

# rochester2 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: 22 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`)

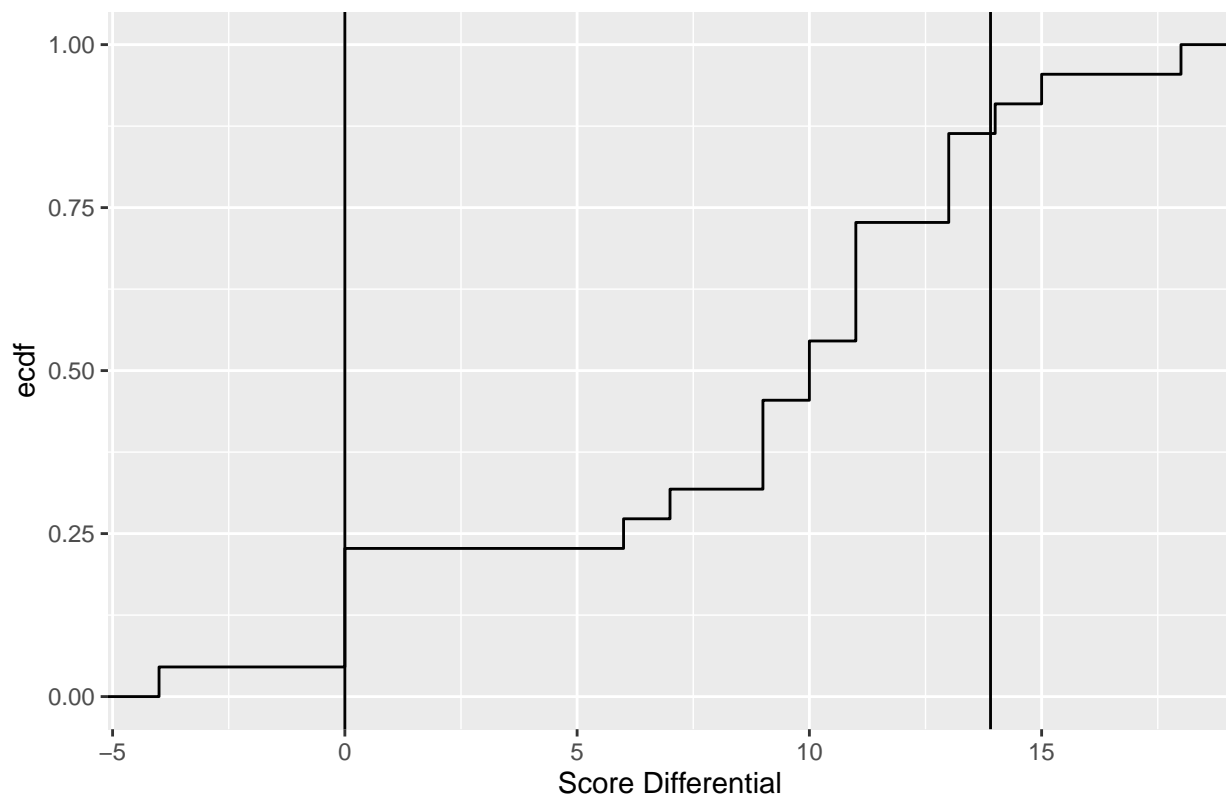
```

```

# 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 =

```

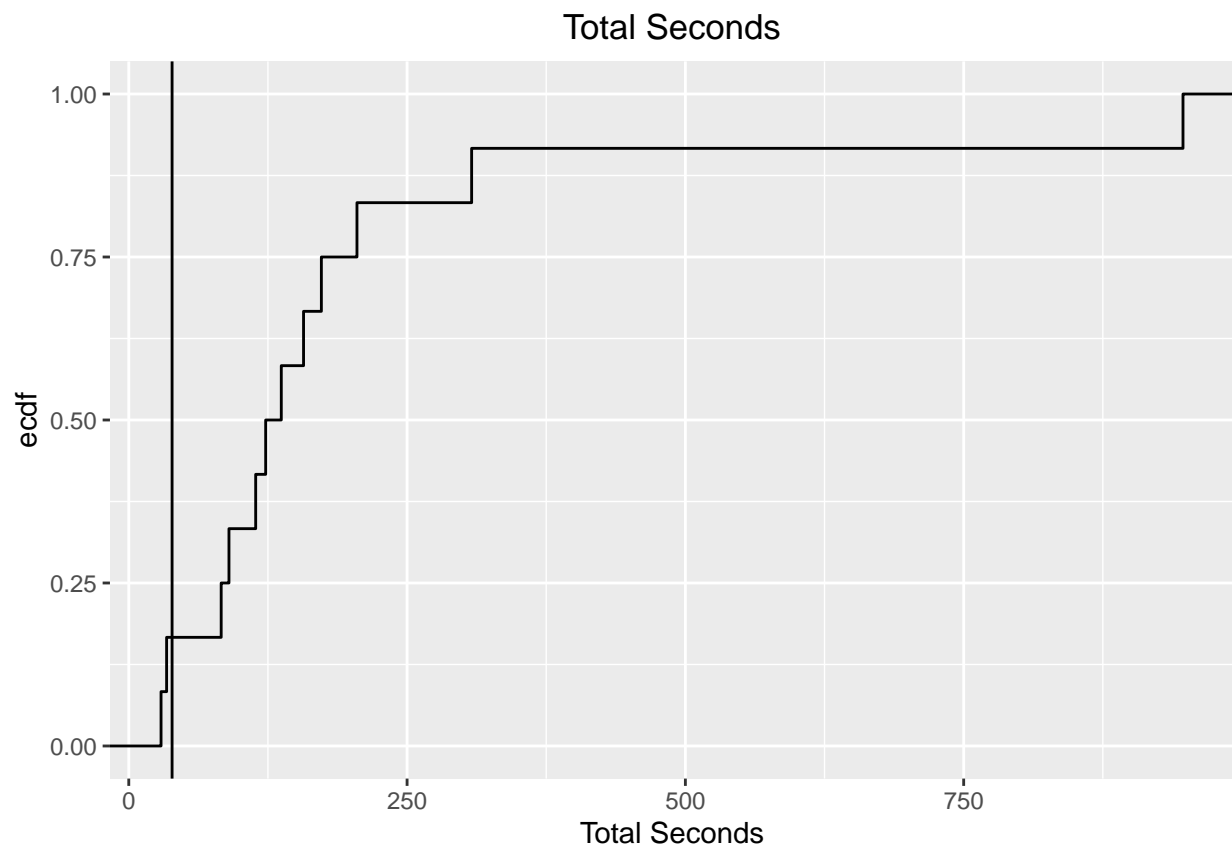
Score Differential



```

# 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
```

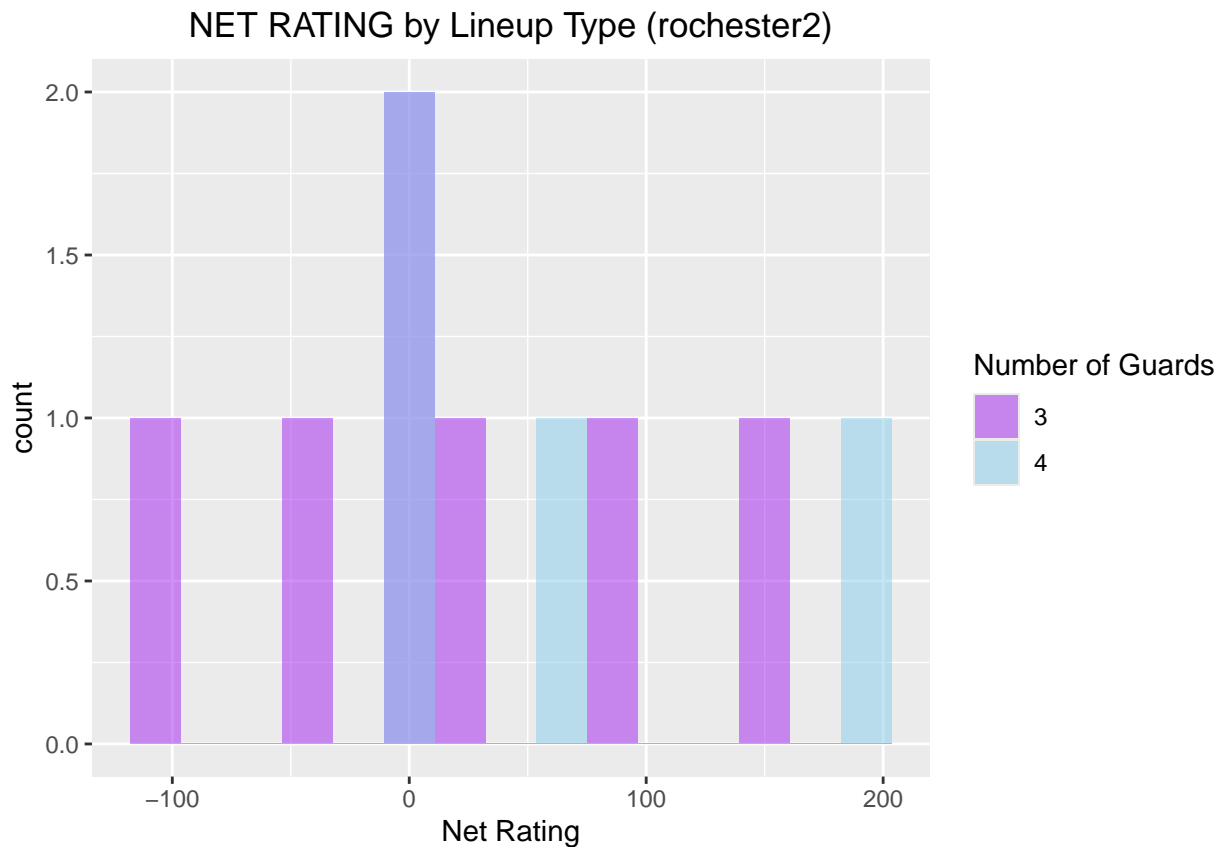
```
## 10%
## 38.9
```

```
#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 1 row 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.
    ## -100.00 -16.67    5.00   16.95   48.48   150.00
    ##
    ## $`4`
    ##      Min. 1st Qu.  Median      Mean 3rd Qu.      Max.      NA's
    ##  -9.524  -2.381   37.500   66.369  106.250  200.000         1
  },
  simplify = FALSE)

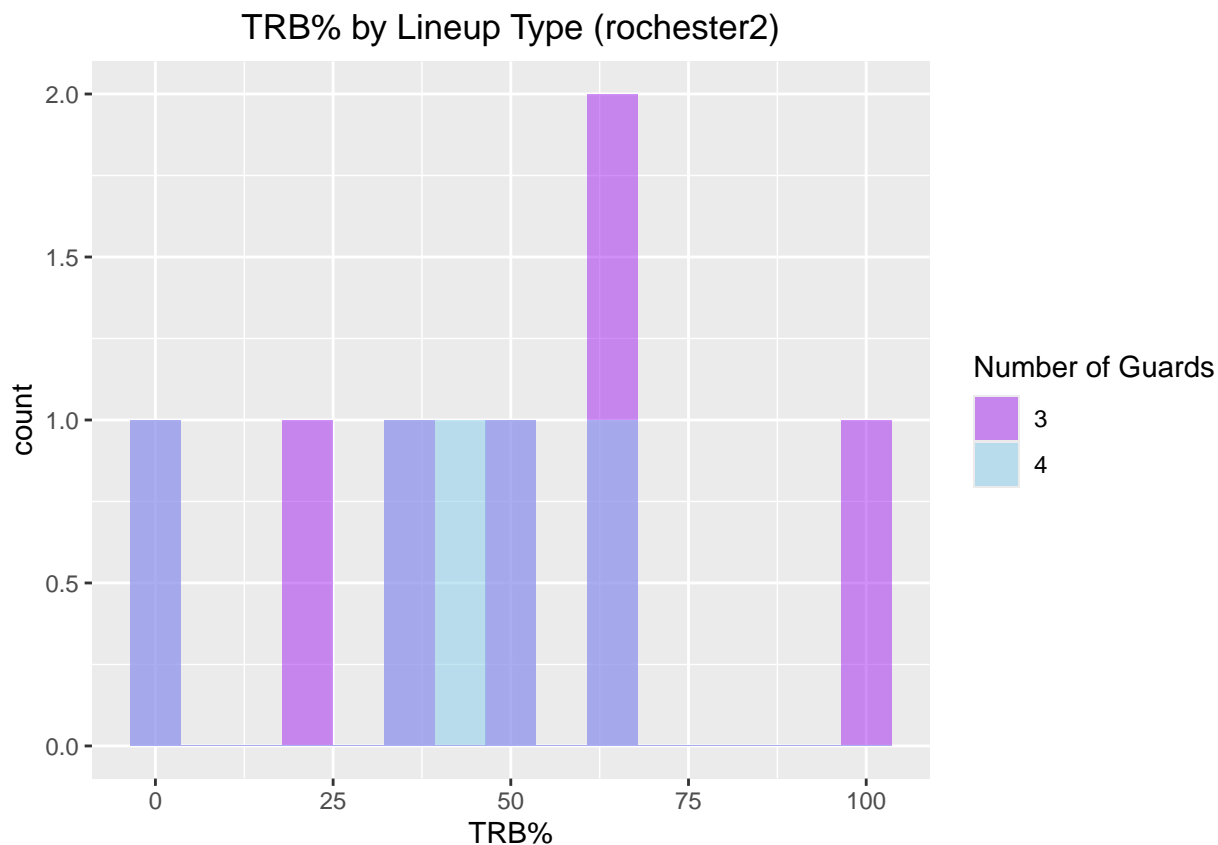
nr3m <- median(game$`NET RATING`[game$`NUMBER OF GUARDS` %in% c(3)], na.rm = TRUE)
nr4m <- median(game$`NET RATING`[game$`NUMBER OF GUARDS` %in% c(4)], na.rm = TRUE)
nr3m

## [1] 5
nr4m

## [1] 37.5
nr_p

## [1] 0.9243755
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`)))
```



```
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.
##      0.00  29.17   50.00   48.57  65.83   100.00
##
```

```
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.00  33.33   40.00   38.00  50.00   66.67
```

```
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] 50
```

```
r4m
```

```
## [1] 40
```

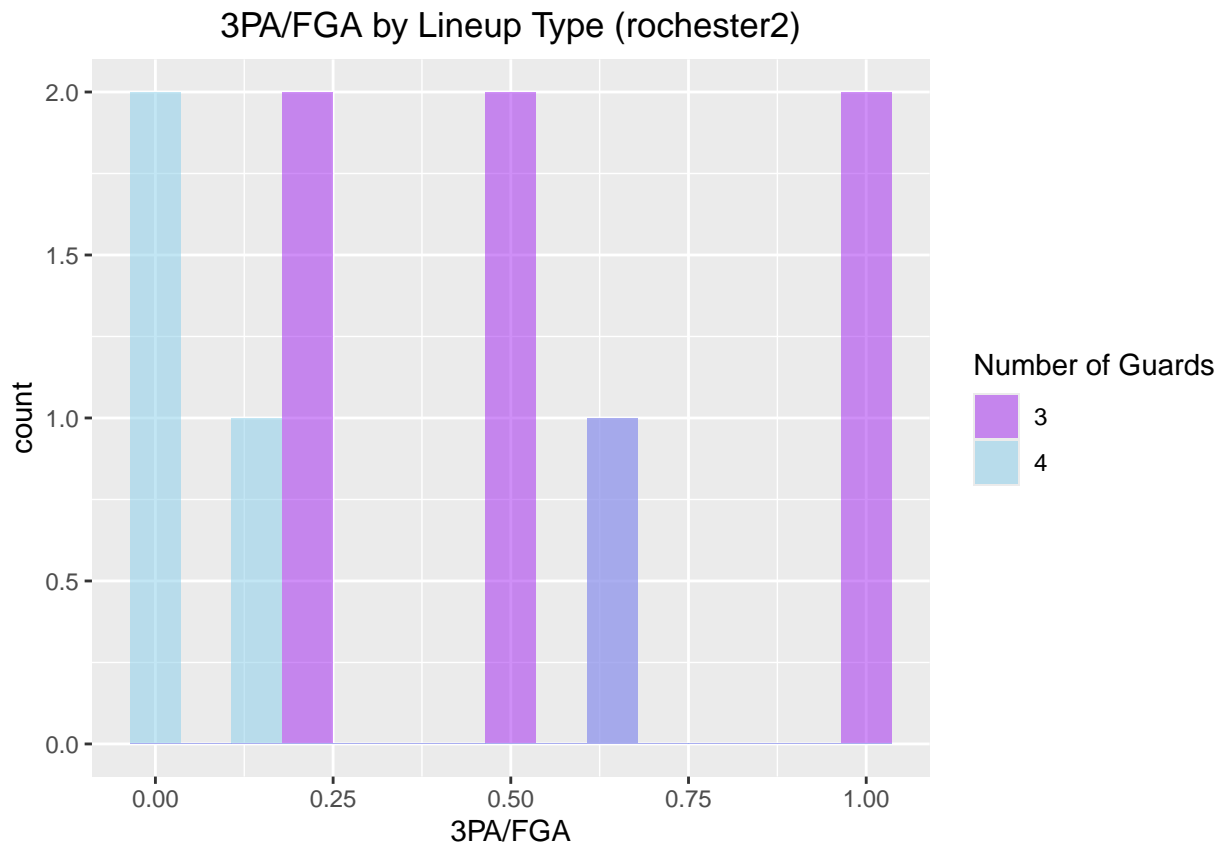
```
r_p
```

```
## [1] 0.7435963
```

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

```
## Warning: Removed 1 row 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`
```

```
## $`3`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.2000 0.3750 0.5000 0.5881 0.8333 1.0000
##
```

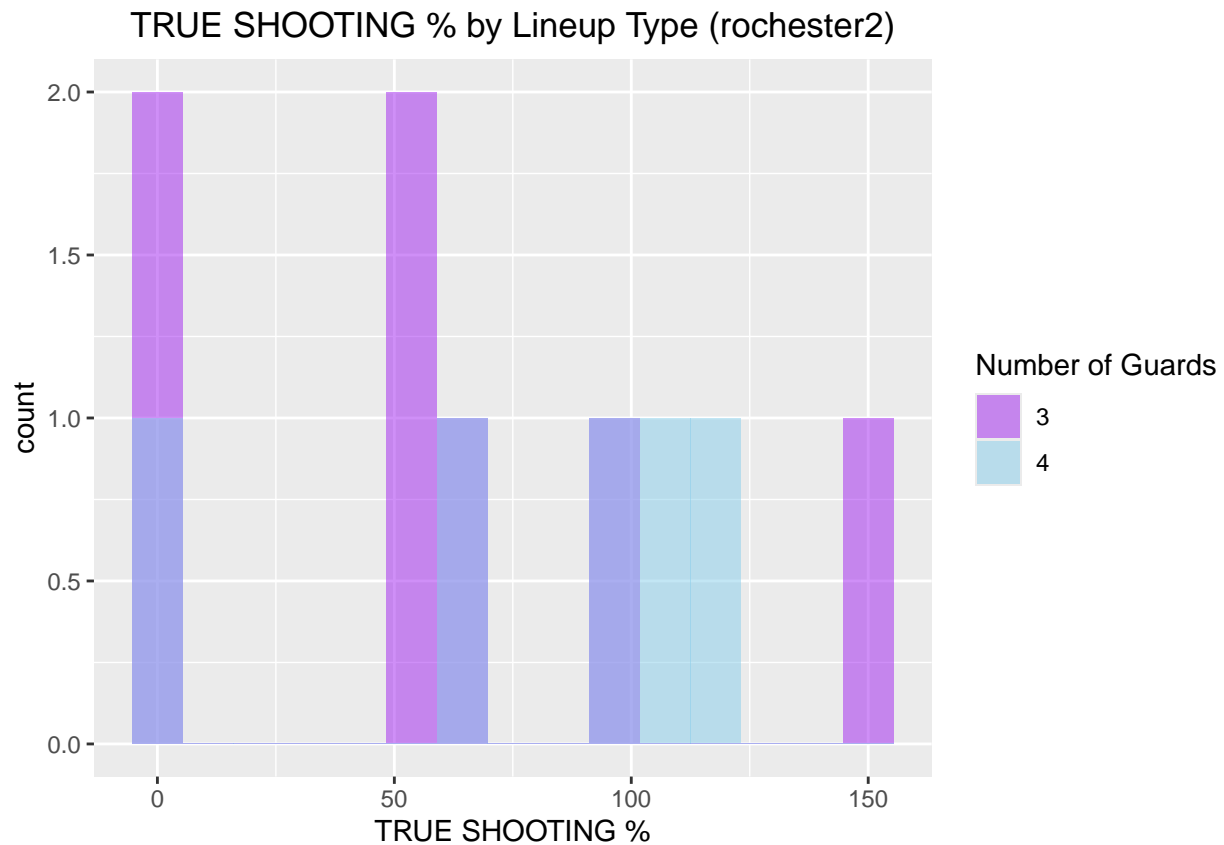
```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
## 0.00000 0.00000 0.05556 0.19444 0.25000 0.66667     1
```

```
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 = fac
```



```
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  25.00   53.19   59.15  80.42  150.00
##
```

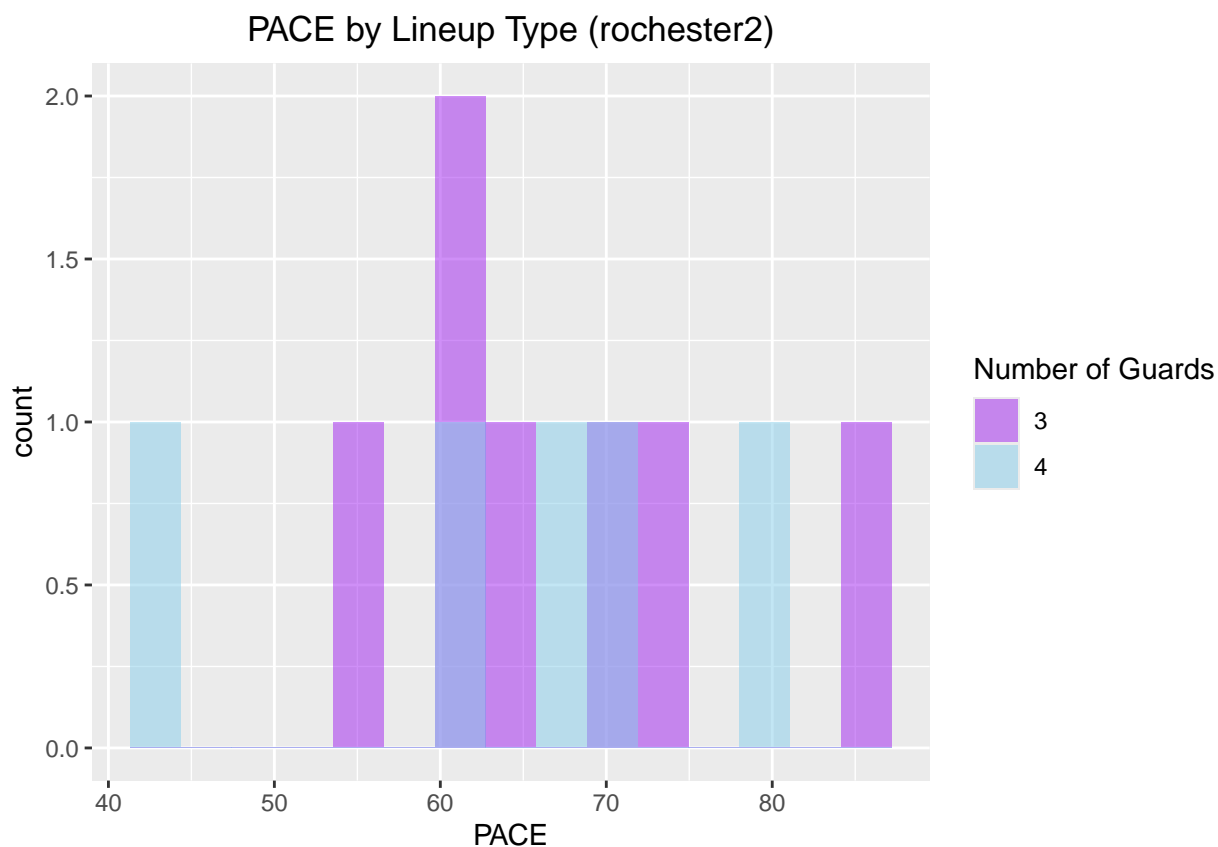
```
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.00  66.67  100.00   77.34 106.38  113.64
```

```
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`)))
```



```
tapply(game$`PACE`[game$`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.
##  55.76  61.79   64.39   67.18  71.18   84.21
##
```

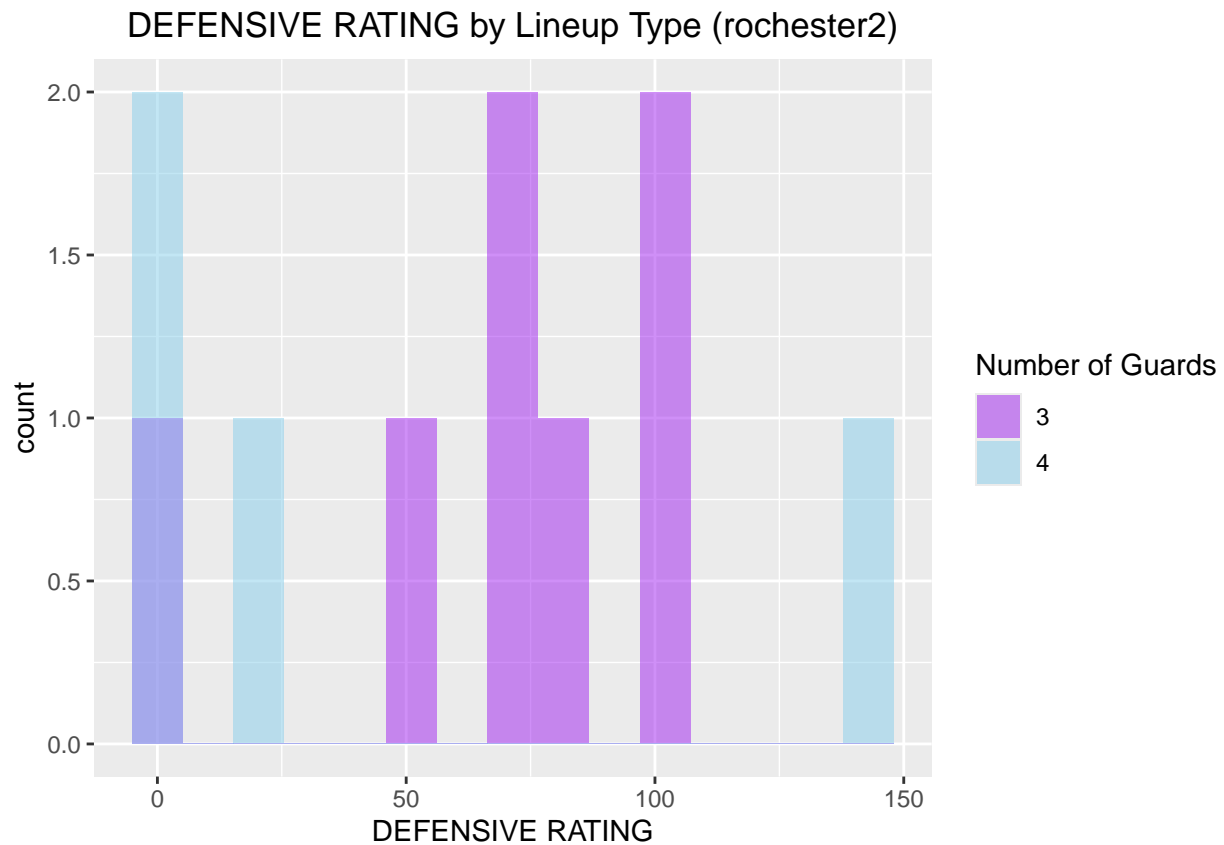
```
## $`4`
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  41.38  62.34   66.67   63.80  70.59   78.05
```

```
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`)))
```

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





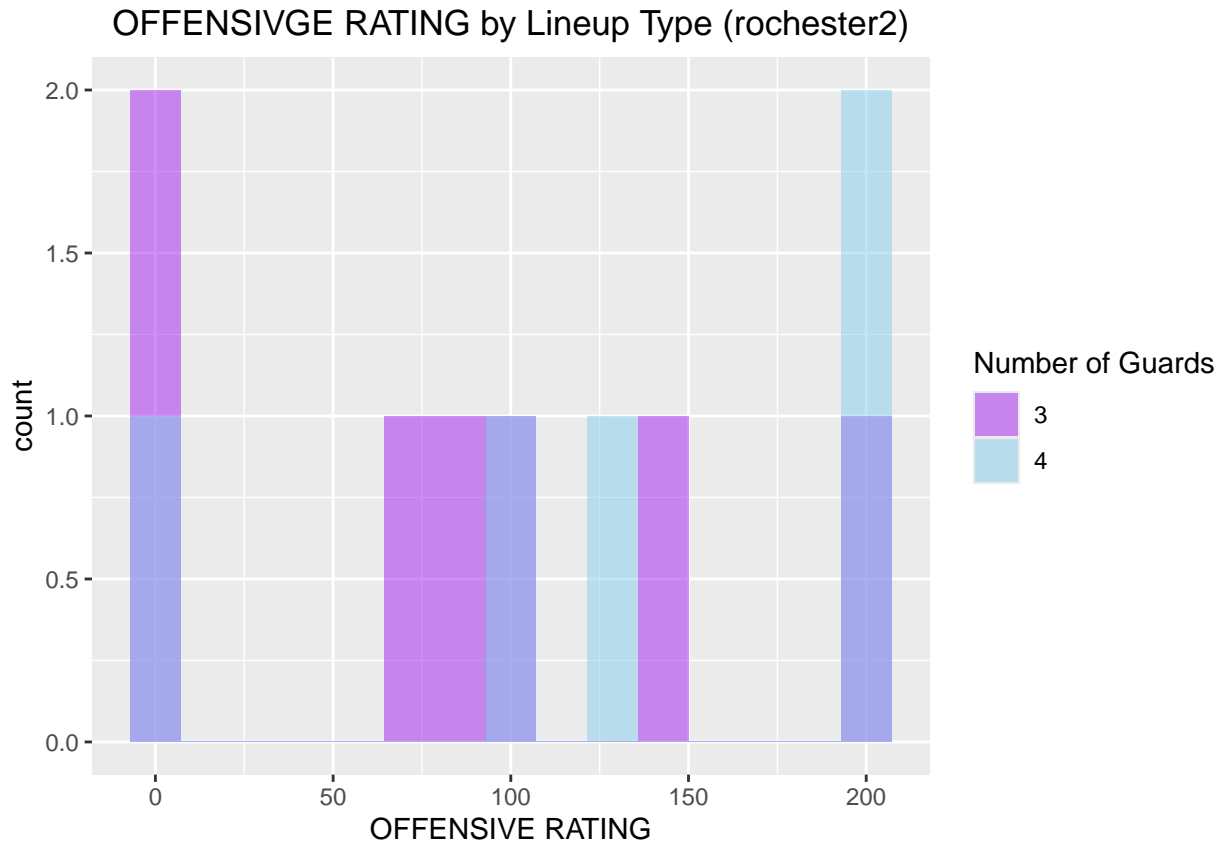
```
tapply(game$`DEFENSIVE RATING`[game$`NUMBER OF GUARDS` %in% t_f], game$`NUMBER OF GUARDS`[game$`NUMBER OF GUARDS` %in% t_f], FUN = function(x) {
  summary(x)
})
```

```
## $`3`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.00  58.33   75.00   68.29  93.18  100.00
##
```

```
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
##      0.00   0.00   12.50   41.96  54.46  142.86      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`))) +
  facet_wrap(~ `NUMBER OF GUARDS`, scales = "free_x") +
  geom_histogram(bins = 10, color = "black", fill = "white") +
  theme_minimal()
```



```
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  33.33   80.00   85.24 125.00   200.00
```

```
## $`4`
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.0   100.0   133.3   126.7  200.0   200.0
```

```
or3m
```

```
## [1] 100
```

```
or4m
```

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

```
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))

```