

washu2 EDA

2025-07-02

```
library("readr")
library("dplyr")
library("ggplot2")
library("readr")
library("stringr")
library("glue")
```

```
g <- params$category
singular_game <- readr::read_csv(glue("Desktop/SURA project code/extended_cmu_data/extended_cmu_data_",
```

```
## New names:
## Rows: 27 Columns: 22
## -- Column specification
## ----- Delimiter: "," chr
## (1): LINEUP (NAMES) dbl (20): ...1, NUMBER OF GUARDS, OPPONENT POSSESSIONS,
## CMU POSSESSIONS, ... time (1): LINEUP MINUTES
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this
## message.
## * `` -> `...1`
```

```
# if negatives in any columns (specifically had problem in possession column)
for (colName in colnames(c("CMU POSSESSIONS", "OPPONENT POSSESSIONS"))){
  singular_game[[colName]][singular_game[[colName]] < 0] <- 0
}
```

```
#individual_games <- readr::read_csv("Desktop/SURA project code/data frames/shortened.csv")
```

```
singular_game$`LINEUP MINUTES` <- sapply(singular_game$`LINEUP MINUTES`, function(t){
  parts <- as.integer(strsplit(as.character(t),":")[[1]])
  parts[1]*60 + parts[2]
})
```

```
singular_game <- singular_game %>% rename('LINEUP SECONDS' = `LINEUP MINUTES`) %>% mutate(LINEUP_SORTED =
  if (is.na(1)) return(NA)
  paste(sort(strsplit(1, ", ")[1]), collapse = " "))
}))
```

```
singular_game <- subset(singular_game, !((`SCORE DIFFERENTIAL WHEN ENTER` <= -11 | `SCORE DIFFERENTIAL` > 11))
```

```
game <- singular_game %>% group_by(`LINEUP_SORTED`) %>% summarise(
  `NUMBER OF GUARDS` = mean(`NUMBER OF GUARDS`),
  `OPPONENT POSSESSIONS` = sum(`OPPONENT POSSESSIONS`, na.rm = TRUE),
  `CMU POSSESSIONS` = sum(`CMU POSSESSIONS`, na.rm = TRUE),
  `LINEUP SECONDS` = sum(`LINEUP SECONDS`, na.rm = TRUE),
  `OPPONENT PTS` = sum(`OPPONENT PTS`, na.rm = TRUE),
  `CMU PTS` = sum(`CMU PTS`, na.rm = TRUE),
```

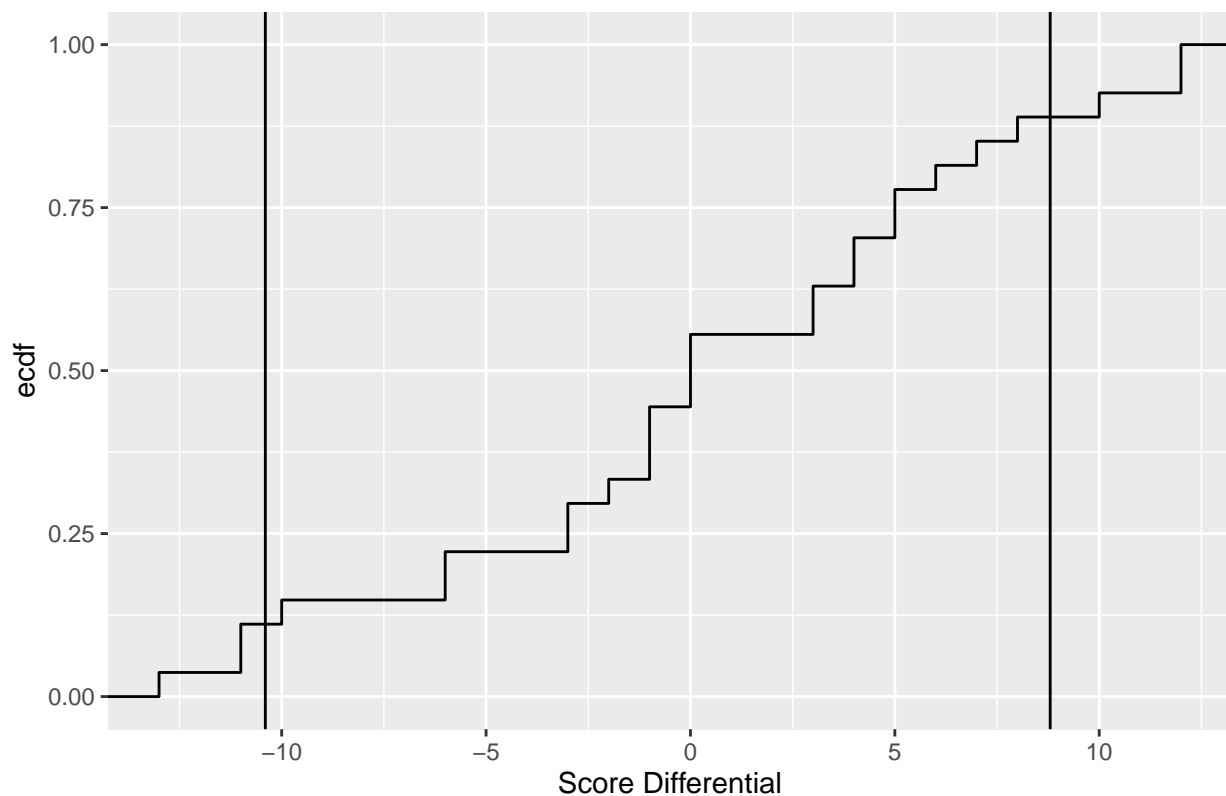
```

`CMU 3PA` = sum(`CMU 3PA`, na.rm = TRUE),
`CMU FGA` = sum(`CMU FGA`, na.rm = TRUE),
`CMU FTA` = sum(`CMU FTA`, na.rm = TRUE),
`CMU REBOUNDS` = sum(`CMU REBOUNDS`, na.rm = TRUE),
`TOTAL REBOUNDS` = sum(`TOTAL REBOUNDS`, na.rm = TRUE),
`SCORE DIFFERENTIAL WHEN ENTER` = paste(`SCORE DIFFERENTIAL WHEN ENTER`, collapse = ", "),
`QUARTER` = paste(`QUARTER`, collapse = ", ")
) %>%mutate(`PACE` = 40 * ((`CMU POSSESSIONS` + `OPPONENT POSSESSIONS`) / (2 * `LINEUP SECONDS`/60)),
`OFFENSIVE RATING` = 100 * (`CMU PTS` / `CMU POSSESSIONS`),
`DEFENSIVE RATING` = 100 * (`OPPONENT PTS` / `OPPONENT POSSESSIONS`),
`NET RATING` = `OFFENSIVE RATING` - `DEFENSIVE RATING`,
`3PA/FGA` = `CMU 3PA` / `CMU FGA`,
`TRUE SHOOTING %` = 100 * (`CMU PTS` / (2 * (`CMU FGA` + (0.44* `CMU FTA`)))),
`TRB%` = 100 * (`CMU REBOUNDS` / `TOTAL REBOUNDS`)

# see where to score differential cut off time -> SHOULD DO THIS AFTER OR BEFORE CUT SCRAP MINUTES?
l <- quantile(singular_game$`SCORE DIFFERENTIAL WHEN ENTER`,probs=c(0.1))
u <- quantile(singular_game$`SCORE DIFFERENTIAL WHEN ENTER`,probs=c(0.9))
ggplot(singular_game, aes(x = `SCORE DIFFERENTIAL WHEN ENTER`)) + stat_ecdf() + geom_vline(xintercept =

```

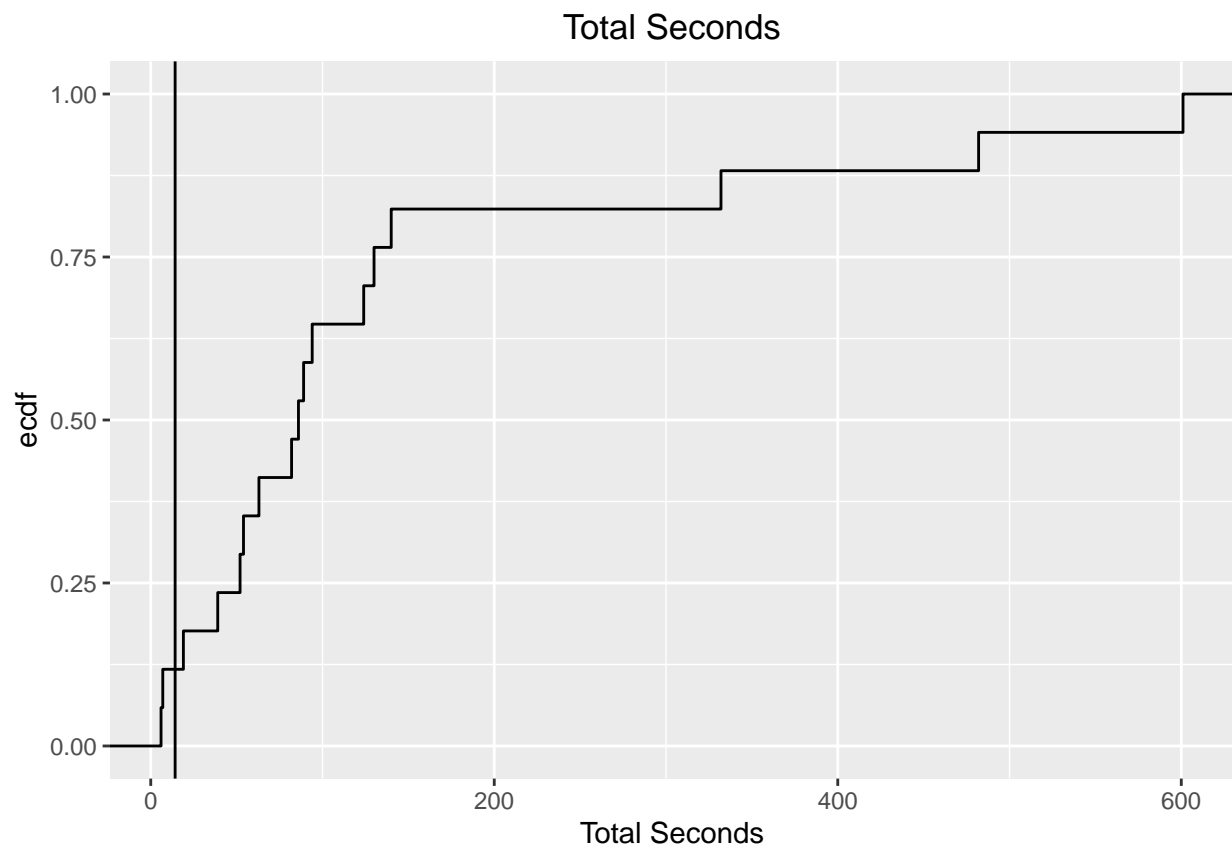
Score Differential



```

# see where to cut time -> SHOULD DO THIS AFTER OR BEFORE CUT SCRAP MINUTES?
p <- quantile(game$`LINEUP SECONDS`,probs=c(0.1))
ggplot(game, aes(x = `LINEUP SECONDS`)) + stat_ecdf() + geom_vline(xintercept = p) + labs(title = "Total

```



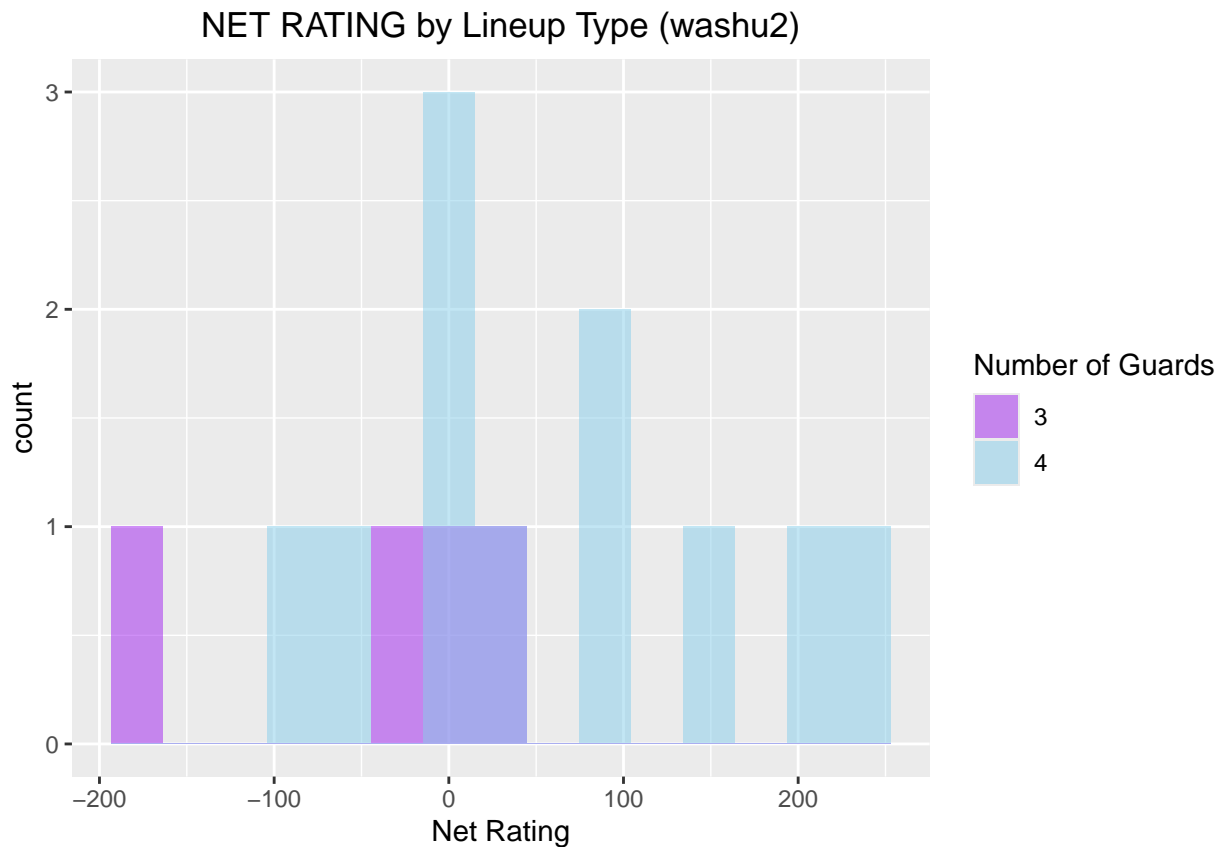
```
#game <- subset(game, `LINEUP SECONDS` >= p)
p

## 10%
## 14.2

#pdf(file = glue("Desktop/SURA project code/sing_game_EDA/{g}_plot.pdf"), width = 6, height = 5)

t_f <- c("3", "4")

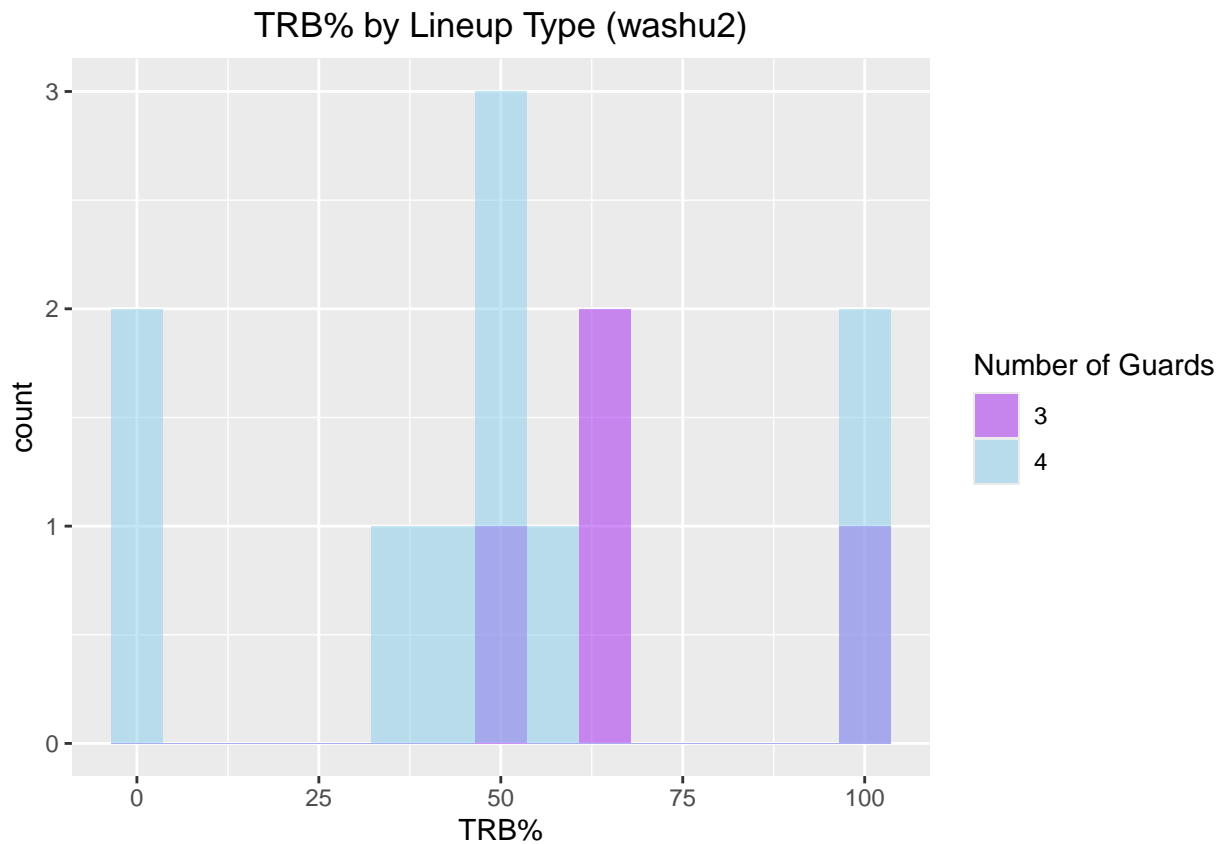
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `NET RATING`, fill = factor(`
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



```
n3 <- sum(game$`NUMBER OF GUARDS` == 3)
n4 <- sum(game$`NUMBER OF GUARDS` == 4)

tapply(game$`NET RATING`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f],
  FUN = function(x) {
    ## $`3`
    ##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
    ## -166.667  -66.667  -16.667  -41.667    8.333   33.333
    ##
    ## $`4`
    ##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.   NA's
    ## -100.000   -4.167   38.312   60.301  125.000  250.000      2
    nr3m <- median(game$`NET RATING`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
    nr4m <- median(game$`NET RATING`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
    nr3m
    ## [1] -16.66667
    nr4m
    ## [1] 38.31169
    nr_p
    ## [1] 0.5218691
    nr_p <- wilcox.test(`NET RATING` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% t_f))
    ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `TRB`, fill = factor(`NUMBER OF GUARDS`)))
```

```
## Warning: Removed 3 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



```
tapply(game$`TRB%`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f],
```

```
## $`3`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  50.00  59.38   64.58   69.79  75.00   100.00
##
```

```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
##   0.00  35.00   50.00   48.33  57.50   100.00     3
```

```
r3m <- median(game$`TRB%`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
```

```
r4m <- median(game$`TRB%`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
```

```
r_p <- wilcox.test(`TRB%` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% t_f), exact = FALSE)
```

```
r3m
```

```
## [1] 64.58333
```

```
r4m
```

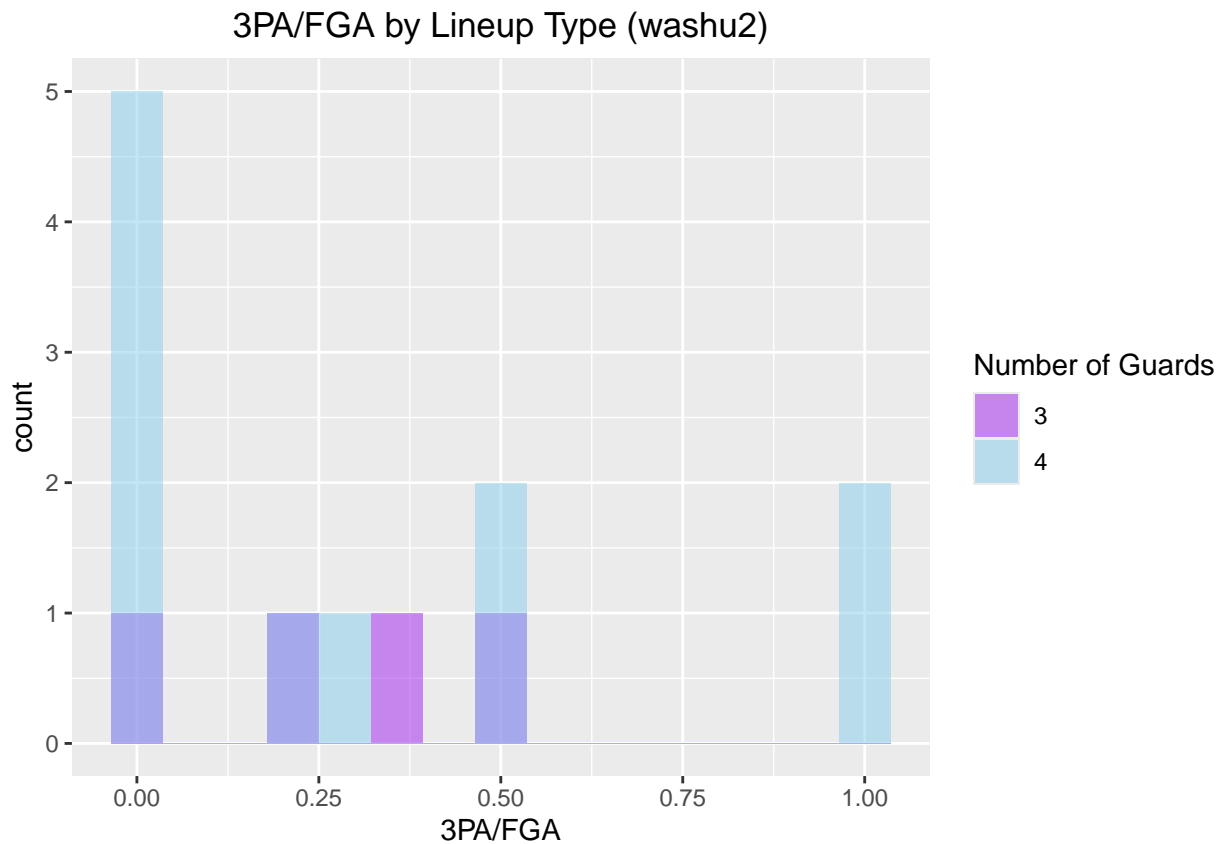
```
## [1] 50
```

```
r_p
```

```
## [1] 0.1504007
```

```
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `3PA/FGA`, fill = factor(`NUMBER OF GUARDS` %in% t_f)))
```

```
## Warning: Removed 2 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



```
tapply(game$`3PA/FGA`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

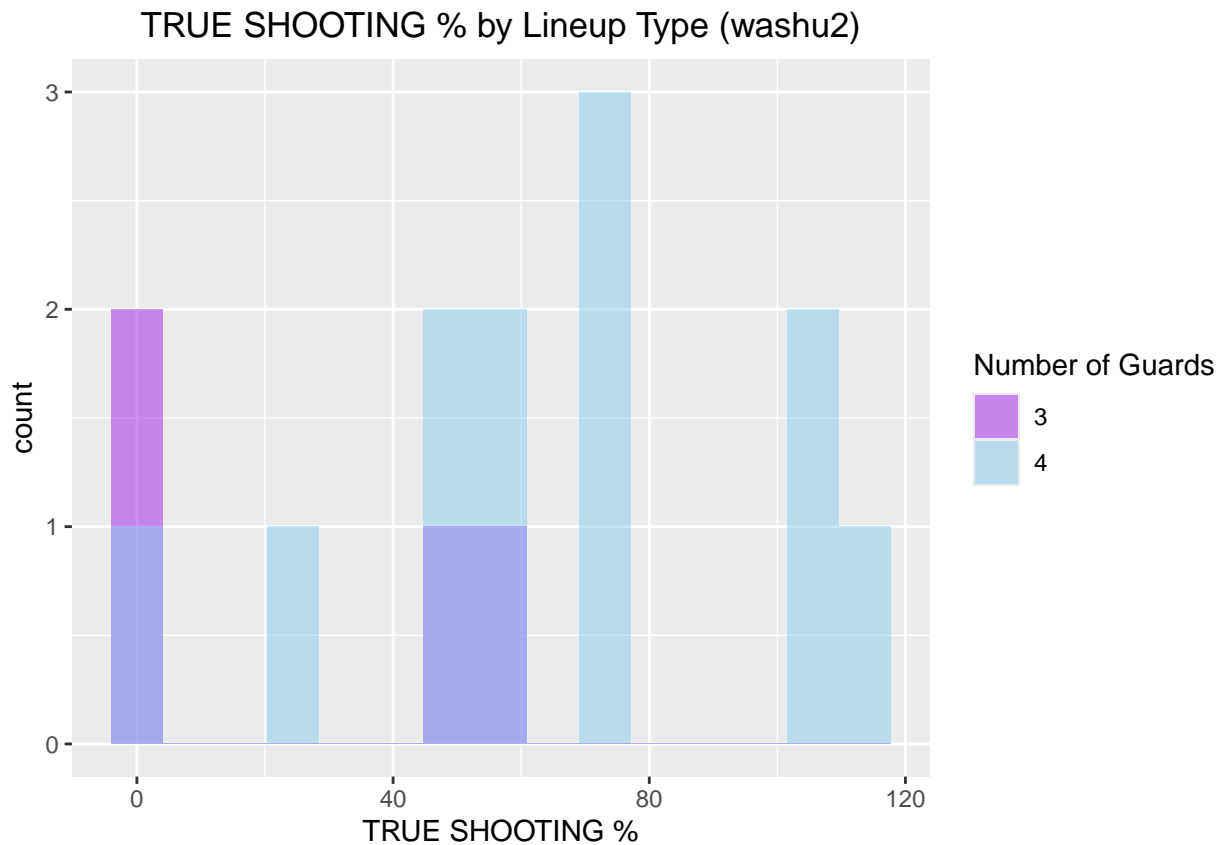
```
## $`3`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.0000 0.1500  0.2667  0.2583 0.3750  0.5000
##
```

```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
## 0.0000 0.0000  0.1818  0.3165 0.5000  1.0000     2
```

```
three3m <- median(game$`3PA/FGA`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
three4m <- median(game$`3PA/FGA`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
three_p <- wilcox.test(`3PA/FGA` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% t_f))
```

```
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `TRUE SHOOTING %`, fill = factor(`NUMBER OF GUARDS`)))
```

```
## Warning: Removed 1 row containing non-finite outside the scale range
## (`stat_bin()`).
```



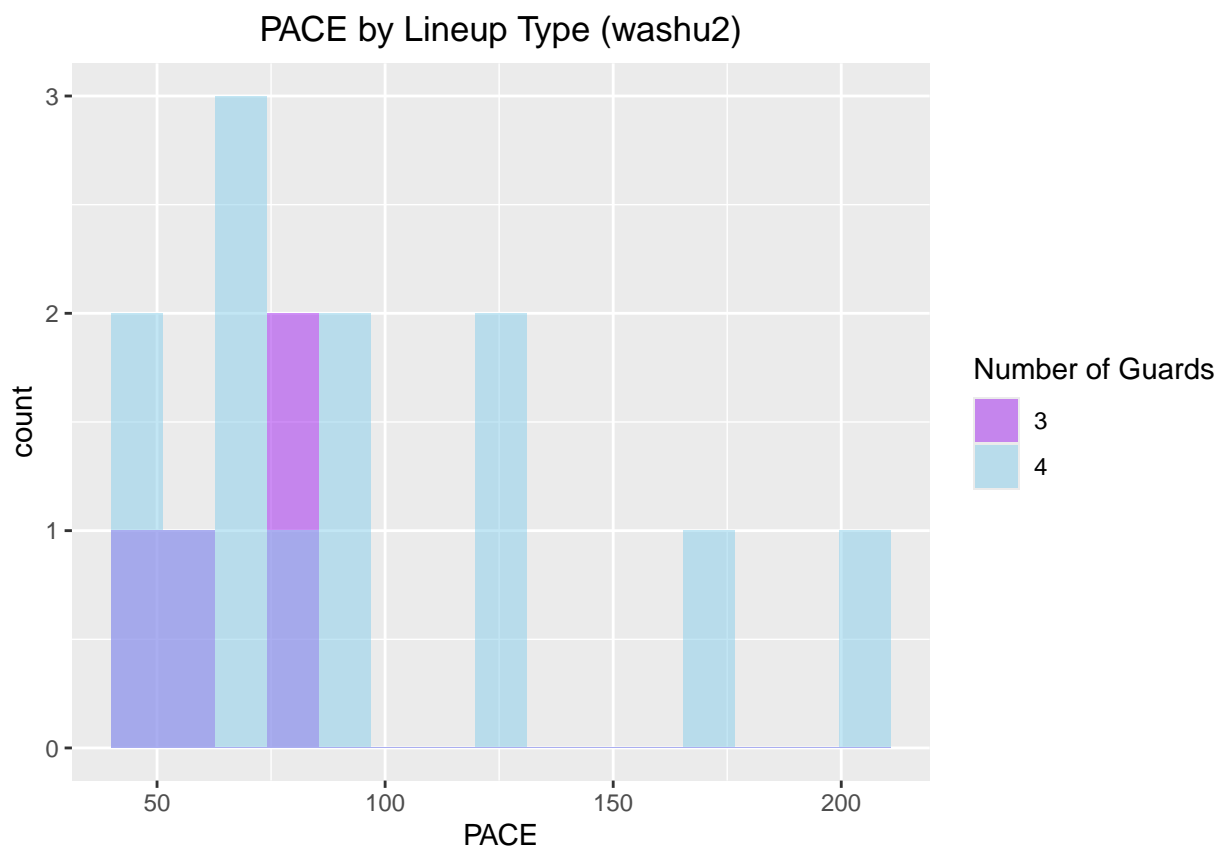
```
tapply(game$`TRUE SHOOTING %`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

```
## $`3`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.00   0.00   26.04   28.02   54.06   60.00
##
```

```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
##   0.00   50.00   63.13   64.34   81.86  113.64     1
```

```
ts3m <- median(game$`TRUE SHOOTING %`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
ts4m <- median(game$`TRUE SHOOTING %`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
ts_p <- wilcox.test(`TRUE SHOOTING %` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% c(3, 4)))
```

```
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `PACE`, fill = factor(`NUMBER OF GUARDS` %in% t_f))) +
  facet_wrap(~ `NUMBER OF GUARDS` %in% t_f, scales = 'free_x') +
  geom_histogram(bins = 10, color = 'black', fill = 'white') +
  theme_minimal()
```



```
tapply(game$`PACE`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

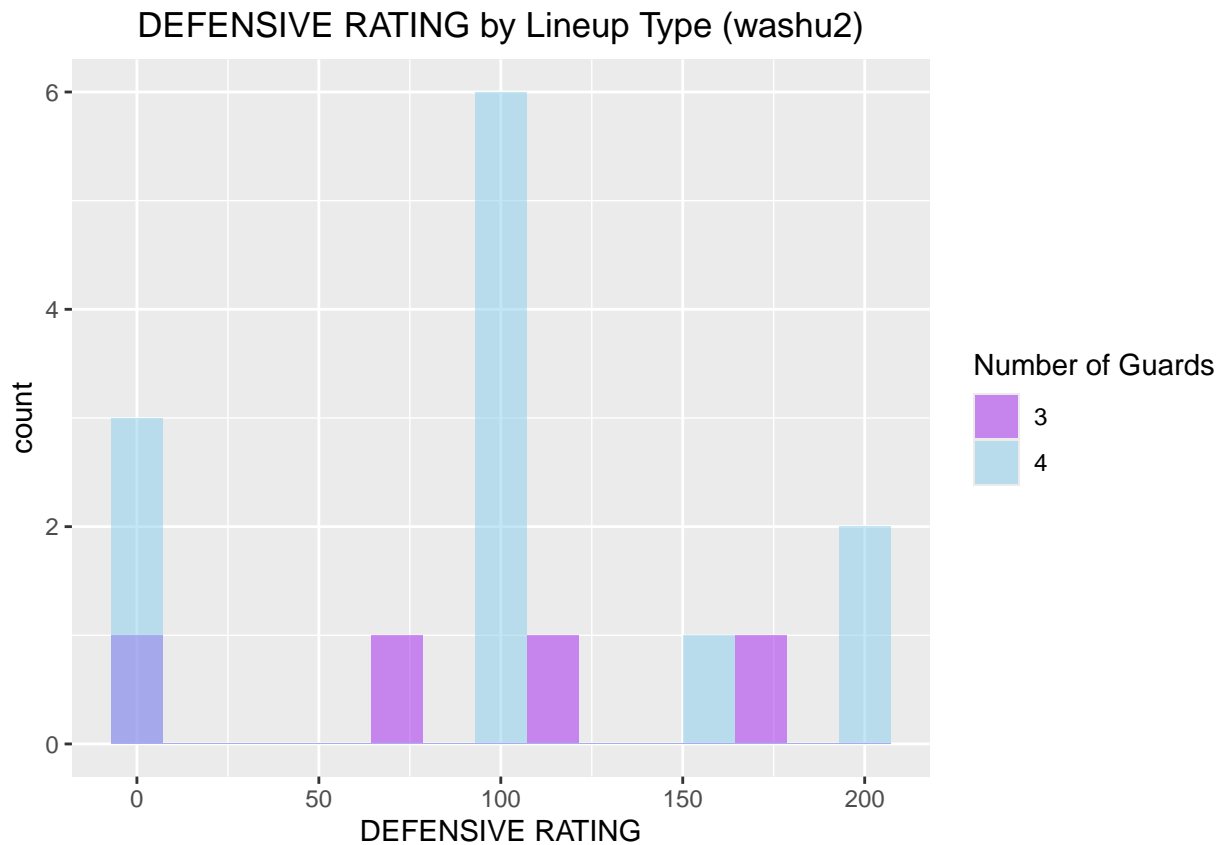
```
## $`3`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  42.86   52.64   66.66   64.98   78.99   83.72
##
```

```
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  40.45   63.83   76.19   93.98  123.08  200.00
```

```
p3m <- median(game$`PACE`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
p4m <- median(game$`PACE`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
p_p <- wilcox.test(`PACE` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% t_f), exact = FALSE)
```

```
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `DEFENSIVE RATING`, fill = factor(`NUMBER OF GUARDS`))) +
  geom_histogram(bins = 30, color = "black", fill = "white")
```

```
## Warning: Removed 1 row containing non-finite outside the scale range
## (`stat_bin()`).
```

```
tapply(game$`DEFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

```
## $`3`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.00  50.00   87.50   85.42 122.92   166.67
##
```

```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
##   0.00  75.00  100.00   97.12 119.94   200.00     1
```

```
dr3m <- median(game$`DEFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
```

```
dr4m <- median(game$`DEFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
```

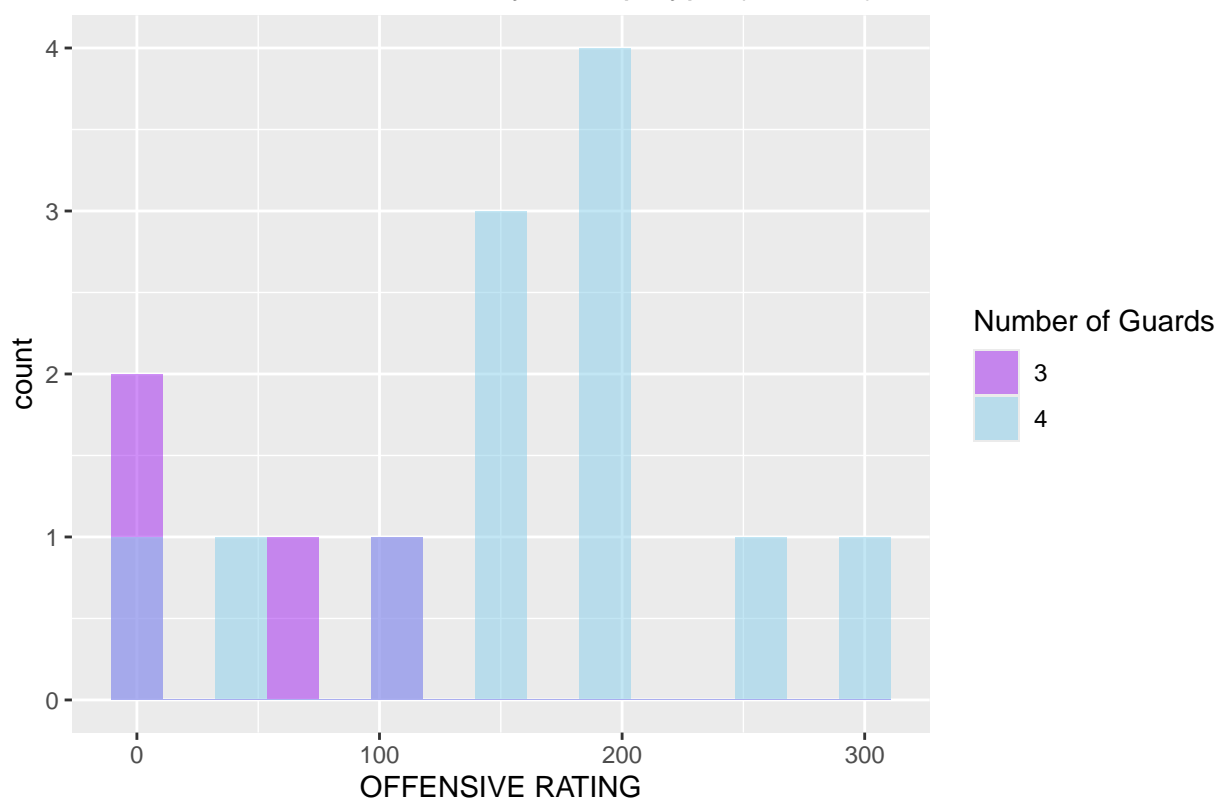
```
dr_p <- wilcox.test(`DEFENSIVE RATING` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% t_f))
```

```
ggplot(data = subset(game, subset = `NUMBER OF GUARDS` %in% t_f), aes(x = `OFFENSIVE RATING`, fill = factor(`NUMBER OF GUARDS`)))
```

```
## Warning: Removed 1 row containing non-finite outside the scale range
```

```
## (`stat_bin()`).
```

OFFENSIVE RATING by Lineup Type (washu2)



```
tapply(game$`OFFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

```
## $`3`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.00   0.00   37.50   43.75   81.25   100.00
##
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      0.0   134.1   175.0   160.7   200.0   300.0      1
```

```
or3m
```

```
## [1] 114.2857
```

```
or4m
```

```
## [1] 134.7594
```

```
or3m <- median(game$`OFFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
or4m <- median(game$`OFFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
or_p <- wilcox.test(`OFFENSIVE RATING` ~ `NUMBER OF GUARDS`, data = subset(game, `NUMBER OF GUARDS` %in% c(3, 4)))
```

```
individual_games <- individual_games %>% add_row(
  `GAME` = g,
  `SCORE` = " ",
  `3G` = n3,
  `4G` = n4,
  `3G MEDIAN NET RATING` = round(nr3m, 2),
  `4G MEDIAN NET RATING` = round(nr4m, 2),
  `NET RATING DIFFERENCE` = round(abs(nr3m - nr4m), 2),
)
```

```

`NET RATING MANN-WHITNEY P-VALUE` = round(nr_p,2),
`3G MEDIAN TRB%` = round(r3m,2),
`4G MEDIAN TRB%` = round(r4m,2),
`TRB% DIFFERENCE` = round(abs(r3m - r4m),2),
`TRB% MANN-WHITNEY P-VALUE` = round(r_p,2),
`3G MEDIAN 3PA/FGA` = round(three3m,2),
`4G MEDIAN 3PA/FGA` = round(three4m,2),
`3PA/FGA DIFFERENCE` = round(abs(three3m - three4m),2),
`3PA/FGA MANN-WHITNEY P-VALUE` = round(three_p,2),
`3G MEDIAN TRUE SHOOTING %` = round(ts3m,2),
`4G MEDIAN TRUE SHOOTING %` = round(ts4m,2),
`TRUE SHOOTING % DIFFERENCE` = round(abs(ts3m - ts4m),2),
`TRUE SHOOTING % MANN-WHITNEY P-VALUE` = round(ts_p,2),
`3G MEDIAN PACE` = round(p3m,2),
`4G MEDIAN PACE` = round(p4m,2),
`PACE DIFFERENCE` = round(abs(p3m - p4m),2),
`PACE MANN-WHITNEY P-VALUE` = round(p_p,2),
`3G MEDIAN DEFENSIVE RATING` = round(dr3m,2),
`4G MEDIAN DEFENSIVE RATING` = round(dr4m,2),
`DEFENSIVE RATING DIFFERENCE` = round(abs(dr3m - dr4m),2),
`DEFENSIVE RATING MANN-WHITNEY P-VALUE` = round(dr_p,2),
`3G MEDIAN OFFENSIVE RATING` = round(or3m,2),
`4G MEDIAN OFFENSIVE RATING` = round(or4m,2),
`OFFENSIVE RATING DIFFERENCE` = round(abs(or3m - or4m),2),
`OFFENSIVE RATING MANN-WHITNEY P-VALUE` = round(or_p,2)
)

# hard coded -> FIX LATER
game_order <- c("allegheeny", "penn state-behrend", "muskingum", "oberlin", "denison", "carlow", "wooster")

individual_games <- individual_games %>% arrange(factor(`GAME`, levels = game_order))

```