Exploratory and Preliminary Data Analysis of Rideshare dataset

Gulzina Kuttubekova & Meltem Ozcan

11/2/2019

Data:

- Where did we get it? Data can be downloaded here: Kaggle
- How it was collected? . It contains game sales information starting from
- By whom? References: Rush Kirubi, Gregory Smith and Kendall Gillies

Data summary:

```
games <- read.csv('data/Video_Games_Sales_as_at_22_Dec_2016.csv',
                  header = TRUE , na.strings=c("", " ", "N/A", "NA"))
games %>% glimpse()
```

```
## Observations: 16,719
## Variables: 16
## $ Name            <fct> Wii Sports, Super Mario Bros., Mario Kart Wii,...
## $ Platform        <fct> Wii, NES, Wii, Wii, GB, GB, DS, Wii, Wii, NES,...
## $ Year_of_Release <int> 2006, 1985, 2008, 2009, 1996, 1989, 2006, 2006...
## $ Genre           <fct> Sports, Platform, Racing, Sports, Role-Playing...
## $ Publisher       <fct> Nintendo, Nintendo, Nintendo, Nintendo, Ninten...
## $ NA_Sales        <dbl> 41.36, 29.08, 15.68, 15.61, 11.27, 23.20, 11.2...
## $ EU_Sales        <dbl> 28.96, 3.58, 12.76, 10.93, 8.89, 2.26, 9.14, 9...
## $ JP_Sales        <dbl> 3.77, 6.81, 3.79, 3.28, 10.22, 4.22, 6.50, 2.9...
## $ Other_Sales     <dbl> 8.45, 0.77, 3.29, 2.95, 1.00, 0.58, 2.88, 2.84...
## $ Global_Sales    <dbl> 82.53, 40.24, 35.52, 32.77, 31.37, 30.26, 29.8...
## $ Critic_Score    <int> 76, NA, 82, 80, NA, NA, 89, 58, 87, NA, NA, 91...
## $ Critic_Count    <int> 51, NA, 73, 73, NA, NA, 65, 41, 80, NA, NA, 64...
## $ User_Score      <fct> 8, NA, 8.3, 8, NA, NA, 8.5, 6.6, 8.4, NA, NA, ...
## $ User_Count      <int> 322, NA, 709, 192, NA, NA, 431, 129, 594, NA, ...
## $ Developer       <fct> Nintendo, NA, Nintendo, Nintendo, NA, NA, Nint...
## $ Rating          <fct> E, NA, E, E, NA, NA, E, E, E, NA, NA, E, NA, E...
```

There are 16,719 instances observed with 16 variables. There are 5 categorical: {Name, Platform, Genre, Publisher, Developer} and 10 numerical: {Year, NA_Sales, EU_Sales, JP_Sales, Other_Sales, Global_Sales, Critic_Score, Critic_Count, User_Score, User_Count, Rating} variables.

```
describe <- function(var) {
# This function summarizes each variable. It prints out descriptive sample
# statistics for numerical variables, and contingency table for categorical
# variables.
#
# Args:
#   var: variable in a vector form
#
# Returns:
#   Summary analysis
```

```
    if (is.numeric(var)) {
        summary(var)
    }

    else if (is.factor(var)) {
        if (length(unique(var)) > 40) {
            sprintf('There are %d unique cases', length(unique(var)))
        } else {
            table(var)
        }
    }
}
```

describe() function applied on each variable in Games dataset, outputs the following result:

```
lapply(games, describe)
```

```
## $Name
## [1] "There are 11563 unique cases"
##
## $Platform
## var
## 2600  3DO  3DS   DC   DS   GB  GBA   GC  GEN   GG  N64  NES   NG   PC PCFX
##  133    3  520   52 2152   98  822  556   29    1  319   98   12  974    1
##   PS  PS2  PS3  PS4  PSP  PSV  SAT  SCD SNES TG16  Wii WiiU   WS X360   XB
## 1197 2161 1331  393 1209  432  173    6  239    2 1320  147    6 1262  824
## XOne
##  247
##
## $Year_of_Release
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1980    2003    2007    2006    2010    2020     269
##
## $Genre
## var
##       Action    Adventure     Fighting         Misc     Platform
##         3370         1303          849         1750          888
##       Puzzle       Racing Role-Playing      Shooter   Simulation
##          580         1249         1500         1323          874
##       Sports     Strategy
##         2348          683
##
## $Publisher
## [1] "There are 582 unique cases"
##
## $NA_Sales
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0800  0.2633  0.2400 41.3600
##
## $EU_Sales
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.020   0.145   0.110  28.960
##
## $JP_Sales
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.0776  0.0400 10.2200
##
## $Other_Sales
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.00000  0.01000  0.04733  0.03000 10.57000
##
## $Global_Sales
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0100  0.0600  0.1700  0.5335  0.4700 82.5300
##
## $Critic_Score
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   13.00   60.00   71.00   68.97   79.00   98.00    8582
##
## $Critic_Count
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    3.00   12.00   21.00   26.36   36.00  113.00    8582
##
## $User_Score
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   6.400   7.500   7.125   8.200   9.700    9129
##
## $User_Count
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     4.0    10.0    24.0   162.2    81.0 10665.0    9129
##
## $Developer
## [1] "There are 1697 unique cases"
##
## $Rating
## var
##   AO    E E10+   EC  K-A    M   RP    T
##    1 3991 1420    8    3 1563    3 2961
```

```
# the same analysis with summary function()
games %>% summary()
```

```
##                           Name          Platform    Year_of_Release
##  Need for Speed: Most Wanted:   12   PS2    :2161   Min.   :1980
##  FIFA 14                     :    9   DS     :2152   1st Qu.:2003
##  LEGO Marvel Super Heroes    :    9   PS3    :1331   Median :2007
##  Madden NFL 07               :    9   Wii    :1320   Mean   :2006
##  Ratatouille                 :    9   X360   :1262   3rd Qu.:2010
##  (Other)                     :16669   PSP    :1209   Max.   :2020
##  NA's                        :    2   (Other):7284   NA's   :269
##           Genre                                Publisher
##  Action      :3370   Electronic Arts             : 1356
##  Sports      :2348   Activision                  :  985
##  Misc        :1750   Namco Bandai Games          :  939
##  Role-Playing:1500   Ubisoft                     :  933
##  Shooter     :1323   Konami Digital Entertainment:  834
##  (Other)     :6426   (Other)                     :11618
##  NA's        :   2   NA's                        :   54
##     NA_Sales          EU_Sales         JP_Sales        Other_Sales
```

```
##  Min.   : 0.0000   Min.   : 0.000   Min.   : 0.0000   Min.   : 0.00000
##  1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.: 0.0000   1st Qu.: 0.00000
##  Median : 0.0800   Median : 0.020   Median : 0.0000   Median : 0.01000
##  Mean   : 0.2633   Mean   : 0.145   Mean   : 0.0776   Mean   : 0.04733
##  3rd Qu.: 0.2400   3rd Qu.: 0.110   3rd Qu.: 0.0400   3rd Qu.: 0.03000
##  Max.   :41.3600   Max.   :28.960   Max.   :10.2200   Max.   :10.57000
##
##   Global_Sales      Critic_Score    Critic_Count      User_Score
##  Min.   : 0.0100   Min.   :13.00   Min.   :  3.00   Min.   :0.000
##  1st Qu.: 0.0600   1st Qu.:60.00   1st Qu.: 12.00   1st Qu.:6.400
##  Median : 0.1700   Median :71.00   Median : 21.00   Median :7.500
##  Mean   : 0.5335   Mean   :68.97   Mean   : 26.36   Mean   :7.125
##  3rd Qu.: 0.4700   3rd Qu.:79.00   3rd Qu.: 36.00   3rd Qu.:8.200
##  Max.   :82.5300   Max.   :98.00   Max.   :113.00   Max.   :9.700
##                    NA's   :8582    NA's   :8582     NA's   :9129
##    User_Count          Developer        Rating
##  Min.   :    4.0   Ubisoft  : 204   E      :3991
##  1st Qu.:   10.0   EA Sports: 172   T      :2961
##  Median :   24.0   EA Canada: 167   M      :1563
##  Mean   :  162.2   Konami   : 162   E10+   :1420
##  3rd Qu.:   81.0   Capcom   : 139   EC     :   8
##  Max.   :10665.0   (Other)  :9252   (Other):   7
##  NA's   :9129      NA's     :6623   NA's   :6769
```

According to the output, there are total 1697 unique developers, and 582 unique pusblishers. Top 3 most popular platforms are *PS2*, *DS* and *PS3*. Top 3 most popular genres are *Action*, *Sports* and *Miscallenous*. The average sales in North America is $263k, while in Europe it's $145k, $78k in Japan and $47k in other parts of the world. The average global sales resulted in $534k.

```
# Write a function to calculate NA's or missing values
count_NAs <- function(var) {
# Counts missing values in each variable
#
# Args:
#   var: variable in a vector form
#
# Returns:
#   Number of missing values in each variable
    sprintf('There are %d missing values', sum(is.na(var)))
}
```

```
lapply(games, count_NAs)
```

```
## $Name
## [1] "There are 2 missing values"
##
## $Platform
## [1] "There are 0 missing values"
##
## $Year_of_Release
## [1] "There are 269 missing values"
##
## $Genre
## [1] "There are 2 missing values"
##
## $Publisher
```

```
## [1] "There are 54 missing values"
##
## $NA_Sales
## [1] "There are 0 missing values"
##
## $EU_Sales
## [1] "There are 0 missing values"
##
## $JP_Sales
## [1] "There are 0 missing values"
##
## $Other_Sales
## [1] "There are 0 missing values"
##
## $Global_Sales
## [1] "There are 0 missing values"
##
## $Critic_Score
## [1] "There are 8582 missing values"
##
## $Critic_Count
## [1] "There are 8582 missing values"
##
## $User_Score
## [1] "There are 9129 missing values"
##
## $User_Count
## [1] "There are 9129 missing values"
##
## $Developer
## [1] "There are 6623 missing values"
##
## $Rating
## [1] "There are 6769 missing values"
```

There are many missing values almost in every variable. For instance, there are 269 observations with unknown release date. Also, there are massive missing values in the variables like Critic_Score, Critic_Count, User_Score, User_Count, Developer and Rating. **Since most of the missing values are in categorical variables**, we can't just replace/fill, so we will drop/omit those missing values.

Data cleaning/manipulation:

```
games <- na.omit(games)
games %>% glimpse()
```

```
## Observations: 6,825
## Variables: 16
## $ Name            <fct> Wii Sports, Mario Kart Wii, Wii Sports Resort,...
## $ Platform        <fct> Wii, Wii, Wii, DS, Wii, Wii, DS, Wii, X360, Wi...
## $ Year_of_Release <int> 2006, 2008, 2009, 2006, 2006, 2009, 2005, 2007...
## $ Genre           <fct> Sports, Racing, Sports, Platform, Misc, Platfo...
## $ Publisher       <fct> Nintendo, Nintendo, Nintendo, Nintendo, Ninten...
## $ NA_Sales        <dbl> 41.36, 15.68, 15.61, 11.28, 13.96, 14.44, 9.71...
## $ EU_Sales        <dbl> 28.96, 12.76, 10.93, 9.14, 9.18, 6.94, 7.47, 8...
## $ JP_Sales        <dbl> 3.77, 3.79, 3.28, 6.50, 2.93, 4.70, 4.13, 3.60...
```

```
## $ Other_Sales     <dbl> 8.45, 3.29, 2.95, 2.88, 2.84, 2.24, 1.90, 2.15...
## $ Global_Sales    <dbl> 82.53, 35.52, 32.77, 29.80, 28.92, 28.32, 23.2...
## $ Critic_Score    <int> 76, 82, 80, 89, 58, 87, 91, 80, 61, 80, 97, 95...
## $ Critic_Count    <int> 51, 73, 73, 65, 41, 80, 64, 63, 45, 33, 50, 80...
## $ User_Score      <dbl> 8.0, 8.3, 8.0, 8.5, 6.6, 8.4, 8.6, 7.7, 6.3, 7...
## $ User_Count      <int> 322, 709, 192, 431, 129, 594, 464, 146, 106, 5...
## $ Developer       <fct> "Nintendo", "Nintendo", "Nintendo", "Nintendo"...
## $ Rating          <fct> E, E, E, E, E, E, E, E, E, E, M, M, E, M, M, E...
```

```
games %>% summary()
```

```
##                                          Name         Platform
##  LEGO Star Wars II: The Original Trilogy  :   8   PS2    :1140
##  Madden NFL 07                            :   8   X360   : 858
##  Need for Speed: Most Wanted              :   8   PS3    : 769
##  Harry Potter and the Order of the Phoenix:   7   PC     : 651
##  Madden NFL 08                            :   7   XB     : 565
##  Need for Speed Carbon                    :   7   Wii    : 479
##  (Other)                                  :6780   (Other):2363
##  Year_of_Release          Genre                          Publisher
##  Min.   :1985    Action      :1630   Electronic Arts            : 944
##  1st Qu.:2004    Sports      : 943   Ubisoft                    : 496
##  Median :2007    Shooter     : 864   Activision                 : 492
##  Mean   :2007    Role-Playing: 712   Sony Computer Entertainment: 316
##  3rd Qu.:2011    Racing      : 581   THQ                        : 307
##  Max.   :2016    Platform    : 403   Nintendo                   : 291
##                  (Other)     :1692   (Other)                    :3979
##     NA_Sales          EU_Sales          JP_Sales        Other_Sales
##  Min.   : 0.0000   Min.   : 0.0000   Min.   :0.00000   Min.   : 0.00000
##  1st Qu.: 0.0600   1st Qu.: 0.0200   1st Qu.:0.00000   1st Qu.: 0.01000
##  Median : 0.1500   Median : 0.0600   Median :0.00000   Median : 0.02000
##  Mean   : 0.3945   Mean   : 0.2361   Mean   :0.06416   Mean   : 0.08268
##  3rd Qu.: 0.3900   3rd Qu.: 0.2100   3rd Qu.:0.01000   3rd Qu.: 0.07000
##  Max.   :41.3600   Max.   :28.9600   Max.   :6.50000   Max.   :10.57000
##
##   Global_Sales      Critic_Score    Critic_Count      User_Score
##  Min.   : 0.0100   Min.   :13.00   Min.   :  3.00   Min.   :0.500
##  1st Qu.: 0.1100   1st Qu.:62.00   1st Qu.: 14.00   1st Qu.:6.500
##  Median : 0.2900   Median :72.00   Median : 25.00   Median :7.500
##  Mean   : 0.7776   Mean   :70.27   Mean   : 28.93   Mean   :7.186
##  3rd Qu.: 0.7500   3rd Qu.:80.00   3rd Qu.: 39.00   3rd Qu.:8.200
##  Max.   :82.5300   Max.   :98.00   Max.   :113.00   Max.   :9.600
##
##    User_Count                Developer        Rating
##  Min.   :    4.0   EA Canada       : 149   T      :2377
##  1st Qu.:   11.0   EA Sports       : 142   E      :2082
##  Median :   27.0   Capcom          : 126   M      :1433
##  Mean   :  174.7   Ubisoft         : 103   E10+   : 930
##  3rd Qu.:   89.0   Konami          :  95   AO     :   1
##  Max.   :10665.0   Ubisoft Montreal:  87   K-A    :   1
##                    (Other)         :6123   (Other):   1
```

After removing all NA cases, we were left with 6947 observations. Hence we lost %58.44 of inital observations.

For more flexibility we can generate new variables out of given 16. Below are given the mechanisms we used to generate those additional features:

Write the reasoning here why do we need this KPI/variable?

1. Year since release: numeric variable

```
games$Year_since_Release <- max(games$Year_of_Release) - games$Year_of_Release
```

2. Year to decade categorical

```
to_decade <- function(var) {
    var[var <= 1980] <- '80s'
    var[1980 < var & var <= 1990] <- '90s'
    var[1990 < var & var <= 2000] <- '00s'
    var[2000 < var & var <= 2010] <- '10s'
    var[2010 < var & var <= 2020] <- '20s'

    return(var)
}
```

```
games$Decade <- to_decade(games$Year_of_Release)
```

3. Platform Developer company

```
platform_company <- function(var) {
    var <- as.character(var)

    var[var == "3DS"|
        var == "DS"|
        var == "GBA"|
        var == "GC"|
        var == "Wii"|
        var == "WiiU"] = 'Nintendo'
    var[var == "DC"] = 'Sega'
    var[var == "PC"|
        var == "X360"|
        var == "XB"|
        var == "XOne"] = 'Microsoft'
    var[var == "PS"|
        var == "PS2"|
        var == "PS3"|
        var == "PS4"|
        var == "PSP"|
        var == "PSV"] = 'SONY'

    return(var)
}
```

```
games$Platform_Company <- platform_company(games$Platform)
```

4. Platforms Generation

Create new variable by generation of platforms:

```
platforms_gen <- function(var) {
    var = as.character(var)

    var[var == "PS" |
        var == "PC"] = 'Fifth Gen'
    var[var == "DC" |
```

```
        var == "PS2" |
        var == "XB" |
        var == "GBA" |
        var == "GC"] = 'Sixth Gen'
    var[var == "DS" |
        var == "PS3" |
        var == "PSP" |
        var == "Wii" |
        var == "X360"] = 'Seventh Gen'
    var[var == "3DS" |
        var == "PS4" |
        var == "PSV" |
        var == "WiiU" |
        var == "XOne"] = 'Eigth Gen'

    return(var)
}
```

```
games$Platform_Gen <- platforms_gen(games$Platform)
```

5. Variable by family of systems

```
family_platforms <- function(var) {
    var = as.character(var)

    var[var == "3DS" |
        var ==  "DS"] = 'Nintendo_DS'
    var[var == "DC" |
        var == "PC" |
        var == "GBA" |
        var == "GC"] = 'Misc'
    var[var == "X360" |
        var == "XB" |
        var == "XOne"] = 'Microsoft_XBOX'
    var[var == "PS" |
        var == "PS2" |
        var == "PS3" |
        var == "PS4" |
        var == "PSP" |
        var == "PSV"] = 'SONY_PS'
    var[var == "Wii" |
        var == "WiiU"] = 'Nintendo_Wii'

    return(var)
}
```

```
games$Family_Platform <- family_platforms(games$Platform)
```

6. Developer (publisher) country information

```
developer_countries <- read.xls('data/Country_data.xlsx', header = TRUE)
developer_countries %>% glimpse()
```

```
## Observations: 647
## Variables: 3
## $ Developer   <fct> Overflow, 1-Up Studio, 1C Company, 2K Czech, 2K Ga...
```

```
## $ Country     <fct> Japan, Japan, Russia, Czech Republic, US, US, US, ...
## $ Established <int> 1997, 2000, 1991, 1997, 2005, 2005, 1987, 2005, 20...
```

```
developer_countries %>% head()
```

```
##     Developer        Country Established
## 1    Overflow          Japan        1997
## 2 1-Up Studio          Japan        2000
## 3  1C Company         Russia        1991
## 4    2K Czech Czech Republic        1997
## 5    2K Games             US        2005
## 6   2K Sports             US        2005
```

Link this information to the developer variable: (we will actually refer to the top 10 largest video game developer companies in the world):

```
filter_developer <- function(var, ids = as.numeric(rownames(games))) {
    var <- as.character(var)

    var[grepl('Ubisoft', var)] <- 'Ubisoft'
    var[grepl('Vivendi', var)] <- 'Ubisoft'
    var[grepl('Ivory Tower', var)] <- 'Ubisoft'
    var[grepl('Massive Entertainment', var)] <- 'Ubisoft'
    var[grepl('Nadeo', var)] <- 'Ubisoft'
    var[grepl('Red Storm', var)] <- 'Ubisoft'
    var[grepl('RedLynx', var)] <- 'Ubisoft'
    var[grepl('Microids', var)] <- 'Ubisoft'
    var[grepl('Related Designs', var)] <- 'Ubisoft'
    var[grepl('Sunflowers Interactive', var)] <- 'Ubisoft'
    var[ids %in% c(2585, 6804)] <- 'Ubisoft' #games developed by Marvelous
    var[ids == 3227] <- 'Ubisoft' #game developed by Media Vision
    var[ids %in% c(4455, 7097, 7213)] <- 'Ubisoft' #games developed by Q entertainment
    var[ids %in% c(8816, 11234)] <- 'Ubisoft' #games developed by Racjin
    var[ids == 5039] <- 'Ubisoft' #games by Yuke's
    var[ids == 11246] <- 'Ubisoft'
    var[ids %in% c(12158, 14003)] <- 'Ubisoft' #Climax
    var[ids %in% c(9025, 9794)] <- 'Ubisoft' #Kuju
    var[ids == 6900] <- 'Ubisoft' #Sumo
    var[ids %in% c(3133, 4934, 5181, 6076, 6685, 7700, 7998,
                   9135, 9945, 10701, 11164, 14849)] <- 'Ubisoft' #Eurocom


    var[grepl('Nintendo', var)] <- 'Nintendo'
    var[grepl('Monolith Soft', var)] <- 'Nintendo'
    var[grepl('Retro Studios', var)] <- 'Nintendo'
    var[grepl('Ambrella', var)] <- 'Nintendo'
    var[grepl('Camelot Software Planning', var)] <- 'Nintendo'
    var[grepl('Creatures', var)] <- 'Nintendo'
    var[grepl('Game Freak', var)] <- 'Nintendo'
    var[grepl('Creatures', var)] <- 'Nintendo'
    var[grepl('Genius Sonority', var)] <- 'Nintendo'
    var[grepl('Good-Feel', var)] <- 'Nintendo'
    var[grepl('HAL', var)] <- 'Nintendo'
    var[grepl('Intelligent Systems', var)] <- 'Nintendo'
    var[grepl('Next Level Games', var)] <- 'Nintendo'
    var[grepl('Sora', var)] <- 'Nintendo'
```

```
var[grepl('TOSE', var)] <- 'Nintendo'
var[ids %in% c(313, 554, 730)] <- 'Nintendo' #games developed by Capcom
var[ids %in% c(4121, 4619)] <- 'Nintendo' #games developed by Marvelous
var[ids == 6984 ] <- 'Nintendo' #game developed by Q Entertainment
var[ids == 6309] <- 'Nintendo' #FreeStyleGame
var[ids %in% c(4789, 5367)] <- 'Nintendo' #Kuju
var[ids %in% c(545, 3098, 3109, 4184)] <- 'Nintendo' #Artoon
var[grepl('n-Space', var)] <- 'Nintendo'


var[grepl('SONY', var)] <- 'SONY'
var[grepl('Sony', var)] <- 'SONY'
var[grepl('SCE', var)] <- 'SONY'
var[grepl('SIE', var)] <- 'SONY'
var[grepl('Bend', var)] <- 'SONY'
var[grepl('Insomniac', var)] <- 'SONY'
var[grepl('Media Molecule', var)] <- 'SONY'
var[grepl('Naughty Dog', var)] <- 'SONY'
var[grepl('Polyphony', var)] <- 'SONY'
var[grepl('Sucker', var)] <- 'SONY'
var[grepl('Bigbig', var)] <- 'SONY'
var[grepl('Evolution', var)] <- 'SONY'
var[grepl('Guerrilla', var)] <- 'SONY'
var[grepl('Incognito', var)] <- 'SONY'
var[grepl('Liverpool', var)] <- 'SONY'
var[grepl('Zipper', var)] <- 'SONY'
var[grepl('Clap Hanz', var)] <- 'SONY'
var[ids == 8837] <- 'SONY' #ALfa Systems, Oreshika game
var[ids == 3087] <- 'SONY' #Dimps, Japan Studio
var[ids == 7065] <- 'SONY' #Matrix Systems, Alundra game
var[ids == 10967] <- 'SONY' #Media Vision, Wild Arms
var[ids ==5456] <- 'SONY' #Shift, Ape Academy
var[grepl('Zener', var)] <- 'SONY'
var[grepl('Bluepoint', var)] <- 'SONY'
var[ids %in% c(4515, 7818, 8028)] <- 'SONY' #Harmonix
var[ids %in% c(3291, 4257, 9503)] <- 'SONY' #High Impact Games
var[grepl('Idol', var)] <- 'SONY'
var[grepl('Magic Pixel', var)] <- 'SONY'
#var[ids == 3885] <- 'SONY' #Mass Media
var[grepl('Ready to Dawn', var)] <- 'SONY'
var[ids %in% c(2979, 3341, 3680, 6765, 7509)] <- 'SONY' #Sanzaru Games
var[ids == 9843] <- 'SONY' #SuperVillain
var[grepl('Workshop', var)] <- 'SONY'
var[grepl('Zindagi', var)] <- 'SONY'
var[ids == 8802] <- 'SONY' #Climax
var[ids == 1596] <- 'SONY' #Double Eleven
var[ids == 13343] <- 'SONY' #FreeStyleGames
var[ids == 4560] <- 'SONY' #Frontier
#var[ids == 1115] <- 'SONY' #Ninja
var[ids == 909] <- 'SONY' #Sumo
var[ids %in% c(464, 1128)] <- 'SONY' #Quantic
var[grepl('989', var)] <- 'SONY'
var[grepl('Cohort', var)] <- 'SONY'
```

```
var[ids %in% c(2720, 5647, 7043)] <- 'SONY' #Game Republic
var[grepl('BottleRocket', var)] <- 'SONY'
var[ids %in% c(396, 6284, 8263)] <- 'SONY' #Eurocom
var[ids %in% c(8291, 4746)] <- 'SONY' #Nihilistic
var[ids %in% c(1588, 4428)] <- 'SONY' #Slant Six
var[grepl('SuperBot', var)] <- 'SONY'
var[ids %in% c(1823, 2394)] <- 'SONY' #UnitedFrontGames
var[grepl('Level 5', var)] <- 'SONY'


var[grepl('EA', var)] <- 'EA'
var[grepl('Electronic Arts', var)] <- 'EA'
var[grepl('DICE', var)] <- 'EA'
var[grepl('Ghost Games', var)] <- 'EA'
var[grepl('PopCap', var)] <- 'EA'
var[grepl('BioWare', var)] <- 'EA'
var[grepl('Frostbite', var)] <- 'EA'
var[ids == 1423] <- 'EA' #developed by Bluepoint
#var[ids %in% c(2119, 2201)] <- 'EA' #developed by Double Fine
var[ids == 740] <- 'EA' #developed by Harmonix
var[ids == 9777] <- 'EA' #developed by Sanzaru Games
var[grepl('Exient', var)] <- 'EA'
var[grepl('Maxis', var)] <- 'EA'
var[grepl('Black Box', var)] <- 'EA'
var[ids == 16314] <- 'EA' #Kuju
var[ids %in% c(405, 859, 2504, 2636, 2959, 3592, 4188,
               5235, 7435, 8924)] <- 'EA' #Eurocom
var[ids %in% c(6116, 4992, 2650)] <- 'EA' #Nihilistic


var[grepl('Microsoft', var)] <- 'Microsoft'
var[grepl('343', var)] <- 'Microsoft'
var[grepl('Compulsion', var)] <- 'Microsoft'
var[grepl('Double Fine', var)] <- 'Microsoft'
var[grepl('Ninja Theory', var)] <- 'Microsoft'
var[grepl('Obsidian', var)] <- 'Microsoft'
var[grepl('Playground', var)] <- 'Microsoft'
var[grepl('Rare', var)] <- 'Microsoft'
var[grepl('Coalition', var)] <- 'Microsoft'
var[grepl('Turn', var)] <- 'Microsoft'
var[grepl('Indie', var)] <- 'Microsoft'
var[grepl('Bungie', var)] <- 'Microsoft'
var[grepl('Twisted', var)] <- 'Microsoft'
var[grepl('BigPark', var)] <- 'Microsoft'
var[grepl('Anvil', var)] <- 'Microsoft'
var[grepl('Ensemble', var)] <- 'Microsoft'
var[grepl('FASA', var)] <- 'Microsoft'
var[grepl('Good Science', var)] <- 'Microsoft'
var[grepl('Lionhead', var)] <- 'Microsoft'
var[ids == 1177] <- 'Microsoft'
var[ids == 10678] <- 'Microsoft' #developed by Media Vision
var[ids == 8981] <- 'Microsoft' #developed by Armature
var[ids %in% c(838, 2526, 7960)] <- 'Microsoft' #developed by Harmonix
```

```
var[ids %in% c(15061, 7826)] <- 'Microsoft' #developed by Climax
var[ids %in% c(676, 1138)] <- 'Microsoft' #developed by Frontier
var[ids == 13474] <- 'Microsoft' #Kuju
var[ids %in% c(2263, 3600, 7434)] <- 'Microsoft' #Artoon
var[ids == 1653] <- 'Microsoft' #Blitz


var[grepl('Epic Games', var)] <- 'Tencent'
var[grepl('Paradox', var)] <- 'Tencent'


var[grepl('Activision', var)] <- 'Activision Blizzard'
var[grepl('Blizzard', var)] <- 'Activision Blizzard'
var[grepl('Vicarious Visions', var)] <- 'Activision Blizzard'
var[grepl('Treyarch', var)] <- 'Activision Blizzard'
var[grepl('Beenox', var)] <- 'Activision Blizzard'


var[grepl('Namco', var)] <- 'Namco'
var[grepl('Bandai', var)] <- 'Namco'
var[grepl('Telenet', var)] <- 'Namco'
var[grepl('Atari', var)] <- 'Namco'


var[grepl('GungHo', var)] <- 'GungHo'
var[grepl('Game Arts', var)] <- 'GungHo'
var[grepl('Interchannel', var)] <- 'GungHo'
var[grepl('Grasshopper', var)] <- 'GungHo'
var[grepl('Acquire', var)] <- 'GungHo'


var[grepl('TT', var)] <- 'Warner Brothers'
var[grepl('Tales', var)] <- 'Warner Brothers'
var[grepl('Giants', var)] <- 'Warner Brothers'
var[grepl('Tales', var)] <- 'Warner Brothers'
var[grepl('Warthog', var)] <- 'Warner Brothers'
var[grepl('Avalanche', var)] <- 'Warner Brothers'
var[grepl('Monolith Productions', var)] <- 'Warner Brothers'
var[grepl('NetherRealm', var)] <- 'Warner Brothers'
var[grepl('Rocksteady', var)] <- 'Warner Brothers'
var[grepl('WB', var)] <- 'Warner Brothers'
var[grepl('Midway', var)] <- 'Warner Brothers'
var[grepl('Eidos', var)] <- 'Warner Brothers'
var[grepl('Turbine', var)] <- 'Warner Brothers'


var[grepl('Square', var)] <- 'Square Enix'
var[grepl('Taito', var)] <- 'Square Enix'
var[grepl('Tri-Ace', var)] <- 'Square Enix'


var[grepl('Visual Concepts', var)] <- 'Sega'
var[grepl('Sega', var)] <- 'Sega'
```

```
    var[grepl('2K', var)] <- 'Sega'
    var[grepl('Take-Two', var)] <- 'Sega'
    var[grepl('Sonic', var)] <- 'Sega'
    var[grepl('Kush', var)] <- 'Sega'


    var[grepl('Volition Inc', var)] <- 'THQ'
    var[grepl('THQ', var)] <- 'THQ'
    var[grepl('Deep Silver', var)] <- 'THQ'
    var[grepl('Pacific Coast', var)] <- 'THQ'
    var[grepl('Relic', var)] <- 'THQ'
    var[grepl('Vigil', var)] <- 'THQ'
    var[grepl('Blue Tongue', var)] <- 'THQ'
    var[grepl('Juice', var)] <- 'THQ'
    var[grepl('Kaos', var)] <- 'THQ'
    var[grepl('Paradigm', var)] <- 'THQ'
    var[grepl('Mass Media', var)] <- 'THQ'
    var[grepl('Helixe', var)] <- 'THQ'
    var[grepl('Locomotive', var)] <- 'THQ'
    var[grepl('Heavy Iron', var)] <- 'THQ'
    var[grepl('Incinerator', var)] <- 'THQ'
    var[grepl('Big Huge', var)] <- 'THQ'
    var[grepl('Nordic', var)] <- 'THQ'


    var[grepl('Konami', var)] <- 'Konami'
    var[grepl('Kojima', var)] <- 'Konami'


    var[grepl('Acclaim', var)] <- 'Disney'
    var[grepl('Disney', var)] <- 'Disney'
    var[grepl('Buena', var)] <- 'Disney'
    var[grepl('Propaganda', var)] <- 'Disney'
    var[grepl('TOYBOX', var)] <- 'Disney'


    var[grepl('Capcom', var)] <- 'Capcom'
    var[grepl('Blue Castle', var)] <- 'Capcom'




    var[!(var == 'EA' | var == 'SONY' | var == 'Nintendo' |
              var == 'Microsoft' | var == 'Ubisoft' | var == 'Tencent'
          | var == 'Activision Blizzard' | var == 'Namco'
          | var == 'GungHo')] <- 'Other'

    return(var)
}
```

Needs more reasearch, later if we have time.

```
games$Main_Developer <- filter_developer(games$Developer)
```

7. Game Developer country

```
developer_country <- function(var) {
    var[var == 'EA' |
        var == 'Disney' |
        var == 'SONY' |
        var == 'Microsoft' |
        var == 'Activision Blizzard'] = 'US'

    var[var == 'Ubisoft'] = 'France'

    var[var == 'Nintendo' |
        var == 'Namco' |
        var == 'GungHo'] = 'Japan'

    var[var == 'Tencent'] = 'China'

    return(var)
}
```

```
games$Developer_Country <- developer_country(games$Main_Developer)
```

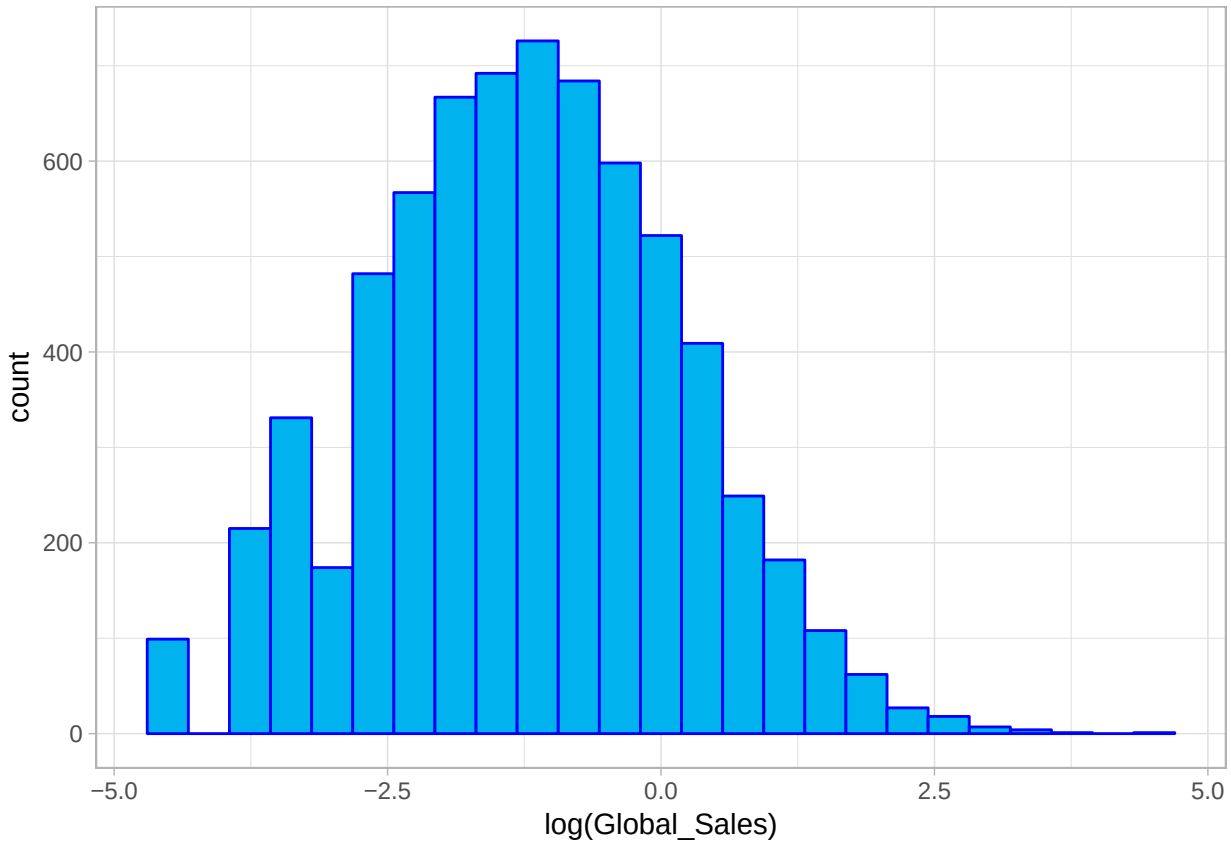
8. Publisher main company

```
games$Main_Publisher <- filter_developer(games$Publisher)
```

Visual analysis of numerical variables:

We are mostly interested in global sales: (We might want to explore which variables affect the global sales)

```
games %>%
    ggplot(aes(x = log(Global_Sales)))+
    geom_histogram(bins = 25, fill = 'deepskyblue2', col = 'blue') +
    theme_light()
```



Comments: since the original scale of global scales is not highly skewed to the right, we did log-transformation on it. For model assumptions: looks pretty good for MLR.

Plot Global sales by region:

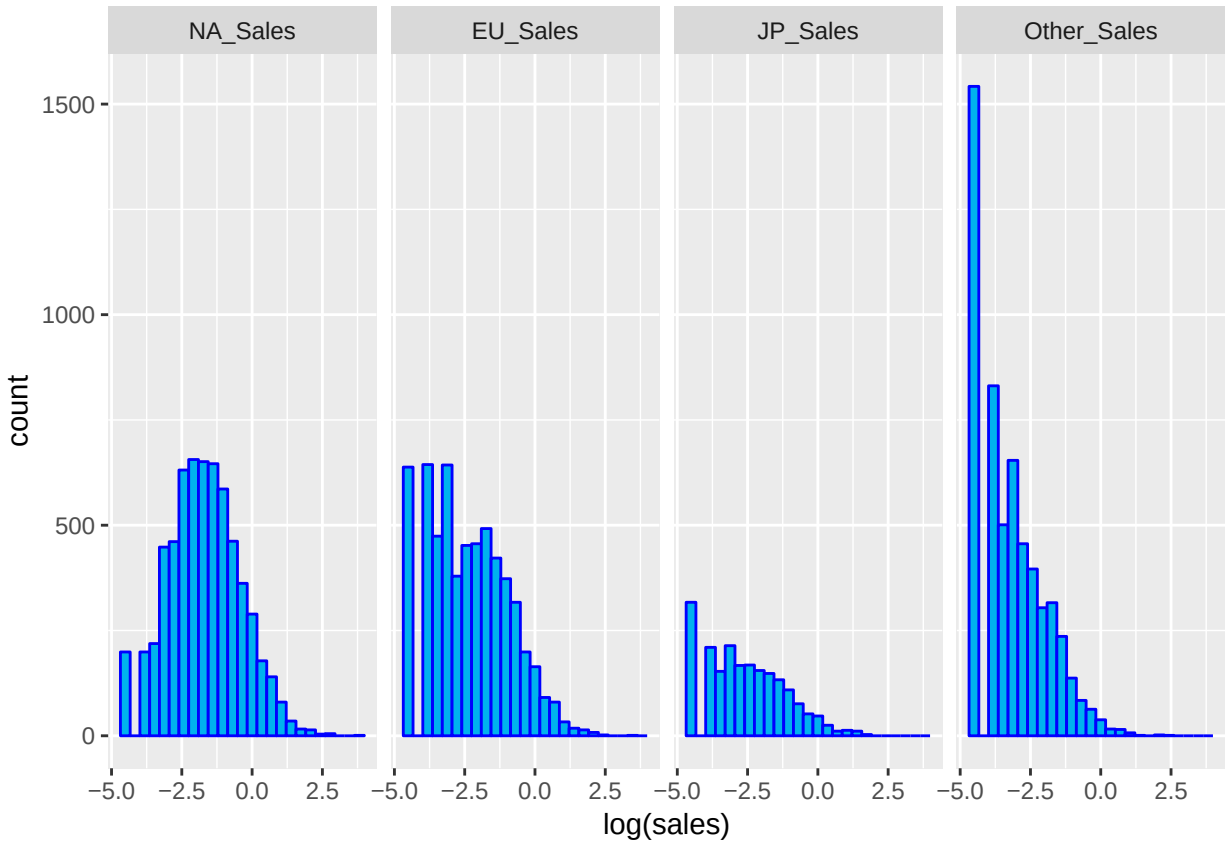
```
sales <- melt(games[,c(6:9)])
```

```
## Using  as id variables
```

```
names(sales) <- c('region', 'sales')

sales %>% ggplot(aes(x = log(sales))) +
    geom_histogram(bins = 25, fill = 'deepskyblue2', col = 'blue') +
    facet_grid(.~ region)
```

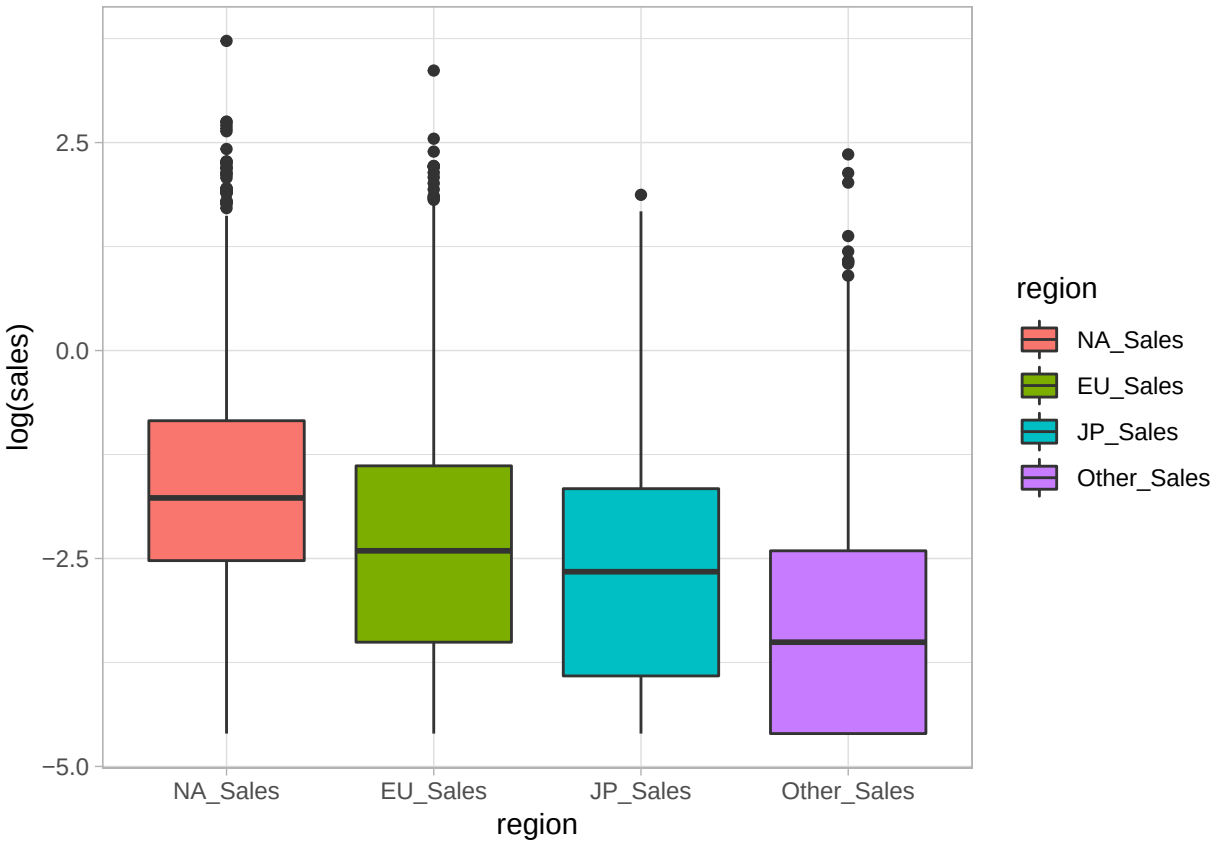
```
## Warning: Removed 7506 rows containing non-finite values (stat_bin).
```



Box-plots by region:

```
sales %>% ggplot(aes(y = log(sales), x = region)) +
    geom_boxplot(aes(fill = region)) +
    theme_light()
```

```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```

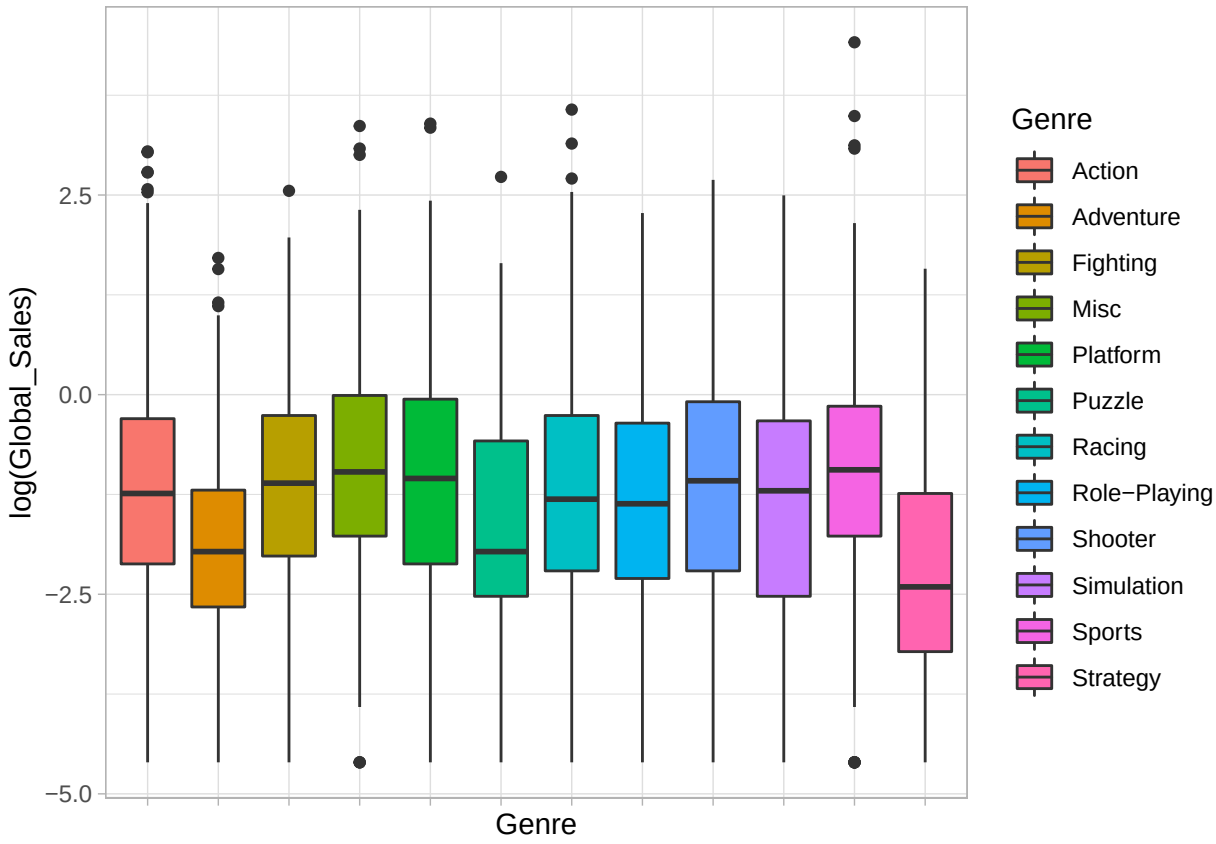


Comments:

We would like to check if the sales differ significantly by region. There is no obvious heteroskedasticity.

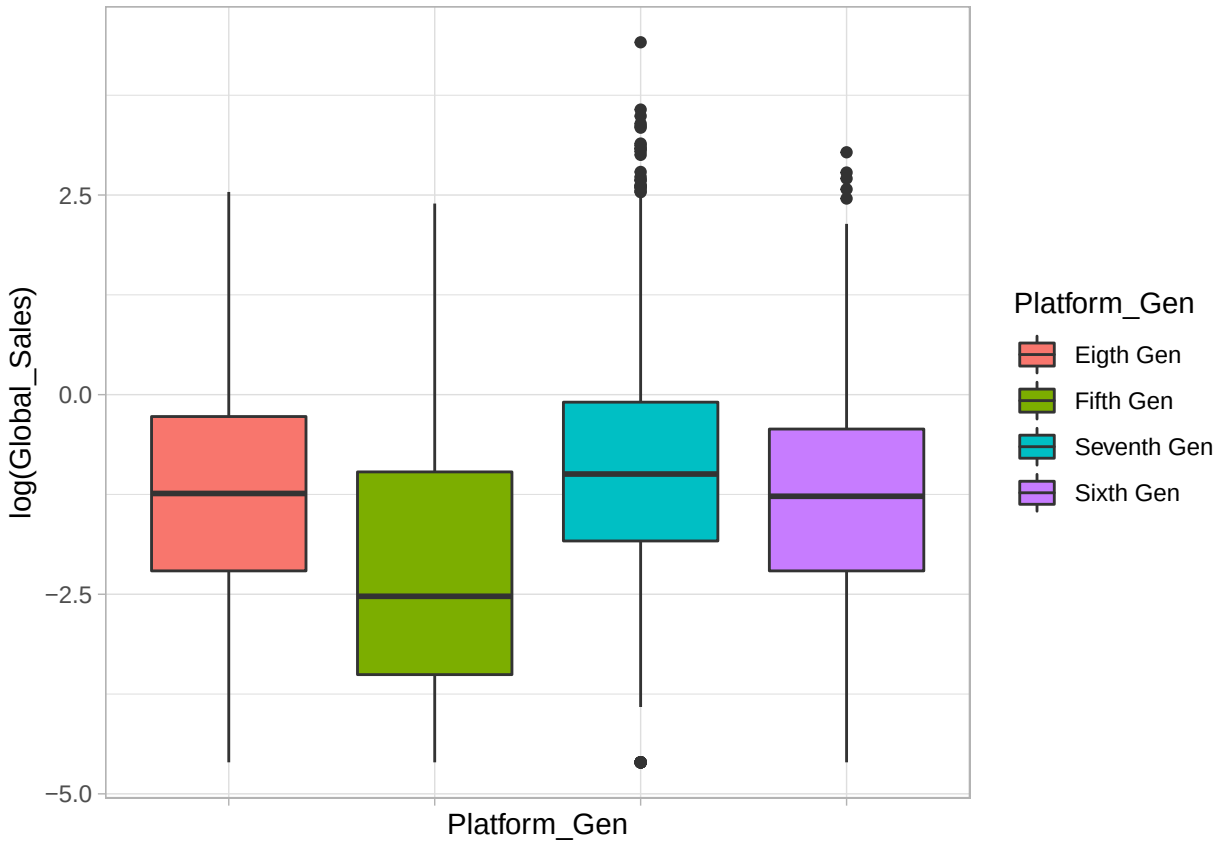
Global sales by genre:

```
games %>% ggplot(aes(y = log(Global_Sales), x = Genre)) +
    geom_boxplot(aes(fill = Genre)) +
    theme_light() +
    theme(axis.text.x = element_blank())
```



Global sales by platform:

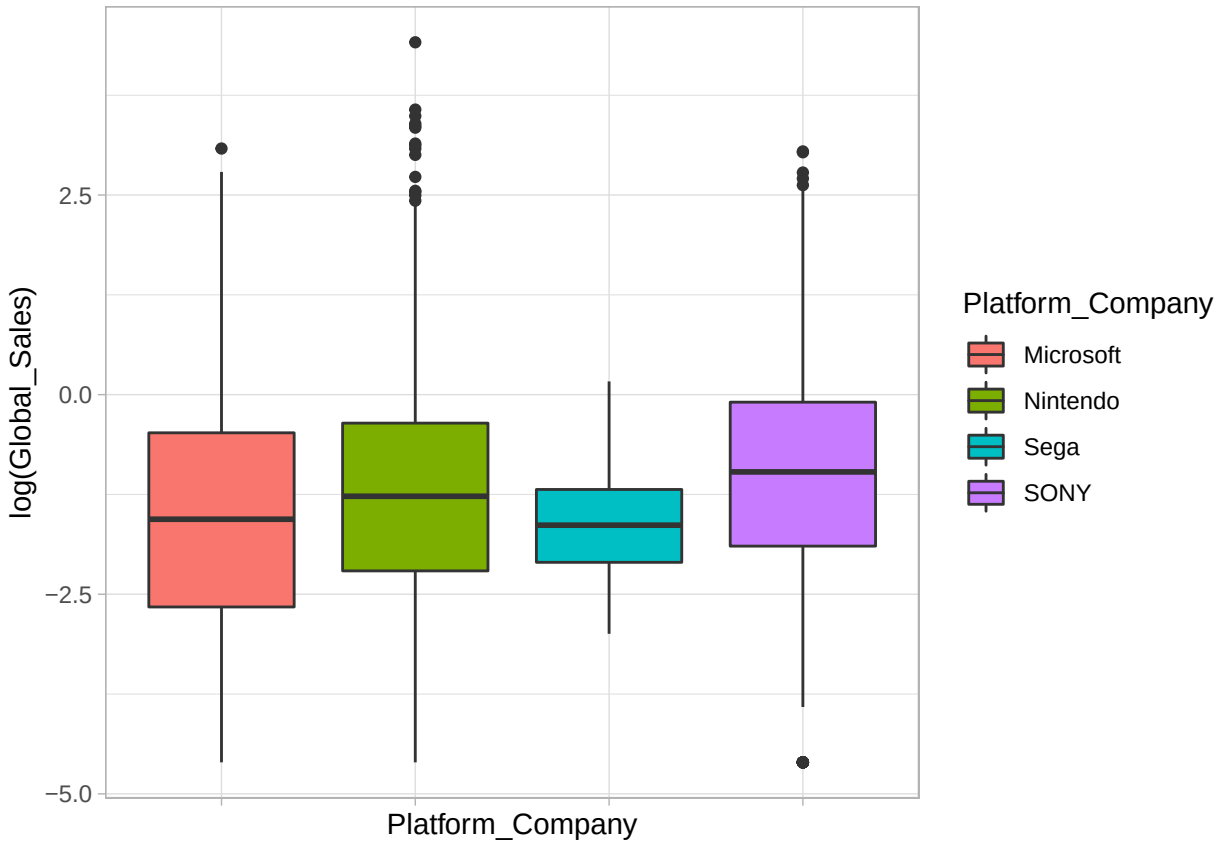
```
games %>% ggplot(aes(y = log(Global_Sales), x = Platform_Gen)) +
    geom_boxplot(aes(fill = Platform_Gen)) +
    theme_light() +
    theme(axis.text.x = element_blank())
```



There are more than platforms, so it would be better to group them: Done

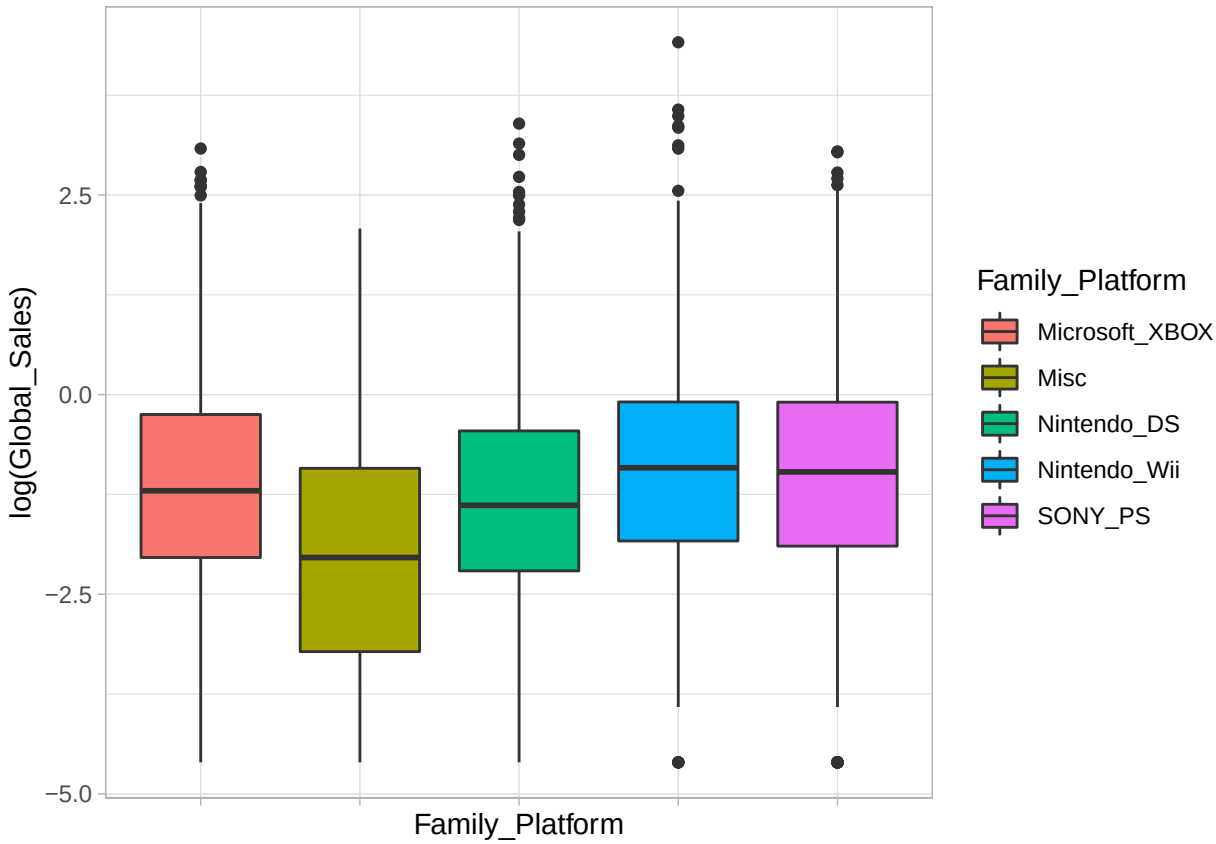
Global sales by platform developing companies:

```
games %>% ggplot(aes(y = log(Global_Sales), x = Platform_Company)) +
    geom_boxplot(aes(fill = Platform_Company)) +
    theme_light() +
    theme(axis.text.x = element_blank())
```



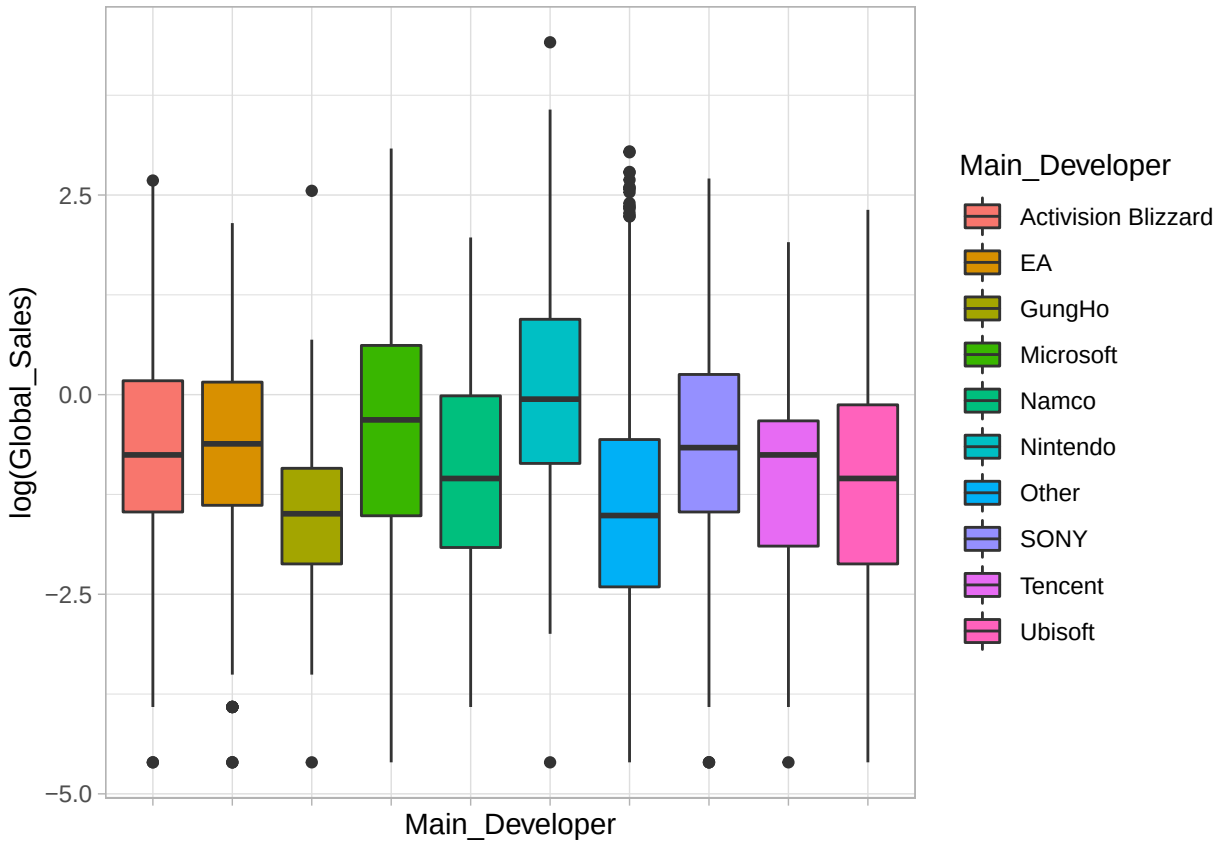
Global sales by platform family:

```
games %>% ggplot(aes(y = log(Global_Sales), x = Family_Platform)) +
    geom_boxplot(aes(fill = Family_Platform)) +
    theme_light() +
    theme(axis.text.x = element_blank())
```



Global sales by main developers:

```
games %>% ggplot(aes(y = log(Global_Sales), x = Main_Developer)) +
    geom_boxplot(aes(fill = Main_Developer)) +
    theme_light() +
    theme(axis.text.x = element_blank())
```



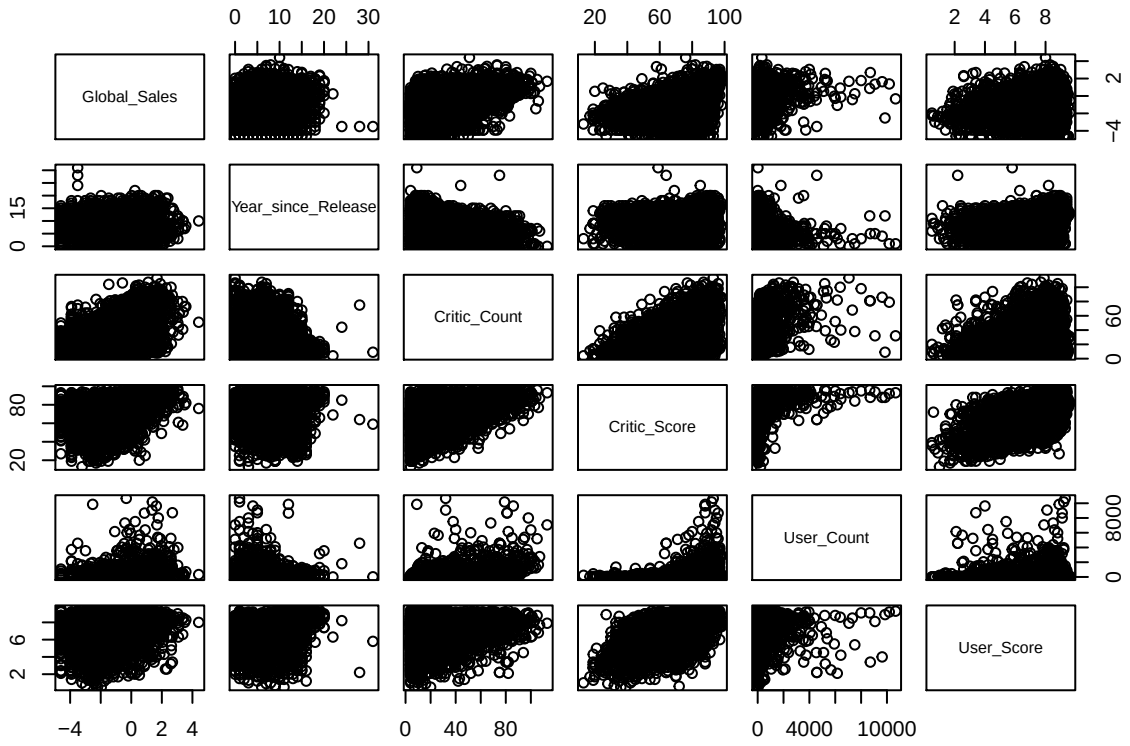
Analysis of numerical variables:

For EDA purposes only, fit their scatter plots and pairwise plots

1. Pairwise plots

```
# choose only numeric variables
X <- games %>%
    select(Global_Sales, Year_since_Release, Critic_Count,
           Critic_Score, User_Count, User_Score) %>%
    mutate_at(c('Global_Sales'), log)
```

```
pairs(X)
```

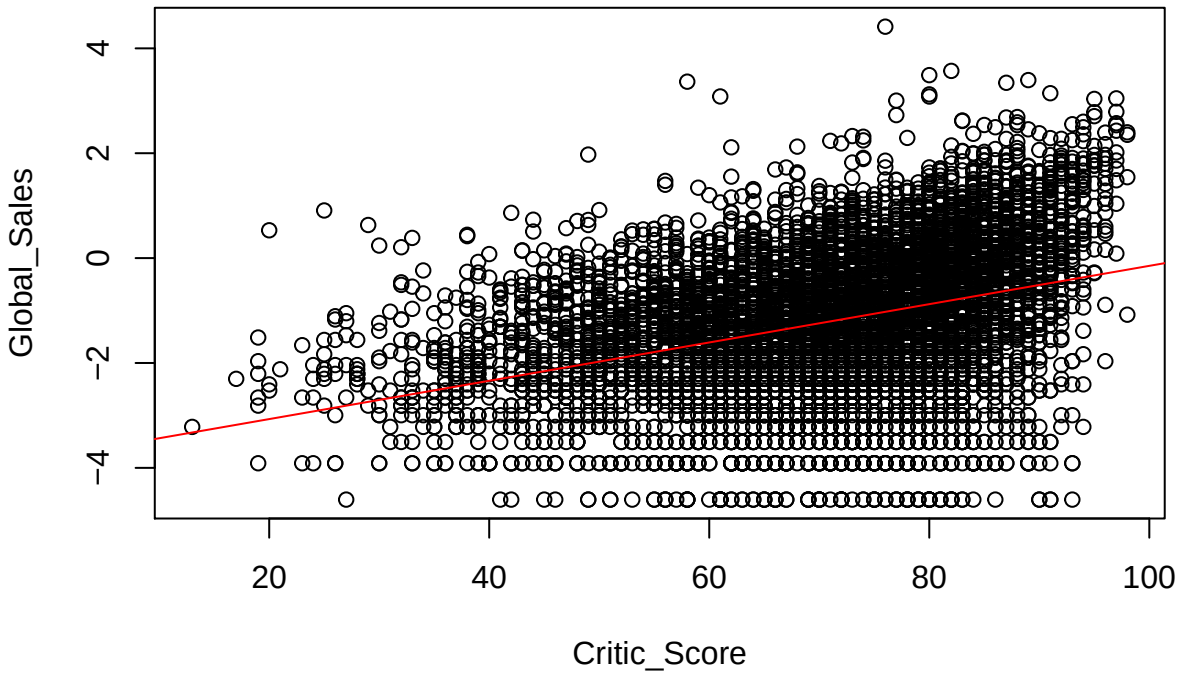


Comments:

There are possible linear relationships between Global_Sales and Critic_Count, Critic_Score and User_Score. More detailed:

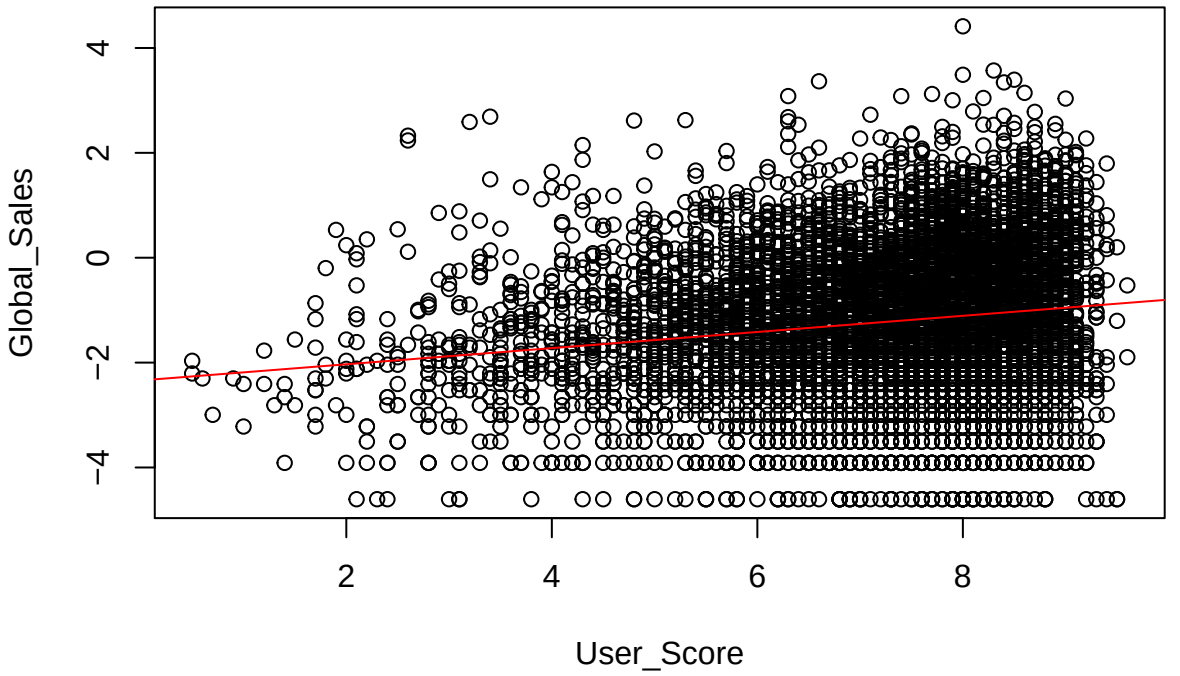
2. Critic_Score on Global_Sales

```
plot(X$Critic_Score, X$Global_Sales,
     xlab = 'Critic_Score', ylab = 'Global_Sales')
abline(lm(X$Global_Sales ~ X$Critic_Score), col = 'red')
```



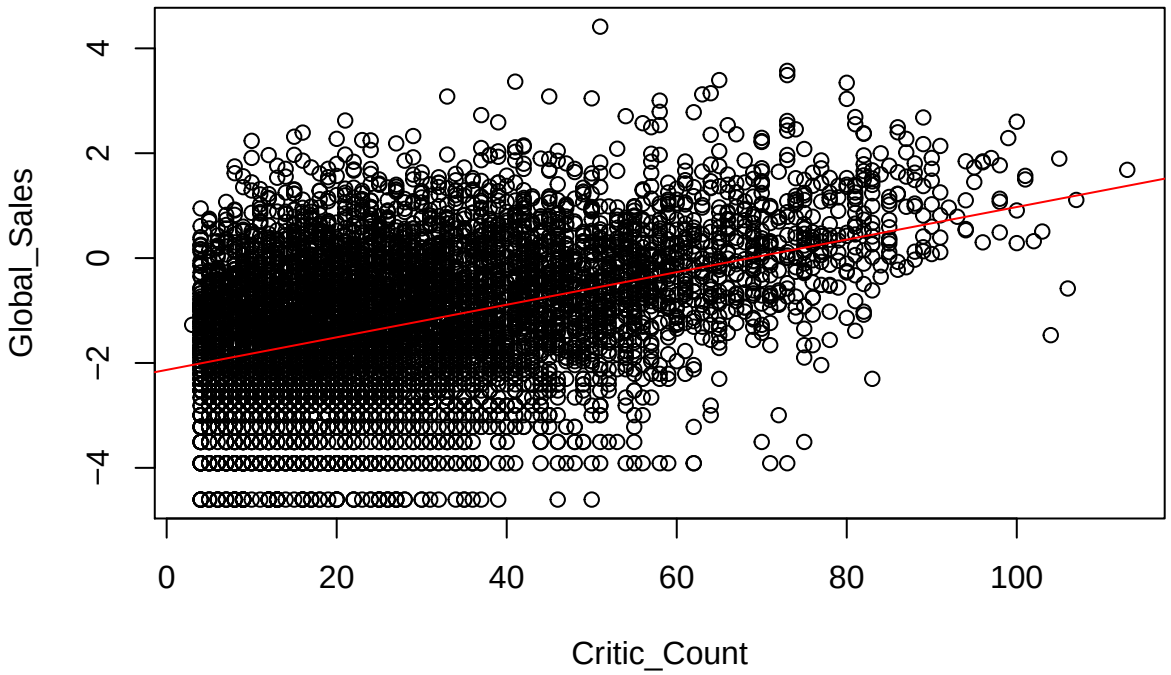
3. User_Score on Global_Sales

```
plot(X$User_Score, X$Global_Sales,
     xlab = 'User_Score', ylab = 'Global_Sales')
abline(lm(X$Global_Sales ~ X$User_Score), col = 'red')
```



4. Global Sales by Critic_Count

```
plot(X$Critic_Count, X$Global_Sales,
     xlab = 'Critic_Count', ylab = 'Global_Sales')
abline(lm(X$Global_Sales ~ X$Critic_Count), col = 'red')
```



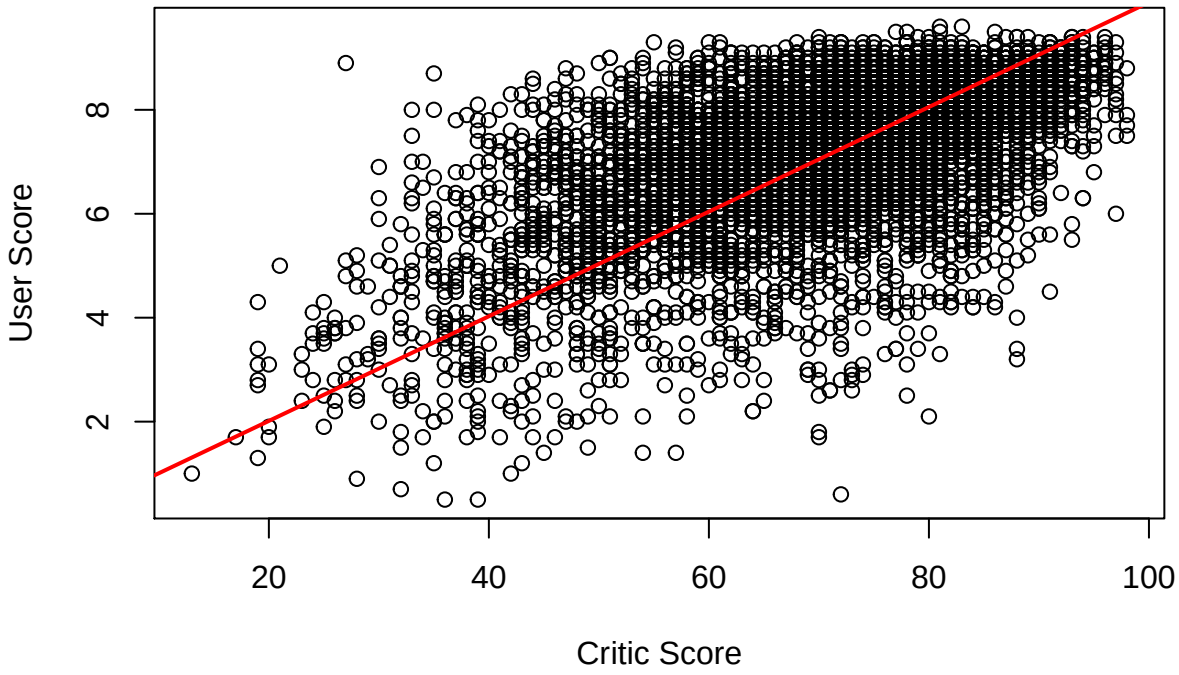
5. Check for possible collinearity between numerical covariates:

```
cor(X[,-1])
```

```
##                    Year_since_Release Critic_Count Critic_Score
## Year_since_Release        1.000000000   -0.2033363  0.007660526
## Critic_Count             -0.203336345    1.0000000  0.396478161
## Critic_Score              0.007660526    0.3964782  1.000000000
## User_Count               -0.199347540    0.3656026  0.265638706
## User_Score                0.253913721    0.1950873  0.580318371
##                     User_Count User_Score
## Year_since_Release -0.19934754 0.25391372
## Critic_Count        0.36560257 0.19508730
## Critic_Score        0.26563871 0.58031837
## User_Count          1.00000000 0.01754604
## User_Score          0.01754604 1.00000000
```

There is possible **collinearity** between Critic_Score and User_Score. If users give their scores after reviwing the critic score for that game, it's somehow biased and there is an obvious impact of critic score on the user score. This can be also checked by simple linear model fit:

```
plot(X$Critic_Score, X$User_Score,
     xlab = 'Critic Score', ylab = 'User Score')
#fit without bias term (intercept)
abline(lm(X$User_Score ~ X$Critic_Score - 1),
       col = 'red', lwd = 2)
```



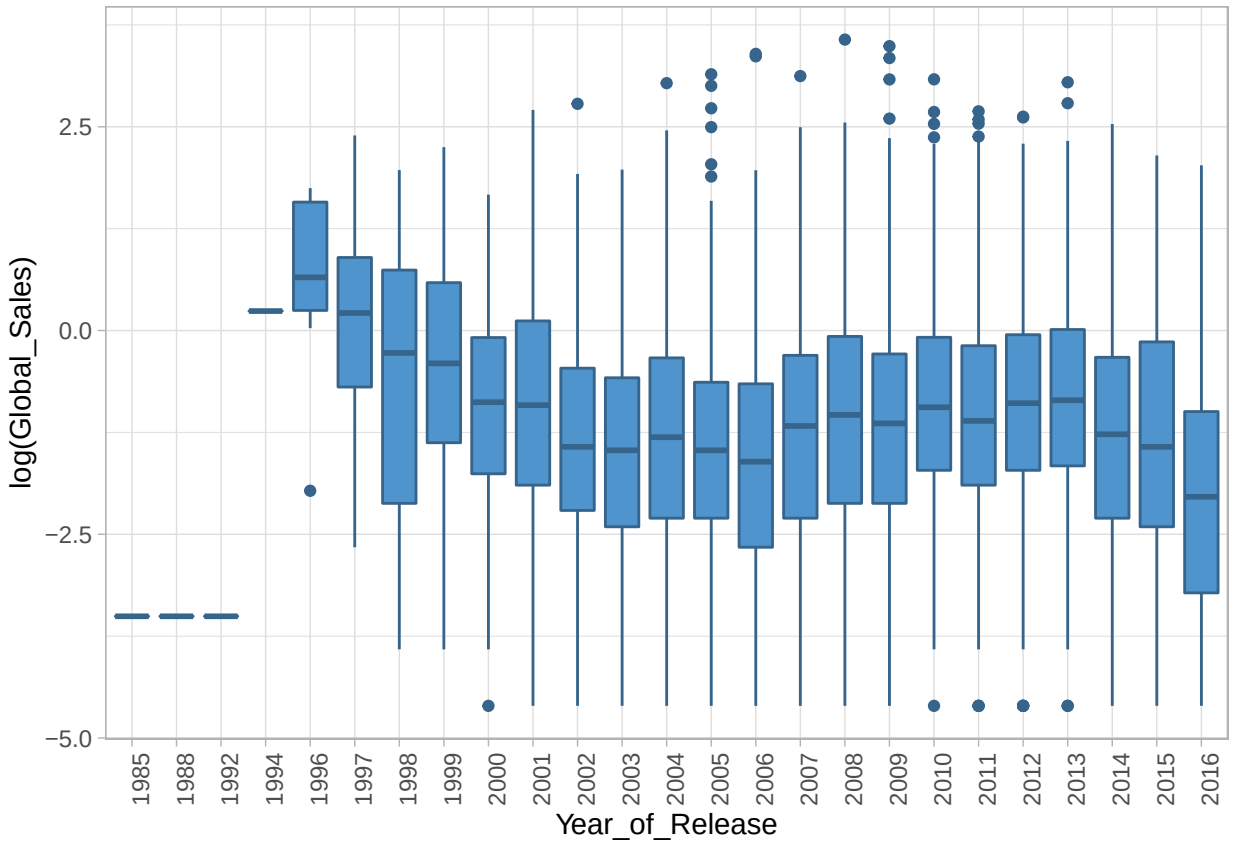

```
lm1 <- lm(X$User_Score ~ X$Critic_Score - 1)
lm1 %>% summary()
```

```
##
## Call:
## lm(formula = X$User_Score ~ X$Critic_Score - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.6490 -0.5619  0.1470  0.8660  6.1816
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## X$Critic_Score 0.1006800  0.0002205   456.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.305 on 6824 degrees of freedom
## Multiple R-squared:  0.9683, Adjusted R-squared:  0.9683
## F-statistic: 2.086e+05 on 1 and 6824 DF,  p-value: < 2.2e-16
```

Fitting the linear model without an intercept results adjusted in R-square = 0.9683, which is a quite big amount. We can, further reveal if this possible collinearity if impacting the model fit in a bad way by analyzing the residual plots.

5. Did Global sales change by year?

```
games %>% filter(Global_Sales < 60) %>%
    ggplot(aes(x = as.factor(Year_of_Release), y = log(Global_Sales))) +
    geom_boxplot(fill = 'steelblue3', col = 'steelblue4') +
    theme_light() +
    xlab('Year_of_Release') +
    theme(axis.text.x = element_text(angle = 90))
```



Significantly noticeable sales started in 1996, with 7 sales records in that year. Before 1996, there are only 4 records in 1985, 1988, 1991 and 1992. Visually there isn't much significant change in average global sales over time from 1996 to 2016. However, to be sure, we should run statistical tests in order sto ap/prove it.

Melt variables

In order to include region as a new variable, we should melt sales by region:

```
# exclude global sales, if you will need them, you can include them later:
sales <- melt(games[,c(-1, -10)],
              measure.vars = c('NA_Sales', 'EU_Sales',
                               'JP_Sales', 'Other_Sales'))
names(sales) <- c('Platform', 'Year_of_Release', 'Genre', 'Publisher',
                  'Critic_Score', 'Critic_Count', 'User_Score',
                  'User_Count', 'Developer', 'Rating', 'Year_since_Release',
                  'Decade', 'Platform_Company', 'Platform_Gen',
                  'Main_Developer', 'Developer_Country', 'Family_Platform',
                  'Main_Publisher', 'Region', 'Sales')

# write this variable into csv file
#write.csv(games, 'games.csv', row.names = FALSE)

# write games variable into csv file
#write.csv(sales, 'sales.csv', row.names = FALSE)
```

Interaction plots

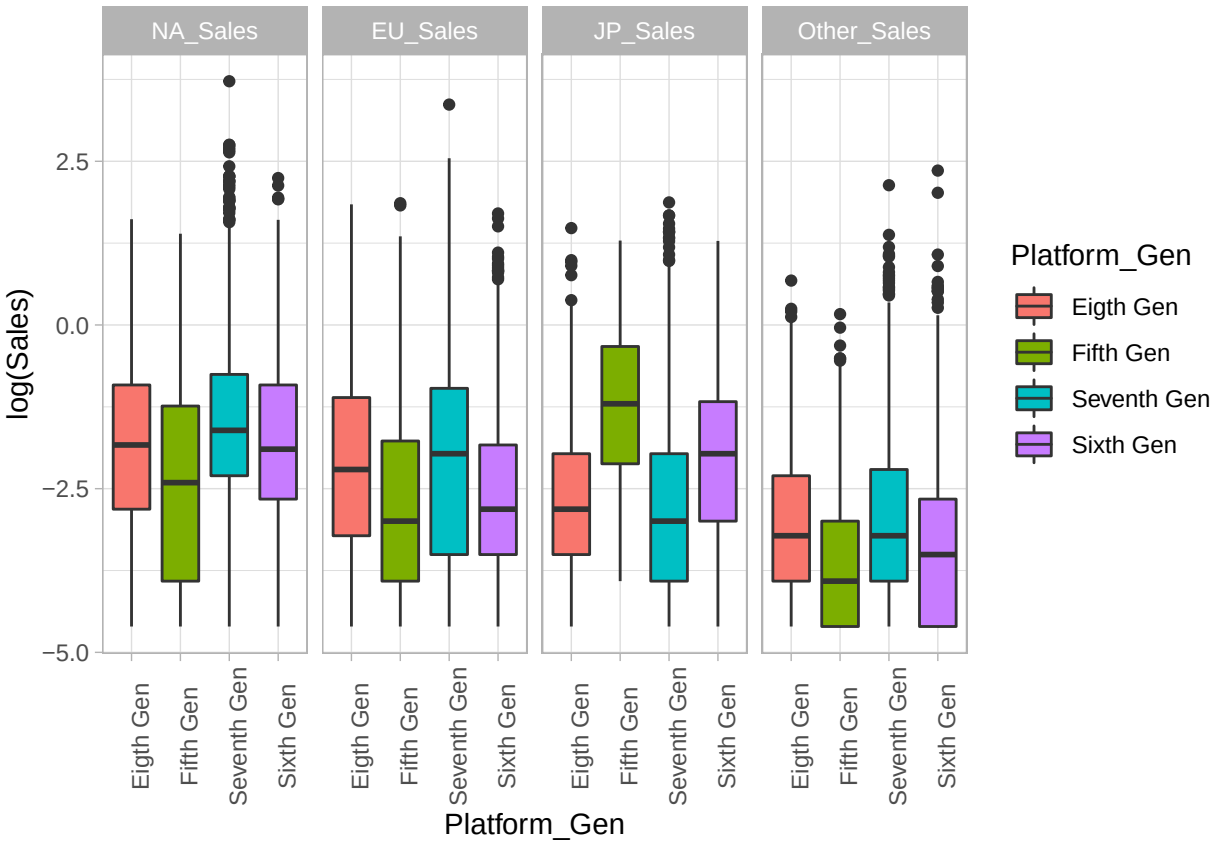
1. Sales by Platform_Gen and Region

```
# (!!!) before transforming the sales, add the minimum sales count for each 0 # value however, doesn't

#min_non_zer_sales <- min(sales$Sales[sales$Sales != 0])
#sales$Sales[sales$Sales == 0] <- min_non_zer_sales
```

```
sales %>% ggplot(aes(y = log(Sales), x = Platform_Gen)) +
    geom_boxplot(aes(fill = Platform_Gen)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```

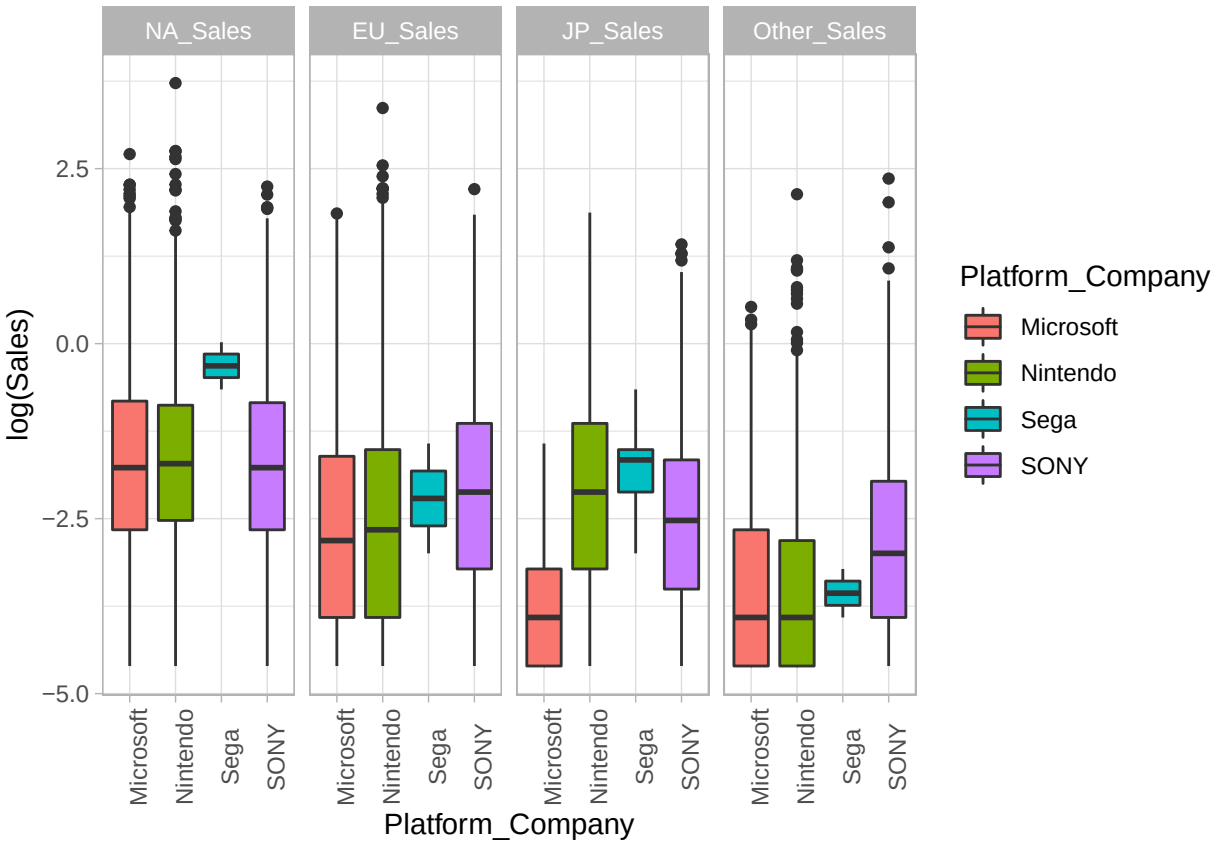


Mosaic plot is already in Meltem's version

2. Sales by Platform_Company and Region

```
sales %>% ggplot(aes(y = log(Sales), x = Platform_Company)) +
    geom_boxplot(aes(fill = Platform_Company)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

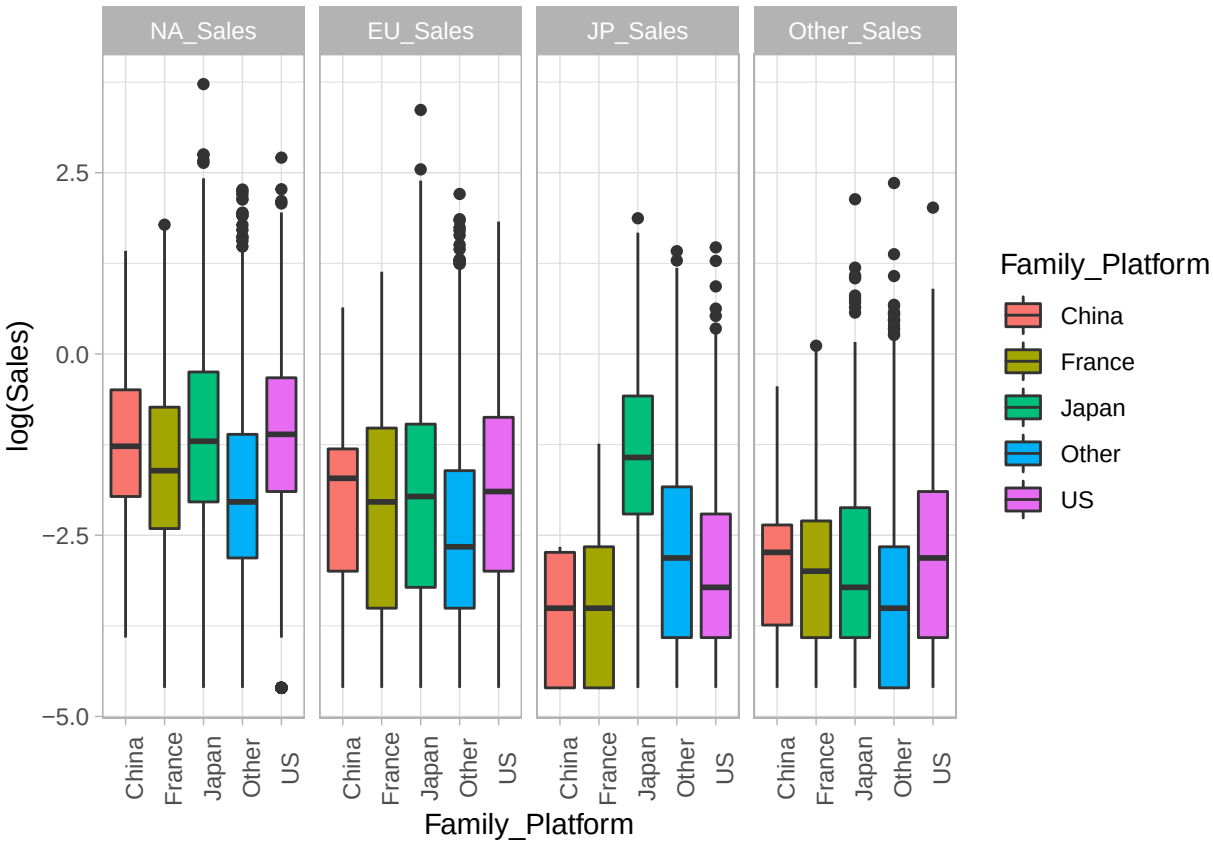
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



3. Sales by Family_Platform and Region

```
sales %>% ggplot(aes(y = log(Sales), x = Family_Platform)) +
    geom_boxplot(aes(fill = Family_Platform)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

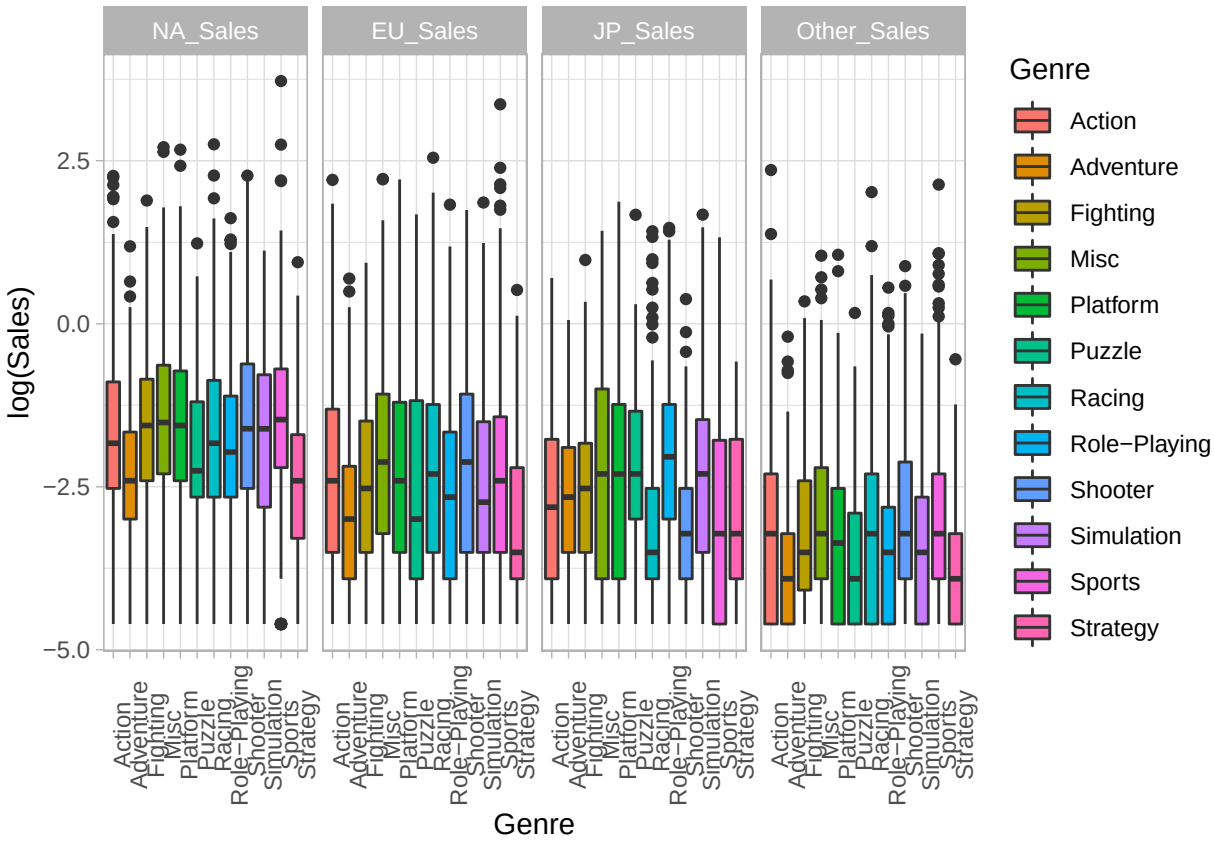
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



4. Sales by Genre and Region

```
sales %>% ggplot(aes(y = log(Sales), x = Genre)) +
    geom_boxplot(aes(fill = Genre)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

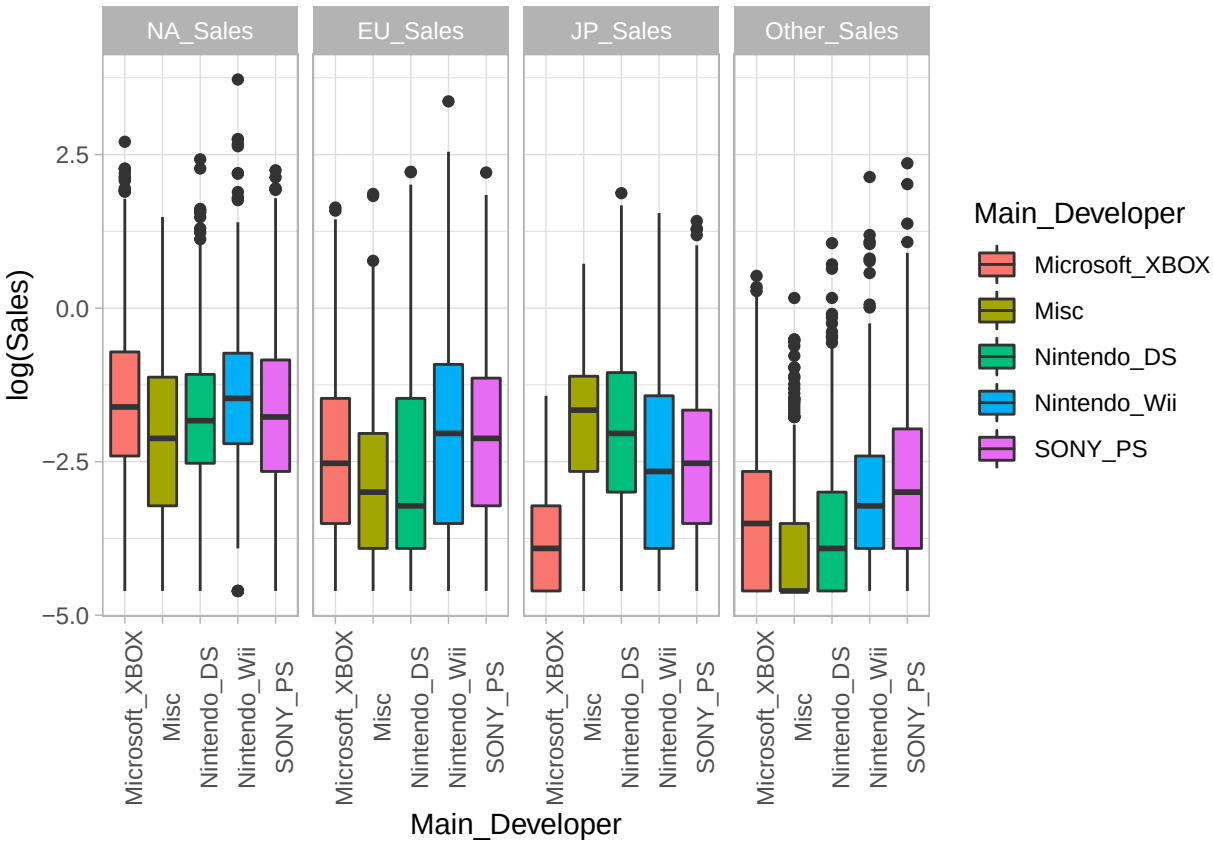
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



5. Sales by Main_Developer and Region

```
sales %>% ggplot(aes(y = log(Sales), x = Main_Developer)) +
    geom_boxplot(aes(fill = Main_Developer)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

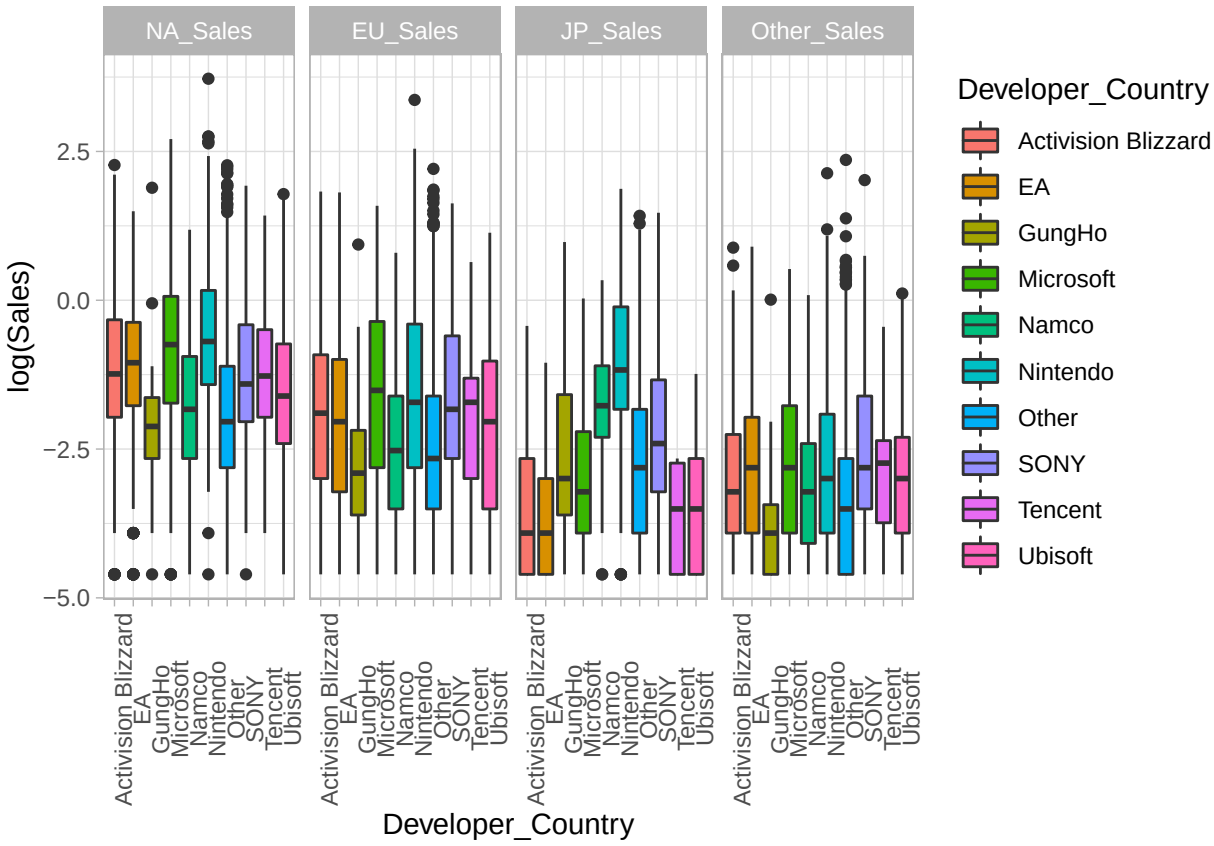
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



6. Sales by Developer_Country and Region ??? makes sense???

```
sales %>% ggplot(aes(y = log(Sales), x = Developer_Country)) +
    geom_boxplot(aes(fill = Developer_Country)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

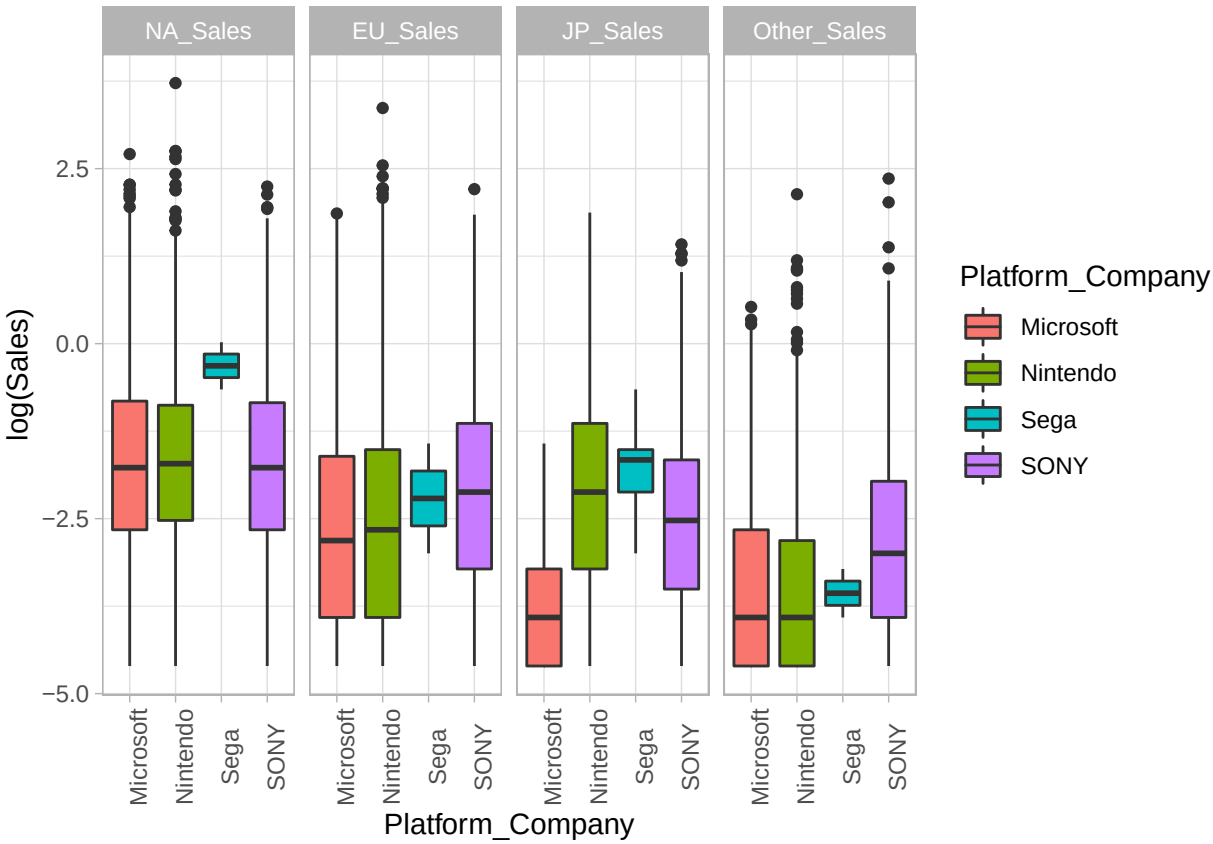
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



7. Sales by Main_Developer and Platform_Compnany

```
sales %>% ggplot(aes(y = log(Sales), x = Platform_Company)) +
    geom_boxplot(aes(fill = Platform_Company)) +
    facet_grid(.~Region) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

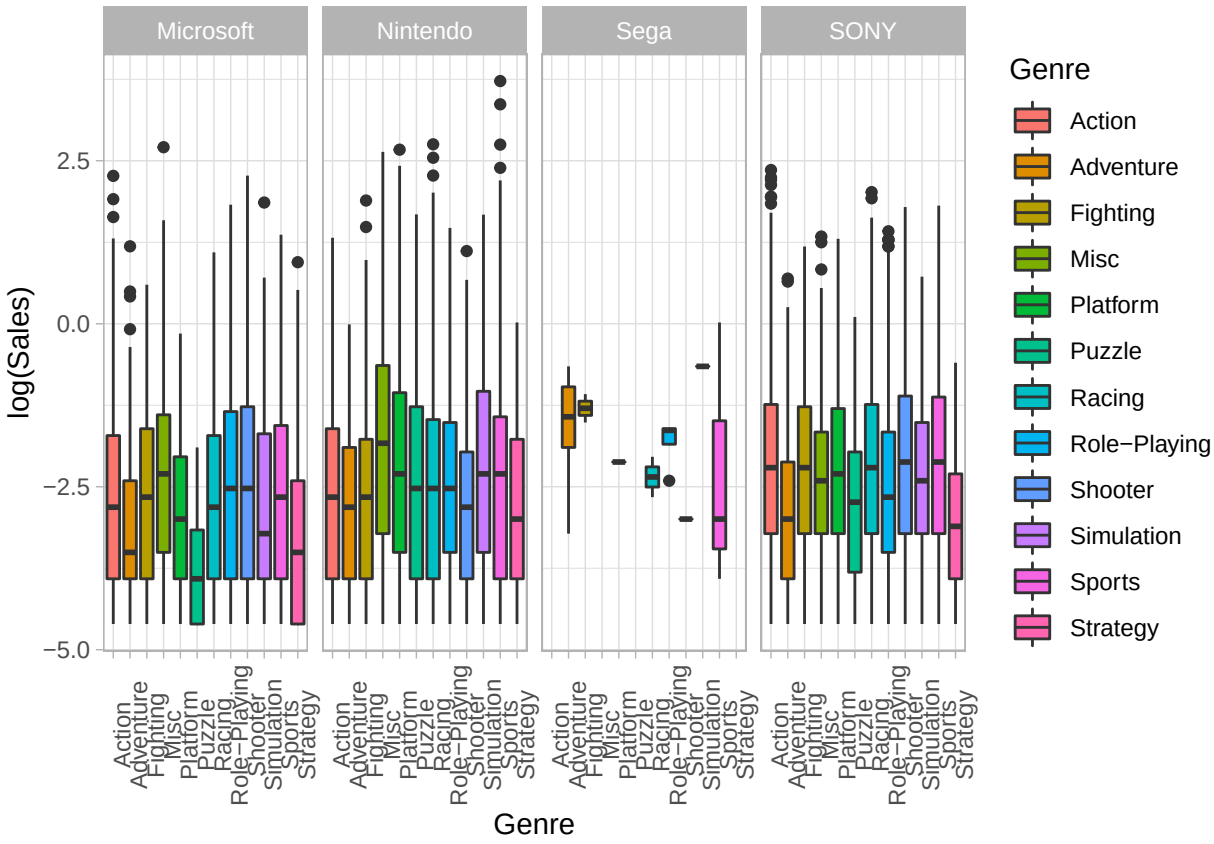
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



8. Sales by Genre and Platform_Company

```
sales %>% ggplot(aes(y = log(Sales), x = Genre)) +
    geom_boxplot(aes(fill = Genre)) +
    facet_grid(.~Platform_Company) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

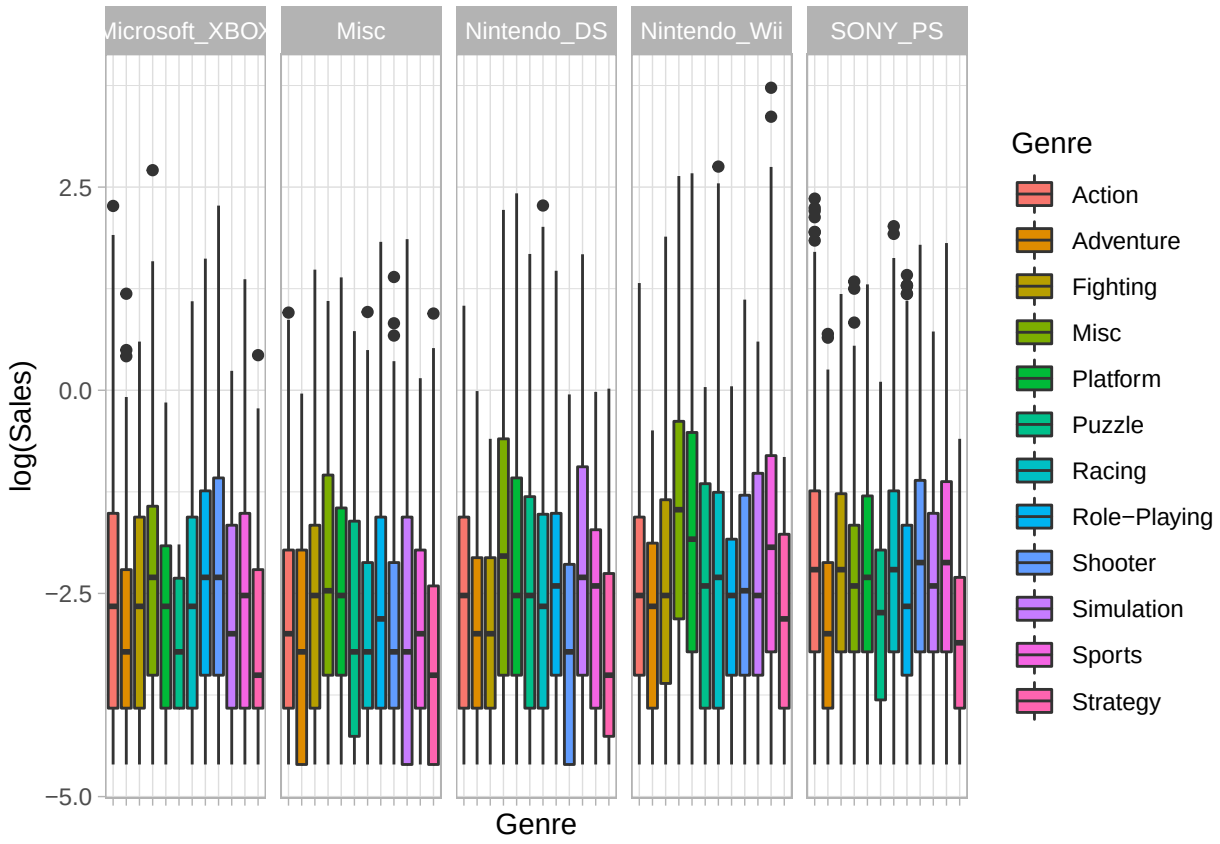
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



9. Sales by Genre and Main_Developer

```
sales %>% ggplot(aes(y = log(Sales), x = Genre)) +
    geom_boxplot(aes(fill = Genre)) +
    facet_grid(.~Main_Developer) +
    theme_light() +
    theme(axis.text.x = element_blank())
```

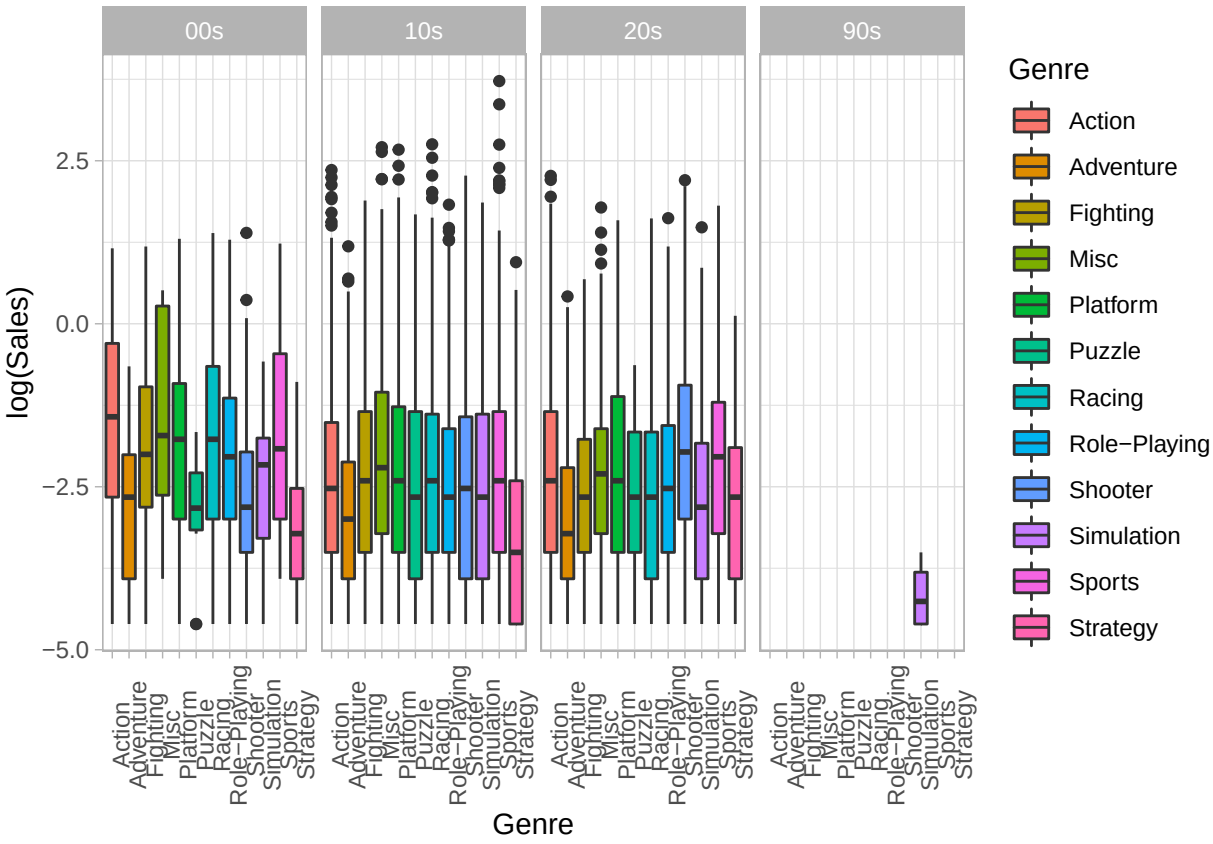
```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



10. Sales by Genre and Decade

```
sales %>% ggplot(aes(y = log(Sales), x = Genre)) +
    geom_boxplot(aes(fill = Genre)) +
    facet_grid(.~Decade) +
    theme_light() +
    theme(axis.text.x = element_text(angle = 90))
```

```
## Warning: Removed 7506 rows containing non-finite values (stat_boxplot).
```



Comments: We can keep all this example plots, so that we can choose *some* of them for our report later.

##. Independence of Categorical variables