684 Midterm Project

Yukun He

2017/12/19

===Intro===

For this project, I want to focus on the video game datasets. I am a huge fan of all kinds of video games, such that I would like to devote myself to the video game industry in the future. I think this project provides a good chance for me to start my analyze on video games related datasets. Video games are generated by computer programming and are based on large datasets to keep the pseudorandomness of themselves. Also, players' own unique play style can come up with interesting statistics for us to analyze. Before choosing this CS:GO dataset, I also tried to explore several different datasets in the field of video game. Throughout the exploration of different dataset, I think this one can be interesting to interpret.

Counter-Strike: Global Offensive (CS:GO) is a multiplayer first-person shooter (FPS) video game on the PC platform. As an online real-time competitive game, CS:GO generates numerous data in many aspects, result from a large amount of players' unique gaming styles. Firstly, lets take a look at the components of this dataset:

- MAP: Name of the map
- Team: Randomly generated team ids, players on the same team have the same team id
- Player: Randomly generated player ids, the same player will have the same player id
- Kills: Number of kills that a particular player score in that game
- Deaths: Number of deaths that a particular player is killed in that game
- ADR: Average damage per round
- KAST.: Percentage of rounds in which the player either had a kill, assist, survived or was traded
- Rating: The overall evaluation of a player based on his/her performance in that game
- Rating.Type : Types of rating players
- MatchID: Randomly generated match ids, players in the same match will have the same match id

From the dataset, we can see that the Rating column provides overall evaluations of players in different games. The higher the rating, the better the performance. In this situation, I think it can be interesting to explore the elements that bring up a well-performed player, based on the different groups of Maps each game takes place in. Since different maps have completely different gaming tactics and landscapes, I

think maps will have huge effect on players' performance. Each player will have maps that he/she is good at and bad at. Professional CS:GO teams can use this analysis to change their training focus of pro players and their map choosing strategies in professional matches. And other players on the internet can find out the aspects they need to improve in the future gaming, in order to get better rating.

===Data Cleaning (Concerns)===

During data cleaning, I think the team id, player id, and match id columns are irrelavent to my analysis, since my focus is on the Rating. Hence I would consider players with same player ids in different matches as different players. Also, I find out that when the Rating Type is 1, the ADR and KAST. columns are all N/As. If I choose to omit the N/As in the dataset, the groups of Maps will be decreased from 14 to 9, which does not satisfy the requirement of minimum groups for this project. In this situation, I have to exclude those two columns from my modeling, and explore the effect from number of kills and deaths. I think Kills and Deaths are two major effects, so that the modeling can still be convincing. However, without other possible elements, this modeling lacks some fun of exploration. But I still believe with current cleaned dataset, I can come up with some useful models.

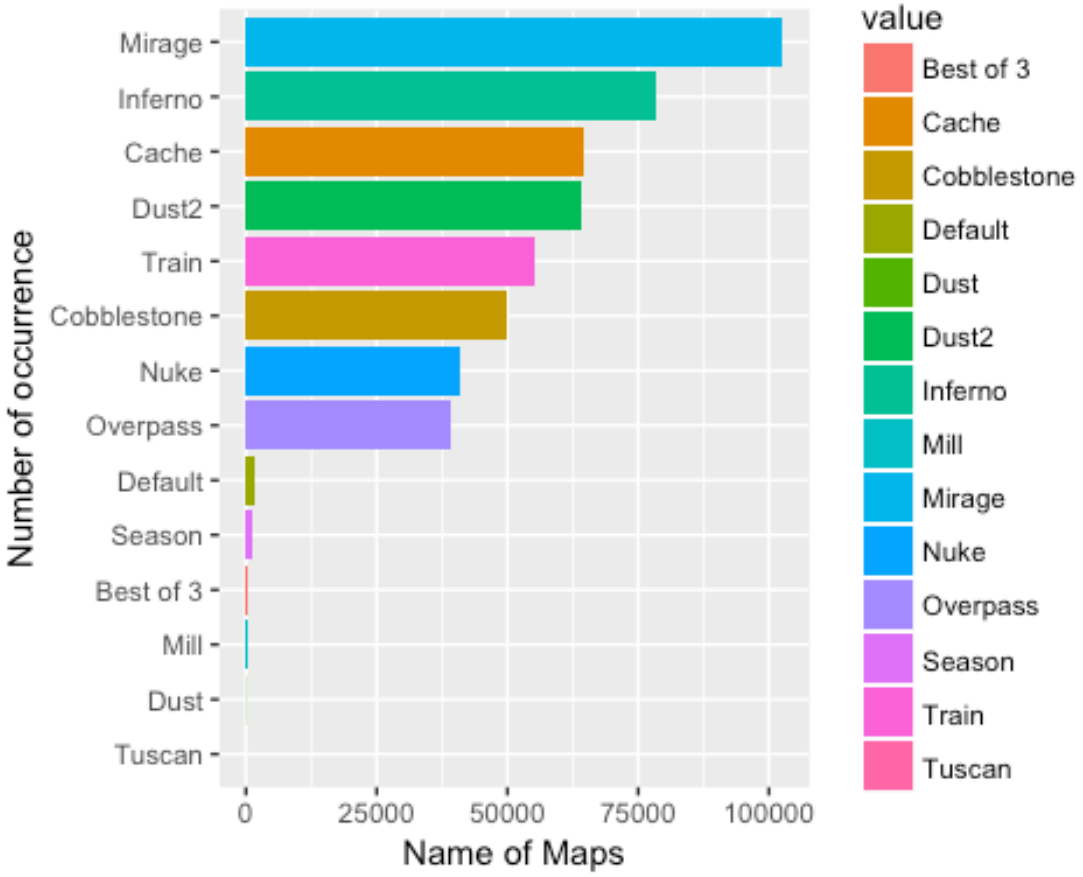
```
##
##   Best of 3       Cache Cobblestone     Default        Dust       Du
st2
##         400       64565       49654        1524          90       64
288
##     Inferno        Mill      Mirage        Nuke    Overpass      Sea
son
##       78503         120      102579       41119       39012        1
171
##       Train      Tuscan
##       55022          10
```

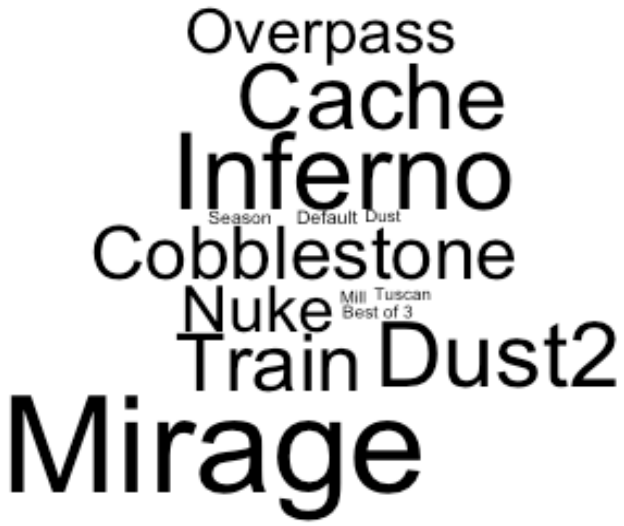
===EDAs===

In this section, I want to present an overview of the dataset I choose via EDAs. Also, I expect to find additional interesting elements that I can mention and explore in my modeling.

PART 1

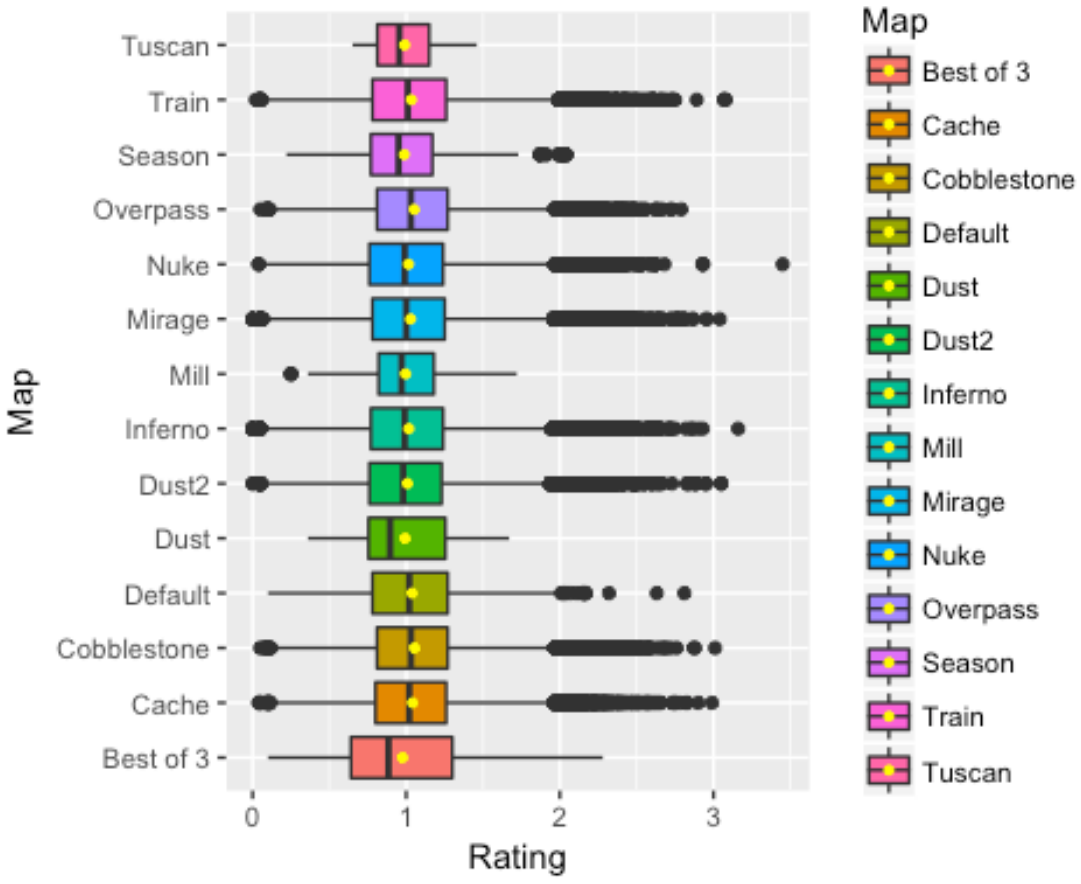
For the first part of EDAs, we can see the number of occurence of each map in our dataset. There are 9 leading maps that people plays the most, and 5 maps people relatively seldomly play. From the wordcloud we can more directly see that the bolder and bigger words are the maps players choose more. Here comes a question that I want to include in my models: Does higher occurrence of maps represents higher proficiency of players in these maps, which results in higher rating?


Mirage
Inferno
Cache
Dust2
Train
Cobblestone
Nuke
Overpass
Default
Season
Best of 3
Mill
Dust
Tuscan
Number of occurrence
0
25000
50000
75000
100000
Name of Maps
value
Best of 3
Cache
Cobblestone
Default
Dust
Dust2
Inferno
Mill
Mirage
Nuke
Overpass
Season
Train
Tuscan


Overpass
Cache
Inferno
Season
Default
Dust
Cobblestone
Nuke
Mill
Tuscan
Best of 3
Train
Dust2
Mirage

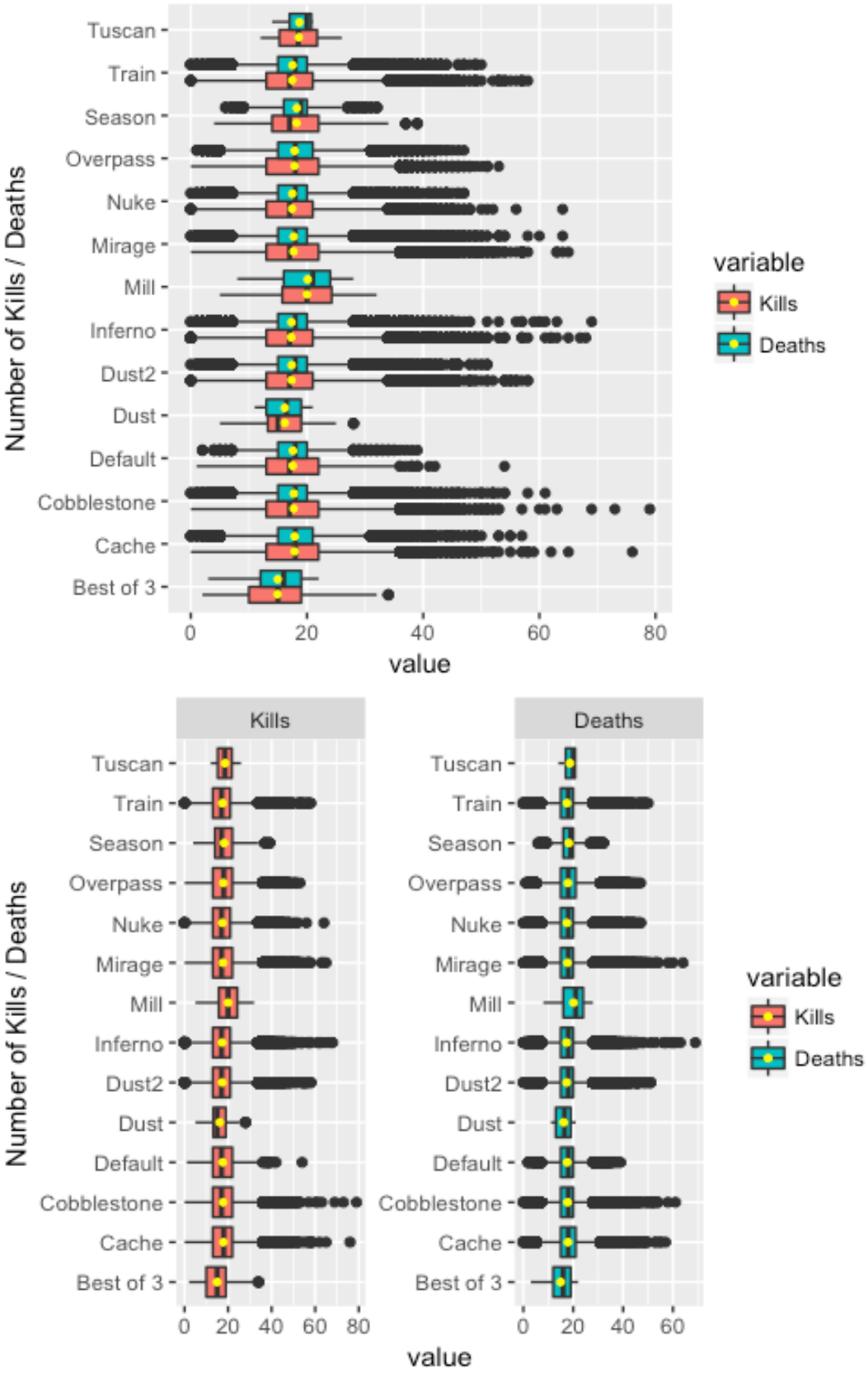
Part 2

In the second part of the EDAs. I seek to use boxplot of rating based on different maps to figure out the mean values of rating for those popular maps. However, the difference of each map does not vary too much. I think regression is needed to distinguish the subtle differences.



Part 3

For the third part of EDAs, we can see the boxplots for Kills and Deaths, grouped by Maps. We observe a large number of outliers. I think this is mainly because of the existence of elite players and rookie players, which results in a huge difference of the max and min number of Kills and Deaths. Rookie players tend to die more and kill less, while elite players tend to die less and kill more. From the first boxplot of this part, we can see the yellow dots representing average kills is slightly lower than the average deaths in every map. And when we compare the kills and deaths separately among maps, the kills and deaths each does not vary too much. Does this means that number of kills and number of deaths have exactly opposite effect on the rating? We should try to solve the question in the modeling.


Number of Kills / Deaths
Tuscan
Train
Season
Overpass
Nuke
Mirage
Mill
Inferno
Dust2
Dust
Default
Cobblestone
Cache
Best of 3
0
20
40
60
80
value
variable
Kills
Deaths
Kills
Deaths

===Fixed Effects===

In this part, I fitted 4 models with fixed effects, by adding group factors and interactions. The R-square for each model does not differ too much, however, considering a large dataset we have, I think R-square cannot be used to judge the models. Furthermore, I compare the AIC values for each model, and I find out that model 3 and model 4 have lower AIC values. While model 3 consists of less variables, I think model 3 is a better one.

Let's interpret the coefficients of model 3: - For intercept, since Deaths and Kills cannot be 0 and there must be one map selected, we do not interpret the intercept alone. - For the coefficient of kills, every one kill for the player in one match will increase 0.05860 points to the basic rating 0.7136. - For the coefficient of deaths, every one death for the player in one match will decrease 0.03216 points to the basic rating 0.7136. - For the coefficient of interaction between kills and deaths, every one kill or death for the player in one match will decrease the effect on basic rating 0.7136 from deaths or kills by 0.0006134. - For the coefficients of the group of maps, different maps will increase the rating by 0.05688, 0.06984, 0.05706, 0.005472, 0.02421, 0.03454, 0.01150, 0.04363, 0.03133, 0.06766, 0.0002677, and 0.05081, except the map Tuscan, which decreases the rating by 0.0009764. This answer the question in EDA part. The popular maps does not warrant higher rating.

As we can see, the coefficients are all quite small, but the mean values for kills and deaths are both around 20. Hence the effect on overall rating for a player is still relatively large.

```
## [1] 0.8992262

## [1] 0.9038144

## [1] 0.9055133

## [1] 0.9056264

##                  df       AIC
## csgo_fixed_reg1  4 -749133.7
## csgo_fixed_reg2  5 -772340.3
## csgo_fixed_reg3 18 -781190.1
## csgo_fixed_reg4 44 -781734.4

##
## Call:
## lm(formula = Rating ~ Kills + Deaths + (Kills * Deaths) + factor(Ma
p),
##     data = data_csgo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.72508 -0.07295 -0.01265  0.06118  1.70409
##
```

```
## Coefficients:
##                            Estimate Std. Error  t value Pr(>|t|)
## (Intercept)              7.136e-01  5.756e-03  123.985  < 2e-16 ***
## Kills                    5.860e-02  8.013e-05  731.302  < 2e-16 ***
## Deaths                  -3.216e-02  8.829e-05 -364.242  < 2e-16 ***
## factor(Map)Cache         5.688e-02  5.541e-03   10.267  < 2e-16 ***
## factor(Map)Cobblestone   6.984e-02  5.546e-03   12.593  < 2e-16 ***
## factor(Map)Default       5.706e-02  6.206e-03    9.195  < 2e-16 ***
## factor(Map)Dust          5.472e-03  1.289e-02    0.425    0.671
## factor(Map)Dust2         2.421e-02  5.540e-03    4.370 1.24e-05 ***
## factor(Map)Inferno       3.454e-02  5.537e-03    6.237 4.47e-10 ***
## factor(Map)Mill          1.150e-02  1.150e-02    1.000    0.317
## factor(Map)Mirage        4.363e-02  5.534e-03    7.884 3.19e-15 ***
## factor(Map)Nuke          3.133e-02  5.550e-03    5.645 1.66e-08 ***
## factor(Map)Overpass      6.766e-02  5.552e-03   12.187  < 2e-16 ***
## factor(Map)Season        2.677e-04  6.398e-03    0.042    0.967
## factor(Map)Train         5.081e-02  5.543e-03    9.166  < 2e-16 ***
## factor(Map)Tuscan       -9.764e-04  3.536e-02   -0.028    0.978
## Kills:Deaths            -6.134e-04  3.957e-06 -155.009  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1104 on 498040 degrees of freedom
## Multiple R-squared:  0.9055, Adjusted R-squared:  0.9055
## F-statistic: 2.983e+05 on 16 and 498040 DF,  p-value: < 2.2e-16
```

===Random Effects===

In this part, I explored and established several multilevel models with group of Maps. Furthermore, I plot the residual plots. By comparing AIC and range of binned residual plot for each model, I choose one better fitted model. I also plot the simulated fixed effects and the simulated random effects on a ggplot2 chart for the model I choose.

Model 1

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Rating ~ Kills + (1 | Map)
##    Data: data_csgo
##
## REML criterion at convergence: 8934.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -8.1119 -0.6130 -0.1936  0.4514  8.9939
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Map      (Intercept) 0.001571 0.03963
##  Residual             0.059598 0.24413
```

```
## Number of obs: 498057, groups:  Map, 14
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept) 3.032e-01  1.124e-02    27.0
## Kills       4.089e-02  5.374e-05   760.9
##
## Correlation of Fixed Effects:
##       (Intr)
## Kills -0.084
```

Firstly, I fit a model to use fixed effect Kills (number of kills) to predict Rating (overall rating), controlling for by-map variability. From the summary we can see the variance of Map, and the variance of the Residual which stands for the variability that's not due to Map. This means that there are still some more influential factors outside of our dataset other than Map which will effect the overall rating. Since our dataset do not contain those factors, we do not want to further explore this aspect.

For the fixed effect part, we can see every one increase in Kills will result in 0.04089 increase in Rating.

Model 2

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Rating ~ Kills + Deaths + (1 | Map)
##    Data: data_csgo
##
## REML criterion at convergence: -757622.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -9.3667 -0.6625 -0.1009  0.5607 13.5743
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  Map      (Intercept) 0.0005685 0.02384
##  Residual             0.0127876 0.11308
## Number of obs: 498057, groups:  Map, 14
##
## Fixed effects:
##               Estimate Std. Error t value
## (Intercept)  9.873e-01  6.682e-03   147.8
## Kills        4.678e-02  2.527e-05  1851.1
## Deaths      -4.489e-02  3.324e-05 -1350.2
##
## Correlation of Fixed Effects:
##        (Intr) Kills
## Kills  -0.051
## Deaths -0.076 -0.173
```

In Model 2, I add Deaths as the additional fixed effect. Compare to the first model I fitted, the variation that's associated with Map dropped considerably. This is because the variation that's due to number of deaths was confounded with the variation that's due to maps. The Residual variance drops either. By adding Deaths, I have changed a large amount of the variance that was previously in the random effects component to the fixed effects component.

For the fixed effect part, we can see the intercept is significantly higher than that from model 1. As now we take number of deaths into consideration, the effect from the number of kills is enhanced, but not too much. Now the effects of Kills and Deaths have almost the opposite values.

```
## $Map
##             (Intercept)      Kills
## Best of 3     0.3602529 0.04088634
## Cache         0.3119708 0.04088634
## Cobblestone   0.3303073 0.04088634
## Default       0.3203917 0.04088634
## Dust          0.3226751 0.04088634
## Dust2         0.2988573 0.04088634
## Inferno       0.3098956 0.04088634
## Mill          0.2079036 0.04088634
## Mirage        0.3051093 0.04088634
## Nuke          0.3021026 0.04088634
## Overpass      0.3227793 0.04088634
## Season        0.2449308 0.04088634
## Train         0.3198947 0.04088634
## Tuscan        0.2878381 0.04088634
##
## attr(,"class")
## [1] "coef.mer"
```

```
## $Map
##             (Intercept)      Kills      Deaths
## Best of 3     0.9516882 0.04678119 -0.04488521
## Cache         1.0092511 0.04678119 -0.04488521
## Cobblestone   1.0221759 0.04678119 -0.04488521
## Default       1.0064109 0.04678119 -0.04488521
## Dust          0.9667941 0.04678119 -0.04488521
## Dust2         0.9763100 0.04678119 -0.04488521
## Inferno       0.9859659 0.04678119 -0.04488521
## Mill          0.9669545 0.04678119 -0.04488521
## Mirage        0.9961186 0.04678119 -0.04488521
## Nuke          0.9837838 0.04678119 -0.04488521
## Overpass      1.0200782 0.04678119 -0.04488521
## Season        0.9556594 0.04678119 -0.04488521
## Train         1.0028016 0.04678119 -0.04488521
## Tuscan        0.9786828 0.04678119 -0.04488521
##
```

```
## attr(,"class")
## [1] "coef.mer"
```

Also, when we take a look at the group coefficients of two models, we can see the intercept for each map is increased considerably. The effect from the maps is stronger when we consider the number of deaths as a fixed effect.

Model 3 and 4

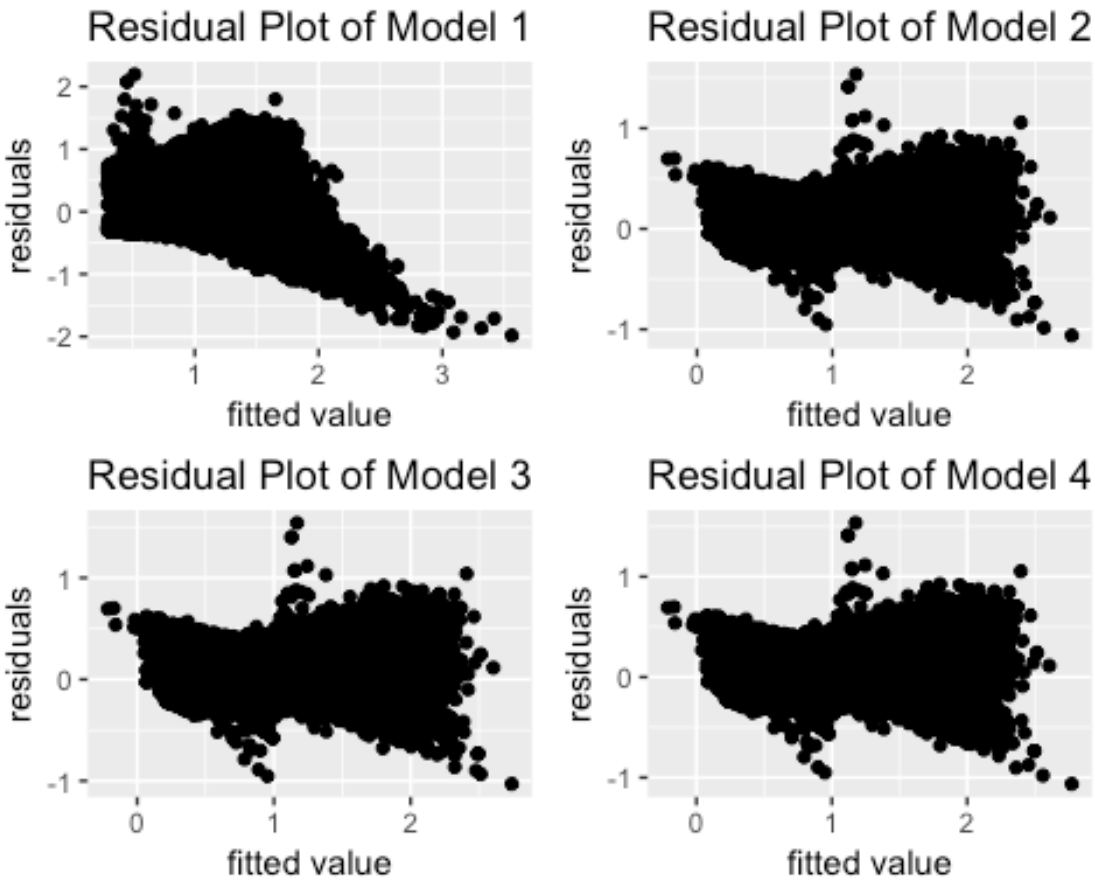
```
## $Map
##             (Intercept)      Kills      Deaths
## Best of 3     0.8810105 0.05134704 -0.04486833
## Cache         1.0181949 0.04626310 -0.04486833
## Cobblestone   1.0363772 0.04596210 -0.04486833
## Default       0.9982543 0.04722189 -0.04486833
## Dust          0.9484436 0.04793021 -0.04486833
## Dust2         0.9648726 0.04742414 -0.04486833
## Inferno       0.9776563 0.04724499 -0.04486833
## Mill          0.9959489 0.04551576 -0.04486833
## Mirage        0.9973871 0.04669260 -0.04486833
## Nuke          0.9646236 0.04786272 -0.04486833
## Overpass      1.0357750 0.04588457 -0.04486833
## Season        0.9555052 0.04679778 -0.04486833
## Train         1.0048351 0.04664788 -0.04486833
## Tuscan        0.9735828 0.04723293 -0.04486833
##
## attr(,"class")
## [1] "coef.mer"

## $Map
##             (Intercept)      Kills      Deaths
## Best of 3     0.9487372 0.04678001 -0.04465619
## Cache         1.0061470 0.04678001 -0.04471056
## Cobblestone   1.0242973 0.04678001 -0.04500376
## Default       1.0273449 0.04678001 -0.04605547
## Dust          0.9570775 0.04678001 -0.04432566
## Dust2         0.9747496 0.04678001 -0.04479410
## Inferno       0.9871599 0.04678001 -0.04495283
## Mill          0.9503823 0.04678001 -0.04409841
## Mirage        0.9950258 0.04678001 -0.04482236
## Nuke          0.9882577 0.04678001 -0.04513969
## Overpass      1.0137068 0.04678001 -0.04452815
## Season        0.9266255 0.04678001 -0.04331425
## Train         1.0074977 0.04678001 -0.04515214
## Tuscan        0.9729653 0.04678001 -0.04458459
##
## attr(,"class")
## [1] "coef.mer"
```

These two random slope models take a long time to run. As we can see from the coefficients of the model 3 and model 4, the slope for Kills and Deaths are different for each different map we choose.

The slopes for Kills in model 3 are always positive and that many of the values are quite similar to each other. This means that despite individual variation, there is also consistency in how number of kills affects the overall rating. For all of the players, the rating tends to increase when score more number of kills in the game, but for some players it goes up slightly more.

The slopes for Deaths in model 4 are always negative and that many of the values are quite similar to each other. This means that despite individual variation, there is also consistency in how number of deaths affects the overall rating. For all of the players, the rating tends to decrease when score more number of deaths in the game, but for some players it goes down slightly more.

```
##                 df         AIC
## csgo_rand_reg1   4    8942.839
## csgo_rand_reg2   5 -757612.078
## csgo_rand_reg3   7 -758161.413
## csgo_rand_reg4   7 -757625.920
```



Then I generate the AIC for the 4 models in Random Effect part, also with their residual plots. From the AIC, model 2 and 4 have similar lower AIC but model 2 has fewer variables. Then we compare the residual plots. Both plots are balanced and with similar range of distribution. Hence, in the random effect part, model 2 is better.

===Conclusion & Limitation===

Through out the coding and analysis, I think I have explored some aspects and interesting facts about this dataset about CS:GO. The questions I mentioned at the beginning are also solved through the process. However, there are still improvements to be made in the future. Firstly, I failed to find a proper way to deal with the N/As in the dataset, such that I lost two columns of data. I could have done more interesting exploration with the help of those two columns. Furthermore, I only used AIC and residual plots to compare the effectiveness of different models. I noticed there are more ways to compare models, like: BIC and cross-calidation. I think in this winter break I should try to master more methods to compare models.

For the limitation, I think there is still something I need to improve in my models. As we can see, the residual plots for my random effect models are not pretty good, with skewness. I tried to take log to relieve this issue but failed. Due to the time limit of this project, I cannot perform further improvement of my work. But I will keep working with the improvement of my models.