# What Stats In Player Unknown’s Battleground Are Important In Determining A Players Probability In Winning

## Introduction

PLAYERUNKNOWN’S BATTLEGROUNDS (PUBG) since its release has become the largest played game of the year, shattering many game records, while still in early access. PUBG recently had a full release and reached a new record of 3 million concurrent players.

## The Problem

The PUBG community often obsess over their statistics. Having the highest kill death average is almost trophy status. But is the number of kills or your kill death ratio the only indicator of your success in the game? Or are there other variables just as important at predicting a players success at PUBG?

Our goal of analysis is to see which statistics are important in determining a player’s probability of winning. Also, to pick out which strategies could be important in increasing your chances of winning.

## Data Set

This data set contains all regions around the world, and each of the three game modes (solo, duo, squad). There are 87,898 players and 150 variables.

Some of the dependent variabless that are in the set include.

* solo\_KillDeathRatio Which is the number of total kills divided by total deaths.
* solo\_RoundsPlayed
* solo\_Top10s
* solo\_bestrating, the best rating that the player has received.
* solo\_Damagepg

Not only are these stats available for solo game matches but duos and squads as well.

The data set can be found at the following link: <https://www.kaggle.com/lazyjustin/pubgplayerstats>

## Data Limitations

The data does not include the length of time each player has been playing the game. It is hard to differentiate between veteran players from the newer players and to compare their strategies. Since the data is a summary of each player and not a log of each match, our results are based off averages instead of totals. Therefore we can see a trends and relationships, but they may not be accurate in numbers.

## Data Wrangling

View data set.

str(pubg)

## 'data.frame': 87898 obs. of 152 variables:  
## $ player\_name : Factor w/ 87897 levels "0-8-15-Wolf",..: 9354 7919 49103 19286 52621 13466 27173 41838 80221 24731 ...  
## $ tracker\_id : int 4405 8199 4454 7729 0 33313 15585 9454 24029 11289 ...  
## $ solo\_KillDeathRatio : num 3.14 4.41 3.6 14 10.5 5.73 2.45 2.71 3.68 4.17 ...  
## $ solo\_WinRatio : num 17.6 18.2 0 50 33.3 ...  
## $ solo\_TimeSurvived : num 18469 33015 4330 13422 9841 ...  
## $ solo\_RoundsPlayed : int 17 33 5 8 6 16 142 19 44 8 ...  
## $ solo\_Wins : int 3 6 0 4 2 5 12 5 6 2 ...  
## $ solo\_WinTop10Ratio : num 0.83 0.36 0 0.67 0.4 0.5 0.18 0.5 0.23 0.25 ...  
## $ solo\_Top10s : int 4 11 1 6 5 10 34 10 13 4 ...  
## $ solo\_Top10Ratio : num 23.5 33.3 20 75 83.3 62.5 23.9 52.6 29.5 50 ...  
## $ solo\_Losses : int 14 27 5 4 4 11 130 14 38 6 ...  
## $ solo\_Rating : num 1560 1885 1256 1799 1669 ...  
## $ solo\_BestRating : num 1416 1861 1267 1765 1617 ...  
## $ solo\_DamagePg : num 255 393 330 752 637 ...  
## $ solo\_HeadshotKillsPg : num 0.65 1.27 0.6 0.5 1.5 0.69 0.64 0.68 1.02 0.5 ...  
## $ solo\_HealsPg : num 1.94 1.82 1 1 2 1.38 1.94 1.53 1.52 1.12 ...  
## $ solo\_KillsPg : num 2.59 3.61 3.6 7 7 3.94 2.24 2 3.18 3.12 ...  
## $ solo\_MoveDistancePg : num 3321 5021 2805 5987 6527 ...  
## $ solo\_RevivesPg : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ solo\_RoadKillsPg : num 0 0.06 0 0 0 0 0.08 0.11 0.05 0 ...  
## $ solo\_TeamKillsPg : num 0 0 0 0 0 0 0.02 0 0 0 ...  
## $ solo\_TimeSurvivedPg : num 1086 1000 866 1678 1640 ...  
## $ solo\_Top10sPg : num 0.24 0.33 0.2 0.75 0.83 0.62 0.24 0.53 0.3 0.5 ...  
## $ solo\_Kills : int 44 119 18 56 42 63 318 38 140 25 ...  
## $ solo\_Assists : int 1 2 1 3 0 2 32 0 5 1 ...  
## $ solo\_Suicides : int 0 0 0 0 0 0 3 0 0 0 ...  
## $ solo\_TeamKills : int 0 0 0 0 0 0 3 0 0 0 ...  
## $ solo\_HeadshotKills : int 11 42 3 4 9 11 91 13 45 4 ...  
## $ solo\_HeadshotKillRatio : num 0.25 0.35 0.17 0.07 0.21 0.17 0.29 0.34 0.32 0.16 ...  
## $ solo\_VehicleDestroys : int 0 3 0 0 0 1 10 0 1 0 ...  
## $ solo\_RoadKills : int 0 2 0 0 0 0 12 2 2 0 ...  
## $ solo\_DailyKills : int 13 18 1 8 15 4 12 6 13 1 ...  
## $ solo\_WeeklyKills : int 19 18 18 48 29 4 76 6 32 10 ...  
## $ solo\_RoundMostKills : int 13 13 10 14 13 10 11 7 11 8 ...  
## $ solo\_MaxKillStreaks : int 1 3 1 2 3 2 4 2 2 2 ...  
## $ solo\_WeaponAcquired : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ solo\_Days : int 14 10 4 5 2 11 38 7 21 5 ...  
## $ solo\_LongestTimeSurvived : num 1910 1988 1578 1981 1924 ...  
## $ solo\_MostSurvivalTime : num 1910 1988 1578 1981 1924 ...  
## $ solo\_AvgSurvivalTime : num 1263 1221 1133 1678 1640 ...  
## $ solo\_WinPoints : int 2425 3812 2061 1484 1384 1757 3162 1667 3000 2309 ...  
## $ solo\_WalkDistance : num 28924 47869 6341 17580 16736 ...  
## $ solo\_RideDistance : num 27538 117838 7683 30315 22425 ...  
## $ solo\_MoveDistance : num 56462 165707 14024 47895 39162 ...  
## $ solo\_AvgWalkDistance : num 2202 2017 1376 2197 2789 ...  
## $ solo\_AvgRideDistance : num 2764 5189 3052 3789 3738 ...  
## $ solo\_LongestKill : num 305 352 126 412 308 ...  
## $ solo\_Heals : int 33 60 5 8 12 22 275 29 67 9 ...  
## $ solo\_Revives : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ solo\_Boosts : int 29 88 3 24 21 50 371 55 97 24 ...  
## $ solo\_DamageDealt : num 4341 12970 1649 6016 3825 ...  
## $ solo\_DBNOs : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ duo\_KillDeathRatio : num 6.5 4.56 6.55 8 5.03 2.94 3.46 2.82 3.16 4 ...  
## $ duo\_WinRatio : num 33.3 35.7 35.3 66.7 25.7 ...  
## $ duo\_TimeSurvived : num 21914 17765 23449 4196 142572 ...  
## $ duo\_RoundsPlayed : int 15 14 17 3 105 75 163 27 22 17 ...  
## $ duo\_Wins : int 5 5 6 2 27 11 38 5 3 3 ...  
## $ duo\_WinTop10Ratio : num 0.25 0.36 0.3 1 0.45 0.16 0.55 0.45 0.3 0.43 ...  
## $ duo\_Top10s : int 10 7 10 2 60 34 104 11 10 7 ...  
## $ duo\_Top10Ratio : num 66.7 50 58.8 66.7 57.1 45.3 63.8 40.7 45.5 41.2 ...  
## $ duo\_Losses : int 10 9 11 1 78 64 125 22 19 14 ...  
## $ duo\_Rating : num 2055 2098 2120 1558 2363 ...  
## $ duo\_BestRating : num 1928 2062 2053 1466 2366 ...  
## $ duo\_DamagePg : num 570 339 528 434 429 ...  
## $ duo\_HeadshotKillsPg : num 1.2 1.07 0.71 1 0.86 0.51 0.57 0.26 0.5 0.41 ...  
## $ duo\_HealsPg : num 2.33 1.93 1.65 0.33 1.9 1.87 3.13 2.26 2.95 1.47 ...  
## $ duo\_KillsPg : num 4.33 2.93 4.24 2.67 3.73 2.51 2.66 2.3 2.73 3.29 ...  
## $ duo\_MoveDistancePg : num 6387 6869 5840 6206 5662 ...  
## $ duo\_RevivesPg : num 0.13 0.14 0.12 0.33 0.25 0.25 0.23 0.15 0.09 0 ...  
## $ duo\_RoadKillsPg : num 0 0 0 0 0.02 0.01 0.02 0 0.05 0 ...  
## $ duo\_TeamKillsPg : num 0 0.14 0.06 0 0 0.01 0.02 0 0 0 ...  
## $ duo\_TimeSurvivedPg : num 1461 1269 1379 1399 1358 ...  
## $ duo\_Top10sPg : num 0.67 0.5 0.59 0.67 0.57 0.45 0.64 0.41 0.45 0.41 ...  
## $ duo\_Kills : int 65 41 72 8 392 188 433 62 60 56 ...  
## $ duo\_Assists : int 24 16 30 3 81 43 143 20 8 13 ...  
## $ duo\_Suicides : int 0 1 1 0 0 0 1 0 0 0 ...  
## $ duo\_TeamKills : int 0 2 1 0 0 1 4 0 0 0 ...  
## $ duo\_HeadshotKills : int 18 15 12 3 90 38 93 7 11 7 ...  
## $ duo\_HeadshotKillRatio : num 0.28 0.37 0.17 0.38 0.23 0.2 0.21 0.11 0.18 0.12 ...  
## $ duo\_VehicleDestroys : int 2 1 0 0 3 3 12 1 0 0 ...  
## $ duo\_RoadKills : int 0 0 0 0 2 1 3 0 1 0 ...  
## $ duo\_DailyKills : int 22 2 27 8 3 14 19 2 10 18 ...  
## $ duo\_WeeklyKills : int 22 2 27 8 3 27 26 18 17 18 ...  
## $ duo\_RoundMostKills : int 9 9 10 7 15 15 14 8 12 10 ...  
## $ duo\_MaxKillStreaks : int 2 2 2 2 4 2 3 2 2 2 ...  
## $ duo\_WeaponAcquired : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ duo\_Days : int 8 9 9 2 18 14 36 9 9 5 ...  
## $ duo\_LongestTimeSurvived : num 1903 2127 1890 1858 2161 ...  
## $ duo\_MostSurvivalTime : num 1903 2127 1890 1858 2161 ...  
## $ duo\_AvgSurvivalTime : num 1194 1020 1369 1399 1358 ...  
## $ duo\_WinPoints : int 2686 2763 2777 1309 1952 2845 3487 1609 1646 1399 ...  
## $ duo\_WalkDistance : num 30604 29229 41541 10296 191882 ...  
## $ duo\_RideDistance : num 65205 66933 57734 8322 402673 ...  
## $ duo\_MoveDistance : num 95809 96162 99275 18618 594555 ...  
## $ duo\_AvgWalkDistance : num 1252 1484 3185 3432 1827 ...  
## $ duo\_AvgRideDistance : num 4744 4109 4049 2774 3835 ...  
## $ duo\_LongestKill : num 401 183 354 193 363 ...  
## $ duo\_Heals : int 35 27 28 1 199 140 510 61 65 25 ...  
## $ duo\_Revives : int 2 2 2 1 26 19 37 4 2 0 ...  
## [list output truncated]

The data is complete and not missing any values, however some variables do not contribute to analysis, for example, the WeaponAcquired for solo, duo, and squads.

The “ratio” factors have to be recalculated also because they show percentages instead of ratios. Convert back to ratios.

Some of the players only have played only one or two rounds in the game. These inviduals do not play enough to show a pattern, and can affect the data if they play one game and win one game they have 100% success. Thus we only want individuals who play above 100 matches.

For the machine learning analysis it is necessary to split the data for the three game modes to keep the extraneous variables consistent (playing with a team vs. playing alone). In the solo game mode there are stats that do not apply because they are not available in the game mode, thus you remove solo\_revives and solo\_DBNOs.

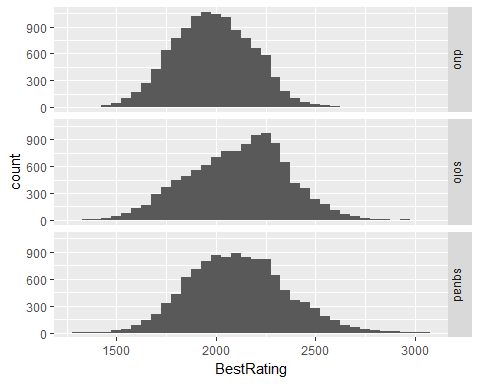
#Solo player data  
pubgsolo <- pubg %>%  
 select(starts\_with("solo")) %>%  
 select(-solo\_Revives, -solo\_DBNOs) %>%  
 mutate(solo\_AvgBoosts = solo\_Boosts/solo\_RoundsPlayed) %>%  
 mutate(AvgHeals = solo\_Heals/solo\_RoundsPlayed)  
  
  
pubgduo <- pubg %>%  
 select(starts\_with("duo")) %>%  
 mutate(duo\_AvgBoosts = duo\_Boosts/duo\_RoundsPlayed) %>%  
 mutate(AvgHeals = duo\_Heals/duo\_RoundsPlayed)  
  
  
pubgsquad <- pubg %>%  
 select(starts\_with("squad")) %>%  
 mutate(squad\_AvgBoosts = squad\_Boosts/squad\_RoundsPlayed) %>%  
 mutate(AvgHeals = squad\_Heals/squad\_RoundsPlayed)

For the preliminary analysis it would be nice to see the three game values therefore if we create a new variable “game mode” and create a new df to split the data into solo, duo and squad modes.

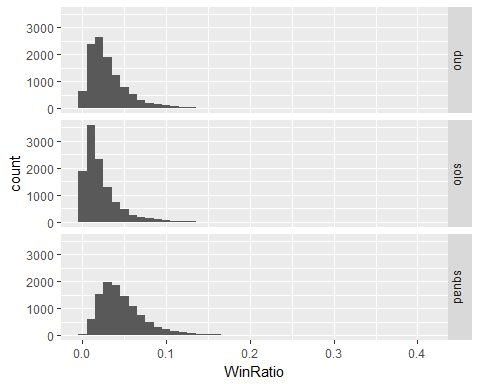
pubgsplit <- pubg %>%  
 gather(Stat, value, solo\_KillDeathRatio:squad\_DBNOs) %>%  
 separate(Stat, c("Game\_Mode", "Variable"), sep = "\_") %>%  
 spread(Variable, value) %>%  
 mutate(AvgBoosts = Boosts/RoundsPlayed) %>%  
 mutate(AvgHeals = Heals/RoundsPlayed)

## Preliminary Analysis

ggplot(pubgsplit, aes(BestRating)) + geom\_histogram(binwidth = 50) + facet\_grid(Game\_Mode ~ .)



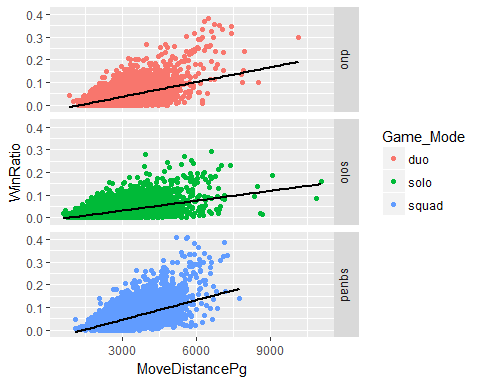
ggplot(pubgsplit, aes(WinRatio)) + geom\_histogram(binwidth = .01) + facet\_grid(Game\_Mode ~ .)



From the histograms above, we can see that we have a normal distribution of spread based on the player’s BestRating and WinRatio. After combing through our data this subset will give a model fitting players of PUBG. Show general trends based on variables.

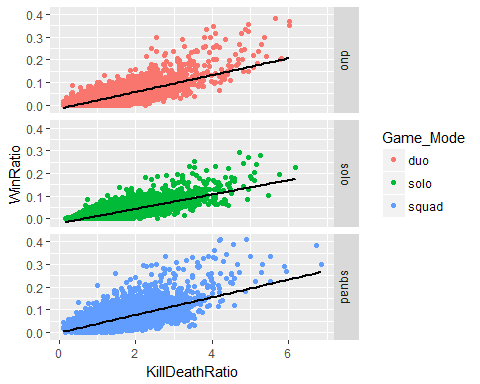
### MoveDistancePg, Distances traveled in game.

ggplot(pubgsplit, aes(MoveDistancePg, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



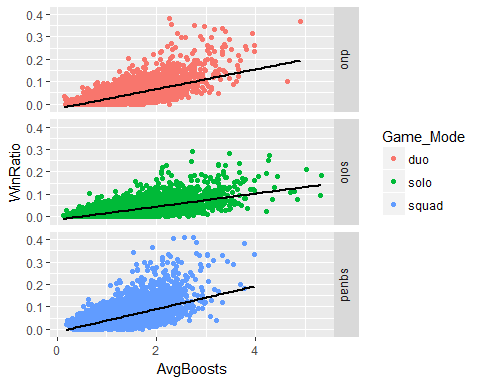
### KillDeathRatio, Number of kills divided by deaths.

ggplot(pubgsplit, aes(KillDeathRatio, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



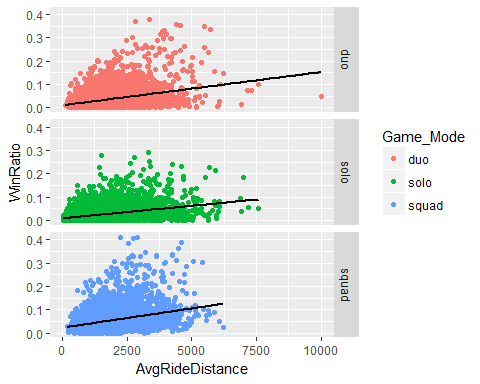
### AvgBoosts, Average boosts used per game.

ggplot(pubgsplit, aes(AvgBoosts, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



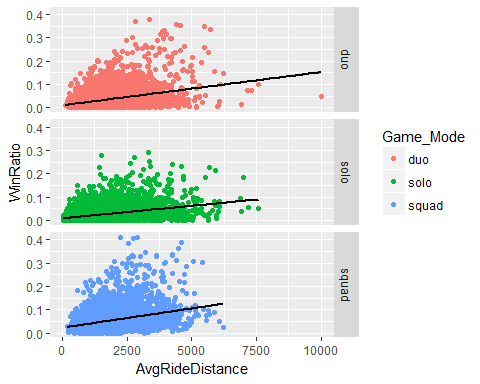
### Average Ride Distance per game.

ggplot(pubgsplit, aes(AvgRideDistance, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



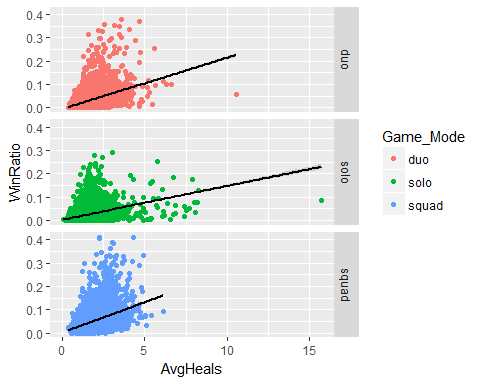
### Average Walk Distance per game.

ggplot(pubgsplit, aes(AvgRideDistance, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



### AvgHeals, Average healing done per game.

ggplot(pubgsplit, aes(AvgHeals, WinRatio, col = Game\_Mode)) + geom\_point() + facet\_grid(Game\_Mode ~.) +   
 geom\_smooth(method = "lm", col = "black")



For the variables of MoveDistancePg, KillDeathRatio, and AvgBoosts seem to show a stronger positive relationship with WinRatio than any of the other variables.

## Machine Learning

Using Machine Learning to show that these other variables are just as important as the KillDeathRatio. Also that these variable can help improve the higher WinRatio in games.

First create a training subset and test subset. Since our data set is large we will use a 75/25 split, 75% training and 25% test. Then run analysis for each game mode type.

Because the data provides number of wins and losses per player, we need to use the cbind() fucntion to provide ratio of successes vs. failures.

pubgsolo$soloWinLoss <- cbind(pubgsolo$solo\_Wins, pubgsolo$solo\_Losses)  
pubgduo$duoWinLoss <- cbind(pubgduo$duo\_Wins, pubgduo$duo\_Losses)  
pubgsquad$squadWinLoss <- cbind(pubgsquad$squad\_Wins, pubgsquad$squad\_Losses)

### Solo Game Mode

set.seed(1234)  
split = sample.split(pubgsolo$solo\_WinRatio, SplitRatio = 0.75)  
str(pubgsolo)

## 'data.frame': 11146 obs. of 50 variables:  
## $ solo\_KillDeathRatio : num 2.45 1.2 4.52 5.16 1.83 4.35 2.23 3.54 3.23 3.11 ...  
## $ solo\_WinRatio : num 0.0845 0.0119 0.1733 0.2377 0.0385 ...  
## $ solo\_TimeSurvived : num 148051 232996 331173 397022 95015 ...  
## $ solo\_RoundsPlayed : int 142 335 277 265 130 270 133 140 259 143 ...  
## $ solo\_Wins : int 12 4 48 63 5 49 5 32 50 14 ...  
## $ solo\_WinTop10Ratio : num 0.18 0.06 0.25 0.41 0.23 0.38 0.28 0.39 0.19 0.26 ...  
## $ solo\_Top10s : int 34 33 109 153 13 117 18 77 133 27 ...  
## $ solo\_Top10Ratio : num 0.239 0.099 0.394 0.577 0.1 0.433 0.135 0.55 0.514 0.189 ...  
## $ solo\_Losses : int 130 331 229 202 125 221 128 108 209 129 ...  
## $ solo\_Rating : num 2605 2336 2928 2956 2185 ...  
## $ solo\_BestRating : num 2632 2411 2940 2950 2212 ...  
## $ solo\_DamagePg : num 285 144 420 460 196 ...  
## $ solo\_HeadshotKillsPg : num 0.64 0.24 1.02 0.91 0.32 0.98 0.42 0.6 0.58 0.85 ...  
## $ solo\_HealsPg : num 1.94 1.21 1.6 1.75 1.34 1.98 1.74 1.31 4.16 1.51 ...  
## $ solo\_KillsPg : num 2.24 1.18 3.74 3.94 1.76 3.56 2.14 2.73 2.61 2.8 ...  
## $ solo\_MoveDistancePg : num 4723 2066 6060 5908 2185 ...  
## $ solo\_RevivesPg : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ solo\_RoadKillsPg : num 0.08 0.02 0.09 0.06 0 0.07 0.02 0.09 0 0.01 ...  
## $ solo\_TeamKillsPg : num 0.02 0.01 0.01 0 0 0.01 0 0 0.01 0.01 ...  
## $ solo\_TimeSurvivedPg : num 1043 696 1196 1498 731 ...  
## $ solo\_Top10sPg : num 0.24 0.1 0.39 0.58 0.1 0.43 0.14 0.55 0.51 0.19 ...  
## $ solo\_Kills : int 318 396 1035 1043 229 961 285 382 675 401 ...  
## $ solo\_Assists : int 32 17 68 65 11 54 11 30 47 22 ...  
## $ solo\_Suicides : int 3 3 2 0 0 2 0 0 3 1 ...  
## $ solo\_TeamKills : int 3 3 2 0 0 2 0 0 3 1 ...  
## $ solo\_HeadshotKills : int 91 82 283 240 41 264 56 84 151 121 ...  
## $ solo\_HeadshotKillRatio : num 0.29 0.21 0.27 0.23 0.18 0.27 0.2 0.22 0.22 0.3 ...  
## $ solo\_VehicleDestroys : int 10 3 21 30 0 36 5 7 9 4 ...  
## $ solo\_RoadKills : int 12 6 26 17 0 18 3 12 0 2 ...  
## $ solo\_DailyKills : int 12 6 52 11 2 76 16 7 25 14 ...  
## $ solo\_WeeklyKills : int 76 9 372 59 3 76 16 9 31 54 ...  
## $ solo\_RoundMostKills : int 11 8 18 14 11 18 15 12 10 14 ...  
## $ solo\_MaxKillStreaks : int 4 3 4 3 2 3 3 3 3 3 ...  
## $ solo\_Days : int 38 43 39 33 34 36 32 33 39 33 ...  
## $ solo\_LongestTimeSurvived: num 2724 1959 2179 2204 2008 ...  
## $ solo\_MostSurvivalTime : num 2724 1959 2179 2204 2008 ...  
## $ solo\_AvgSurvivalTime : num 1239 839 965 1498 540 ...  
## $ solo\_WinPoints : int 3162 2934 4841 2460 4013 3460 2032 5049 3533 5209 ...  
## $ solo\_WalkDistance : num 201112 313104 374699 438117 146445 ...  
## $ solo\_RideDistance : num 469526 379037 1303816 1127482 137573 ...  
## $ solo\_MoveDistance : num 670638 692141 1678515 1565599 284018 ...  
## $ solo\_AvgWalkDistance : num 1939 1277 1165 1653 811 ...  
## $ solo\_AvgRideDistance : num 5190 1972 4096 4255 712 ...  
## $ solo\_LongestKill : num 464 344 441 453 434 ...  
## $ solo\_Heals : int 275 406 443 463 174 535 232 183 1077 216 ...  
## $ solo\_Boosts : int 371 332 853 972 136 1094 186 475 1091 233 ...  
## $ solo\_DamageDealt : num 40410 48407 116260 121904 25501 ...  
## $ solo\_AvgBoosts : num 2.613 0.991 3.079 3.668 1.046 ...  
## $ AvgHeals : num 1.94 1.21 1.6 1.75 1.34 ...  
## $ soloWinLoss : int [1:11146, 1:2] 12 4 48 63 5 49 5 32 50 14 ...

pubgTraining = subset(pubgsolo, split == TRUE)  
pubgTest = subset(pubgsolo, split == FALSE)  
  
#Check to make sure the Training set is approximately 75%  
nrow(pubgTraining)

## [1] 8470

nrow(pubgTest)

## [1] 2676

nrow(pubgTraining)/sum(nrow(pubgTraining)+(nrow(pubgTest)))

## [1] 0.7599139

Now use backwards selection to build a logistic model, where you start with all independent variables and remove those that have no significance.

pubgsololog1 <- glm(soloWinLoss ~ ., data = pubgTraining, family = binomial)  
summary(pubgsololog1)

##   
## Call:  
## glm(formula = soloWinLoss ~ ., family = binomial, data = pubgTraining)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -7.5472 -0.4536 0.0074 0.3920 8.7794   
##   
## Coefficients: (4 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -7.123e+00 1.703e-01 -41.823 < 2e-16 \*\*\*  
## solo\_KillDeathRatio -5.699e+00 1.289e-01 -44.223 < 2e-16 \*\*\*  
## solo\_WinRatio 3.534e+01 8.342e-01 42.362 < 2e-16 \*\*\*  
## solo\_TimeSurvived -1.177e-06 6.145e-07 -1.916 0.055353 .   
## solo\_RoundsPlayed -1.132e-03 3.545e-04 -3.193 0.001408 \*\*   
## solo\_Wins -1.269e-02 3.034e-03 -4.185 2.86e-05 \*\*\*  
## solo\_WinTop10Ratio 7.121e-01 4.711e-02 15.115 < 2e-16 \*\*\*  
## solo\_Top10s 4.943e-03 1.635e-03 3.024 0.002497 \*\*   
## solo\_Top10Ratio -1.826e+00 1.913e+00 -0.954 0.339962   
## solo\_Losses NA NA NA NA   
## solo\_Rating -1.592e-04 9.996e-05 -1.593 0.111206   
## solo\_BestRating 3.409e-04 1.062e-04 3.211 0.001323 \*\*   
## solo\_DamagePg 1.100e-03 1.355e-03 0.812 0.417029   
## solo\_HeadshotKillsPg 5.511e-01 1.940e-01 2.840 0.004505 \*\*   
## solo\_HealsPg -1.227e+00 1.870e+00 -0.656 0.511701   
## solo\_KillsPg 5.857e+00 1.903e-01 30.780 < 2e-16 \*\*\*  
## solo\_MoveDistancePg -4.058e-05 2.282e-05 -1.778 0.075327 .   
## solo\_RevivesPg NA NA NA NA   
## solo\_RoadKillsPg 6.563e-02 5.173e-01 0.127 0.899044   
## solo\_TeamKillsPg 8.150e-01 1.292e+00 0.631 0.528024   
## solo\_TimeSurvivedPg 7.933e-04 1.665e-04 4.765 1.89e-06 \*\*\*  
## solo\_Top10sPg 2.707e-01 1.887e+00 0.143 0.885915   
## solo\_Kills 3.584e-04 5.584e-04 0.642 0.520949   
## solo\_Assists -1.608e-03 6.966e-04 -2.308 0.020976 \*   
## solo\_Suicides -2.946e-03 6.790e-03 -0.434 0.664353   
## solo\_TeamKills NA NA NA NA   
## solo\_HeadshotKills -1.676e-03 5.293e-04 -3.167 0.001541 \*\*   
## solo\_HeadshotKillRatio -2.800e-01 2.836e-01 -0.987 0.323453   
## solo\_VehicleDestroys 2.206e-03 1.656e-03 1.332 0.182931   
## solo\_RoadKills -1.530e-04 1.879e-03 -0.081 0.935086   
## solo\_DailyKills -1.059e-03 4.871e-04 -2.175 0.029633 \*   
## solo\_WeeklyKills 2.158e-04 1.584e-04 1.363 0.172953   
## solo\_RoundMostKills 9.574e-03 2.767e-03 3.460 0.000539 \*\*\*  
## solo\_MaxKillStreaks -3.506e-03 4.830e-03 -0.726 0.467961   
## solo\_Days 1.061e-03 5.894e-04 1.800 0.071838 .   
## solo\_LongestTimeSurvived 5.935e-04 6.161e-05 9.632 < 2e-16 \*\*\*  
## solo\_MostSurvivalTime NA NA NA NA   
## solo\_AvgSurvivalTime 4.978e-06 6.869e-05 0.072 0.942232   
## solo\_WinPoints 1.152e-05 4.657e-06 2.473 0.013386 \*   
## solo\_WalkDistance -2.274e-01 4.260e-02 -5.337 9.47e-08 \*\*\*  
## solo\_RideDistance -2.274e-01 4.260e-02 -5.337 9.47e-08 \*\*\*  
## solo\_MoveDistance 2.274e-01 4.260e-02 5.337 9.47e-08 \*\*\*  
## solo\_AvgWalkDistance -8.614e-06 2.657e-05 -0.324 0.745751   
## solo\_AvgRideDistance 3.699e-06 1.217e-05 0.304 0.761230   
## solo\_LongestKill 7.720e-05 4.979e-05 1.550 0.121063   
## solo\_Heals 7.736e-05 7.159e-05 1.081 0.279907   
## solo\_Boosts 1.526e-04 1.232e-04 1.238 0.215701   
## solo\_DamageDealt 3.832e-06 5.535e-06 0.692 0.488718   
## solo\_AvgBoosts -7.939e-02 3.069e-02 -2.587 0.009681 \*\*   
## AvgHeals 1.208e+00 1.870e+00 0.646 0.518316   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 38494.4 on 8469 degrees of freedom  
## Residual deviance: 4848.5 on 8424 degrees of freedom  
## AIC: 28013  
##   
## Number of Fisher Scoring iterations: 4

Select all variables with significance above 0.1.

pubgsololog2 <- glm(soloWinLoss ~ solo\_KillDeathRatio + solo\_WinRatio + solo\_RoundsPlayed + solo\_Wins + solo\_WinTop10Ratio + solo\_Top10s + solo\_BestRating + solo\_HeadshotKillsPg + solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg + solo\_Assists + solo\_HeadshotKills + solo\_DailyKills + solo\_RoundMostKills + solo\_LongestTimeSurvived + solo\_WinPoints + solo\_WalkDistance + solo\_RideDistance + solo\_MoveDistance + solo\_AvgBoosts, data = pubgTraining, family = binomial)  
summary(pubgsololog2)

##   
## Call:  
## glm(formula = soloWinLoss ~ solo\_KillDeathRatio + solo\_WinRatio +   
## solo\_RoundsPlayed + solo\_Wins + solo\_WinTop10Ratio + solo\_Top10s +   
## solo\_BestRating + solo\_HeadshotKillsPg + solo\_KillsPg + solo\_MoveDistancePg +   
## solo\_TimeSurvivedPg + solo\_Assists + solo\_HeadshotKills +   
## solo\_DailyKills + solo\_RoundMostKills + solo\_LongestTimeSurvived +   
## solo\_WinPoints + solo\_WalkDistance + solo\_RideDistance +   
## solo\_MoveDistance + solo\_AvgBoosts, family = binomial, data = pubgTraining)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -7.9889 -0.4618 0.0023 0.3971 8.6813   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -7.100e+00 1.351e-01 -52.549 < 2e-16 \*\*\*  
## solo\_KillDeathRatio -5.621e+00 1.203e-01 -46.745 < 2e-16 \*\*\*  
## solo\_WinRatio 3.226e+01 6.457e-01 49.969 < 2e-16 \*\*\*  
## solo\_RoundsPlayed -9.841e-04 1.428e-04 -6.892 5.49e-12 \*\*\*  
## solo\_Wins -2.845e-03 2.436e-03 -1.168 0.242904   
## solo\_WinTop10Ratio 7.520e-01 4.361e-02 17.246 < 2e-16 \*\*\*  
## solo\_Top10s 3.804e-04 7.346e-04 0.518 0.604629   
## solo\_BestRating 1.945e-04 2.820e-05 6.895 5.39e-12 \*\*\*  
## solo\_HeadshotKillsPg -1.144e-01 7.539e-02 -1.518 0.129060   
## solo\_KillsPg 6.101e+00 1.271e-01 48.016 < 2e-16 \*\*\*  
## solo\_MoveDistancePg -4.278e-05 1.502e-05 -2.847 0.004412 \*\*   
## solo\_TimeSurvivedPg 3.801e-04 8.075e-05 4.707 2.51e-06 \*\*\*  
## solo\_Assists -1.043e-03 6.234e-04 -1.674 0.094226 .   
## solo\_HeadshotKills 7.268e-04 2.139e-04 3.397 0.000681 \*\*\*  
## solo\_DailyKills -9.207e-04 4.191e-04 -2.197 0.028043 \*   
## solo\_RoundMostKills 1.055e-02 2.666e-03 3.958 7.56e-05 \*\*\*  
## solo\_LongestTimeSurvived 5.900e-04 6.113e-05 9.652 < 2e-16 \*\*\*  
## solo\_WinPoints 1.455e-05 3.534e-06 4.119 3.81e-05 \*\*\*  
## solo\_WalkDistance -2.315e-01 4.156e-02 -5.571 2.54e-08 \*\*\*  
## solo\_RideDistance -2.315e-01 4.156e-02 -5.571 2.54e-08 \*\*\*  
## solo\_MoveDistance 2.315e-01 4.156e-02 5.571 2.54e-08 \*\*\*  
## solo\_AvgBoosts -4.954e-02 1.364e-02 -3.633 0.000280 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 38494.4 on 8469 degrees of freedom  
## Residual deviance: 4941.7 on 8448 degrees of freedom  
## AIC: 28058  
##   
## Number of Fisher Scoring iterations: 4

Continuing removing insignificant factors.

pubgsololog3 <- glm(soloWinLoss ~ solo\_KillDeathRatio + solo\_WinRatio + solo\_RoundsPlayed + solo\_WinTop10Ratio + solo\_BestRating + solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg + solo\_Assists + solo\_HeadshotKills + solo\_DailyKills + solo\_RoundMostKills + solo\_LongestTimeSurvived + solo\_WinPoints + solo\_WalkDistance + solo\_RideDistance + solo\_MoveDistance + solo\_AvgBoosts, data = pubgTraining, family = binomial)  
summary(pubgsololog3)

##   
## Call:  
## glm(formula = soloWinLoss ~ solo\_KillDeathRatio + solo\_WinRatio +   
## solo\_RoundsPlayed + solo\_WinTop10Ratio + solo\_BestRating +   
## solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg +   
## solo\_Assists + solo\_HeadshotKills + solo\_DailyKills + solo\_RoundMostKills +   
## solo\_LongestTimeSurvived + solo\_WinPoints + solo\_WalkDistance +   
## solo\_RideDistance + solo\_MoveDistance + solo\_AvgBoosts, family = binomial,   
## data = pubgTraining)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -8.0569 -0.4645 0.0030 0.3982 8.7130   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -7.141e+00 1.294e-01 -55.204 < 2e-16 \*\*\*  
## solo\_KillDeathRatio -5.662e+00 1.156e-01 -48.975 < 2e-16 \*\*\*  
## solo\_WinRatio 3.189e+01 5.262e-01 60.605 < 2e-16 \*\*\*  
## solo\_RoundsPlayed -8.528e-04 1.086e-04 -7.855 4.01e-15 \*\*\*  
## solo\_WinTop10Ratio 7.439e-01 4.260e-02 17.462 < 2e-16 \*\*\*  
## solo\_BestRating 1.912e-04 2.809e-05 6.805 1.01e-11 \*\*\*  
## solo\_KillsPg 6.129e+00 1.231e-01 49.776 < 2e-16 \*\*\*  
## solo\_MoveDistancePg -3.722e-05 1.211e-05 -3.073 0.002118 \*\*   
## solo\_TimeSurvivedPg 4.050e-04 6.669e-05 6.072 1.26e-09 \*\*\*  
## solo\_Assists -9.939e-04 5.848e-04 -1.700 0.089207 .   
## solo\_HeadshotKills 4.625e-04 1.248e-04 3.705 0.000211 \*\*\*  
## solo\_DailyKills -8.782e-04 4.182e-04 -2.100 0.035718 \*   
## solo\_RoundMostKills 1.033e-02 2.663e-03 3.879 0.000105 \*\*\*  
## solo\_LongestTimeSurvived 5.950e-04 6.092e-05 9.767 < 2e-16 \*\*\*  
## solo\_WinPoints 1.394e-05 3.512e-06 3.969 7.21e-05 \*\*\*  
## solo\_WalkDistance -2.380e-01 4.109e-02 -5.793 6.93e-09 \*\*\*  
## solo\_RideDistance -2.380e-01 4.109e-02 -5.793 6.93e-09 \*\*\*  
## solo\_MoveDistance 2.380e-01 4.109e-02 5.793 6.93e-09 \*\*\*  
## solo\_AvgBoosts -4.898e-02 1.361e-02 -3.599 0.000320 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 38494.4 on 8469 degrees of freedom  
## Residual deviance: 4944.5 on 8451 degrees of freedom  
## AIC: 28055  
##   
## Number of Fisher Scoring iterations: 4

Since we really want to determine variables per game to help individuals figure out a strategy per game. We will remove all variables that are not relevant to an average game.

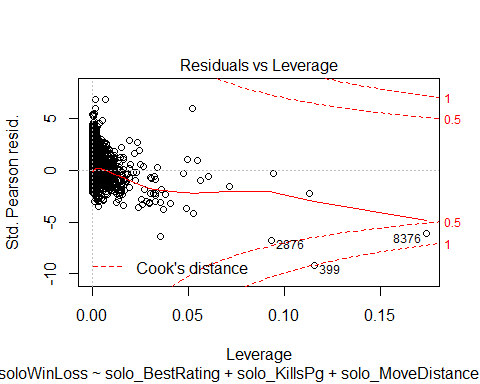
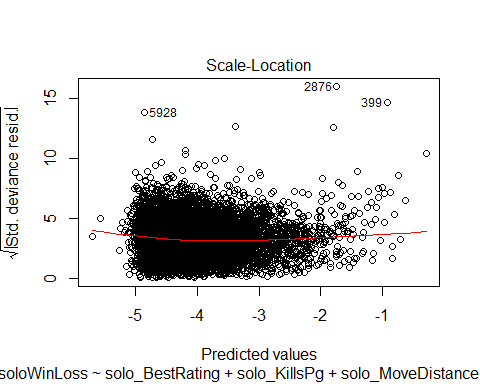
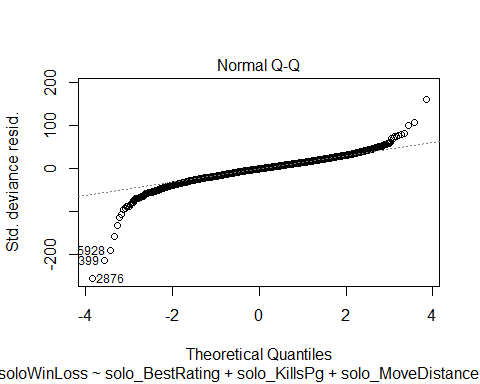
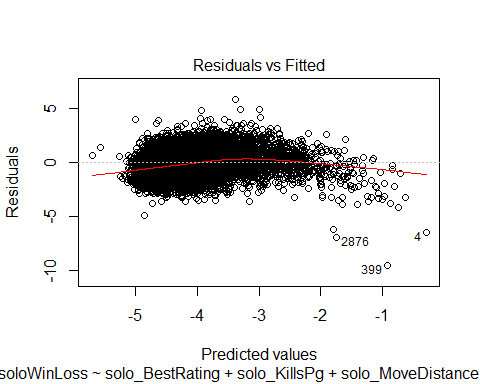
pubgsololog3 <- glm(soloWinLoss ~ solo\_KillDeathRatio + solo\_BestRating + solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg + solo\_AvgBoosts + solo\_DamagePg + solo\_HeadshotKillsPg + solo\_HealsPg + solo\_KillsPg + solo\_RoadKillsPg + solo\_AvgWalkDistance + solo\_AvgRideDistance , data = pubgTraining, family = binomial)  
summary(pubgsololog3)

##   
## Call:  
## glm(formula = soloWinLoss ~ solo\_KillDeathRatio + solo\_BestRating +   
## solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg +   
## solo\_AvgBoosts + solo\_DamagePg + solo\_HeadshotKillsPg + solo\_HealsPg +   
## solo\_KillsPg + solo\_RoadKillsPg + solo\_AvgWalkDistance +   
## solo\_AvgRideDistance, family = binomial, data = pubgTraining)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -9.5045 -0.8819 -0.1186 0.6676 5.8409   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -7.075e+00 5.928e-02 -119.357 < 2e-16 \*\*\*  
## solo\_KillDeathRatio 4.917e-02 5.585e-02 0.880 0.37864   
## solo\_BestRating 1.768e-04 2.623e-05 6.739 1.60e-11 \*\*\*  
## solo\_KillsPg 9.601e-01 9.795e-02 9.802 < 2e-16 \*\*\*  
## solo\_MoveDistancePg 2.299e-05 1.175e-05 1.955 0.05053 .   
## solo\_TimeSurvivedPg 1.838e-03 5.853e-05 31.397 < 2e-16 \*\*\*  
## solo\_AvgBoosts 7.130e-02 1.423e-02 5.012 5.39e-07 \*\*\*  
## solo\_DamagePg -1.614e-03 6.714e-04 -2.405 0.01619 \*   
## solo\_HeadshotKillsPg -2.589e-01 6.593e-02 -3.927 8.62e-05 \*\*\*  
## solo\_HealsPg 5.477e-03 8.651e-03 0.633 0.52668   
## solo\_RoadKillsPg -8.791e-01 2.713e-01 -3.240 0.00119 \*\*   
## solo\_AvgWalkDistance -5.264e-05 1.463e-05 -3.598 0.00032 \*\*\*  
## solo\_AvgRideDistance 4.947e-05 8.600e-06 5.752 8.79e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 38494 on 8469 degrees of freedom  
## Residual deviance: 11940 on 8457 degrees of freedom  
## AIC: 35038  
##   
## Number of Fisher Scoring iterations: 5

pubgsololog4 <- glm(soloWinLoss ~ solo\_BestRating + solo\_KillsPg + solo\_MoveDistancePg + solo\_TimeSurvivedPg + solo\_AvgBoosts + solo\_DamagePg + solo\_HeadshotKillsPg + solo\_KillsPg + solo\_RoadKillsPg + solo\_AvgWalkDistance + solo\_AvgRideDistance , data = pubgTraining, family = binomial)  
summary(pubgsololog4)

##   
## Call:  
## glm(formula = soloWinLoss ~ solo\_BestRating + solo\_KillsPg +   
## solo\_MoveDistancePg + solo\_TimeSurvivedPg + solo\_AvgBoosts +   
## solo\_DamagePg + solo\_HeadshotKillsPg + solo\_KillsPg + solo\_RoadKillsPg +   
## solo\_AvgWalkDistance + solo\_AvgRideDistance, family = binomial,   
## data = pubgTraining)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -9.5177 -0.8844 -0.1181 0.6664 5.8205   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -7.086e+00 5.607e-02 -126.376 < 2e-16 \*\*\*  
## solo\_BestRating 1.751e-04 2.618e-05 6.689 2.24e-11 \*\*\*  
## solo\_KillsPg 1.020e+00 7.234e-02 14.097 < 2e-16 \*\*\*  
## solo\_MoveDistancePg 2.434e-05 1.167e-05 2.085 0.037046 \*   
## solo\_TimeSurvivedPg 1.852e-03 5.609e-05 33.026 < 2e-16 \*\*\*  
## solo\_AvgBoosts 7.440e-02 1.325e-02 5.615 1.97e-08 \*\*\*  
## solo\_DamagePg -1.671e-03 6.686e-04 -2.499 0.012470 \*   
## solo\_HeadshotKillsPg -2.493e-01 6.523e-02 -3.821 0.000133 \*\*\*  
## solo\_RoadKillsPg -9.103e-01 2.692e-01 -3.381 0.000721 \*\*\*  
## solo\_AvgWalkDistance -5.513e-05 1.432e-05 -3.849 0.000119 \*\*\*  
## solo\_AvgRideDistance 4.986e-05 8.588e-06 5.806 6.41e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 38494 on 8469 degrees of freedom  
## Residual deviance: 11941 on 8459 degrees of freedom  
## AIC: 35035  
##   
## Number of Fisher Scoring iterations: 4

plot(pubgsololog4)



From the tests we find that pubgsololog1 had the smallest AIC and was the most accurate even though it contains insignificant variables. This would be the most accurate regression to determine a player’s ratio of wins.

To increase chances of winning per game you can use pubgsololog4 to see what your probability of winning is. From the data in solo matches an individual wants to have high kills, move large distances, use boosts and use a vehicle. Most interesting of all your KillDeathRatio does not matter in determining if you will in a current game.