Malcolm Newell Reds Code

Malcolm Newell

2024-10-14

Libraries

```r
suppressWarnings(suppressMessages({
  library(dplyr)
  library(tidyr)
  library(readr)
  library(mgcv)
  library(ggplot2)
  library(ggeasy)
}))
```

Functions

```r
get_gam_model <- function(train_data){

  model <- gam(usage_percent ~
                 s(DELTA_RUN_EXP, BALLS) +
                 s(DELTA_RUN_EXP, STRIKES) +
                 as.factor(BAT_SIDE) + as.factor(THROW_SIDE),
                      data = train_data, method = "REML")

}

# Get Player Search Data
get_search_data <- function(player_name){

  df <- search_joined_data %>%
    filter(PLAYER_NAME == "player_name") %>%
    arrange(desc(xwOBA)) %>%
    select("Side" = THROW_SIDE, "Pitch Group" = pitch_group,
           "Usage %" = usage_percent, `Max EV`, `Avg EV`, LA, xBA, xwOBA, wOBA)

}

get_prediction_plot <- function(df_predictions){

  data_long <- df_predictions %>%
  dplyr::rename(
    Fastball = PITCH_TYPE_FB,
    "Breaking Ball" = PITCH_TYPE_BB,
    Offspeed = PITCH_TYPE_OS
  ) %>%
  tidyr::pivot_longer(cols = c(Fastball, `Breaking Ball`, Offspeed),
                      names_to = "Pitch Group",
                      values_to = "Usage_Percentage") %>%
  dplyr::mutate(
    `Pitch Group` = factor(`Pitch Group`,
                           levels = c("Fastball", "Breaking Ball", "Offspeed"))
  )


# Create bar plot
ggplot(data_long, aes(x = `Pitch Group`, y = Usage_Percentage, fill = `Pitch Group`)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Usage_Percentage, "%")),
            position = position_stack(vjust = 0.5),
            color = "white") +
  labs(title = paste0("Pitch Usage Percentage for ", unique(data_long$PLAYER_NAME), " in 2024"),
       x = "",
       y = "Usage %") +
  theme_minimal() +
  ggeasy::easy_center_title() +
  ggeasy::easy_remove_legend() +
  scale_fill_manual(values = c("Fastball" = "#C6011F",
                                "Breaking Ball" = "skyblue",
                                "Offspeed" = "darkgreen")) +
  theme(panel.grid.major = element_blank()) +
  ylim(0, 50)

}
```

Reading Data

```
data <- read_csv(file = "Data/data.csv")
predictions <- read_csv(file = "Data/predictions.csv")
sample <- read_csv(file = "Data/sample_submission.csv")
```

Changing pitch types to pitch groups

```


pitch_data <- data %>%
  dplyr::mutate(
    pitch_id = row_number(),
    pitch_group = ifelse(
    PITCH_TYPE %in% c("FF", "SI"), "FB",
    ifelse(
      PITCH_TYPE %in% c("CH", "FO", "FS", "SC"), "OS",
      ifelse(
        PITCH_TYPE %in% c("CS", "CU", "FC", "KC", "SL", "ST", "SV"), "BB",
        "Other"
      )
    )
  )) %>%
  dplyr::filter(pitch_group != "Other") %>%
  select(pitch_id, BATTER_ID, PLAYER_NAME, pitch_group,
         BAT_SIDE, THROW_SIDE,
         BALLS, STRIKES, DELTA_RUN_EXP)
```

I did not include the Ephus, Knuckleball, Other, or Pitch Out in the groups

Finding Usage % for each pitch group and player every year

```
usages <- pitch_data %>%
  group_by(BATTER_ID, PLAYER_NAME, THROW_SIDE) %>%
  dplyr::mutate(total_pitches = n()) %>%
  ungroup() %>%
  group_by(BATTER_ID, PLAYER_NAME, THROW_SIDE, pitch_group) %>%
  dplyr::summarize(group_pitches = n(),
                   total_pitches = first(total_pitches)) %>%
  dplyr::mutate(usage_percent = (group_pitches / total_pitches) * 100) %>%
  ungroup()
```

Combine pitch_data with usages

```
joined_data <- usages %>%
  dplyr::left_join(pitch_data, by = c("BATTER_ID", "PLAYER_NAME",
                                      "THROW_SIDE", "pitch_group"))
```

Set the seed and prep the model by filtering dataframes for each pitch group

```
addTaskCallback(function(...){set.seed(123);TRUE})
```

```
## 1
## 1
```

```
fb_data <- joined_data %>% dplyr::filter(pitch_group == "FB")
bb_data <- joined_data %>% dplyr::filter(pitch_group == "BB")
os_data <- joined_data %>% dplyr::filter(pitch_group == "OS")
```

Fastball Model

```r
dt_fb <- sample(nrow(fb_data), nrow(fb_data) * .7)
train_fb <- fb_data[dt_fb,]
test_fb <- fb_data[-dt_fb,]

fb_usage_model <- get_gam_model(train_fb)

summary(fb_usage_model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## usage_percent ~ s(DELTA_RUN_EXP, BALLS) + s(DELTA_RUN_EXP, STRIKES) +
##     as.factor(BAT_SIDE) + as.factor(THROW_SIDE)
##
## Parametric coefficients:
##                           Estimate Std. Error  t value            Pr(>|t|)
## (Intercept)            49.9326555  0.0143073 3490.009 <0.0000000000000002 ***
## as.factor(BAT_SIDE)R   -0.0005735  0.0124560   -0.046               0.963
## as.factor(THROW_SIDE)R -1.6625771  0.0137090 -121.276 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Approximate significance of smooth terms:
##                              edf Ref.df     F             p-value
## s(DELTA_RUN_EXP,BALLS)     2.526   2.93 1.032               0.396
## s(DELTA_RUN_EXP,STRIKES) 18.112  21.91 8.465 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## R-sq.(adj) =  0.0338   Deviance explained = 3.38%
## -REML = 1.2214e+06  Scale est. = 16.154    n = 434632
```

Breaking Ball Model

```r
dt_bb <- sample(nrow(bb_data), nrow(bb_data) * .7)
train_bb <- bb_data[dt_bb,]
test_bb <- bb_data[-dt_bb,]

bb_usage_model <- get_gam_model(train_bb)

summary(bb_usage_model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## usage_percent ~ s(DELTA_RUN_EXP, BALLS) + s(DELTA_RUN_EXP, STRIKES) +
##     as.factor(BAT_SIDE) + as.factor(THROW_SIDE)
```

```
##
## Parametric coefficients:
##                        Estimate Std. Error t value            Pr(>|t|)
## (Intercept)            34.18453    0.02115  1616.1 <0.0000000000000002 ***
## as.factor(BAT_SIDE)R    4.62156    0.01833   252.1 <0.0000000000000002 ***
## as.factor(THROW_SIDE)R  2.95972    0.02047   144.6 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Approximate significance of smooth terms:
##                              edf Ref.df      F             p-value
## s(DELTA_RUN_EXP,BALLS)     6.715  8.639 62.212 <0.0000000000000002 ***
## s(DELTA_RUN_EXP,STRIKES)  13.615 17.427  9.144 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## R-sq.(adj) =  0.195   Deviance explained = 19.5%
## -REML = 1.0581e+06  Scale est. = 27.473    n = 344026
```

Off-speed Model

```r
dt_os <- sample(nrow(os_data), nrow(os_data) * .7)
train_os <- os_data[dt_os,]
test_os <- os_data[-dt_os,]

os_usage_model <- get_gam_model(train_os)

summary(os_usage_model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## usage_percent ~ s(DELTA_RUN_EXP, BALLS) + s(DELTA_RUN_EXP, STRIKES) +
##     as.factor(BAT_SIDE) + as.factor(THROW_SIDE)
##
## Parametric coefficients:
##                        Estimate Std. Error t value            Pr(>|t|)
## (Intercept)            24.87990    0.03761   661.4 <0.0000000000000002 ***
## as.factor(BAT_SIDE)R   -7.82057    0.03143  -248.8 <0.0000000000000002 ***
## as.factor(THROW_SIDE)R -6.59237    0.03452  -191.0 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Approximate significance of smooth terms:
##                             edf Ref.df       F             p-value
## s(DELTA_RUN_EXP,BALLS)    7.276  9.705   9.229 <0.0000000000000002 ***
## s(DELTA_RUN_EXP,STRIKES) 1.146  1.277 590.199 <0.0000000000000002 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
```

```
## R-sq.(adj) =  0.368   Deviance explained = 36.8%
## -REML = 3.5111e+05  Scale est. = 20.919    n = 119450
```

Predictions

```
# Predict fastballs
predictions_fb <- test_fb %>%
  dplyr::mutate(GAME_YEAR = 2024)

predictions_fb$fastball_usage <- predict(fb_usage_model, newdata =
                                           predictions_fb, type = "response")

predictions_fb <- predictions_fb %>%
  dplyr::select(pitch_id, BATTER_ID, PLAYER_NAME, GAME_YEAR,
                THROW_SIDE, fastball_usage)

# Predict breaking balls
predictions_bb <- test_bb %>%
  mutate(GAME_YEAR = 2024)

predictions_bb$breaking_usage <- predict(bb_usage_model, newdata =
                                           predictions_bb, type = "response")

predictions_bb <- predictions_bb %>%
  dplyr::select(pitch_id, BATTER_ID, PLAYER_NAME, GAME_YEAR,
                THROW_SIDE, breaking_usage)

# Predict off-speed
predictions_os <- test_os %>%
  mutate(GAME_YEAR = 2024)

predictions_os$offspeed_usage <- predict(os_usage_model, newdata =
                                           predictions_os, type = "response")
predictions_os <- predictions_os %>%
  dplyr::select(pitch_id, BATTER_ID, PLAYER_NAME, GAME_YEAR,
                THROW_SIDE, offspeed_usage)
```

Combining Predictions into one dataframe

```
joined_predictions <- predictions_fb %>%
  dplyr::full_join(
    predictions_bb, by = c("pitch_id", "BATTER_ID", "PLAYER_NAME",
                           "GAME_YEAR", "THROW_SIDE")
  ) %>%
  dplyr::full_join(
    predictions_os, by = c("pitch_id", "BATTER_ID", "PLAYER_NAME",
                           "GAME_YEAR", "THROW_SIDE")
  )
```

```
predictions <- joined_predictions %>%
  group_by(BATTER_ID, PLAYER_NAME, GAME_YEAR) %>%
  dplyr::summarize(
    PITCH_TYPE_FB = round(mean(fastball_usage, na.rm = TRUE),1),
    PITCH_TYPE_BB = round(mean(breaking_usage, na.rm = TRUE),1),
    PITCH_TYPE_OS = round(mean(offspeed_usage, na.rm = TRUE),1)
  ) %>%
  ungroup() %>%
  dplyr::select(BATTER_ID, PLAYER_NAME, GAME_YEAR, PITCH_TYPE_FB,
                PITCH_TYPE_BB, PITCH_TYPE_OS)
```

Joining all predictions then aggregating the results

```
## ‘summarise()‘ has grouped output by ’BATTER_ID’, ’PLAYER_NAME’. You can
## override using the ‘.groups‘ argument.
```

Graphics

```
search_pitch_data <- data %>%
  dplyr::mutate(
    pitch_id = row_number(),
    pitch_group = ifelse(
    PITCH_TYPE %in% c("FF", "SI"), "FB",
    ifelse(
      PITCH_TYPE %in% c("CH", "FO", "FS", "SC"), "OS",
      ifelse(
        PITCH_TYPE %in% c("CS", "CU", "FC", "KC", "SL", "ST", "SV"), "BB",
        "Other"
      )
    )
  )) %>%
  dplyr::filter(pitch_group != "Other")

aggregate_data <- search_pitch_data %>%
  group_by(BATTER_ID, PLAYER_NAME,  THROW_SIDE, pitch_group) %>%
  dplyr::summarize(
    "Max EV" = round(max(LAUNCH_SPEED, na.rm = T),1),
    "Avg EV" = round(mean(LAUNCH_SPEED, na.rm = T),1),
    LA = round(mean(LAUNCH_ANGLE, na.rm = T)),
    xBA = round(mean(ESTIMATED_BA_USING_SPEEDANGLE, na.rm = T),3),
    xwOBA = round(mean(ESTIMATED_WOBA_USING_SPEEDANGLE, na.rm = T),3),
    wOBA = round(mean(WOBA_VALUE, na.rm = T),3)
  ) %>%
  ungroup()
```

Using this data to find three interesting players

```
## Warning: There were 12 warnings in ‘dplyr::summarize()‘.
## The first warning was:
## i In argument: ‘Max EV = round(max(LAUNCH_SPEED, na.rm = T), 1)‘.
## i In group 1172: ‘BATTER_ID = 666163‘, ‘PLAYER_NAME = "Rortvedt, Ben"‘,
##   ‘THROW_SIDE = "L"‘, ‘pitch_group = "OS"‘.
## Caused by warning in ‘max()‘:
## ! no non-missing arguments to max; returning -Inf
## i Run ‘dplyr::last_dplyr_warnings()‘ to see the 11 remaining warnings.
```

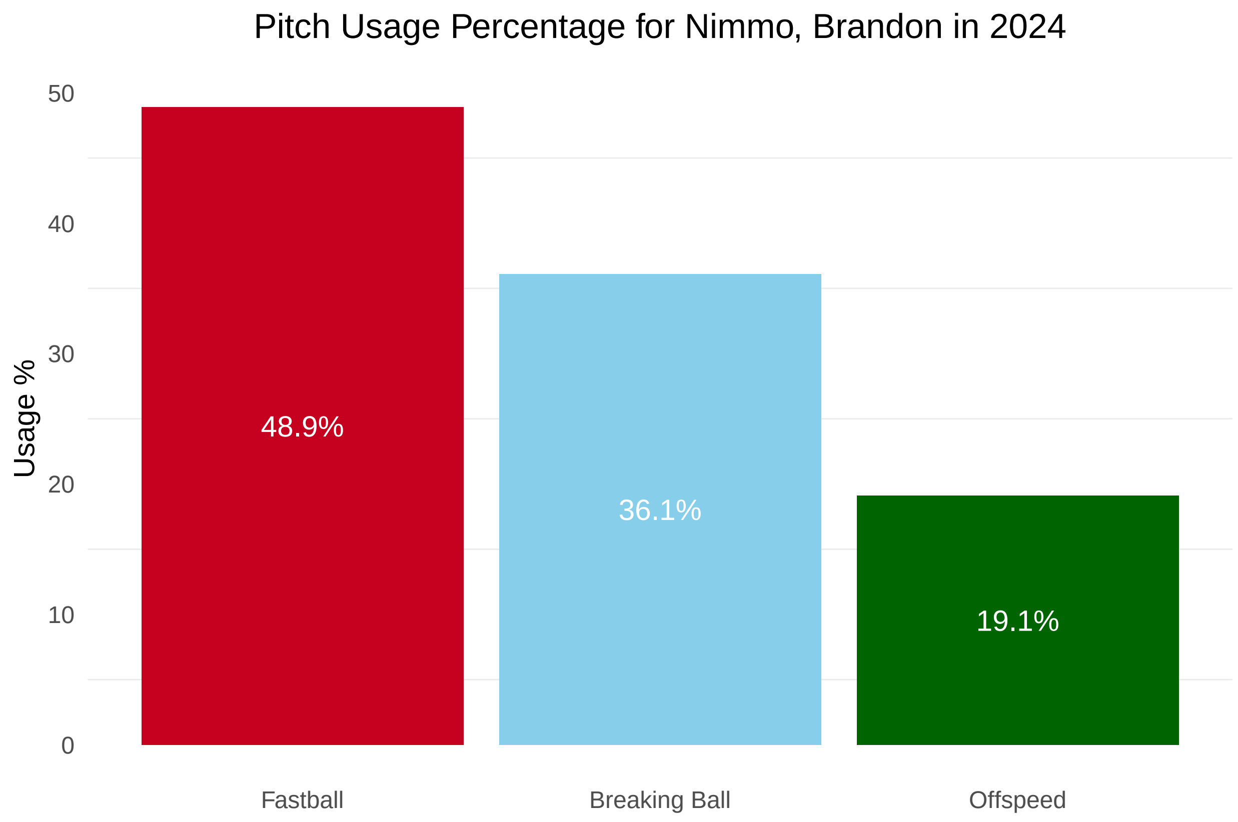
```
search_usages <- search_pitch_data %>%
  group_by(BATTER_ID, PLAYER_NAME, THROW_SIDE) %>%
  dplyr::mutate(total_pitches = n()) %>%
  ungroup() %>%
  group_by(BATTER_ID, PLAYER_NAME, THROW_SIDE, pitch_group) %>%
  dplyr::summarize(group_pitches = n(),
                   total_pitches = first(total_pitches)) %>%
  dplyr::mutate(usage_percent = round((group_pitches /
                                       total_pitches) * 100, 1)) %>%
  ungroup()

search_joined_data <- search_usages %>%
  dplyr::left_join(aggregate_data, by = c("BATTER_ID", "PLAYER_NAME",
                                          "THROW_SIDE", "pitch_group"))
```

Individual Player Metrics

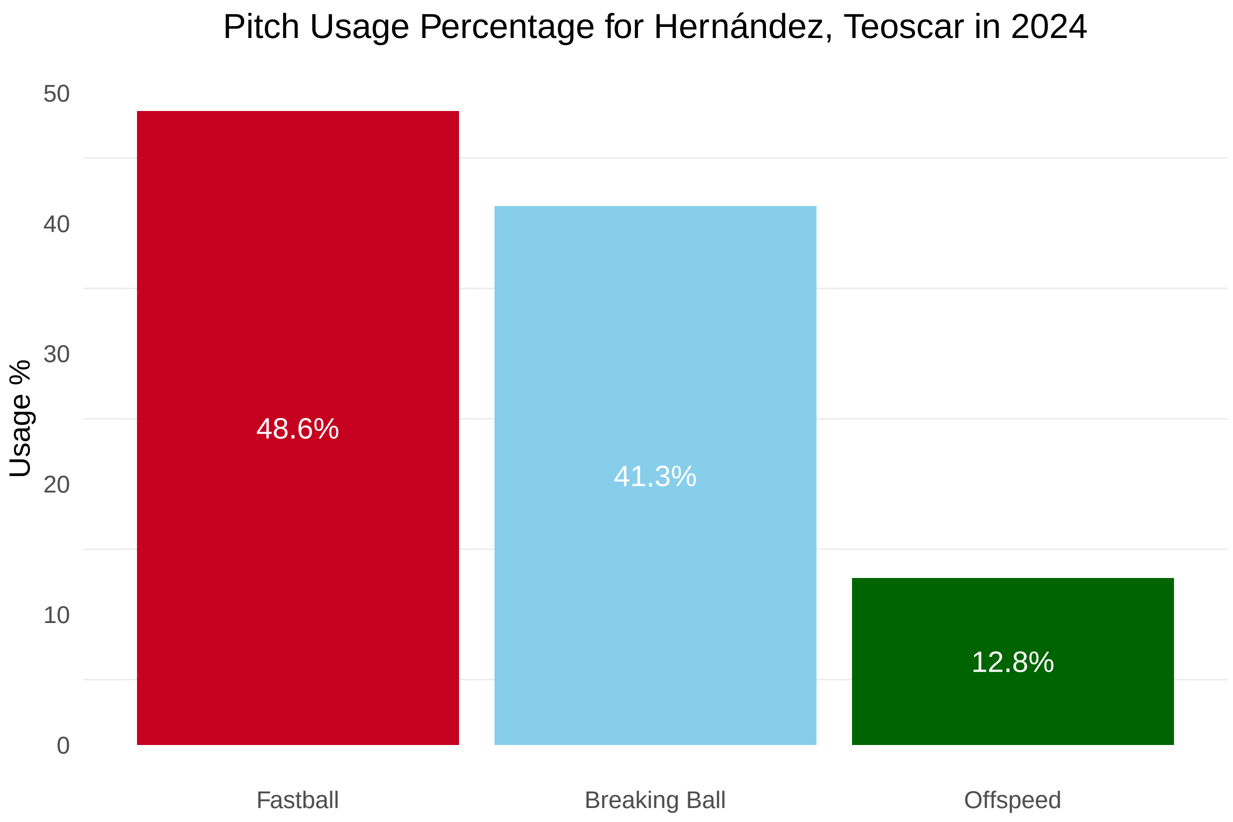
```
nimmo_df <- get_search_data("Nimmo, Brandon")

teoscar_df <- get_search_data("Hernández, Teoscar")

steer_df <- get_search_data("Steer, Spencer")
```

View Prediction Plots

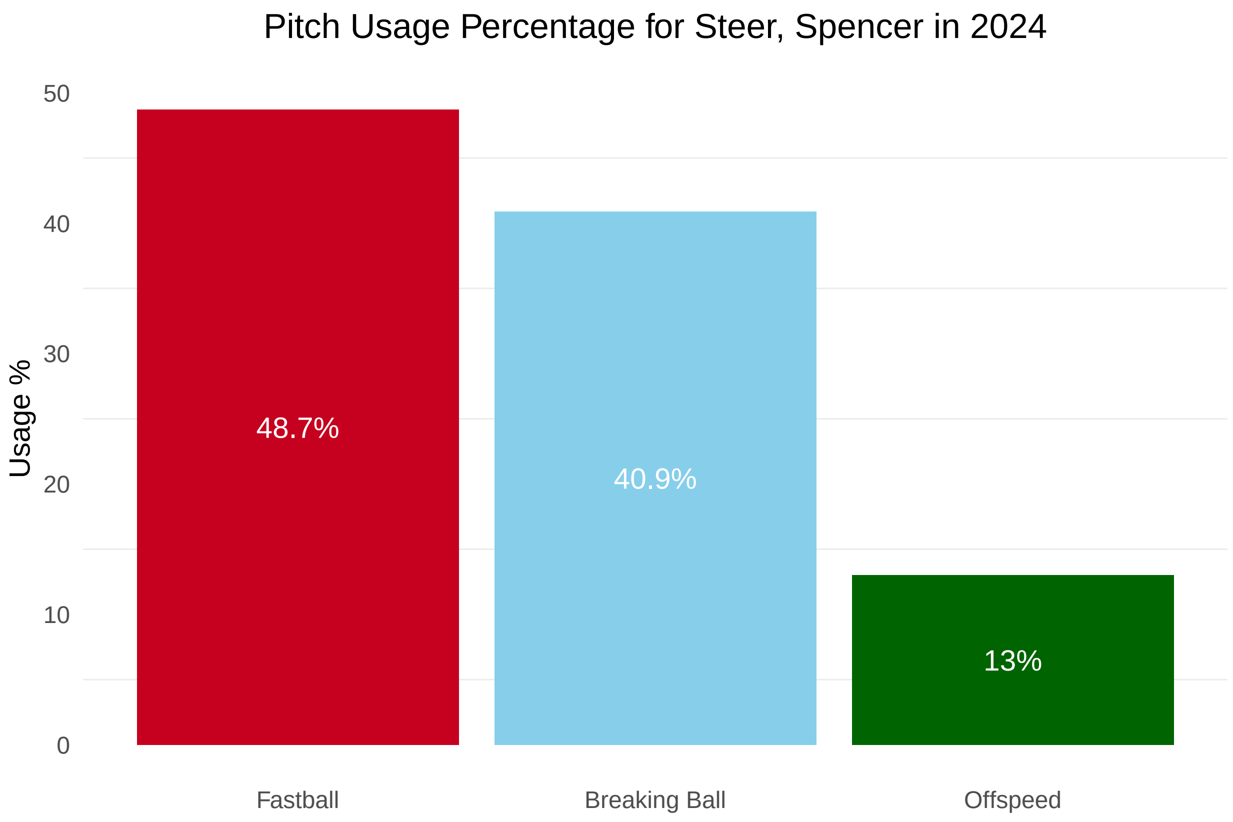
```
nimmo_predictions <- predictions %>%
  filter(PLAYER_NAME == "Nimmo, Brandon")

teoscar_predictions <- predictions %>%
  filter(PLAYER_NAME == "Hernández, Teoscar")

steer_predictions <- predictions %>%
  filter(PLAYER_NAME == "Steer, Spencer")

get_prediction_plot(nimmo_predictions)
```



```
get_prediction_plot(teoscar_predictions)
```



```
get_prediction_plot(steer_predictions)
```


Pitch Usage Percentage for Steer, Spencer in 2024
Usage %
50
40
30
20
10
0
48.7%
40.9%
13%
Fastball
Breaking Ball
Offspeed