Linear Regression on Top Selling Video Games

vgSales is a data set with 55792 rows and 23 columns about the top selling video games and their scores, rating, genre, etc.

Data found here: https://www.kaggle.com/datasets/ashaheedq/video-games-sales-2019 (https://www.kaggle.com/datasets/ashaheedq/video-games-sales-2019)

Manipulating Data

```
df <- read.csv("vgsales-short.csv")
str(df)
```

```
## 'data.frame':    55792 obs. of  16 variables:
##  $ Rank         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Name         : chr  "Wii Sports" "Super Mario Bros." "Mario Kart Wii" "PlayerUnknown's
Battlegrounds" ...
##  $ Genre        : chr  "Sports" "Platform" "Racing" "Shooter" ...
##  $ ESRB_Rating  : chr  "E" "" "E" "" ...
##  $ Platform     : chr  "Wii" "NES" "Wii" "PC" ...
##  $ Publisher    : chr  "Nintendo" "Nintendo" "Nintendo" "PUBG Corporation" ...
##  $ Developer    : chr  "Nintendo EAD" "Nintendo EAD" "Nintendo EAD" "PUBG Corporation" ...
##  $ Critic_Score : num  7.7 10 8.2 NA 8 9.4 9.1 NA 8.6 10 ...
##  $ User_Score   : num  NA NA 9.1 NA 8.8 NA 8.1 NA 9.2 NA ...
##  $ Total_Shipped: num  82.9 40.2 37.1 36.6 33.1 ...
##  $ Global_Sales : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ NA_Sales     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ PAL_Sales    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ JP_Sales     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Other_Sales  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Year         : num  2006 1985 2008 2017 2009 ...
```

There seems to be a lot of rows that are largely empty and don't contain the data we want. We'll trim this down to the rows that contains data in at least one of User_Score or Critic_Score. When Total_Shipped is N/A Global_Sales isn't, and vice versa, and also seem to be used interchangeably in this ranking, so we will fill in any N/As wherever possible. NOTE: The most sold game is Wii Sports because it was included in every single Wii sale by default. For this reason it is skewing the data and will not be included.

```
df <- df[!with(df, is.na(df$Critic_Score)& is.na(df$User_Score)),]
df2 <- df[-1,]
df2$Total_Shipped[is.na(df2$Total_Shipped)] <- df2$Global_Sales[is.na(df2$Total_Shipped)]
```

Divide into train and test sets

Randomly sample rows to get a vector i with row indices to help divide into train/test sets. We are using an 80/20 ratio

```
set.seed(1234)
i <- sample(1:nrow(df2), nrow(df2)*0.8, replace=FALSE)
train <- df2[i,]
test <- df2[-i,]
```

Data exploration

```
str(train)
```

```
## 'data.frame':    5321 obs. of  16 variables:
##  $ Rank         : int  1758 896 6528 1604 16760 7586 4628 8564 6856 35010 ...
##  $ Name         : chr  "[Prototype]" "Saints Row" "Etrian Odyssey III: The Drowned City" "
Metroid Prime 3: Corruption" ...
##  $ Genre        : chr  "Action" "Action" "Role-Playing" "Shooter" ...
##  $ ESRB_Rating  : chr  "M" "M" "E10" "T" ...
##  $ Platform     : chr  "X360" "X360" "DS" "Wii" ...
##  $ Publisher    : chr  "Activision" "THQ" "Atlus" "Nintendo" ...
##  $ Developer    : chr  "Radical Entertainment" "Volition Inc." "Atlus Co. / Lancarse" "Ret
ro Studios" ...
##  $ Critic_Score : num  7.9 8 7.3 9 5.9 4.5 3.8 3.1 6.9 8 ...
##  $ User_Score   : num  NA NA NA 9.4 NA NA NA NA NA NA ...
##  $ Total_Shipped: num  1.31 2.18 0.31 1.41 0.03 0.25 0.49 0.21 0.29 NA ...
##  $ Global_Sales : num  1.31 2.18 0.31 NA 0.03 0.25 0.49 0.21 0.29 NA ...
##  $ NA_Sales     : num  0.84 1.17 0.12 NA 0.02 0.24 0.33 0.17 0.24 NA ...
##  $ PAL_Sales    : num  0.35 0.77 NA NA 0.01 NA 0.12 0.04 0.03 NA ...
##  $ JP_Sales     : num  NA 0.02 0.19 NA NA NA NA NA NA NA ...
##  $ Other_Sales  : num  0.12 0.22 0.01 NA 0 0.02 0.04 0 0.02 NA ...
##  $ Year         : num  2009 2006 2010 2007 1999 ...
```

```
head(train, 50)
```

	Rank	Name
	<int>	<chr>
1758	1758	[Prototype]
896	896	Saints Row
6528	6528	Etrian Odyssey III: The Drowned City
1604	1604	Metroid Prime 3: Corruption
16760	16760	Charlie Blasts Territory
7586	7586	Spelling Challenges and more!
4628	4628	Babysitting Mama

	Rank <int>	Name <chr>
8564	8564	ClayFighter 63 1/3
6856	6856	Zoo Tycoon 2 DS
35010	35010	Kahoots

1-10 of 50 rows | 1-3 of 17 columns Previous **1** 2 3 4 5 Next

```
nrow(train[with(train, !is.na(train$Critic_Score)),])
```

```
## [1] 5232
```

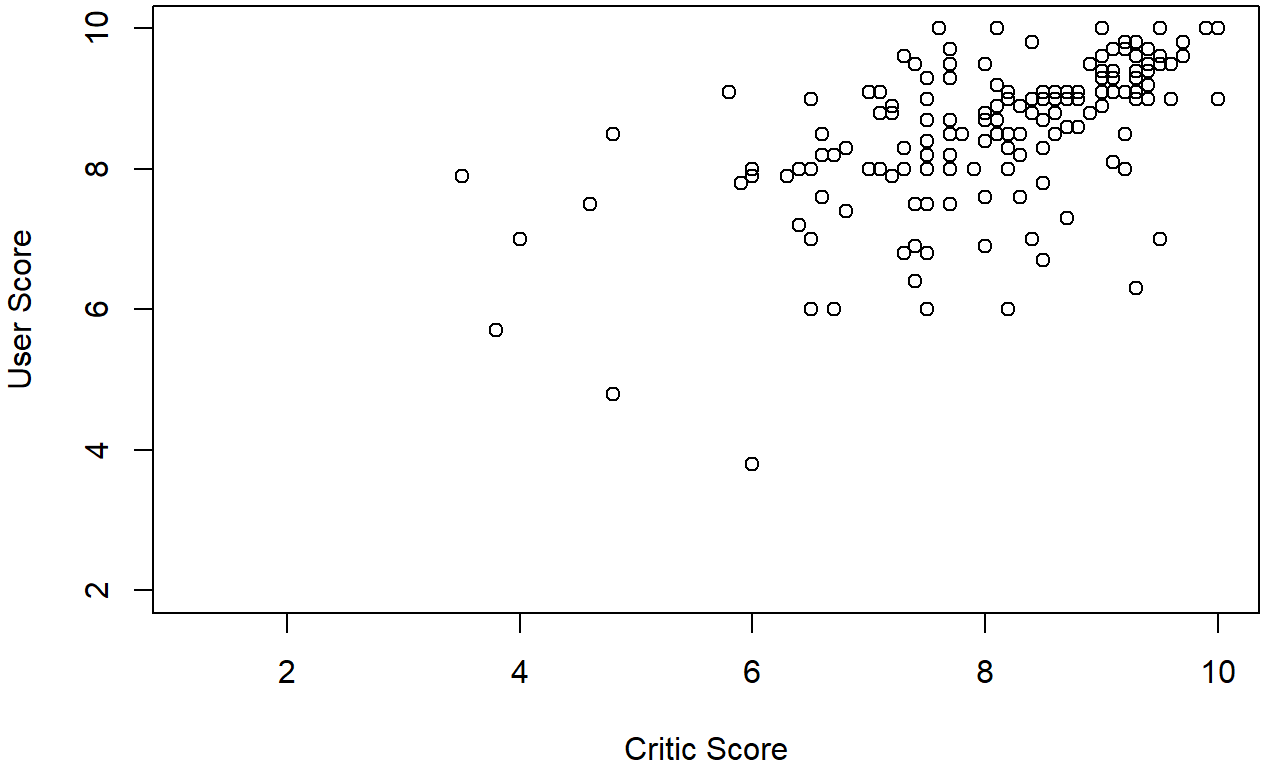
```
nrow(train[with(train, !is.na(train$User_Score)),])
```

```
## [1] 257
```

```
nrow(train[with(train, !is.na(train$Critic_Score) & !is.na(train$User_Score)),])
```

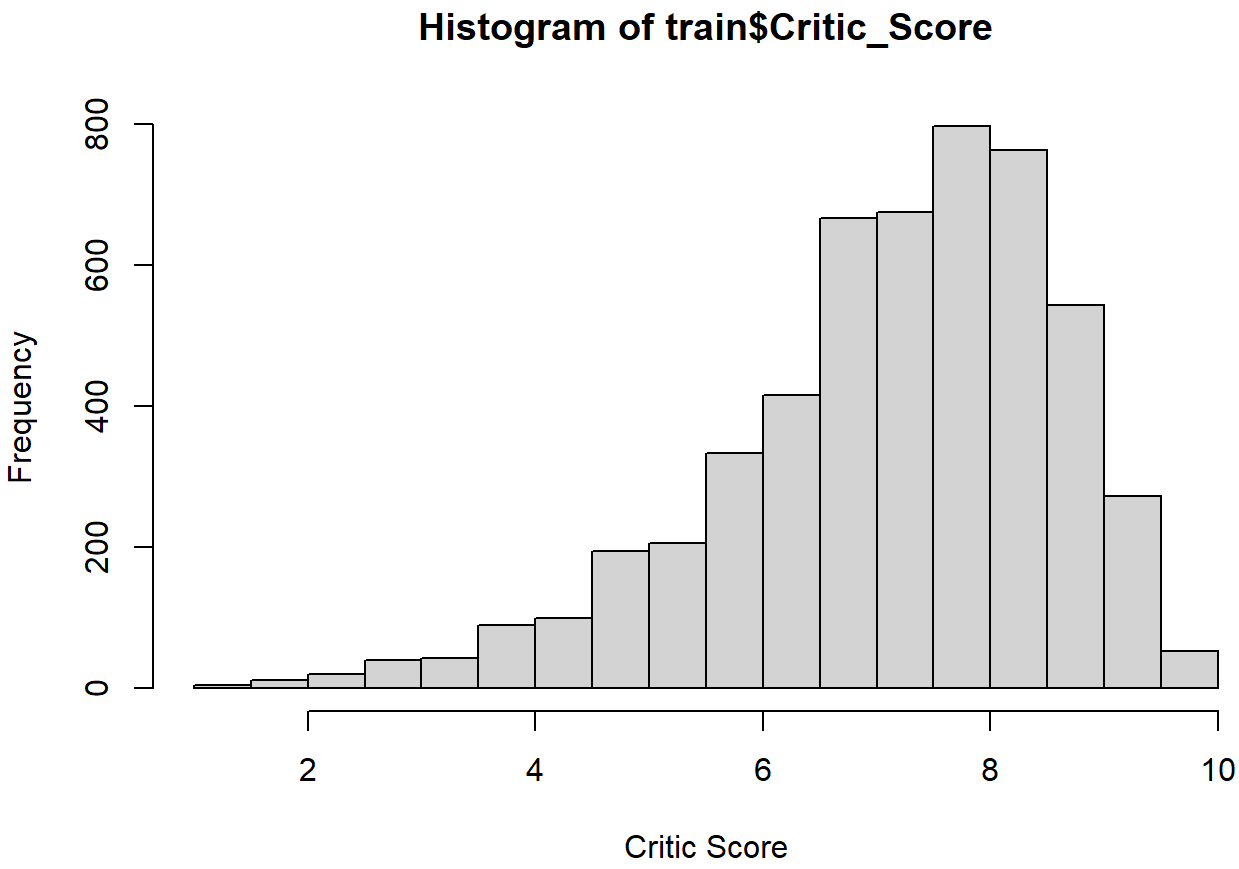
```
## [1] 168
```

```
plot(train$Critic_Score, train$User_Score, xlab="Critic Score", ylab="User Score")
```




```
hist(train$Total_Shipped, xlab = "Total Shipped")
```


```
hist(train$Critic_Score, xlab = "Critic Score")
```



Simple linear regression

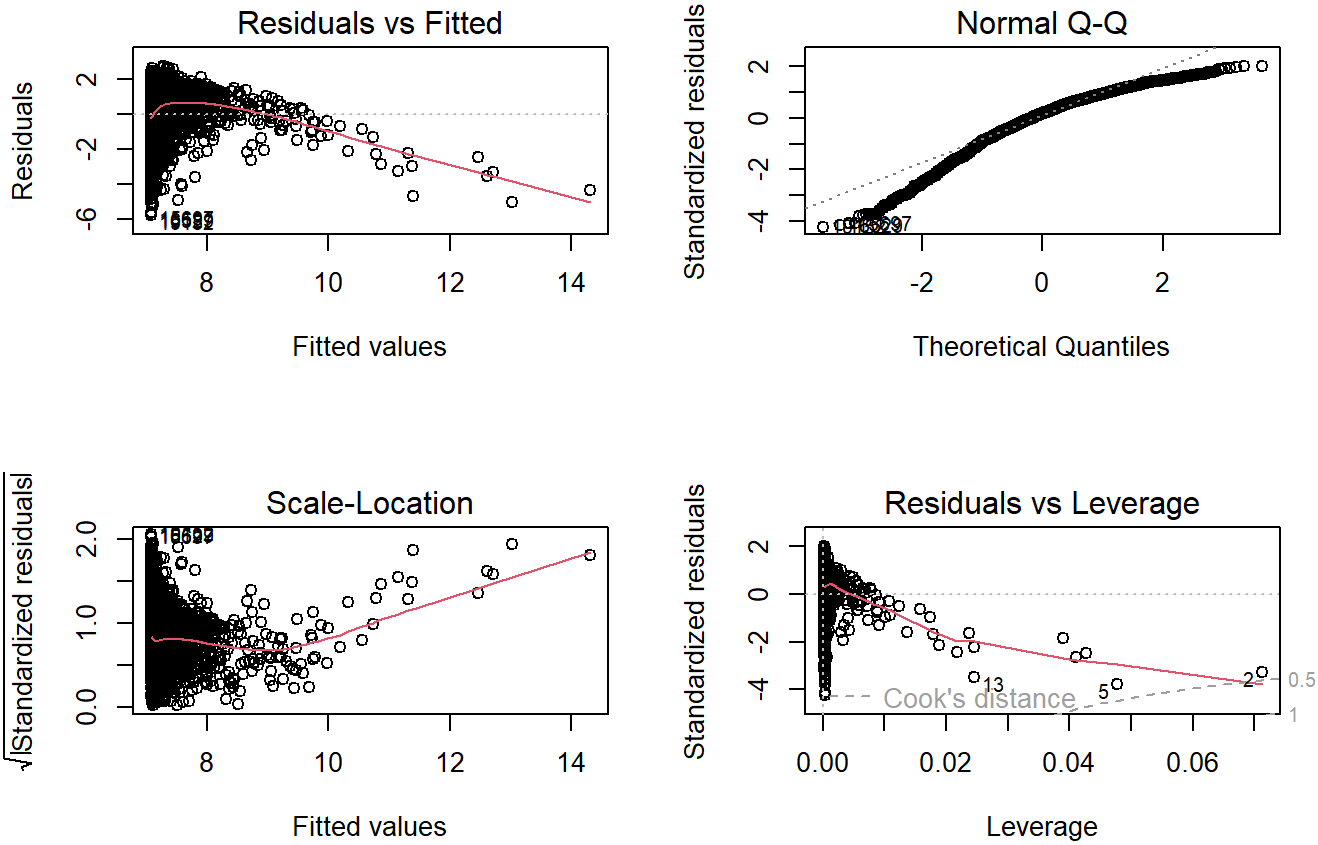
```
lm1 <- lm(Critic_Score~Total_Shipped, data=train)
summary(lm1)
```

```
##
## Call:
## lm(formula = Critic_Score ~ Total_Shipped, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7703 -0.6991  0.2342  0.9785  2.7316
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.068500   0.024092  293.39   <2e-16 ***
## Total_Shipped 0.180095   0.009257   19.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.36 on 3884 degrees of freedom
##   (1435 observations deleted due to missingness)
## Multiple R-squared:  0.0888, Adjusted R-squared:  0.08856
## F-statistic: 378.5 on 1 and 3884 DF,  p-value: < 2.2e-16
```

Under the coefficients section, the estimate of Total_Shipped is the coefficient m of x (y = mx + b), which indicated how one unit of Total_Shipped effects Critic_Score. Standard error is the average distance our observed values stay from the plot line. Pr(>|t|) is our p-value with respect to t, which is quite good being very close to zero. We have a lot of observations deleted due to missing data however, which will skew our data. The multiple and adjusted R-squared values are measures how well our model fit the data. Having a value <0.1 shows a poor fit between this model and our selected data.

Plotting the residuals

```
par(mfrow=c(2,2))
plot(lm1)
```



Our Residuals vs Fitted graph isn't good, as it doesn't show a very linear trend and has extremely high varriance for lower fitted values. This is an indication of a possible non-linear relationship and that this model is best. Normal Q-Q isn't too bad, just tends to deviate slightly on both ends of the graph. This shows that the residuals are fairly normally distributed. Our Scale-Location is also quite poor as we have much fewer data points for higher values and is skewing our data. Residuals vs Leverage showcases the influential cases, which we can see there are quite a few for the higher values once again.

Evaluate on the test set

Evaluate on test set, disregarding any NA elements in the data.

```
pred1 <- predict(lm1, newdata=test)
cor1 <- cor(pred1, test$Critic_Score, use="complete.obs")
mse1 <- mean((pred1-test$Critic_Score)^2, na.rm=TRUE)
rmse1 <- sqrt(mse1)

print(paste('correlation:', cor1))
```

```
## [1] "correlation: 0.247856887833476"
```

```
print(paste('mse:', mse1))
```

```
## [1] "mse: 1.96335770704899"
```

```
print(paste('rmse:', rmse1))
```

```
## [1] "rmse: 1.40119866794434"
```

The low correlation indicates that there isn't a strong connection between critic score and total shipped, or the total sales for a video game.

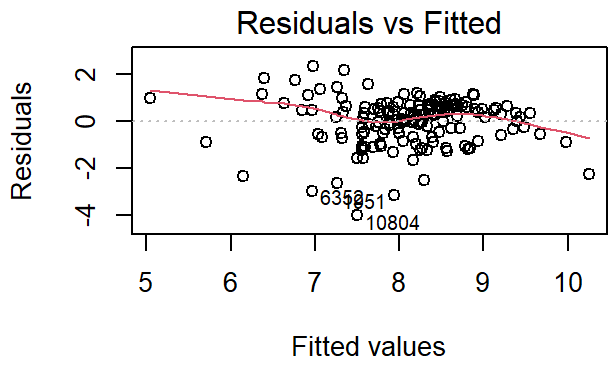
Multiple Linear Regression

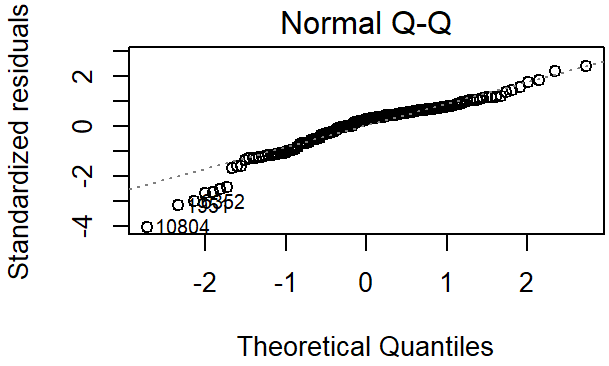
We'll try predicting using both Total_Shipped as well as User_Score, although our number of data points goes down drastically. If we were to want to most accurately compare lm1 and lm2, we would need to alter the data set used in lm1 to only include data with valid values for User_Score.

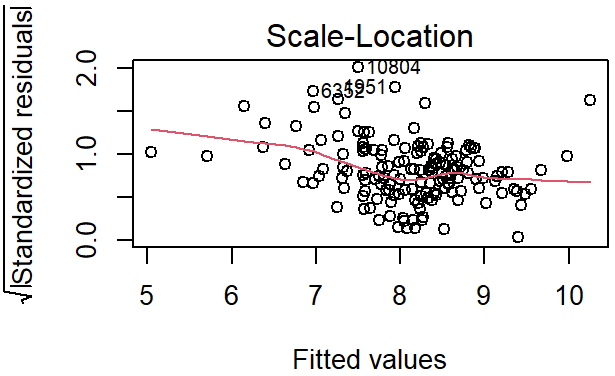
```
lm2 <- lm(Critic_Score~Total_Shipped+User_Score, data=train)
summary(lm2)
```

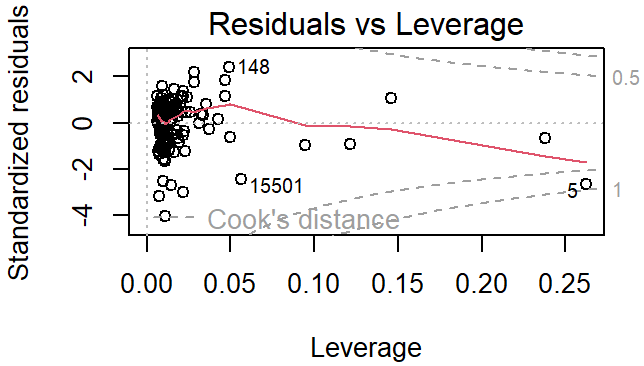
```
##
## Call:
## lm(formula = Critic_Score ~ Total_Shipped + User_Score, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0023 -0.5332  0.2749  0.6238  2.3270
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.66927    0.67389   3.961 0.000114 ***
## Total_Shipped  0.06682    0.01692   3.949 0.000119 ***
## User_Score     0.61059    0.07894   7.734 1.23e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9966 on 155 degrees of freedom
##   (5163 observations deleted due to missingness)
## Multiple R-squared:  0.3788, Adjusted R-squared:  0.3708
## F-statistic: 47.26 on 2 and 155 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(lm2)
```









Our p-values have remained very low and our R-squared values have increased dramatically, but still aren't good. This shows this model is a better fit than the previous, but still not great. Our amount of data has gone down drastically as many data points contain either user score OR critic score, so it's difficult to accurately compare lm1 with lm2. Overall our residuals have improved slightly, but still have some extreme outliers in them.

Evaluate on the test set (Multiple Linear Regression)

```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$Critic_Score, use="complete.obs")
mse2 <- mean((pred2-test$Critic_Score)^2, na.rm=TRUE)
rmse2 <- sqrt(mse2)

print(paste('correlation:', cor2))
```

```
## [1] "correlation: 0.530165992648165"
```

```
print(paste('mse:', mse2))
```

```
## [1] "mse: 0.786588286113511"
```

```
print(paste('rmse:', rmse2))
```

```
## [1] "rmse: 0.886898126119066"
```

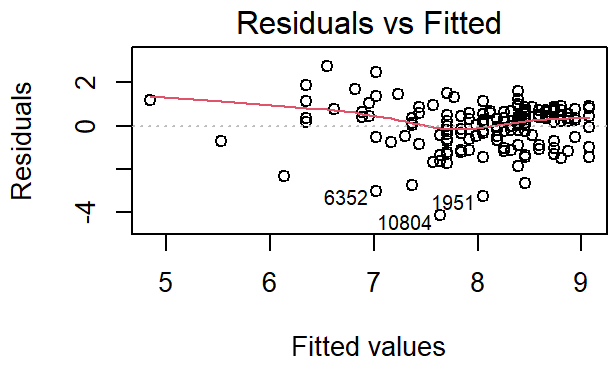
This shows that lm2 has a much better correlation and mse than lm1 and is overall a much better predictor of critic score.

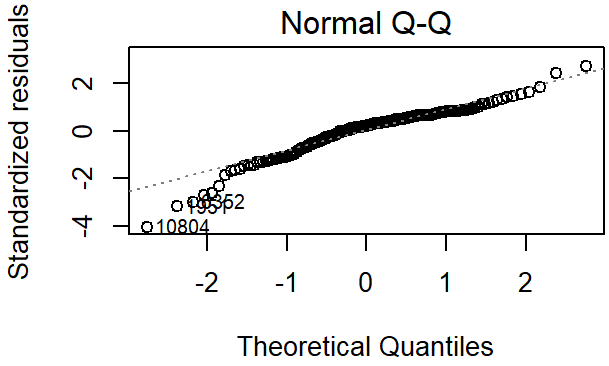
Third Linear Regression (Only User_Score)

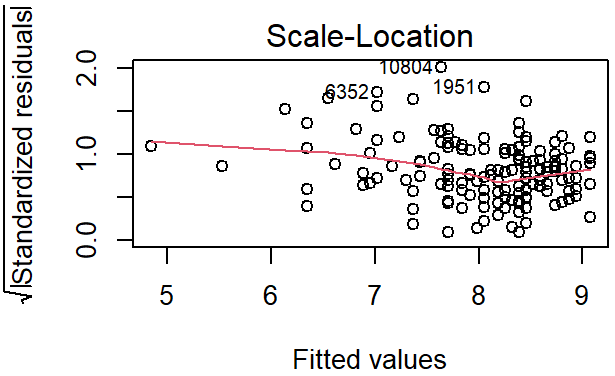
```
lm3 <- lm(Critic_Score~User_Score, data=train)
summary(lm3)
```

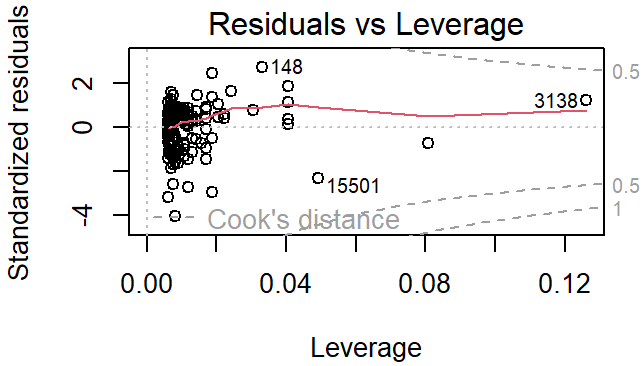
```
##
## Call:
## lm(formula = Critic_Score ~ User_Score, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1399 -0.5282  0.2196  0.6696  2.7531
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.24332    0.64513   3.477 0.000647 ***
## User_Score   0.68310    0.07485   9.126 2.33e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.026 on 166 degrees of freedom
##   (5153 observations deleted due to missingness)
## Multiple R-squared:  0.3341, Adjusted R-squared:  0.3301
## F-statistic: 83.29 on 1 and 166 DF,  p-value: 2.33e-16
```

```
par(mfrow=c(2,2))
plot(lm3)
```









Surprisingly, lm3 has a worse R squared value, despite comparing one score with another. We'll see later that the correlation is higher though, which shows that it might be a better predictor, but linear regression isn't a great model of choice for this data set though. These residuals are the best so far (but again not great), showing the slopes and trends that we want in each.

```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$Critic_Score, use="complete.obs")
mse3 <- mean((pred3-test$Critic_Score)^2, na.rm=TRUE)
rmse3 <- sqrt(mse3)

print(paste('correlation:', cor3))
```

```
## [1] "correlation: 0.614150612184448"
```

```
print(paste('mse:', mse3))
```

```
## [1] "mse: 0.650773275330372"
```

```
print(paste('rmse:', rmse3))
```

```
## [1] "rmse: 0.806705197287319"
```

This is the best correlation and mse so far, which shows that critic score and user score have the highest correlation and predicting power of the data that we tested. This makes sense as it's comparing two human scores which directly relate to what they thought about the quality of a game, rather than how many sold, which could have many additional factors, such as being included in a game system bundle (which is why we removed the best selling game, due to how much it skewed the data).