# Capstone - Predicting Video Game Sales

Lionel Fournier

14/06/2020

## Contents

# 1 Overview

In a few decades, video games have reached a major position in the entertainement industry. If there are still small independent games made by one person, today standard is the triple A, that cost huge amount of money and need big team to be made. It's important to have an idea of how you can optimise sales before taking such a big risk. Data science can help you with that.

# 2 Introduction

In this project, we are going to use machine learning to predict the ratings users. We are going to use a dataset containing a list of video games with sales greater than 100,000 copies. It was generated by a scrape of vgchartz.com. This data set contains video games released before 2017. We are going to first prepare the data and then analyse it. We then will construct our machine algorithm, starting trying different method to achieve better result.

# 3 Methodology

First we are going to download and prepare the data. Then we are going to analyse them. The goal is to dive in some element that may affect our different models.

We are specificaly going to look into global sales and how the different variables impact it. Different gaming systems, number of critics, the score of the games with critics and users can all affect global sales. Different genres may also have an influence on the global sales.

To take into account critic and user scores, we are going to have to limit the dataset. Unfortunately, there are a lot of missing values for those variables, and we want to be able to use them. Although, the set will remain large enough with more almost 7000 entries.

From there, we are going to create our models, taking into account what we learned in our analysis. First we are going to split it into a testing set and a training one. Then we are going to look at which predictors are the most relevant and then we are going to use different machine learning techniques such as linear model, generelized linear model and random forest.

To test the accuracy of each model, we are going to use the residual mean squared error (RMSE). The lower the value will be the better.

# 4 Data preparation

## 4.1 Creation of the data set

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(cowplot)) install.packages("RCurl", repos = "http://cran.us.r-project.org")
if(!require(olsrr)) install.packages("RCurl", repos = "http://cran.us.r-project.org")
if(!require(RCurl)) install.packages("RCurl", repos = "http://cran.us.r-project.org")

#This script was made on R 3.6.2
#Download the data from my github repo
dl <-getURL("https://raw.githubusercontent.com/lifnr/video_games_machine_learning/master/Video_Games_Sa
Video_Games<-read.csv(text = dl)
vg<-Video_Games
```

## 4.2 Data preparation

Our first step will be to have a quick look at the data. Then make some minor changes for ease of use. As mentioned, one problem we face is missing users and critics data. For now, we are going to keep everything. We are going to remove all the rows with NA later in the process to be able to use the full extent of the data.

```
#Checking the first entries of the edx dataset
head(vg)
```

```
##                       Name Platform Year_of_Release        Genre Publisher
## 1               Wii Sports      Wii            2006       Sports  Nintendo
## 2        Super Mario Bros.      NES            1985     Platform  Nintendo
## 3           Mario Kart Wii      Wii            2008       Racing  Nintendo
## 4        Wii Sports Resort      Wii            2009       Sports  Nintendo
## 5 Pokemon Red/Pokemon Blue       GB            1996 Role-Playing  Nintendo
## 6                  Tetris       GB            1989       Puzzle  Nintendo
##   NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales Critic_Score Critic_Count
## 1    41.36    28.96     3.77        8.45        82.53           76           51
## 2    29.08     3.58     6.81        0.77        40.24           NA           NA
## 3    15.68    12.76     3.79        3.29        35.52           82           73
## 4    15.61    10.93     3.28        2.95        32.77           80           73
## 5    11.27     8.89    10.22        1.00        31.37           NA           NA
## 6    23.20     2.26     4.22        0.58        30.26           NA           NA
##   User_Score User_Count Developer Rating
## 1          8        322  Nintendo      E
## 2                   NA
## 3        8.3        709  Nintendo      E
## 4          8        192  Nintendo      E
## 5                   NA
## 6                   NA
```

```r
#Checking the structure of the data
str(vg)
```

```
## 'data.frame':    16719 obs. of  16 variables:
##  $ Name           : Factor w/ 11563 levels "","'98 Koshien",..: 11059 9406 5573 11061 7411 9771 6693
##  $ Platform       : Factor w/ 31 levels "2600","3DO","3DS",..: 26 12 26 26 6 6 5 26 26 12 ...
##  $ Year_of_Release: Factor w/ 40 levels "1980","1981",..: 27 6 29 30 17 10 27 27 30 5 ...
##  $ Genre          : Factor w/ 13 levels "","Action","Adventure",..: 12 6 8 12 9 7 6 5 6 10 ...
##  $ Publisher      : Factor w/ 582 levels "10TACLE Studios",..: 371 371 371 371 371 371 371 371 371 37
##  $ NA_Sales       : num  41.4 29.1 15.7 15.6 11.3 ...
##  $ EU_Sales       : num  28.96 3.58 12.76 10.93 8.89 ...
##  $ JP_Sales       : num  3.77 6.81 3.79 3.28 10.22 ...
##  $ Other_Sales    : num  8.45 0.77 3.29 2.95 1 0.58 2.88 2.84 2.24 0.47 ...
##  $ Global_Sales   : num  82.5 40.2 35.5 32.8 31.4 ...
##  $ Critic_Score   : int  76 NA 82 80 NA NA 89 58 87 NA ...
##  $ Critic_Count   : int  51 NA 73 73 NA NA 65 41 80 NA ...
##  $ User_Score     : Factor w/ 97 levels "","0","0.2","0.3",..: 79 1 82 79 1 1 84 65 83 1 ...
##  $ User_Count     : int  322 NA 709 192 NA NA 431 129 594 NA ...
##  $ Developer      : Factor w/ 1697 levels "","10tacle Studios",..: 1035 1 1035 1035 1 1 1035 1035 103
##  $ Rating         : Factor w/ 9 levels "","AO","E","E10+",..: 3 1 3 3 1 1 3 3 3 1 ...
```

```r
#Checking the dimension of the data
dim(vg)
```

```
## [1] 16719    16
```

```r
#Number of NA in the dataset
sum(is.na(vg))
```

```
## [1] 26293
```

```r
#Changing the year into integers and user score as numeric
vg<-vg%>%mutate(Year_of_Release=as.integer(as.character(Year_of_Release)), User_Score=as.numeric(as.cha
```

We also want to create a new column with the gaming system. The plateform is mentionned for every game, but we can gather that in 4 big family of gaming systems (Nintendo, Sony, Xbox and PC) and older ones like Sega and others.

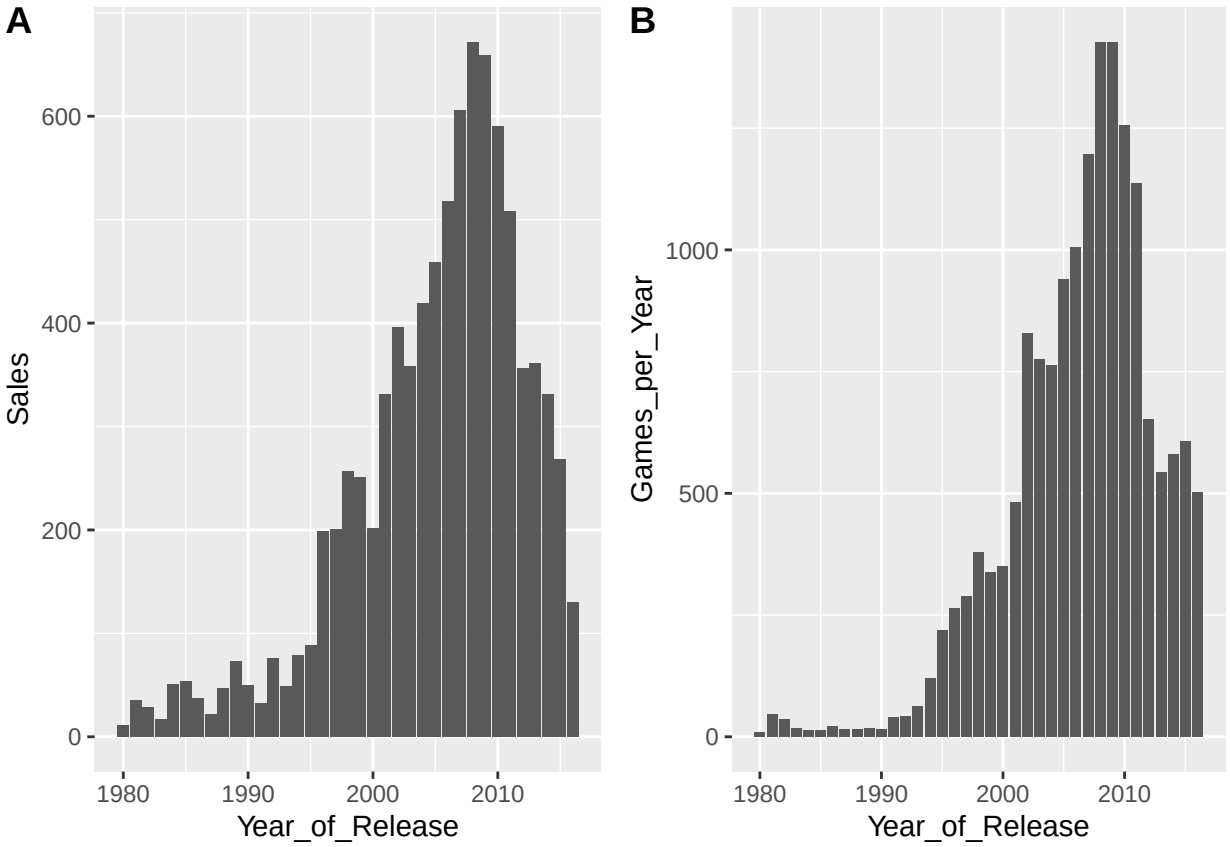
```r
nes <- c("3DS","DS","GB","GBA","N64","GC", "NES","SNES","Wii","WiiU")
sony <- c("PS","PS2","PSP","PS3","PS4","PSV")
sega <- c("GEN","SCD","DC","GG","SAT")
xbox <- c("XB","X360", "XOne")
other <- c("2600","3DO","NG","PCFX","TG16","WS")
pc <- c("PC")

vg$System[vg$Platform %in% nes] <- "Nintendo"
vg$System[vg$Platform %in% sony] <- "Sony"
vg$System[vg$Platform %in% xbox] <- "XBox"
vg$System[vg$Platform %in% sega] <- "Sega"
vg$System[vg$Platform %in% pc] <- "PC"
vg$System[vg$Platform %in% other] <- "Other"
```

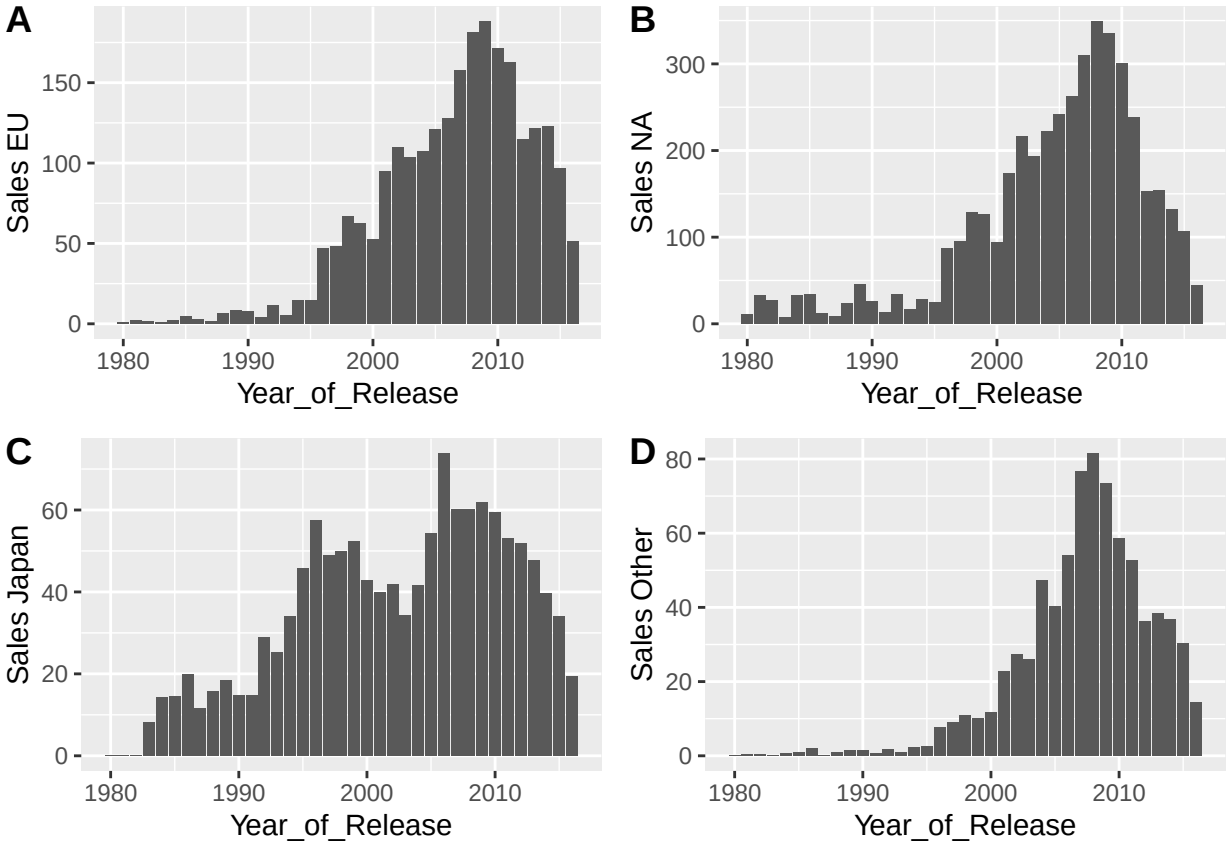
# 5 Analysis

Before we start building our models, we need to analyse our data in order to better understand the importance of different predictors.

## 5.1 Sales




We can see that sales have been growing since the 1980's. They reached a peak around 2010. The number of games and the sales have droped after that. We can make the hypothesis that the financial crisis of 2008 had an impact on many households which may have limited the number of games bought by many people. As a side note, when the data will be available, it would be interesting to see how the coronavirus pandemic has affected the video game sales. The lockdown has given an incentive for people to find an occupation at home. Such as, playing video games, and at the moment of writing this report, there is a shortage of Nintendo Switch.

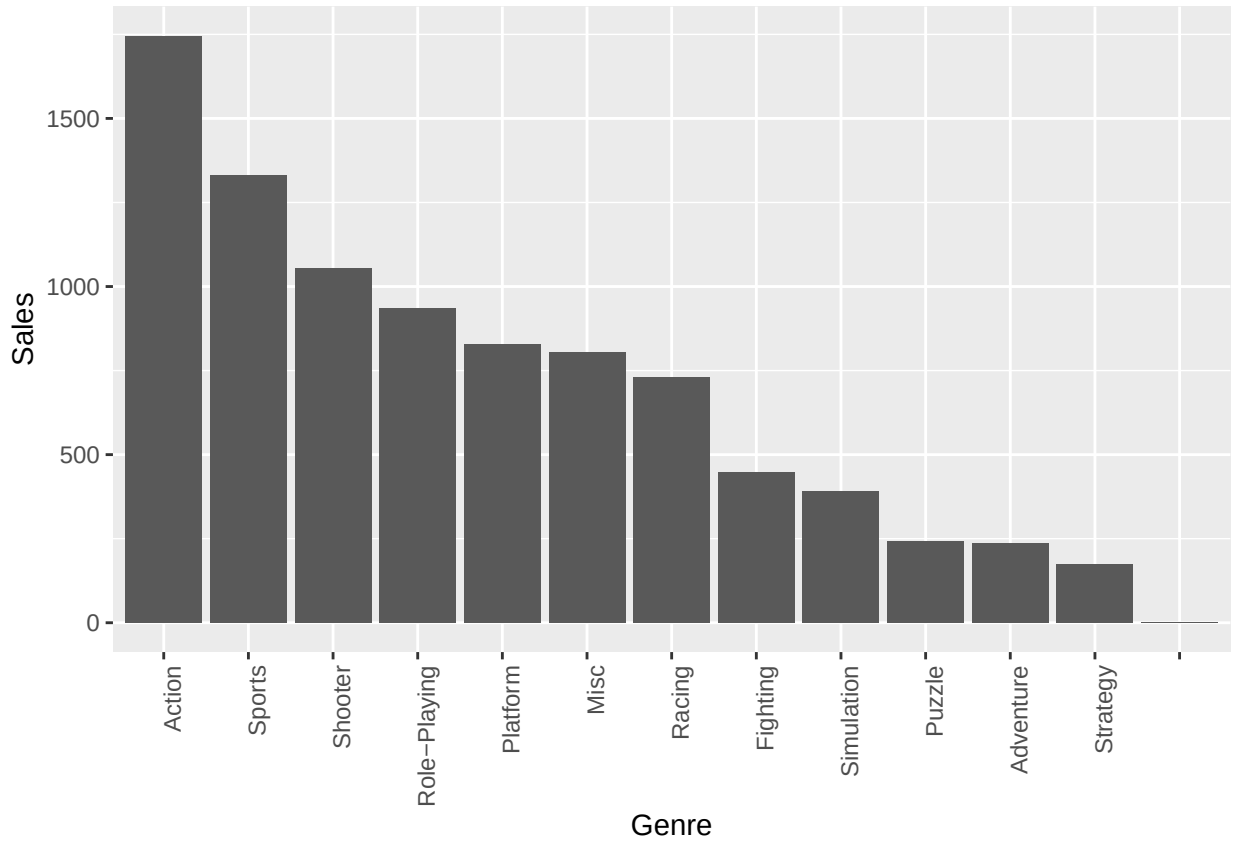



The down trend after 2010 is noticeable in the four sales regions included in the data. Altough it's interesting to notice Japan particularity. Video games seem to be way more largely sold there, and it's logical with some of the main video game companies based in Japan. This country also had a less steep curve after 2010, losing sales more gradualy.

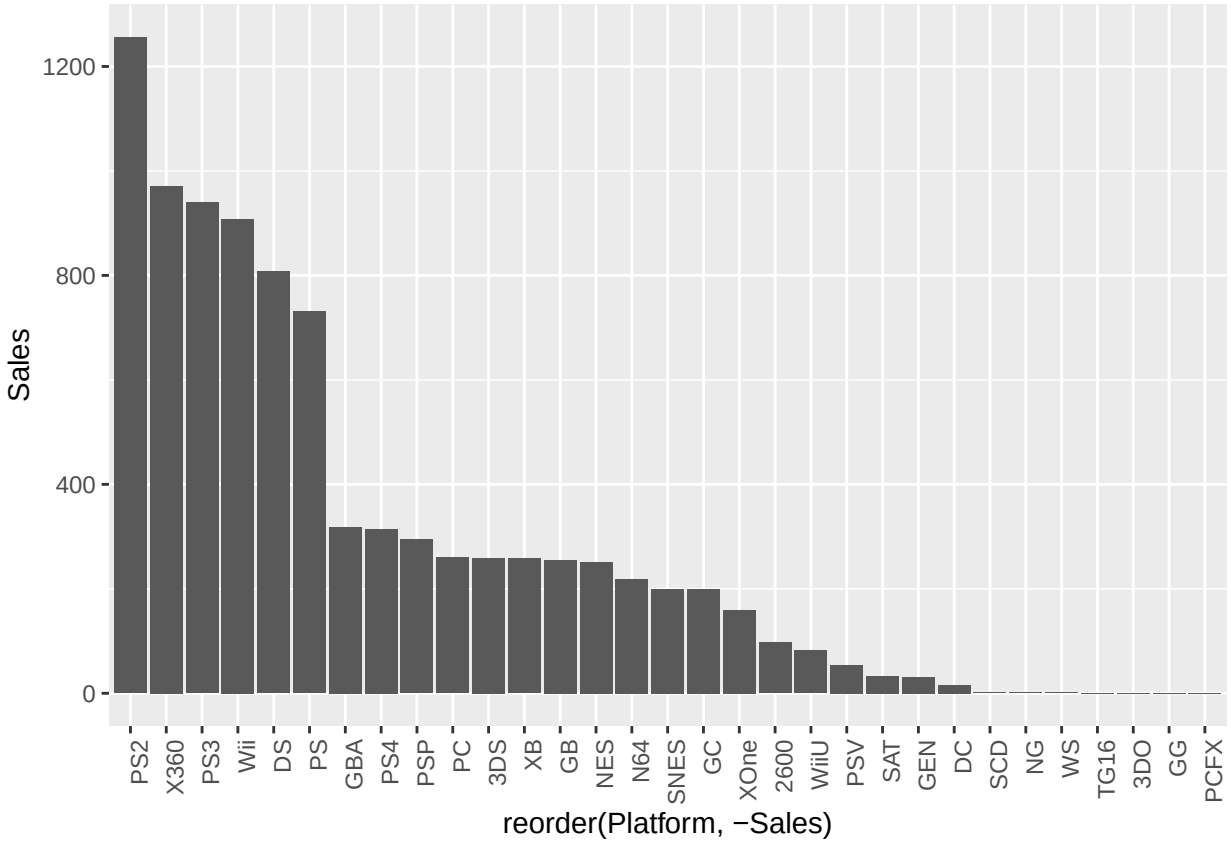
This market, while very important, is smaller than the European one. But the main one remains the North America one, with more sales than all the others combined at the peak of the curve. It would be wise to take that market in consideration when making a video game.

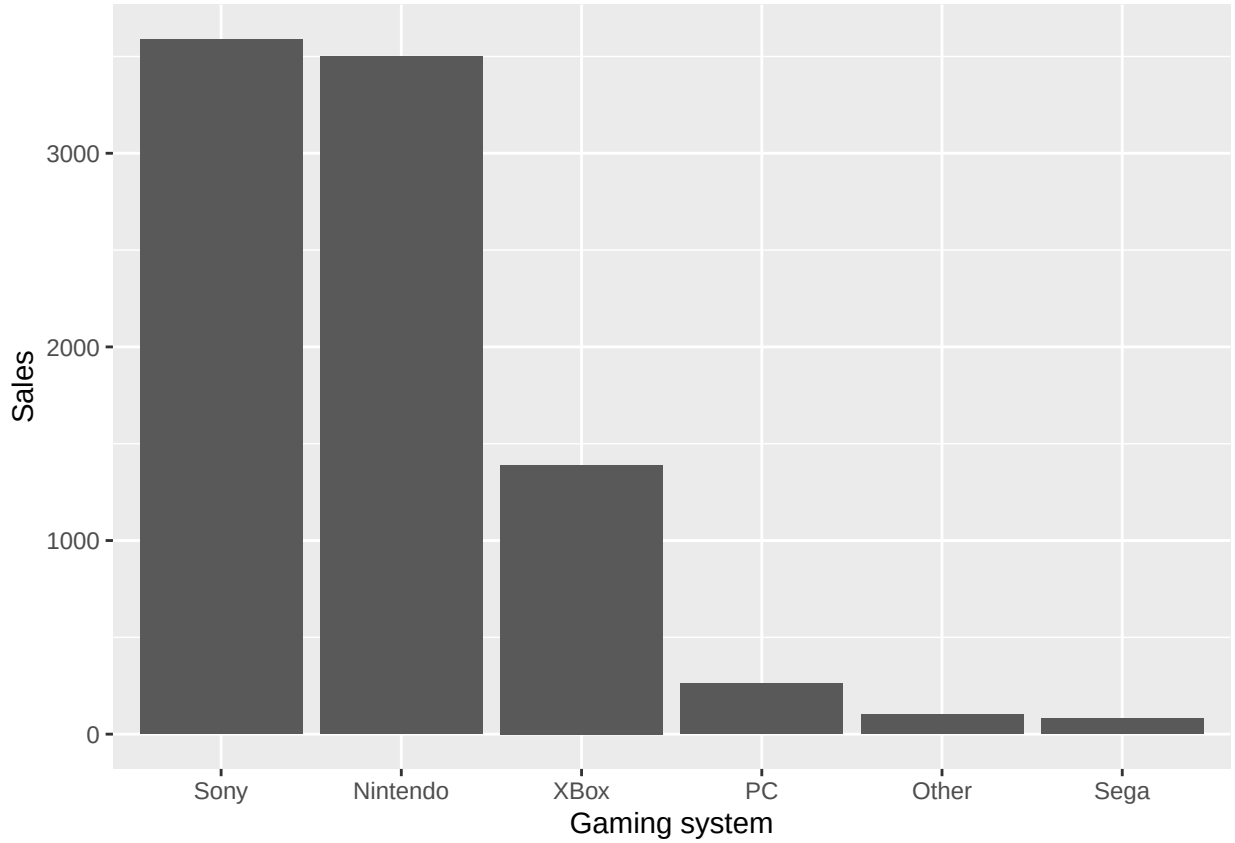
## 5.2 Genre and plateform

Diffrent aspects are going to influence the number of sales. Some genres are more popular, while others aim at a more niche audience. The plateform also has an influence. Realising a game only in one system may limit the number of units sold, especially if the plateform hasn't met a huge success.




We can see that the most sales are made for games within action, sports and shooters genres. On the other side, strategy or adventure games have lower sale numbers. Those games do have a strong community, but require more engagement, and seem to be more of a niche market.


Sales
1200
800
400
0
PS2
X360
PS3
Wii
DS
PS
GBA
PS4
PSP
PC
3DS
XB
GB
NES
N64
SNES
GC
XOne
2600
WiiU
PSV
SAT
GEN
DC
SCD
NG
WS
TG16
3DO
GG
PCFX
reorder(Platform, −Sales)




Plateform matters, not for a question of console wars, but for sales. We can see that Sony and Nintendo game consoles are two platforms where you can sell more game. It's also interesting to see that PC is kind of far away in term of sales. It does require a larger investment to buy a gaming pc, while a gaming consoles may be a fraction of the price. Although, to maximize sales, a lot of editors choose to release their games on almost every current platform.

```
## # A tibble: 582 x 2
##    Publisher                     count
##    <fct>                         <dbl>
##  1 Nintendo                      1789.
##  2 Electronic Arts               1117.
##  3 Activision                     731.
##  4 Sony Computer Entertainment    606.
##  5 Ubisoft                        472.
##  6 Take-Two Interactive           404.
##  7 THQ                            338.
##  8 Konami Digital Entertainment   282.
##  9 Sega                           270.
## 10 Namco Bandai Games             255.
## # ... with 572 more rows
```

```
## # A tibble: 11,563 x 2
##    Name                          count
##    <fct>                         <dbl>
##  1 Wii Sports                     82.5
##  2 Grand Theft Auto V             56.6
```

```
##  3 Super Mario Bros.               45.3
##  4 Tetris                          35.8
##  5 Mario Kart Wii                  35.5
##  6 Wii Sports Resort               32.8
##  7 Pokemon Red/Pokemon Blue        31.4
##  8 Call of Duty: Black Ops         30.8
##  9 Call of Duty: Modern Warfare 3  30.6
## 10 New Super Mario Bros.           29.8
## # ... with 11,553 more rows
```

It may be smart to be on multiple consoles, although the most successful editor in term of sales is Nintendo, with exclusivity for its own gaming system. Six of the ten most sold games are from Nintendo.

## 5.3 User and critic effect

Now to analyse the impact of user and critic effect, we are going to remove all the rows with NA from the data set. This limited dataset will also be used for our modelling.

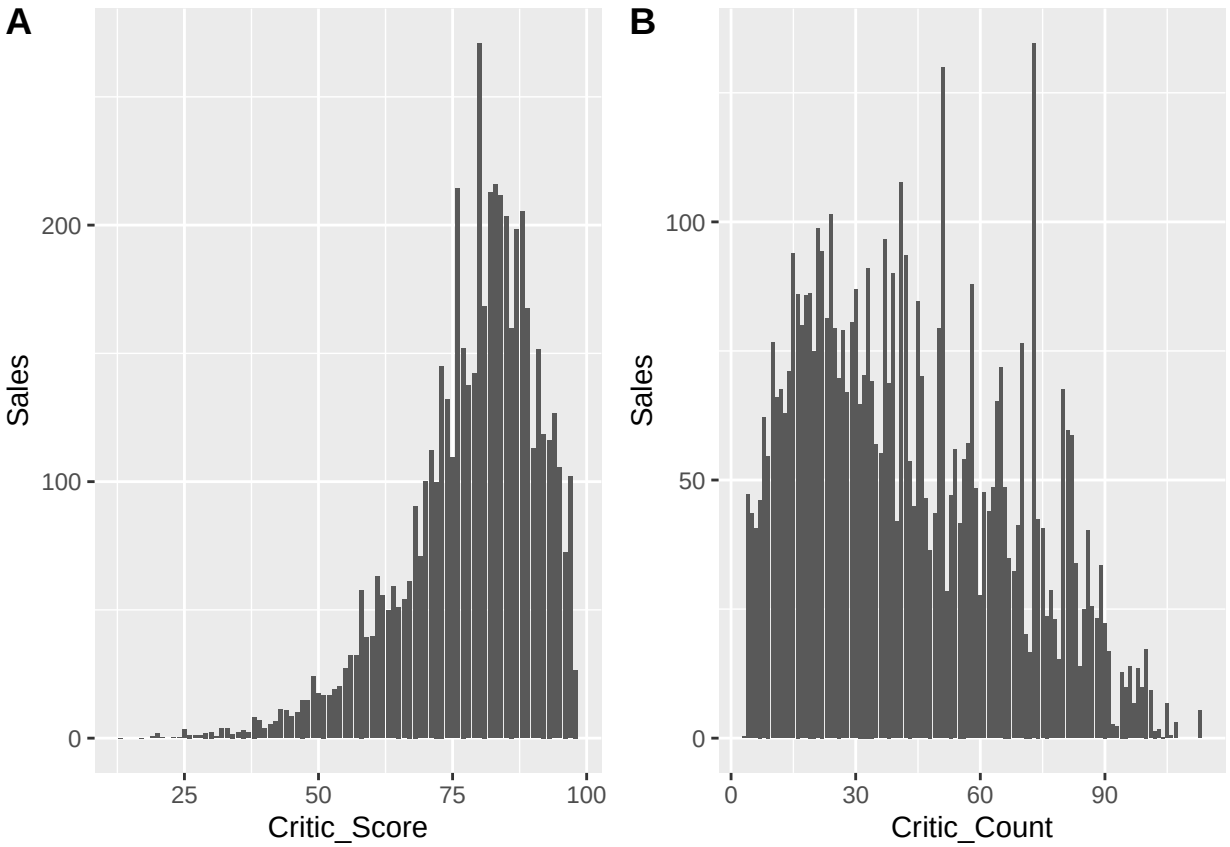
```
#Removing NA from the set and keeping only variable we are going to use
vg_short <- vg%>%filter(vg$Year_of_Release<=2016)%>%
  select(Global_Sales,Name,Year_of_Release,Genre, Critic_Score, Critic_Count,User_Score,User_Count,Syst
  na.omit(vg)
sum(is.na(vg_short))
```

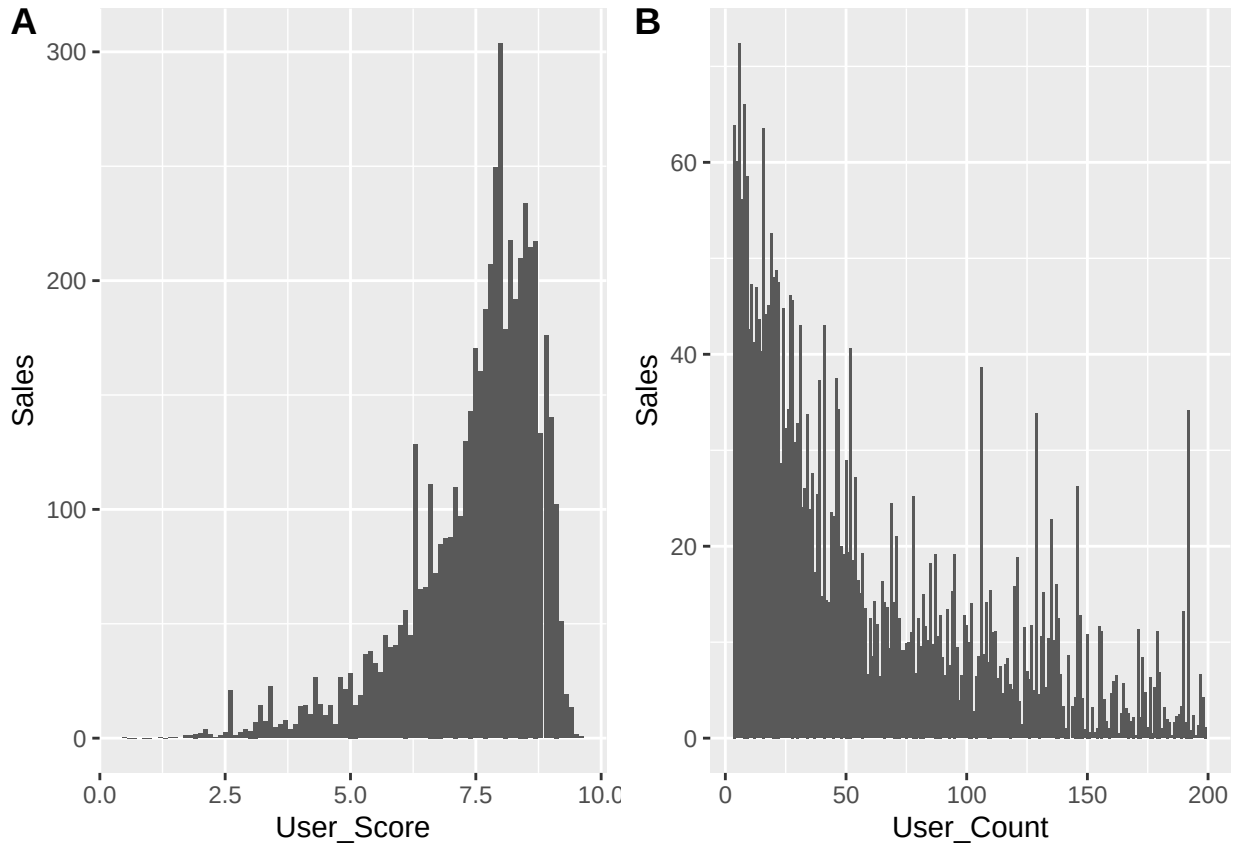
```
## [1] 0
```

```
dim(vg_short)
```

```
## [1] 6894    9
```

As we can see there are no more NA's, and the dataset is now 6894 rows long. While doing this step we also selected the column we are going to use in our model and removed the other ones.


A
Sales
200
100
0
25
50
75
100
Critic_Score
B
Sales
100
50
0
0
30
60
90
Critic_Count




# 6 Modeling

Based on our analysis, we are going to build few machine learning algorithms through a training set and a test set. Our goal is to achieve the lowest RMSE. We did most of the data preparation previously.

```
#Creating a test_set and a training set

set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = vg_short$Global_Sales, times = 1,
                                  p = 0.2, list = FALSE)
train_set <- vg_short[-test_index,]
test_set <- vg_short[test_index,]
```

## 6.1 Subset

Our first step will be to look which predictors may be useful for our data.

```
##                                 Best Subsets Regression
## ---------------------------------------------------------------------------------------------
## Model Index    Predictors
## ---------------------------------------------------------------------------------------------
##      1         Critic_Count
##      2         Critic_Count User_Count
```

```
##      3         Critic_Count User_Count System
##      4         Critic_Score Critic_Count User_Count System
##      5         Genre Critic_Score Critic_Count User_Count System
##      6         Year_of_Release Genre Critic_Score Critic_Count User_Count System
##      7         Year_of_Release Genre Critic_Score Critic_Count User_Score User_Count System
## ----------------------------------------------------------------------------------------
##
##                                                    Subsets Regression Summary
## ------------------------------------------------------------------------------------------------------------
##                        Adj.        Pred
## Model    R-Square    R-Square    R-Square      C(p)         AIC          SBIC          SBC
## ------------------------------------------------------------------------------------------------------------
##   1        0.0813      0.0811      0.0801    504.4489    22652.5723    7007.0093    22672.4169    19
##   2        0.1061      0.1058      0.1033    343.9780    22503.5733    6858.0038    22530.0328    19
##   3        0.1305      0.1296      0.1264    186.3060    22359.0337    6707.5648    22411.9526    18
##   4        0.1462      0.1451       0.142     85.9970    22261.0284    6609.6758    22320.5622    18
##   5        0.1563      0.1535      0.1488     21.8041    22217.3458    6546.1082    22349.6431    18
##   6        0.1589      0.1560      0.1511      6.7770    22202.3227    6531.1248    22341.2349    17
##   7        0.1610      0.1579      0.1531     -5.0000    22190.5103    6519.3552    22336.0374    17
## ------------------------------------------------------------------------------------------------------------
## AIC: Akaike Information Criteria
##  SBIC: Sawa's Bayesian Information Criteria
##  SBC: Schwarz Bayesian Criteria
##  MSEP: Estimated error of prediction, assuming multivariate normality
##  FPE: Final Prediction Error
##  HSP: Hocking's Sp
##  APC: Amemiya Prediction Criteria
```

We can see here that critic and user count may be our first predictors. That does make sense, as if they are more critics available, more people will be able to learn about the game and then buy it. However, our analysis was a bit sceptical about that effect, we are going to use it and add other variables in our model.

## 6.2 Linear model

Our first two models are going to use a linear model. At firest, we will only use user and critic count and then add the scores and system.

```
fit <- train(Global_Sales~Critic_Count+User_Count, data=train_set, method="lm")
trn <- predict(fit,train_set)
RMSE(trn, test_set$Global_Sales)
```

```
## [1] 1.94054
```

```
fit$results['RMSE']
```

```
##       RMSE
## 1 1.900543
```

```
fit2 <- train(Global_Sales~Critic_Score+Critic_Count+User_Score+User_Count+System, data=train_set, metho
trn2<- predict(fit2,train_set)
RMSE(trn2, test_set$Global_Sales)
```

```
## [1] 1.952353
```

```
fit2$results['RMSE']
```

```
##       RMSE
## 1 1.890291
```

The first model gives us a rmse of 1.900543. The second 1.890291 showing an improvement.

## 6.3 Generalized Linear Model

Now we will use Generalized Linear Model, with the same values from our second one with lm.

```
#GLM Model
glm_fit <- train(Global_Sales~Critic_Score+Critic_Count+User_Score+User_Count+System,
             data=train_set, method="glm")
trn_glm <- predict(glm_fit,train_set)
RMSE(trn_glm, test_set$Global_Sales)
```

```
## [1] 1.952353
```

```
glm_fit$results['RMSE']
```

```
##       RMSE
## 1 1.800426
```

With a RMSE of 1.800426, this model shows clear improvement, but leaves some room to do better.

## 6.4 Random forest

Finally, we are going to use the Random forest model.

```
#Random forest model
rf_fit <- train(Global_Sales~Critic_Score+Critic_Count+User_Score+User_Count+System,
                data=train_set, tuneLength=2, method="rf")
trn_rf <- predict(rf_fit,train_set)
RMSE(trn_rf, test_set$Global_Sales)
```

```
## [1] 1.797835
```

```
rf_fit$results[row.names(rf_fit$bestTune),'RMSE']
```

```
## [1] 1.582424
```

With a new rmse of 1.582424, we definitely improve our prediction using the random forest.

# 7 Results

As the result of our different model, we can see a clear improvement from our first model with a rmse of 1.900543 to the final one with a rmse of 1.582424. Adding predictors helped us reduce the rmse, but the main diffrence is the use of the random forest.

# 8 Conclusion

Our models help us predict video game sales and which variables to look for. It may be helpful when developing the next big it, but major companies already have figured it out in a lot of aspects. Major triple A are supported by a very consequent marketing budget, ensuring maximum visibility, and are on multiplatforms. Although Nintendo shows that you can be strong with only one plateform. It should be possible to improve the model further by adding other effects that may influence global sales. It will also be possible to add tuning parameters for each of the effects, in order to better fine tune the model. We may also use different machine learning models, as matrix factorization to improve the rmse, although it may be heavy on computers that don't have the necessary hardware.