Sports Analytics Final Project

Jack Harenchar

2025-02-20

Load Necessary Libraries

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(ggplot2)
library(sportyR)
library(dplyr)
```

Loading the Data

```r
load("Tarpons_2024.rda")
load("summer_statcast.rda") #individual pitch data for June through August 2024
load("Rays_2024.rda")
load("minor_league.rda") #Pitch data for all Tampa Tarpons 2024 home games
```

Create Necessary Variables and Data Subsets

```r
# Filtering MLB summer data by balls hit into play by the Rays at Home
Rays_Home_Hits_All <- full_data %>%
  filter(home_team == "TB") %>%
  filter(description == "hit_into_play") %>%
  filter(inning_topbot == "Bot")
```

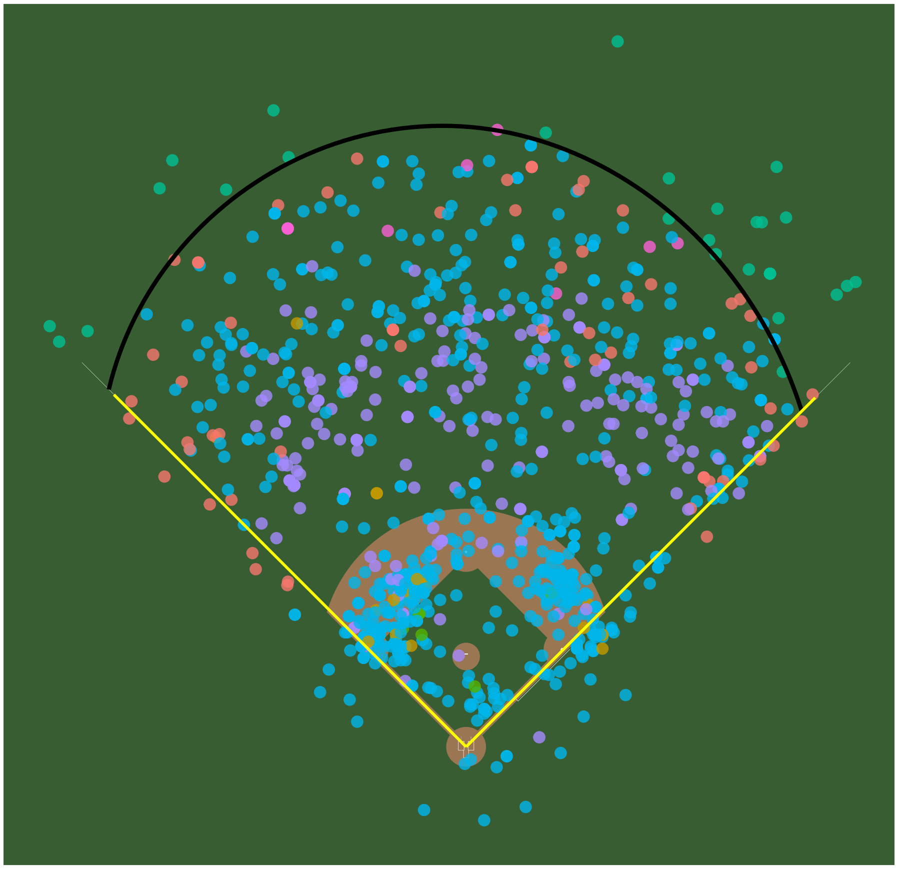
```r
minor_res$game_date <- as.Date(minor_res$game_date, format = "%Y-%m-%d")
Steinbrenner_BiP_summer <- minor_res %>%
  filter(batting_team == "Tampa Tarpons") %>%
  filter(details.isInPlay == TRUE) %>%
  filter(between(game_date, as.Date("2024-06-01"), as.Date("2024-08-31")))

Steinbrenner_BiP_summer <- Steinbrenner_BiP_summer %>%
  mutate(hit_result = ifelse(details.isOut == TRUE, "Out", result.event)) %>%
  mutate(hit_result = recode(hit_result,
                             "Field Error" = "Error"))

Rays_Home_Hits_All <- Rays_Home_Hits_All %>%
  mutate(hit_result = recode(events,
                             "single" = "Single",
                             "field_out" = "Out",
                             "double" = "Double",
                             "sac_fly" = "Out",
                             "home_run" = "Home Run",
                             "grounded_into_double_play" = "Out",
                             "field_error" = "Error",
                             "sac_bunt" = "Out",
                             "force_out" = "Out",
                             "triple" = "Triple",
                             "fielders_choice_out" = "Out",
                             "fielders_choice" = "Fielders Choice",
                             "double_play" = "Out"))
```

Tarpons Hit Map at Steinbrenner Field

```r
library(ggplot2)

# Define field distances (in feet)
wall_distances <- data.frame(
  angle = c(-45, -22.5, 0, 22.5, 45),  # Corresponding angles from home plate
  distance = c(310, 370, 405, 390, 330)  # Given wall distances
)

# Convert angles to radians
wall_distances$angle_rad <- wall_distances$angle * pi / 180

# Compute x and y coordinates
wall_distances$x <- wall_distances$distance * cos(wall_distances$angle_rad)
wall_distances$y <- wall_distances$distance * sin(wall_distances$angle_rad)

# Interpolate smooth curve between points
smooth_wall <- data.frame(
  angle = seq(-45, 45, length.out = 100)
)
smooth_wall$angle_rad <- smooth_wall$angle * pi / 180
smooth_wall$distance <- spline(wall_distances$angle, wall_distances$distance,
                               xout = smooth_wall$angle)$y
```

```r
smooth_wall$x <- smooth_wall$distance * cos(smooth_wall$angle_rad)
smooth_wall$y <- smooth_wall$distance * sin(smooth_wall$angle_rad)

# Rotate 90 degrees to the right (swap x and y, negate x)
smooth_wall_rotated <- data.frame(
  x = -smooth_wall$y,  # Negating y-coordinates to flip horizontally
  y = smooth_wall$x
)


geom_baseball(league = "mlb", display_range = "full", rotation = 0) +
    geom_point(alpha = 0.8, data = Steinbrenner_BiP_summer,
               aes(x= 2.5 * (hitData.coordinates.coordX - 125.42),
                   y=  2.5 *(198.27 - hitData.coordinates.coordY),
                   color = hit_result)) +
geom_path(data = smooth_wall_rotated, aes(x, y), color = "black",
          linewidth = .75) +
  coord_fixed() +
  geom_segment(aes(x = 0, xend = 228, y= 0, yend = 228), colour = "yellow") +
  geom_segment(aes(x = 0, xend = -230, y= 0, yend = 230), colour = "yellow") +
  theme(legend.position = "none") +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_blank(),
        panel.background = element_blank())
```

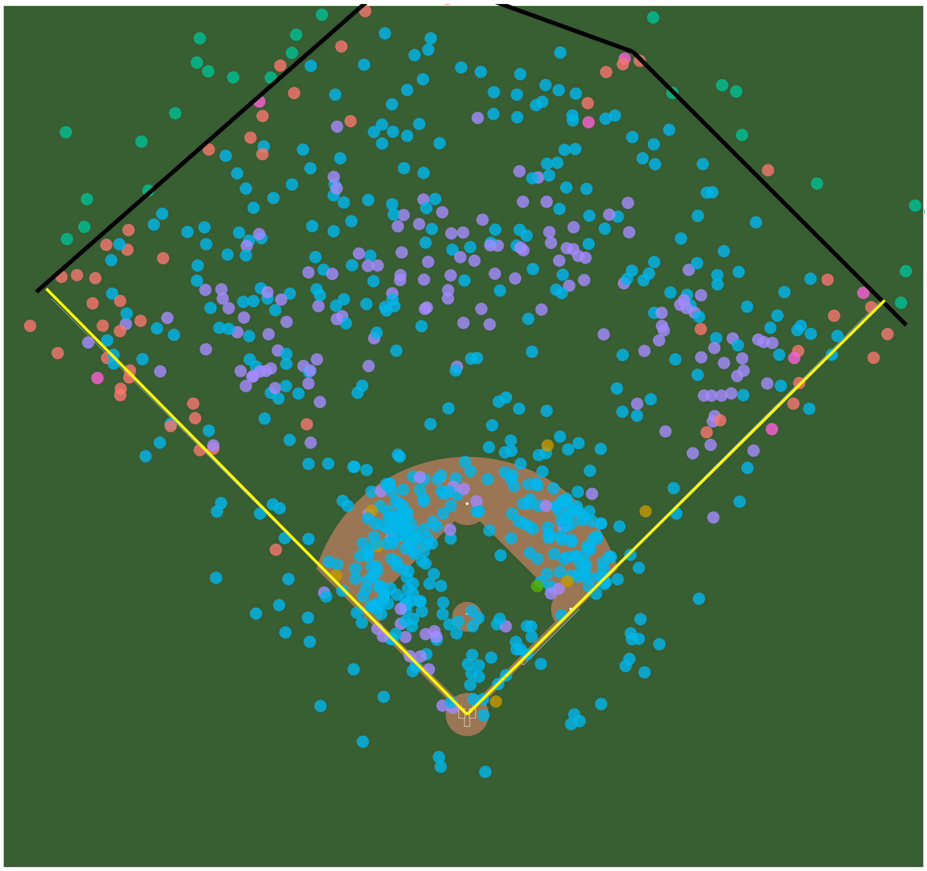
```
## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.
```



```
ggsave("Tarpons_hits.png", width = 8, height = 6, dpi = 500)
```

Rays Hits at Tropicana Field

```
geom_baseball(league = "MLB", display_range = "full", rotation = 0) +
    geom_point(alpha = 0.8, data = Rays_Home_Hits_All,
               aes(x= 2.5 * (hc_x - 125.42), y=  2.5 *(198.27 - hc_y),
                   color = hit_result)) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_blank(),
        panel.background = element_blank()) +
  theme(legend.position = "none") +
  #labs(color = "Hit Result") +
  geom_segment(aes(x = 265, xend = 100, y= 235, yend =  400), colour = "black",
               linewidth = .75) +
  geom_segment(aes(x = -260, xend = -38, y= 255, yend = 450), colour = "black",
               linewidth = .75) +
  geom_segment(aes(x = -38, xend = 100, y= 450, yend = 400), colour = "black",
               linewidth = .75) +
  geom_segment(aes(x = 0, xend = 252, y= 0, yend = 250), colour = "yellow") +
  geom_segment(aes(x = 0, xend = -254, y= 0, yend = 257), colour = "yellow")
```



```
ggsave("Rays_hits.png", width = 8, height = 6, dpi = 500)
```

Predicting Rays Hit Result based off Tarpons Hits (Trial)

```
library(xgboost) # Load XGBoost
```

```
##
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(fastDummies)
model_data <- Steinbrenner_BiP_summer[, c( "hitData.coordinates.coordX",
                                           "hitData.coordinates.coordY",
                                           "hitData.trajectory")]
model_response <- Steinbrenner_BiP_summer$hit_result
summary(as.factor(model_response))
```

```
##          Double            Error Fielders Choice        Home Run                Out
##              61               19               6              28              604
##          Single          Triple
##             198               9
```

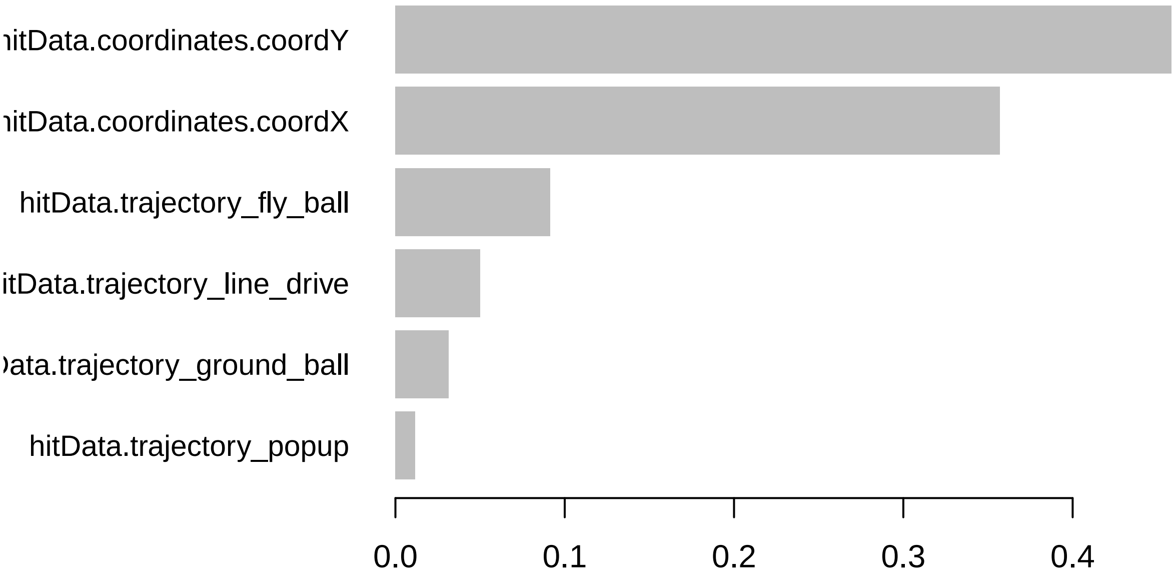
```r
new_resp <- as.numeric(as.factor(model_response)) - 1
table(new_resp, model_response)
```

```
##         model_response
## new_resp Double Error Fielders Choice Home Run Out Single Triple
##        0     61     0               0        0   0      0      0
##        1      0    19               0        0   0      0      0
##        2      0     0               6        0   0      0      0
##        3      0     0               0       28   0      0      0
##        4      0     0               0        0 604      0      0
##        5      0     0               0        0   0    198      0
##        6      0     0               0        0   0      0      9
```

```r
t1 <- cbind.data.frame(new_resp, model_response)
t1 <- unique(t1)
mdat <- dummy_cols(model_data, remove_selected_columns = TRUE)
# Create training matrix
dtrain <- xgb.DMatrix(data = as.matrix(mdat), label = new_resp)
set.seed(111111)
bst_1 <- xgboost(data = dtrain, # Set training data
               num_class = length(unique(new_resp)),
               nrounds = 100, # Set number of rounds

               verbose = 1, # 1 - Prints out fit
                print_every_n = 20, # Prints out result every 20th iteration

               objective = "multi:softmax", # Set objective
               eval_metric = "merror") # Set evaluation metric to use
```

```
## [1]  train-merror:0.143784
## [21] train-merror:0.058378
## [41] train-merror:0.015135
## [61] train-merror:0.005405
## [81] train-merror:0.003243
## [100]    train-merror:0.001081
```

```r
# Extract importance
imp_mat <- xgb.importance(model = bst_1)
# Plot importance (top 10 variables)
xgb.plot.importance(imp_mat, top_n = 10)
```



```
ggsave("var_imp_1.png", width = 8, height = 6, dpi = 500)
```

```
model_data2 <- Rays_Home_Hits_All[, c( "hc_x", "hc_y", "bb_type")]
names(model_data2) <- names(model_data)
model_response2 <- Rays_Home_Hits_All$hit_result
mdat2 <- dummy_cols(model_data2, remove_selected_columns = TRUE)
# Create test matrix
dtest <- xgb.DMatrix(data = as.matrix(mdat2))
xgb_preds <- predict(bst_1, dtest)
xgb_preds_conv <- rep(NA, length(xgb_preds))
for(i in 1:nrow(t1)){
  xgb_preds_conv[which(xgb_preds == t1$new_resp[i])] <- t1$model_response[i]
}
table(model_response2, xgb_preds_conv)
```

```
##                   xgb_preds_conv
## model_response2    Double Error Home Run Out Single Triple
##   Double               33     0        5   9      4      2
##   Error                 0     0        0   6      1      0
##   Fielders Choice       0     0        0   1      0      0
##   Home Run              3     0       23   2      0      0
##   Out                  18     4        5 485     33      3
##   Single               12     0        0  77     71      0
##   Triple                5     0        1   1      0      0
```

Removing Tarpon NA's

```
Steinbrenner_BiP_summer$hitData.launchSpeed[is.na(Steinbrenner_BiP_summer$hitData.launchSpeed)] <-
  mean(Steinbrenner_BiP_summer$hitData.launchSpeed, na.rm = TRUE)

Steinbrenner_BiP_summer$hitData.totalDistance[is.na(Steinbrenner_BiP_summer$hitData.totalDistance)] <-
  mean(Steinbrenner_BiP_summer$hitData.totalDistance, na.rm = TRUE)

Steinbrenner_BiP_summer$hitData.launchAngle[is.na(Steinbrenner_BiP_summer$hitData.launchAngle)] <-
  mean(Steinbrenner_BiP_summer$hitData.launchAngle, na.rm = TRUE)
```

Training Model

```
model_data3 <- Steinbrenner_BiP_summer[, c( "hitData.coordinates.coordX",
                                            "hitData.coordinates.coordY",
                                            "hitData.trajectory",
                                            "hitData.totalDistance",
                                            "hitData.launchAngle",
                                            "hitData.launchSpeed")]
model_response3 <- Steinbrenner_BiP_summer$hit_result

summary(as.factor(model_response3))
```

```
##          Double           Error Fielders Choice        Home Run             Out
##              61              19               6              28             604
##          Single          Triple
##             198               9
```

```
new_resp <- as.numeric(as.factor(model_response3)) - 1

mdat3 <- dummy_cols(model_data3, remove_selected_columns = TRUE)

t1 <- cbind.data.frame(new_resp, model_response3)
t1 <- unique(t1)

# Create training matrix
dtrain <- xgb.DMatrix(data = as.matrix(mdat3), label = new_resp)

set.seed(111111)
bst_1 <- xgboost(data = dtrain, # Set training data
               num_class = length(unique(new_resp)),
               nrounds = 100, # Set number of rounds

               verbose = 1, # 1 - Prints out fit
                print_every_n = 20, # Prints out result every 20th iteration

               objective = "multi:softmax", # Set objective
               eval_metric = "merror") # Set evaluation metric to use
```

```
## [1]  train-merror:0.118919
## [21] train-merror:0.011892
## [41] train-merror:0.000000
## [61] train-merror:0.000000
## [81] train-merror:0.000000
## [100]    train-merror:0.000000
```

Model Predictions

```
model_data4 <- Rays_Home_Hits_All[, c( "hc_x", "hc_y", "bb_type",
                                       "hit_distance_sc", "launch_angle",
                                       "launch_speed")]
names(model_data4) <- names(model_data3)
model_response4 <- Rays_Home_Hits_All$hit_result

mdat4 <- dummy_cols(model_data4, remove_selected_columns = TRUE)

dtest <- xgb.DMatrix(data = as.matrix(mdat4))

xgb_preds <- predict(bst_1, dtest)


xgb_preds_conv <- rep(NA, length(xgb_preds))
for(i in 1:nrow(t1)){
  xgb_preds_conv[which(xgb_preds == t1$new_resp[i])] <- t1$model_response[i]
}

table(model_response4, xgb_preds_conv)
```

```
##                   xgb_preds_conv
## model_response4   Double Error Home Run Out Single Triple
##   Double              28     0        5  11      8      1
##   Error                0     0        0   6      1      0
##   Fielders Choice      0     0        0   1      0      0
##   Home Run             1     0       27   0      0      0
##   Out                 19     1        5 498     22      3
##   Single               8     0        0  45    107      0
##   Triple               3     0        1   2      1      0
```

Creating Weather Columns

```
library(lubridate)
Rays_Home_Hits_All$month <- month(Rays_Home_Hits_All$game_date, label=FALSE)
Rays_Home_Hits_All$avg_temp <- 72
Rays_Home_Hits_All$avg_wind <- 0

Steinbrenner_BiP <- minor_res %>%
  filter(batting_team == "Tampa Tarpons") %>%
  filter(details.isInPlay == TRUE)
Steinbrenner_BiP <- Steinbrenner_BiP %>%
  mutate(hit_result = ifelse(details.isOut == TRUE, "Out", result.event)) %>%
  mutate(hit_result = recode(hit_result,
                             "Field Error" = "Error"))

Steinbrenner_BiP$month <- as.integer(month(Steinbrenner_BiP$game_date, label=FALSE))
Steinbrenner_BiP$avg_temp <- ifelse(Steinbrenner_BiP$month == 4, 73.23,
                            ifelse(Steinbrenner_BiP$month == 5, 82.08,
                            ifelse(Steinbrenner_BiP$month == 6, 82.67,
                            ifelse(Steinbrenner_BiP$month == 7, 83.86,
                            ifelse(Steinbrenner_BiP$month == 8, 83.28, 82.44)))))
Steinbrenner_BiP$avg_wind <- ifelse(Steinbrenner_BiP$month == 4, 7.99,
                              ifelse(Steinbrenner_BiP$month == 5, 7.4,
                              ifelse(Steinbrenner_BiP$month == 6, 6.82,
                              ifelse(Steinbrenner_BiP$month == 7, 5.13,
                              ifelse(Steinbrenner_BiP$month == 8, 7.02, 6.25)))))
```

Removing NAs for Tarpons Full Season Data

```
Steinbrenner_BiP$hitData.launchSpeed[is.na(Steinbrenner_BiP$hitData.launchSpeed)] <-
  mean(Steinbrenner_BiP$hitData.launchSpeed, na.rm = TRUE)

Steinbrenner_BiP$hitData.totalDistance[is.na(Steinbrenner_BiP$hitData.totalDistance)] <-
  mean(Steinbrenner_BiP$hitData.totalDistance, na.rm = TRUE)

Steinbrenner_BiP$hitData.launchAngle[is.na(Steinbrenner_BiP$hitData.launchAngle)] <-
  mean(Steinbrenner_BiP$hitData.launchAngle, na.rm = TRUE)
```

Full Model Train

```
model_data5 <- Steinbrenner_BiP[, c( "hitData.coordinates.coordX",
                                     "hitData.coordinates.coordY",
                                     "hitData.trajectory", "hitData.totalDistance",
                                     "hitData.launchAngle", "hitData.launchSpeed",
                                     "avg_temp", "avg_wind")]
model_response5 <- Steinbrenner_BiP$hit_result

summary(as.factor(model_response5))
```

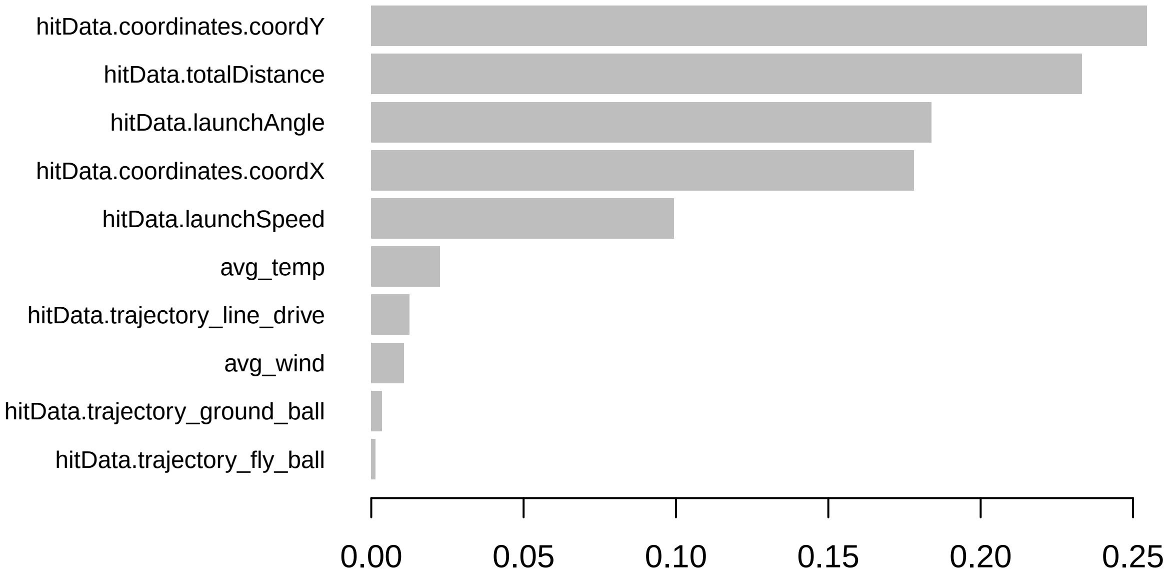
```
##          Double           Error Fielders Choice        Home Run             Out
##             112              32               8              47            1057
##          Single          Triple
##             355              15
```

```
new_resp <- as.numeric(as.factor(model_response5)) - 1

mdat5 <- dummy_cols(model_data5, remove_selected_columns = TRUE)

t1 <- cbind.data.frame(new_resp, model_response5)
```

```
t1 <- unique(t1)

# Create training matrix
dtrain <- xgb.DMatrix(data = as.matrix(mdat5), label = new_resp)

set.seed(111111)
bst_1 <- xgboost(data = dtrain, # Set training data
               num_class = length(unique(new_resp)),
               nrounds = 100, # Set number of rounds

               verbose = 1, # 1 - Prints out fit
                print_every_n = 20, # Prints out result every 20th iteration

               objective = "multi:softmax", # Set objective
               eval_metric = "merror") # Set evaluation metric to use
```

```
## [1]  train-merror:0.120541
## [21] train-merror:0.046740
## [41] train-merror:0.007380
## [61] train-merror:0.000000
## [81] train-merror:0.000000
## [100]    train-merror:0.000000
```

```
# Extract importance
imp_mat <- xgb.importance(model = bst_1)
# Plot importance (top 10 variables)
xgb.plot.importance(imp_mat, top_n = 10)
```



Full Model Predictions

```
model_data6 <- Rays_Home_Hits_All[, c( "hc_x", "hc_y", "bb_type",
                                       "hit_distance_sc", "launch_angle",
                                       "launch_speed", "avg_temp", "avg_wind")]
names(model_data6) <- names(model_data5)
model_response6 <- Rays_Home_Hits_All$hit_result

mdat6 <- dummy_cols(model_data6, remove_selected_columns = TRUE)

dtest <- xgb.DMatrix(data = as.matrix(mdat6))

xgb_preds <- predict(bst_1, dtest)

xgb_preds_conv <- rep(NA, length(xgb_preds))
for(i in 1:nrow(t1)){
  xgb_preds_conv[which(xgb_preds == t1$new_resp[i])] <- t1$model_response[i]
}

table(model_response6, xgb_preds_conv)
```

```
##                  xgb_preds_conv
## model_response6   Double Error Home Run Out Single Triple
##   Double              36     0        2   7      6      2
##   Error                0     0        0   6      1      0
##   Fielders Choice      0     0        0   1      0      0
##   Home Run             3     0       25   0      0      0
##   Out                 17     1        2 505     20      3
##   Single               5     1        0  39    114      1
##   Triple               3     0        0   3      0      1
```