Project 1: Regression

Jordon Zeigler

Getting started

Here are the steps for getting started:

- Start with the assignment link that creates a repo on GitHub with starter documents. I have sent this to you through email.
- Clone this repo in RStudio
- Make any changes needed as outlined by the tasks you need to complete for the assignment
- Periodically commit changes (the more often the better, for example, once per each new task)
  - Remember, git will yell at you when you try to commit before running the following lines in the terminal
    - `git config --global user.name "Your Name Here"`
    - `git config --global user.email "Your Email Here"`
- Push all your changes back to your GitHub repo

and voila, you're done! Once you push your changes back you do not need to do anything else to "submit" your work. And you can of course push multiple times throughout the assignment. At the time of the deadline I will take whatever is in your repo and consider it your final submission, and grade the state of your work at that time (which means even if you made mistakes before then, you wouldn't be penalized for them as long as the final state of your work is correct).

Assignment Description

In this project you are going to use the skills that you've learned about regression on a dataset of your own. You may choose any dataset that you wish as long as it is not one that we've already discussed in the course. You may want to consult me about your choice of dataset, just to make sure it is suitable.

After making a suitable dataset choice, you need to complete the following steps:

- Narrative: You need to formulate a question in which you can address using your chosen techniques. This is the overall goal of your analysis.
- You need to perform proper pre-processing and cleaning of the data before your analysis begins. Depending on your data, this step may be fairly short or quite lengthy.
- You need to have a substantial exploratory data analysis (EDA) section. This section should include summaries, graphs (univariate, bivariate, and possibly multivariate), and other techniques from DS 1 to describe your data. You should also investigate possible interactions between variables. Your EDA should show a progression of understanding about the data and your research question.
- You need to choose at least two regression techniques (most likely a multiple linear regression model and a penalized regression method) to use in your analysis. You should explain your modeling choices and how they were informed by your EDA.
- You need to address the assumptions of each method with graphical and/or numeric evidence.
- You need to use cross-validation or a related method to compare the two or more methods.
- You need to come to your final answer using an iterative process that you show throughout your project.
- You need to discuss the shortcomings of your modeling approach. Also, if appropriate, you discuss improvements that could be made.
- You need to discuss how the model approach/output works toward answering the question.

- You need to discuss your major takeaways from the project. This part is meant to be a reflection on what you learned about the data and your increase in knowledge about data science during the process of the project.

Place Work Below

```
library("tidyverse");theme_set(theme_bw())
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1
```

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

```
library("tidymodels")
```

```
## Registered S3 method overwritten by 'tune':
##   method                   from
##   required_pkgs.model_spec parsnip
```

```
## -- Attaching packages --------------------------------------- tidymodels 0.1.4 --
```

```
## v broom        0.7.9      v rsample      0.1.0
## v dials        0.0.10     v tune         0.1.6
## v infer        1.0.0      v workflows    0.2.3
## v modeldata    0.1.1      v workflowsets 0.1.0
## v parsnip      0.1.7      v yardstick    0.0.8
## v recipes      0.1.17
```

```
## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
library("janitor")
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library("knitr")
library("caret")
```

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

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

```
## The following objects are masked from 'package:yardstick':
##
##     precision, recall, sensitivity, specificity

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

```
library("leaps")
library("olsrr")
```

```
##
## Attaching package: 'olsrr'

## The following object is masked from 'package:datasets':
##
##     rivers
```

```
library("glmnet")
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-2
```

```
library("fastDummies")
library(Metrics)
```

```
##
## Attaching package: 'Metrics'

## The following objects are masked from 'package:caret':
##
##     precision, recall

## The following objects are masked from 'package:yardstick':
##
##     accuracy, mae, mape, mase, precision, recall, rmse, smape
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:olsrr':
##
##     cement

## The following object is masked from 'package:dplyr':
##
##     select
```

```
Game_Sales <- read_csv("Video_Games_Sales_as_at_22_Dec_2016.csv")
```

```
## Rows: 16719 Columns: 16
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): Name, Platform, Year_of_Release, Genre, Publisher, Developer, Rating
## dbl (9): NA_Sales, EU_Sales, JP_Sales, Other_Sales, Global_Sales, Critic_Sco...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The main question for this project will be what variables seem to correlate the most with the number of copies that a video game has sold globally, excluding a game's name or title and of course the NA, EU, JP and Other sales as they all effectively add up to the global sales amount which will be considered the main response variable of this analysis. User_Count and Critic_Count will also not be considered as possible predictors for this analysis, the User_Score and Critic_Score will be considered however.

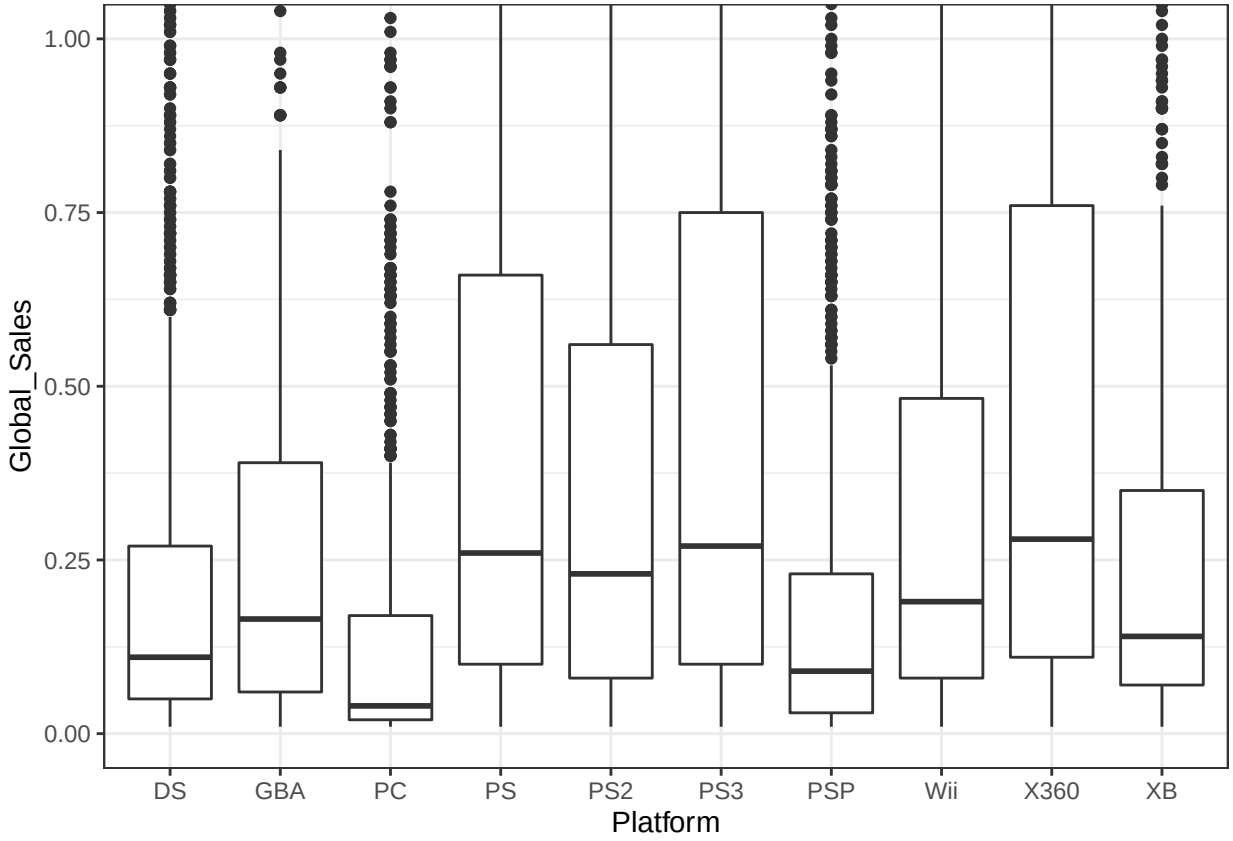
Exploratory Data Analysis section

```
summary(Game_Sales)
```

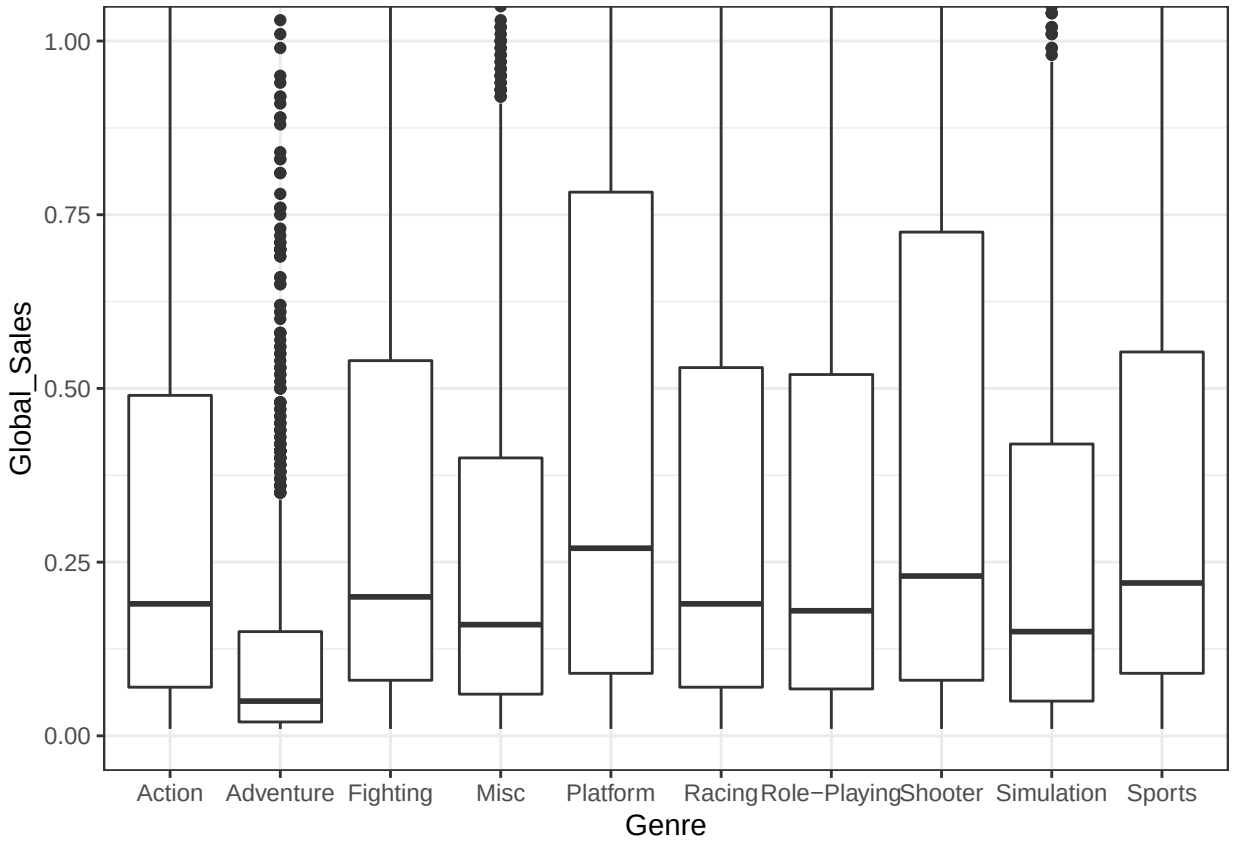
```
##      Name              Platform         Year_of_Release       Genre
##  Length:16719       Length:16719       Length:16719       Length:16719
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   Publisher            NA_Sales          EU_Sales         JP_Sales
##  Length:16719       Min.   : 0.0000   Min.   : 0.000   Min.   : 0.0000
##  Class :character   1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.: 0.0000
##  Mode  :character   Median : 0.0800   Median : 0.020   Median : 0.0000
##                     Mean   : 0.2633   Mean   : 0.145   Mean   : 0.0776
##                     3rd Qu.: 0.2400   3rd Qu.: 0.110   3rd Qu.: 0.0400
##                     Max.   :41.3600   Max.   :28.960   Max.   :10.2200
##
##   Other_Sales        Global_Sales      Critic_Score    Critic_Count
##  Min.   : 0.00000   Min.   : 0.0100   Min.   :13.00   Min.   :  3.00
##  1st Qu.: 0.00000   1st Qu.: 0.0600   1st Qu.:60.00   1st Qu.: 12.00
##  Median : 0.01000   Median : 0.1700   Median :71.00   Median : 21.00
##  Mean   : 0.04733   Mean   : 0.5335   Mean   :68.97   Mean   : 26.36
##  3rd Qu.: 0.03000   3rd Qu.: 0.4700   3rd Qu.:79.00   3rd Qu.: 36.00
##  Max.   :10.57000   Max.   :82.5300   Max.   :98.00   Max.   :113.00
##                                       NA's   :8582    NA's   :8582
##    User_Score      User_Count       Developer           Rating
##  Min.   :0.000   Min.   :    4.0   Length:16719       Length:16719
##  1st Qu.:6.400   1st Qu.:   10.0   Class :character   Class :character
##  Median :7.500   Median :   24.0   Mode  :character   Mode  :character
##  Mean   :7.125   Mean   :  162.2
##  3rd Qu.:8.200   3rd Qu.:   81.0
##  Max.   :9.700   Max.   :10665.0
##  NA's   :9129    NA's   :9129
```

```
# produces box plot involving the 10 most common platforms that games were released on and global sales
platform_graph_variables <- Game_Sales %>% group_by(Platform) %>% filter(n() > 5)%>%  summarize(Count =

platform_graph_base <- Game_Sales %>%
    filter(Platform %in% platform_graph_variables$Platform)
```

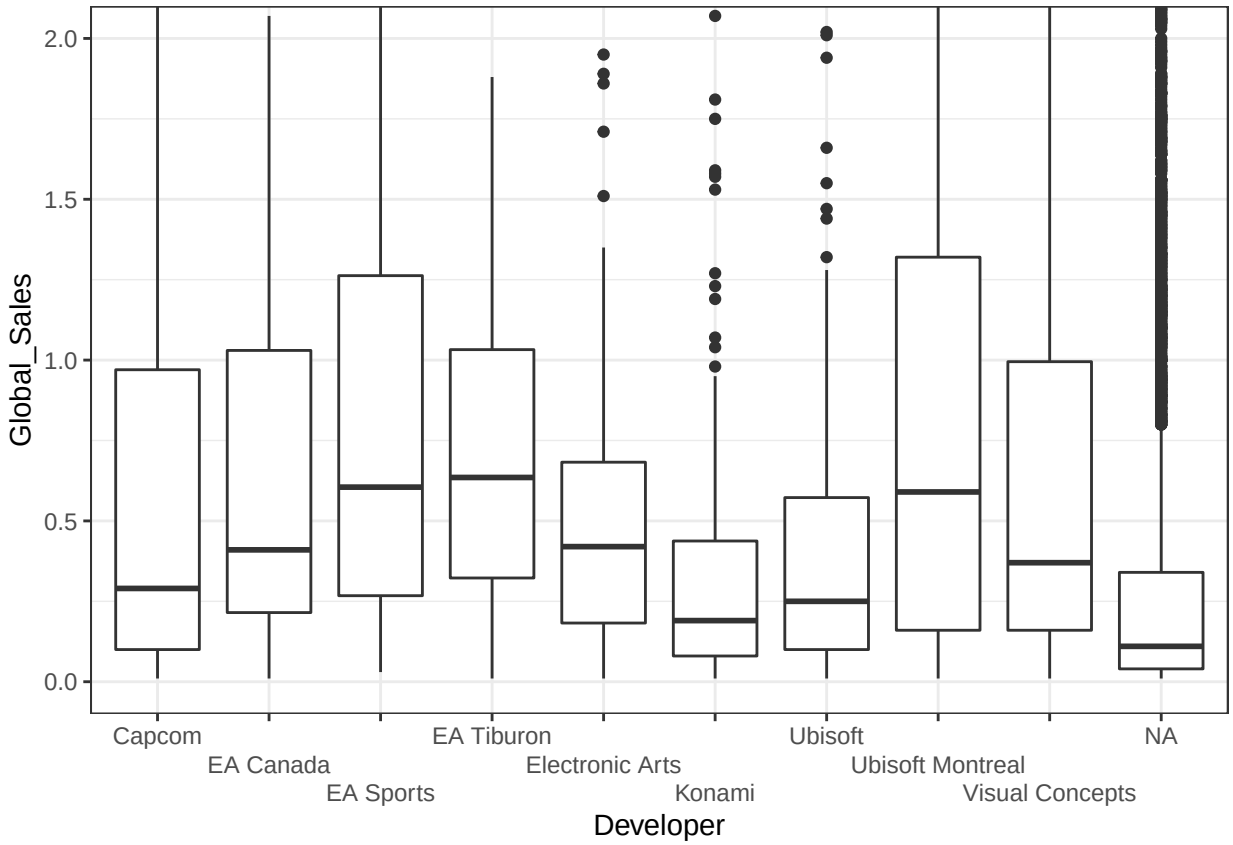
```
ggplot(platform_graph_base, aes(x = Platform, y = Global_Sales)) +
    geom_boxplot() + coord_cartesian(ylim = c(0, 1))
```



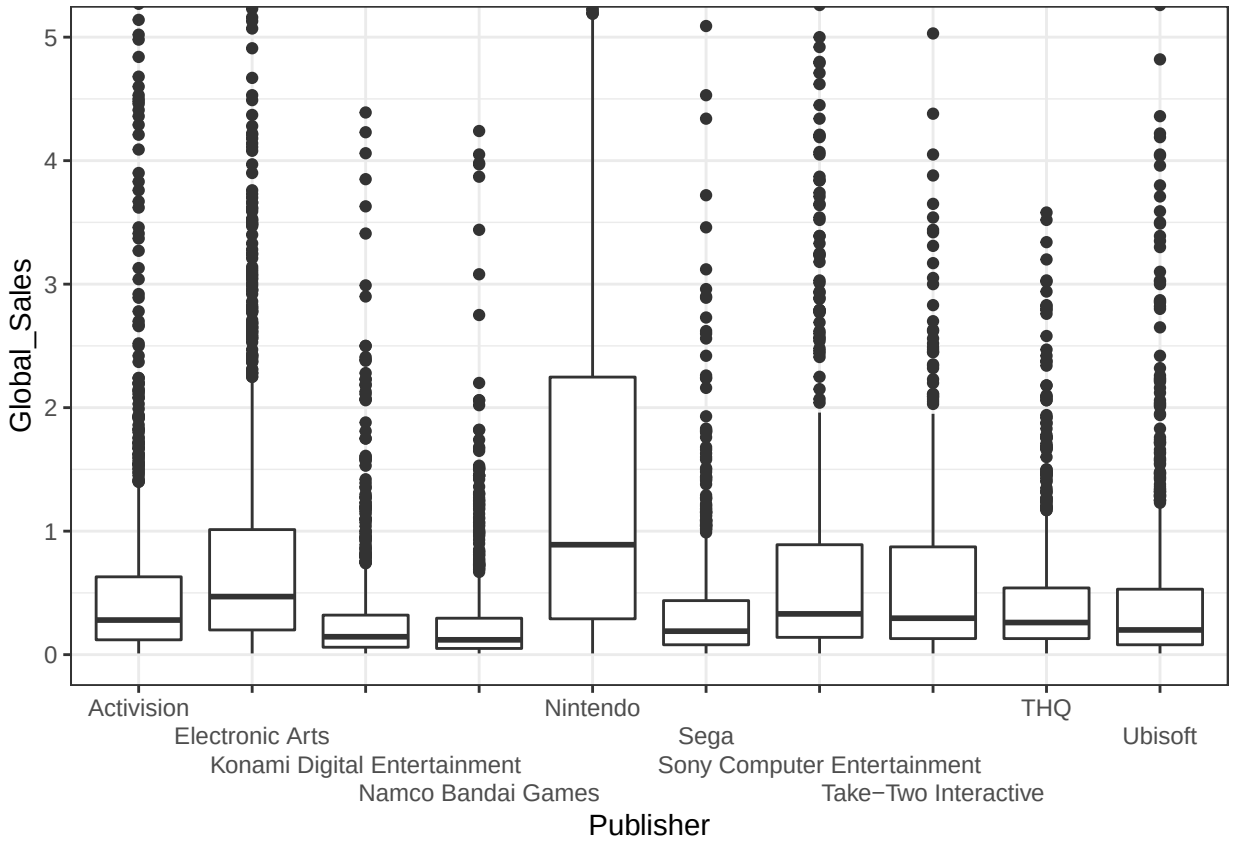

```
# will produce a boxplot involving the genres (categories) of each game and the global sales number for
genre_graph_variables <- Game_Sales %>% group_by(Genre) %>% filter(n() > 5)%>%  summarize(Count = n()) %

genre_graph_base <- Game_Sales %>%
    filter(Genre %in% genre_graph_variables$Genre)

ggplot(genre_graph_base, aes(x = Genre, y = Global_Sales)) +
    geom_boxplot() + coord_cartesian(ylim = c(0, 1))
```



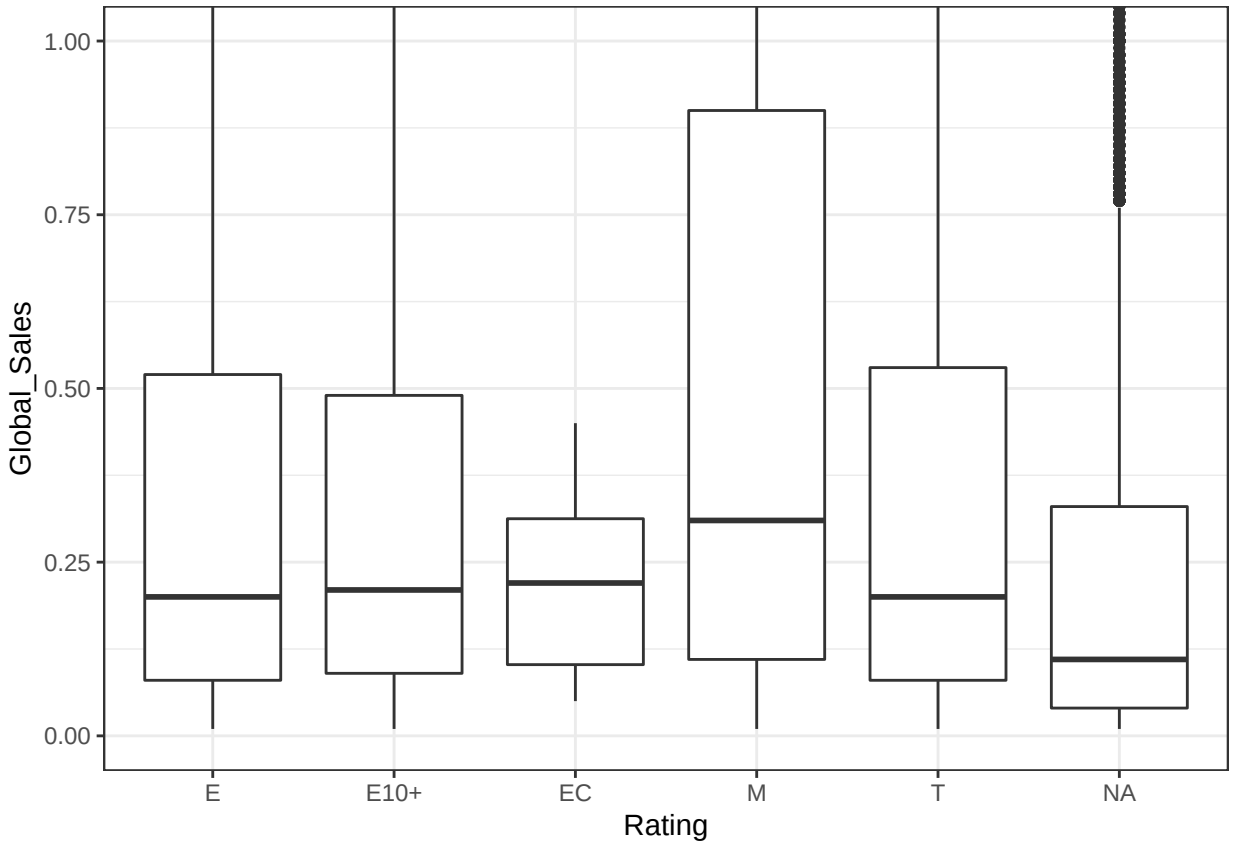

```
# will produce a boxplot involving the developers of each game and the global sales number for each gam
developer_graph_variables <- Game_Sales %>% group_by(Developer) %>% filter(n() > 5)%>%  summarize(Count

developer_graph_base <- Game_Sales %>%
    filter(Developer %in% developer_graph_variables$Developer)

ggplot(developer_graph_base, aes(x = Developer, y = Global_Sales)) +
    geom_boxplot() + scale_x_discrete(guide = guide_axis(n.dodge = 3)) + coord_cartesian(ylim = c(0, 2)
```



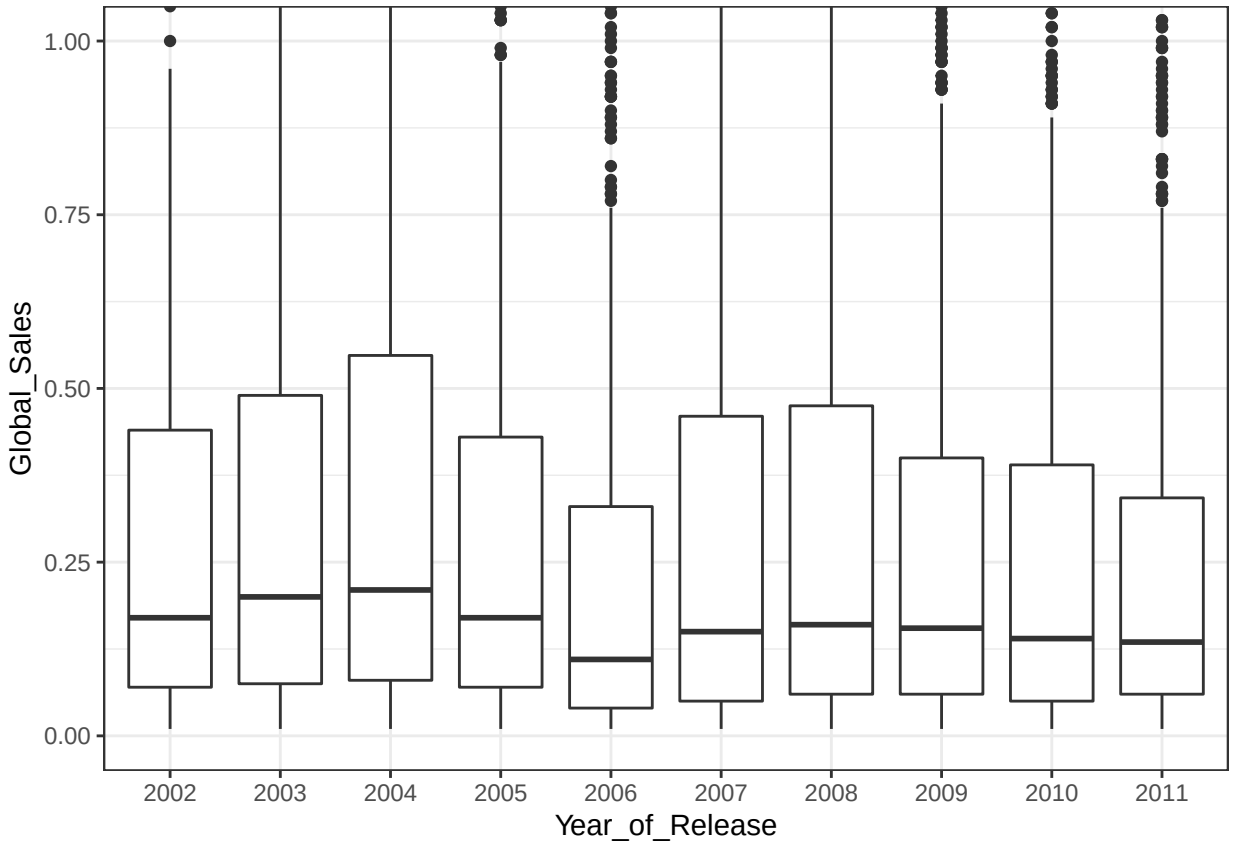

```
# will produce a boxplot involving the publishers of each game and the global sales number for each gam
publisher_graph_variables <- Game_Sales %>% group_by(Publisher) %>% filter(n() > 5)%>%  summarize(Count

publisher_graph_base <- Game_Sales %>%
    filter(Publisher %in% publisher_graph_variables$Publisher)

ggplot(publisher_graph_base, aes(x = Publisher, y = Global_Sales)) +
    geom_boxplot() + scale_x_discrete(guide = guide_axis(n.dodge = 4)) + coord_cartesian(ylim = c(0, 5))
```




```
# will produce a box plot involving the relation between a game's age rating and it's global sales
rating_graph_variables <- Game_Sales %>% group_by(Rating) %>% filter(n() > 5)%>%  summarize(Count = n())

rating_graph_base <- Game_Sales %>%
    filter(Rating %in% rating_graph_variables$Rating)

ggplot(rating_graph_base, aes(x = Rating, y = Global_Sales)) +
    geom_boxplot() + coord_cartesian(ylim = c(0, 1))
```

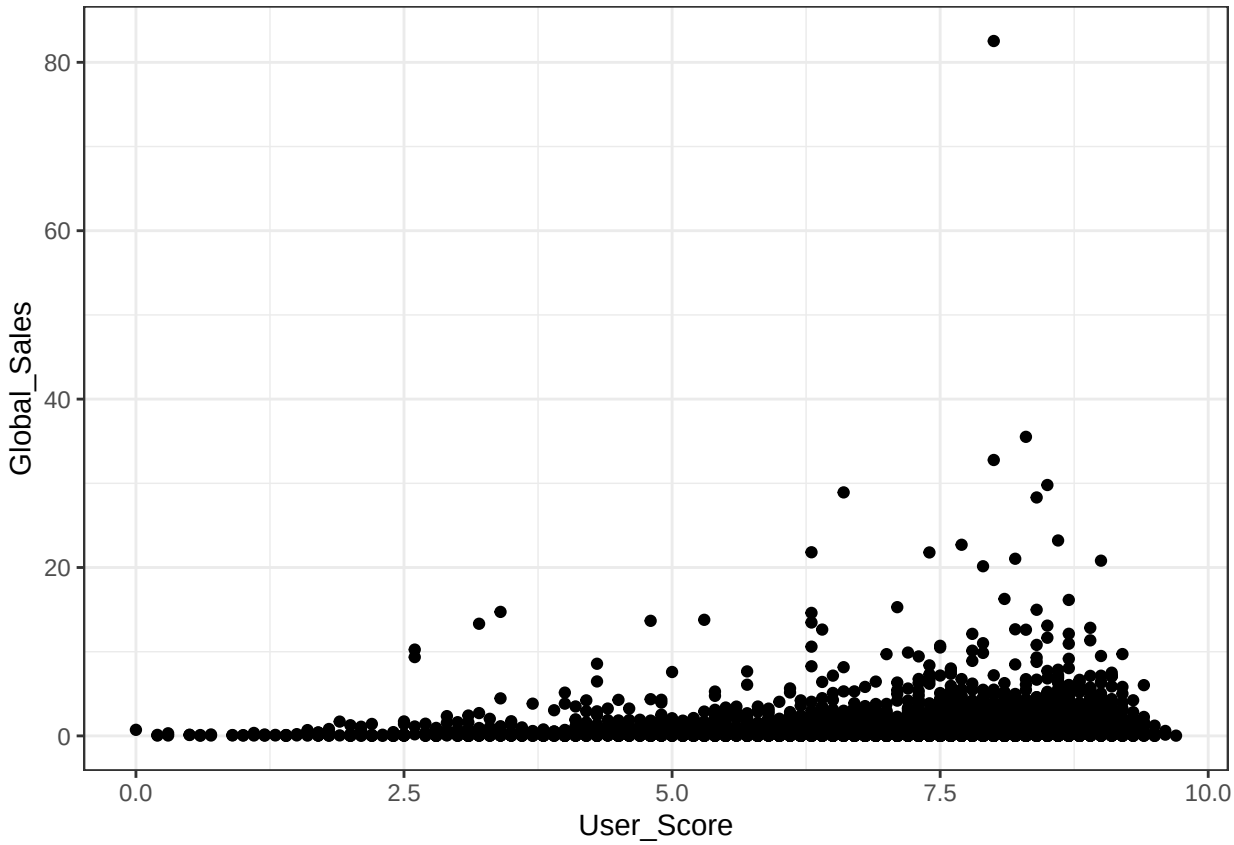



```
# will produce a box plot involving the relation between a game's year of release and it's global sales
year_graph_variables <- Game_Sales %>% group_by(Year_of_Release) %>% filter(n() > 5)%>%  summarize(Coun

year_graph_base <- Game_Sales %>%
    filter(Year_of_Release %in% year_graph_variables$Year_of_Release)

ggplot(year_graph_base, aes(x = Year_of_Release, y = Global_Sales)) +
    geom_boxplot() + coord_cartesian(ylim = c(0, 1))
```



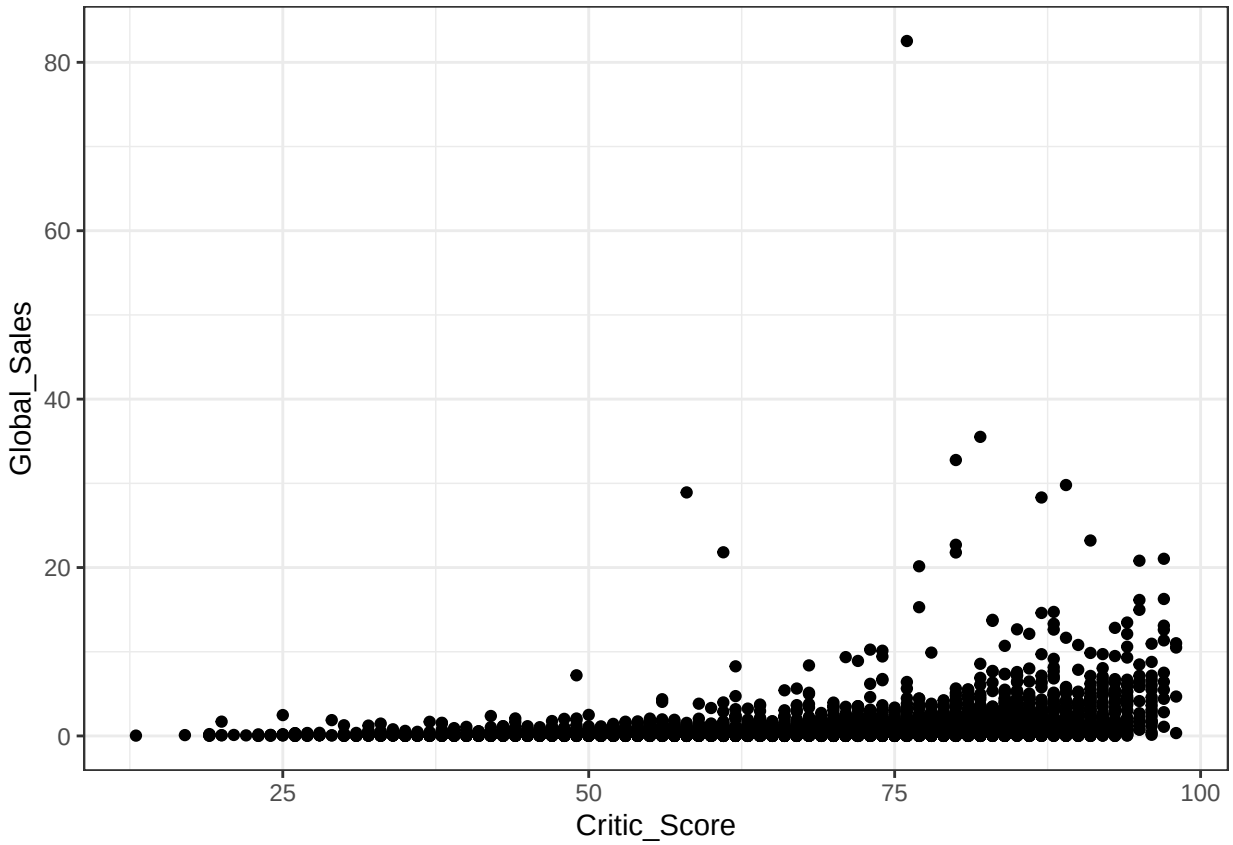

```
# plot relating user scores of games to their global sales
ggplot(Game_Sales, aes(x = User_Score, y = Global_Sales)) + geom_point()
```

```
## Warning: Removed 9129 rows containing missing values (geom_point).
```




```
# plot relating critic scores of games to their global sales
ggplot(Game_Sales, aes(x = Critic_Score, y = Global_Sales)) + geom_point()
```

```
## Warning: Removed 8582 rows containing missing values (geom_point).
```



The graphs produced in this section have shown a few trends for each of the categorical variables and their correlation to the global sales variable. For almost all of the boxplots involving the categorical variables have several outlier values. The platform graph indicates that the ps2, ps, ps3 and xbox 360 have a very wide interquarile range (wide variety of values found in the middle 50% of values) compared to the other platforms, especially the PC. The genre graph indicates that each of the 10 most common genres have fairly similar interquartile ranges, except for the adventure and miscellaneous genre. The adventure genre especially features many of its game's sales ranging between 0.02 million and 0.12 million, although there are many outlier values present. The developer graph indicates that Ubisoft Montreal and EA sports have the widest interquartile range and have the highest possible values for said interquartile range as well. The publisher graph indicates that out of the 10 most prominent publishers, Nintendo has the highest interquartile range by a wide amount and also have a larger third quartile, meaning that the middle 50% of the values have a wide range of values and tend to appear on the high end. The rating graph indicated that the mature rating category had the largest interquartile range with a massive third quartile implying that the sales of most mature rated games are generally higher than other rated games. The year of release for a game doesn't appear to have any significant differences, at least amongst the top 10 years chosen due to their presence in the data. As for the numerical variables, critic score and user score, both have mildly similar point distributions with both having a very mild positive correlation with the score variables, although not by much and there are quite a few outliers present.

Linear Regression Model the platform, genre, developer, publisher, user score, rating and critic score variables will be used as predictors with global sales being the response variable. The year of release variable will be excluded as it has been shown to not be very relevant as seen from the EDA section. The developer and publisher variables will likely be excluded since while the EDA has shown that while these variables appear to have some influence on the range of values, trying to incorporate either of them into the model and attempting to perform subset regression with either of them is too resource intensive for rstudio cloud.

```
split_data <- split(Game_Sales, sample(1:nrow(Game_Sales) > round(nrow(Game_Sales) * .1)))
Training_Game_Sales <- split_data$`TRUE`
```

```r
Test_Game_Sales <- split_data$`FALSE`
```

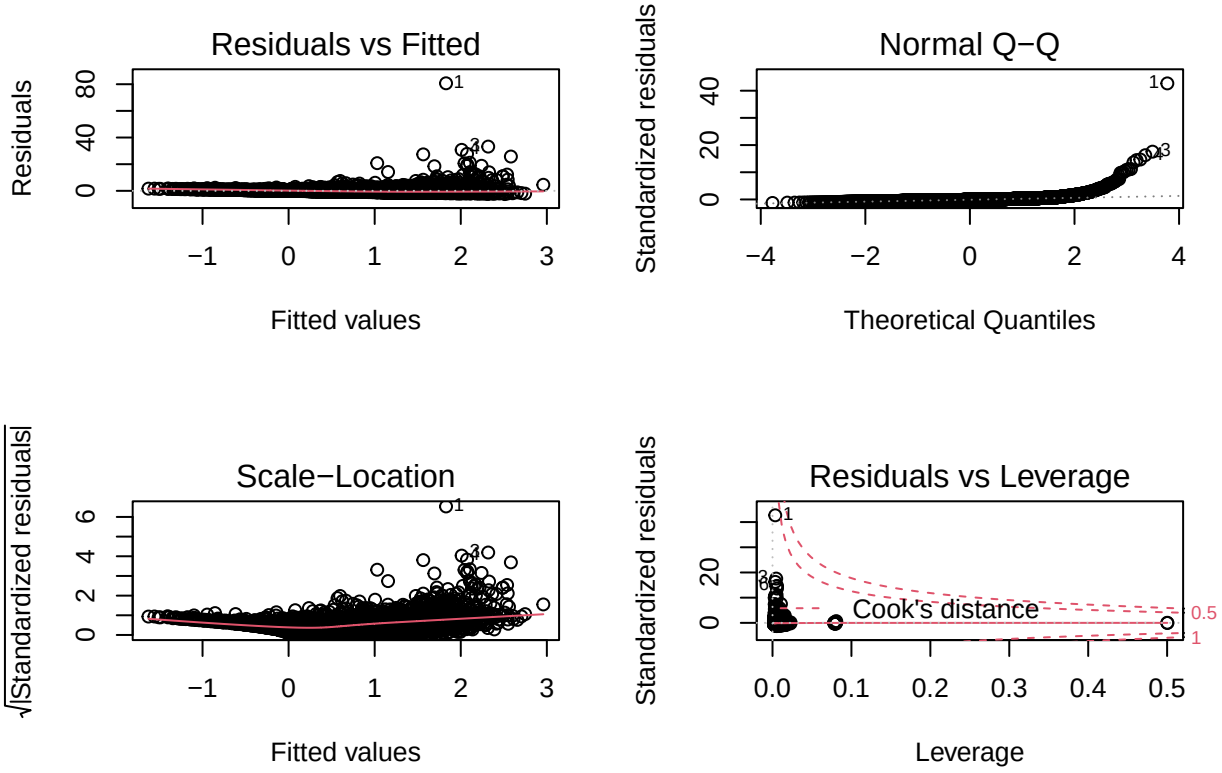
```r
sales_predictor_model <- lm(Global_Sales ~ Platform + Rating + Genre + User_Score + Critic_Score, data =
summary(sales_predictor_model)
```

```
##
## Call:
## lm(formula = Global_Sales ~ Platform + Rating + Genre + User_Score +
##     Critic_Score, data = Training_Game_Sales)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.422 -0.655 -0.250  0.240 80.701
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.2455372  0.2148145  -5.798 7.03e-09 ***
## PlatformDC        -1.1327879  0.5505365  -2.058 0.039669 *
## PlatformDS         0.1539464  0.1815111   0.848 0.396394
## PlatformGBA       -0.4180338  0.2039581  -2.050 0.040445 *
## PlatformGC        -0.4964390  0.1911599  -2.597 0.009427 **
## PlatformPC        -0.9386573  0.1795567  -5.228 1.77e-07 ***
## PlatformPS         0.3785394  0.2273865   1.665 0.096015 .
## PlatformPS2       -0.0269562  0.1698335  -0.159 0.873894
## PlatformPS3       -0.0372058  0.1744186  -0.213 0.831090
## PlatformPS4       -0.0725278  0.2049020  -0.354 0.723378
## PlatformPSP       -0.3157565  0.1872941  -1.686 0.091868 .
## PlatformPSV       -0.6036497  0.2410007  -2.505 0.012279 *
## PlatformWii        0.7430466  0.1811905   4.101 4.17e-05 ***
## PlatformWiiU      -0.2084987  0.2646639  -0.788 0.430852
## PlatformX360       0.0390160  0.1729788   0.226 0.821556
## PlatformXB        -0.6118888  0.1810152  -3.380 0.000728 ***
## PlatformXOne      -0.4334563  0.2262444  -1.916 0.055427 .
## RatingE10+        -0.4311801  0.0861167  -5.007 5.68e-07 ***
## RatingK-A         -0.7346804  1.9044707  -0.386 0.699683
## RatingM            0.0275030  0.0941041   0.292 0.770097
## RatingRP           0.3734190  1.3475899   0.277 0.781711
## RatingT           -0.2815829  0.0761217  -3.699 0.000218 ***
## GenreAdventure    -0.3652032  0.1382088  -2.642 0.008253 **
## GenreFighting     -0.1418622  0.1159487  -1.223 0.221191
## GenreMisc          0.1634945  0.1153520   1.417 0.156430
## GenrePlatform      0.0642568  0.1175089   0.547 0.584518
## GenrePuzzle       -0.4633558  0.2006355  -2.309 0.020952 *
## GenreRacing        0.0167030  0.1035342   0.161 0.871840
## GenreRole-Playing -0.1251654  0.0909697  -1.376 0.168901
## GenreShooter       0.0979809  0.0864735   1.133 0.257226
## GenreSimulation    0.0004406  0.1284279   0.003 0.997263
## GenreSports       -0.2339597  0.0980550  -2.386 0.017062 *
## GenreStrategy     -0.3146020  0.1366246  -2.303 0.021330 *
## User_Score        -0.1106616  0.0221066  -5.006 5.72e-07 ***
## Critic_Score       0.0454030  0.0022811  19.904  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.894 on 6230 degrees of freedom
##   (8782 observations deleted due to missingness)
## Multiple R-squared:  0.1099, Adjusted R-squared:  0.105
## F-statistic: 22.61 on 34 and 6230 DF,  p-value: < 2.2e-16
```

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

```
## Warning: not plotting observations with leverage one:
##   550
```

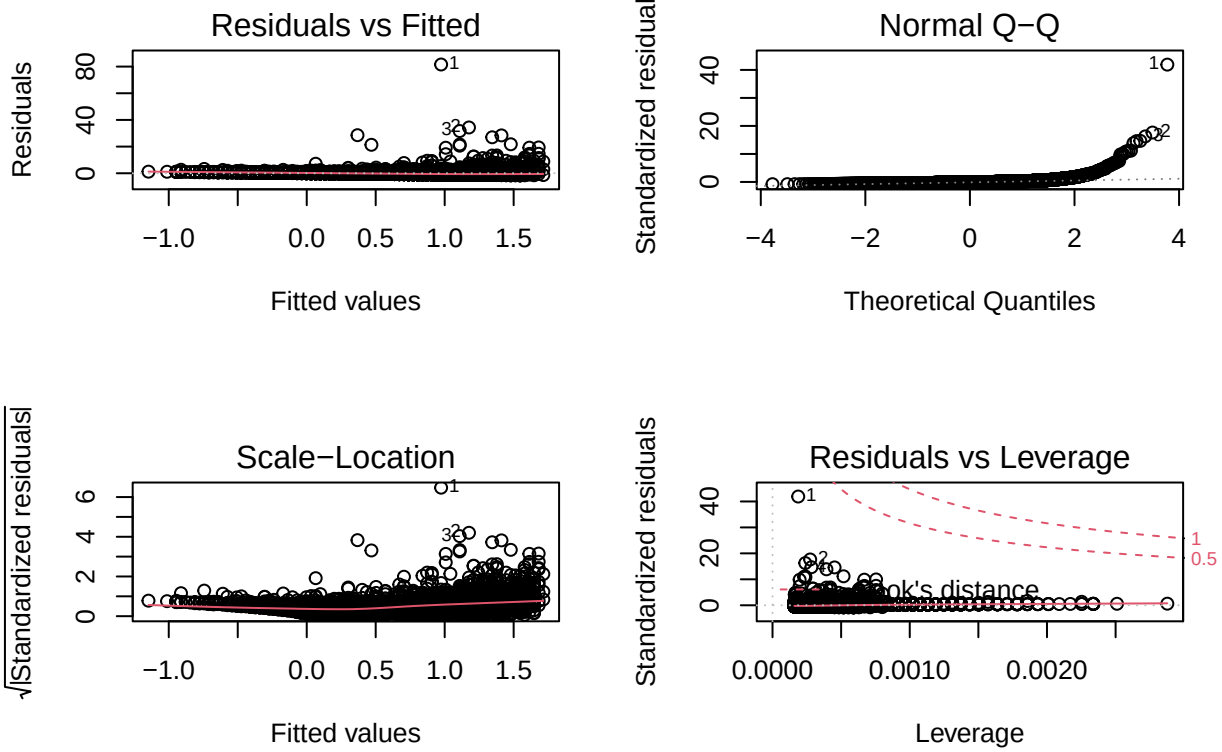



```
best_predictor_linear <- ols_step_best_subset(sales_predictor_model)
best_predictor_linear
```

```
##                   Best Subsets Regression
## ---------------------------------------------------------------
## Model Index    Predictors
## ---------------------------------------------------------------
##      1         Critic_Score
##      2         Platform Critic_Score
##      3         Platform Rating Critic_Score
##      4         Platform Rating Genre Critic_Score
##      5         Platform Rating Genre User_Score Critic_Score
## ---------------------------------------------------------------
##
##                                                    Subsets Regression Summary
## ------------------------------------------------------------------------------------------------------------
##                        Adj.        Pred
## Model    R-Square    R-Square    R-Square      C(p)          AIC          SBIC          SBC
## ------------------------------------------------------------------------------------------------------------
##   1        0.0582      0.0581      0.0576    -618.1562    29508.1981    8715.4267    29528.8960    24
```

```
##   2        0.0952      0.0931      0.0912    -879.8910    29246.1693    8423.8314    29377.2564    23
##   3        0.1020      0.0992        -Inf    -859.8024    28975.6925    8352.7824    29141.0359    23
##   4        0.1069      0.1028        -Inf    -892.9490    28957.6597    8315.0807    29198.7855    23
##   5        0.1099      0.1050        -Inf     -23.0000    25818.8828    7981.6504    26061.6213    23
## ------------------------------------------------------------------------------------------------------
## AIC: Akaike Information Criteria
##  SBIC: Sawa's Bayesian Information Criteria
##  SBC: Schwarz Bayesian Criteria
##  MSEP: Estimated error of prediction, assuming multivariate normality
##  FPE: Final Prediction Error
##  HSP: Hocking's Sp
##  APC: Amemiya Prediction Criteria
```

```
final_sales_predictor_model_lin <- lm(Global_Sales ~ Critic_Score, data = na.omit(Training_Game_Sales))
par(mfrow =c(2,2))
plot(final_sales_predictor_model_lin)
```




```
Test_Game_Sales <- na.omit(Test_Game_Sales)


rmse(Test_Game_Sales$Global_Sales, predict(final_sales_predictor_model_lin, Test_Game_Sales))
```

```
## [1] 1.301678
```

This section has determined that the best variable or at least the variable with the most influence on the global sales variable is the critic score variable and a new model has been formed using only that as a predictor for comparisons to the penalized regression later. concerning the assumptions of linear regression (focusing only on the model that only uses the critic score as a predictor) the model satisfies the assumption of linearity as the residuals vs fitted graph indicates a graph that is fairly close to 0 consistently. The assumption of homogenity of variance isn't valid as the scale location graph doesn't feature very evenly distributed points. The normality of Residuals assumption isn't valid as the Normal Q-Q plot doesn't feature a straight line

for its points. Finally the assumption of high leverage is also not valid as a fair number points exceed a leverage value 0.002 implying that those points have too much influence on the model. Overall this model is not necessarily the most suitable for this dataset.

Penalized Regression Model will include the same predictors that were originally chosen for the linear regression model, platform, genre, rating, critic score and user score

```
Training_Game_Sales <- na.omit(Training_Game_Sales)
Game_Sales_recipe <- recipe(Global_Sales ~ User_Score + Critic_Score + Platform + Genre + Rating, data =
Game_Sales_recipe <- Game_Sales_recipe %>% step_center(all_numeric_predictors()) %>% step_scale(all_num

folds <- vfold_cv(Training_Game_Sales, v = 10)

Game_Sales_regression_model <- linear_reg(penalty = tune(), mixture = 1) %>% set_engine("glmnet")

Game_Sales_workflow <- workflow() %>% add_recipe(Game_Sales_recipe) %>% add_model(Game_Sales_regression_

tuning_grid <- grid_regular(penalty(), levels = 50)

tuning_grid <- tune_grid(Game_Sales_workflow, resamples = folds, grid = tuning_grid)
```

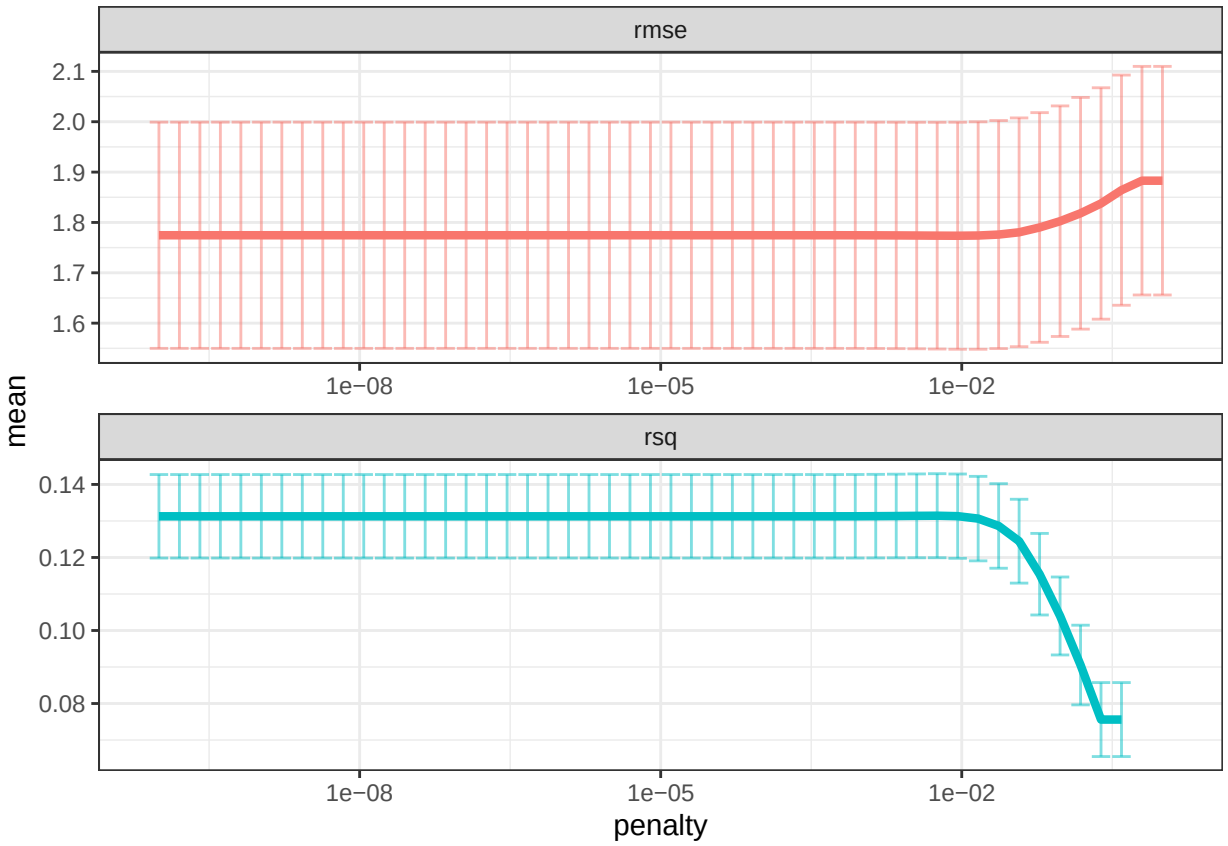
```
## ! Fold01: internal: A correlation computation is required, but `estimate` is const...
## ! Fold02: internal: A correlation computation is required, but `estimate` is const...
## ! Fold03: internal: A correlation computation is required, but `estimate` is const...
## ! Fold04: internal: A correlation computation is required, but `estimate` is const...
## ! Fold05: internal: A correlation computation is required, but `estimate` is const...
## ! Fold06: internal: A correlation computation is required, but `estimate` is const...
## ! Fold07: internal: A correlation computation is required, but `estimate` is const...
## ! Fold08: internal: A correlation computation is required, but `estimate` is const...
## ! Fold09: internal: A correlation computation is required, but `estimate` is const...
## ! Fold10: preprocessor 1/1, model 1/1 (predictions): There are new levels in a fac...
## ! Fold10: internal: A correlation computation is required, but `estimate` is const...
```

```
tuning_grid %>% collect_metrics() %>% filter(.metric == "rmse") %>% arrange(mean)
```

```
## # A tibble: 50 x 7
##     penalty .metric .estimator  mean     n std_err .config
##       <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
##  1 9.10e- 3 rmse    standard    1.77    10   0.225 Preprocessor1_Model40
##  2 5.69e- 3 rmse    standard    1.77    10   0.225 Preprocessor1_Model39
##  3 1.46e- 2 rmse    standard    1.77    10   0.226 Preprocessor1_Model41
##  4 3.56e- 3 rmse    standard    1.77    10   0.225 Preprocessor1_Model38
##  5 2.22e- 3 rmse    standard    1.77    10   0.225 Preprocessor1_Model37
##  6 1.39e- 3 rmse    standard    1.77    10   0.225 Preprocessor1_Model36
##  7 8.69e- 4 rmse    standard    1.77    10   0.225 Preprocessor1_Model35
##  8 1   e-10 rmse    standard    1.77    10   0.225 Preprocessor1_Model01
##  9 1.60e-10 rmse    standard    1.77    10   0.225 Preprocessor1_Model02
## 10 2.56e-10 rmse    standard    1.77    10   0.225 Preprocessor1_Model03
## # ... with 40 more rows
```

```
Game_Sales_regression_model2 <- linear_reg(penalty = 0.001389, mixture = 1) %>% set_engine("glmnet")

Game_Sales_workflow2 <- workflow() %>% add_recipe(Game_Sales_recipe) %>% add_model(Game_Sales_regressior

fit_Game_Sales_workflow <- fit(Game_Sales_workflow2, Training_Game_Sales)

tuning_grid %>%
    collect_metrics() %>%
    ggplot(aes(penalty, mean, color = .metric)) +
    geom_errorbar(aes(
        ymin = mean - std_err,
        ymax = mean + std_err
    ),
    alpha = 0.5
    ) +
    geom_line(size = 1.5) +
    facet_wrap(~.metric, scales = "free", nrow = 2) +
    scale_x_log10() +
    theme(legend.position = "none")
```

```
## Warning: Removed 2 row(s) containing missing values (geom_path).
```




```
penalized_predictions <- predict(fit_Game_Sales_workflow, Test_Game_Sales)
```

```
## Warning: There are new levels in a factor: AO
```

```
penalized_predictions <- na.omit(penalized_predictions)
```

```
rmse(Test_Game_Sales$Global_Sales, penalized_predictions$.pred)
```

```
## Warning in actual - predicted: longer object length is not a multiple of shorter
## object length
```

```
## [1] 1.243238
```

based on the results of the extract_parsnip function it would appear that out of the variables present, the critic score appears to still have the highest influence on the global sales numbers for each game, with each other variable having less than half the amount of influence that the critic score variable had.

Conclusion: using the method of cross validation it has been shown that the penalized regression model that incorporates the genre, rating, platform, user score and critic score variables as predictors is mildly better than the linear regression model which only uses the critic score variable as a predictor due to the results of the best subset selection. This has been determined based off the root mean squared estimate that each model provided when paired with the test dataset, The linear regression model will consistently have a higher rmse value than the penalized regression model. (the actual rsme value between both can vary whenever the entire set of r code is run, likely due to the training and test sets being chosen at random, but everytime the entire set of code has been run, the linear regression model always has a slightly larger rmse value). The other main question of this project of determining which variable is most important in predicting the global sales amount for any given game and both models have indicated that the critic score is the most important, the penalized regression model especially so.

In terms of shortcomings the linear regression model was found to be not fully suitable for the dataset and predictors chosen, however I was unsure of what other modelling method to use for this project. Another major shortcoming of this project was the inability to use the developer and publisher variables in either of the models to see how much of an impact either of them had on the global sales amount due to their inclusion causing rstudio cloud to crash, likely due due to the large variety of developers and publishers found.

The approach I took for the linear regression section was to find the best combination of variables for predicting the global sales amount using the ols step best subset function, which yielded the result of the critic score being the most important. I decided to form the linear regression model using only the critic score variable to compare it against the penalized regression form while it still used all of the previous variables being considered. The next step involved forming a penalized regression model while determining the best possible penalty value for it and then looking at the results of calling extract parsnip on that model/workflow to see which variable led to the highest or most significant estimate value and once again the critic score was shown to be the most significant by a very large amount, although the other variables were shown to have some mild effect on the outcome. The final part of this project was comparing the rmse values of each model's attempts to predict global sales values against the actual test portion of the dataset. This was done to determine which of the two models was more accurate overall, the model only focusing on the critic score as a predictor or the model involving all of the initial predictors (albeit with a penalty). the results showed that the latter was more accurate indicating that while indeed the critic score is the most important variable in predicting global sales numbers, the combination of all of them as predictors can yield slightly more accurate results.

Analyzing the data has shown me that a game's critic rating is alot more indicative of a game's success than I initially thought, truthfully I didn't think it or the user score variables would be very relevant overall. This project has helped me review the main aspects of linear modelling and variable selection, although I do think that there is more stuff concerning penalized regression that I need to look over once more. This project was also a good opportunity to go over the basics of model comparison using cross validation. I would say another important lesson I learned for future projects in r is to choose datasets that don't contain so many varied categorical variables, I'm referring mainly to the developer and publisher variables, as I was legitimately expecting those two to have significant results, however caveats had to be made due to using the rstudio cloud environment.