Video Games Sales Prediction

Nirmal Sai Swaroop Janapaneedi

30/04/2020

Overview

This report details the analysis of Games Sales data and attempts to construct an algorithm that predicts Global sales on different variables. This document gives the reasoning behind every step taken and a conclusion will be given based on the findings.

First we create the data with the sales data.

load library

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ---------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.4
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
## Warning: package 'forcats' was built under R version 3.6.3
```

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

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

```
## Loading required package: caret
```

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

```
## Warning: package 'lattice' was built under R version 3.6.3
```

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

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

```
library(tidyverse)
library(caret)
```

upload data and create data set

```
Games <- read_csv("https://raw.githubusercontent.com/viguerrero/Games-Sales-Analysis/master/Video_Games
```

```
##  Parsed with column specification:
##  cols(
##    Name = col_character(),
##    Platform = col_character(),
##    Year_of_Release = col_character(),
##    Genre = col_character(),
##    Publisher = col_character(),
##    NA_Sales = col_double(),
##    EU_Sales = col_double(),
##    JP_Sales = col_double(),
##    Other_Sales = col_double(),
##    Global_Sales = col_double(),
##    Critic_Score = col_double(),
##    Critic_Count = col_double(),
##    User_Score = col_character(),
##    User_Count = col_double(),
##    Developer = col_character(),
##    Rating = col_character()
##  )
```

convert data.frame to tibble

```
Games <- as_tibble(Games)
```

Confirm Class

```
class(Games)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Remove from data set incomplete data and N/As

```
Games <- Games[complete.cases(Games),]
```

Confirm no N/As in data

```
any(is.na(Games))
```

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

Analysis of data set created

```
nrow(Games)
```

```
## [1] 6947
```

```
ncol(Games)
```

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

```
names(Games)
```

```
##  [1] "Name"            "Platform"        "Year_of_Release" "Genre"
##  [5] "Publisher"       "NA_Sales"        "EU_Sales"        "JP_Sales"
##  [9] "Other_Sales"     "Global_Sales"    "Critic_Score"    "Critic_Count"
## [13] "User_Score"      "User_Count"      "Developer"       "Rating"
```

The data set consists of 6927 rows and 16 variables, (columns) as can be seen.

Next we evaluate the relevant elements of the data first finding unique values in each column of interest.

Analysis

See Unique Values in Categorical Columns of interest

```
n_distinct(Games$Name)
```

```
## [1] 4420
```

```
n_distinct(Games$Platform)
```

```
## [1] 17
```

```
n_distinct(Games$Publisher)
```

```
## [1] 263
```

```
n_distinct(Games$Genre)
```

```
## [1] 12
```

```
n_distinct(Games$Developer)
```

```
## [1] 1297
```

It can be noticed that there are fewer unique names than rows in the data as such we try to determine why this is, in an effort to better understand the data set.

Create Vector of duplicate names, in alpabetical order

```
duplicates.games <- sort(Games$Name[duplicated(Games$Name)])
```

View reason why there are duplicates of a specific game

```
duplicates.view1 <- Games %>% filter(Name==duplicates.games[1])

view(duplicates.view1)
```

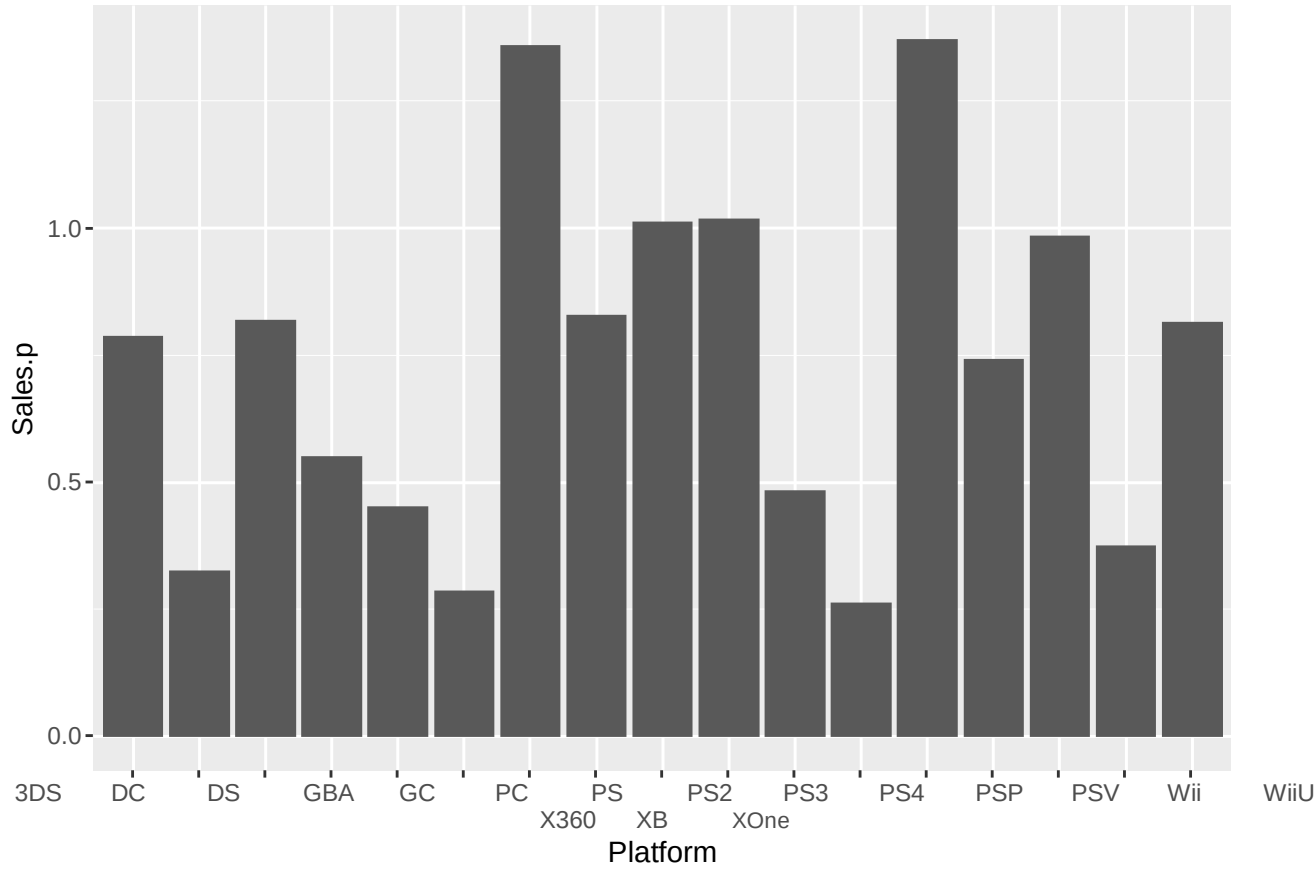
With this we find out that games with the same names are divided by platform.

Next we try to determine the variance in the different variables and how they affect the mean sales.

Starting with Platform.

Calculate mean sales by Platform and plot the findings

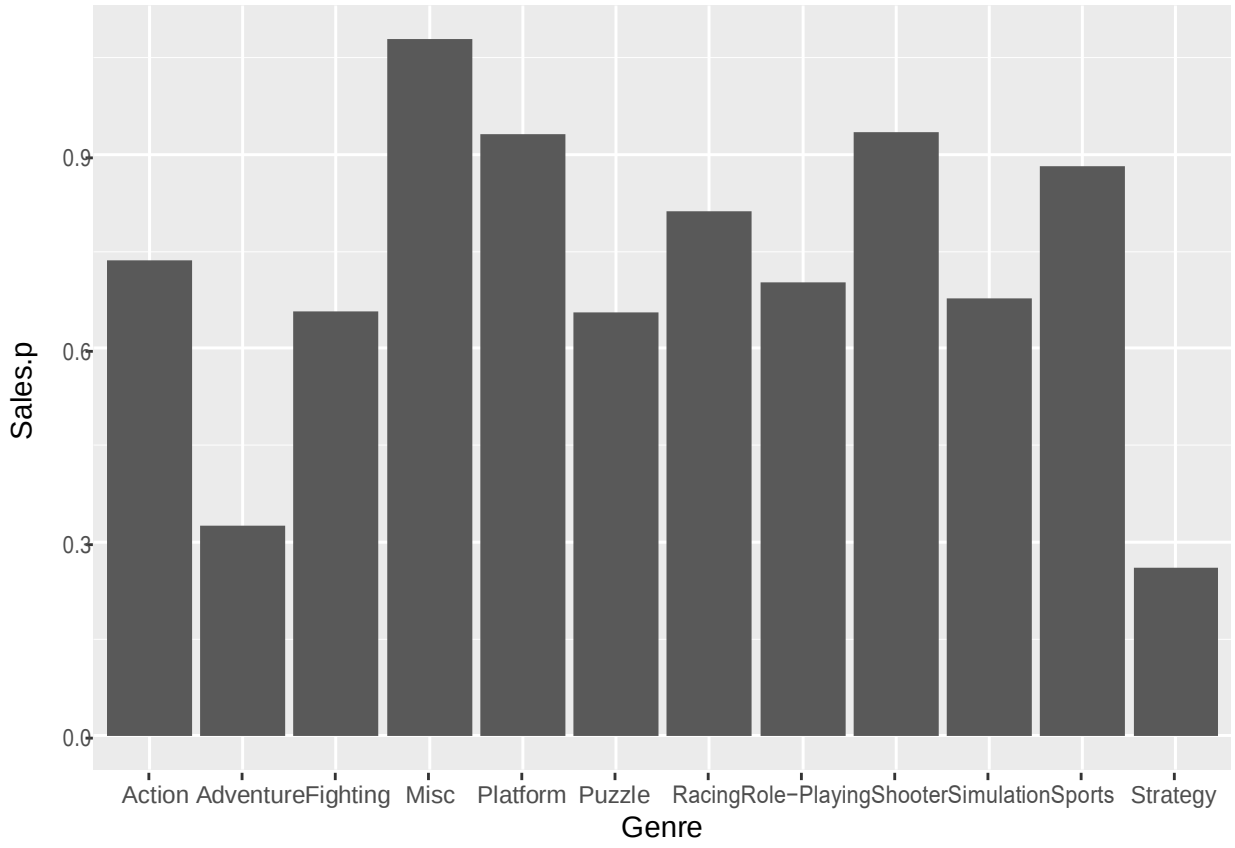
```
Platform_sales <- Games %>%
  group_by(Platform) %>%
  summarize(Sales.p = mean(Global_Sales))

ggplot(Platform_sales, aes(x=Platform,y = Sales.p)) +geom_bar(stat = "identity")
```



We can see from the plot that there is significant difference between platforms. Next we see mean sales by Genre to see if we find differences in Genre type.

Calculate mean sales by Genre and plot the findings

```
Genre_sales <- Games %>%
        group_by(Genre) %>%
        summarize(Sales.p = mean(Global_Sales))
ggplot(Genre_sales, aes(x=Genre,y = Sales.p)) +geom_bar(stat = "identity")
```

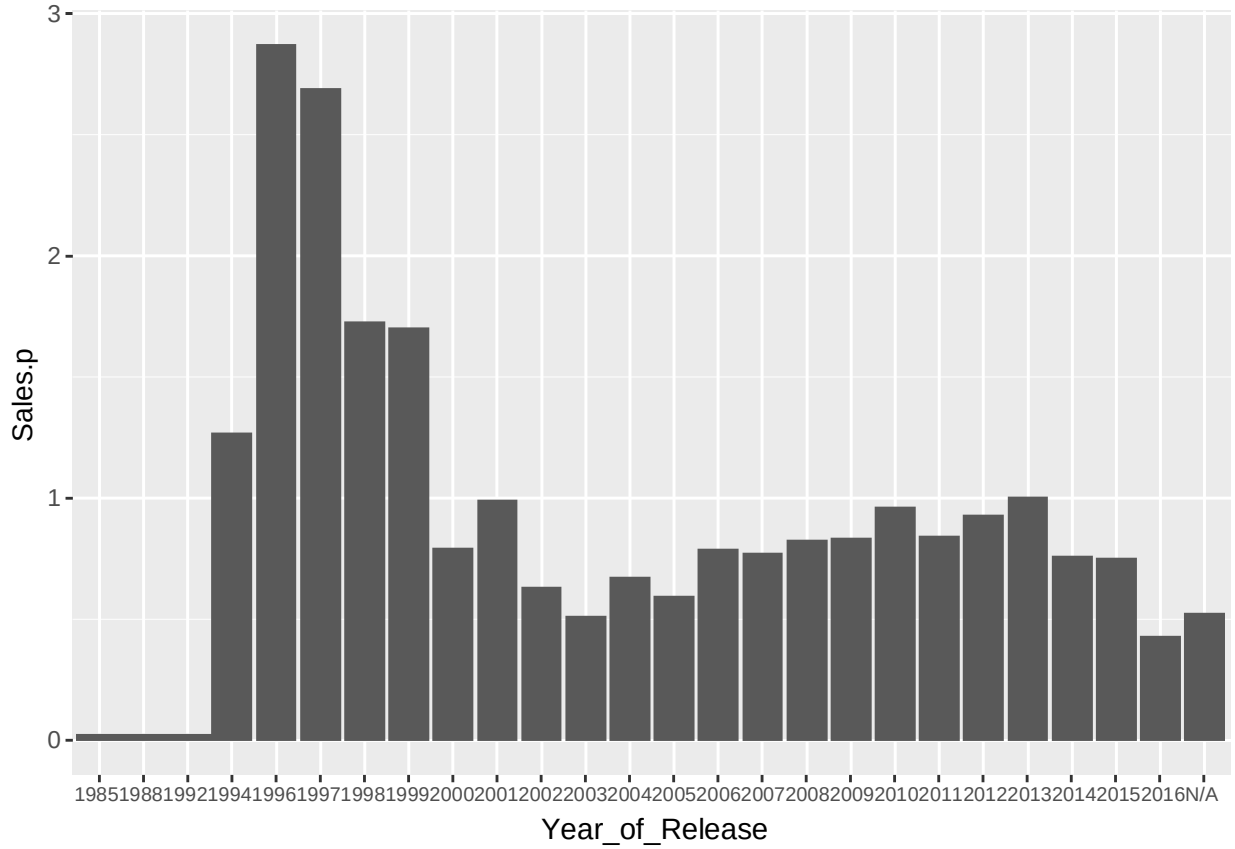


As we can see there are significant diffences in sales by genre, Misc being the highest and strategy being the lowest.

Now we will evaluate if year of release has a significant impact in the mean sales.

Calculate mean sales by year and plot the findings

```
Year_release_sales <- Games %>%
   group_by(Year_of_Release) %>%
   summarize(Sales.p = mean(Global_Sales))
ggplot(Year_release_sales, aes(x=Year_of_Release,y = Sales.p)) +geom_bar(stat = "identity")
```



As you can see “year of release” has indeed a great impact in average sales. Now we will try to deterime why this is. First we will analyse games released per year.

See games released per year

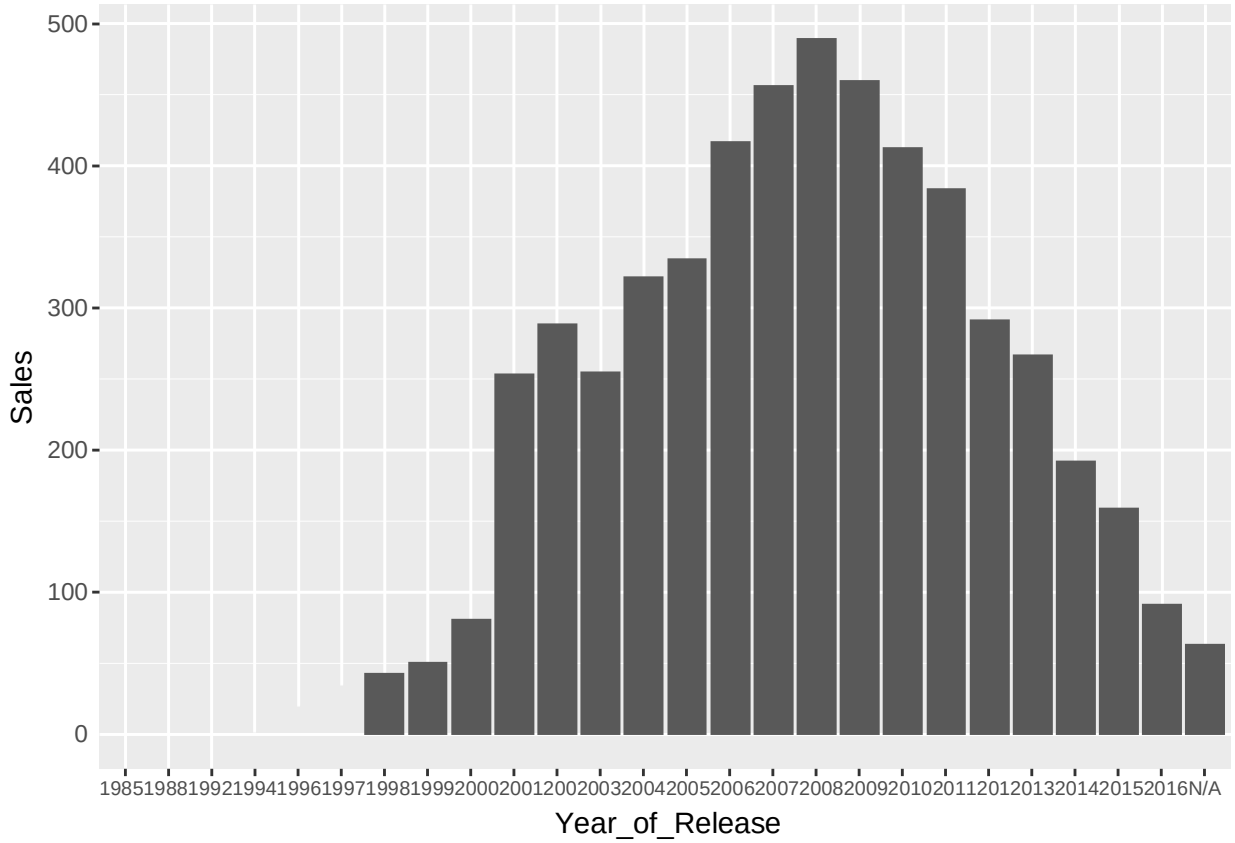
```
Games_per_year <- Games %>%
        group_by(Year_of_Release) %>%
        count(Year_of_Release)
```

Understand why N/As for year appear in data

```
Games_per_year.Na <- Games %>%
                              group_by(Year_of_Release) %>%
                              filter(Year_of_Release== "N/A")
```

Determine Total Sales per Year of release and plot findings*

```
Sales_per_Year <- Games %>%
  group_by(Year_of_Release) %>%
  summarize(Sales = sum(Global_Sales))
ggplot(Sales_per_Year, aes(x=Year_of_Release,y = Sales)) +geom_bar(stat = "identity")
```



As can be seen a lot fewer games where made before the 2000s as such Sales per game is considerably higher. However, we can also see that the market sales, total sales. increased considerably in 2001 and peaked in 2009. Although we must assume that this is not the entire data thus no conclusion can be drawn from it.

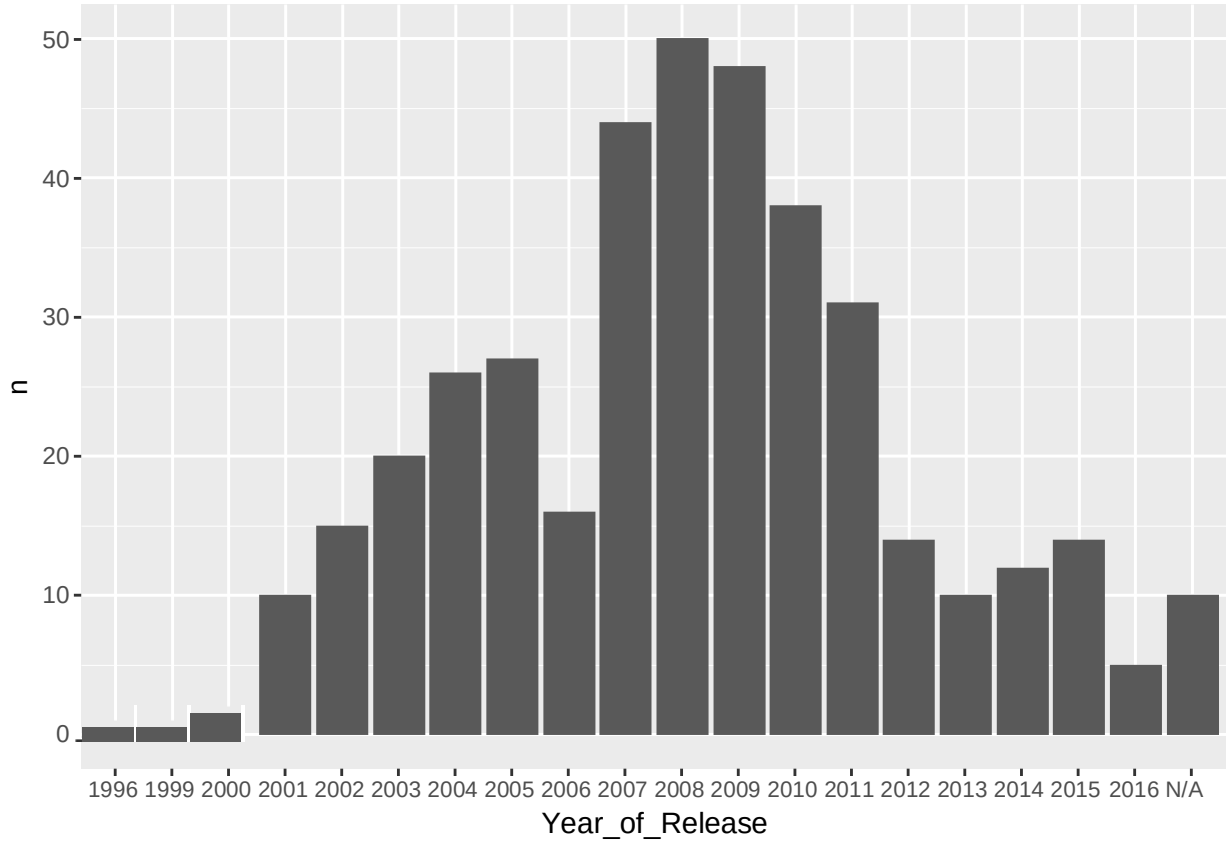
Now we will determine whether games released of a certain genre has influenced the total sales of the year.

See Genres per Year

```
Genres_per_year <- Games %>%
                group_by(Year_of_Release,Genre) %>%
                count(Year_of_Release,Genre)
```

See Misc Games per Year

```
Misc_per_year <- Genres_per_year %>% filter(Genre == "Misc")

ggplot(Misc_per_year, aes(x=Year_of_Release,y = n)) +geom_bar(stat = "identity")
```

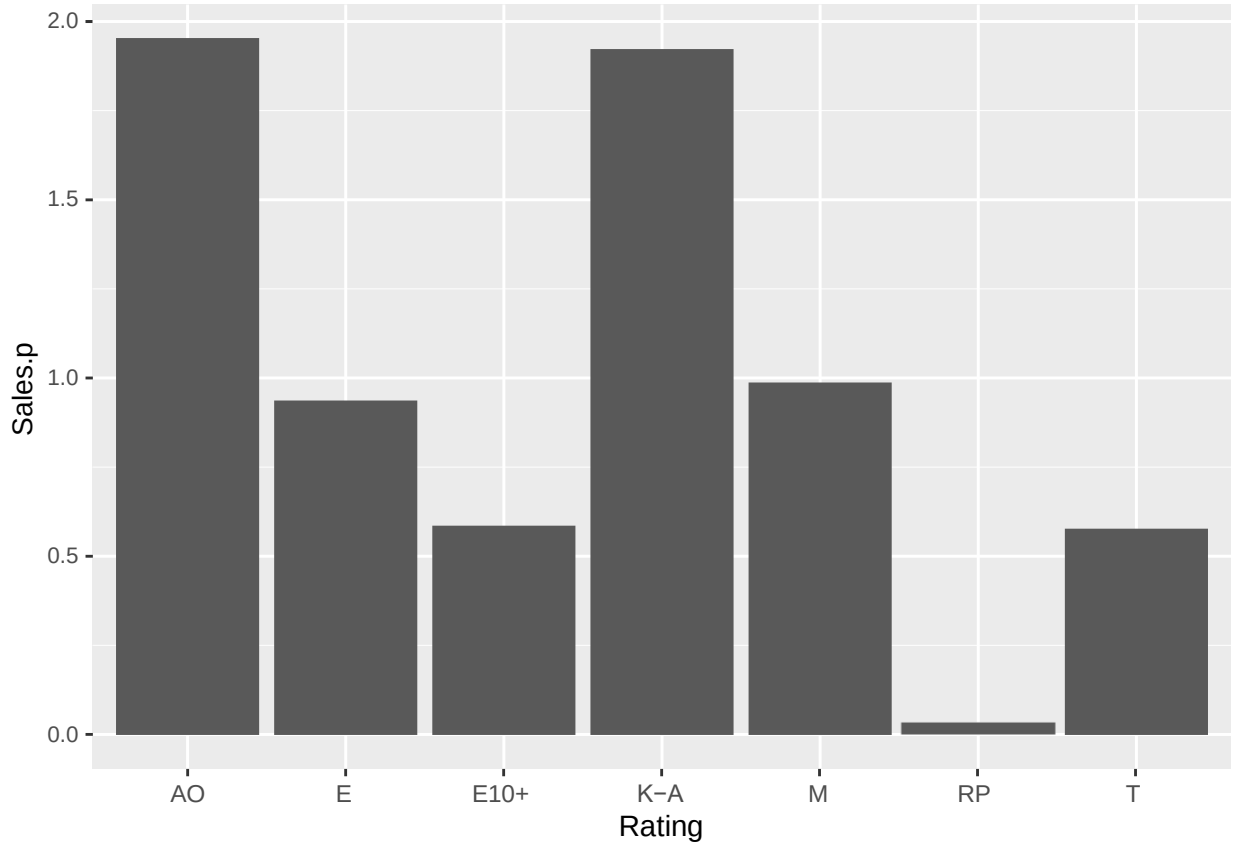


We focus on Misc and can see that there is a positive correlation between Misc games released in a year and total sales that year.

Now we evaluate variance based on Rating, Developer and Publisher

Calculate Mean per Game Rating

```
Rating_sales <- Games %>%
   group_by(Rating) %>%
   summarize(Sales.p = mean(Global_Sales))
ggplot(Rating_sales, aes(x=Rating,y = Sales.p)) + geom_bar(stat = "identity")
```



Calculate Mean per publisher and see top 10 publishers

```
Publisher_sales <- Games %>%
  group_by(Publisher) %>%
  summarize(Sales.p = mean(Global_Sales))

Publisher_sales %>%
                top_n(n=10) %>%
                arrange(desc(Sales.p))
```

```
## Selecting by Sales.p
```

```
## # A tibble: 10 x 2
##       Publisher                          Sales.p
##       <chr>                                <dbl>
##    1 Nintendo                               2.91
##    2 GT Interactive                         2.83
##    3 SquareSoft                             2.76
##    4 RedOctane                              2.17
##    5 Hello Games                            1.7
##    6 Valve                                  1.7
##    7 Microsoft Game Studios                 1.54
##    8 Sony Computer Entertainment Europe     1.53
##    9 Bethesda Softworks                     1.49
## 10  Hasbro Interactive                      1.43
```

Calculate mean per developer and see top 10 Developers

```
Developer_sales <- Games %>%
  group_by(Developer) %>%
  summarize(Sales.p = mean(Global_Sales))

Developer_sales %>%
  top_n(n=10) %>%
  arrange(desc(Sales.p))
```

```
## Selecting by Sales.p
```

```
## # A tibble: 10 x 2
##    Developer                                                      Sales.p
##    <chr>                                                            <dbl>
##  1 Good Science Studio                                               21.8
##  2 Retro Studios, Entertainment Analysis & Development Division      12.7
##  3 Infinity Ward, Sledgehammer Games                                 9.92
##  4 Polyphony Digital                                                 9.31
##  5 Rockstar North                                                    8.53
##  6 DMA Design                                                        8.26
##  7 Nintendo                                                          7.79
##  8 Bungie                                                            5.90
##  9 Bungie Software, Bungie                                           5.73
## 10 Bungie Software                                                   5.20
```

Determine what kind of games developers create

```
Developers.genres <- Games %>%
                              count(Developer, Genre) %>%
                              group_by(Developer)
```

Verifying top developer

```
Good.sc.studio.genres <- Developers.genres %>%
                                      filter(Developer=="Good Science Studio")

view(Good.sc.studio.genres)
```

Verifying a well known creator

```
Developers.genres %>%
        filter(Developer=="Nintendo")
```

```
## # A tibble: 10 x 3
## # Groups:   Developer [1]
##    Developer  Genre          n
##    <chr>      <chr>      <int>
##  1 Nintendo   Action        12
##  2 Nintendo   Adventure      5
##  3 Nintendo   Misc           8
##  4 Nintendo   Platform      15
##  5 Nintendo   Puzzle         6
```

```
##  6 Nintendo   Racing       4
##  7 Nintendo   Shooter      4
##  8 Nintendo   Simulation   5
##  9 Nintendo   Sports       6
## 10 Nintendo   Strategy     3
```

We can see that Rating has a significant influence in the sales being AO and K-A the highest mean sales.

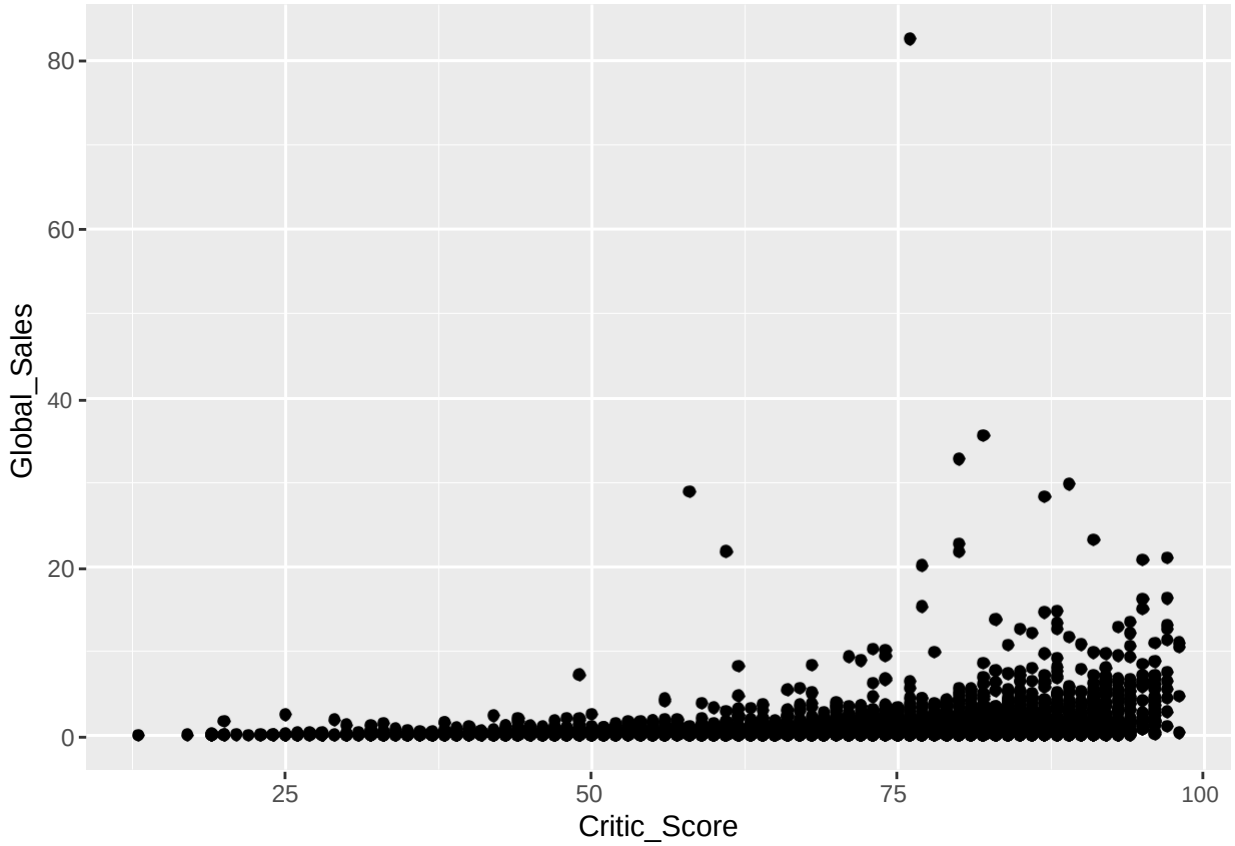
We can also see that there is a great variance based on the game's developer. However, we can see that the top developer has only released a Game that being a Misc Game.

We then verify a well known developer to check that the code yields correct results and we confirm indeed Nintendo has developed games of various Genres.

Next we will try to find correlation between Global sales and the continous variables of Critic Score, Critic Count and User score and user count.

Plot Crit_Score and Global_Sales

```
Games %>%
        ggplot(aes(Critic_Score, Global_Sales)) +
        geom_point(alpha = 0.5)
```



From the plot it is not clear how correlated it is as such a more direct approach is favoured below.

Find Correlation between Crit_Score and Global_Sales

```
cor(Games$Global_Sales,Games$Critic_Score)
```

[1] 0.2373968

Find correlation between Crit_count and Global Sales

```
cor(Games$Global_Sales,Games$Critic_Count)
```

[1] 0.2889865

Convert User_score to numeric and find correlation with Global_Sales

```
Games <- Games %>% mutate(User_Score= as.numeric(User_Score))
class(Games$User_Score)
```

[1] "numeric"

```
cor(Games$Global_Sales,Games$User_Score)
```

[1] 0.08856755

Find correlation between User_count and Global Sales

```
cor(Games$Global_Sales,Games$User_Count)
```

[1] 0.2630594

As we can see we confirm that there is no correlation between the Crit and user, score and count and the global sales. As such we create a table of mean Scores per genre to confirm scores per genre.

Find User and Crit Score per Genre

```
Score_Per_Genre<- Games %>%
  group_by(Genre) %>%
  summarize(Critics= mean(Critic_Score), Users= mean(User_Score))

View(Score_Per_Genre)
```

As we can see Misc Genre is the only genre with mean ratings below 70 for crtic score and 7 for user score.

As such it can be assumed that different genres have a user and critic mean score bias. Therefore, we will study if there is a correlation between Critic and User score and count between games of the same Genre.

Get all Misc Genre data

```
All_Misc <- Games %>%
                        filter(Genre=="Misc")
```

Correlation of critic_score and User Score in Misc Games

```
cor(All_Misc$Global_Sales,All_Misc$Critic_Score)
```

[1] 0.08597568

```
cor(All_Misc$Global_Sales,All_Misc$User_Score)
```

[1] 0.0734291

```
cor(All_Misc$Global_Sales,All_Misc$Critic_Count)
```

[1] 0.230206

```
cor(All_Misc$Global_Sales,All_Misc$User_Count)
```

[1] 0.2874169

Get all shooter Genre data and Correlation of critic_score and User Score

```
All_shooter <- Games %>%
  filter(Genre=="Shooter")

cor(All_shooter$Global_Sales,All_shooter$Critic_Score)
```

[1] 0.3470809

```
cor(All_shooter$Global_Sales,All_shooter$User_Score)
```

[1] -0.03021377

```
cor(All_shooter$Global_Sales,All_shooter$Critic_Count)
```

[1] 0.3755382

```
cor(All_shooter$Global_Sales,All_shooter$User_Count)
```

[1] 0.4514587

Get Role-Playing Genre data and Correlation of critic_score and User Score

```
All_Role_Playing <- Games %>%
  filter(Genre=="Role-Playing")

cor(All_Role_Playing$Global_Sales,All_Role_Playing$Critic_Score)
```

[1] 0.4056516

```
cor(All_Role_Playing$Global_Sales,All_Role_Playing$User_Score)
```

```
## [1] 0.1277234
```

```
cor(All_Role_Playing$Global_Sales,All_Role_Playing$Critic_Count)
```

```
## [1] 0.4063109
```

```
cor(All_Role_Playing$Global_Sales,All_Role_Playing$User_Count)
```

```
## [1] 0.3749556
```

Get Strategy Genre data and Correlation of critic_score and User Score

```
All_Strategy <- Games %>%
  filter(Genre=="Strategy")

cor(All_Strategy$Global_Sales,All_Strategy$Critic_Score)
```

```
## [1] 0.2726191
```

```
cor(All_Strategy$Global_Sales,All_Strategy$User_Score)
```

```
## [1] 0.07138455
```

```
cor(All_Strategy$Global_Sales,All_Strategy$Critic_Count)
```

```
## [1] 0.410635
```

```
cor(All_Strategy$Global_Sales,All_Strategy$User_Count)
```

```
## [1] 0.3966192
```

As we can see when filtered by Genre the correlation between Global sales and Critic and User score and count is higher but not enough to be significant.

Method

As can be seen the continous variables, Critic and User score and count, are not reliable predictors of Global Sales. Moreover, the only reliable variance we have are in Categorical variables: Platform, Genre and Rating.
For developers the information is unreliable as some developers only has 1 game to their name and we want to create a an algorithm that accounts for new developers, the same thing with Publisher.
For this algorithm we will not contemplate linear regression because there isn't significant correlation between the continous variables. Morever, KNN nearest neighbours and logical trees are discarded because the variable we are trying to predict, Global Sales, is a continous variable.

As such we are going to use a weighted average approach to try to predict the Game Global Sales.

First we divide The data into Training and Test Sets, we set the seed to avoid differences due to randomness. WE create the training set out of 90% of the data due to the high variability of the data set thus trying to have as much data to train from as possible.

Divide data into Train and Test set after setting the seed

```
set.seed(1)
y <- Games$Global_Sales
test_index <- createDataPartition(y, times = 1, p = 0.1, list = FALSE)
test_set <- Games %>% slice(test_index)
train_set <- Games %>% slice(-test_index)
```

Next we find the mean sales of all movies in the train set.

Find train Set mean

```
mu <- mean(train_set$Global_Sales)
```

We create lambda values for later randomly, in this case based on previous successes and we create the function that evaluates the effectiveness of the algorithm.

Create Lambda and RMSE check

```
lambdas <- seq(0, 10, 0.25)

        RMSE <- function(true_ratings, predicted_ratings){
           sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

This algorithm gives more weight to variables with more lines. With the line +l. Below we determine the best Lamda Value.

Determine best lambda value

```
rmses <- sapply(lambdas, function(l){
  b_i <- train_set %>%
    group_by(Platform) %>%
    summarize(b_i = sum(Global_Sales - mu)/(n()+l))
  b_u <- train_set %>%
    left_join(b_i, by="Platform") %>%
    group_by(Genre) %>%
    summarize(b_u = sum(Global_Sales - b_i - mu)/(n()+l))
  predicted_Sales <-
    test_set %>%
    left_join(b_i, by = "Platform") %>%
    left_join(b_u, by = "Genre") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred
  return(RMSE(test_set$Global_Sales, predicted_Sales))
})
```

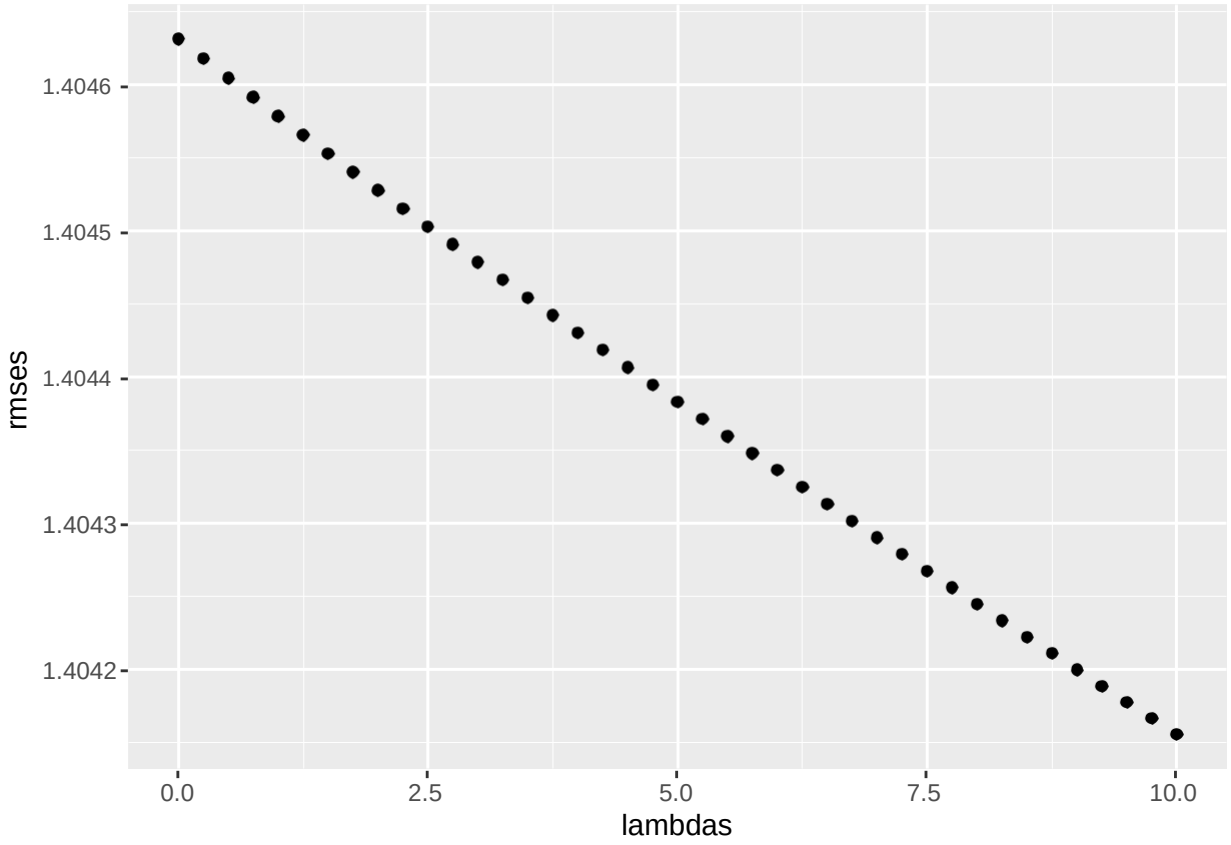
Determine lowest lambda value

```
lambda <- lambdas[which.min(rmses)]
lambda
```

[1] 10

visualize lambda values

```
qplot(lambdas, rmses)
```



We can see that the best Value is 10 but it is not clear if the RMSE would decrease further if it was larger as such a new lambda value is chosen.

New lambda Value

```
lambdas <- seq(10, 100, 10)
```

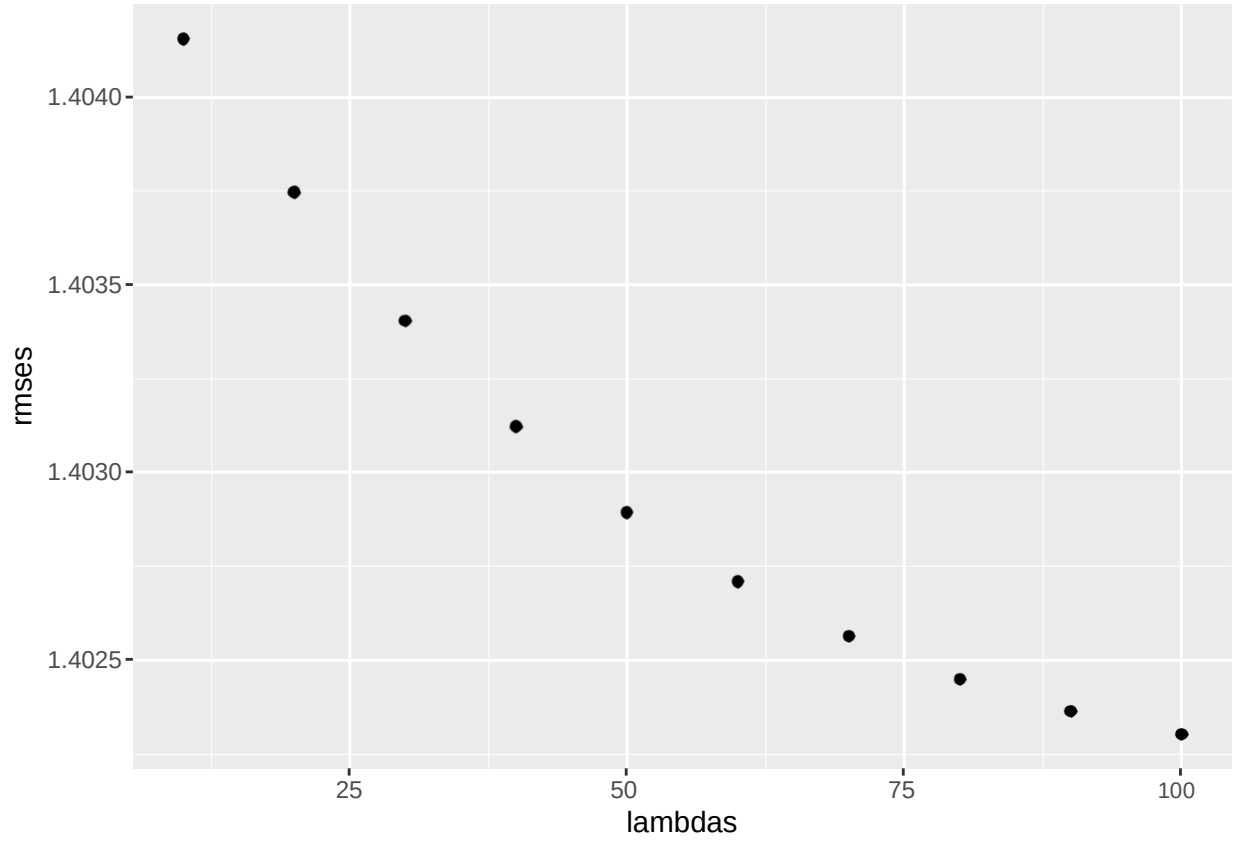
Rerun of lambda Analysis

```
rmses <- sapply(lambdas, function(l){
  b_i <- train_set %>%
    group_by(Platform) %>%
    summarize(b_i = sum(Global_Sales - mu)/(n()+l))
  b_u <- train_set %>%
    left_join(b_i, by="Platform") %>%
    group_by(Genre) %>%
```

```
    summarize(b_u = sum(Global_Sales - b_i - mu)/(n()+l))
  predicted_Sales <-
    test_set %>%
    left_join(b_i, by = "Platform") %>%
    left_join(b_u, by = "Genre") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred
  return(RMSE(test_set$Global_Sales, predicted_Sales))
})
```

New check of lambda plot

```
qplot(lambdas, rmses)
```




```
lambda <- lambdas[which.min(rmses)]
lambda
```

[1] 100

With the new values we perceive the same effect as such we choose new lambda values to re-evaluate the algorithm.

New lambda Value

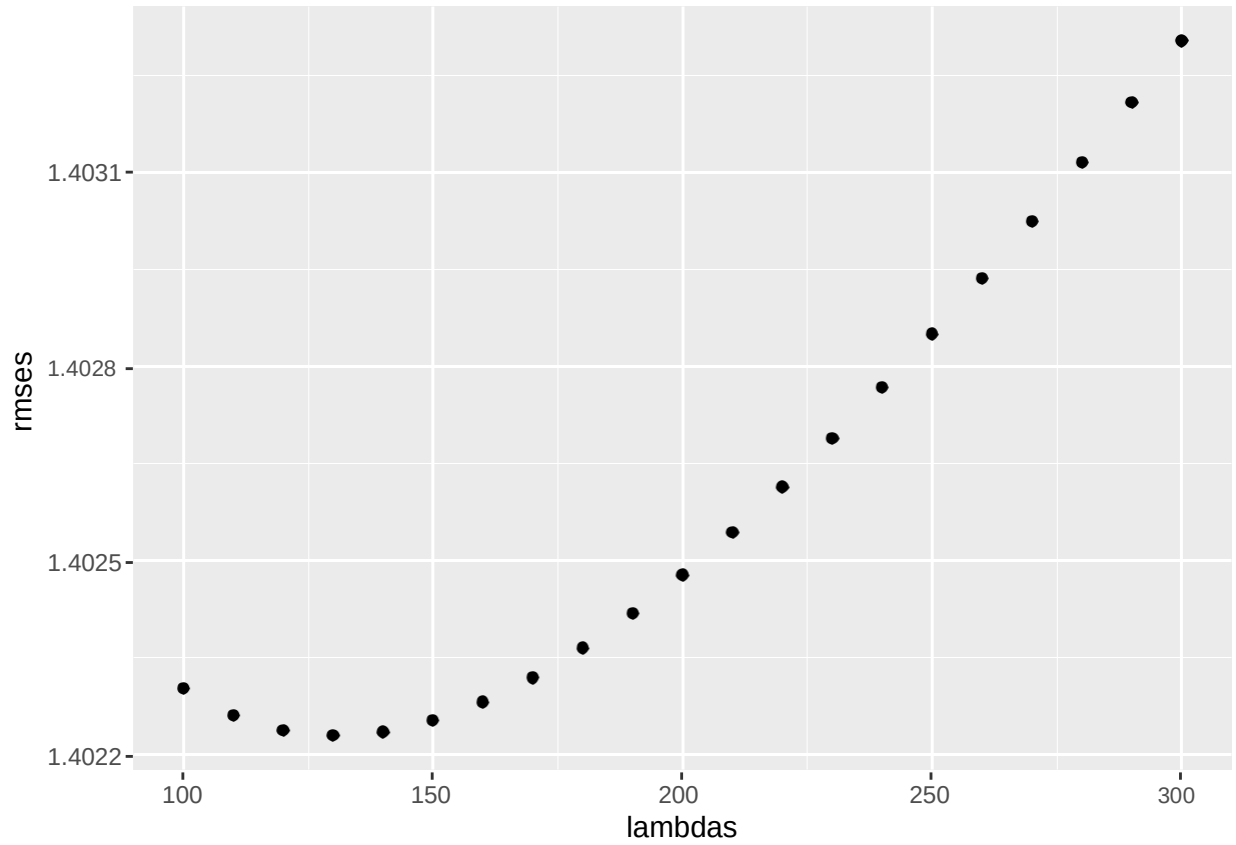
```
lambdas <- seq(100, 300, 10)
```

#Rerun of lambda Analysis a 3rd time

```
rmses <- sapply(lambdas, function(l){
   b_i <- train_set %>%
      group_by(Platform) %>%
      summarize(b_i = sum(Global_Sales - mu)/(n()+l))
   b_u <- train_set %>%
      left_join(b_i, by="Platform") %>%
      group_by(Genre) %>%
      summarize(b_u = sum(Global_Sales - b_i - mu)/(n()+l))
   predicted_Sales <-
      test_set %>%
      left_join(b_i, by = "Platform") %>%
      left_join(b_u, by = "Genre") %>%
      mutate(pred = mu + b_i + b_u) %>%
      .$pred
   return(RMSE(test_set$Global_Sales, predicted_Sales))
})
```

New check of lambda plot

```
qplot(lambdas, rmses)
```



```
lambda <- lambdas[which.min(rmses)]
lambda
```

```
## [1] 130
```

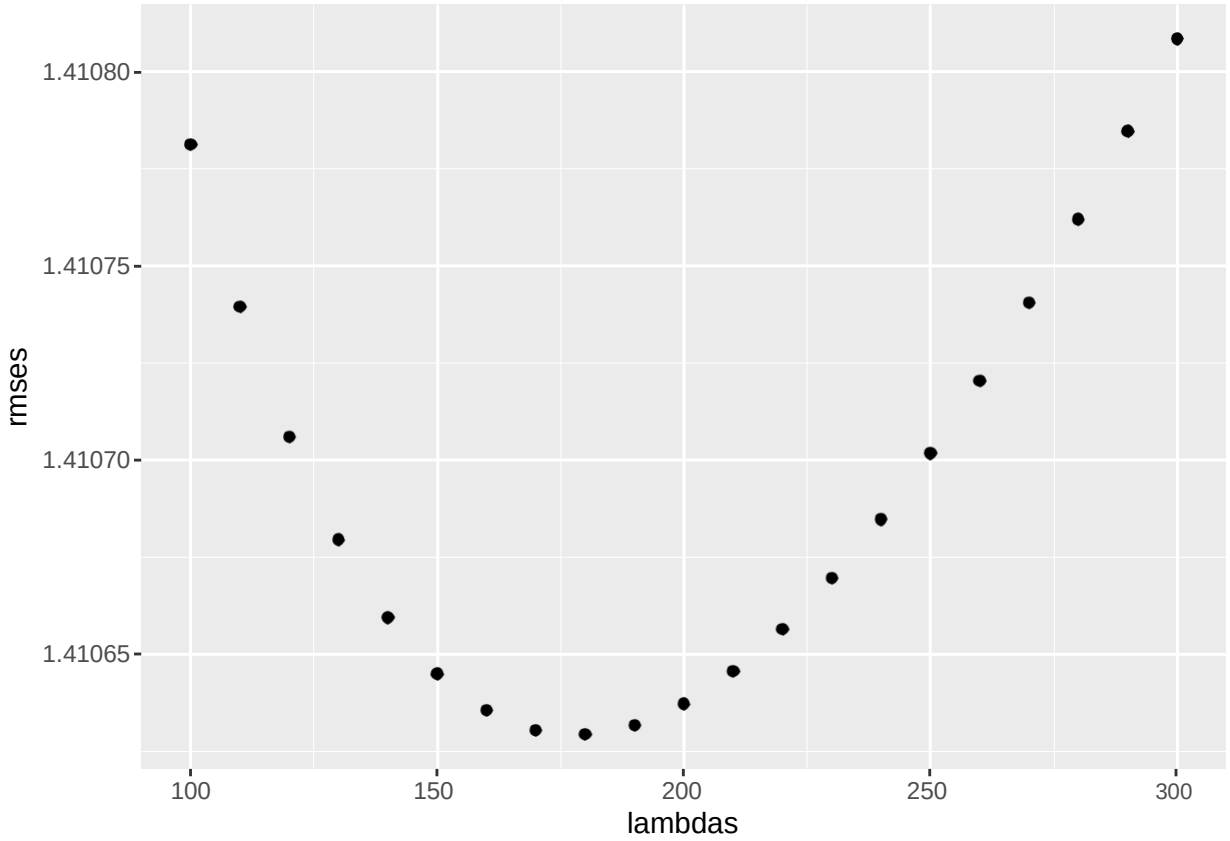
Finally we find an appropiate range of lambda value.Now we test the algorithm with the Genre and Rating combination.

Check with Rating and Genre combination keep lambda values

```
rmses <- sapply(lambdas, function(l){
  b_i <- train_set %>%
    group_by(Rating) %>%
    summarize(b_i = sum(Global_Sales - mu)/(n()+l))
  b_u <- train_set %>%
    left_join(b_i, by="Rating") %>%
    group_by(Genre) %>%
    summarize(b_u = sum(Global_Sales - b_i - mu)/(n()+l))
  predicted_Sales <-
    test_set %>%
    left_join(b_i, by = "Rating") %>%
    left_join(b_u, by = "Genre") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred
  return(RMSE(test_set$Global_Sales, predicted_Sales))
})
```

New check of lambda plot

```
qplot(lambdas, rmses)
```



As we can see the lowest RMSE of the Rating Genre Combination is not lower than the Platform Genre Combination. As such the Platform, Genre combination is preferred. Below the final algorithm.

Results

Final predictive algorithm with lambda still set at 130

```
b_i <- train_set %>%
    group_by(Platform) %>%
    summarize(b_i = sum(Global_Sales - mu)/(n()+lambda))
b_u <- train_set %>%
    left_join(b_i, by="Platform") %>%
    group_by(Genre) %>%
    summarize(b_u = sum(Global_Sales - b_i - mu)/(n()+lambda))
predicted_Sales <-
    test_set %>%
    left_join(b_i, by = "Platform") %>%
    left_join(b_u, by = "Genre") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred
RMSE(test_set$Global_Sales, predicted_Sales)
```

[1] 1.402229

Conclusion

In general we have found that there is little to no correlation between Critic and User Score and Count and Global Sales, as such, Developers should not worry too much about them. Moreover, Misc Games seem to be preferred as they consistently have more sales than other mainstream genres such as shooting.

The Game industry seems to have a market share which is divided by the games that are developed. The market seems to have peaked in 2009 with the largest volume of sales and has since declined.This decline is proportional to the games released.

This suggests that the Game Industry must create a wide variety of games that appeal to different demographics, focusing on the best selling genres such as Misc and Shooting and releasing less of others like strategy.

Additionally, the platform in which the game is released greatly influences the sales of the Game.

The Final algorithm's RMSE is almost twice the size of the mean Global Sales at 0.77 million. However, considering the variance in sales, up to 82 million, it is acceptable.

Variance in sales

```
max(Games$Global_Sales)-min(Games$Global_Sales)
```

```
## [1] 82.52
```

However, a limitation of this algorithm is that it can only accept two variables and further improvement could be achieved if Rating was included in the model.Furthermore, the algorithm would be more reliable if we had more complete cases, with more data a more concise conclusion could be established.