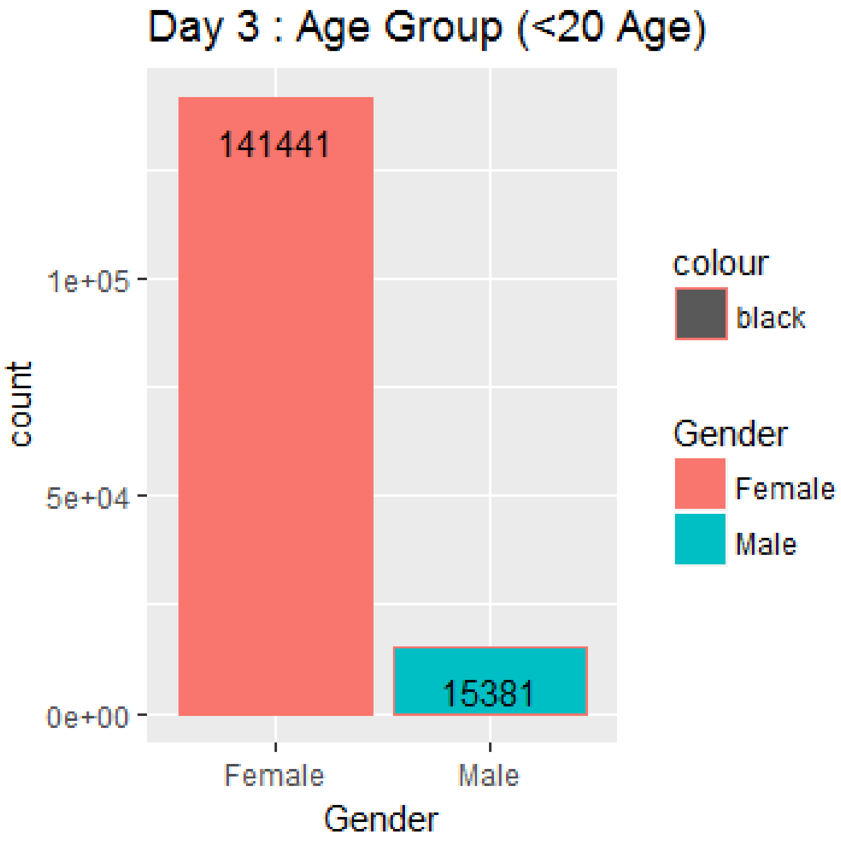
```
male_vs_female(day_2, "<20", "Day 2")
```





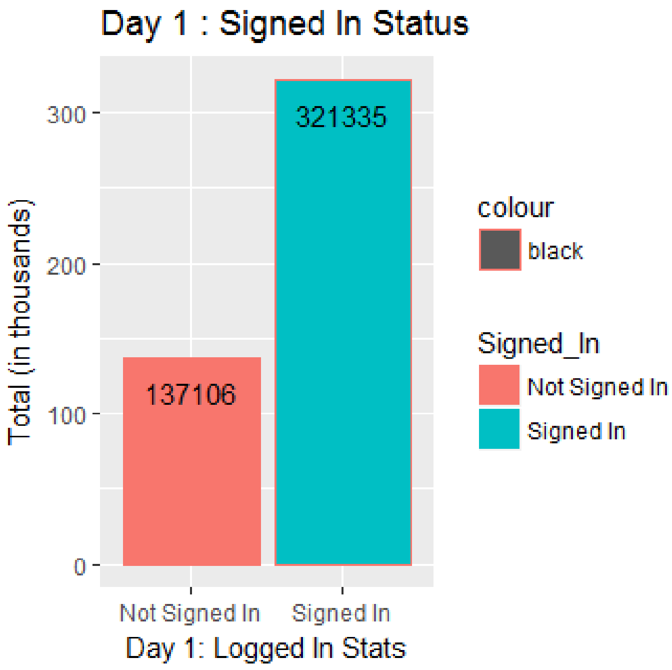


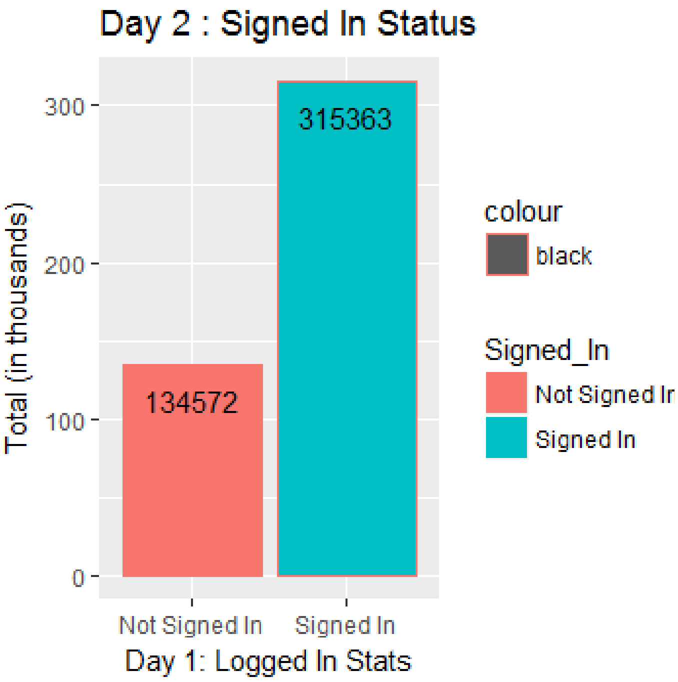
```
male_vs_female(day_3, "<20", "Day3")
```



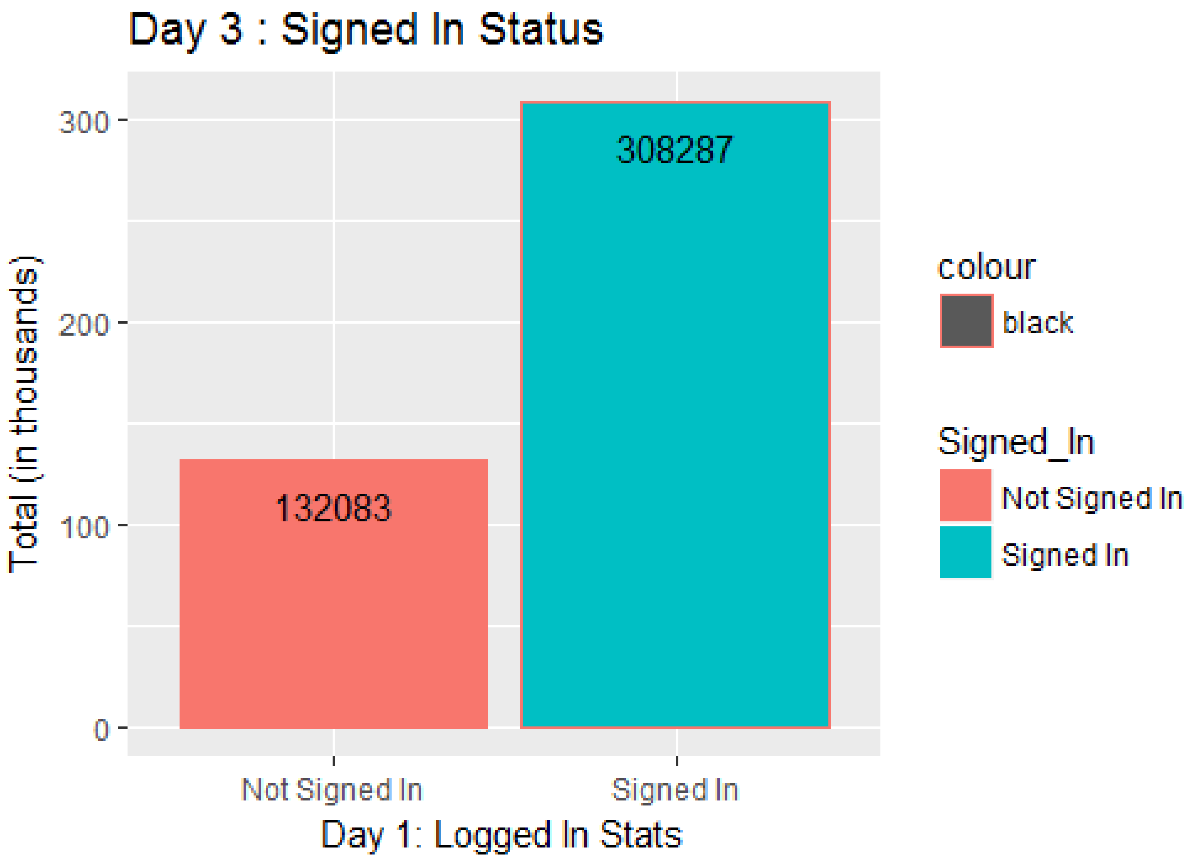
```
loggedin_vs_not(day_1, "Day 1")
```

```
loggedin_vs_not(day_2, "Day 2")
```

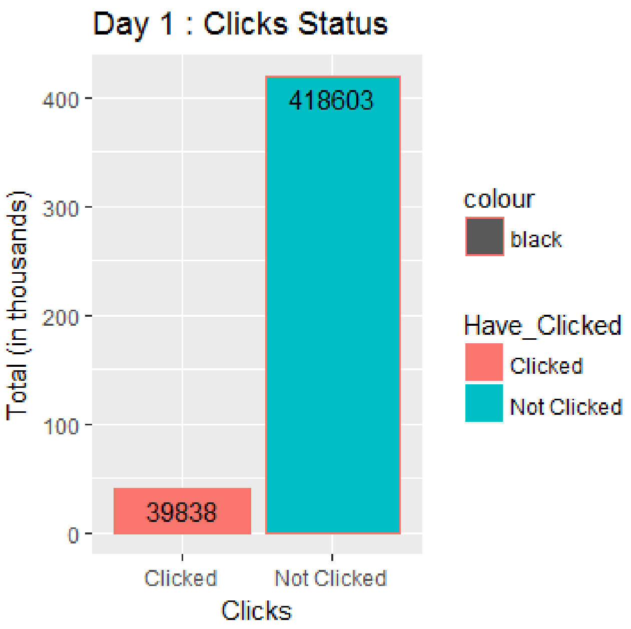




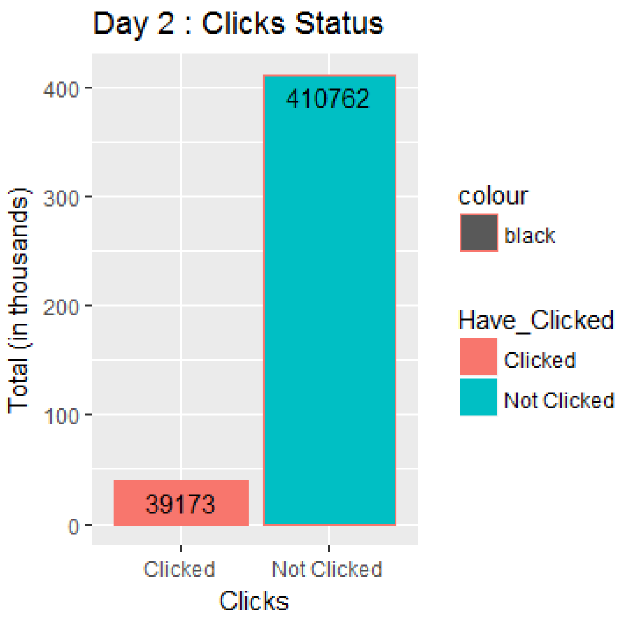
```
loggedin_vs_not(day_3, "Day 3")
```



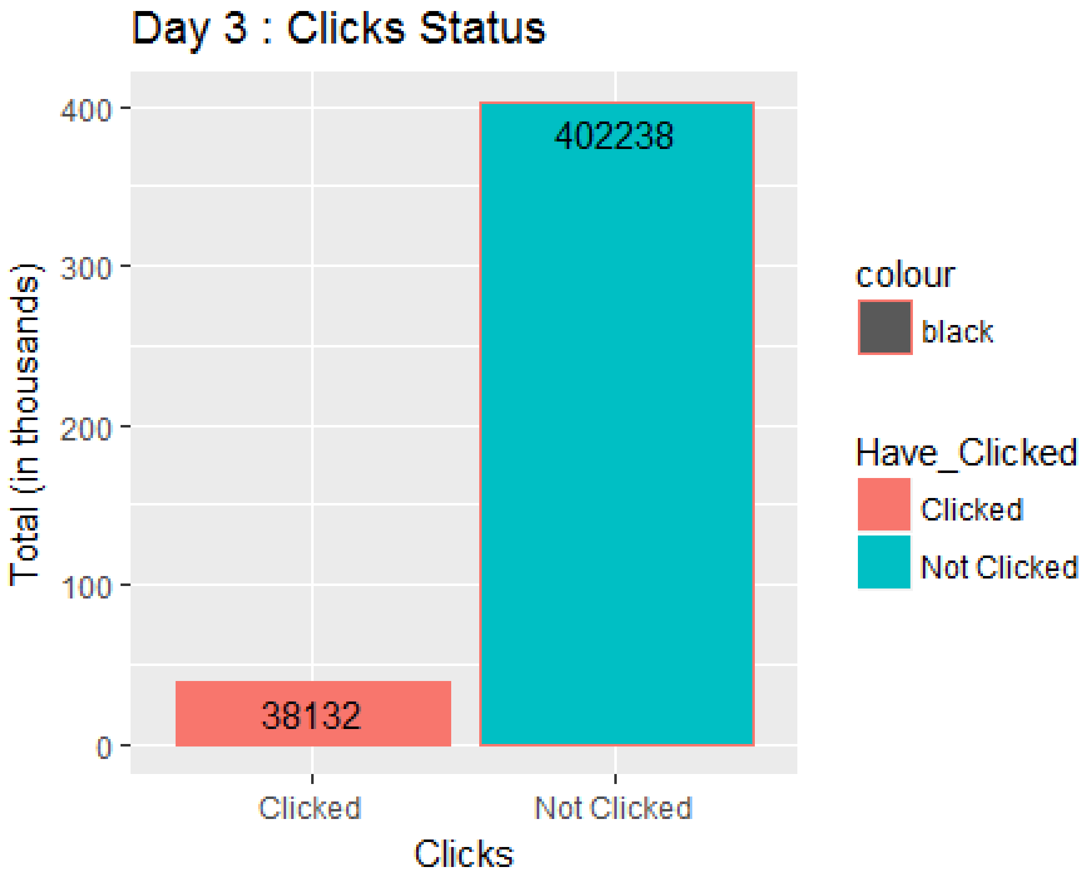
```r
clicked_vs_not(day_1, "Day 1")
```



```r
clicked_vs_not(day_2, "Day 2")
```



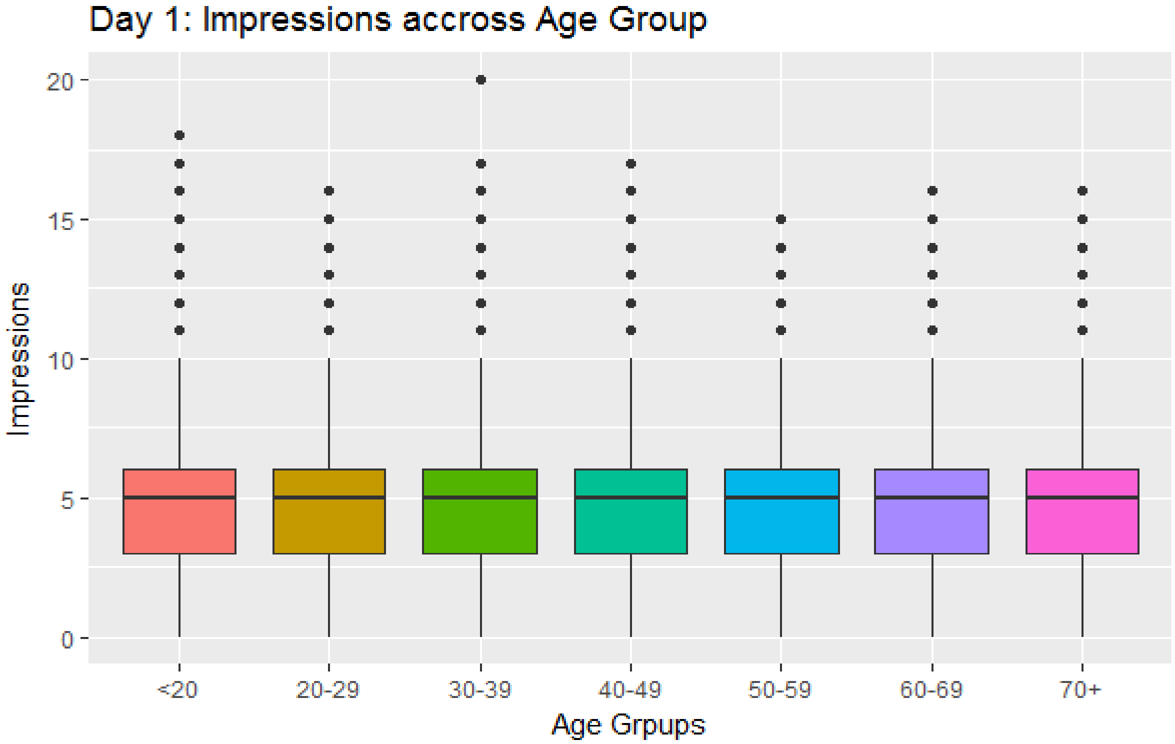
```r
clicked_vs_not(day_3, "Day 3")
```



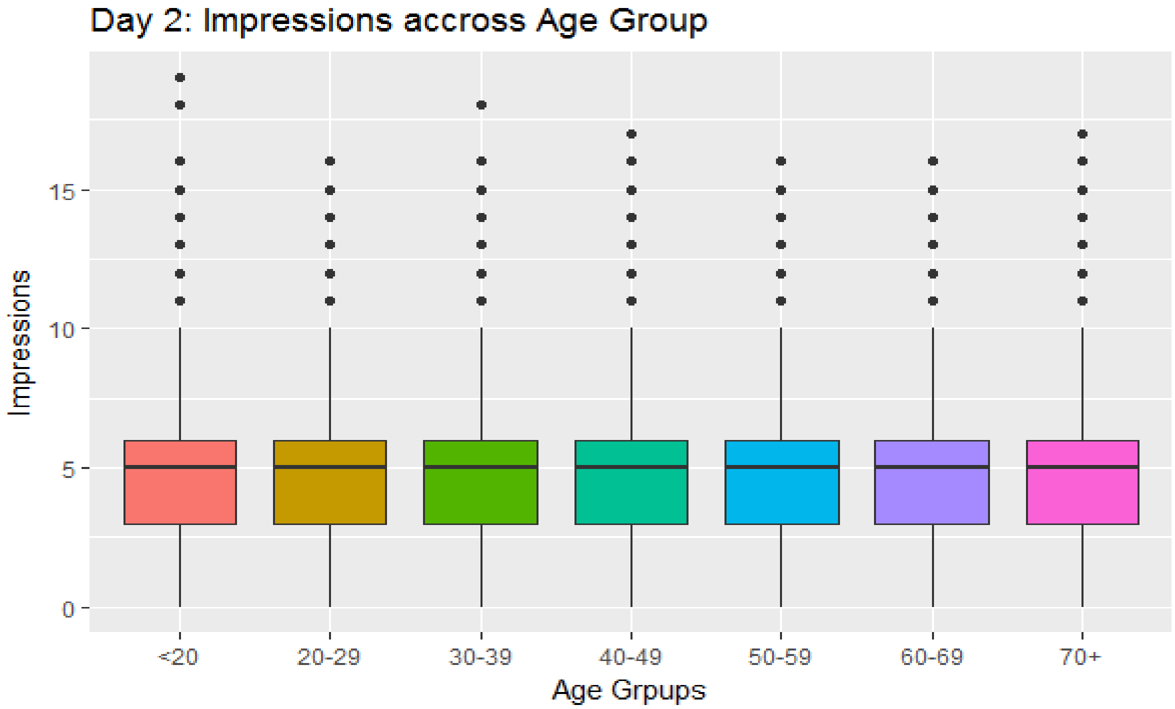
Metrics and Distribution Analysis

#IMPRESSION ACROSS AGE GROUPS
#Plots

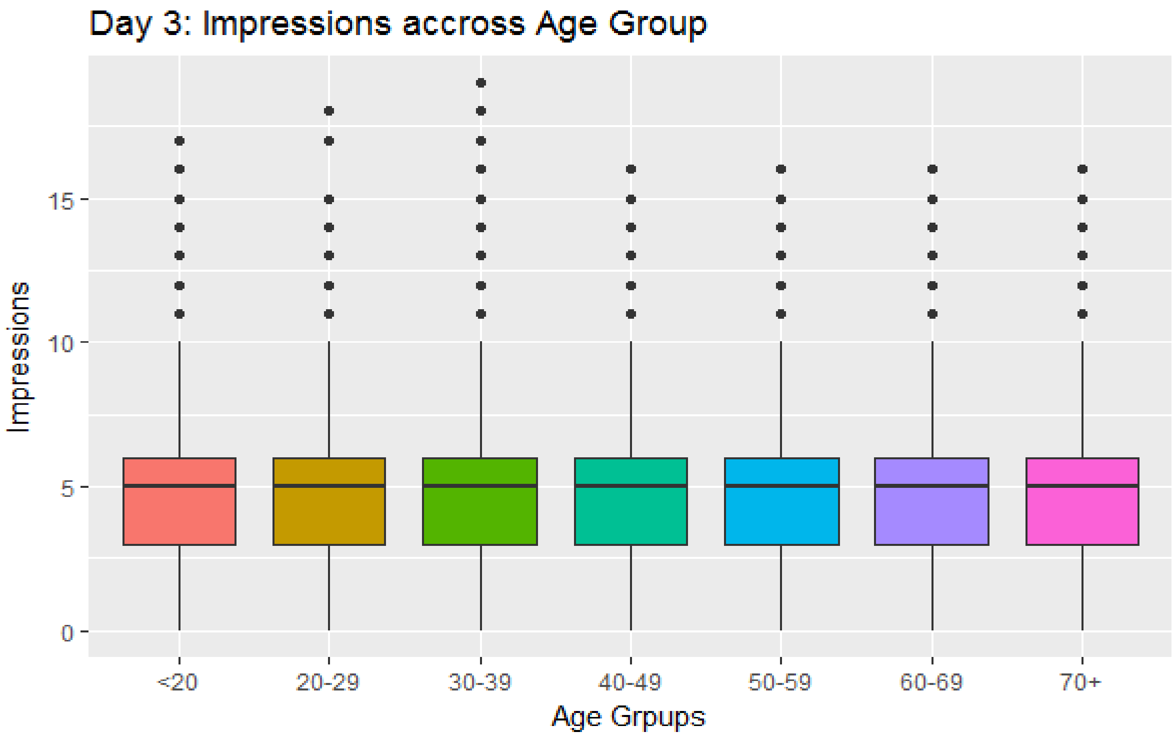
```
ggplot(day_1,aes(x=Age_Group, y=Impressions, fill=Age_Group)) + geom_boxplot() +
  xlab("Age Grpups") + guides(fill=FALSE) + ggtitle("Day 1: Impressions accross Age Group")
```



```
ggplot(day_2,aes(x=Age_Group, y=Impressions, fill=Age_Group)) + geom_boxplot() +
  xlab("Age Grpups") + guides(fill=FALSE) + ggtitle("Day 2: Impressions accross Age Group")
```

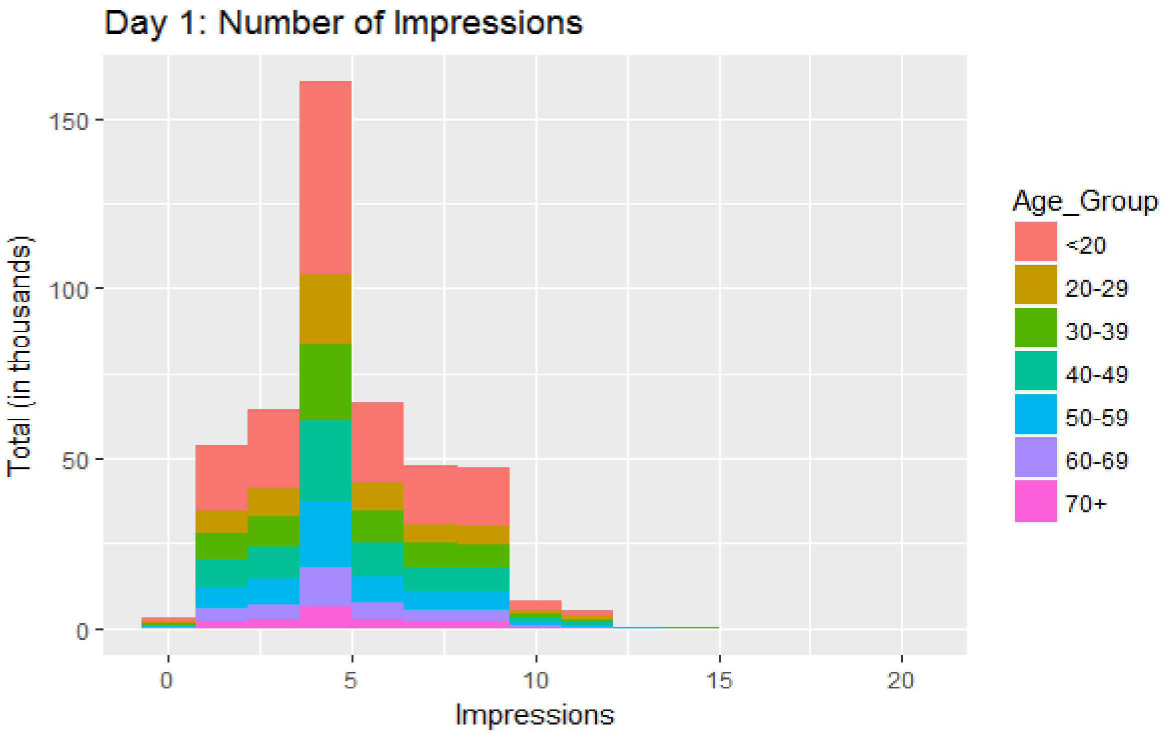


```
ggplot(day_3,aes(x=Age_Group, y=Impressions, fill=Age_Group)) + geom_boxplot() + xlab("Age Grpups") +
  guides(fill=FALSE) + ggtitle("Day 3: Impressions accross Age Group")
```

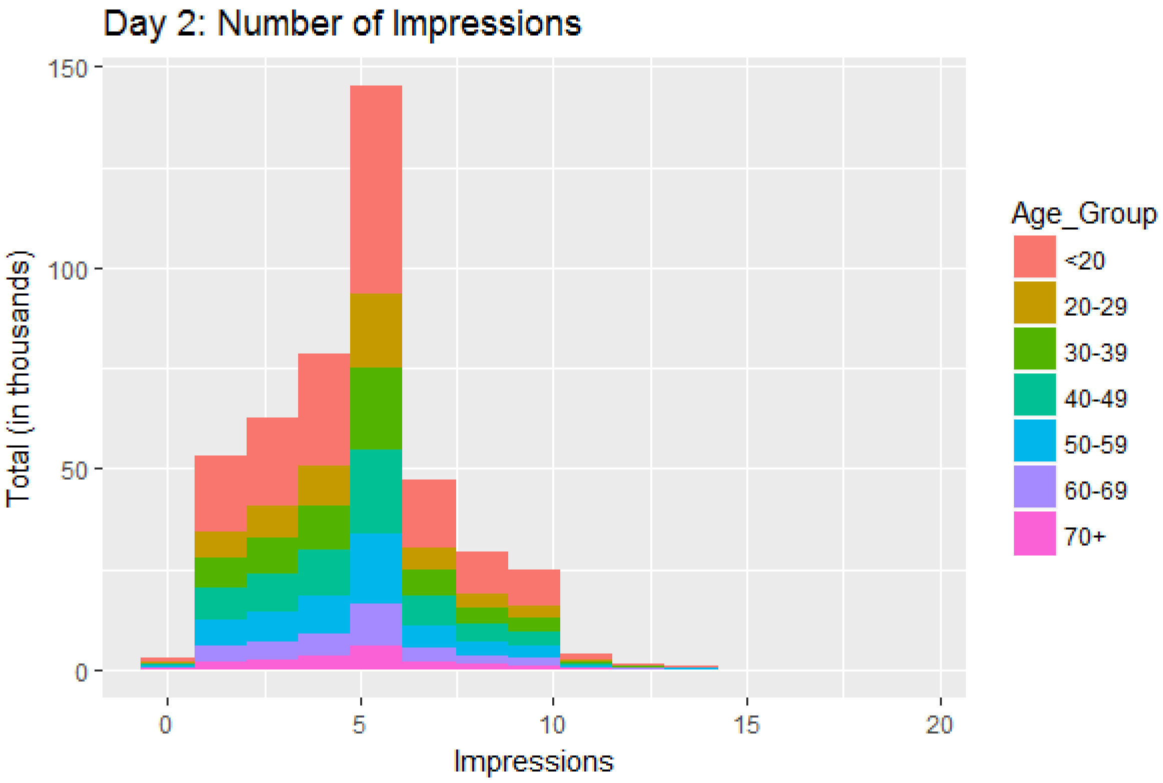


```
#NUMBER OF IMPRESSIONS ACROSS AGE GROUPS
#Plots
```

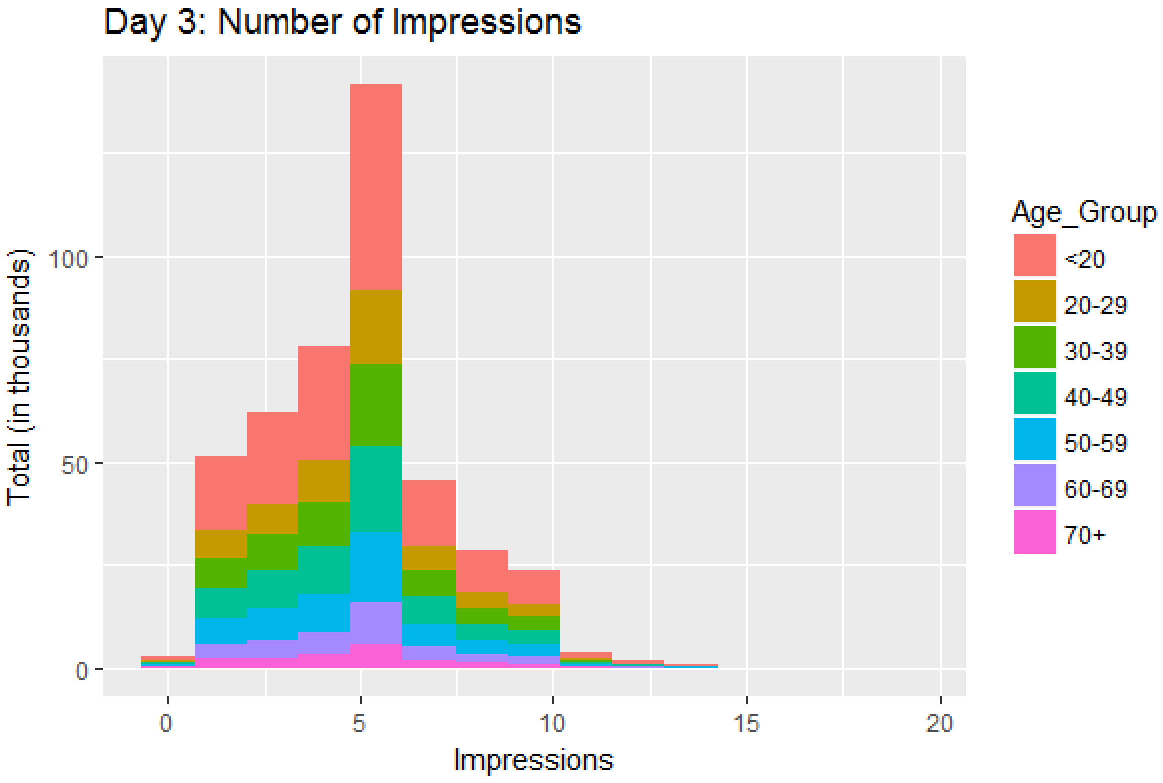
```
ggplot(day_1, aes(x=Impressions, y = ..count../1000, fill=Age_Group)) +
  ylab("Total (in thousands)") + geom_histogram(bins = 15) + ggtitle("Day 1: Number of Impressions")
```



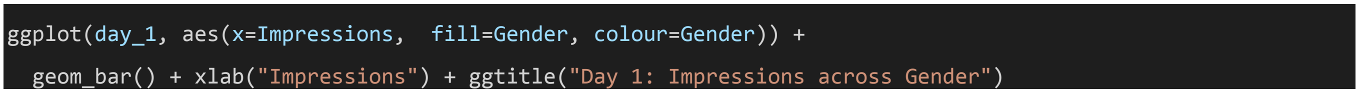
```
ggplot(day_2, aes(x=Impressions, y = ..count../1000, fill=Age_Group)) +
  ylab("Total (in thousands)") + geom_histogram(bins = 15) + ggtitle("Day 2: Number of Impressions")
```



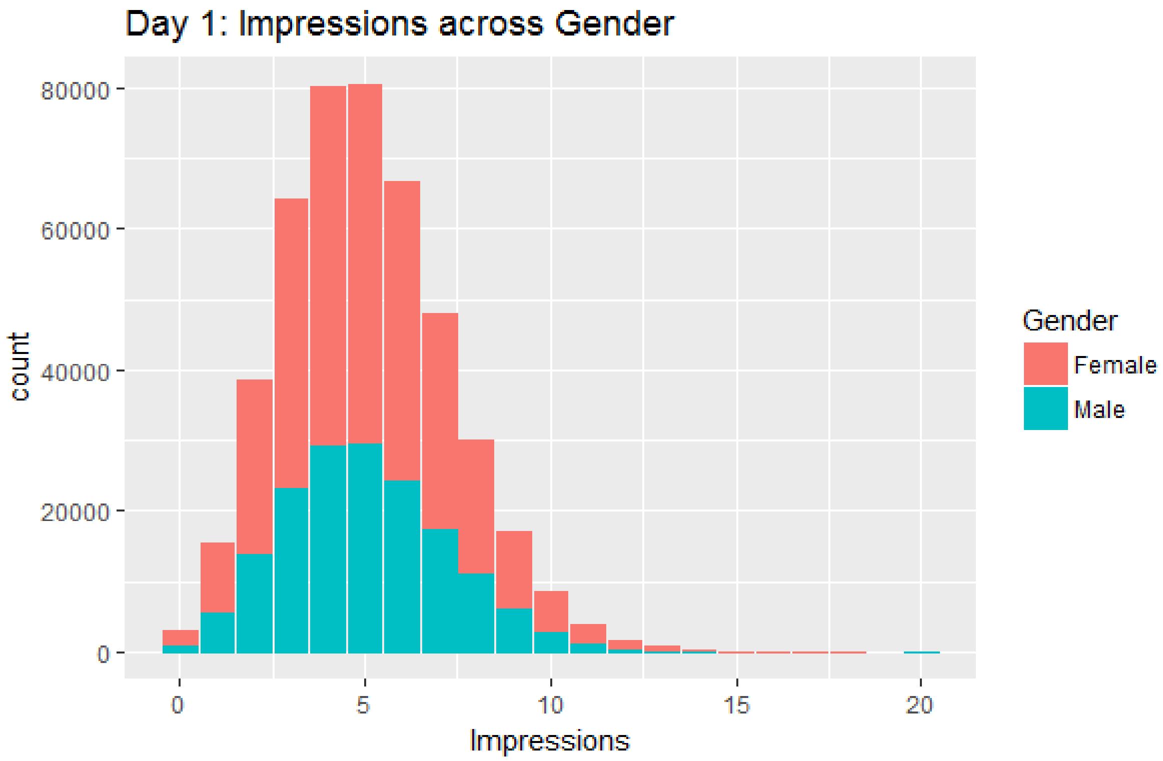
```
ggplot(day_3, aes(x=Impressions, y = ..count../1000, fill=Age_Group)) +
  ylab("Total (in thousands)") + geom_histogram(bins = 15) + ggtitle("Day 3: Number of Impressions")
```

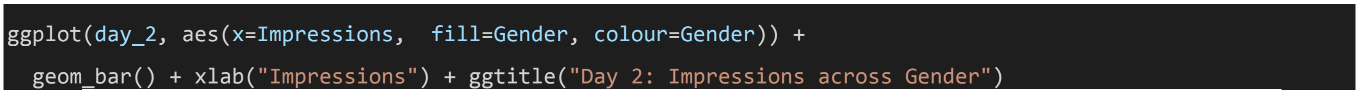


#NUMBER OF IMPRESSIONS ACROSS GENDER
#Plots

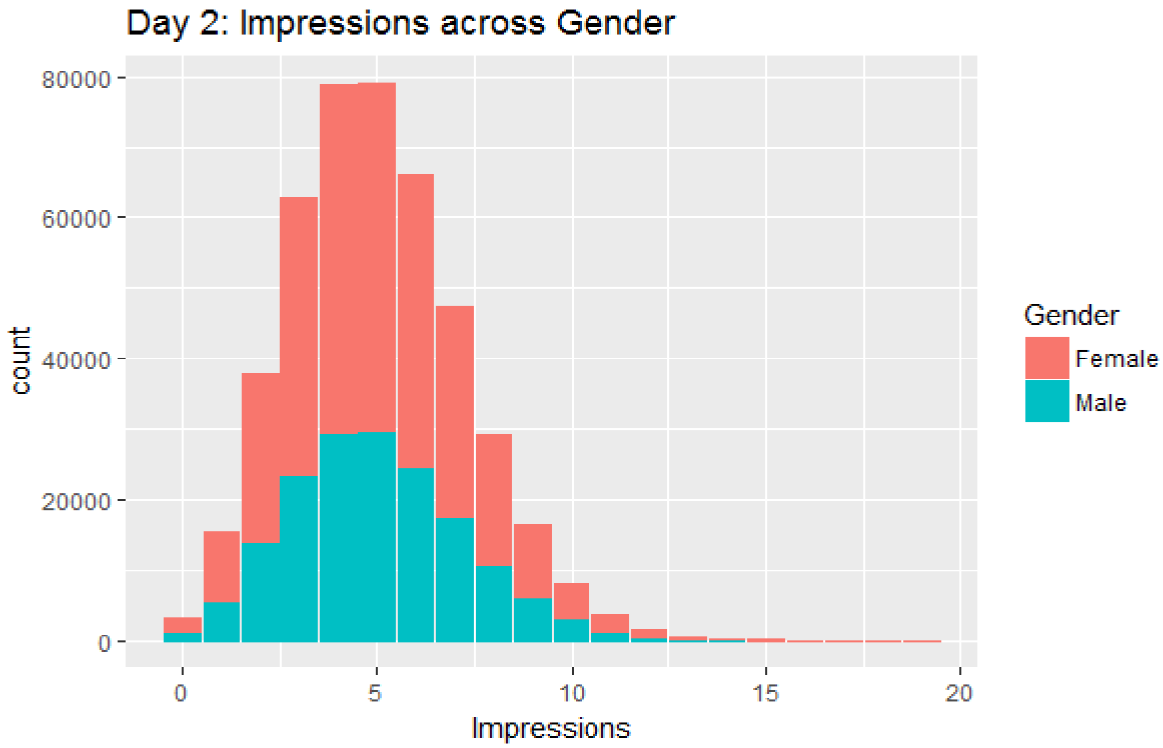


```
ggplot(day_1, aes(x=Impressions,  fill=Gender, colour=Gender)) +
  geom_bar() + xlab("Impressions") + ggtitle("Day 1: Impressions across Gender")
```

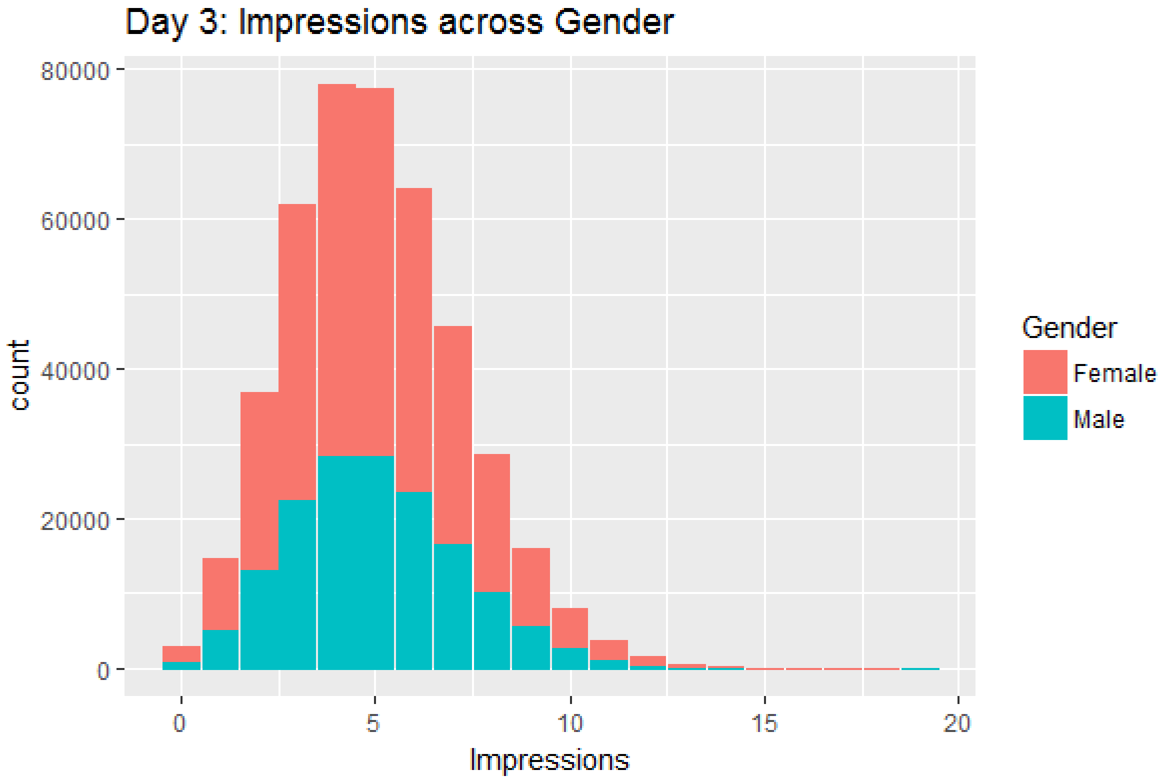




```
ggplot(day_2, aes(x=Impressions,  fill=Gender, colour=Gender)) +
  geom_bar() + xlab("Impressions") + ggtitle("Day 2: Impressions across Gender")
```

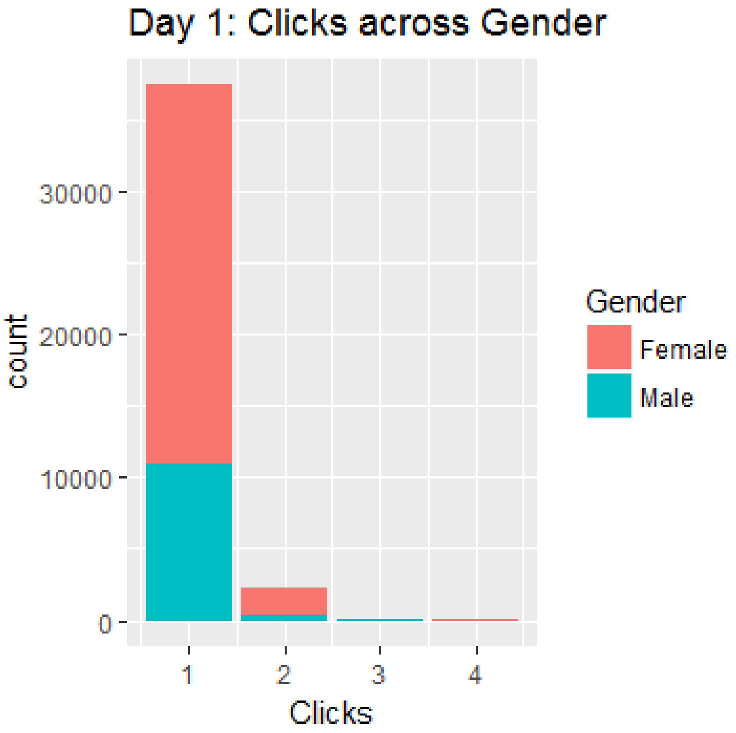


```
ggplot(day_3, aes(x=Impressions,  fill=Gender, colour=Gender)) +
  geom_bar() + xlab("Impressions") + ggtitle("Day 3: Impressions across Gender")
```

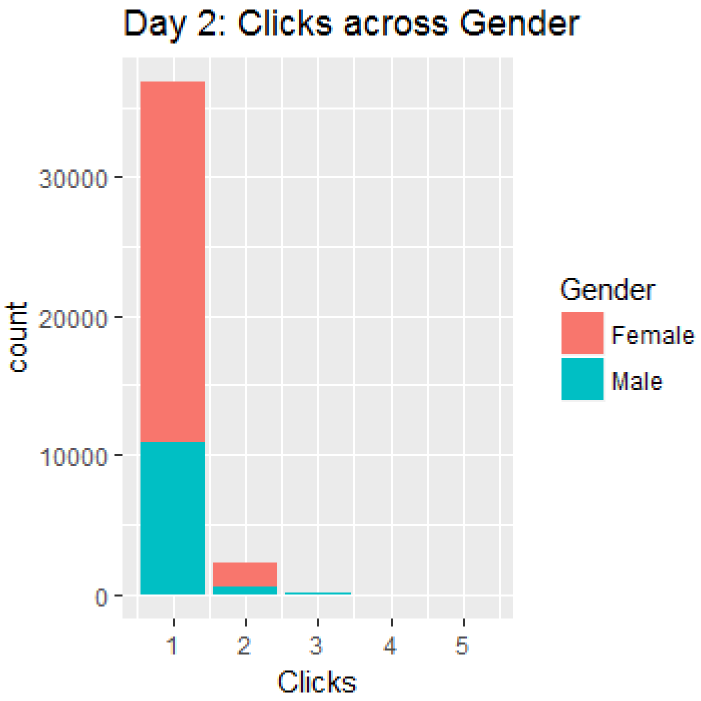


```
#DISTRIBUTION OF CLICKS ACROSS GENDER
#Plots
```

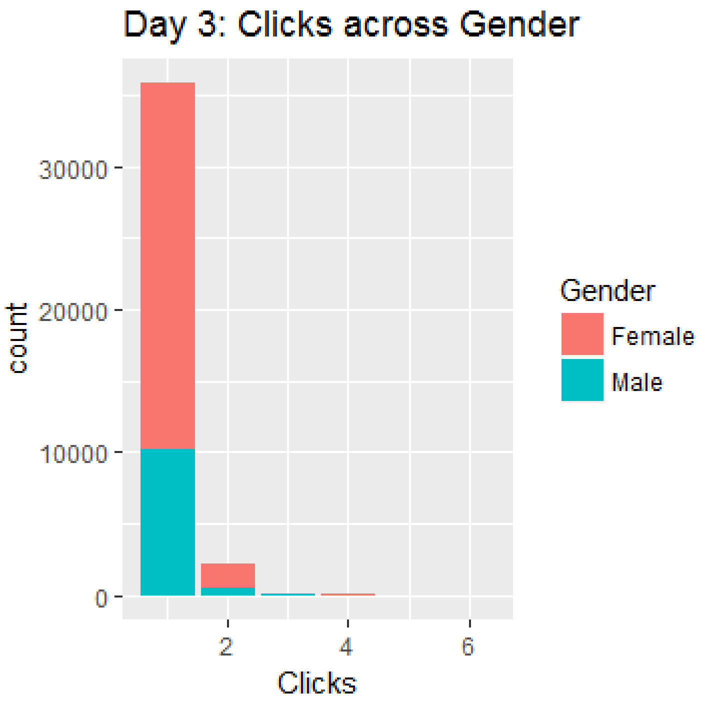
```
ggplot(subset(day_1, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Gender)) + geom_bar() +
  ggtitle("Day 1: Clicks across Gender")
```



```
ggplot(subset(day_2, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Gender)) + geom_bar() +
  ggtitle("Day 2: Clicks across Gender")
```

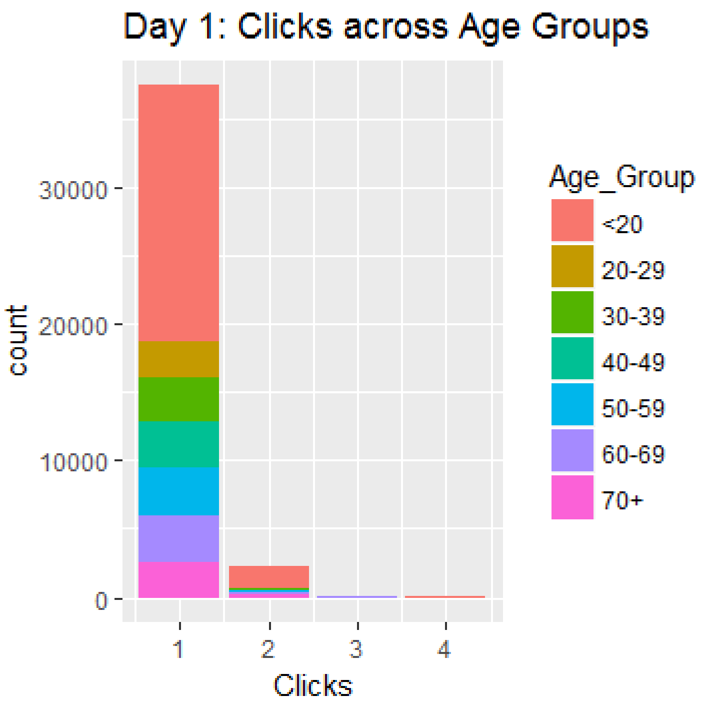


```
ggplot(subset(day_3, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Gender)) + geom_bar() +
  ggtitle("Day 3: Clicks across Gender")
```

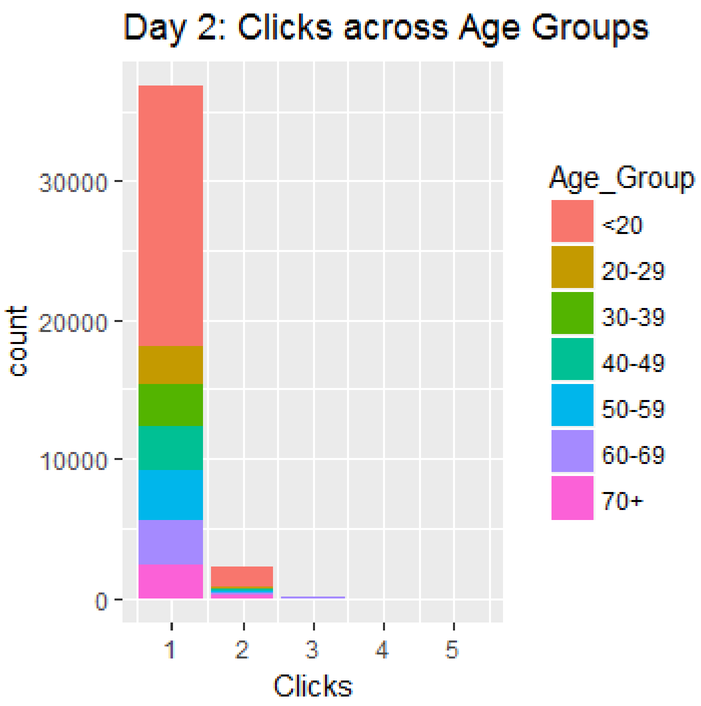


```
#DISTRIBUTION OF CLICKS ACROSS AGE GROUPS
#Plots
```

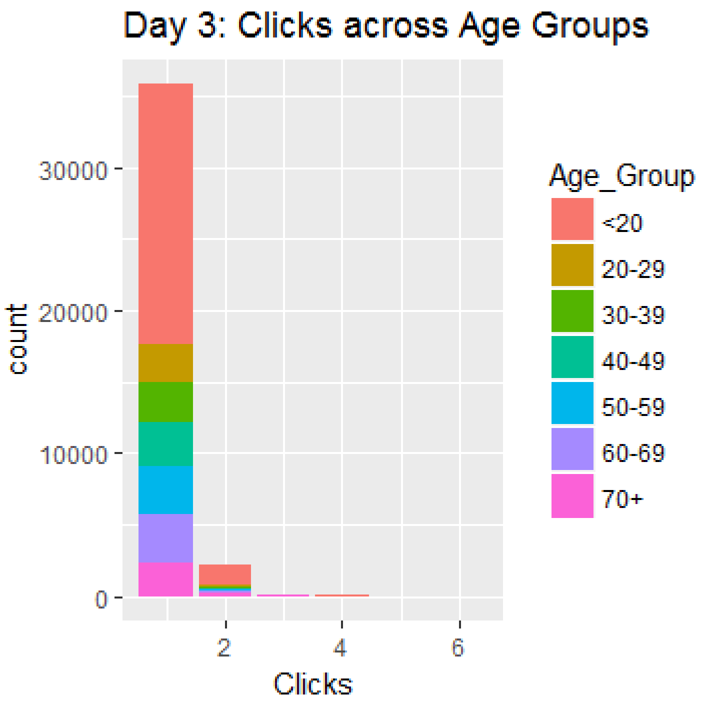
```
ggplot(subset(day_1, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Age_Group)) + geom_bar() +
  ggtitle("Day 1: Clicks across Age Groups")
```



```
ggplot(subset(day_2, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Age_Group)) + geom_bar() +
  ggtitle("Day 2: Clicks across Age Groups")
```

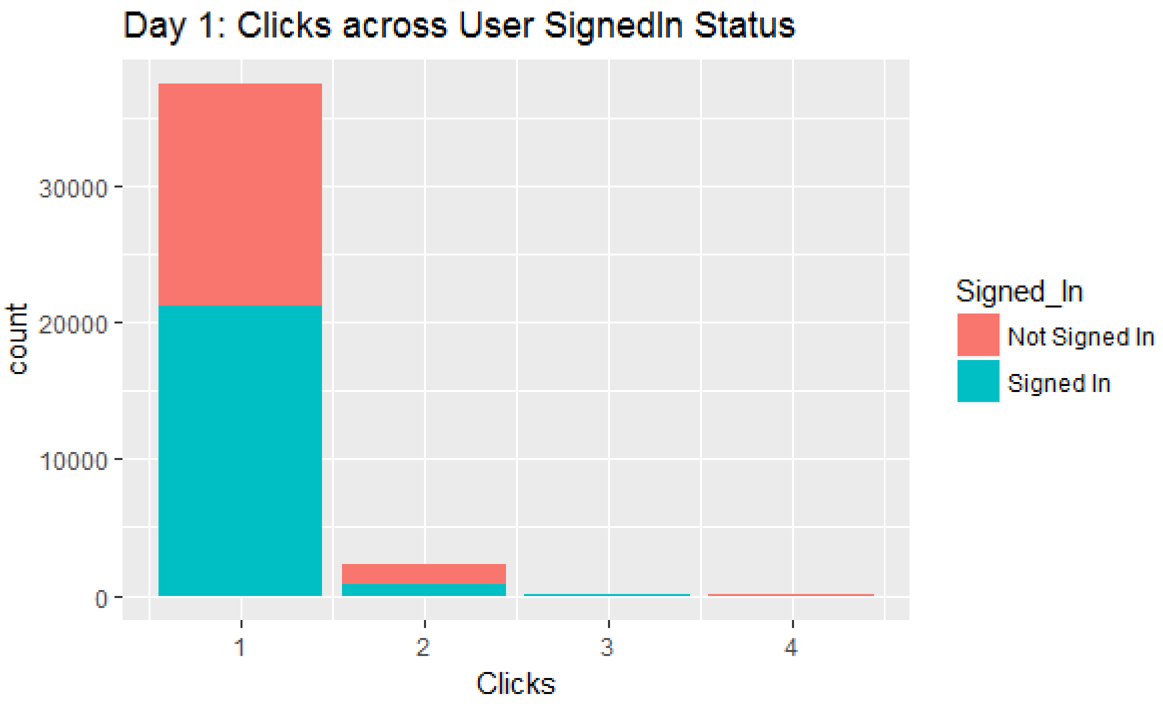


```r
ggplot(subset(day_3, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Age_Group)) + geom_bar() +
  ggtitle("Day 3: Clicks across Age Groups")
```

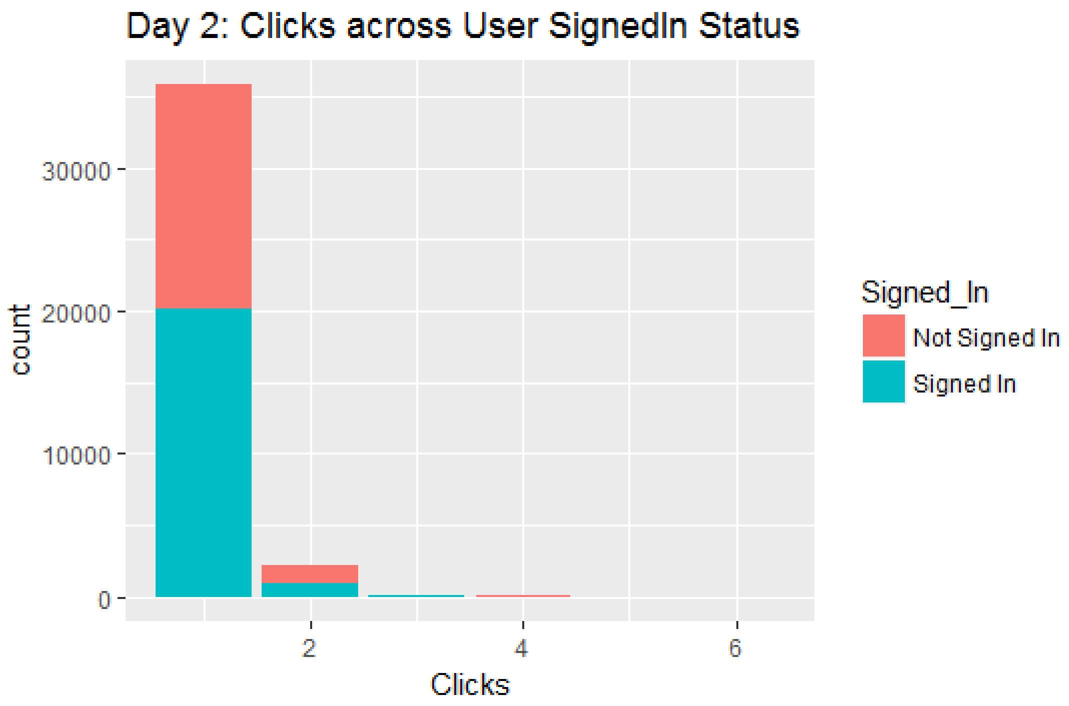


#DISTRIBUTION OF CLICKS ACROSS SIGNED AND UNSIGNED USERS
#Plots

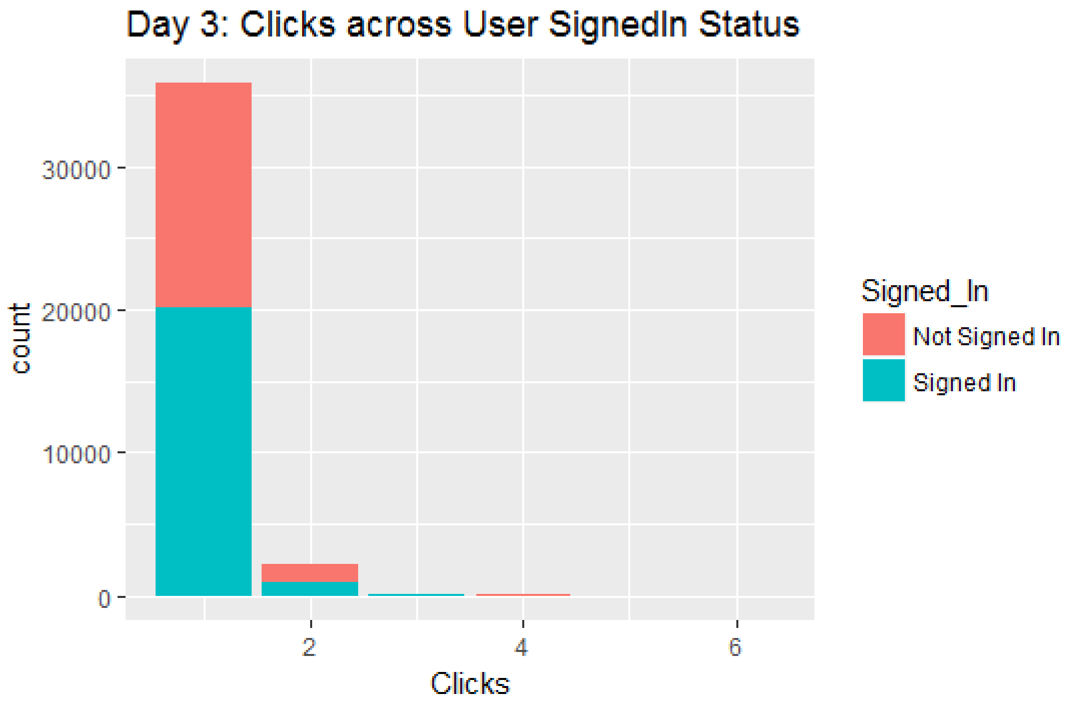
```r
ggplot(subset(day_1, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Signed_In)) + geom_bar() +
  ggtitle("Day 1: Clicks across User SignedIn Status")
```



```r
ggplot(subset(day_3, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Signed_In)) + geom_bar() +
  ggtitle("Day 2: Clicks across User SignedIn Status")
```



```r
ggplot(subset(day_3, Impressions > 0 & Clicks > 0), aes(x=Clicks, fill=Signed_In)) + geom_bar() +
  ggtitle("Day 3: Clicks across User SignedIn Status")
```


Day 3: Clicks across User SignedIn Status
count
Clicks
Signed_In
🟥 Not Signed In
🟦 Signed In