

Choose Your Own Project: Video Game Sales with Ratings

Catherine Wong

5/25/2019

Abstract

In this report, a video game sales in North America predictive model will be built based on selected training sets of the Video Game Sales with Ratings. Video game sales from Vgchartz and corresponding ratings from Metacritic dataset are extracted from Kaggle which is an online community of data scientists and machine learners, owned by Google LLC. Four algorithms will be applied: *Linear Regression*, *Polynomial Regression*, *Elastic Net* and *Random Forest*. The result will be compared and analysed by the performance of Residual Mean Squared Error (RMSE).

1. Introduction

Video game is an electronic game that involves interaction with a user interface to generate visual feedback on devices. Playing video games is a kind of popular entertainment for both kids and adults. The market is growing. Publishers would like to predict video game sales for production and better allocation of limited resource. Predictive model is to predict the video games sales in North America based on the Metascore in Metacritic which is a website that aggregates reviews of media products: films, TV shows, music albums, video games, and formerly, books. Metascore is a weighted average of the most respected critics writing reviews online and in print. The scores range from 0 to 100. Scores below 20 represents overwhelming dislike, whereas scores over 90 represents universal acclaim.

Video Game Sales with Ratings dataset from Kaggle website (<https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings>) will be used. In this dataset, 80% of data is set as training data to build the predictive model and the other 20% of data is to evaluate the model by measuring Residual Mean Squared Error (RMSE). Four algorithms are developed for comparison.

The goal of this project is to develop a machine learning algorithm to predict video game sales in North America based on the Metascore. The lower the RMSE, the better the performance of the algorithm.

2. Method

2.1 Data Cleaning

The source data was uploaded to Kaggle on Nov 2016.

```
# Install and Load Packages
if(!require(plotly)) install.packages("plotly", repos = "http://cran.us.r-project.org")
if(!require(corrplot)) install.packages("corrplot", repos = "http://cran.us.r-project.org")
if(!require(RCurl)) install.packages("RCurl", repos = "http://cran.us.r-project.org")
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(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(kableExtra)) install.packages("kableExtra", repos = "http://cran.us.r-project.org")

library(dplyr)
library(ggplot2)
library(caret)
library(tidyr)
library(plotly)
library(RCurl)
library(corrplot)
library(randomForest)
library(kableExtra)

# Video Game Sales with Ratings
# Source File: https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings
# File Download Path: https://github.com/catherine831/Harvard-Prof-Cert-Prog-CYO/raw/master/Video_Games_Sales_as_at_22_Dec_2016.csv"

URL <- tempfile()
download.file("https://github.com/catherine831/Harvard-Prof-Cert-Prog-CYO/raw/master/Video_Games_Sales_as_at_22_Dec_2016.csv",URL)

rawdata <- read.csv(file=URL)
```

The data type and summary statistics of each column of raw data downloaded are as follows:

```
# Raw Data Checking: Type of each Column
str(rawdata)
```

```
## 'data.frame':    16719 obs. of  16 variables:
##  $ Name          : Factor w/ 11563 levels "", "Beyblade Burst",...: 11077 9391 5615 11079
7394 9780 6769 11075 6772 2617 ...
##  $ Platform      : Factor w/ 31 levels "2600","3DO","3DS",...: 27 12 27 27 6 6 5 27 27 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",...: 363 363 363 363 363 363 363
363 363 363 ...
##  $ 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",...: 1022 1 1022 1022 1 1 10
22 1022 1022 1 ...
##  $ Rating        : Factor w/ 9 levels "", "AO", "E", "E10+",...: 3 1 3 3 1 1 3 3 3 1 ...
```

```
# Raw Data Checking: Statistic of each colum
summary(rawdata)
```

```
##                               Name           Platform   Year_of_Release
## Need for Speed: Most Wanted:   12    PS2           :2161    2008    :1427
## FIFA 14                        :    9    DS           :2152    2009    :1426
## LEGO Marvel Super Heroes      :    9    PS3           :1331    2010    :1255
## Madden NFL 07                :    9    Wii           :1320    2007    :1197
## Ratatouille                   :    9    X360         :1262    2011    :1136
## Angry Birds Star Wars         :    8    PSP           :1209    2006    :1006
## (Other)                       :16663   (Other):7284   (Other):9272
##                               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
## Adventure        :1303   THQ                  :  715
## (Other)          :5125   (Other)              :10957
##   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           :6704
## 1st Qu.: 0.0600   1st Qu.:60.00   1st Qu.: 12.00   tbd    :2425
## Median : 0.1700   Median :71.00   Median : 21.00   7.8    : 324
## Mean    : 0.5335   Mean    :68.97   Mean    : 26.36   8      : 290
## 3rd Qu.: 0.4700   3rd Qu.:79.00   3rd Qu.: 36.00   8.2    : 282
## Max.    :82.5300   Max.    :98.00   Max.    :113.00   8.3    : 254
##                               NA's    :8582   NA's    :8582   (Other):6440
##   User_Count      Developer      Rating
## Min.    :    4.0           :6623           :6769
## 1st Qu.:   10.0   Ubisoft : 204   E      :3991
## Median :   24.0   EA Sports: 172   T      :2961
## Mean    :  162.2   EA Canada: 167   M      :1563
## 3rd Qu.:   81.0   Konami   : 162   E10+   :1420
## Max.    :10665.0   Capcom   : 139   EC      :  8
## NA's    :9129   (Other)  :9252   (Other):  7
```

As the dataset were extracted on Nov 2016, records with “Year_of_Release” after 2016 are invalid. Those records marked “NA” are also invalid. These invalid records are required to be removed from the dataset.

```
# Data Cleansing: Remove invalid records of "Year of Release" marked "NA"
cleandata <- rawdata %>% filter(!is.na(rawdata$Year_of_Release))

# Data generated in Nov 2016
# Data Cleansing: Change "Year of Release" to numeric and Remove invalid records of "Year of Release" after 2016
cleandata <- cleandata%>% dplyr::filter((as.numeric(as.character(cleandata$Year_of_Release)))
<=2016)
```

The column “Rating” refers to the ESRB ratings. No “RP” in this rating system and is to be replaced with correct rating.

```
# Data Cleansing: Correct record with wrong "Rating"
cleandata_rp <- cleandata %>% filter(Rating=="RP")
cleandata_rp
```

```
##
## 1 Supreme Ruler: Cold War      PC      2011 Strategy
##
##      Publisher NA_Sales EU_Sales JP_Sales Other_Sales Global_Sales
## 1 Paradox Interactive      0      0.03      0      0.01      0.03
##  Critic_Score Critic_Count User_Score User_Count      Developer
## 1      63      12      6.8      27 BattleGoat Studios
##  Rating
## 1      RP
```

```
cleandata$Rating[cleandata$Rating == 'RP'] <- "E10+"
```

Those records with blank or “NA” rows are also removed from the dataset.

```
# Data Cleansing: Remove invalid records of game with blank in "name"
cleandata <- cleandata %>% filter(cleandata$Name!="")

# Data Cleansing: Change "User_Score" to numeric
cleandata$User_Score <- as.numeric(as.character(cleandata$User_Score))

# Data Cleansing: Remove NA rows
finaldata <- na.omit(cleandata)
```

2.2 Data Exploration

The structure of final dataset is as follows:

2.2.1 No. of Records and no. of video games

```
# Data Exploration: No. of rows and columns final dataset
dim(finaldata)
```

```
## [1] 6894    16
```

```
finaldata_record<-nrow(finaldata)
```

```
# Data Exploration: Statistic of each colum of final dataset
summary(finaldata)
```

```
##                               Name           Platform
## LEGO Star Wars II: The Original Trilogy :    8    PS2       :1140
## Madden NFL 07                          :    8   X360       : 861
## Need for Speed: Most Wanted              :    8   PS3        : 775
## Harry Potter and the Order of the Phoenix:    7    PC        : 688
## Madden NFL 08                          :    7    XB        : 566
## Need for Speed Carbon                   :    7    Wii        : 480
## (Other)                                :6849  (Other):2384
## Year_of_Release      Genre              Publisher
## 2008    : 595    Action      :1644    Electronic Arts      : 945
## 2007    : 590    Sports      : 951     Ubisoft              : 498
## 2005    : 562    Shooter     : 868    Activision           : 492
## 2009    : 554    Role-Playing: 715    Sony Computer Entertainment: 316
## 2006    : 528    Racing      : 586     THQ                  : 307
## 2003    : 499    Platform    : 403    Nintendo             : 293
## (Other):3566    (Other)     :1727    (Other)              :4043
## NA_Sales      EU_Sales      JP_Sales      Other_Sales
## Min.   : 0.0000    Min.   : 0.0000    Min.   :0.00000    Min.   : 0.000
## 1st Qu.: 0.0600    1st Qu.: 0.0200    1st Qu.:0.00000    1st Qu.: 0.010
## Median : 0.1500    Median : 0.0600    Median :0.00000    Median : 0.020
## Mean   : 0.3909    Mean   : 0.2345    Mean   :0.06387    Mean   : 0.082
## 3rd Qu.: 0.3900    3rd Qu.: 0.2100    3rd Qu.:0.01000    3rd Qu.: 0.070
## Max.   :41.3600    Max.   :28.9600    Max.   :6.50000    Max.   :10.570
##
## 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 : 24.00    Median :7.500
## Mean   : 0.7715    Mean   :70.26    Mean   : 28.84    Mean   :7.184
## 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      :2378
## 1st Qu.: 11.0    EA Sports      : 142    E      :2082
## Median : 27.0    Capcom         : 126    M      :1433
## Mean   : 174.4    Ubisoft        : 103    E10+   : 931
## 3rd Qu.: 89.0    Konami         : 97     : 68
## Max.   :10665.0    Ubisoft Montreal: 87    AO      : 1
## (Other)      :6190  (Other): 1
```

```
# Data Exploration: No. of video games in final dataset
n_distinct(finaldata$Name)
```

```
## [1] 4428
```

```
game_no<-n_distinct(finaldata$Name)
```

The total number of records are **6894**.

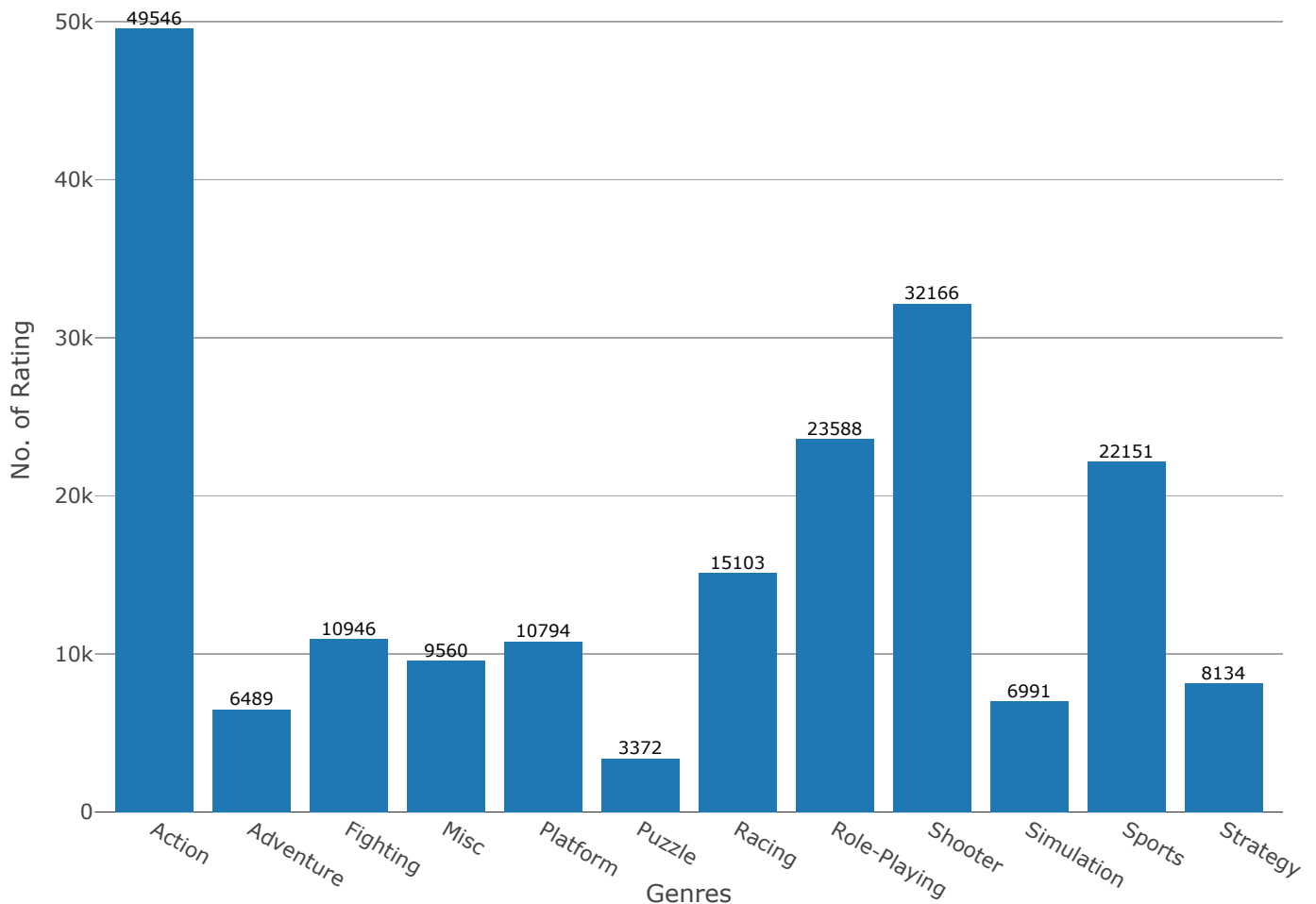
The total number of video games are **4428**.

2.2.2 No. of Ratings by Genres

- Below chart shows top 5 genres are **Action, Shooter, Role-Playing, Sports** and **Racing**.

```
# Data Exploration: No. of Critic ratings in final dataset
finaldata_genres <- finaldata %>% group_by(Genre) %>%
  summarise(Critic_Rating=sum(Critic_Count)) %>%
  arrange(desc(Critic_Rating))

# Data Exploration: No. of Ratings by Genres Plot
finaldata_genres_p <-finaldata_genres%>%plot_ly(
  x = finaldata_genres$Genre,
  y = finaldata_genres$Critic_Rating,
  name = "Rating Distribution by Genres",
  type = "bar"
) %>%
  add_text(text=finaldata_genres$Critic_Rating, hoverinfo='none', textposition = 'top', showlegend = FALSE,
           textfont=list(size=10, color="black"))%>%
  layout(xaxis = list(title = "Genres"),
         yaxis = list(title = "No. of Rating"))
finaldata_genres_p
```



2.2.3 Top 10 video game with the greatest No. of Critic ratings (Metascore)

- Below chart shows top 5 video games with the greatest no. of Critic ratings (Metascore) are **Spider-Man 2**, **Grand Theft Auto V**, **Need for Speed: Most Wanted**, **Tomb Raider: Legend** and **Mass Effect 2**.

```
# Data Exploration: Top 10 video game with the greatest No. of Critic ratings
finaldata_rating <- finaldata %>% group_by(Name) %>%
  summarize(Critic_Rating_Count=sum(Critic_Count)) %>%
  top_n(10) %>%
  arrange(desc(Critic_Rating_Count))
kable(finaldata_rating) %>%
  kable_styling(full_width = F) %>%
  column_spec(1, width = "20em")
```

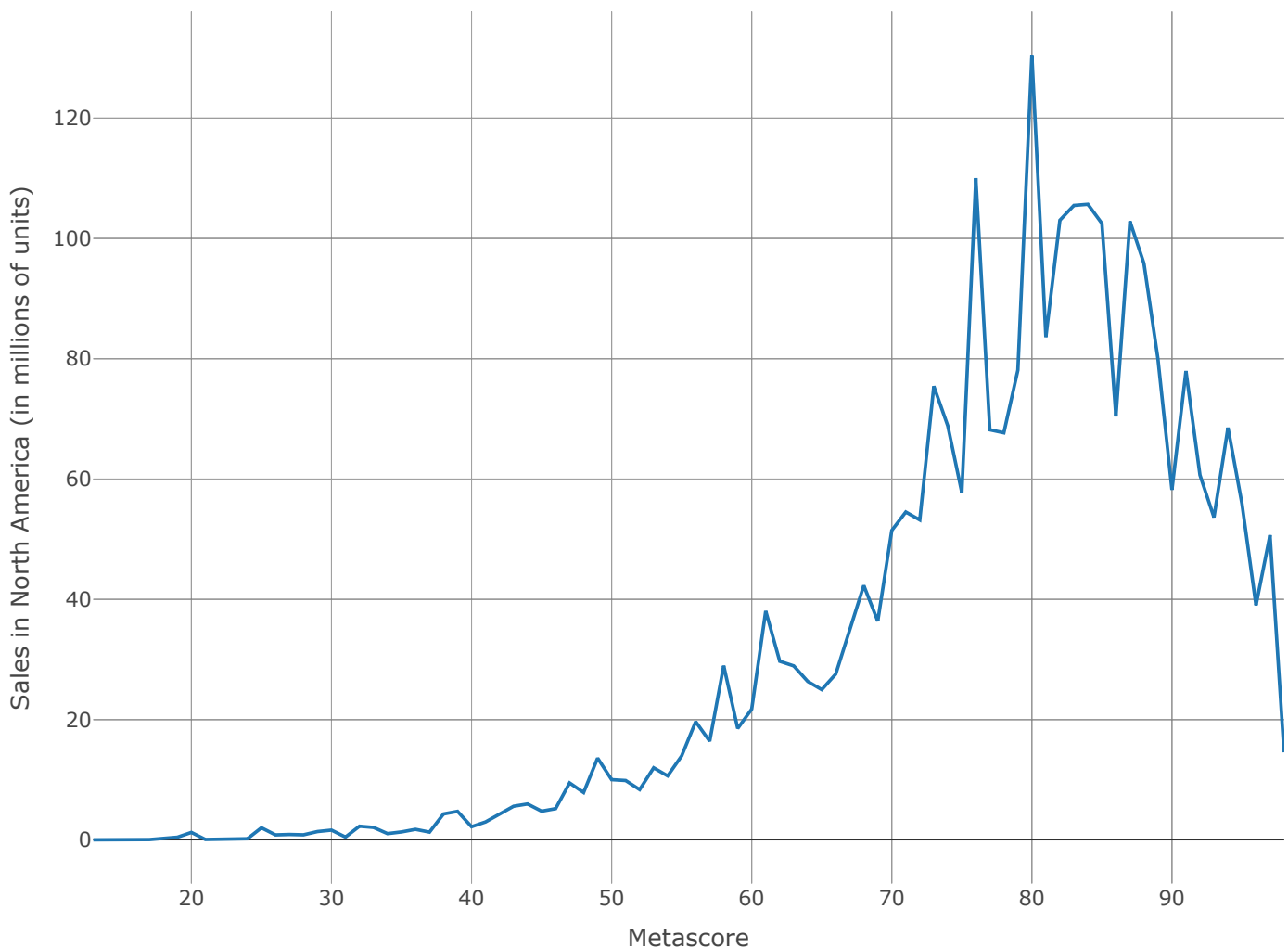
Name	Critic_Rating_Count
Spider-Man 2	252
Grand Theft Auto V	245
Need for Speed: Most Wanted	236
Tomb Raider: Legend	217
Mass Effect 2	215
Call of Duty: Modern Warfare 2	207
Marvel: Ultimate Alliance	204
Madden NFL 07	197
Resident Evil 5	197
Call of Duty: World at War	196
X-Men: The Official Game	196

2.2.4 Sales Trend in North America by Metascore

Below chart shows the sales volume in North America by Metascore

```
# Data Exploration: Sales in North America vs Critic Scores
finaldata_NAsales <- finaldata %>%
  group_by(Critic_Score) %>%
  summarize(NA_Sales=sum(NA_Sales))

# Data Exploration: Sales in North America vs Critic Scores Plot
finaldata_NAsales_p <- plot_ly(finaldata_NAsales, x = ~finaldata_NAsales$Critic_Score, y = ~finaldata_NAsales$NA_Sales, type = 'scatter', mode = 'lines')%>%
  layout(xaxis = list(title = "Metascore"),
         yaxis = list(title = "Sales in North America (in millions of units)"))
finaldata_NAsales_p
```



In general, the sales volume of video game is higher with the higher the Metascore.

2.3 Create a train set and test set from final dataset

80% of final dataset will be set as training data and 20% of final dataset will be the testing data.

```
# test set will be 20% of finaldata
set.seed(1)
NASales_test_index <- createDataPartition(y = finaldata$NA_Sales, times = 1, p = 0.2, list = FALSE)
NASales_train_set <- finaldata[-NASales_test_index,]
NASales_test_set <- finaldata[NASales_test_index,]
```

2.4 RMSE Definition

Evaluation of prediction is based on Residual Mean Squared Error (RMSE). RMSE is the typical error made when predicting sales in North America. The lower the RMSE, the better the performance of the predication.

```
RMSE <- function(true_NA_Sales, predicted_NA_Sales){  
  sqrt(mean((true_NA_Sales - predicted_NA_Sales)^2))  
}
```

2.5 Models

2.5.1 Model: Linear Regression

In this model, variable “Metascore” (Critic_Score) is to predict the sales in North America.

```
# Build the model on train dataset  
lmModel <- lm(NA_Sales ~ Critic_Score, data=NASales_train_set)  
  
# Predict test dataset  
lmPred <- predict(lmModel, NASales_test_set)  
  
# Model prediction performance  
lm_rmse <- RMSE(lmPred, NASales_test_set$NA_Sales)  
lm_rmse
```

```
## [1] 0.8208182
```

```
# Create a Results Table  
rmse_results <- data_frame(method = "Linear Regression", RMSE = lm_rmse)  
rmse_results
```

```
## # A tibble: 1 x 2  
##   method      RMSE  
##   <chr>      <dbl>  
## 1 Linear Regression 0.821
```

2.5.2 Model: Polynomial Regression

A third-degree polynomial formula is developed in this model.

```
# Build the model on train dataset  
polyModel <- lm(NA_Sales ~ Critic_Score+ I(Critic_Score^2) + I(Critic_Score^3), data=NASales_train_set)  
  
# Predict test dataset  
polyPred <- predict(polyModel, NASales_test_set)  
  
# Model prediction performance  
poly_rmse <- RMSE(polyPred, NASales_test_set$NA_Sales)  
poly_rmse
```

```
## [1] 0.7936608
```

```
# Add Polynomial Regression result to the Results Table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Polynomial Regression",
                                     RMSE = poly_rmse))
```

2.5.3 Elastic Net

Elastic Net is a penalized model which is effectively shrink coefficients and to set some coefficients to zero.

```
# Build the model on train dataset
enModel <- train(
  NA_Sales~Critic_Score+ I(Critic_Score^2) + I(Critic_Score^3), data = NASales_train_set, method = "glmnet",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10
)

# Model coefficients
coef(enModel$finalModel, enModel$bestTune$lambda)
```

```
## 4 x 1 sparse Matrix of class "dgCMatrix"
##              1
## (Intercept)  -1.717643e+00
## Critic_Score  1.294229e-01
## I(Critic_Score^2) -2.734746e-03
## I(Critic_Score^3)  1.827375e-05
```

```
# Make predictions
enPred<- enModel %>% predict(NASales_test_set)

# Model prediction performance
en_rmse <- RMSE(enPred, NASales_test_set$NA_Sales)
en_rmse
```

```
## [1] 0.7943541
```

```
# Add Elastic net result to the Results Table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Elastic Net",
                                     RMSE = en_rmse))
```

2.5.4 Random Forest

Random Forest is used to improve prediction performance and reduce instability by averaging multiple decision trees.

```
# Build the model on train dataset
rfModel <- randomForest(NA_Sales ~ Critic_Score, data = NASales_train_set, importance = TRUE)

# Predict test dataset
rfPred <- predict(rfModel, NASales_test_set)

# Model prediction performance
rf_rmse <- RMSE(rfPred, NASales_test_set$NA_Sales)
rf_rmse
```

```
## [1] 0.7771054
```

```
# Add Random Forest result to the Results Table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Random Forest",
                                      RMSE = rf_rmse))
```

3. Results

3.1 Result of Four Models

3.1.1 Model: Linear Regression

RMWE is 0.8208182

3.1.2 Model: Polynomial Regression

RMSE is 0.7936608

3.1.3 Model: Elastic Net

RMSE is 0.7943541

3.1.4 Model: Random Forest

RMSE is 0.7771054

```
kable(rmse_results) %>%
  kable_styling(full_width = F) %>%
  column_spec(1, width = "20em")
```

method	RMSE
Linear Regression	0.8208182
Polynomial Regression	0.7936608
Elastic Net	0.7943541
Random Forest	0.7771054

The **best** model is **Random Forest** with **RMSE 0.7771054**.

4. Conclusions

In this project, the Video Game Sales with Ratings dataset are used to build an algorithm to predict video game sales in North America based on the Metascore. Four models, including “Linear Regression”, “Polynomial Regression”, “Elastic Net” and “Random Forest”, are applied. “**Random Forest**” got the best result, i.e. best RMSE, to predict video game sales in North America by Metascore.