Video Games Sales Predictor Model

Low Yao Dong

6/22/2021

#1 Introduction

This report documents the creation of a video games sales predictor model based on historical sales. The data set was retrieved from Kaggle (https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings). First, descriptive analytics was used to summarize and visualize historical data to yield useful information for the model building. After that, a multivariate linear regression and random forest regression were compared to determine which algorithm was better in terms of root mean squared error (RMSE) value.

#2 Methodology

##2.1 Data Pre-Processing

To begin, the video games sales data set was retrieved by means of a relative path. The data was found to be generally clean and only little pre-processing was required. This included removing unwanted sales columns and converting the user score data to integer. Then, the data set was split 80/20 into training and test sets respectively. This ratio was selected to ensure sufficient data in the test set and to avoid over-training.

```
####################################################
######## Video games sales predictor model ########
####################################################

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

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

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

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

```
## Loading required package: caret

## Loading required package: lattice
```

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

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

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

```
## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

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

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

```
## Loading required package: RCurl

##
## Attaching package: 'RCurl'

## The following object is masked from 'package:tidyr':
##
##     complete
```

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

```
## Loading required package: randomForest

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
# Load required packages
library(tidyverse)
library(caret)
library(data.table)
library(dplyr)
library(knitr)
library(stringr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(tinytex)
library(RCurl)
library(randomForest)

# Provide relative path to dataset
game_sales <- read.csv("https://github.com/lowy0047/Video-game-sales-capstone/blob/main/Video_Games_Sal
game_sales <- na.omit(game_sales)

# Keep only global_sales column
game_sales <- subset(game_sales, select = -c(NA_Sales, EU_Sales, JP_Sales, Other_Sales))

# Remove tbd from user scores and convert column to integer class
game_sales$User_Score <- gsub('tbd', '', game_sales$User_Score)
game_sales$User_Score <- as.numeric(game_sales$User_Score)
summary(game_sales)
```

```
##      Name             Platform         Year_of_Release       Genre
##  Length:7017        Length:7017        Length:7017        Length:7017
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   Publisher          Global_Sales      Critic_Score    Critic_Count
##  Length:7017        Min.   : 0.0100   Min.   :13.00   Min.   :  3.00
##  Class :character   1st Qu.: 0.1100   1st Qu.:62.00   1st Qu.: 14.00
##  Mode  :character   Median : 0.2900   Median :72.00   Median : 24.00
##                     Mean   : 0.7671   Mean   :70.25   Mean   : 28.78
##                     3rd Qu.: 0.7500   3rd Qu.:80.00   3rd Qu.: 39.00
##                     Max.   :82.5300   Max.   :98.00   Max.   :113.00
##    User_Score      User_Count       Developer            Rating
```

```
##  Min.   :0.500   Min.   :    4.0   Length:7017        Length:7017
##  1st Qu.:6.500   1st Qu.:   11.0   Class :character   Class :character
##  Median :7.500   Median :   27.0   Mode  :character   Mode  :character
##  Mean   :7.182   Mean   :  173.4
##  3rd Qu.:8.200   3rd Qu.:   89.0
##  Max.   :9.600   Max.   :10665.0
```

```
# Set seed to 1 then split data into 80/20 training/test set
set.seed(1, sample.kind = "Rounding")
```

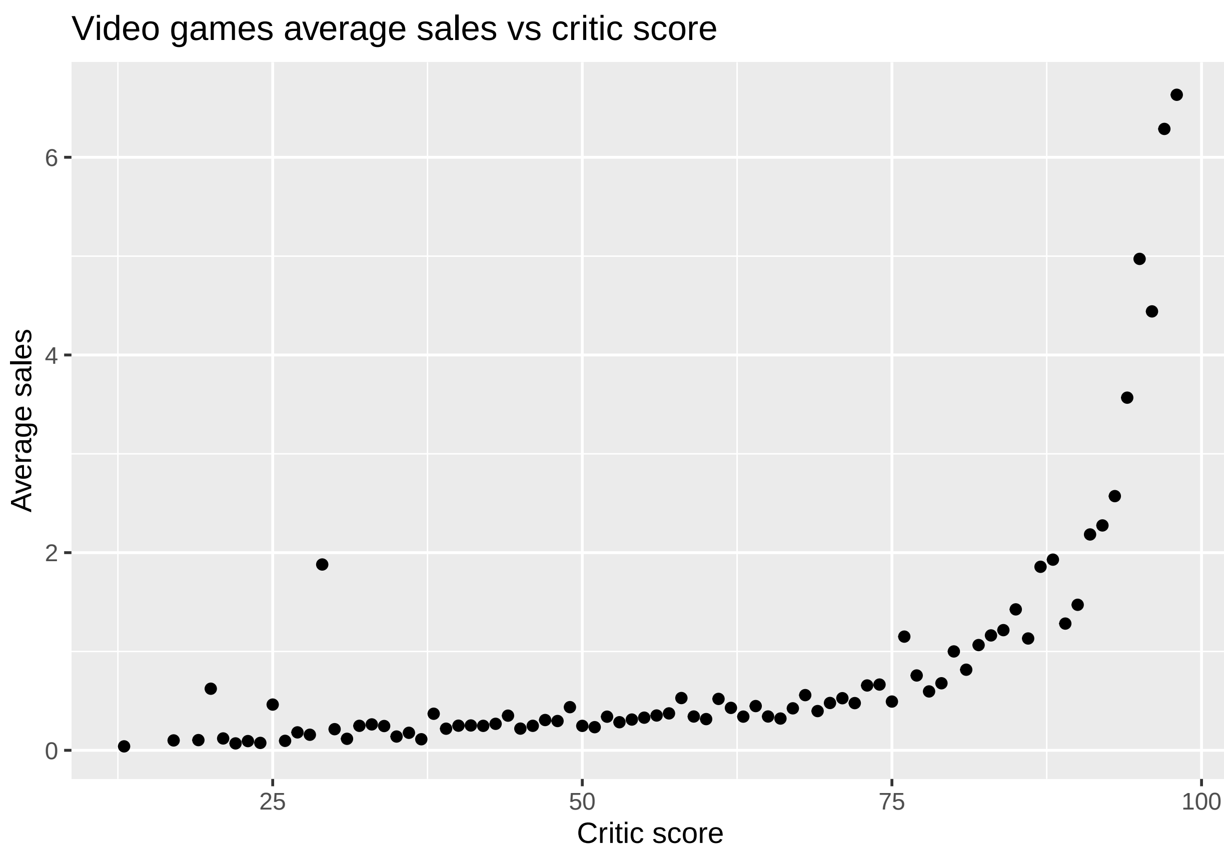
```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <- createDataPartition(y = game_sales$Global_Sales, times = 1, p = 0.2, list = FALSE)
train_set <- game_sales[-test_index,]
test_set <- game_sales[test_index,]
```

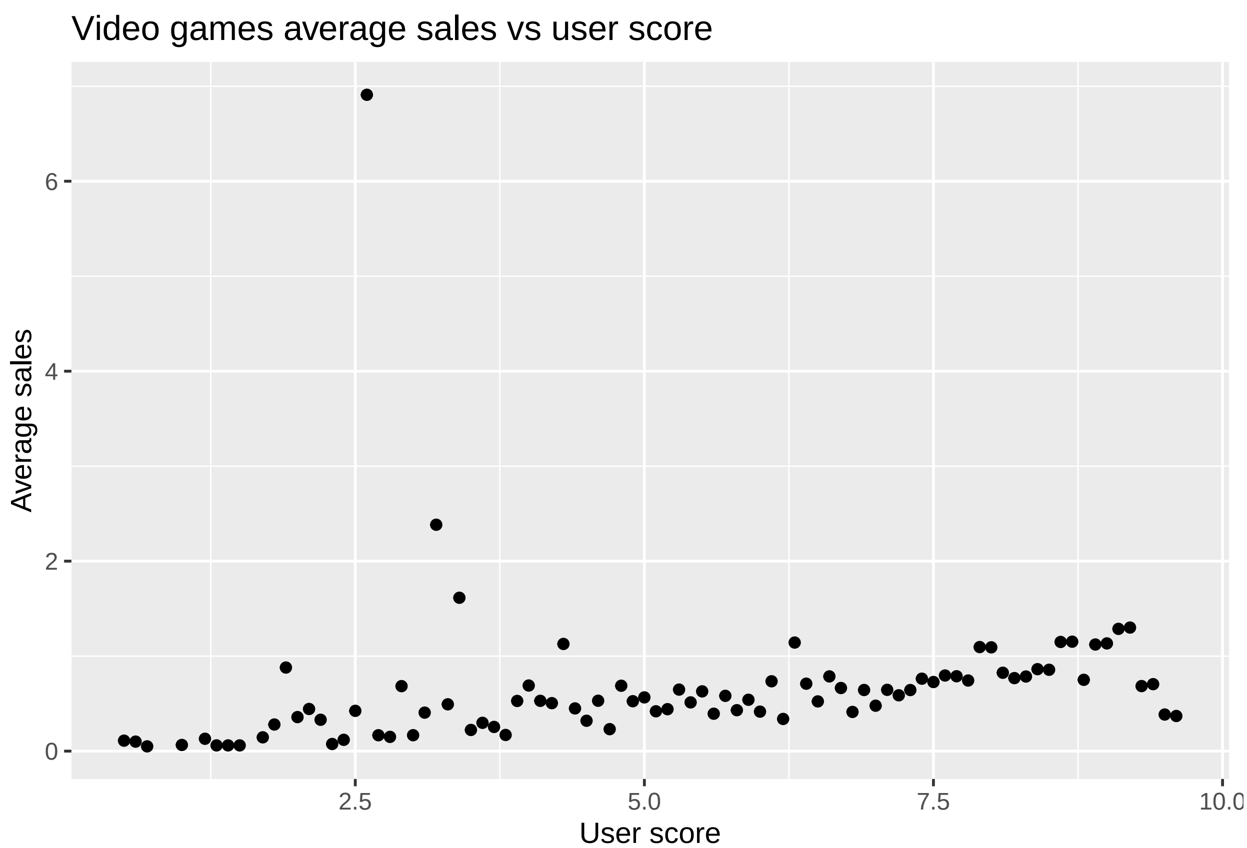
##2.2 Descriptive Analytics

By plotting the average sales against critic and user score, it was observed that there is a correlation between global sales and the scores. An exponential relationship was observed between average sales and critic score. On the other hand, the average sales and user score can be described by having a linear relationship. Both critic and user scores can be expected to have an effect on the model.

```
# Plot average sales vs critic score
train_set %>% group_by(Critic_Score) %>%
  summarize(avg_sales = mean(Global_Sales)) %>%
  ggplot(aes(Critic_Score, avg_sales)) +
  geom_point() +
  labs(title = "Video games average sales vs critic score", x = "Critic score",
       y = "Average sales")
```

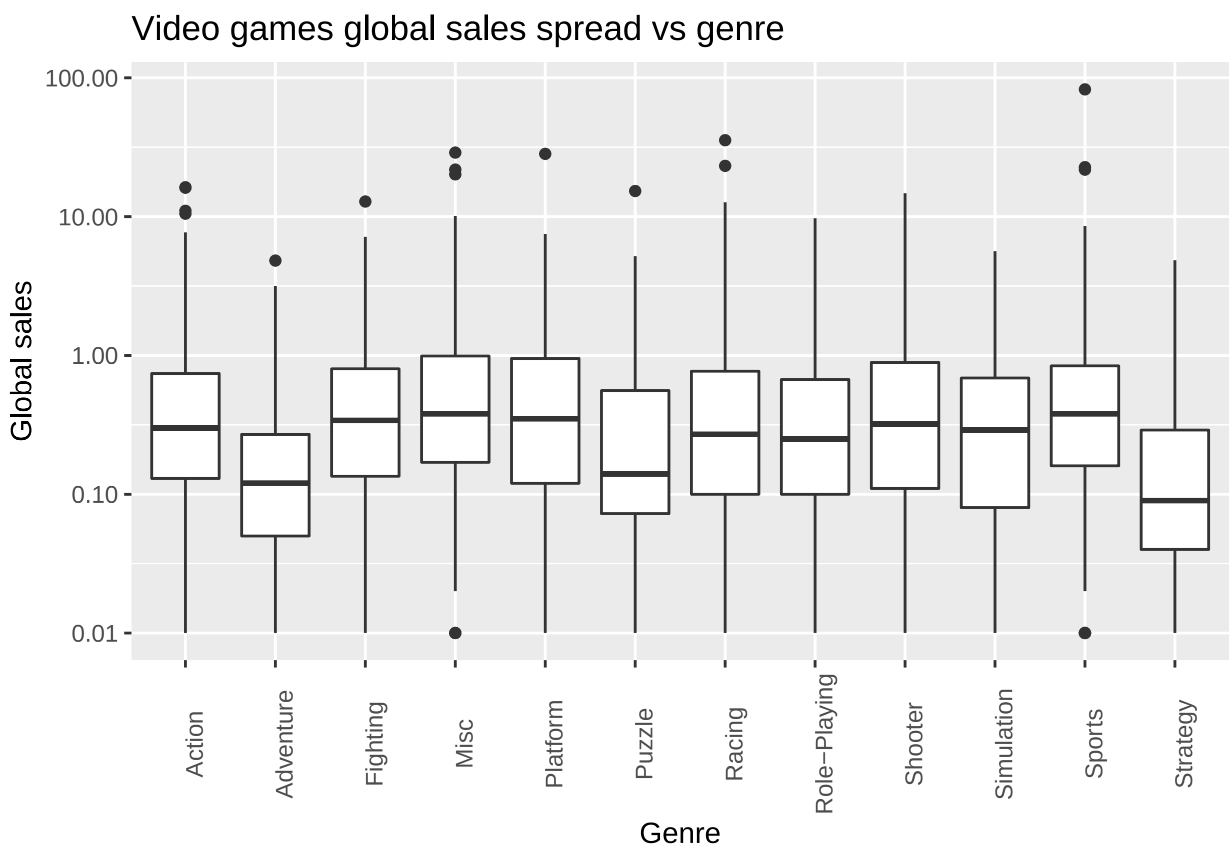



```
# Plot average sales vs user score
train_set %>% group_by(User_Score) %>%
  summarize(avg_sales = mean(Global_Sales)) %>%
  ggplot(aes(User_Score, avg_sales)) +
  geom_point() +
  labs(title = "Video games average sales vs user score", x = "User score",
       y = "Average sales")
```

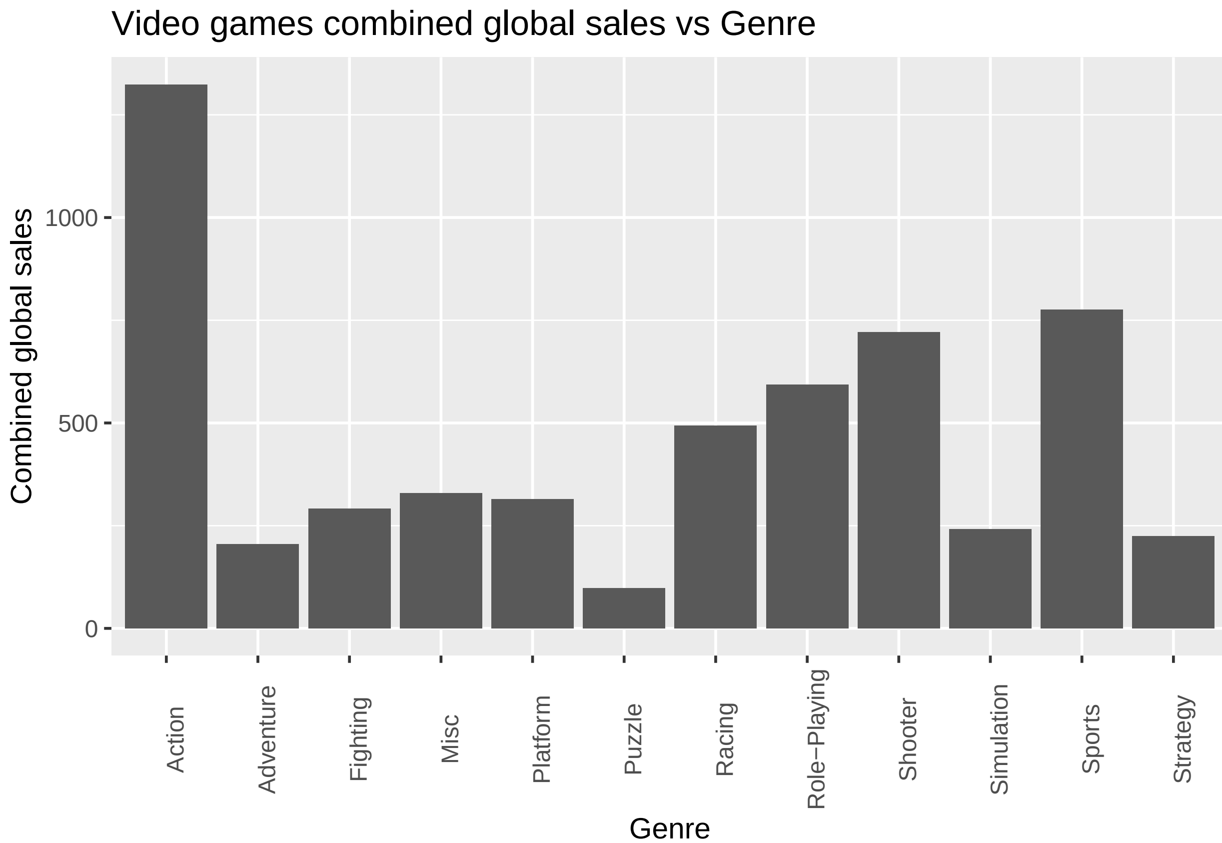


Next, the effects of genres were studied. There appeared to be a significant difference between the best selling genre (action) and the worst selling one (puzzle). Also, there is a noticeable difference between genres in terms of their sales spread.

```
# Plot spread of global sales vs genre
options(scipen=999) # Remove scientific notation
ggplot(train_set, aes(Genre, Global_Sales)) +
  geom_boxplot() +
  scale_y_log10() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(title = "Video games global sales spread vs genre", x = "Genre",
           y = "Global sales")
```

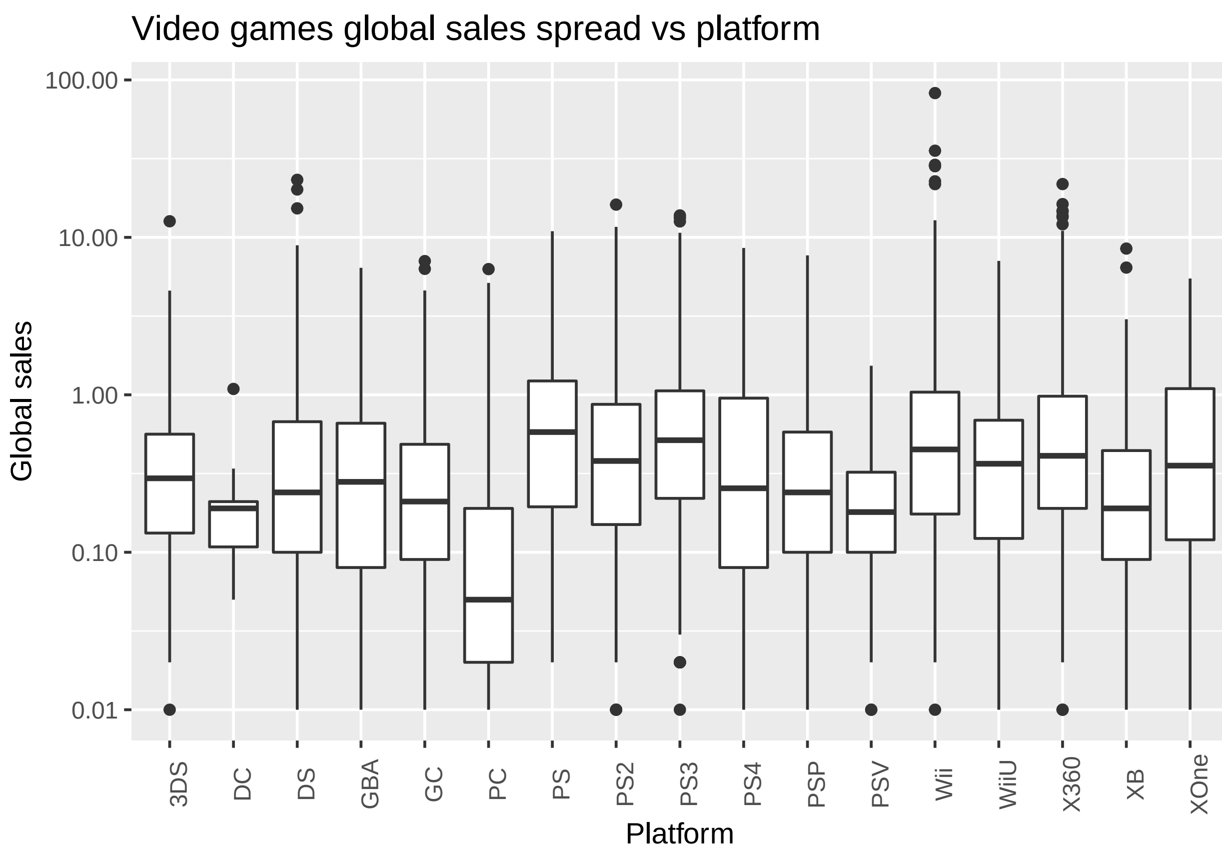



```r
# Plot combined global sales vs genre
train_set %>%
  ggplot(aes(Genre)) +
  geom_bar() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(title = "Video games combined global sales vs Genre", x = "Genre",
       y = "Combined global sales")
```

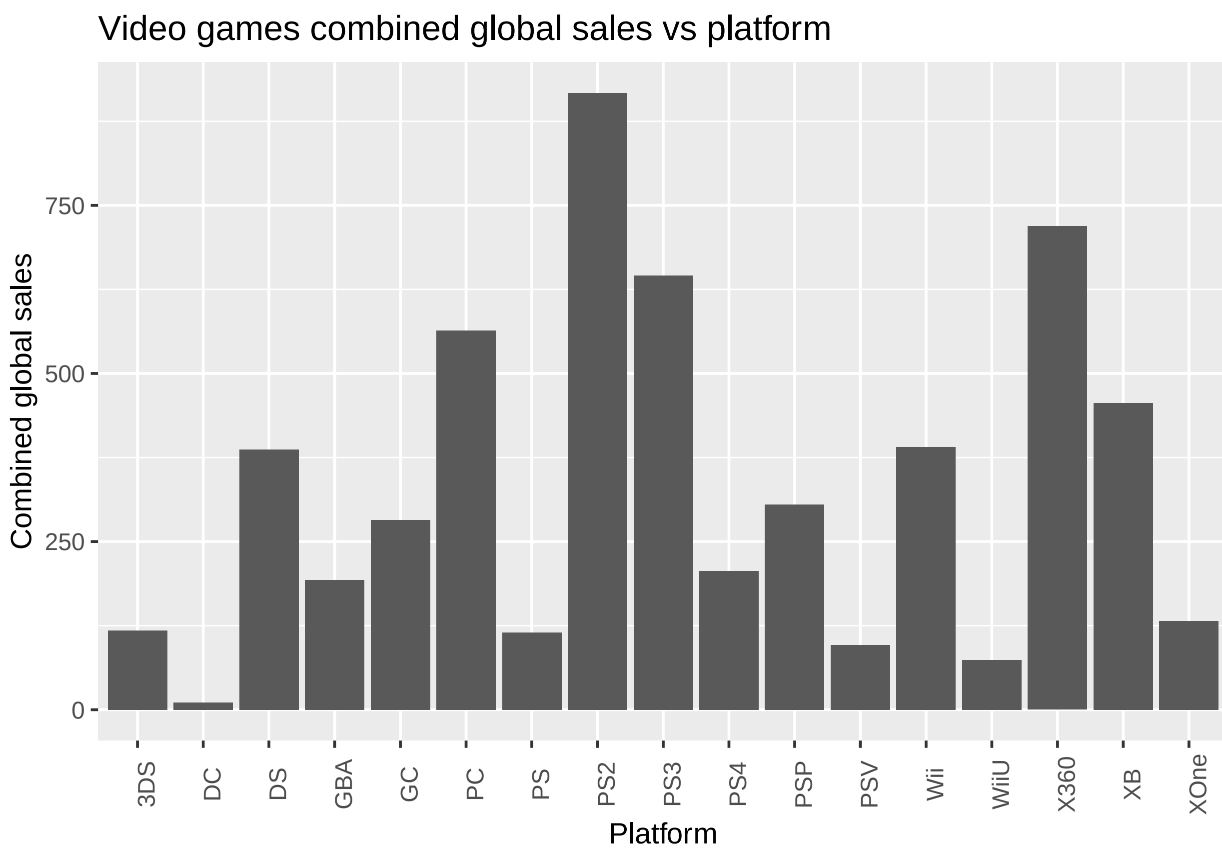


Last but not least, the effects of platforms were studied. Compared to genres, the gap between the best selling platform (PS2) and the worst selling one (DC) was even wider. The same applies for platform sales spread. This phenomenon suggests that platform has a profound effect on the video games sales.

```
# Plot spread of global sales vs platform
ggplot(train_set, aes(Platform, Global_Sales)) +
  geom_boxplot() +
  scale_y_log10() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(title = "Video games global sales spread vs platform", x = "Platform",
       y = "Global sales")
```




```
# Plot combined global sales vs platform
train_set %>%
  ggplot(aes(Platform)) +
  geom_bar() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(title = "Video games combined global sales vs platform", x = "Platform",
       y = "Combined global sales")
```



#3 Methodology

##3.1 Baseline model

A baseline model is established by taking the average of all video games sales. It assumes the same sales without adjusting for any effects such as genre, platform, user or critic scores. Since regression models are utilized in this study, the root-mean-square error (RMSE) is computed for each model and compared. A new value is generated for each model in order to understand how well the model is performing.

```
# Baseline model: Average of global sales
mu <- mean(train_set$Global_Sales)
base_rmse <- RMSE(test_set$Global_Sales, mu)
rmse_table <- tibble(Method = "Baseline average model", RMSE = base_rmse)
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775

##3.2 Linear regression models

Next, the effects of genre, platform, user and critic scores investigated through a series of multivariate regression.

```
# Model 1: Linear regression modeling critic score
model1 <- lm(Global_Sales~Critic_Score, data = train_set)
predicted <- predict(model1, test_set)
```

```r
model1rmse <- RMSE(predicted, test_set$Global_Sales)
rmse_table <- rbind(rmse_table, tibble(Method = "Critic score linear model",
                                       RMSE = model1rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038

```r
# Model 2: Linear regression modeling critic and user scores
model2 <- lm(Global_Sales~Critic_Score+User_Score, data = train_set)
predicted2 <- predict(model2, test_set)
model2rmse <- RMSE(predicted2, test_set$Global_Sales)
rmse_table <- rbind(rmse_table, tibble(Method = "Critic & user scores linear model",
                                       RMSE = model2rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038
Critic & user scores linear model	1.877979

```r
# Model 3: Linear regression modeling platform, critic and user scores
model3 <- lm(Global_Sales~Critic_Score+User_Score+Platform, data = train_set)
predicted3 <- predict(model3, test_set)
model3rmse <- RMSE(predicted3, test_set$Global_Sales)
rmse_table <- rbind(rmse_table,
                    tibble(Method = "Platform, critic & user scores linear model",
                           RMSE = model3rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038
Critic & user scores linear model	1.877979
Platform, critic & user scores linear model	1.839069

```r
# Model 4: Linear regression modeling genre, platform, critic and user scores
model4 <- lm(Global_Sales~Critic_Score+User_Score+Platform+Genre, data = train_set)
predicted4 <- predict(model4, test_set)
model4rmse <- RMSE(predicted4, test_set$Global_Sales)
rmse_table <- rbind(rmse_table, tibble(Method = "Genre, platform, critic &
                                       user scores linear model", RMSE = model4rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038
Critic & user scores linear model	1.877979
Platform, critic & user scores linear model	1.839069
Genre, platform, critic & user scores linear model	1.835694

##3.3 Advanced regression models

Finally, other more advanced regression techniques, namely k-nearest neighbors and random forest were compared against multivariate linear regression.

```
# Model 5: k-Nearest neighbors
model5 <- knnreg(Global_Sales~Critic_Score+User_Score+Platform+Genre,
                       data = train_set)
predicted5 <- predict(model5, test_set)
model5rmse <- RMSE(predicted5, test_set$Global_Sales)
rmse_table <- rbind(rmse_table, tibble(Method = "Genre, platform, critic &
                                       user scores kNN model", RMSE = model5rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038
Critic & user scores linear model	1.877979
Platform, critic & user scores linear model	1.839069
Genre, platform, critic & user scores linear model	1.835694
Genre, platform, critic & user scores kNN model	1.747547

```
# Model 6: Random forests
model6 <- randomForest(Global_Sales~Critic_Score+User_Score+Platform+Genre,
                 data = train_set)
predicted6 <- predict(model6, test_set)
model6rmse <- RMSE(predicted6, test_set$Global_Sales)
rmse_table <- rbind(rmse_table, tibble(Method = "Genre, platform, critic &
                                       user scores random forest model",
                                       RMSE = model6rmse))
knitr::kable(rmse_table)
```

Method	RMSE
Baseline average model	1.960775
Critic score linear model	1.880038
Critic & user scores linear model	1.877979
Platform, critic & user scores linear model	1.839069
Genre, platform, critic & user scores linear model	1.835694

Method	RMSE
Genre, platform, critic & user scores kNN model	1.747547
Genre, platform, critic & user scores random forest model	1.607281

Conclusion

After initial data exploration, it was confirmed through linear regression that genre, platform, user and critic scores was able to improve the sales prediction model. A further comparison was made between linear and other advanced regression techniques to determine which was the best performing in model. The random forest model was considered the best amongst the three due to its superior RMSE value.