Lab 12: Categorical Variables

Sunghwu Song

April 30, 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. We start by creating the following 8 new 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*

The table below gives a preview of the current data.

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

bgg3=bgg2 %>%
  mutate(duration=2018-year+1,
         vote.per.year=num_votes/duration,
         own.per.year=owned/duration,
         player.range=max_players-min_players,
         time.range=max_time-min_time,
         log_vote=log(num_votes+1),
         log_own=log(owned+1),
         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 Mana
## 3
## 4                                                                                             Card Di
## 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 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 time.range  log_vote   log_own diff_rating
## 1            3         60  9.640628 10.163117     0.37035
## 2            2          0 10.168310 10.636000     0.15977
## 3            2         60  9.421654  9.670862     0.30490
## 4            4          0 10.166044 10.414543     0.18547
## 5            0         60 10.351437 10.667862     0.14167
## 6            2         60  9.498297  9.937067     0.30894
```

Extended Board Game Analysis

Q1

My favorite mechanics in bard games are “Co-operative Play”, “Tile Placement”, “Worker Placement”, and “Card Drafting”. I want you to create the three following binary variables:

- *coop = 1 if Co-operative Play is a mechanic in the game and 0 otherwise*
- *tile = 1 if Tile Placement is a mechanic in the game and 0 otherwise*
- *worker = 1 if Worker Placement is a mechanic in the game and 0 otherwise*
- *draft = 1 if Card Drafting is a mechanic in the game and 0 otherwise*

Notice how we can use the `str_detect()` function to return TRUE if a pattern exists in a string, and then, we use `as.numeric()` to convert TRUE to a 1. We get 0 whenever the pattern is not detected in the string. This function can be vectorized like I do below in the creation of coop. Repeat this for the other variables that you are asked to create. Put this all in a new object named `bgg4`.

Look at random observations in the data to make sure everything worked.

```
x=c("Hello","Little","Buddy")
as.numeric(str_detect(x,pattern="Buddy"))
```

```
## [1] 0 0 1
```

```
as.numeric(str_detect(x,pattern="Friend"))
```

```
## [1] 0 0 0
```

```
bgg4=bgg3 %>%
  mutate(coop=as.numeric(str_detect(mechanic, pattern="Co-operative Play")),
         tile=as.numeric(str_detect(mechanic, pattern="Tile Placement")),
         worker=as.numeric(str_detect(mechanic, pattern="Worker Placement")),
         draft=as.numeric(str_detect(mechanic, pattern="Card Drafting")))
```

Q2

We want to explore the relationship with these newly created categorical (binary) variables and the geek rating. Build a linear regression model using the 4 variables created above to predict geek rating. Save the model as an object called `mod1` and print out the model using the `tidy()` function.

```
mod1=lm(geek_rating~coop+tile+worker+draft, bgg4)
tidy(mod1)
```

```
## # A tibble: 5 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   6.02     0.00792    760.   0
## 2 coop          0.165    0.0249       6.63 3.69e-11
## 3 tile          0.0897   0.0216       4.16 3.24e- 5
## 4 worker        0.273    0.0269      10.2  5.09e-24
## 5 draft         0.233    0.0201      11.6  9.27e-31
```

Question: In a complete sentence, interpret the intercept of the model. Reference the estimate of the intercept and explain what it represents in way a common person would understand. The intercept (6.01906827) shows the geek rating when none of the four mechanics are present.

Question: In a complete sentence, explain the estimated coefficient for the Worker Placement variable. Reference the estimated value and explain in a way that a common person would understand. The estimated coeff (0.27286769) shows how much the geek rating would increase when the worker placement mechanic is present.

Q3

Use the `data_grid()` function to create an object called `GRID` that contains all combinations of four categorical (binary) variables. Then, use the predict function to get the predicting geek ratings for each combination of these four variables.

In the end, use the `head()` function to print out the grid of fitted values along with their confidence intervals.

```
GRID=bgg4 %>%
        data_grid(
          coop,
          tile,
          worker,
          draft
        )
?predict()
GRID2=cbind(GRID,predict(mod1, GRID, interval="confidence"))

head(GRID2)
```

```
##   coop tile worker draft      fit      lwr      upr
## 1    0    0      0     0 6.019068 6.003544 6.034593
## 2    0    0      0     1 6.252550 6.214928 6.290172
## 3    0    0      1     0 6.291936 6.240220 6.343652
## 4    0    0      1     1 6.525418 6.465115 6.585721
## 5    0    1      0     0 6.108812 6.068210 6.149414
## 6    0    1      0     1 6.342294 6.289566 6.395022
```

Q4

From the last model, it seems that Worker Placement and Card Drafting games typically having higher geek ratings than games without these mechanics. Create a model called `mod2` that only has these two variables in the model along with the interaction term between these two variables. Use the `tidy()` function to display the model.

```
mod2=lm(geek_rating~worker+draft, bgg4)
tidy(mod2)
```

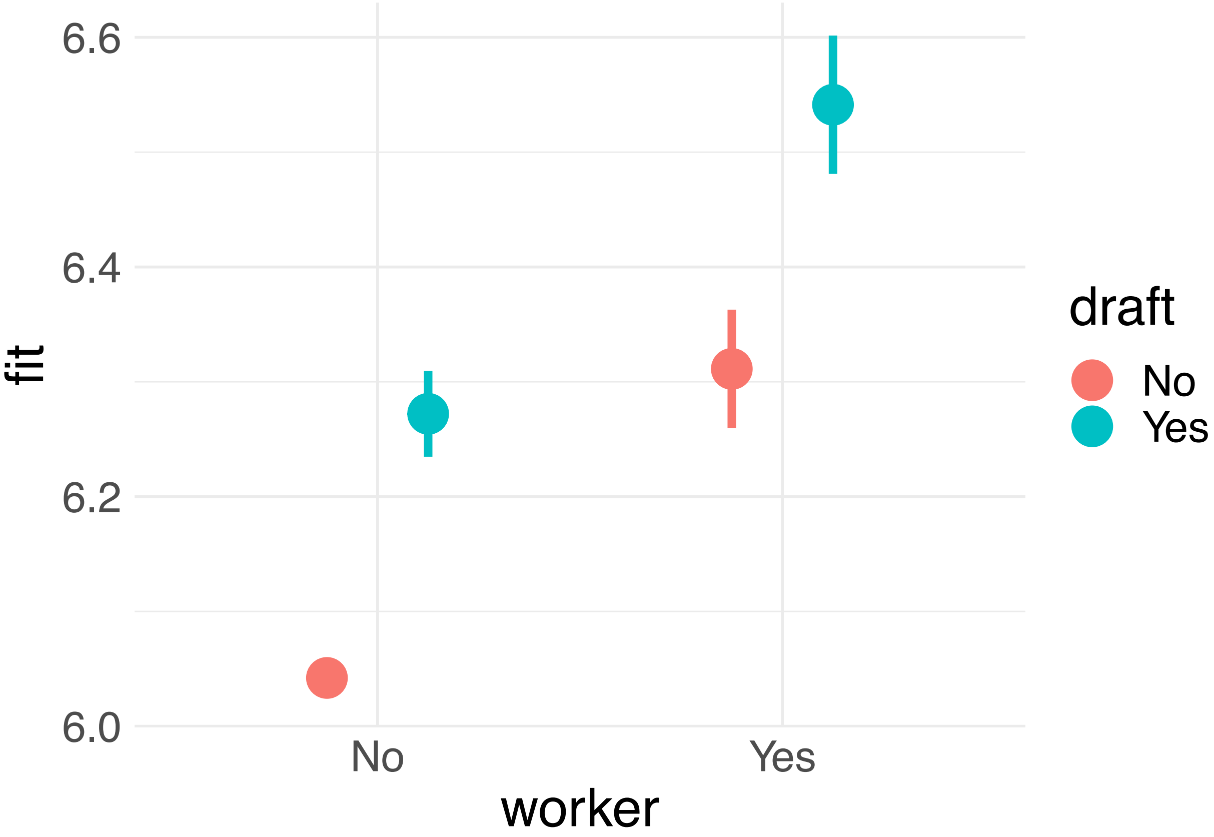
```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    6.04    0.00732    825.   0
## 2 worker         0.269   0.0270       9.98 3.00e-23
## 3 draft          0.230   0.0202      11.4  1.21e-29
```

Q5

Use the `data_grid()` function as in Q3 for this model named `mod2`, and then after obtaining the fitted values and confidence intervals, plot the confidence intervals as seen in lecture. Place `worker` on the x-axis and use `draft` to modify the color. Find a way to change the values of "0" and "1" to "No" and "Yes". I recommend using the `ifelse()` function.

```
GRID3=bgg4 %>%
        data_grid(
          worker=unique(worker),
          draft=unique(draft)
        )
GRID4=cbind(GRID3,predict(mod2,GRID3,interval="confidence"))
GRID4$worker <- ifelse(GRID4$worker==0, "No", "Yes")
GRID4$draft <- ifelse(GRID4$draft==0, "No", "Yes")

ggplot(GRID4) +
  geom_pointrange(aes(x=worker, y=fit, ymin=lwr, ymax=upr, color=draft),
                  position=position_dodge(width=0.5),size=1.5) +
  theme_minimal()+theme(text=element_text(size=20))
```



Question: In complete sentences, why is the confidence interval for board games where there is no worker placement and no card drafting so small that only a dot appears in the picture? The confidence interval is probably so small because there are less variables with both worker and draft mechanics involved.

Q6

Below I create a dataset called `final.bgg` that removes variables that were used to create other variables and removes some other categorical variables.

```
final.bgg=bgg4 %>%
  select(-max_players,-max_time,-year,-avg_rating,-num_votes,-owned,-category,-mechanic,-designer,-name
head(final.bgg)
```

```
##   min_players avg_time min_time geek_rating age weight duration vote.per.year
## 1           1      120       60     8.61858  12 3.7543        2      7688.000
## 2           2       60       60     8.50163  13 2.8210        4      6515.750
## 3           2      240      180     8.30183  14 4.3678        4      3088.000
## 4           1      120      120     8.19914  12 3.2456        3      8668.000
## 5           2      180      120     8.19787  13 3.5518       14      2235.786
## 6           2      240      180     8.16545  14 3.6311        3      4445.333
##   own.per.year player.range time.range  log_vote   log_own diff_rating coop
## 1     12964.00            3         60  9.640628 10.163117     0.37035    1
## 2     10401.25            2          0 10.168310 10.636000     0.15977    1
## 3      3962.00            2         60  9.421654  9.670862     0.30490    0
## 4     11113.33            4          0 10.166044 10.414543     0.18547    0
## 5      3068.00            0         60 10.351437 10.667862     0.14167    0
## 6      6894.00            2         60  9.498297  9.937067     0.30894    0
##   tile worker draft
## 1    0      0     0
## 2    0      0     0
## 3    0      0     1
## 4    1      0     1
## 5    0      0     0
## 6    0      0     0
```

Create a new dataset called `final.bgg2` where you create a new variable called `Favorite` that equals 1 if the the board game has at least one of my four favorite mechanics, and then remove the four variables we created named `coop`, `tile`, `worker`, and `draft`. Then use the `str()` function to show `final.bgg2`

```
final.bgg2=final.bgg %>%
              mutate(Favorite=ifelse(coop | tile | worker | draft,1,0)) %>%
              select(-coop,-tile,-worker,-draft)
str(final.bgg2)
```

```
## 'data.frame':    4999 obs. of  15 variables:
##  $ min_players  : int  1 2 2 1 2 2 1 2 2 1 ...
##  $ avg_time     : int  120 60 240 120 180 240 115 150 150 1000 ...
##  $ min_time     : int  60 60 180 120 120 180 90 60 75 5 ...
##  $ geek_rating  : num  8.62 8.5 8.3 8.2 8.2 ...
##  $ age          : int  12 13 14 12 13 14 14 12 12 14 ...
##  $ weight       : num  3.75 2.82 4.37 3.25 3.55 ...
##  $ duration     : num  2 4 4 3 14 3 3 7 3 2 ...
##  $ vote.per.year: num  7688 6516 3088 8668 2236 ...
##  $ own.per.year : num  12964 10401 3962 11113 3068 ...
##  $ player.range : int  3 2 2 4 0 2 4 3 2 3 ...
##  $ time.range   : int  60 0 60 0 60 60 25 90 75 995 ...
##  $ log_vote     : num  9.64 10.17 9.42 10.17 10.35 ...
##  $ log_own      : num  10.16 10.64 9.67 10.41 10.67 ...
##  $ diff_rating  : num  0.37 0.16 0.305 0.185 0.142 ...
##  $ Favorite     : num  1 1 1 1 0 0 0 0 0 1 ...
```

Question: In a complete sentence, what percent of games in `final.bgg` have at least one my four favorite mechanics. Use inline R code to insert your answer directly into your sentence. You can calculate this directly from the new variable named Favorite. Use the round function to round your percentage to 2 decimal places.

There are 32.83 of games with at least 1 of 4 mechanics.

Q7

Build 3 different logistic regression models called “Model_1”, “Model_2” and “Model 3” to classify a game as a favorite of Dr. Mario. Each of the three models should have five different variables in them. You can pick from any of the variables in final.bgg2. You can have variables that are in multiple models, but none of the three models should have the same exact 5 variables.

Then, I want you to build a table that shows the name of the models and the proportion of time that the model accurately classified a board game as one of my favorites. You can calculate this measure of accuracy in the original dataset final.bgg2. There is no need here to split the data into training and testing datasets. Then, print out this entire table.

```
Model_1=glm(Favorite~min_players+avg_time+min_time+geek_rating+age, family = "binomial", final.bgg2)
Model_2=glm(Favorite~weight+duration+vote.per.year+own.per.year+weight, family = "binomial", final.bgg2
Model_3=glm(Favorite~player.range+time.range+log_vote+log_own+diff_rating, family = "binomial", final.bg

final.bgg3=final.bgg2 %>%
              mutate(p1=predict(Model_1),
                     p2=predict(Model_2),
                     p3=predict(Model_3),
                     S1=ifelse(p1<=0,0,1),
                     S2=ifelse(p2<=0,0,1),
                     S3=ifelse(p3<=0,0,1))

Model_1.accuracy=mean(final.bgg3$S1)
Model_2.accuracy=mean(final.bgg3$S2)
Model_3.accuracy=mean(final.bgg3$S3)

tibble(Model=c("Model_1","Model_2","Model_3"),Accuracy=c(Model_1.accuracy,Model_2.accuracy,Model_3.accu
```

```
## # A tibble: 3 x 2
##   Model   Accuracy
##   <chr>      <dbl>
## 1 Model_1   0.126
## 2 Model_2   0.0622
## 3 Model_3   0.0430
```

Question: In complete sentences, describe your best model and how accurately it classified board games as a favorite of mine. Talk about what variables are in your best model. Talk about how good the best model was relative to the other 3 models. The best model seems to be model 1 with the highest accuracy. It had the variables min_players, avg_time, min_time, geek_rating, and age. Model 1 in comparison to model 2 and 3 had twice, and three times, respectively, the accuracies.