Generating a novel 'Clutch' metric for NFL wide receivers in the 2022 season

Matt Sheridan

2023-06-16

Personal Introduction

My name is Matthew Sheridan, I'm a recent college graduate, and this is my quick, rough way of putting who I am on paper.

Now this is obviously a rough metric, with MANY assumptions made. I've sparingly worked with NFL pbp data before, so not only is this a showing of my abilities, it's also an exemplification of a journey from knowing little to being more comfortable and familiar.

If you have any further questions, I can be reached at 617-529-2271 or matthewsheridan3627@gmail.com

Project Introduction

Wide receivers. . . are weird. It is quite hard to isolate the impact that an individual receiver has over another because so many factors are different between all of them. Quarterback play, head coach, offensive schemes, opposing corner backs, and many, many more factors are difficult to be isolated with limited resources and computing power.

To evaluate how 'clutch' a receiver is, I want to use two statistics: separation and yards after the catch.

The first large assumption I'll be making is that yards after the catch are one of the most important things receivers can affect individually. This is a large jump, because individual skill contributes a lot to actually getting to a spot to make the catch, but the actual throw is arguably just as crucial. Thus, since a receiver-defender(s) matchup will define the YAC, using the YAC win probability added metric will be a good proxy for actual receiver outcome (considering WPA is a good baseline for evaluating how 'clutch' or important a play is).

Now that that's out of the way, the basis for the project is being able to take the win probability added from YAC and provide in game context for it - that is it say, how important is the game in which a receiver is playing in. A game where the receiver's team has a 40% chance of making the playoffs at, say 6 and 6, is far, far more important than one at say, 14-2, where the team is already in the playoffs and what happens doesn't really affect the team's outcome.

Then, we also want to evaluate the separation a receiver is able to get on average, adding that in as a way to quantify how good their route running is.

Thus, we are evaluating two of the most major parts of a receiver - how good they are at getting open, and what they do once the ball is thrown at them.

With player tracking data, we would be able to further evaluate more advanced things, such as double coverage, speed, etc, however considering that all I currently have access to is a summary of nextgen stats and play by play data, that is a future endeavor.

Ultimately, I intend for this project to showcase the way that I think, my ability to code, and hopefully provide a fun new way to at least think about receivers. Whether the actual statistic is evaluative or would hold up over time is certainly a concern, but for someone who has basically never worked with the pbp/nextgen data before, this is a start. ## Data / Tools

In this analysis, we will be weighting YAC WPA by a function depending on the team's rough playoff probability in a specific game. For example, if the Dolphins are 3-3 and heave a 42% chance of making the playoffs, then Tyreek Hill breaking 3 tackles for a 55 yard touchdown and adding, say, 9 percent win probability to his team is ultimately more clutch than, say, Darnell Mooney doing the exact same thing but for the 1-5 bears. This relies on the assumption that the players are aware of the pressure that is on them and that they know something about how if they improve to 4-3, their playoff chances are far higher.

Therefore, we will be using this chart to determine playoff probabilities, as it is a bit beyond my current scope to be able to incorporate every factor of playoff probability:

```
library(tidyverse)
library(ggrepel)
library(nflreadr)
library(nflplotR)
library(coefplot)
library(dplyr)
library(rvest)
options(scipen=9999)
```

```
playoff_probs <- (matrix(c(0.44, 0.32, 0.22, 0.14, 0.08,0.04,0.02,0.01,0,0,0,0,0,0,0,0,0,0,
                        0.55,0.43,0.32,0.21,0.13,0.07,0.03,0.01,0,0,0,0,0,0,0,0,0,0,
                        0.67,0.55,0.42,0.3,0.2,0.11,0.06,0.02,0.01,0,0,0,0,0,0,0,0,0,
                        0.76,0.67,0.55,0.42,0.29,0.18,0.1,0.04,0.01,0,0,0,0,0,0,0,0,0,
                        0.85,0.77,0.67,0.54,0.41,0.28,0.16,0.08,0.03,0.01,0,0,0,0,0,0,0,0,
                        0.91,0.85,0.77,0.67,0.54,0.39,0.25,0.13,0.05,0.01,0,0,0,0,0,0,0,0,
                        0.95,0.92,0.87,0.78,0.67,0.53,0.37,0.22,0.09,0.03,0,0,0,0,0,0,0,0,
                        0.98,0.96,0.93,0.88,0.8,0.69,0.53,0.34,0.17,0.04,0.01,0,0,0,0,0,0,0,
                        0.99,0.98,0.97,0.95,0.9,0.82,0.7,0.52,0.28,0.11,0,0,0,0,0,0,0,0,
                        1,1,0.99,0.98,0.96,0.93,0.86,0.74,0.5,0,0,0,0,0,0,0,0,0,
                        1,1,1,1,0.99,0.98,0.97,0.91,0,0,0,0,0,0,0,0,0,0,
                        1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,
                        1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,
                        1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                        1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
                        nrow = 18, ncol = 18))

rownames(playoff_probs) <- 0:17
colnames(playoff_probs) <- 0:17

payoff_function <- function(x){
#  ifelse(x > 0.5, (1.5 - x + 0.5)^2, (x + 1)^2)
  -4*((x-0.5)^2) + 1
}

playoff_probs
```

```
##        0     1     2     3     4     5     6     7     8     9    10  11  12  13  14  15  16  17
```

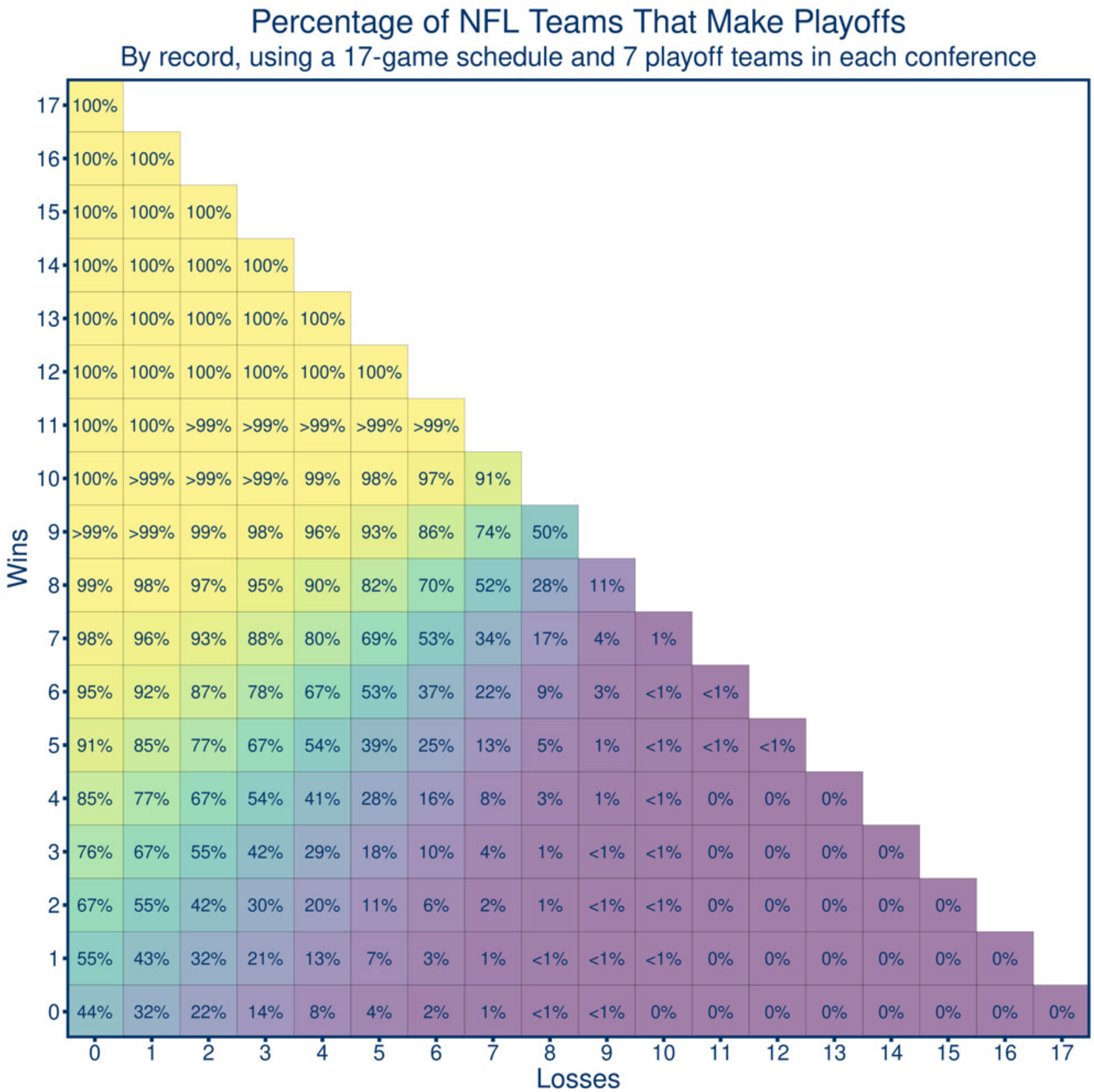
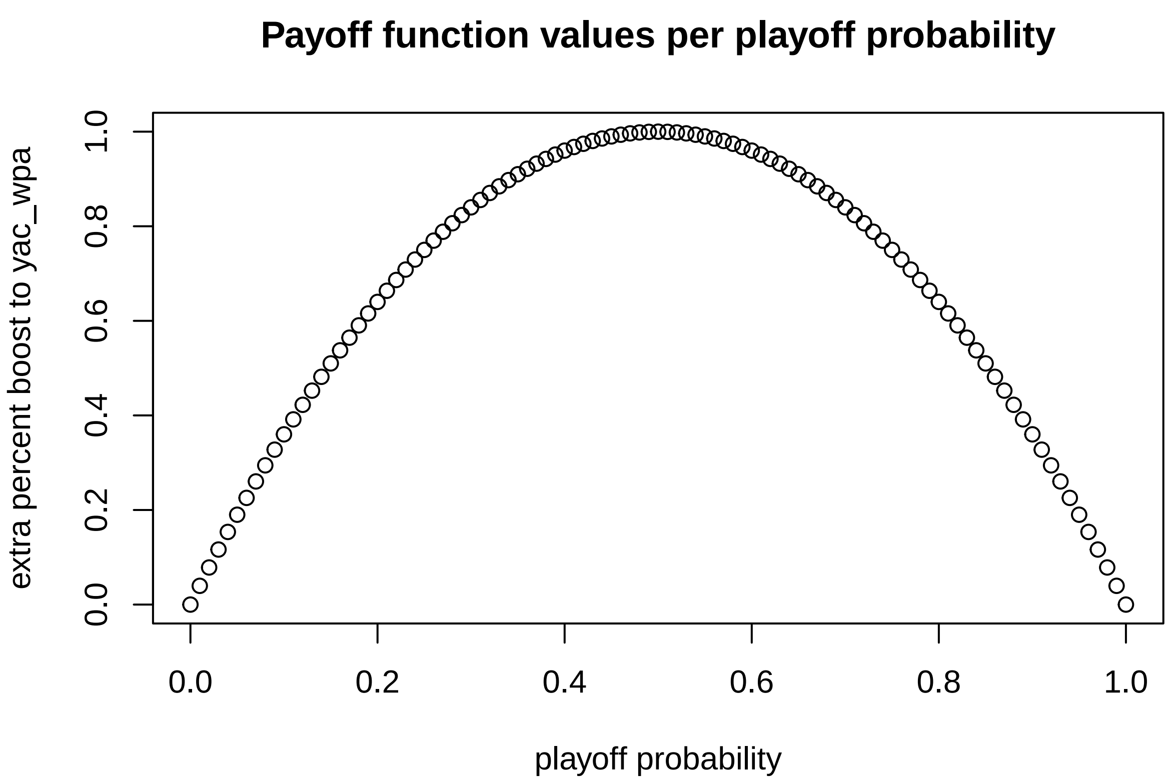


Figure 1: Playoff Probabilities by Record

```
## 0  0.44 0.55 0.67 0.76 0.85 0.91 0.95 0.98 0.99 1.00 1.00  1  1  1  1  1  1  1
## 1  0.32 0.43 0.55 0.67 0.77 0.85 0.92 0.96 0.98 1.00 1.00  1  1  1  1  1  1  0
## 2  0.22 0.32 0.42 0.55 0.67 0.77 0.87 0.93 0.97 0.99 1.00  1  1  1  1  1  0  0
## 3  0.14 0.21 0.30 0.42 0.54 0.67 0.78 0.88 0.95 0.98 1.00  1  1  1  1  0  0  0
## 4  0.08 0.13 0.20 0.29 0.41 0.54 0.67 0.80 0.90 0.96 0.99  1  1  1  0  0  0  0
## 5  0.04 0.07 0.11 0.18 0.28 0.39 0.53 0.69 0.82 0.93 0.98  1  1  0  0  0  0  0
## 6  0.02 0.03 0.06 0.10 0.16 0.25 0.37 0.53 0.70 0.86 0.97  1  0  0  0  0  0  0
## 7  0.01 0.01 0.02 0.04 0.08 0.13 0.22 0.34 0.52 0.74 0.91  0  0  0  0  0  0  0
## 8  0.00 0.00 0.01 0.01 0.03 0.05 0.09 0.17 0.28 0.50 0.00  0  0  0  0  0  0  0
## 9  0.00 0.00 0.00 0.00 0.01 0.01 0.03 0.04 0.11 0.00 0.00  0  0  0  0  0  0  0
## 10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.01 0.00 0.00 0.00  0  0  0  0  0  0  0
## 11 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 12 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 14 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 15 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 16 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
## 17 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  0  0  0  0  0  0  0
```

```r
plot(seq(0,1,by = 0.01), payoff_function(seq(0,1,by = 0.01)),
     main = "Payoff function values per playoff probability",
     xlab = 'playoff probability',
     ylab = 'extra percent boost to yac_wpa')
```



```r
get_wr_data <- function(YEARS){

  schedule <- filter(load_schedules(seasons = 2022), game_type == "REG")

  rolling_wins <- matrix(nrow = 32, ncol = 18)

  teams <- sort(unique(schedule$away_team))

  for (i in 1:32){
  cur_team <- teams[i]
    for (j in 1:18){
      rolling_wins[i,j] <- ifelse(j == 1, 0, rolling_wins[i,j-1] )
      tmp <- filter(schedule, (home_team == cur_team) | away_team == cur_team, week == j-1)

      if (nrow(tmp) != 0){
        if (tmp$away_team[1] == cur_team){
            rolling_wins[i,j] <- ifelse(j>1,
                                        (1 * (tmp$result[1] < 0)) + rolling_wins[i,(j-1)],
                                        0)
        }
        else if (tmp$home_team[1] == cur_team){
           rolling_wins[i,j] <- ifelse(j>1,
                                        (1 * (tmp$result[1] > 0)) + rolling_wins[i,(j-1)],
                                        0)
        }

      }
    }
  }

  rolling_wins <- as.data.frame(rolling_wins)
  rownames(rolling_wins) <- teams
  colnames(rolling_wins) <- 1:18


  data = load_pbp(YEARS)
  data <- data %>%
    left_join(load_teams(), by = c('posteam' = 'team_abbr'))

  contracts_with_id <- filter(load_contracts(), is_active == T) %>% inner_join(load_players()[,c('gsis_

  nextgen <- filter(load_nextgen_stats(seasons = YEARS,stat_type = c("receiving"),
                     file_type = getOption("nflreadr.prefer", default = "rds")), week == 0)

  #Assessing WPA by receiver

  passes <- filter(data, pass_attempt == 1, season_type == "REG", !is.na(yards_after_catch))

  i <- cbind(match(passes$posteam, rownames(rolling_wins)),
        match(passes$week, colnames(rolling_wins)))

  passes <- cbind(rolling_win_ct = rolling_wins[i], passes)
```

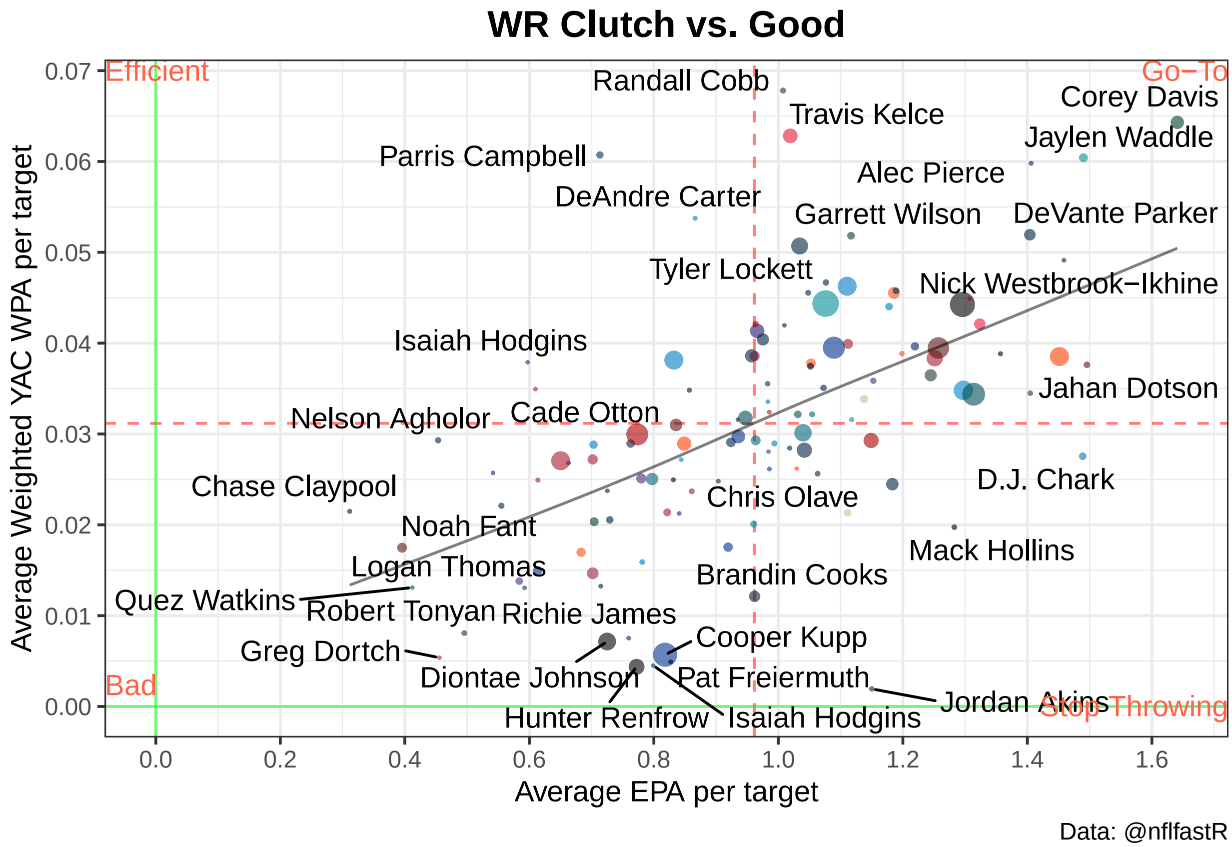
```
  i <- cbind(match(passes$week - 1 - passes$rolling_win_ct, rownames(playoff_probs)),
        match(passes$rolling_win_ct, colnames(playoff_probs)))

  passes <- cbind(playoff_prob = playoff_probs[i],passes)

  pass_counts <- passes %>% count(receiver_player_name, posteam, receiver_player_id, team_color)

#  avg_wpa_wrs <- setNames(aggregate(wpa ~ receiver_player_name + posteam + receiver_player_id, data = p

  avg_epa_wrs <- aggregate(epa ~ receiver_player_name + posteam + receiver_player_id, data = passes, FUN

#  tot_epa_wrs <- aggregate(epa ~ receiver_player_name + posteam + receiver_player_id, data = passes, FU

#  colnames(tot_epa_wrs)[4] = "tot_epa"

#  avg_cpoe_wrs <- aggregate(cpoe ~ receiver_player_name + posteam + receiver_player_id, data = passes,

  passes$w_yac_wpa <- passes$yac_wpa * (1+payoff_function(passes$playoff_prob))


  avg_w_yacwpa_wrs <- setNames(aggregate(w_yac_wpa ~ receiver_player_name + posteam + receiver_player_id

  avg_uw_yacwpa_wrs <- setNames(aggregate(yac_wpa ~ receiver_player_name + posteam + receiver_player_id

#  avg_yac_wrs <- aggregate(yards_after_catch ~ receiver_player_name + posteam + receiver_player_id, dat

#  wr_data <- avg_wpa_wrs %>% merge(pass_counts, keep=F) %>% merge(avg_epa_wrs, keep=F) %>% merge(avg_cp

    wr_data <- avg_epa_wrs %>% merge(pass_counts, keep=F) %>% merge(avg_w_yacwpa_wrs, keep=F) %>% merge

  team_pa <- aggregate(n~posteam, data = wr_data, FUN=sum)

  wr_data <- wr_data %>% merge(team_pa, by='posteam', suffixes = c("_player", "_team"))

  wr_data <- wr_data %>% inner_join(contracts_with_id, by = join_by(receiver_player_id == gsis_id))

  #wr_data <- filter(wr_data, n_player>50)

  ng_wr_clean <- wr_data %>% inner_join(nextgen, by = join_by(receiver_player_id==player_gsis_id))

  return(ng_wr_clean)
}

data_2022 = get_wr_data(2022)
data_2021 = get_wr_data(2021)
data_2020 = get_wr_data(2020)
```

```
df <- data.frame(x2=rnorm(100),y2=rnorm(100))

annotations <- data.frame(
        xpos = c(-Inf,-Inf,Inf,Inf),
        ypos =  c(-Inf, Inf,-Inf,Inf),
        annotateText = c("Bad","Efficient"
```

```r
                        ,"Stop Throwing","Go-To"),
        hjustvar = c(0,0,1,1) ,
        vjustvar = c(-2,1,-1,1)) #<- adjust

data_2022 %>%
  ggplot(aes(y = w_yac_wpa, x = epa)) +
  #horizontal line with mean EPA
  geom_hline(yintercept = mean(data_2022$w_yac_wpa), color = "red", linetype = "dashed", alpha=0.5) +
  #vertical line with mean CPOE
  geom_vline(xintercept =  mean(data_2022$epa), color = "red", linetype = "dashed", alpha=0.5) +
  geom_hline(yintercept = 0, color = "green", linetype = "solid", alpha=0.5) +
  #vertical line with mean CPOE
  geom_vline(xintercept =  0, color = "green", linetype = "solid", alpha=0.5) +
  #add points for the QBs with the right colors
  #cex controls point size and alpha the transparency (alpha = 1 is normal)
  geom_point(color = data_2022$team_color, cex=data_2022$apy / mean(data_2022$apy), alpha = .6) +
  #add names using ggrepel, which tries to make them not overlap
  geom_text_repel(aes(label=player), max.overlaps = 12) +
  #add a smooth line fitting wpa + epa
  stat_smooth(geom='line', alpha=0.5, se=FALSE, method='gam')+
  #titles and caption
  labs(x = "Average EPA per target",
       y = "Average Weighted YAC WPA per target",
       title = "WR Clutch vs. Good",
       caption = "Data: @nflfastR") +
  #uses the black and white ggplot theme
  theme_bw() +
  #center title with hjust = 0.5
  theme(
    plot.title = element_text(size = 14, hjust = 0.5, face = "bold")
  ) +
  #make ticks look nice
  #if this doesn't work, `install.packages('scales')`
  scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
  geom_text(data=annotations,aes(x=xpos,y=ypos,hjust=hjustvar,vjust=vjustvar,label=annotateText), color
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```

```
## Warning: ggrepel: 86 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```r
data_2022$clutch <- scale(data_2022$w_yac_wpa) +
  scale(data_2022$avg_separation)

data_2021$clutch <- scale(data_2021$w_yac_wpa) +
  scale(data_2021$avg_separation)

data_2020$clutch <- scale(data_2020$w_yac_wpa) +
  scale(data_2020$avg_separation)

cor(data_2022[, colnames(data_2022) %in% c("w_yac_wpa", "avg_separation")])
```

```
##                 w_yac_wpa avg_separation
## w_yac_wpa       1.0000000     -0.2650688
## avg_separation -0.2650688      1.0000000
```

```r
dta <- data_2022 %>%
  inner_join(data_2021, by = join_by(receiver_player_id), suffix = c("_22", "_21")) %>%
  inner_join(data_2020, by = join_by(receiver_player_id), suffix = c("", "_20"))
```

```
## Warning in inner_join(., data_2021, by = join_by(receiver_player_id), suffix = c("_22", : Detected a
## i Row 2 of `x` matches multiple rows in `y`.
## i Row 36 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
## Warning in inner_join(., data_2020, by = join_by(receiver_player_id), suffix = c("", : Detected an u
## i Row 70 of `x` matches multiple rows in `y`.
## i Row 79 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
cor(dta$clutch_22, dta$clutch_21)
```

```
##           [,1]
## [1,] 0.4259905
```

```
cor(dta$clutch_21, dta$clutch)
```

```
##           [,1]
## [1,] 0.3933053
```

```
cor(dta$clutch_22, dta$clutch)
```

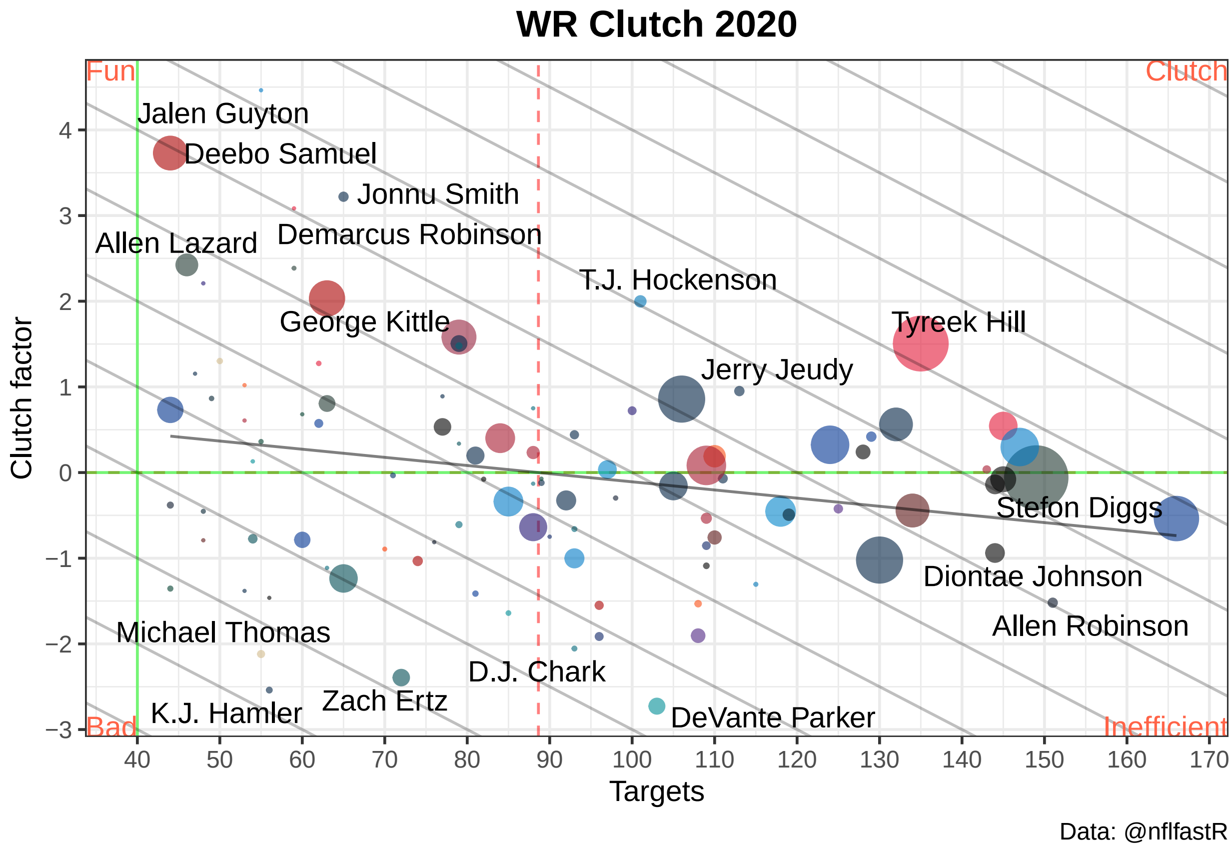
```
##           [,1]
## [1,] 0.3328056
```

```
df <- data.frame(x2=rnorm(100),y2=rnorm(100))

annotations <- data.frame(
        xpos = c(-Inf,-Inf,Inf,Inf),
        ypos =  c(-Inf, Inf,-Inf,Inf),
        annotateText = c("Bad","Fun"
                        ,"Inefficient","Clutch"),
        hjustvar = c(0,0,1,1) ,
        vjustvar = c(0,1,0,1)) #<- adjust

plot_clutch <- function(dataframe, yr){
  dataframe %>%
    ggplot(aes(y = clutch, x = targets)) +
    #horizontal line with mean EPA
    geom_hline(yintercept = mean(dataframe$clutch), color = "red", linetype = "dashed", alpha=0.5) +
    #vertical line with mean CPOE
    geom_vline(xintercept =  mean(dataframe$targets), color = "red", linetype = "dashed", alpha=0.5) +
    geom_hline(yintercept = 0, color = "green", linetype = "solid", alpha=0.5) +
    #vertical line with mean CPOE
    geom_vline(xintercept =  40, color = "green", linetype = "solid", alpha=0.5) +
    #add points for the QBs with the right colors
    #cex controls point size and alpha the transparency (alpha = 1 is normal)
    geom_point(color = dataframe$team_color, cex=dataframe$value / median(dataframe$value), alpha = .6)
    #add names using ggrepel, which tries to make them not overlap
    geom_text_repel(aes(label=player), max.overlaps = 5) +
    #add a smooth line fitting wpa + epa
    stat_smooth(geom='line', alpha=0.5, se=FALSE, method='lm')+
    geom_abline(slope = -.05, intercept = (15:-5), alpha = .25)+
    #titles and caption
    labs(x = "Targets",
```

```
        y = "Clutch factor",
        title = paste("WR Clutch", yr),
        caption = "Data: @nflfastR") +
    #uses the black and white ggplot theme
    theme_bw() +
    #center title with hjust = 0.5
    theme(
      plot.title = element_text(size = 14, hjust = 0.5, face = "bold")
    ) +
    #make ticks look nice
    #if this doesn't work, `install.packages('scales')`
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
    geom_text(data=annotations,aes(x=xpos,y=ypos,hjust=hjustvar,vjust=vjustvar,label=annotateText), col
}

plot_clutch(data_2020, "2020")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

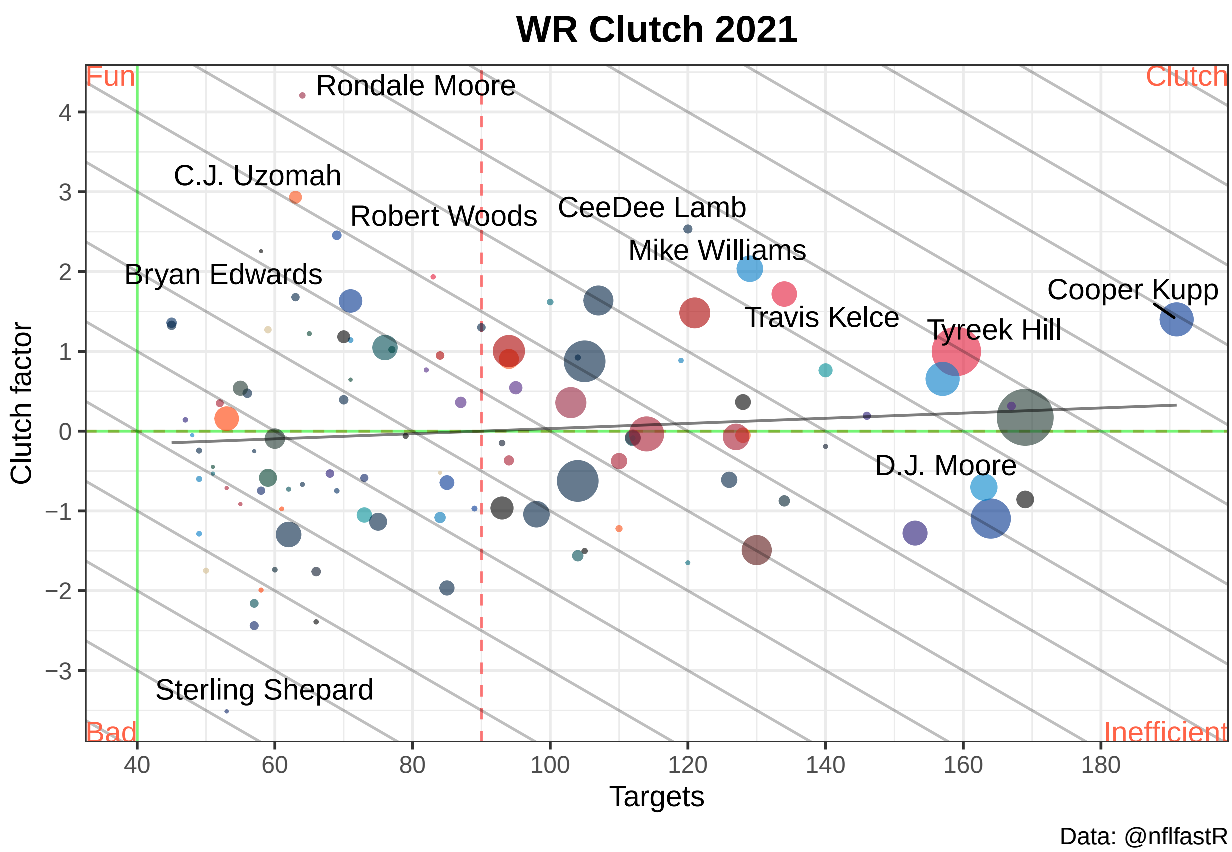
```
## Warning: ggrepel: 84 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
plot_clutch(data_2021, "2021")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

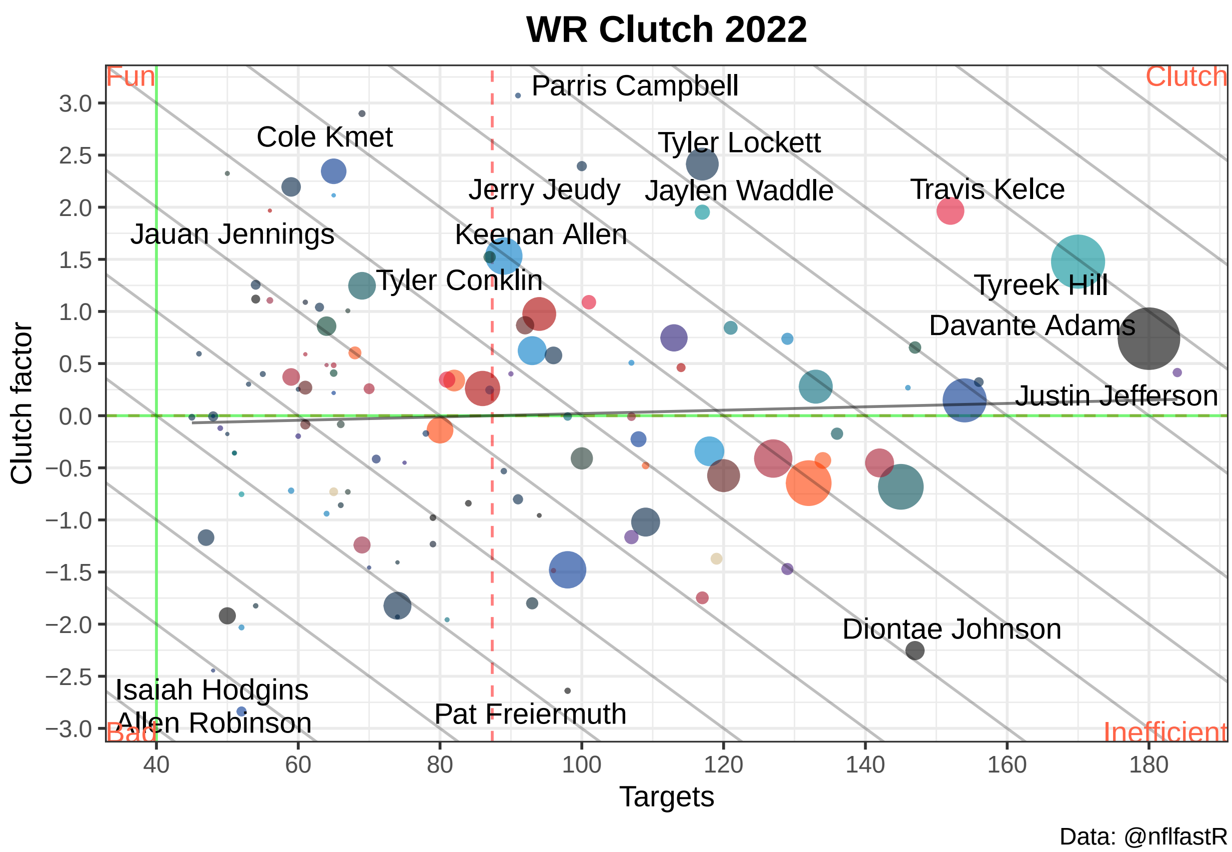
```
## Warning: ggrepel: 95 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



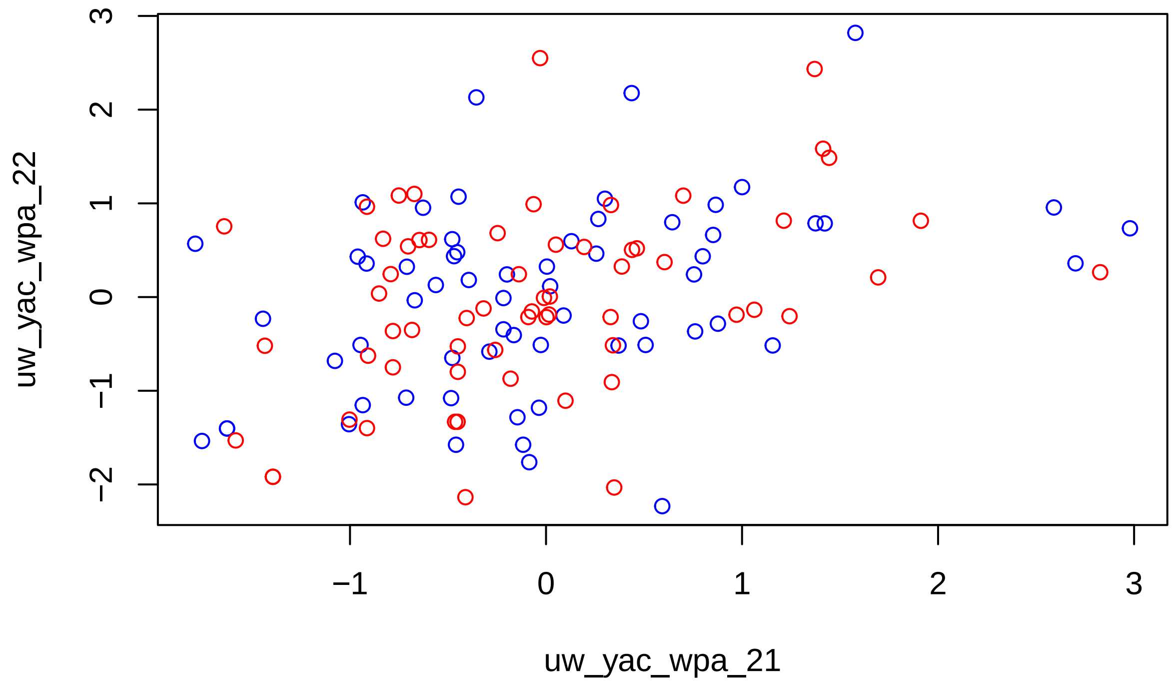

```
plot_clutch(data_2022, "2022")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: ggrepel: 102 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
dta[,colnames(dta) %in% c("w_yac_wpa_22", "uw_yac_wpa_22",
                          "uw_yac_wpa_21", "w_yac_wpa_21",
                          "uw_yac_wpa", "w_yac_wpa")] <-scale(dta[,colnames(dta) %in% c("w_yac_wpa_
                          "uw_yac_wpa_21", "w_yac_wpa_21",
                          "uw_yac_wpa", "w_yac_wpa")])
plot(uw_yac_wpa_22~uw_yac_wpa_21, dta, col = 'blue')
points(w_yac_wpa_22~w_yac_wpa_21, dta, col = 'red')
```



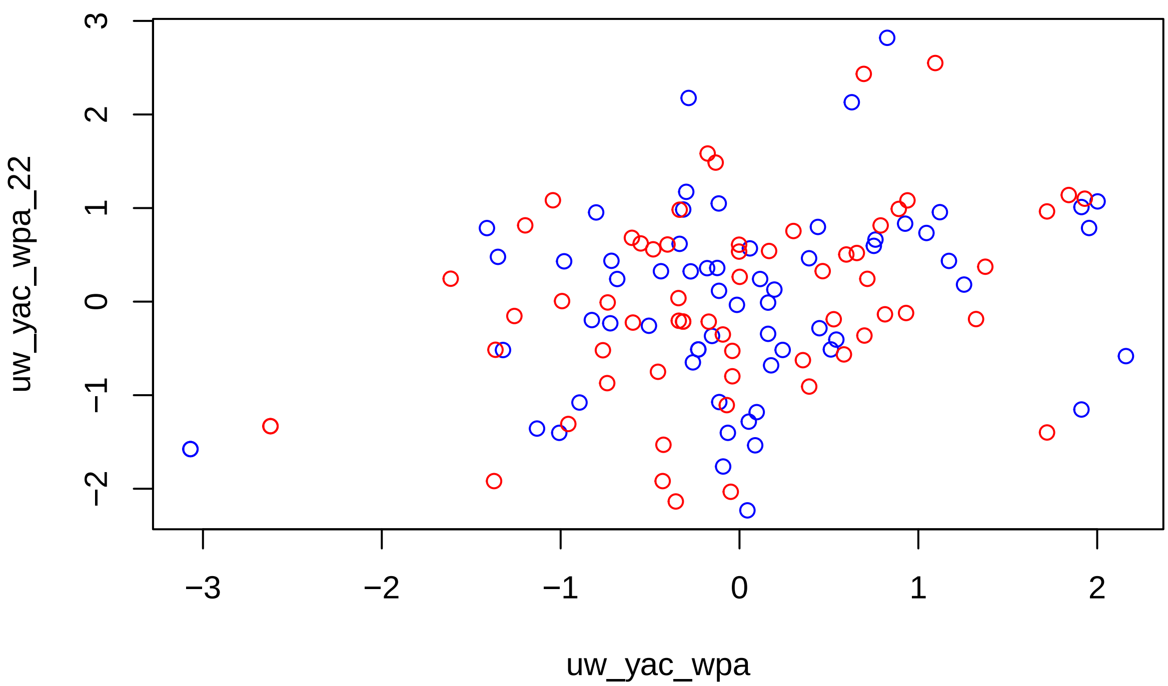

```
cor(dta$uw_yac_wpa_22,dta$uw_yac_wpa_21)
```

```
## [1] 0.377042
```

```
cor(dta$w_yac_wpa_22,dta$w_yac_wpa_21)
```

```
## [1] 0.4109587
```

```
plot(uw_yac_wpa_22~uw_yac_wpa, dta, col = 'blue')
points(w_yac_wpa_22~w_yac_wpa, dta, col = 'red')
```



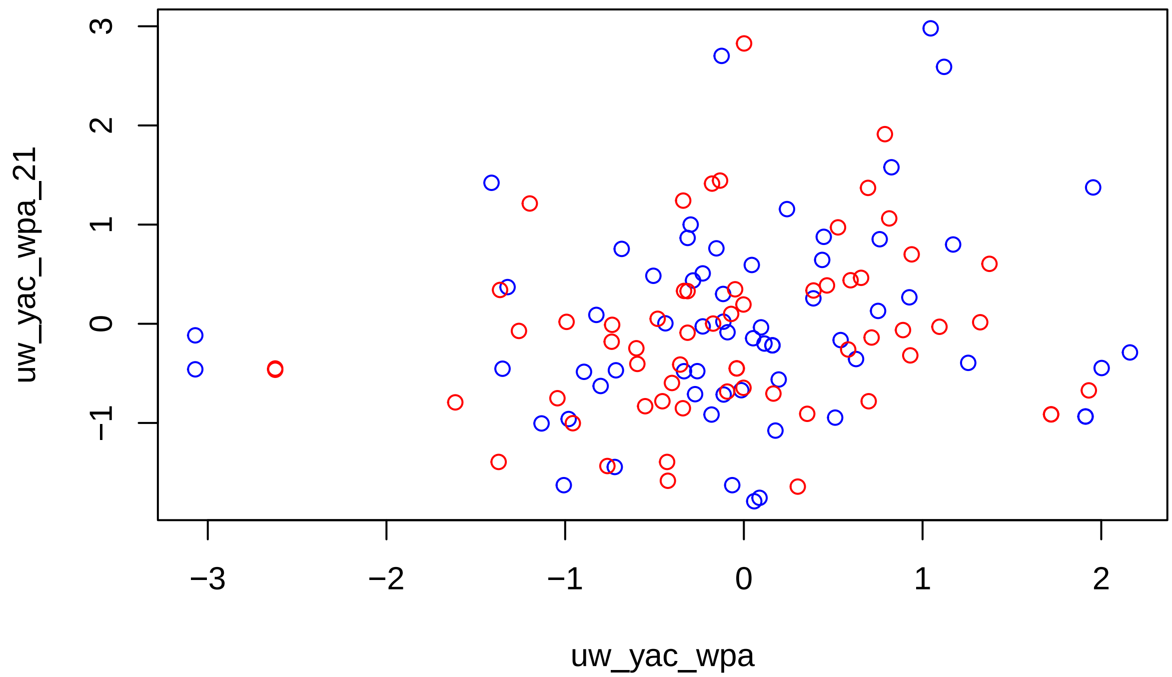

```
cor(dta$uw_yac_wpa_22,dta$uw_yac_wpa)
```

```
## [1] 0.3161489
```

```
cor(dta$w_yac_wpa_22,dta$w_yac_wpa)
```

```
## [1] 0.3631849
```

```
plot(uw_yac_wpa_21~uw_yac_wpa, dta, col = 'blue')
points(w_yac_wpa_21~w_yac_wpa, dta, col = 'red')
```




```
cor(dta$uw_yac_wpa_21,dta$uw_yac_wpa)
```

```
## [1] 0.1710394
```

```
cor(dta$w_yac_wpa_21,dta$w_yac_wpa)
```

```
## [1] 0.3103868
```

```
cor(dta[,colnames(dta) %in% c("w_yac_wpa_22", "uw_yac_wpa_22",
                              "uw_yac_wpa_21", "w_yac_wpa_21",
                              "uw_yac_wpa", "w_yac_wpa")])
```

```
##                w_yac_wpa_22 uw_yac_wpa_22 w_yac_wpa_21 uw_yac_wpa_21 w_yac_wpa
## w_yac_wpa_22      1.0000000     0.9411451    0.4109587     0.3883583 0.3631849
## uw_yac_wpa_22     0.9411451     1.0000000    0.3653340     0.3770420 0.3752708
## w_yac_wpa_21      0.4109587     0.3653340    1.0000000     0.9564084 0.3103868
## uw_yac_wpa_21     0.3883583     0.3770420    0.9564084     1.0000000 0.2501677
## w_yac_wpa         0.3631849     0.3752708    0.3103868     0.2501677 1.0000000
## uw_yac_wpa        0.2693315     0.3161489    0.1975255     0.1710394 0.9509893
##                uw_yac_wpa
## w_yac_wpa_22    0.2693315
## uw_yac_wpa_22   0.3161489
## w_yac_wpa_21    0.1975255
## uw_yac_wpa_21   0.1710394
```

```
## w_yac_wpa      0.9509893
## uw_yac_wpa     1.0000000
```

```r
data_2022[order(data_2022$w_yac_wpa, decreasing = T),
          colnames(data_2022) %in% c("posteam", "receiver_player_name", "uw_yac_wpa", "w_yac_wpa")]
```

```
##     posteam receiver_player_name   w_yac_wpa  uw_yac_wpa
## 43       GB               R.Cobb 0.067811363 0.036481713
## 90      NYJ              C.Davis 0.064306536 0.035077597
## 54       KC              T.Kelce 0.062819672 0.040050282
## 49      IND           P.Campbell 0.060718790 0.035619391
## 72      MIA             J.Waddle 0.060413106 0.032626957
## 48      IND             A.Pierce 0.059795669 0.035842773
## 62      LAC             D.Carter 0.053748182 0.028067544
## 79       NE             D.Parker 0.051936801 0.028164340
## 89      NYJ             G.Wilson 0.051831784 0.028155392
## 101     SEA            T.Lockett 0.050696732 0.026801363
## 111     TEN  N.Westbrook-Ikhine 0.049127439 0.026229139
## 114     TEN              T.Burks 0.046678281 0.026128026
## 61      LAC           M.Williams 0.046264165 0.024990852
## 31      DEN              J.Jeudy 0.045765361 0.027425813
## 110     TEN             A.Hooper 0.045545875 0.026575795
## 22      CIN               T.Boyd 0.045530225 0.025455889
## 103      SF           J.Jennings 0.044893633 0.025876737
## 69      MIA               T.Hill 0.044362187 0.023997488
## 65       LV              D.Adams 0.044256515 0.035410665
## 38      DET          T.Hockenson 0.044020092 0.026990587
## 105      SF             B.Aiyuk 0.042115793 0.025367818
## 55       KC  M.Valdes-Scantling 0.042104018 0.026595196
## 112     TEN            C.Okonkwo 0.041961466 0.023142733
## 8       BAL            M.Andrews 0.041342030 0.023801863
## 81       NE             J.Meyers 0.040413324 0.022842736
## 6       ATL              K.Pitts 0.039922705 0.022168120
## 85      NYG            D.Slayton 0.039644766 0.022803108
## 13      BUF              S.Diggs 0.039497976 0.024150311
## 116     WAS           T.McLaurin 0.039475370 0.022029069
## 23      CIN            T.Higgins 0.038840095 0.022038030
## 97      PIT            G.Pickens 0.038834397 0.030013808
## 78       NE              H.Henry 0.038608833 0.022276414
## 56       KC    J.Smith-Schuster 0.038578622 0.020567149
## 24      CLE             A.Cooper 0.038528912 0.027268115
## 109      TB              M.Evans 0.038339075 0.023039330
## 60      LAC              K.Allen 0.038134414 0.021436719
## 15      BUF            I.Hodgins 0.037897158 0.020110995
## 21      CIN              J.Chase 0.037790630 0.021012310
## 117     WAS             J.Dotson 0.037614828 0.021655034
## 68       LV             F.Moreau 0.037460959 0.032921442
## 39       GB             A.Lazard 0.036461008 0.022834673
## 75      MIN          J.Jefferson 0.035857264 0.024480873
## 91      NYJ              E.Moore 0.035530433 0.018333645
## 30      DAL               C.Lamb 0.035076257 0.022290152
## 108      TB              C.Otton 0.034942089 0.021207888
## 18      CHI               C.Kmet 0.034822706 0.023147582
## 16      CAR              D.Moore 0.034802387 0.025708813
```

```
## 40       GB             C.Watson 0.034494906 0.024901541
## 93      PHI              A.Brown 0.034375291 0.025374144
## 83       NO            J.Johnson 0.033845126 0.024029082
## 35      DET          A.St. Brown 0.033555715 0.024584443
## 5       ATL          O.Zaccheaus 0.032403147 0.018417278
## 50      JAX              M.Jones 0.032170362 0.019443427
## 92      PHI              D.Smith 0.032159395 0.030024325
## 94      PHI            D.Goedert 0.031761302 0.021438262
## 70      MIA          T.Sherfield 0.031582168 0.019021437
## 17      CHI             D.Mooney 0.031577980 0.017829096
## 118     WAS             C.Samuel 0.030992285 0.016849864
## 51      JAX               C.Kirk 0.030133990 0.021008404
## 102      SF             D.Samuel 0.029957307 0.015963379
## 12      BUF               D.Knox 0.029722940 0.018269436
## 77       NE            N.Agholor 0.029304413 0.015488126
## 52      JAX              Z.Jones 0.029299993 0.020385955
## 104      SF             G.Kittle 0.029280931 0.017640199
## 113     TEN              R.Woods 0.029078725 0.017044266
## 36      DET           J.Reynolds 0.028960209 0.016000475
## 28      DAL            D.Schultz 0.028952231 0.020525731
## 25      CLE              D.Njoku 0.028940356 0.017053223
## 64      LAC            G.Everett 0.028825551 0.015004992
## 80       NE           T.Thornton 0.028448662 0.018060800
## 33      DEN             C.Sutton 0.028215394 0.016578492
## 74      MIN             K.Osborn 0.028059973 0.016950911
## 34      DET              D.Chark 0.027548686 0.020313722
## 107      TB               R.Gage 0.027199961 0.020623086
## 63      LAC             J.Palmer 0.027177154 0.014123889
## 106      TB             C.Godwin 0.027052284 0.016001683
## 11      BAL           D.Robinson 0.026811486 0.014238558
## 26      CLE      D.Peoples-Jones 0.026189586 0.016936253
## 14      BUF           I.McKenzie 0.026146991 0.016025769
## 10      BAL             I.Likely 0.025713225 0.013740274
## 27      DAL              N.Brown 0.025630251 0.014837158
## 73      MIN            A.Thielen 0.025108926 0.015957976
## 53      JAX             E.Engram 0.025043453 0.018021794
## 96      PIT           C.Claypool 0.024954075 0.017207257
## 1       ARI              R.Moore 0.024915610 0.014334658
## 32      DEN            G.Dulcich 0.024801720 0.018746131
## 29      DAL             M.Gallup 0.024475226 0.016762225
## 46      HOU            N.Collins 0.023741549 0.015977544
## 3       ARI              M.Brown 0.023685988 0.014775465
## 100     SEA               N.Fant 0.022098788 0.011940699
## 19      CHI           C.Claypool 0.021481424 0.019615779
## 7       ATL             D.London 0.021371651 0.015118666
## 84       NO              C.Olave 0.021324967 0.014402680
## 9       BAL           D.Duvernay 0.021250450 0.010951339
## 82       NE             K.Bourne 0.020546516 0.011904751
## 88      NYJ            T.Conklin 0.020346816 0.012003207
## 71      MIA            M.Gesicki 0.020077175 0.011163257
## 66       LV            M.Hollins 0.019737831 0.013776494
## 58       LA          Al.Robinson 0.017543322 0.010429893
## 115     WAS             L.Thomas 0.017480453 0.010227158
## 20      CIN              H.Hurst 0.016974584 0.009336949
```

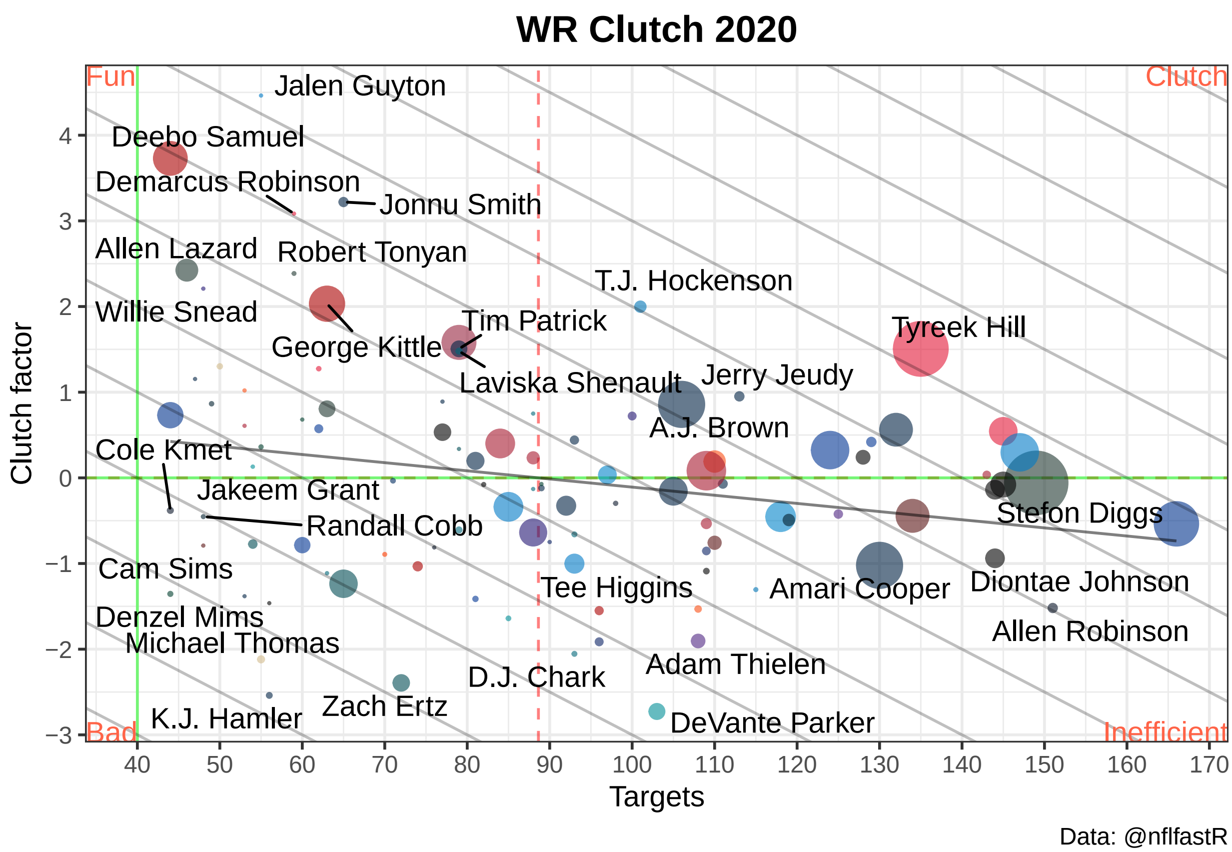
```
## 37      DET            K.Raymond 0.015903617 0.014193874
## 57       LA             T.Higbee 0.014962958 0.009895789
## 2       ARI              Z.Ertz 0.014663983 0.008313572
## 76      MIN          T.Hockenson 0.013801742 0.011367046
## 44      HOU              C.Moore 0.013247173 0.011964464
## 95      PHI            Q.Watkins 0.013091972 0.009794641
## 41       GB              R.Doubs 0.013062021 0.008091228
## 47      HOU              B.Cooks 0.012120600 0.008601829
## 42       GB             R.Tonyan 0.008081039 0.004892449
## 86      NYG              R.James 0.007524867 0.003688882
## 98      PIT            D.Johnson 0.007155365 0.009566317
## 59       LA               C.Kupp 0.005695722 0.003583528
## 4       ARI             G.Dortch 0.005372836 0.002498867
## 99      PIT         P.Freiermuth 0.004906543 0.006929647
## 87      NYG            I.Hodgins 0.004504008 0.002137981
## 67       LV            H.Renfrow 0.004368829 0.006967857
## 45      HOU              J.Akins 0.001936607 0.003172899
```

```
dta <- clutch_2022 %>%
  inner_join(clutch_2021, by = join_by(receiver_player_id), suffix = c("_22", "_21")) %>%
  inner_join(clutch_2020, by = join_by(receiver_player_id), suffix = c("", "_20"))

plot(clutch_22~clutch_21, dta)

cor(dta$clutch_22, dta$clutch_21)

plot(clutch_21~clutch, dta)

cor(dta$clutch_21, dta$clutch)

plot(clutch_22~clutch, dta)

cor(dta$clutch_22, dta$clutch)
```

```
df <- data.frame(x2=rnorm(100),y2=rnorm(100))

annotations <- data.frame(
        xpos = c(-Inf,-Inf,Inf,Inf),
        ypos =  c(-Inf, Inf,-Inf,Inf),
        annotateText = c("Bad","Fun"
                        ,"Inefficient","Clutch"),
        hjustvar = c(0,0,1,1) ,
        vjustvar = c(0,1,0,1)) #<- adjust

plot_clutch <- function(dataframe, yr){
  dataframe %>%
    ggplot(aes(y = clutch, x = targets)) +
    #horizontal line with mean EPA
    geom_hline(yintercept = mean(dataframe$clutch), color = "red", linetype = "dashed", alpha=0.5) +
    #vertical line with mean CPOE
    geom_vline(xintercept =  mean(dataframe$targets), color = "red", linetype = "dashed", alpha=0.5) +
    geom_hline(yintercept = 0, color = "green", linetype = "solid", alpha=0.5) +
    #vertical line with mean CPOE
    geom_vline(xintercept =  40, color = "green", linetype = "solid", alpha=0.5) +
```

```
    #add points for the QBs with the right colors
    #cex controls point size and alpha the transparency (alpha = 1 is normal)
    geom_point(color = dataframe$team_color, cex=dataframe$value / median(dataframe$value), alpha = .6)
    #add names using ggrepel, which tries to make them not overlap
    geom_text_repel(aes(label=player), max.overlaps = 7) +
    #add a smooth line fitting wpa + epa
    stat_smooth(geom='line', alpha=0.5, se=FALSE, method='lm')+
    geom_abline(slope = -.05, intercept = (15:-5), alpha = .25)+
    #titles and caption
    labs(x = "Targets",
         y = "Clutch factor",
         title = paste("WR Clutch", yr),
         caption = "Data: @nflfastR") +
    #uses the black and white ggplot theme
    theme_bw() +
    #center title with hjust = 0.5
    theme(
      plot.title = element_text(size = 14, hjust = 0.5, face = "bold")
    ) +
    #make ticks look nice
    #if this doesn't work, `install.packages('scales')`
    scale_y_continuous(breaks = scales::pretty_breaks(n = 10)) +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
    geom_text(data=annotations,aes(x=xpos,y=ypos,hjust=hjustvar,vjust=vjustvar,label=annotateText), col
}

plot_clutch(data_2020, "2020")
```

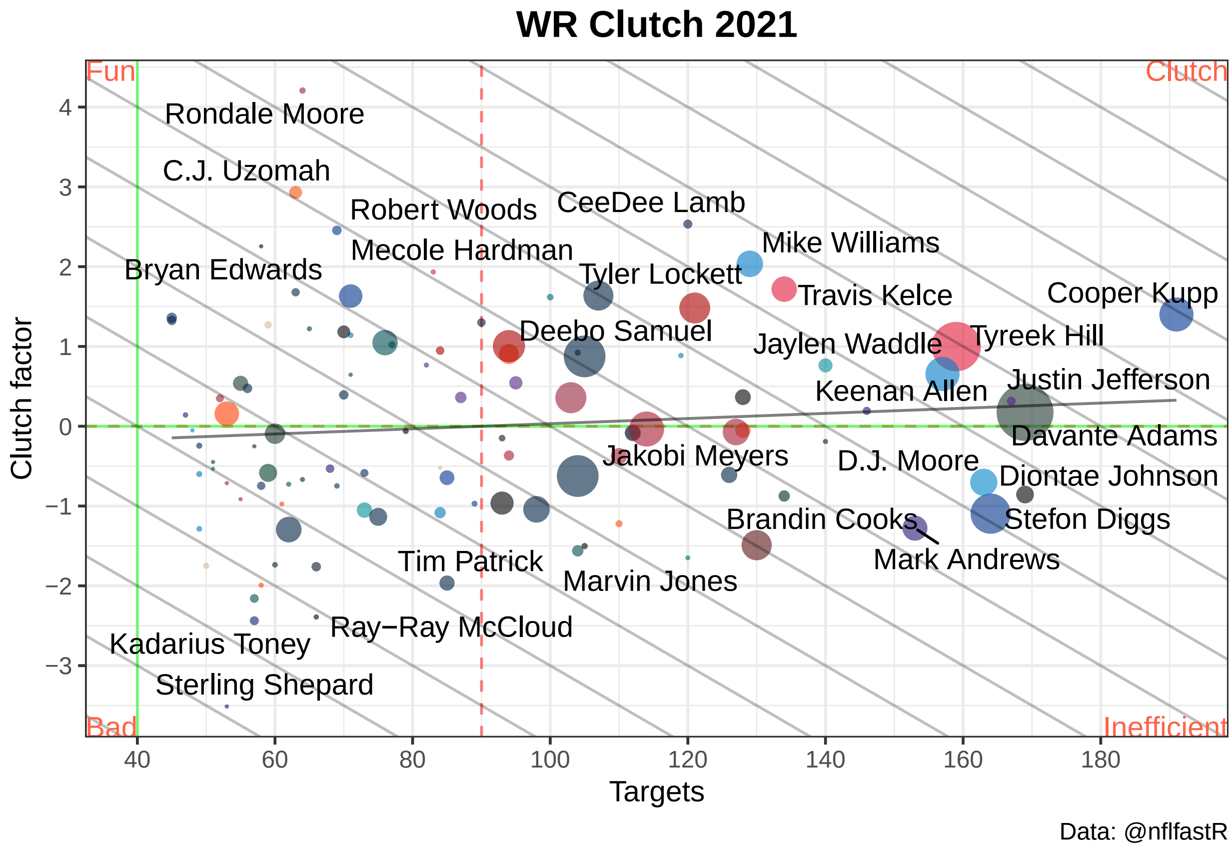
```
## 'geom_smooth()' using formula = 'y ~ x'

## Warning: ggrepel: 71 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
plot_clutch(data_2021, "2021")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

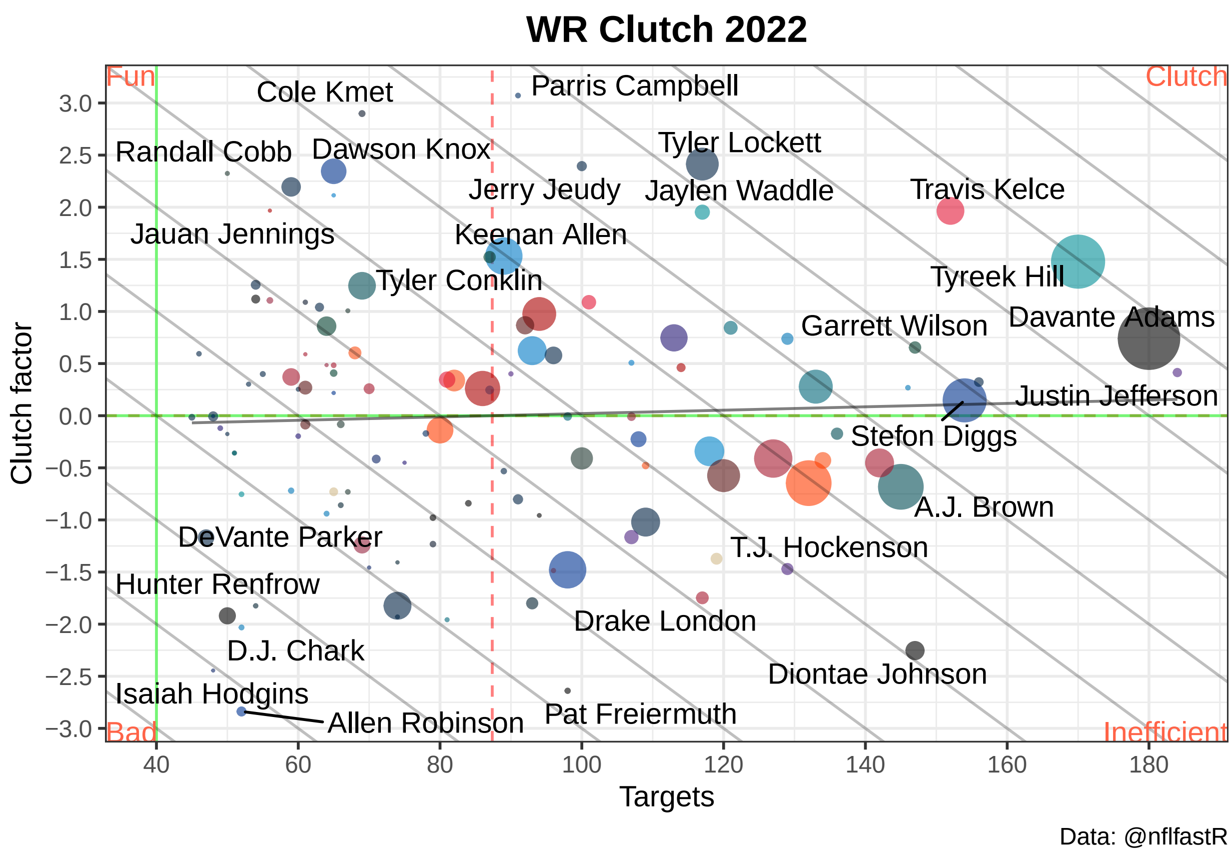
```
## Warning: ggrepel: 79 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```




```
plot_clutch(data_2022, "2022")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: ggrepel: 92 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```


WR Clutch 2022
Fun
Clutch
Bad
Inefficient
Clutch factor
Targets
Cole Kmet
Parris Campbell
Randall Cobb
Dawson Knox
Tyler Lockett
Jerry Jeudy
Jaylen Waddle
Travis Kelce
Jauan Jennings
Keenan Allen
Tyler Conklin
Tyreek Hill
Garrett Wilson
Davante Adams
Justin Jefferson
Stefon Diggs
A.J. Brown
DeVante Parker
T.J. Hockenson
Hunter Renfrow
Drake London
D.J. Chark
Diontae Johnson
Isaiah Hodgins
Pat Freiermuth
Allen Robinson
Data: @nflfastR