Lab 11: Modeling Basics II

Sunghwu Song

April 23, 2021

Introduction

In this lab, you will build predictive models for board game ratings. The dataset below was scraped from boardgamegeek.com and contains information on the top 4,999 board games. Below, you will see a preview of the data

```
bgg<-read.csv("bgg.csv")
bgg2=bgg[,c(4:13,15:20)]
head(bgg2)
```

```
##                                        names min_players max_players
## 1                                 Gloomhaven           1           4
## 2                  Pandemic Legacy: Season 1           2           4
## 3 Through the Ages: A New Story of Civilization           2           4
## 4                          Terraforming Mars           1           5
## 5                          Twilight Struggle           2           2
## 6                      Star Wars: Rebellion           2           4
##   avg_time min_time max_time year avg_rating geek_rating num_votes age
## 1      120       60      120 2017    8.98893     8.61858     15376  12
## 2       60       60       60 2015    8.66140     8.50163     26063  13
## 3      240      180      240 2015    8.60673     8.30183     12352  14
## 4      120      120      120 2016    8.38461     8.19914     26004  12
## 5      180      120      180 2005    8.33954     8.19787     31301  13
## 6      240      180      240 2016    8.47439     8.16545     13336  14
##
## 1 Action / Movement Programming, Co-operative Play, Grid Movement, Hand Management, Modular Board, R
## 2                                            Action Point Allowance System, Co-operative Play, Hand Mana
## 3
## 4                                                                                                 Card D
## 5                                                        Area Control / Area Influence, Campaign / I
## 6                                                                    Area Control / Area Influence, Ar
##   owned
## 1 25928
## 2 41605
## 3 15848
## 4 33340
## 5 42952
## 6 20682
##                                                                 category
## 1                                   Adventure, Exploration, Fantasy, Fighting, Miniatures
```

```
## 2                                                                    Environmental, Medical
## 3                                                  Card Game, Civilization, Economic
## 4 Economic, Environmental, Industry / Manufacturing, Science Fiction, Territory Building
## 5                                                  Modern Warfare, Political, Wargame
## 6              Fighting, Miniatures, Movies / TV / Radio theme, Science Fiction, Wargame
##                      designer weight
## 1               Isaac Childres 3.7543
## 2     Rob Daviau, Matt Leacock 2.8210
## 3               Vlaada Chvátil 4.3678
## 4              Jacob Fryxelius 3.2456
## 5 Ananda Gupta, Jason Matthews 3.5518
## 6              Corey Konieczka 3.6311
```

You will need to modify the code chunks so that the code works within each of chunk (usually this means modifying anything in ALL CAPS). You will also need to modify the code outside the code chunk. When you get the desired result for each step, change `Eval=F` to `Eval=T` and knit the document to HTML to make sure it works. After you complete the lab, you should submit your HTML file of what you have completed to Sakai before the deadline.

Board Game Analysis

Q1

There are 16 variables and we want to create some more. Create a new dataframe called *bgg3* where you use the mutate function to create the following variables:

- *duration=2018-year+1*
- *vote.per.year=num_votes/duration*
- *own.per.year=owned/duration*
- *player.range=max_players-min_players*
- *log_vote=log(num_votes+1)*
- *log_own=log(owned+1)*
- *diff_rating=avg_rating-geek_rating*

```
bgg3 <- bgg2 %>% mutate(duration=2018-year+1)%>%
  mutate(vote.per.year=num_votes/duration) %>%
  mutate(own.per.year=owned/duration) %>%
  mutate(player.range=max_players-min_players) %>%
  mutate(log_vote=log(num_votes+1)) %>%
  mutate(log_own=log(owned+1)) %>%
  mutate(diff_rating=avg_rating-geek_rating)
head(bgg3)
```

```
##                                       names min_players max_players
## 1                                 Gloomhaven           1           4
## 2                     Pandemic Legacy: Season 1           2           4
## 3 Through the Ages: A New Story of Civilization           2           4
## 4                           Terraforming Mars           1           5
## 5                           Twilight Struggle           2           2
## 6                       Star Wars: Rebellion           2           4
##   avg_time min_time max_time year avg_rating geek_rating num_votes age
```

```
## 1      120       60       120 2017    8.98893     8.61858     15376  12
## 2       60       60        60 2015    8.66140     8.50163     26063  13
## 3      240      180       240 2015    8.60673     8.30183     12352  14
## 4      120      120       120 2016    8.38461     8.19914     26004  12
## 5      180      120       180 2005    8.33954     8.19787     31301  13
## 6      240      180       240 2016    8.47439     8.16545     13336  14
##
## 1 Action / Movement Programming, Co-operative Play, Grid Movement, Hand Management, Modular Board, Ro
## 2                                            Action Point Allowance System, Co-operative Play, Hand Manag
## 3
## 4                                                                                                  Card Di
## 5                                                                   Area Control / Area Influence, Campaign / B
## 6                                                                            Area Control / Area Influence, Are
##   owned
## 1 25928
## 2 41605
## 3 15848
## 4 33340
## 5 42952
## 6 20682
##                                                                                          category
## 1                                     Adventure, Exploration, Fantasy, Fighting, Miniatures
## 2                                                                    Environmental, Medical
## 3                                                         Card Game, Civilization, Economic
## 4 Economic, Environmental, Industry / Manufacturing, Science Fiction, Territory Building
## 5                                                        Modern Warfare, Political, Wargame
## 6             Fighting, Miniatures, Movies / TV / Radio theme, Science Fiction, Wargame
##                      designer weight duration vote.per.year own.per.year
## 1              Isaac Childres 3.7543        2      7688.000     12964.00
## 2    Rob Daviau, Matt Leacock 2.8210        4      6515.750     10401.25
## 3              Vlaada Chvátil 4.3678        4      3088.000      3962.00
## 4             Jacob Fryxelius 3.2456        3      8668.000     11113.33
## 5 Ananda Gupta, Jason Matthews 3.5518       14      2235.786      3068.00
## 6             Corey Konieczka 3.6311        3      4445.333      6894.00
##   player.range  log_vote   log_own diff_rating
## 1            3  9.640628 10.163117     0.37035
## 2            2 10.168310 10.636000     0.15977
## 3            2  9.421654  9.670862     0.30490
## 4            4 10.166044 10.414543     0.18547
## 5            0 10.351437 10.667862     0.14167
## 6            2  9.498297  9.937067     0.30894
```

Question: In complete sentences, what is the purpose of adding 1 for the log transformed variables? The log of zero is undefined so it is there to prevent that from happening.

Question: In complete sentences, what is the purpose of adding 1 in the creation of the year variable? The log of zero is undefined so it is there to prevent that from happening.

Q2

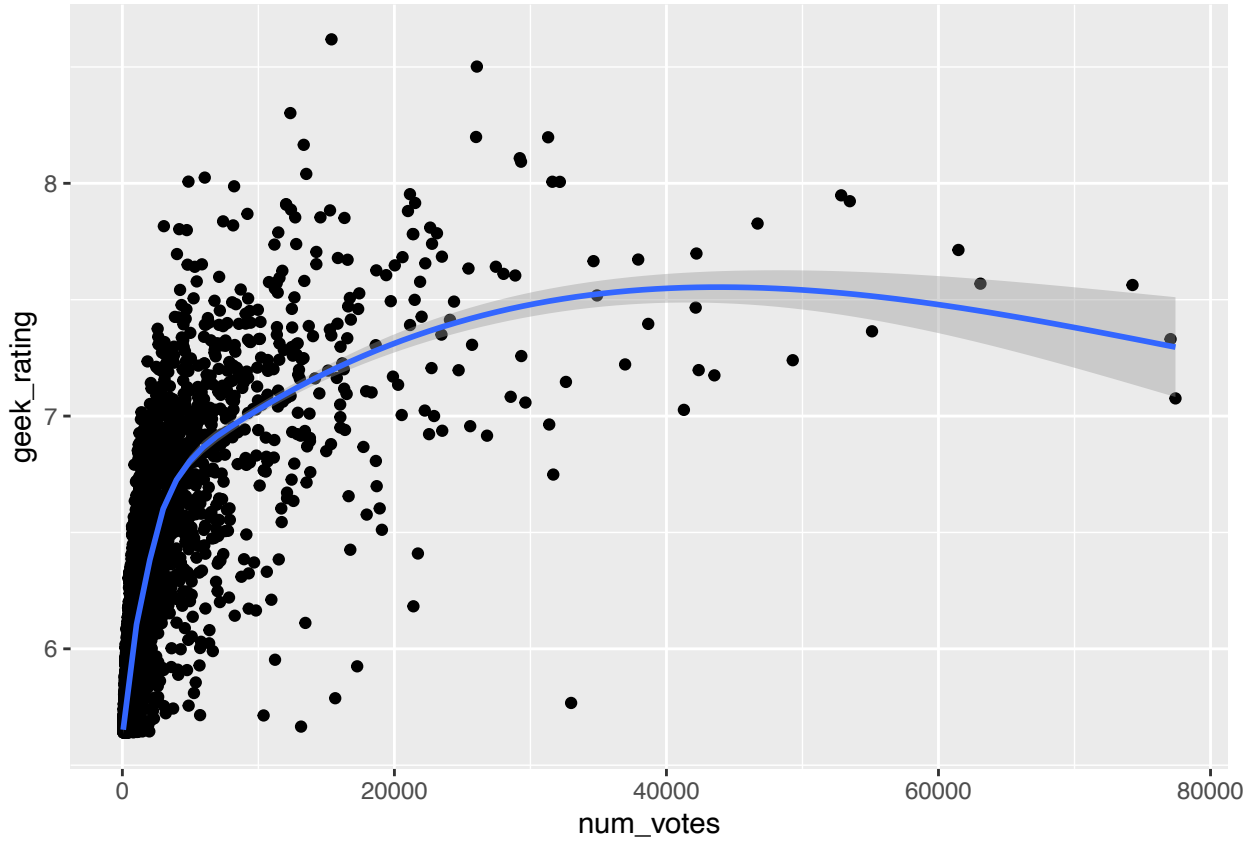
We hypothesize the geek rating increases when the number of votes increases and/or the ownership increases. Create four scatter plots showing the association with geek_rating and the following variables:

- *num_votes*

- *owned*
- *log_vote*
- *log_own*

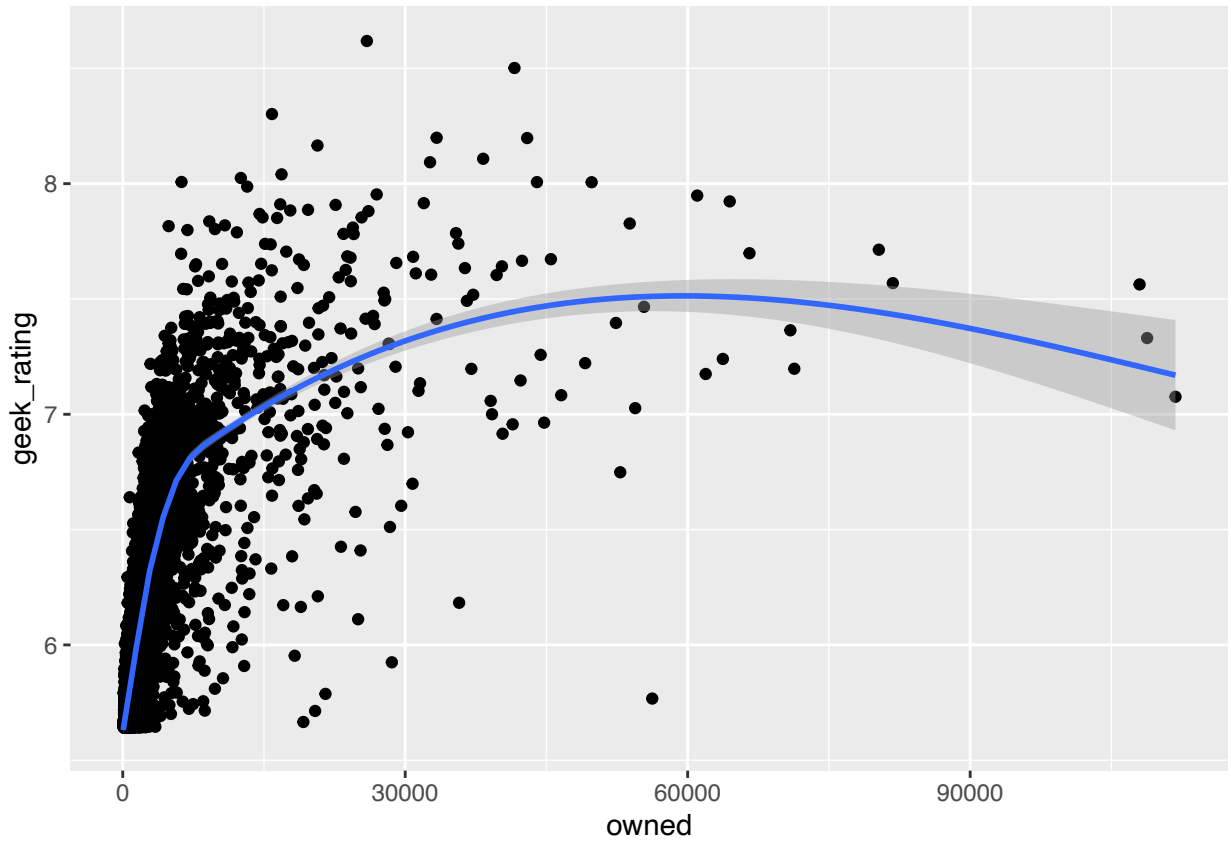
```
ggplot(bgg3,aes(x=num_votes, y=geek_rating)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



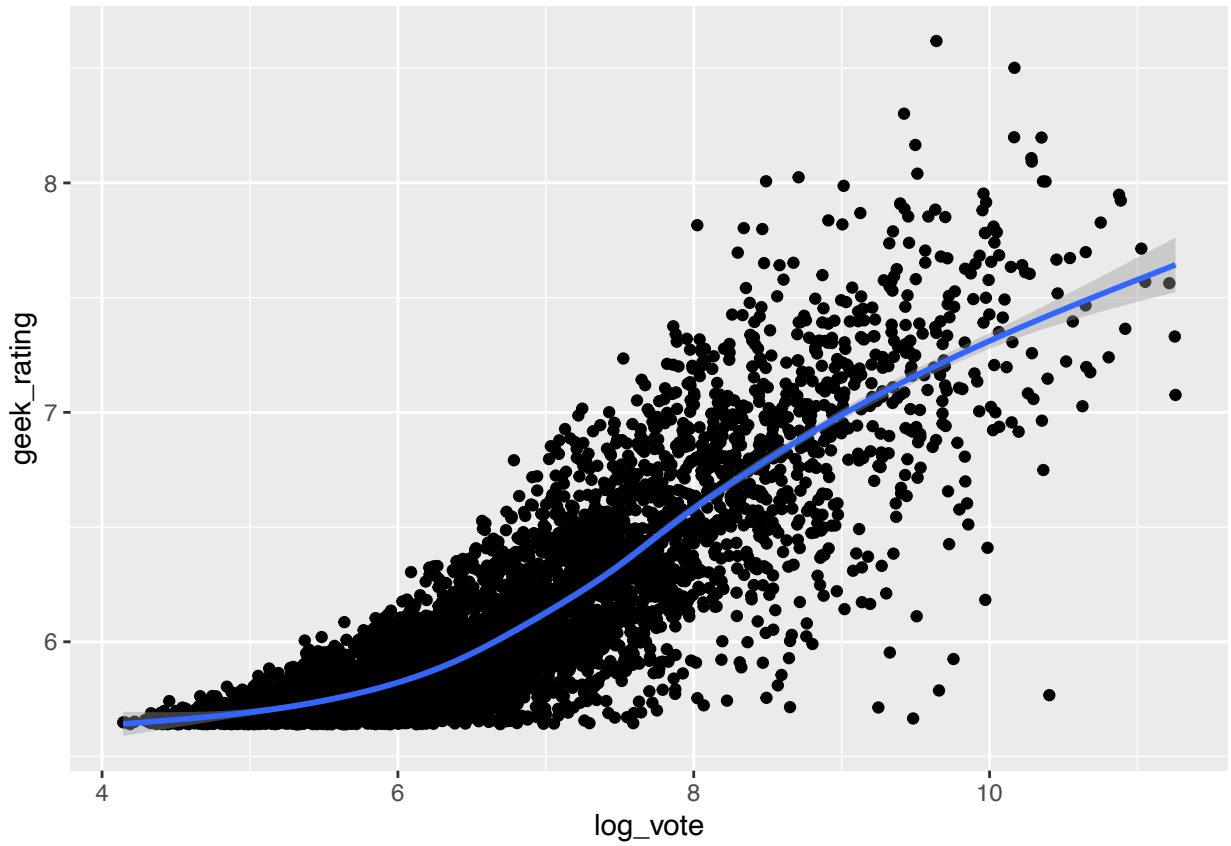

```
ggplot(bgg3,aes(x=owned, y=geek_rating)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



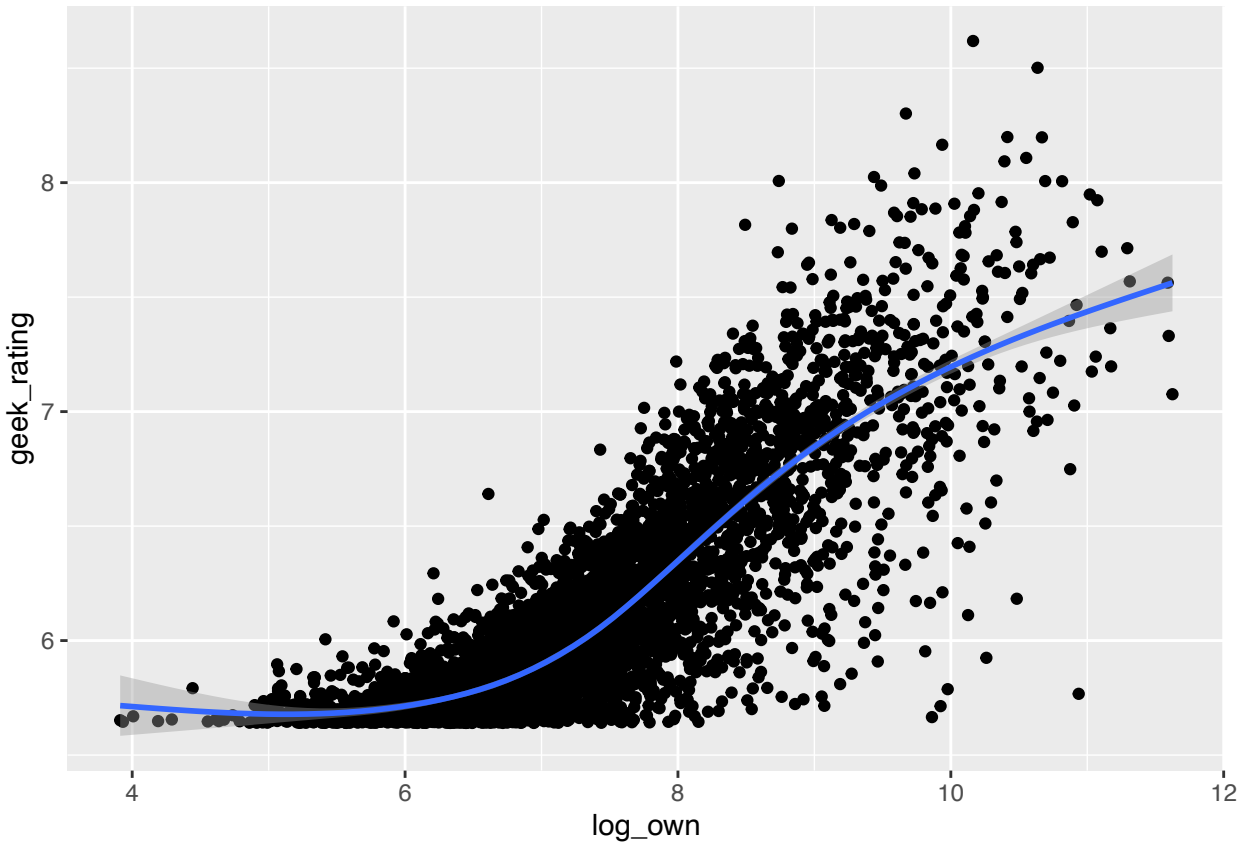

```
ggplot(bgg3,aes(x=log_vote, y=geek_rating)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
ggplot(bgg3,aes(x=log_own, y=geek_rating)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



Question: In complete sentences, describe how the relationship changes when you take the log of the independent variable.

After taking the log of the independent variable the graph changes and becomes a positive trend instead of positive then negative trend.

Q3

Randomly sample approximately 80% of the data in `bgg3` for a training dataset and the remaining will act as a test set. Call the training dataset `train.bgg` and the testing dataset `test.bgg`.

```
set.seed(111)

bgg4= bgg3 %>%
        mutate(Set=sample(c("Train","Test"), size=4999, prob = c(.8, .2), replace = TRUE))

train.bgg<-filter(bgg4,Set=="Train")
test.bgg<-filter(bgg4,Set=="Test")
```

Q4

Now, we want to fit models to the training dataset. Use the `lm()` function to create 3 model objects in R called `lm1`, `lm2`, `lm3` based on the following linear models, respectively:

```
lm1 = lm(geek_rating ~ log(num_votes),data = train.bgg)
lm2 = lm(geek_rating ~ log(owned),data = train.bgg)
lm3 = lm(geek_rating ~ log(owned) + vote.per.year + weight,data = train.bgg)
```

Q5

Add predictions and residuals for all 3 models to the test set. Create a new data frame called `test.bgg2` and give all your predictions and residuals different names. Use the `str()` function to show these variables were created

```
test.bgg2 <- test.bgg %>%
  mutate(lm1pred = predict(lm1, test.bgg),
  lm2pred = predict(lm2, test.bgg),
  lm3pred = predict(lm3, test.bgg),
  lm1resid = geek_rating - lm1pred,
  lm2resid = geek_rating - lm2pred,
  lm3resid = geek_rating - lm3pred)

str(test.bgg2)
```

```
## 'data.frame':    948 obs. of  30 variables:
##  $ names         : chr  "Mage Knight Board Game" "Star Wars: Imperial Assault" "The Voyages of Marco
##  $ min_players   : int  1 2 2 2 2 1 2 2 2 2 ...
##  $ max_players   : int  4 5 4 8 4 2 5 4 4 5 ...
##  $ avg_time      : int  240 120 100 15 240 120 45 150 120 120 ...
##  $ min_time      : int  60 60 40 15 90 60 45 75 60 120 ...
##  $ max_time      : int  240 120 100 15 240 120 45 150 120 120 ...
##  $ year          : int  2011 2014 2015 2015 2015 2014 2014 2015 2011 2012 ...
##  $ avg_rating    : num  8.12 8.14 7.96 7.79 7.88 ...
##  $ geek_rating   : num  7.92 7.85 7.74 7.7 7.68 ...
##  $ num_votes     : int  21524 14563 12780 42207 15847 4795 18671 7117 11347 17418 ...
##  $ age           : int  14 14 12 14 12 13 13 12 12 14 ...
##  $ mechanic      : chr  "Card Drafting, Co-operative Play, Deck / Pool Building, Dice Rolling, Grid M
##  $ owned         : int  31976 25367 15116 66565 24194 7787 23686 9143 13359 27719 ...
##  $ category      : chr  "Adventure, Exploration, Fantasy, Fighting" "Adventure, Fighting, Miniatures,
##  $ designer      : chr  "Vlaada Chvátil" "Justin Kemppainen, Corey Konieczka, Jonathan Ying" "Simone
##  $ weight        : num  4.24 3.28 3.19 1.33 2.65 ...
##  $ duration      : num  8 5 4 4 4 5 5 4 8 7 ...
##  $ vote.per.year: num  2690 2913 3195 10552 3962 ...
##  $ own.per.year : num  3997 5073 3779 16641 6048 ...
##  $ player.range : int  3 3 2 6 2 1 3 2 2 3 ...
##  $ log_vote      : num  9.98 9.59 9.46 10.65 9.67 ...
##  $ log_own       : num  10.37 10.14 9.62 11.11 10.09 ...
##  $ diff_rating   : num  0.2026 0.2815 0.2255 0.0915 0.2043 ...
##  $ Set           : chr  "Test" "Test" "Test" "Test" ...
##  $ lm1pred       : Named num  7.17 7.05 7 7.39 7.07 ...
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
##  $ lm2pred       : Named num  7.2 7.12 6.93 7.46 7.1 ...
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
##  $ lm3pred       : Named num  7.69 7.55 7.46 9.17 7.68 ...
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
##  $ lm1resid      : Named num  0.742 0.807 0.735 0.307 0.605 ...
```

```
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
##  $ lm2resid     : Named num  0.717 0.738 0.807 0.239 0.58 ...
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
##  $ lm3resid     : Named num  0.2293 0.29958 0.28336 -1.47153 -0.00276 ...
##   ..- attr(*, "names")= chr [1:948] "1" "2" "3" "4" ...
```

Q6

Create a function called `MAE.func()` that returns the mean absolute error of the residuals and test your function on the vector called `test`

Solution 1:

```
test=c(-5,-2,0,3,5)

MAE.func <- function(x){
  mae=mean(abs(x), na.rm=T)
  return(mae)
}

MAE.func(test)
```

```
## [1] 3
```

Q7

Use your function to calculate the mean absolute error based on the residuals to calculate the out-of-sample MAE. Make sure you display the mean absolute error from these different models in your output.

```
MAE.func(test.bgg2$lm1resid)
```

```
## [1] 0.1844415
```

```
MAE.func(test.bgg2$lm2resid)
```

```
## [1] 0.191509
```

```
MAE.func(test.bgg2$lm3resid)
```

```
## [1] 0.1702852
```

Question: Which model does the best job at predicting the geek rating of these board games? The second model is the best at predicting.

Q8

For the third model, use 10-fold cross-validation and measure the mean absolute error. Print this measure of error out.

```
dataframe = na.omit(test.bgg2) %>% crossv_kfold(10)

func = function(data, i, j) {
  return (lm(geek_rating~log_own+vote.per.year+weight, data))
}

df2 = dataframe %>%
  mutate(mod = map(train, func, i = 5, j = 2))

df2pred = df2 %>%
  mutate(pred = map2(test, mod,~augment(.y,newdata = .x))) %>%
  select(pred) %>%
  unnest(cols = c(pred))

MAE.func(df2pred$.resid)
```

```
## [1] 0.174252
```

Question: What is the absolute difference between the out-of-sample mean absolute error measured using a test set and the mean absolute error measured using cross validation? When you type your answer in complete sentences use inline R code to calculate the absolute difference and input it directly into your sentence.

The absolute difference between the two is 0.174.