Econometrics Final

Joel Ward

2023-05-09

## Introduction

Board games are a large industry across the world, and nearly every first-world household possesses at least one board game.  
The purpose of this project is to examine a dataset regarding board games from BoardGameGeek.com and create a predictive model to allow board game creators to know what factors contribute to the number of sales of a given game.

This data is an aggregation of reviews per game by users of BoardGameGeek.com. Individual reviews are not identifiable.  
While there is a column for the year release for each game, the data does not included changes over time, and the ‘Year’ column is used as a continuous independent variable. Therefore, we consider this data to be cross-sectional.

## Data

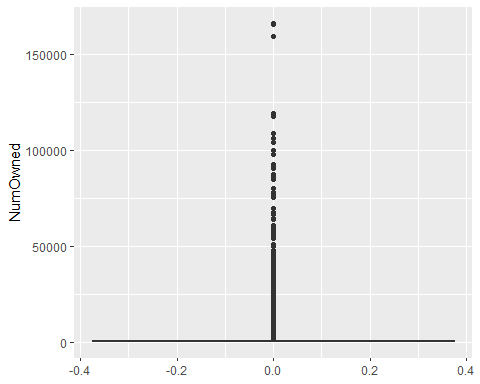
#Load backup to primary dataframe  
games <- games.bckup  
  
#Remove Description, ImagePath, NumWeightVotes, and Family columns, as these are not useful for regression  
#Data source documentation describes NumWeightVotes as '? Unknown'  
#Families indicates what franchise of game an entry falls under, if any. This is not useful for our prediction, as we are not assuming we can print from any given intellectual property  
games = subset(games, select = -c(Description,ImagePath,NumWeightVotes,Family) )

## Data Exploration

#Show Initial statistics on NumOwned, the variable to predict  
summary(games$NumOwned)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0 150 320 1468 899 166497

games %>%  
 ggplot(aes(y=NumOwned)) +  
 geom\_boxplot()



#These show that, while there is a wide range, most of the data is concentrated on the low end  
  
#How much of the data is at the lowest end? How low?  
nrow(games)

## [1] 21925

nrow(filter(games,NumOwned > 0))

## [1] 21924

nrow(filter(games,NumOwned > 100))

## [1] 19053

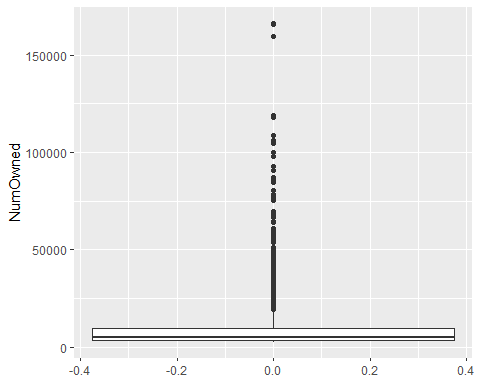
nrow(filter(games,NumOwned > 1000))

## [1] 5050

nrow(filter(games,NumOwned > 2500))

## [1] 2367

#Visualize the subset of games with more than 2500 owners  
subset(games,games$NumOwned > 2500) %>%  
 ggplot(aes(y=NumOwned)) +  
 geom\_boxplot()



#Because we intend to predict the factors that result in high sales, we would rather deal with less data than flood the model with low-sales data, which could impair our predictive ability.  
#Therefore, we remove rows containing games that show less than 2500 owners  
games<- filter(games,NumOwned > 2500)

#We would have checked for missing values earlier, but most rows with missing data are removed by taking only NumOwned > 2500  
#Showing the missing rows before this operation is excessively lengthy  
#Check for missing values  
sum(is.na(games))

## [1] 10

games[!complete.cases(games),]

## BGGId Name YearPublished  
## 1192 142057 Carcassonne Big Box 2006  
## 1368 160081 Dungeon Saga: Dwarf King's Quest 2015  
## 1705 198487 Smash Up: Cease and Desist 2016  
## 1783 207062 Dungeons & Dragons: Rock Paper Wizard 2016  
## 1832 216632 Tournament at Camelot 2017  
## 1952 232595 Skulk Hollow 2019  
## 2042 246701 DOS 2018  
## 2265 286537 Burgle Bros 2: The Casino Capers 2021  
## 2336 310203 Tiny Epic Pirates: Kickstarter Deluxe Edition 2021  
## 2340 312318 Century: Golem Edition – An Endless World 2020  
## GameWeight AvgRating BayesAvgRating StdDev MinPlayers MaxPlayers  
## 1192 2.3125 7.56939 6.23770 1.20947 2 5  
## 1368 2.4889 7.23863 6.05416 1.55229 2 5  
## 1705 2.5000 7.48337 6.28915 1.31308 2 2  
## 1783 1.3333 6.62641 5.95502 1.40830 3 6  
## 1832 2.3182 7.21534 6.20478 1.44692 3 6  
## 1952 2.2667 7.44614 6.29892 1.30961 2 2  
## 2042 1.3571 5.03760 5.32634 1.87973 2 4  
## 2265 2.5200 7.54948 6.10550 1.28227 1 4  
## 2336 2.6667 7.32749 5.79578 1.27474 1 4  
## 2340 2.0455 7.76483 6.10965 1.13179 2 4  
## ComAgeRec LanguageEase BestPlayers GoodPlayers NumOwned NumWant NumWish  
## 1192 8.00000 NA 0 [] 2802 22 97  
## 1368 9.00000 NA 0 [] 3010 81 543  
## 1705 NA 214.0 0 [] 5075 108 514  
## 1783 8.60000 NA 0 [] 2643 36 305  
## 1832 NA 272.5 0 [] 2567 199 1269  
## 1952 9.00000 NA 0 [] 3198 226 1516  
## 2042 7.00000 NA 0 [] 2886 5 34  
## 2265 10.80000 NA 0 [] 3213 192 1007  
## 2336 12.00000 NA 0 [] 2703 12 41  
## 2340 12.33333 NA 0 [] 2588 115 580  
## MfgPlaytime ComMinPlaytime ComMaxPlaytime MfgAgeRec NumUserRatings  
## 1192 45 45 45 8 1006  
## 1368 120 30 120 10 1050  
## 1705 45 45 45 12 1210  
## 1783 30 30 30 14 1225  
## 1832 45 45 45 14 1281  
## 1952 40 40 40 8 1256  
## 2042 45 30 45 7 751  
## 2265 70 45 70 10 741  
## 2336 60 30 60 14 341  
## 2340 45 30 45 8 629  
## NumComments NumAlternates NumExpansions NumImplementations  
## 1192 0 0 23 0  
## 1368 0 5 9 2  
## 1705 0 4 0 0  
## 1783 0 2 2 1  
## 1832 0 0 1 0  
## 1952 0 0 1 0  
## 2042 0 1 0 0  
## 2265 0 2 0 1  
## 2336 0 0 1 0  
## 2340 0 1 0 1  
## IsReimplementation Kickstarted Rank.boardgame Rank.strategygames  
## 1192 0 0 21926 21926  
## 1368 1 1 2271 21926  
## 1705 0 0 1501 21926  
## 1783 0 0 2760 21926  
## 1832 0 0 1759 21926  
## 1952 0 1 1473 758  
## 2042 0 0 21551 21926  
## 2265 1 1 2078 21926  
## 2336 0 1 21926 21926  
## 2340 1 0 2070 21926  
## Rank.abstracts Rank.familygames Rank.thematic Rank.cgs Rank.wargames  
## 1192 21926 21926 21926 21926 21926  
## 1368 21926 21926 409 21926 21926  
## 1705 21926 21926 21926 21926 21926  
## 1783 21926 21926 21926 21926 21926  
## 1832 21926 21926 21926 21926 21926  
## 1952 21926 21926 21926 21926 21926  
## 2042 21926 2275 21926 21926 21926  
## 2265 21926 21926 357 21926 21926  
## 2336 21926 21926 21926 21926 21926  
## 2340 21926 489 21926 21926 21926  
## Rank.partygames Rank.childrensgames Cat.Thematic Cat.Strategy Cat.War  
## 1192 21926 21926 0 0 0  
## 1368 21926 21926 1 0 0  
## 1705 21926 21926 0 0 0  
## 1783 179 21926 0 0 0  
## 1832 21926 21926 0 0 0  
## 1952 21926 21926 0 1 0  
## 2042 21926 21926 0 0 0  
## 2265 21926 21926 1 0 0  
## 2336 21926 21926 0 0 0  
## 2340 21926 21926 0 0 0  
## Cat.Family Cat.CGS Cat.Abstract Cat.Party Cat.Childrens  
## 1192 0 0 0 0 0  
## 1368 0 0 0 0 0  
## 1705 0 0 0 0 0  
## 1783 0 0 0 1 0  
## 1832 0 0 0 0 0  
## 1952 0 0 0 0 0  
## 2042 1 0 0 0 0  
## 2265 0 0 0 0 0  
## 2336 0 0 0 0 0  
## 2340 1 0 0 0 0

games$ComAgeRec <- ifelse(is.na(games$ComAgeRec),0,games$ComAgeRec)  
games$LanguageEase <- ifelse(is.na(games$LanguageEase),0,games$LanguageEase)  
games[!complete.cases(games),]

## [1] BGGId Name YearPublished   
## [4] GameWeight AvgRating BayesAvgRating   
## [7] StdDev MinPlayers MaxPlayers   
## [10] ComAgeRec LanguageEase BestPlayers   
## [13] GoodPlayers NumOwned NumWant   
## [16] NumWish MfgPlaytime ComMinPlaytime   
## [19] ComMaxPlaytime MfgAgeRec NumUserRatings   
## [22] NumComments NumAlternates NumExpansions   
## [25] NumImplementations IsReimplementation Kickstarted   
## [28] Rank.boardgame Rank.strategygames Rank.abstracts   
## [31] Rank.familygames Rank.thematic Rank.cgs   
## [34] Rank.wargames Rank.partygames Rank.childrensgames  
## [37] Cat.Thematic Cat.Strategy Cat.War   
## [40] Cat.Family Cat.CGS Cat.Abstract   
## [43] Cat.Party Cat.Childrens   
## <0 rows> (or 0-length row.names)

#Initial statistics of BestPlayers column  
summary(games$BestPlayers)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 3.000 2.512 4.000 15.000

#This shows a trend of mostly zeros in this column  
sum(games$BestPlayers==0)

## [1] 650

nrow(games)

## [1] 2367

nrow(filter(games,BestPlayers > 0))

## [1] 1717

#These confirm this trend, BestPlayers does not appear to be useful

#Summary of GoodPlayers  
summary(games$GoodPlayers)

## Length Class Mode   
## 2367 character character

#This is a character variable, not useful for regression

#Summary of NumComments  
summary(games$NumComments)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0 0 0 0 0 0

unique(games$NumComments)

## [1] 0

#This variable is not useful for regression

str(games)

## 'data.frame': 2367 obs. of 44 variables:  
## $ BGGId : int 1 3 5 7 9 10 11 12 13 15 ...  
## $ Name : chr "Die Macher" "Samurai" "Acquire" "Cathedral" ...  
## $ YearPublished : int 1986 1998 1964 1978 1998 1998 1997 1999 1995 1977 ...  
## $ GameWeight : num 4.32 2.49 2.5 1.79 3.18 ...  
## $ AvgRating : num 7.61 7.46 7.34 6.52 6.45 ...  
## $ BayesAvgRating : num 7.1 7.24 7.14 6.14 5.92 ...  
## $ StdDev : num 1.58 1.18 1.34 1.33 1.43 ...  
## $ MinPlayers : int 3 2 2 2 2 2 2 2 3 2 ...  
## $ MaxPlayers : int 5 4 6 2 4 6 7 5 4 6 ...  
## $ ComAgeRec : num 14.37 9.31 11.41 8.14 11.78 ...  
## $ LanguageEase : num 1.4 1 21.15 51 1.11 ...  
## $ BestPlayers : int 5 3 4 0 3 4 5 3 4 4 ...  
## $ GoodPlayers : chr "['4', '5']" "['2', '3', '4']" "['3', '4', '5']" "[]" ...  
## $ NumOwned : int 7498 15578 23735 6021 2611 10115 59803 19094 165651 4326 ...  
## $ NumWant : int 501 799 548 101 87 179 362 1238 484 122 ...  
## $ NumWish : int 2039 3450 2671 589 311 759 2676 4798 5865 462 ...  
## $ MfgPlaytime : int 240 60 90 20 90 60 45 60 120 90 ...  
## $ ComMinPlaytime : int 240 30 90 20 90 60 45 45 60 90 ...  
## $ ComMaxPlaytime : int 240 60 90 20 90 60 45 60 120 90 ...  
## $ MfgAgeRec : int 14 10 12 8 13 10 13 12 10 12 ...  
## $ NumUserRatings : int 5354 15146 18655 3320 1389 8324 39886 19685 107141 3883 ...  
## $ NumComments : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ NumAlternates : int 2 6 6 5 1 1 20 4 59 5 ...  
## $ NumExpansions : int 0 0 2 0 0 6 27 0 88 11 ...  
## $ NumImplementations : int 0 1 0 0 0 3 11 3 29 4 ...  
## $ IsReimplementation : int 0 0 0 0 0 1 0 0 0 0 ...  
## $ Kickstarted : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ Rank.boardgame : int 316 224 290 1969 2948 1117 468 186 427 1190 ...  
## $ Rank.strategygames : int 180 166 220 21926 1297 21926 21926 138 409 723 ...  
## $ Rank.abstracts : int 21926 21926 21926 137 21926 21926 21926 21926 21926 21926 ...  
## $ Rank.familygames : int 21926 21926 21926 21926 21926 308 116 21926 129 21926 ...  
## $ Rank.thematic : int 21926 21926 21926 21926 21926 21926 21926 21926 21926 308 ...  
## $ Rank.cgs : int 21926 21926 21926 21926 21926 21926 21926 21926 21926 21926 ...  
## $ Rank.wargames : int 21926 21926 21926 21926 21926 21926 21926 21926 21926 21926 ...  
## $ Rank.partygames : int 21926 21926 21926 21926 21926 21926 21926 21926 21926 21926 ...  
## $ Rank.childrensgames: int 21926 21926 21926 21926 21926 21926 21926 21926 21926 21926 ...  
## $ Cat.Thematic : int 0 0 0 0 0 0 0 0 0 1 ...  
## $ Cat.Strategy : int 1 1 1 0 1 0 0 1 1 1 ...  
## $ Cat.War : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ Cat.Family : int 0 0 0 0 0 1 1 0 1 0 ...  
## $ Cat.CGS : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ Cat.Abstract : int 0 0 0 1 0 0 0 0 0 0 ...  
## $ Cat.Party : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ Cat.Childrens : int 0 0 0 0 0 0 0 0 0 0 ...

#confirm these variables are binary  
unique(games$IsReimplementation)

## [1] 0 1

unique(games$Kickstarted)

## [1] 0 1

unique(games$Cat.Thematic)

## [1] 0 1

unique(games$Cat.Strategy)

## [1] 1 0

unique(games$Cat.War)

## [1] 0 1

unique(games$Cat.Family)

## [1] 0 1

unique(games$Cat.CGS)

## [1] 0 1

unique(games$Cat.Abstract)

## [1] 0 1

unique(games$Cat.Party)

## [1] 0 1

unique(games$Cat.Childrens)

## [1] 0 1

#They are binary  
  
#Check how many of games fall into these groups  
games %>% count(IsReimplementation)

## IsReimplementation n  
## 1 0 1869  
## 2 1 498

games %>% count(Kickstarted)

## Kickstarted n  
## 1 0 1881  
## 2 1 486

games %>% count(Cat.Thematic)

## Cat.Thematic n  
## 1 0 1900  
## 2 1 467

games %>% count(Cat.Strategy)

## Cat.Strategy n  
## 1 0 1435  
## 2 1 932

games %>% count(Cat.War)

## Cat.War n  
## 1 0 2213  
## 2 1 154

games %>% count(Cat.Family)

## Cat.Family n  
## 1 0 1610  
## 2 1 757

games %>% count(Cat.CGS)

## Cat.CGS n  
## 1 0 2308  
## 2 1 59

games %>% count(Cat.Abstract)

## Cat.Abstract n  
## 1 0 2253  
## 2 1 114

games %>% count(Cat.Party)

## Cat.Party n  
## 1 0 2150  
## 2 1 217

games %>% count(Cat.Childrens)

## Cat.Childrens n  
## 1 0 2304  
## 2 1 63

#Of these groups, the most are strategy games, family games, and reimplementations  
#The least are a card games, childrens games, and abstract games

#Initial statistics of first appearing subgroup rank  
#Strategy games  
summary(games$Rank.strategygames)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.0 604.5 21926.0 13499.3 21926.0 21926.0

#This summary shows that the Median, 3rd Quartile, and Maximum are all equal to 21926  
  
head(games[,c('Rank.strategygames','Cat.Strategy')],10)

## Rank.strategygames Cat.Strategy  
## 1 180 1  
## 2 166 1  
## 3 220 1  
## 4 21926 0  
## 5 1297 1  
## 6 21926 0  
## 7 21926 0  
## 8 138 1  
## 9 409 1  
## 10 723 1

#By viewing the rank values and category indicator together we see that games not in the category have rank values of 21926  
  
  
#We will replace the 21926 values in each subgroup with zero values  
games$Rank.strategygames <- ifelse(games$Rank.strategygames==21926,0,games$Rank.strategygames)  
games$Rank.abstracts <- ifelse(games$Rank.abstracts==21926,0,games$Rank.abstracts)  
games$Rank.familygames <- ifelse(games$Rank.familygames==21926,0,games$Rank.familygames)  
games$Rank.thematic <- ifelse(games$Rank.thematic==21926,0,games$Rank.thematic)  
games$Rank.cgs <- ifelse(games$Rank.cgs==21926,0,games$Rank.cgs)  
games$Rank.wargames <- ifelse(games$Rank.wargames==21926,0,games$Rank.wargames)  
games$Rank.partygames <- ifelse(games$Rank.partygames==21926,0,games$Rank.partygames)  
games$Rank.childrensgames <- ifelse(games$Rank.childrensgames==21926,0,games$Rank.childrensgames)  
#This removal will alter the predictive ability of these variables, but we do not expect this alteration to be greatly significant  
  
#Initial statistics of Rank.boardgame  
summary(games$Rank.boardgame)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.0 592.5 1198.0 2513.9 2122.5 21926.0

#This column appears regular, and does not have the problems of the subgroups

summary(subset(games$NumOwned,(games$Cat.Abstract+games$Cat.Thematic+games$Cat.Strategy+games$Cat.War+games$Cat.Family+games$Cat.CGS+games$Cat.Party+games$Cat.Childrens > 1)))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2507 3498 5210 10574 9534 166497

summary(games$NumOwned)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2501 3478 5099 9658 9770 166497

summary(subset(games$NumOwned,games$Cat.Family==1))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2502 3447 5429 11084 10533 166497

summary(aov(NumOwned ~ Cat.Abstract+Cat.Family+Cat.Strategy+Cat.War+Cat.CGS+Cat.Party+Cat.Childrens,data = games))

## Df Sum Sq Mean Sq F value Pr(>F)   
## Cat.Abstract 1 1.144e+08 1.144e+08 0.635 0.425531   
## Cat.Family 1 2.268e+09 2.268e+09 12.588 0.000396 \*\*\*  
## Cat.Strategy 1 1.208e+09 1.208e+09 6.705 0.009671 \*\*   
## Cat.War 1 8.553e+08 8.553e+08 4.748 0.029439 \*   
## Cat.CGS 1 5.866e+07 5.866e+07 0.326 0.568311   
## Cat.Party 1 8.915e+08 8.915e+08 4.949 0.026206 \*   
## Cat.Childrens 1 1.057e+09 1.057e+09 5.869 0.015486 \*   
## Residuals 2359 4.250e+11 1.802e+08   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

summary(aov(NumOwned ~ Cat.Abstract\*Cat.Family\*Cat.Strategy\*Cat.War\*Cat.CGS\*Cat.Party\*Cat.Childrens,data = games))

## Df Sum Sq Mean Sq F value Pr(>F)  
## Cat.Abstract 1 1.144e+08 1.144e+08 0.637 0.424793  
## Cat.Family 1 2.268e+09 2.268e+09 12.628 0.000387  
## Cat.Strategy 1 1.208e+09 1.208e+09 6.727 0.009556  
## Cat.War 1 8.553e+08 8.553e+08 4.763 0.029182  
## Cat.CGS 1 5.866e+07 5.866e+07 0.327 0.567693  
## Cat.Party 1 8.915e+08 8.915e+08 4.964 0.025969  
## Cat.Childrens 1 1.057e+09 1.057e+09 5.888 0.015323  
## Cat.Abstract:Cat.Family 1 1.088e+09 1.088e+09 6.056 0.013930  
## Cat.Abstract:Cat.Strategy 1 1.156e+08 1.156e+08 0.644 0.422513  
## Cat.Family:Cat.Strategy 1 2.564e+08 2.564e+08 1.428 0.232230  
## Cat.Family:Cat.War 1 3.950e+08 3.950e+08 2.199 0.138201  
## Cat.Strategy:Cat.War 1 3.303e+08 3.303e+08 1.839 0.175174  
## Cat.Strategy:Cat.CGS 1 1.163e+08 1.163e+08 0.648 0.421032  
## Cat.War:Cat.CGS 1 1.595e+08 1.595e+08 0.888 0.346134  
## Cat.Family:Cat.Party 1 1.078e+09 1.078e+09 6.001 0.014370  
## Cat.Strategy:Cat.Party 1 2.164e+08 2.164e+08 1.205 0.272472  
## Cat.Abstract:Cat.Childrens 1 3.598e+07 3.598e+07 0.200 0.654483  
## Cat.Family:Cat.Childrens 1 1.687e+07 1.687e+07 0.094 0.759237  
## Cat.Party:Cat.Childrens 1 4.745e+07 4.745e+07 0.264 0.607282  
## Cat.Family:Cat.Strategy:Cat.Party 1 4.396e+06 4.396e+06 0.024 0.875684  
## Cat.Family:Cat.Party:Cat.Childrens 1 9.758e+06 9.758e+06 0.054 0.815699  
## Residuals 2345 4.211e+11 1.796e+08   
##   
## Cat.Abstract   
## Cat.Family \*\*\*  
## Cat.Strategy \*\*   
## Cat.War \*   
## Cat.CGS   
## Cat.Party \*   
## Cat.Childrens \*   
## Cat.Abstract:Cat.Family \*   
## Cat.Abstract:Cat.Strategy   
## Cat.Family:Cat.Strategy   
## Cat.Family:Cat.War   
## Cat.Strategy:Cat.War   
## Cat.Strategy:Cat.CGS   
## Cat.War:Cat.CGS   
## Cat.Family:Cat.Party \*   
## Cat.Strategy:Cat.Party   
## Cat.Abstract:Cat.Childrens   
## Cat.Family:Cat.Childrens   
## Cat.Party:Cat.Childrens   
## Cat.Family:Cat.Strategy:Cat.Party   
## Cat.Family:Cat.Party:Cat.Childrens   
## Residuals   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

summary(aov(NumOwned ~ Cat.Family\*Cat.Strategy,data = games))

## Df Sum Sq Mean Sq F value Pr(>F)   
## Cat.Family 1 2.266e+09 2.266e+09 12.511 0.000412 \*\*\*  
## Cat.Strategy 1 1.083e+09 1.083e+09 5.978 0.014558 \*   
## Cat.Family:Cat.Strategy 1 1.515e+08 1.515e+08 0.836 0.360501   
## Residuals 2363 4.279e+11 1.811e+08   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

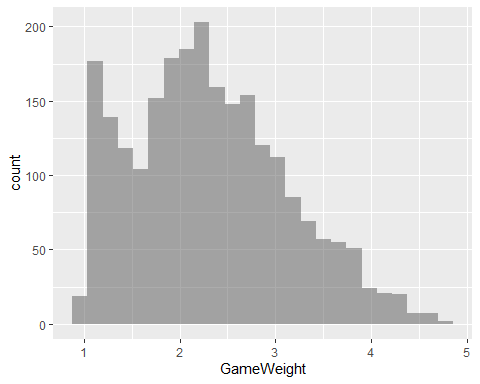
summary(aov(NumOwned ~ Rank.abstracts+Rank.familygames+Rank.strategygames+Rank.wargames+Rank.cgs+Rank.partygames+Rank.childrensgames,data = games))

## Df Sum Sq Mean Sq F value Pr(>F)   
## Rank.abstracts 1 1.034e+08 1.034e+08 0.592 0.4416   
## Rank.familygames 1 2.975e+09 2.975e+09 17.039 3.79e-05 \*\*\*  
## Rank.strategygames 1 1.441e+10 1.441e+10 82.516 < 2e-16 \*\*\*  
## Rank.wargames 1 4.913e+08 4.913e+08 2.814 0.0936 .   
## Rank.cgs 1 8.969e+08 8.969e+08 5.138 0.0235 \*   
## Rank.partygames 1 2.181e+08 2.181e+08 1.249 0.2638   
## Rank.childrensgames 1 5.326e+08 5.326e+08 3.051 0.0808 .   
## Residuals 2359 4.118e+11 1.746e+08   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

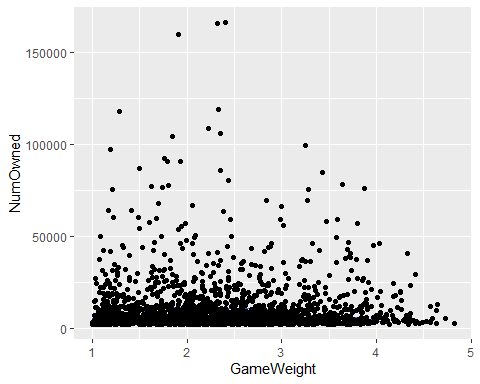
#GameWeight exploration  
summary(games$GameWeight)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.000 1.716 2.235 2.308 2.843 4.824

#Game weights are on a 0 to 5 scale  
#We will visualize this variable  
gf\_histogram(games, ~GameWeight)



gf\_plot(games, x = ~GameWeight, y = ~NumOwned)%>%  
 gf\_density\_2d() %>%  
 gf\_point()



#There doesn't appear to be much of a linear relationship but there may be some quadratic relationship, so we will test this now  
games$gweightsq <- games$GameWeight\*games$GameWeight  
summary(lm(NumOwned~GameWeight+gweightsq, data = games))

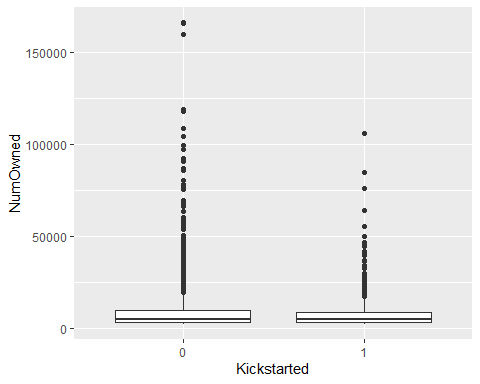
##   
## Call:  
## lm(formula = NumOwned ~ GameWeight + gweightsq, data = games)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -8096 -6172 -4563 95 156922   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 9991.1 2145.6 4.657 3.39e-06 \*\*\*  
## GameWeight -548.7 1816.9 -0.302 0.763   
## gweightsq 156.1 358.8 0.435 0.664   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 13510 on 2364 degrees of freedom  
## Multiple R-squared: 0.000264, Adjusted R-squared: -0.0005818   
## F-statistic: 0.3122 on 2 and 2364 DF, p-value: 0.7319

#We concluced that Gameweight is not useful linearly or quadratically  
#We will also discard NumComments, BestPlayers, and GoodPlayers at this point  
games = subset(games, select = -c(GameWeight,gweightsq,NumComments,BestPlayers,GoodPlayers) )

#Lets take a look at Kickstarted games, and how does their popularity compare  
#Make Kickstarted a factor to better facilitate upcoming work  
games$Kickstarted <- as.factor(games$Kickstarted)  
  
#Summary  
summary(games$Kickstarted)

## 0 1   
## 1881 486

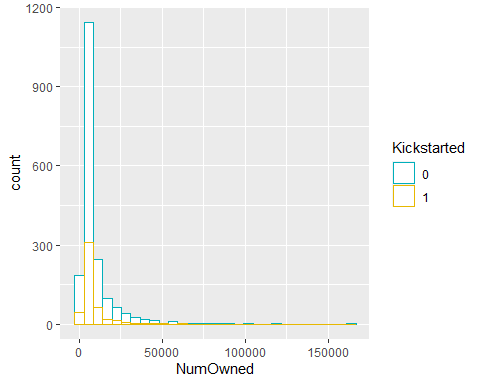
#Visualize with a boxplot  
games %>%  
 ggplot(aes(x=Kickstarted, y=NumOwned)) +  
 geom\_boxplot()



#Compare mean NumOwned for Kickstarted and non-Kickstarted games  
games %>%  
 group\_by((Kickstarted)) %>%  
 summarize(mean = mean(NumOwned))

## # A tibble: 2 × 2  
## `(Kickstarted)` mean  
## <fct> <dbl>  
## 1 0 9840.  
## 2 1 8951.

#Visualize the difference between the groups with a histogram  
ggplot(games, aes(x = NumOwned)) +  
 geom\_histogram(aes(color = Kickstarted), fill = "white",  
 position = "identity", bins = 30) +  
 scale\_color\_manual(values = c("#00AFBB", "#E7B800"))



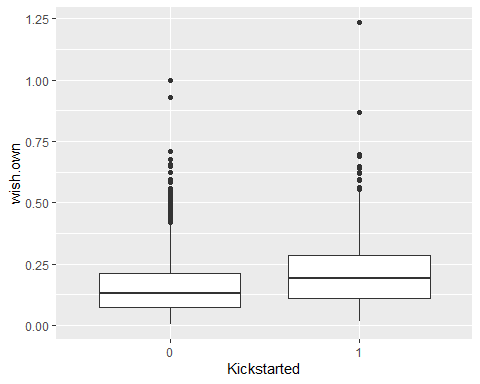
#Is the ratio of wished for to owned games different for Kickstarted games?  
#Are people more likely to purchase a game they are reasonably sure they like if it is a Kickstarter?  
games$wish.own <- (games$NumWish/games$NumOwned)  
summary(games$wish.own)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.004267 0.079280 0.141493 0.165692 0.227441 1.235519

#This summary indicated a generally very low ratio  
  
#Compare W/O ratio for kickstarted games  
games %>%  
 group\_by((Kickstarted)) %>%  
 summarize(mean = mean(wish.own))

## # A tibble: 2 × 2  
## `(Kickstarted)` mean  
## <fct> <dbl>  
## 1 0 0.154  
## 2 1 0.213

#Visualize with a pair of boxplots  
games %>%  
 ggplot(aes(x=Kickstarted, y=wish.own)) +  
 geom\_boxplot()

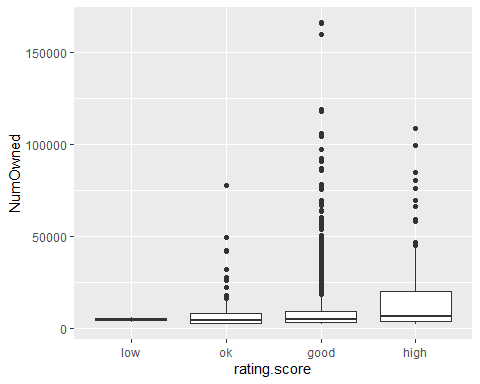


#Do a t test to see if difference in means is significant  
t.test(wish.own ~ as.numeric(Kickstarted),data = games)

##   
## Welch Two Sample t-test  
##   
## data: wish.own by as.numeric(Kickstarted)  
## t = -8.7386, df = 657.29, p-value < 2.2e-16  
## alternative hypothesis: true difference in means between group 1 and group 2 is not equal to 0  
## 95 percent confidence interval:  
## -0.07236624 -0.04581150  
## sample estimates:  
## mean in group 1 mean in group 2   
## 0.1535595 0.2126484

#This tells us that the difference is statistically significant,

games$rating.score <- (ifelse(games$AvgRating > 8, 'high',  
 ifelse(games$AvgRating > 6, 'good',  
 ifelse(games$AvgRating > 4, 'ok',  
 ifelse(games$AvgRating > 2, 'low',  
 'bad')))))  
  
games %>%  
 arrange(AvgRating) %>%  
 mutate(rating.score = factor(rating.score, levels=c("bad", "low", "ok", "good", "high"))) %>%  
 ggplot(aes(x=rating.score, y=NumOwned)) +  
 geom\_boxplot()



sum(games$AvgRating < 2)

## [1] 0

#Removing the least owned games also removed the worst rated games  
#Compare number of 'good' entries to 'high' entries  
sum(games$rating.score == 'good')

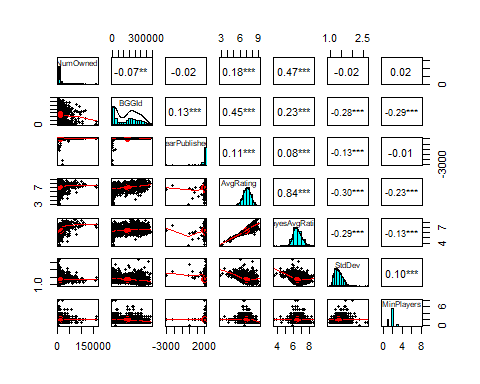
## [1] 2086

sum(games$rating.score == 'high')

## [1] 145

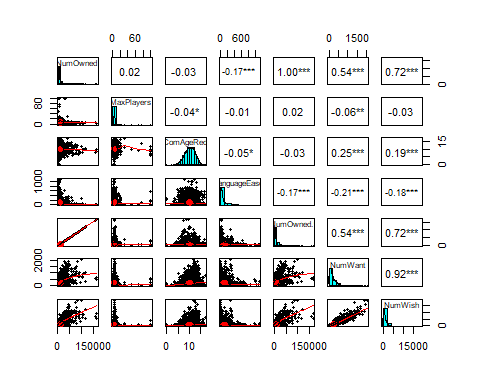
#We will now run pairs.panels to see relationship between NumOwned and other variables, and relations between the independent variables   
pairs.panels(games[,c(11, 1,3,4,5,6,7)],stars = TRUE)

## Warning in x \* y: NAs produced by integer overflow



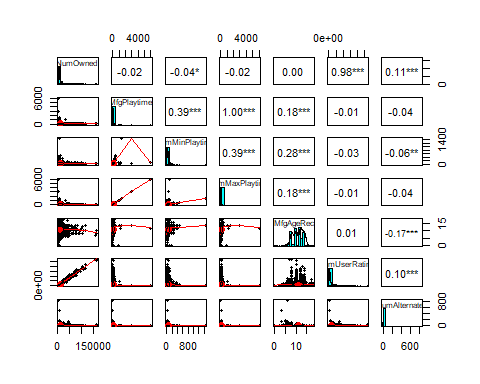
pairs.panels(games[,c(11, 8,9,10,11,12,13)],stars = TRUE)

## Warning in x \* y: NAs produced by integer overflow

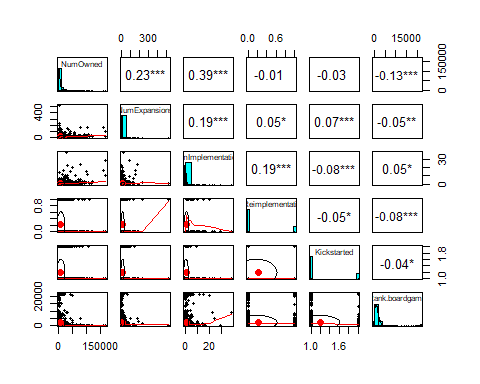


pairs.panels(games[,c(11, 14,15,16,17,18,19)],stars = TRUE)

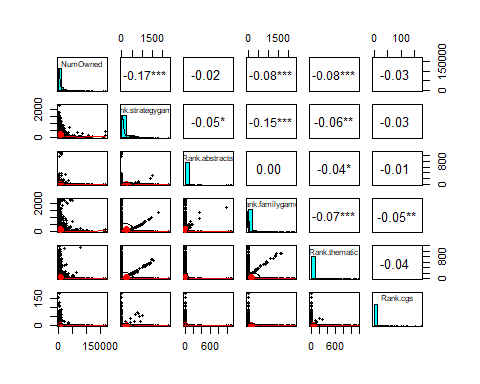
## Warning in x \* y: NAs produced by integer overflow



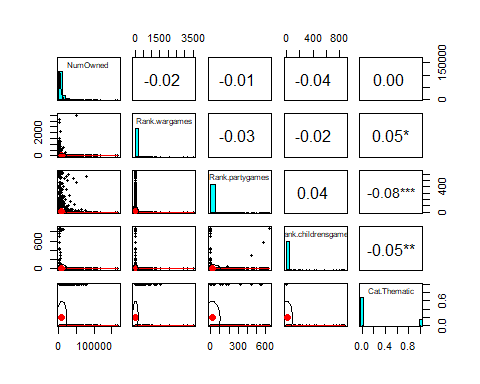
pairs.panels(games[,c(11, 20,21,22,23,24)],stars = TRUE)



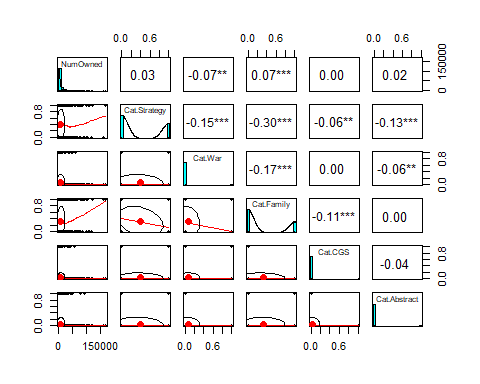
pairs.panels(games[,c(11, 25,26,27,28,29)],stars = TRUE)



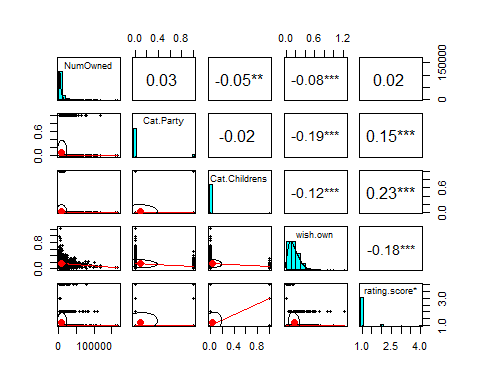
pairs.panels(games[,c(11, 30,31,32,33)],stars = TRUE)



pairs.panels(games[,c(11, 34,35,36,37,38)],stars = TRUE)



pairs.panels(games[,c(11, 39,40,41,42)],stars = TRUE)



#Num User Rating is not a significant predictor, as most persons who review a game purchase the game before reviewing it  
#Additionally, we will not include NumWish or NumWant, as these indicate how well a game is likely, but do not provide predictive ability to determine how well a game is expected to be liked

From these panels, we determine that BayesAvgRating, LanguageEase, ComMinPlaytime, NumAlternates, NumExpansions, NumImplementations, Rank.boardgame, Rank.strategygames, Rank.familygames, Rank.wargames, Rank.wargames, Cat.Family, Cat.War, Cat.Strategy, and Rating.score appear most useful, and will be entered into our prediction model.

## Linear Regression

For our prediction, we will use a multivariate linear regression model, of the form

This form of regression model is of similar function to a single variable model, but allows more factors. Compared to other model, though, it’s comparatively simple nature may mean it is less precise. However, given the data set, most other forms of model simply are not necessary. For example, as the data is cross-sectional and has no time-dependant component, a time-series regression of the form

would be overcomplicated to no gain. A multivariable linear model such as this is subject to the classical linear model assumptions, namely linear parameters, random sampling, no perfect collinearity, assumption of zero conditional mean , homoskedasticity, and an independent and normal error. If any of these assumptions are violated, we lose confidence in the exact predictive power of the model. For the time being, we will proceed under the assumption we meet these criteria, and re-examine them after the model is constructed.

We have already selected the variables to go into the model, as above, to our model will take the form of

Note that there are four categorical variables in this model, and that rating.score is a categorical variable with four levels, and so will appear 3 times, with one level acting as a base.

#Create a linear regression model  
games.model <- lm(NumOwned ~BayesAvgRating+LanguageEase+ComMinPlaytime+NumAlternates+NumExpansions+NumImplementations+Rank.boardgame+Rank.strategygames+Rank.familygames+Rank.wargames+Cat.Family+Cat.War+Cat.Strategy+rating.score, data = games)  
summary(games.model)

##   
## Call:  
## lm(formula = NumOwned ~ BayesAvgRating + LanguageEase + ComMinPlaytime +   
## NumAlternates + NumExpansions + NumImplementations + Rank.boardgame +   
## Rank.strategygames + Rank.familygames + Rank.wargames + Cat.Family +   
## Cat.War + Cat.Strategy + rating.score, data = games)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -67413 -4600 -1073 2642 116960   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.048e+05 4.261e+03 -24.607 < 2e-16 \*\*\*  
## BayesAvgRating 1.726e+04 6.549e+02 26.360 < 2e-16 \*\*\*  
## LanguageEase 1.476e+00 1.278e+00 1.155 0.248118   
## ComMinPlaytime -1.122e+01 4.220e+00 -2.660 0.007876 \*\*   
## NumAlternates 5.827e+01 9.342e+00 6.237 5.28e-10 \*\*\*  
## NumExpansions 6.844e+01 1.153e+01 5.935 3.37e-09 \*\*\*  
## NumImplementations 2.353e+03 1.144e+02 20.560 < 2e-16 \*\*\*  
## Rank.boardgame 3.597e-02 6.230e-02 0.577 0.563771   
## Rank.strategygames 5.531e+00 1.156e+00 4.785 1.81e-06 \*\*\*  
## Rank.familygames -5.504e-01 8.814e-01 -0.625 0.532348   
## Rank.wargames 4.148e+00 1.511e+00 2.745 0.006101 \*\*   
## Cat.Family 2.902e+03 6.788e+02 4.275 1.99e-05 \*\*\*  
## Cat.War -3.442e+03 1.092e+03 -3.153 0.001638 \*\*   
## Cat.Strategy -5.524e+03 8.755e+02 -6.310 3.33e-10 \*\*\*  
## rating.scorehigh -3.730e+03 9.658e+02 -3.862 0.000116 \*\*\*  
## rating.scorelow 3.076e+04 5.316e+03 5.787 8.10e-09 \*\*\*  
## rating.scoreok 1.150e+04 1.321e+03 8.705 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 9914 on 2350 degrees of freedom  
## Multiple R-squared: 0.4647, Adjusted R-squared: 0.461   
## F-statistic: 127.5 on 16 and 2350 DF, p-value: < 2.2e-16

#To ensure that our model is as accurate as possible, we will run a stepwise regression model  
games.step.model <- stepAIC(games.model, direction = "both", trace = FALSE)  
summary(games.step.model)

##   
## Call:  
## lm(formula = NumOwned ~ BayesAvgRating + ComMinPlaytime + NumAlternates +   
## NumExpansions + NumImplementations + Rank.strategygames +   
## Rank.wargames + Cat.Family + Cat.War + Cat.Strategy + rating.score,   
## data = games)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -67653 -4582 -1060 2599 117407   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.038e+05 3.800e+03 -27.322 < 2e-16 \*\*\*  
## BayesAvgRating 1.716e+04 5.927e+02 28.957 < 2e-16 \*\*\*  
## ComMinPlaytime -1.160e+01 4.209e+00 -2.757 0.005885 \*\*   
## NumAlternates 5.815e+01 9.296e+00 6.256 4.68e-10 \*\*\*  
## NumExpansions 6.783e+01 1.151e+01 5.893 4.35e-09 \*\*\*  
## NumImplementations 2.340e+03 1.140e+02 20.517 < 2e-16 \*\*\*  
## Rank.strategygames 5.670e+00 1.137e+00 4.988 6.56e-07 \*\*\*  
## Rank.wargames 4.271e+00 1.507e+00 2.834 0.004633 \*\*   
## Cat.Family 2.553e+03 4.866e+02 5.246 1.70e-07 \*\*\*  
## Cat.War -3.560e+03 1.085e+03 -3.280 0.001055 \*\*   
## Cat.Strategy -5.738e+03 8.509e+02 -6.744 1.94e-11 \*\*\*  
## rating.scorehigh -3.557e+03 9.319e+02 -3.817 0.000139 \*\*\*  
## rating.scorelow 3.101e+04 5.191e+03 5.974 2.67e-09 \*\*\*  
## rating.scoreok 1.134e+04 1.103e+03 10.280 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 9912 on 2353 degrees of freedom  
## Multiple R-squared: 0.4642, Adjusted R-squared: 0.4613   
## F-statistic: 156.8 on 13 and 2353 DF, p-value: < 2.2e-16

The stepwise method reduces our model to

This model indicates to us through t values that Bayes Average Rating and Number of Implementations are the most significant variables. Overall, these tell us that we can expect less sales from games with a high minimum play time, war games, strategy games, and games rated particularly highly.  
We expect the number of alternate versions and implementations, and the number of expansions to contribute to higher sales. Also, while war games and strategy games as categories indicate lower sales, we expect games rated highly in these categories to have increased sales.

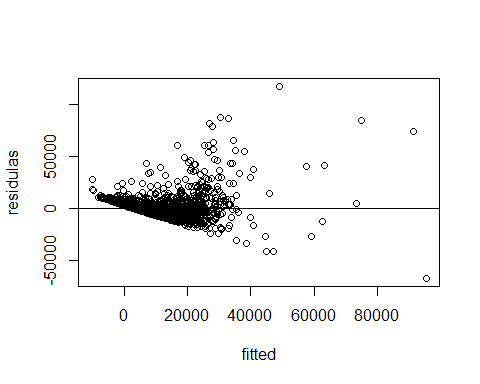
### Model Validation

For total confidence in this model, it must meet the classical linear regression assumptions, as above.  
To begin, all of our variables are linear or categorical variables handled linearly. We can see from our pair.panels plots that none of the independent variables are perfectly collinear with the dependent variable. We must simply assume a zero conditional mean, and an independent and normal error. Next, however, we must acknowledge that the sampling for this data is an aggregation of self-selected data across persons who choose to use the BoardGameGeek.com website, which is far from a statistically advisable random sample.

Our next step is to check this model with several methods to check its validity, including heteroskedastisity.

First we will check for heteroskedastisity, which is the trend for the error in fitted values to increase as the fitted values themselves increase  
We will do this through visualization and through the Breuch-Pagan test for heteroskedastisity.

#Visualize by plotting residuals against fitted values  
residulas <- resid(games.step.model)  
fitted <- fitted(games.step.model)  
plot(fitted,residulas) + abline(0,0)



## integer(0)

#Conduct B-P test  
bptest(games.step.model)

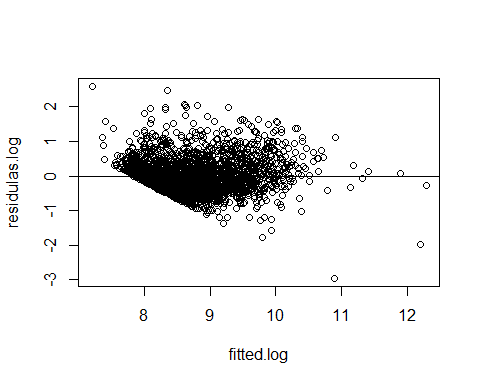
##   
## studentized Breusch-Pagan test  
##   
## data: games.step.model  
## BP = 431.66, df = 13, p-value < 2.2e-16

Unfortunately, these results indicate that this model suffers from notable heteroskedastisity. In an attempt to remedy this, we recreated the model, while performing a log transformation on the dependant variable, NumOwned, using the same original variables and an identical stepwise process, which will give us a model of the form

#Create log transformed model  
games.model.log <- lm(log(NumOwned) ~BayesAvgRating+LanguageEase+ComMinPlaytime+NumAlternates+NumExpansions+NumImplementations+Rank.boardgame+Rank.strategygames+Rank.familygames+Rank.wargames+Cat.Family+Cat.War+Cat.Strategy+rating.score, data = games)  
#Stepwise method  
games.step.model.log <- stepAIC(games.model.log, direction = "both", trace = FALSE)  
summary(games.step.model.log)

##   
## Call:  
## lm(formula = log(NumOwned) ~ BayesAvgRating + LanguageEase +   
## ComMinPlaytime + NumAlternates + NumExpansions + NumImplementations +   
## Rank.strategygames + Rank.wargames + Cat.Family + Cat.War +   
## Cat.Strategy + rating.score, data = games)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.95639 -0.32180 -0.07374 0.25137 2.58753   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.200e-01 1.963e-01 2.139 0.032553 \*   
## BayesAvgRating 1.285e+00 3.022e-02 42.520 < 2e-16 \*\*\*  
## LanguageEase -1.985e-04 6.322e-05 -3.140 0.001709 \*\*   
## ComMinPlaytime -7.809e-04 2.091e-04 -3.735 0.000192 \*\*\*  
## NumAlternates 3.580e-03 4.618e-04 7.753 1.33e-14 \*\*\*  
## NumExpansions 2.232e-03 5.714e-04 3.906 9.63e-05 \*\*\*  
## NumImplementations 9.350e-02 5.673e-03 16.481 < 2e-16 \*\*\*  
## Rank.strategygames 2.160e-04 5.642e-05 3.828 0.000133 \*\*\*  
## Rank.wargames 3.115e-04 7.481e-05 4.163 3.25e-05 \*\*\*  
## Cat.Family 1.180e-01 2.417e-02 4.880 1.13e-06 \*\*\*  
## Cat.War -3.281e-01 5.390e-02 -6.088 1.33e-09 \*\*\*  
## Cat.Strategy -3.586e-01 4.230e-02 -8.478 < 2e-16 \*\*\*  
## rating.scorehigh -4.275e-01 4.653e-02 -9.188 < 2e-16 \*\*\*  
## rating.scorelow 2.502e+00 2.587e-01 9.672 < 2e-16 \*\*\*  
## rating.scoreok 9.271e-01 5.560e-02 16.673 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4919 on 2352 degrees of freedom  
## Multiple R-squared: 0.6142, Adjusted R-squared: 0.612   
## F-statistic: 267.5 on 14 and 2352 DF, p-value: < 2.2e-16

#heteroskedastisity tests for log model  
#Visualize  
residulas.log <- resid(games.step.model.log)  
fitted.log <- fitted(games.step.model.log)  
plot(fitted.log,residulas.log) + abline(0,0)



## integer(0)

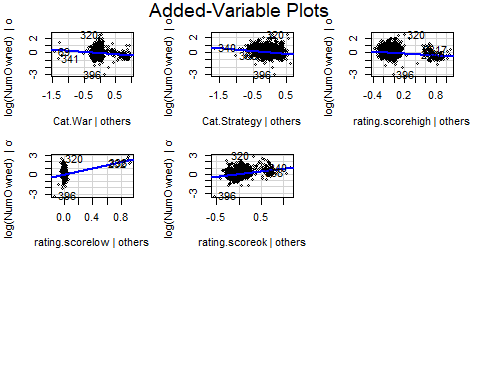
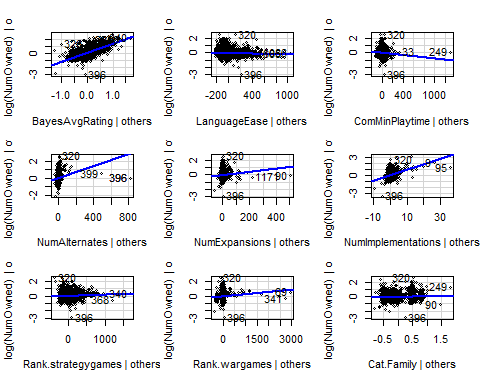
#B-P test  
bptest(games.step.model.log)

##   
## studentized Breusch-Pagan test  
##   
## data: games.step.model.log  
## BP = 452.99, df = 14, p-value < 2.2e-16

These results show us an improvement in skedastisity, where the visualization appear to indicate less of a spreading trend, but the Breuch-Pagan test still returns a unfavorable result.  
At this point, we have no method to further remedy the heteroskedastisity, as it appears to be a function of the data itself. We will carry on using the log model in all further discussion. Also note that the stepwise method in this instance includes LanguageEase, with a negative coefficient, indicating games with easier to read rules sell worse.

We next observe the Added-Variable Plots, to see how each variable in the regression contributes to the prediction ‘ceteris paribus’

#Added Variable plots  
avPlots(games.step.model.log)

 In a broad sense, these plots show us that Bayes Avg Rating and Language Ease have the most direct linear relationships with NumOwned, and that many of the other variables show a heavy clump of data points, such that while there is an increase in NumOwned as they increase, it is difficult to describe their behavior as a linear trend. Additionally, many of the factors in the model are categorical, so instead of presenting a trend they are more a comparison of average values for the categories in question.

Next we will check the Variance Inflation Factor (VIF) for the variables in the model, to check for multicollinearity.

#Check VIF  
vif(games.step.model.log)

## GVIF Df GVIF^(1/(2\*Df))  
## BayesAvgRating 2.840574 1 1.685400  
## LanguageEase 1.142957 1 1.069092  
## ComMinPlaytime 1.301133 1 1.140672  
## NumAlternates 1.072013 1 1.035381  
## NumExpansions 1.101242 1 1.049401  
## NumImplementations 1.104553 1 1.050977  
## Rank.strategygames 3.688557 1 1.920562  
## Rank.wargames 1.437383 1 1.198909  
## Cat.Family 1.243412 1 1.115084  
## Cat.War 1.728253 1 1.314630  
## Cat.Strategy 4.177933 1 2.043999  
## rating.score 2.009438 3 1.123343

These VIF results are all notably low- the highest being Cat.Strategy’s VIF of 4.177933- so we are confident that multicollinearity is not present in the model.

All together, these tests indicate to us that there are notable flaws in the model. The origin of our data is subpar, and our model exhibits heteroskedastisity, which in turn means we cannot assume an independent and normal error. However, given the shortcomings we have noted, we have corrected for them to the best of our ability, and we are confident that we have constructed the best possible model given the data and methods available. We remain convinced that even if the exact results of the model are not certain, the trends presented by this model are still useful, and present useful predictive factors, if not an ability to make precise predictions

### Conclusion

The intent of this model is to predict sales of board games given multiple factors, to determine the factors that will cause games to sell better in the future. Following, our model tells us that marketing (indicated by number of alternates, expansions, and implementations), quality (indicated by Bayes average rating, rating score, and the category ranks), and complexity (indicated by language ease and minimum playtime) are the best options to increase sales. What might be most interesting about this model is the negative coefficient for high rating score, indicating that the games considered the very best do not tend to sell the best. While at first counter-intuitive, these predictions appear to follow the trend that the most popular board games are popular largely because of their marketing, or their public image as popular. As an anecdotal example, most people do not consider Monopoly to be the best board game, or even one of their favorites. Despite this, it is easily one of the most recognizable and most purchased board games. Ultimately, using the predictions in this model, a potential producer of a new board game would want to heavily advertise, franchise, and re-release a reasonably complex game of good quality to maximize their sales return.