Predicting Global Video Sales Using Machine Learning

Tan Dinh

June 15, 2019

I. Project Overview

This document describes the data and methods used to develop and apply a machine learning algorithm to predict video game global sales. The dataset used on the development and testing of the machine learning algorithm to predict global video sales is titled **Video Game Sales as at 22 Dec 2016** and comes from **Kaggle** internet website, a popular website used in hosting and making datasets publicly available. A machine learning algorithm, using techniques both linear and non-linear, will be developed and tested to predict global sales. The overall accuracy in terms of root mean square (RMSE) value will be used as the metric to measure the overall effectiveness of the algorithm.

II. Introduction

Video games have been around awhile and have seen commercial popularity starting in the 1980s with large consoles for playing games. These consoles are found largely at movies, family entertainment centers, and trendy fast food restaurants. It was not until the introduction of personal video game systems, through companies such as Atari and Nintendo, that video game sales have risen meteorically from its commercial application. Since then, Microsoft Xbox and Sony Playstation, have overtaken sales and popularity as today's personal video game console of choice.

With its popularity continuing to rise, increasingly among adults, there is a need to understand what is driving the global sales of video games. In this project, a review of features through the application of data science and machine learning is examined for their effects on global sales of video games. In question is what are the parameters in video games that drive as well as those that do not drive global sales.

III. Methods and Analysis

The method and analysis used to develop the machine learning algorithm for predicting global sales follow the methods found in Rafael Irizarry"s Harvard EDX class titled **Machine Learning**. This online class teaches basic principles and techniques of machine learning. Of interest, from the course, the techniques and applications of regression used in the Movielens example and non-parametric regression modeling such as locally weighted regression (LOESS) used in the polls_2008 example of the course are followed closely and applied here to this project.

A. Exploratory Data Analysis

A1. *Data Extraction*

The **Video Game Sales as at 22 Dec 2016** was downloaded from the Kaggle website using the following link:

https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings

Downloading at Kaggle requires an account which is available and free upon sign-up at the web site. This downloaded document is provided in a compressed, zip file format that requires extraction before usage in a spreadsheet or database application. For this project, the file was downloaded and extracted as a csv file. The data analysis and work were performed on a computer using Rstudio, a computer platform application that uses the R computer language. The data analysis, machine learning code, and report write-up were all completed on Rstudio.

Following Rafael Irizarry's **Machine Learning** class, the data file was then partitioned into two datasets, (1) a training set and (2) a test set. The training set is used to model the data and is partitioned from the original dataset and contains roughly 90 percent of the original data. The test set is used to validate the model and contains the remainder or about 10 percent of the overall data.

A2. *Data Cleaning*

Data in its raw state generally has some defects or deficiencies such as format or errors that require correction. This task involves some type of data cleaning or curating the data to make it useful for the application of this project.

Initial review showed that there is missing, blank, or non-recorded observations as well as data recorded as N/As in the dataset. Because a complete set of data is an essential requirement for machine learning application, missing data and N/As were removed from the dataset. The original dataset contained 16719 observations. Because of the cleaning and removal of non-recorded observations (blanks) and recorded missing observations (N/As), the curated data was left with only 6825 observations, leaving roughly less than half of the data from the original dataset.

A3. *Data Exploration*

Having a curated data file, the data from the file is partitioned into two sets of data: a training set and a test set. The training set is used for model development while the test set is used for tesing the accuracy of the model. In this section data exploration is performed on the training set.

The first exploration looks at the features and observations in the data. There are 6825 observations and 16 features in the dataset. They are listed below.

```r
dim(train_set)
```

```
## [1] 6152   16
```

A summary of the data layout is given below. The observations contain both numerical and character values. Additionally, the year is given as a data value.

```r
glimpse(train_set)
```

```
## Observations: 6,152
## Variables: 16
## $ Name            <chr> "Wii Sports", "Mario Kart Wii", "Wii Sports Re...
## $ Platform        <chr> "Wii", "Wii", "Wii", "DS", "Wii", "Wii", "DS",...
## $ Year_of_Release <chr> "2006", "2008", "2009", "2006", "2006", "2009"...
## $ Genre           <chr> "Sports", "Racing", "Sports", "Platform", "Mis...
## $ Publisher       <chr> "Nintendo", "Nintendo", "Nintendo", "Nintendo"...
## $ 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    <dbl> 76, 82, 80, 89, 58, 87, 91, 80, 61, 80, 97, 95...
## $ Critic_Count    <dbl> 51, 73, 73, 65, 41, 80, 64, 63, 45, 33, 50, 80...
## $ User_Score      <chr> "8", "8.3", "8", "8.5", "6.6", "8.4", "8.6", "...
## $ User_Count      <dbl> 322, 709, 192, 431, 129, 594, 464, 146, 106, 5...
## $ Developer       <chr> "Nintendo", "Nintendo", "Nintendo", "Nintendo"...
## $ Rating          <chr> "E", "E", "E", "E", "E", "E", "E", "E", "E", "...
```

To help visualize the data, the first ten rows of the data are given below.

First Ten Rows of Dataset in three chunks:

```r
train_set1  <-  train_set %>%
  select(1:5)
head(train_set1, n=10) %>%
  knitr::kable()
```

Name	Platform	Year_of_Release	Genre	Publisher
Wii Sports	Wii	2006	Sports	Nintendo
Mario Kart Wii	Wii	2008	Racing	Nintendo
Wii Sports Resort	Wii	2009	Sports	Nintendo
New Super Mario Bros.	DS	2006	Platform	Nintendo
Wii Play	Wii	2006	Misc	Nintendo
New Super Mario Bros. Wii	Wii	2009	Platform	Nintendo
Mario Kart DS	DS	2005	Racing	Nintendo
Wii Fit	Wii	2007	Sports	Nintendo
Kinect Adventures!	X360	2010	Misc	Microsoft Game Studios
Wii Fit Plus	Wii	2009	Sports	Nintendo

```r
train_set1  <-  train_set %>%
  select(6:10)
head(train_set1, n=10) %>%
  knitr::kable()
```

NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
41.36	28.96	3.77	8.45	82.53
15.68	12.76	3.79	3.29	35.52
15.61	10.93	3.28	2.95	32.77
11.28	9.14	6.50	2.88	29.80
13.96	9.18	2.93	2.84	28.92
14.44	6.94	4.70	2.24	28.32
9.71	7.47	4.13	1.90	23.21
8.92	8.03	3.60	2.15	22.70
15.00	4.89	0.24	1.69	21.81
9.01	8.49	2.53	1.77	21.79

```r
train_set1  <-  train_set %>%
  select(11:16)
head(train_set1, n=10) %>%
  knitr::kable()
```

Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
76	51	8	322	Nintendo	E
82	73	8.3	709	Nintendo	E
80	73	8	192	Nintendo	E
89	65	8.5	431	Nintendo	E
58	41	6.6	129	Nintendo	E
87	80	8.4	594	Nintendo	E
91	64	8.6	464	Nintendo	E
80	63	7.7	146	Nintendo	E
61	45	6.3	106	Good Science Studio	E
80	33	7.4	52	Nintendo	E

Statistical analysis was reviewed on the data. The table below presents a five number summary of the global sales data.

Five Number Summary for Global Sales

```
train_set2 <- train_set %>%
  select(Global_Sales)
summary(train_set2)
```

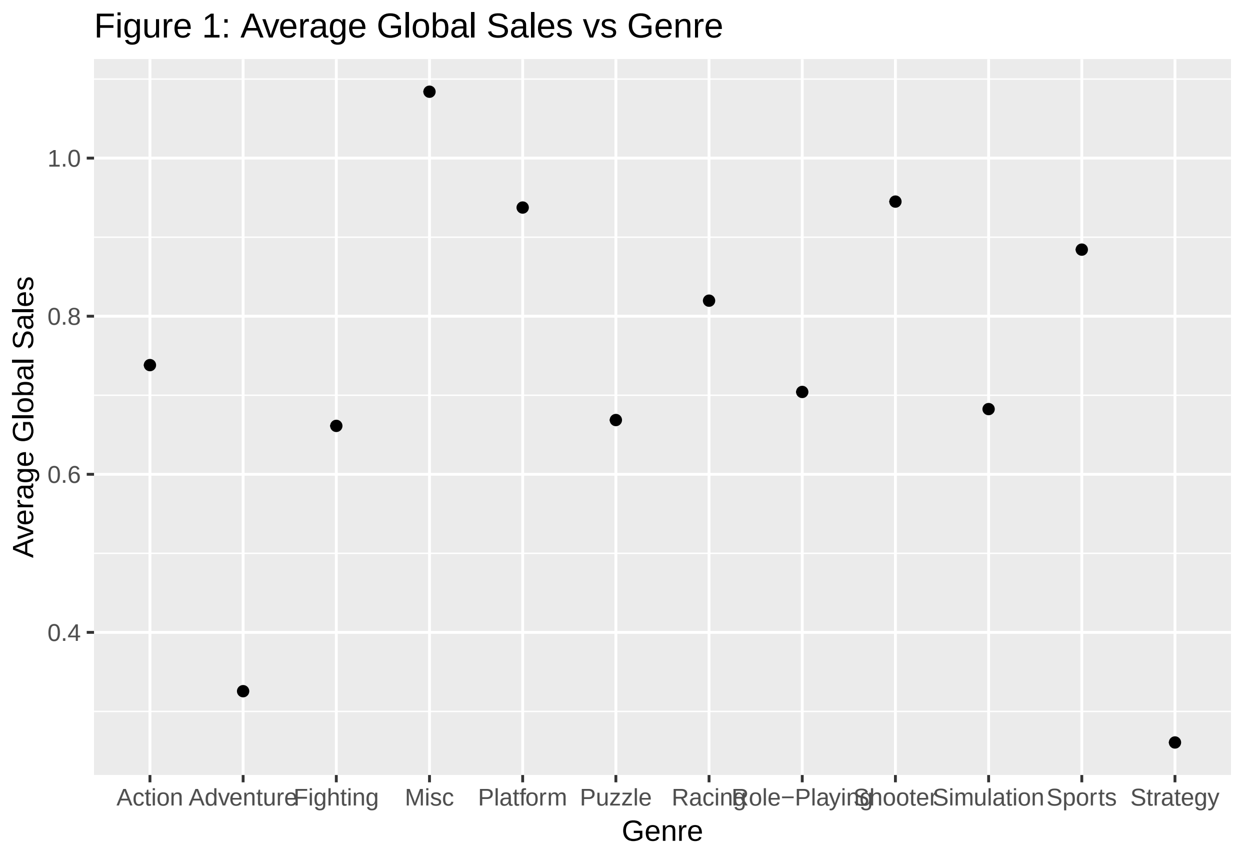
```
##   Global_Sales
##  Min.   : 0.0100
##  1st Qu.: 0.1100
##  Median : 0.2900
##  Mean   : 0.7857
##  3rd Qu.: 0.7500
##  Max.   :82.5300
```

A4. *Data Visualization*

The data was plotted to see different trends in the data.

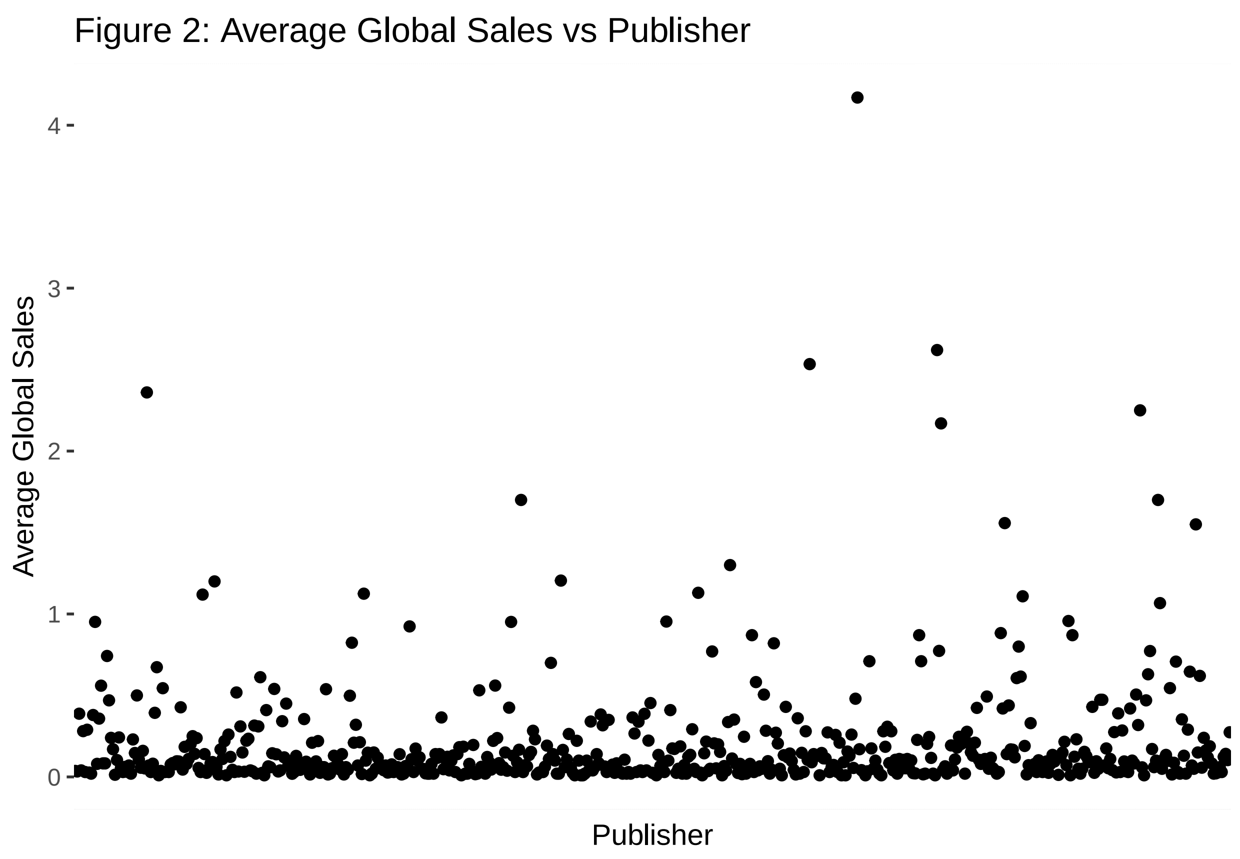
Figure 1 shows a scatter plot of global sales average versus genre. From this figure, we see that genre does have an impact on global sales. The miscelaneous genre is the rate highest in terms of global sales while the strategy genre has the lowest global sales overall. Also, the average global sales for each genre ranges roughly from 0.25 to 1.10.

```
#plot by Genre
c <- video_game_sales_final %>%
  mutate(n=1) %>%
  group_by(Genre) %>%
  summarize(average=mean(Global_Sales))

ggplot(c, aes(x=Genre, average)) +
  geom_point()+
  labs(y = "Average Global Sales")+
  ggtitle("Figure 1: Average Global Sales vs Genre")
```



The publisher effect on global sales was also reviewed to see any corresponding trends. The average global sales for each publisher ranges roughly from 0 to 4.5.

```r
c <- video_game_sales %>%
  mutate(n=1) %>%
  group_by(Publisher) %>%
  summarize(average=mean(Global_Sales))

ggplot(c, aes(Publisher, average))+
  geom_point()+
  ggtitle("Figure 2: Average Global Sales vs Publisher")+
  labs(y = "Average Global Sales")+
  theme(axis.text.x=element_blank(),
        axis.ticks.x=element_blank())
```



An analysis of the year the video game was release compared to global sales. From Figure 4, the year 1985-1988 shows video sales to be at the lowest point from 1986 to 2016, the range of our dataset. Also, the average global sales for each Year of Release ranges roughly from 0 to 3. There is some yearly trend noticeable in the figure.

```r
c <- video_game_sales_final %>%
  mutate(n=1) %>%
  group_by(Year_of_Release) %>%
  summarize(average=mean(Global_Sales))
d <- c %>%
  mutate(year=as.numeric(Year_of_Release))

ggplot(d, aes(Year_of_Release, average))+
  geom_point() +
  ggtitle("Figure 3: Average Global Sales vs Year of Release")+
    labs(y = "Average Global Sales", x="Year of Release")+
  scale_x_discrete(breaks = seq(1985, 2016, by = 5))
```

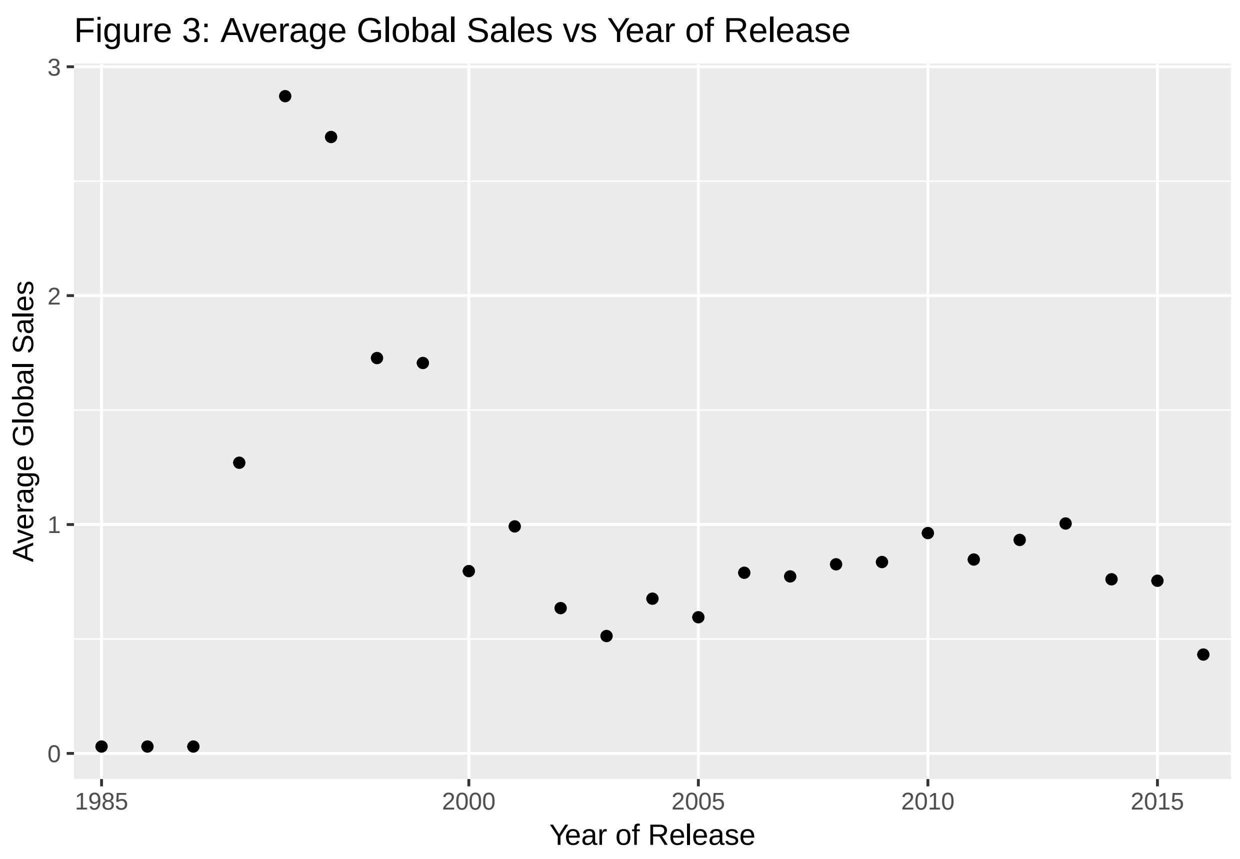
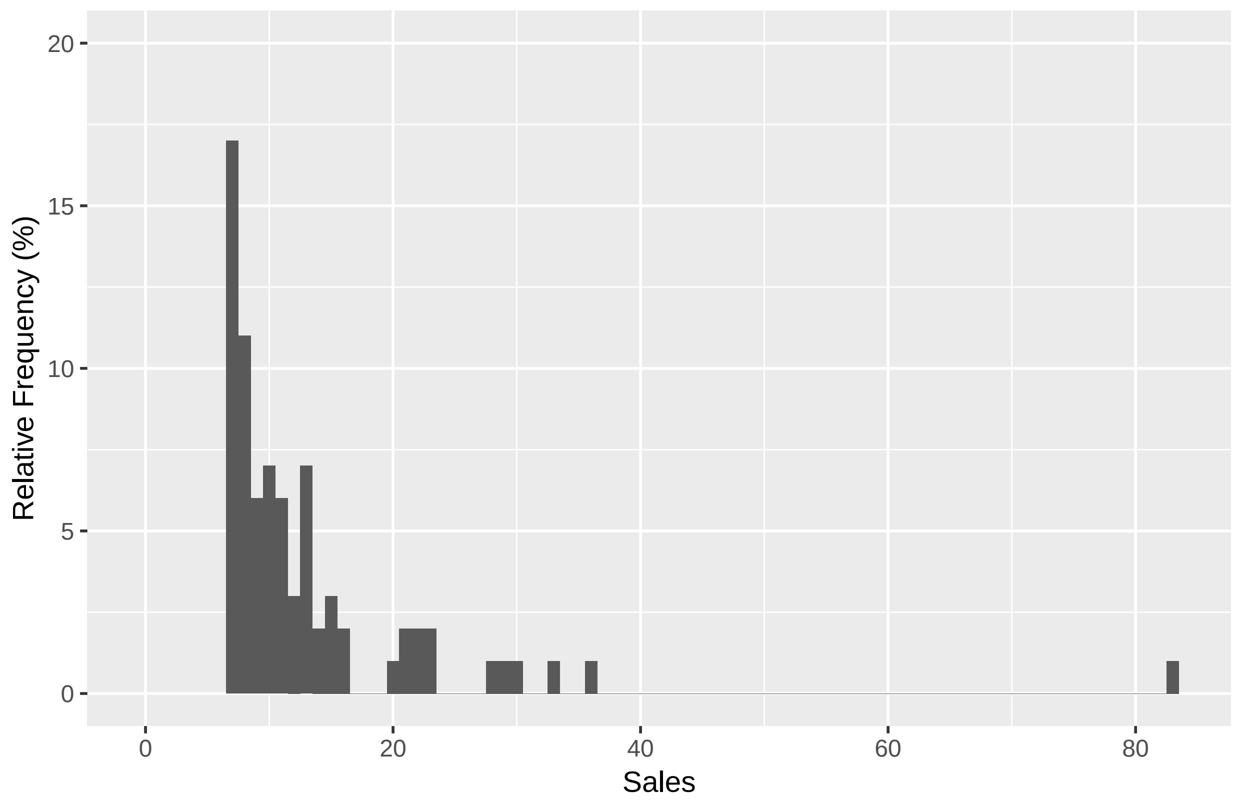


Figure 4 below shows a histogram of global sales relative frequency. Most of the games have global sales within roughly 5 to 10 million units range. The largest global sale is over 80. There is a large gap between 40 to 80 million units range, making the 80 million units point an obvious outlier. It was found that only game fit this point.

```
# d <- train_set %>%
#   ggplot(aes(Global_Sales)) +
#   geom_histogram(binwidth = 1) +
#   scale_x_continuous() +
#   scale_y_continuous() +
#   xlab("Sales") +
#   ylab("Relative Frequency") +
#   ggtitle("Global Sale Histogram")+
#   ylim(0, 20)
```

Figure 4: Global Sale Histogram



B. Data Modeling

To build an algorithm based on observations of features, various methods will be utilized to predict global sales based on features. In this project, we will only focused on the following features: (1) year of release, (2) genre, and (3) publisher. The models use include just the average, linear regression, local weighted regression (LOESS), and prediction based on genre and publisher effects.

B1. *Modeling Using the Global Sales Average*

This model takes global sales and used an average value of the total sales as the predictor. The predictor is then tested on the test set to determine the RMSE. This will serve as the baseline for comparison with RMSEs from other models.

mu=sum(globalsales/number of global sales)

y_hat = mu

where:

y_hat = predicted global sales for any input

mu = average of total sales

B2. *Linear Model Global Sales Average by Year of Release*

The linear regression model uses simple linear regression to create a prediction line based on the relationship between the year of release and global sales.

y_hat = b_o + b_1 * x

where:

y_hat = predicted global sales for a given year

x = year of release

b_o = y-intercept for regression line

b_1 = slope of regression line

B3. *Local Weighted Regression Model (LOESS)*

This method utilizes a localized regression based on a moving window on years of release. Two models of LOESS in R are calculated. One version is using degree 1 where a localized linear regression is performed. The other uses degree 2 which is parabolic. See the Results section below for R code of this model.

B4. *Model Based on Genre and Publisher Effect*

To build an algorithm based on observations of features, an equation was developed and modeled to predict video global sales. This equation correlates features to global sales and takes account errors and biases in the features.

We begin with a baseline model (see B1 above) where we assume an average value of global sales and calculate the root mean square error (RMSE) value on it. This simple method will establish a baseline RMSE value that uses the same predictor (average global sales) for all titles in our dataset.

To improve this model, we add the genre and publisher classes biases. The modeling equation is provided as:

y_g,p = mu + b_g + b_p + e_g,p

where:

y_g,p= predicted global sale for all video titles, for genre (g) for publisher (p)

mu = “true” global sales of all videos

b_g = genre effect for video titles (g)

b_p = publisher effect for title (p)

e_g,p= residual (errors)

IV Results

The results are presented in this section. For each model, an RMSE value was calculated based on predicted value for the test set.

Result for Model B1

The average model was first used to established an RMSE baseline. In this model, the average global sales was used as a predictor (in this case a constant value) to estimate the RMSE value against the test set values. An RMSE of 2.027 was calculated based on this average value, see Table 1 below.

The value of mu is this:

```r
mu <- mean(train_set$Global_Sales)
  #define mu (average) of ratings column in train_video_sales
  #dataset global sales (in millions of units)
mu
```

```
## [1] 0.7857168
```

The value of the rmse for this model is this:

```r
naive_rmse <- RMSE(train_set$Global_Sales, mu)
```

method	RMSE
Just the Average	2.027919

Result for Model B2

The linear model was first used to established a second RMSE. In this model, a linear regression was used to obtain an RMSE value on global sales. An RMSE of 1.224 was calculated based on this average value which is an improvement over model B1.

```r
#-----------------------------------------------------------------------

  # Since Year_Of_Release is type chr, I add column that converts year to numeric.
  train_set_numyr <- train_set %>% mutate(numyr = as.numeric(Year_of_Release))
  test_set_numyr <- test_set %>% mutate(numyr = as.numeric(Year_of_Release))

# - ---------------------------------------------------------------------

  fit_lm <- lm(Global_Sales ~ numyr, data=train_set_numyr)
  lm_predict <- predict(fit_lm, test_set_numyr)
  rmse_lm <- RMSE(test_set_numyr$Global_Sales, lm_predict)

  rmse_results <- bind_rows(rmse_results,
                        tibble(method="Linear Model",
                               RMSE = rmse_lm))  # putting data into a table
  rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	2.027919
Linear Model	1.223772

Results for Model B3

Third, the LOESS model was used to established RMSEs with values at degree 1 (linear) and degree 2 (parbolic). The RMSE values for LOESS degree 1 and degree 2 are 1.215, 1.209 respectively. Both are slight improvement over Model B2, the linear model.

```r
    # loess degree 1
  fit_loess1 <- loess(Global_Sales ~ numyr, degree=1, data=train_set_numyr)
  loess1_predict <- predict(fit_loess1, test_set_numyr)
  rmse_loess1 <- RMSE(test_set_numyr$Global_Sales, loess1_predict)
  rmse_loess1
```

```
## [1] 1.215092
```

```
  # putting data into a table
  rmse_results <- bind_rows(rmse_results,
                          tibble(method = "LOESS degree 1", RMSE = rmse_loess1))

    # loess degree 2
  fit_loess2 <- loess(Global_Sales ~ numyr, degree=2, data=train_set_numyr)
  loess2_predict <- predict(fit_loess2, test_set_numyr)
  rmse_loess2 <- RMSE(test_set_numyr$Global_Sales, loess2_predict)
  rmse_loess2
```

```
## [1] 1.20936
```

```
  # putting data into a table
  rmse_results <- bind_rows(rmse_results,
                          tibble(method = "LOESS degree 2", RMSE = rmse_loess2))

rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	2.027919
Linear Model	1.223772
LOESS degree 1	1.215092
LOESS degree 2	1.209360

Result for Model B4

Lastly, the model based on the genre and publisher effect was used to calculate an RMSE value. In this model, both genre and publisher effects were modeled as biases into the equation to obtain an RMSE value on global sales. The RMSE value for this model is 1.201 which is the best value among all models that we considered in this project.

```
  ### incorporating genre and publisher effect to overall equation
genre_avgs <- train_set %>%  #incorporating genre effect into equation, b_i is for genre
  group_by(Genre) %>%
  summarize(b_g = mean(Global_Sales - mu))

predicted_sales <- mu +
  test_set %>%  #testing predicted Global_Sales data to test_validation dataset
  left_join(genre_avgs, by = 'Genre') %>%
  pull(b_g)

# model_1_rmse <- RMSE(predicted_sales, test_set$Global_Sales)  #this test genre effect

publisher_avgs <- train_set %>%
  left_join(genre_avgs, by='Genre') %>%
  group_by(Publisher) %>%
  summarize(b_publisher = mean(Global_Sales - mu - b_g))

predicted_sales_publisher <- test_set %>%   #putting both movie and user effects into model
  left_join(genre_avgs, by='Genre') %>%
  left_join(publisher_avgs, by='Publisher') %>%
  mutate(pred = mu +b_publisher + b_g) %>%
  #na.omit() %>%
  pull(pred)

model_2_rmse <- RMSE(predicted_sales_publisher, test_set$Global_Sales)
  #calculate residual mean square error for the two effects
rmse_results <- bind_rows(rmse_results,
                          tibble(method="Publisher + Genre Effects Model",
                                 RMSE = model_2_rmse))  # putting data into a table

rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	2.027919
Linear Model	1.223772
LOESS degree 1	1.215092
LOESS degree 2	1.209360
Publisher + Genre Effects Model	1.200712

V Conclusion

In this project, several machine learning algorithms were developed to predict global sales of video games. A summary of the performance of each algorithm is presented in the following table:

```
rmse_results %>% knitr::kable()
```

method	RMSE
Just the Average	2.027919
Linear Model	1.223772
LOESS degree 1	1.215092
LOESS degree 2	1.209360
Publisher + Genre Effects Model	1.200712

We started with a baseline model which only consider the global sale average as the predictor. Then, we tried to improve upon the prediction with the development of the linear model which showed substantial improvement from the baseline model with an RMSE of 2.028 down to 1.224. To seek further improvement, we look at LOESS with degree 1 and degree 2, which showed additional RMSE reductions down to 1.215 and 1.209 respectively. As a last model, we used the publisher and genre effect model, which showed the most improvement among all models considered, with an RMSE value of 1.201.