Predicting Pitched Ball Velocity in Baseball

Billy Lozowski

2025-04-22

Contents

Installing Java

First, we'll load the packages required for our analysis.

- *If the packages aren't already installed, run the following code:

install.packages("tidyverse")

install.packages("tictoc")

Sys.setenv(JAVA_HOME = "C:/Program Files/Java/jdk-") install.packages("rJava")

install.packages("glmulti")

install.packages("party")

devtools::install_github("dustinfife/flexplot")*

devtools::install_github("strengejacke/strengejacke")

install.packages("sjPlot")

We'll then load our data "Pitch Speed.csv"

```
ball.speed <- read.csv("Pitch Speed.csv", header = TRUE)
```

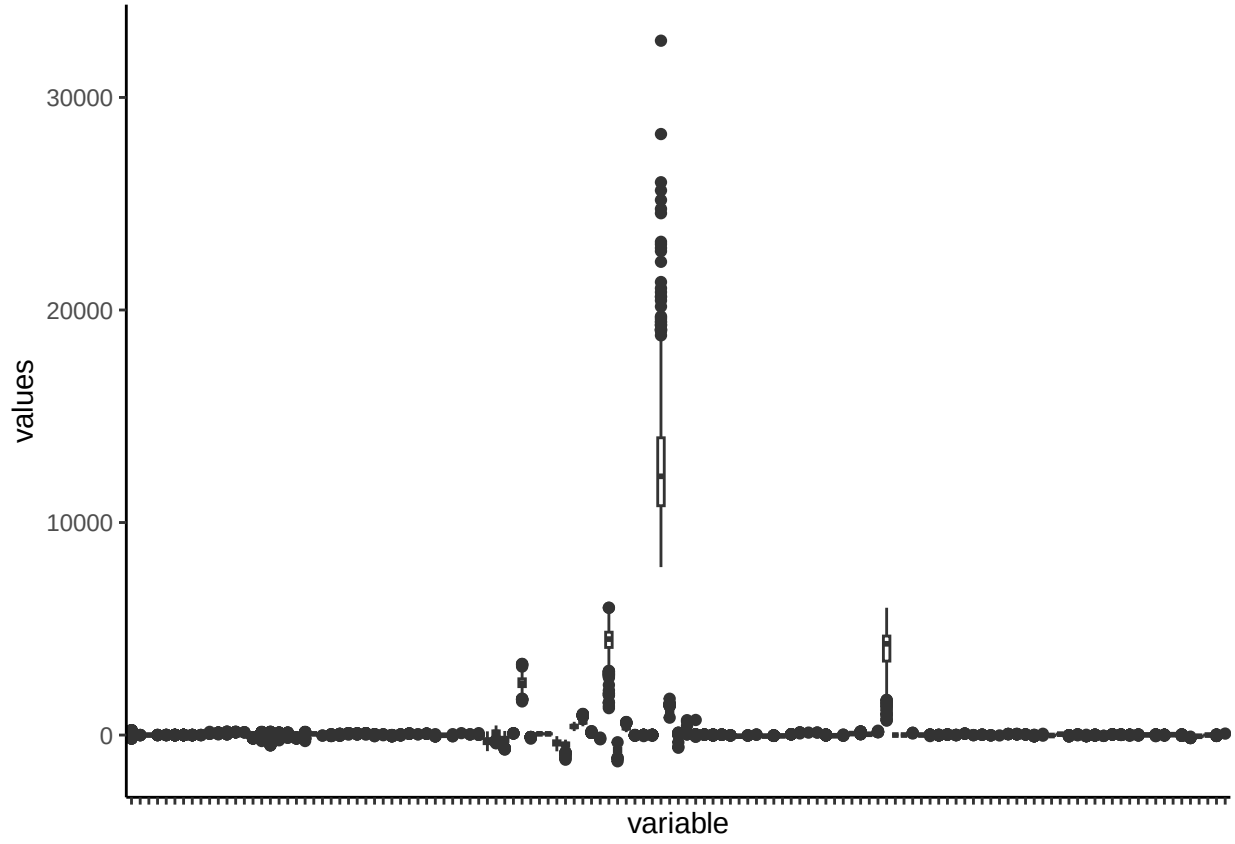
We're going to subset the data to include only the variables we're interested in

```
# subset data
ball.speed.subset <- ball.speed %>%
  filter(Pitch.Type == "Fastball") %>%
  # select only columns with data we're interested in
  select(RelSpeed..m.s., 24:ncol(ball.speed))
```

Now we'll 'clean' the data. First, we'll plot each variable, then we'll see if we need to remove any potential outliers. To do this, we'll first pivot the data to a long format.

```
# pivot the data frame
ball.speed.subset.long <- ball.speed.subset %>%
  pivot_longer(cols = everything(),
               names_to = "variable",
               values_to = "values")
```

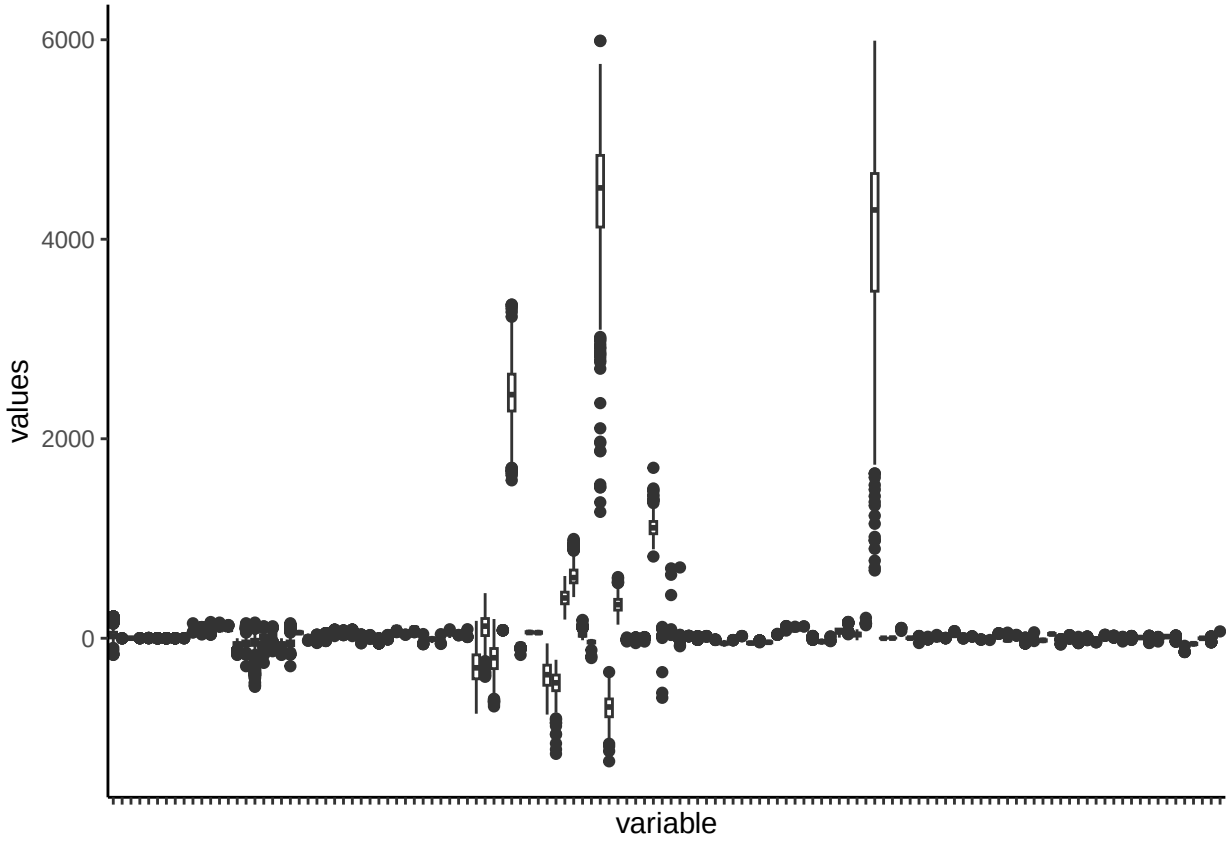
```r
# plot the data
ggplot(ball.speed.subset.long, aes(variable, values)) +
  geom_boxplot() +
  theme_classic() +
  theme(axis.text.x = element_blank())
```



Plotting the data doesn't really help us with values on the lower end because of some extreme values for one particular variable)"max trunk rotation acceleration". As such, we'll remove this variable and re-plot the data.

```r
options(scipen = 999) # prevent scientific number output

# calculate column means to see which variable is the extreme one
column.means <- as.data.frame(colMeans(ball.speed.subset))

# remove "max trunk rotation acceleration"
ball.speed.subset <- ball.speed.subset %>%
  select(-Max.Trunk.Rotation.Acceleration..deg.s.2.)


# pivot the data frame
ball.speed.subset.long <- ball.speed.subset %>%
  pivot_longer(cols = everything(),
               names_to = "variable",
               values_to = "values")
```

```r
# plot the data
ggplot(ball.speed.subset.long, aes(variable, values)) +
  geom_boxplot() +
  theme_classic() +
  theme(axis.text.x = element_blank())
```

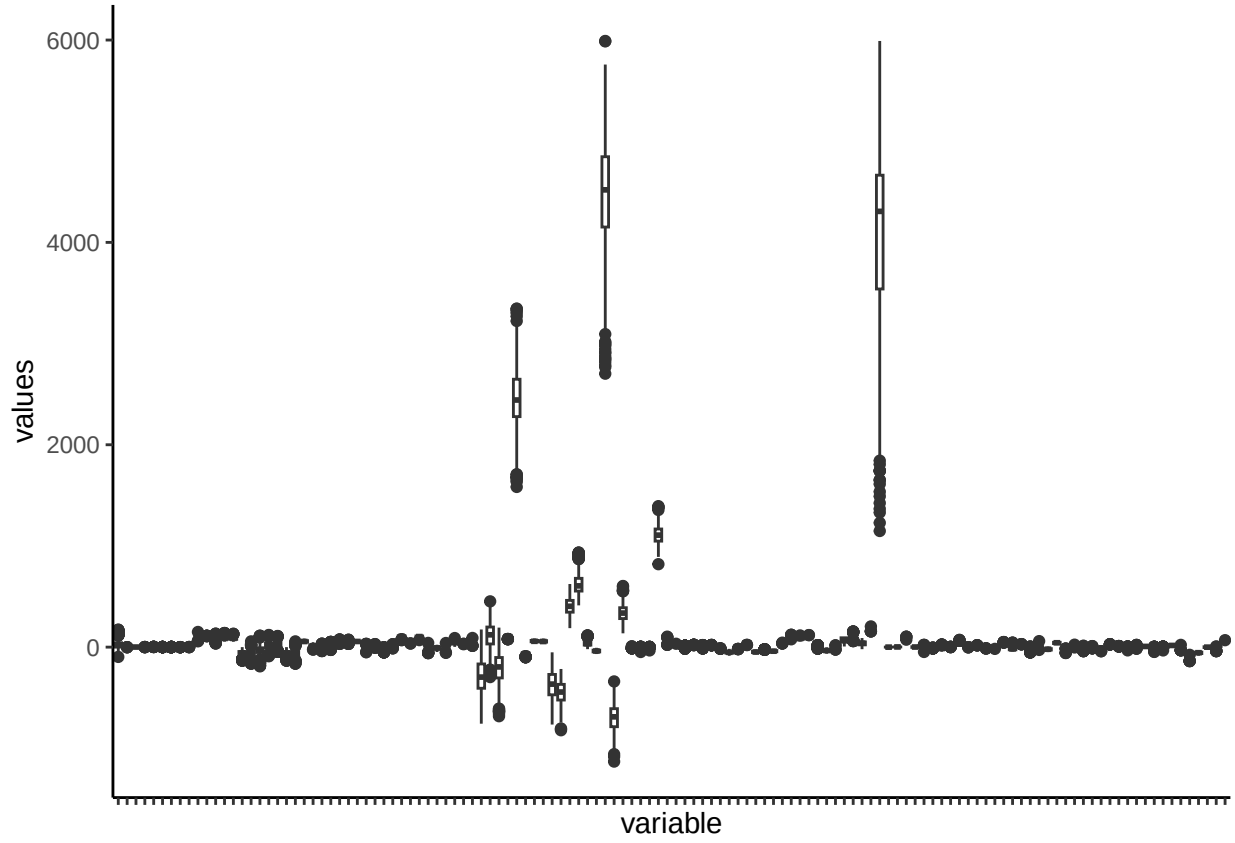


This is a bit better, but it's still not very clear. Given the scale of difference in variable values, it doesn't seem appropriate to keep removing the highest values. So, instead, since outliers do seem to be present in the data, we'll go ahead and remove these.

```r
# loop over each numeric column in a data frame
remove_outliers <- function(data) {
  data[] <- lapply(data, function(column) {
    if (is.numeric(column)) {

      # calculate Q1, Q3, and IQR
      Q1 <- quantile(column, 0.25, na.rm = TRUE)
      Q3 <- quantile(column, 0.75, na.rm = TRUE)
      IQR_val <- Q3 - Q1

      # set thresholds for outliers
      lower_limit <- Q1 - 2 * IQR_val
      upper_limit <- Q3 + 2 * IQR_val

      # replace outliers with NA (or you can choose to filter them out)
```

```
      column[column < lower_limit | column > upper_limit] <- NA
    }
    return(column)
  })
  return(data)
}

# apply the function to your data frame
ball.speed.clean <- remove_outliers(ball.speed.subset)

# pivot the cleaned data frame
ball.speed.clean.long <- ball.speed.clean %>%
  pivot_longer(cols = everything(),
               names_to = "variable",
               values_to = "values")

# plot the cleaned data
ggplot(ball.speed.clean.long, aes(variable, values)) +
  geom_boxplot() +
  theme_classic() +
  theme(axis.text.x = element_blank())
```

```
## Warning: Removed 912 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



Since we've removed outliers, we'll need to account for this so our random forest model can run properly.

```r
# filter the missing values from the data set
ball.speed.filtered <- ball.speed.clean %>%
  filter(!if_any(everything(), is.na))
```

Now we're going to construct the random forest (rf) model for ball speed.

```r
# construct the rf model
rf.ball.speed <- cforest(RelSpeed..m.s. ~ ., data = ball.speed.filtered)
```

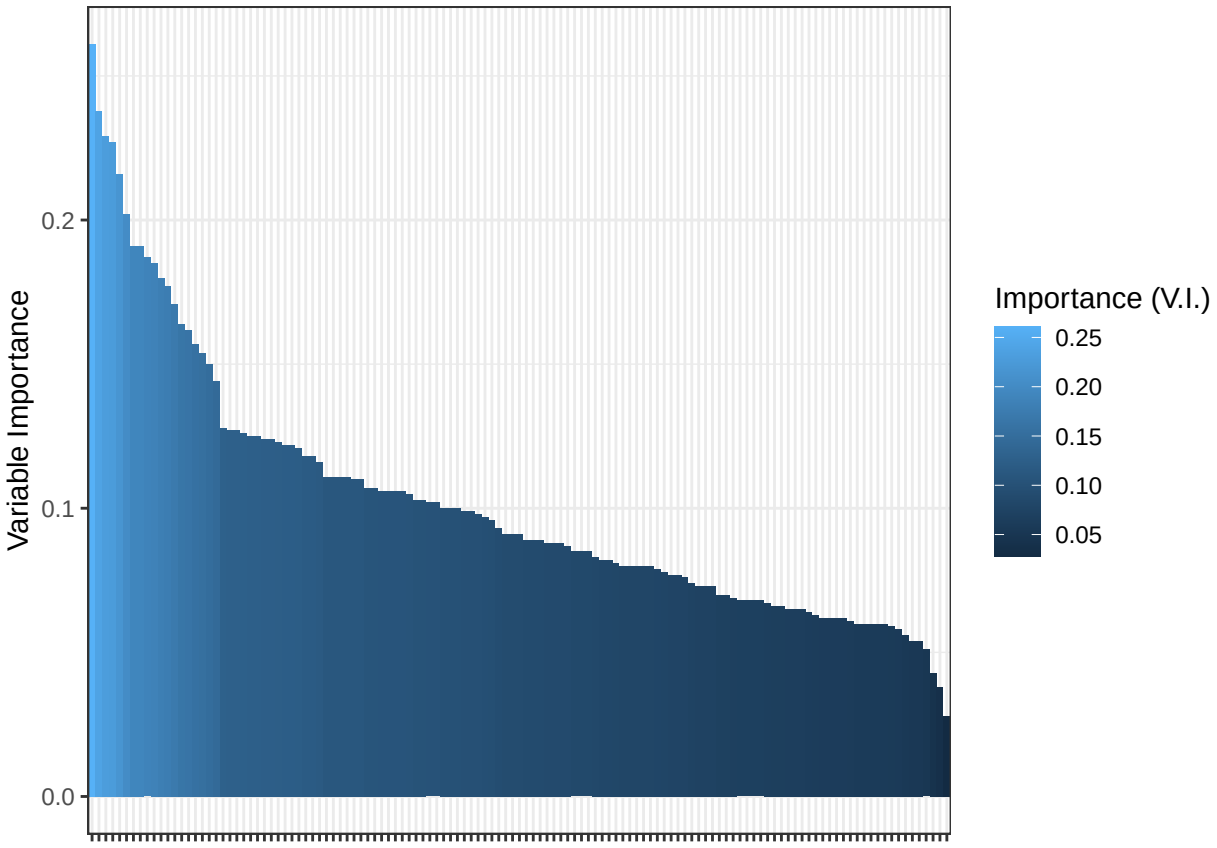
To determine which variables we want to keep in our model later on, we'll use "variable importance" as a guide!

Mahieu, Qannari & Jaillais (2023)

```r
# extract the estimates for rf.ball.speed
estimates.ball.speed <- estimates(rf.ball.speed)

# create a data frame with variables and their importance in
variable.importance <- as.data.frame(estimates.ball.speed[["importance"]])

# add a new column for the variable name and re-order the columns
variable.importance <- variable.importance %>%
  rownames_to_column(var = "Variable") %>%
  select(Variable, everything())

# rename the importance column
colnames(variable.importance)[2] <- "Importance (V.I.)"

# ensure variables are arranged by their importance (highest to lowest)
variable.importance <- variable.importance %>%
  arrange(desc(`Importance (V.I.)`))%>%
  mutate(Variable = factor(Variable, levels = Variable))
```

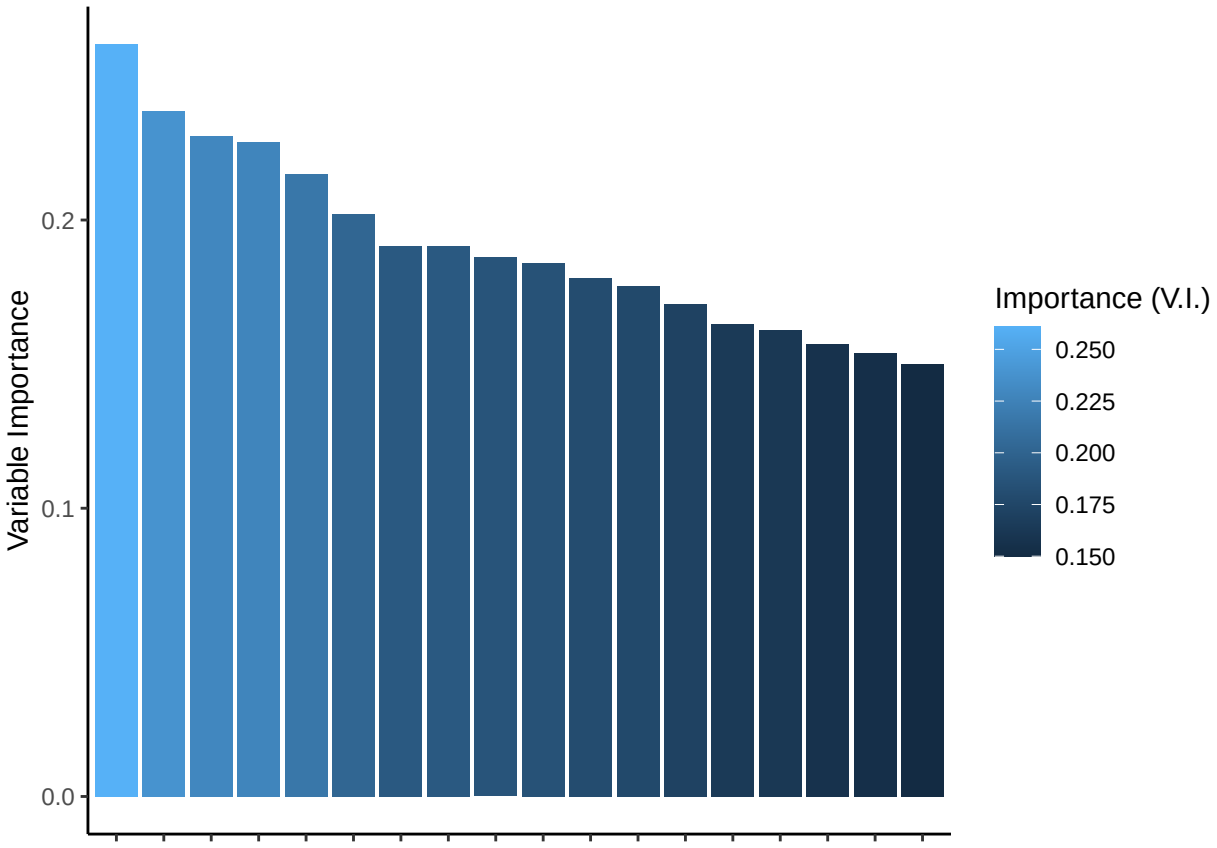
Since the variables are arranged in order of their importance, we'll visualise this to see if there are any clear cut-offs (i.e. variables we should vs. variables we maybe shouldn't include).

```r
ggplot(variable.importance, aes(Variable, `Importance (V.I.)`, fill = `Importance (V.I.)`)) +
geom_bar(stat = "identity") +
  theme_bw() +
  labs(x = NULL,
       y = "Variable Importance") +
  theme(axis.text.x = element_blank(),
        x.ticks = NULL)
```



From this first plot, It seems as if there might be a cut-off in terms of our variable importance. Let's zoom in and see if we can get a better idea how many variables this includes.

```
# subset the data
subset1 <- subset(variable.importance, `Importance (V.I.)` >= 0.15)

# plot the subset data
ggplot(subset1, aes(Variable, `Importance (V.I.)`, fill = `Importance (V.I.)`)) +
geom_bar(stat = "identity") +
  theme_classic() +
  labs(x = NULL,
       y = "Variable Importance") +
  theme(axis.text.x = element_blank(),
        x.ticks = NULL)
```



Four variables are clearly out ahead on their own. As such, we'll subset the data again to include those 4 (i.e. V.I. values >= 0.2), and move onto our 'glmulti' analysis.

```
# subset the data
subset2 <- subset1 %>%
  filter(`Importance (V.I.)` >= 0.2)

# return the variable names
subset2$Variable
```

```
## [1] Stride.Orientation                    Step.Width..m.
## [3] Lead.Ankle.Flexion.at.FC..deg.        Center.of.Mass.A.P.at.FC.Zeroed..m.
## [5] Max.Lead.Hip.Flexion.Velocity..deg.s. Center.of.Mass.M.L.at.FC.Zeroed..m.
## 125 Levels: Stride.Orientation Step.Width..m. ... Trail.Ankle.Eversion.Inversion.at.FC..deg.
```

Our 4 variables are:

[1] Step.Width..m.
[2] Lead.Ankle.Flexion.at.FC..deg.
[3] Center.of.Mass.A.P.at.FC.Zeroed..m. [4] Stride.Orientation

We'll output a general summary of how many regression models we'll assess.

```
glmulti(RelSpeed..m.s. ~
          Step.Width..m. +
          Lead.Ankle.Flexion.at.FC..deg. +
```

```
          Center.of.Mass.A.P.at.FC.Zeroed..m. +
          Stride.Orientation,
        data = ball.speed.filtered,
        crit = bic,          # model fit criterion
        level = 1,           # 1 without interactions, 2 with interactions
        method = "d",        # simple summary of candidates
                             # other options include "g" or "h"
        family = gaussian,
        fitfunction = glm,   # specify the model type (lm or glm)
        confsetsize = 100)   # keep the 100 best models (confidence set)
```

```
## Initialization...
## TASK: Diagnostic of candidate set.
## Sample size: 311
## 0 factor(s).
## 4 covariate(s).
## 0 f exclusion(s).
## 0 c exclusion(s).
## 0 f:f exclusion(s).
## 0 c:c exclusion(s).
## 0 f:c exclusion(s).
## Size constraints: min =  0 max = -1
## Complexity constraints: min =  0 max = -1
## Your candidate set contains 16 models.

## [1] 16
```

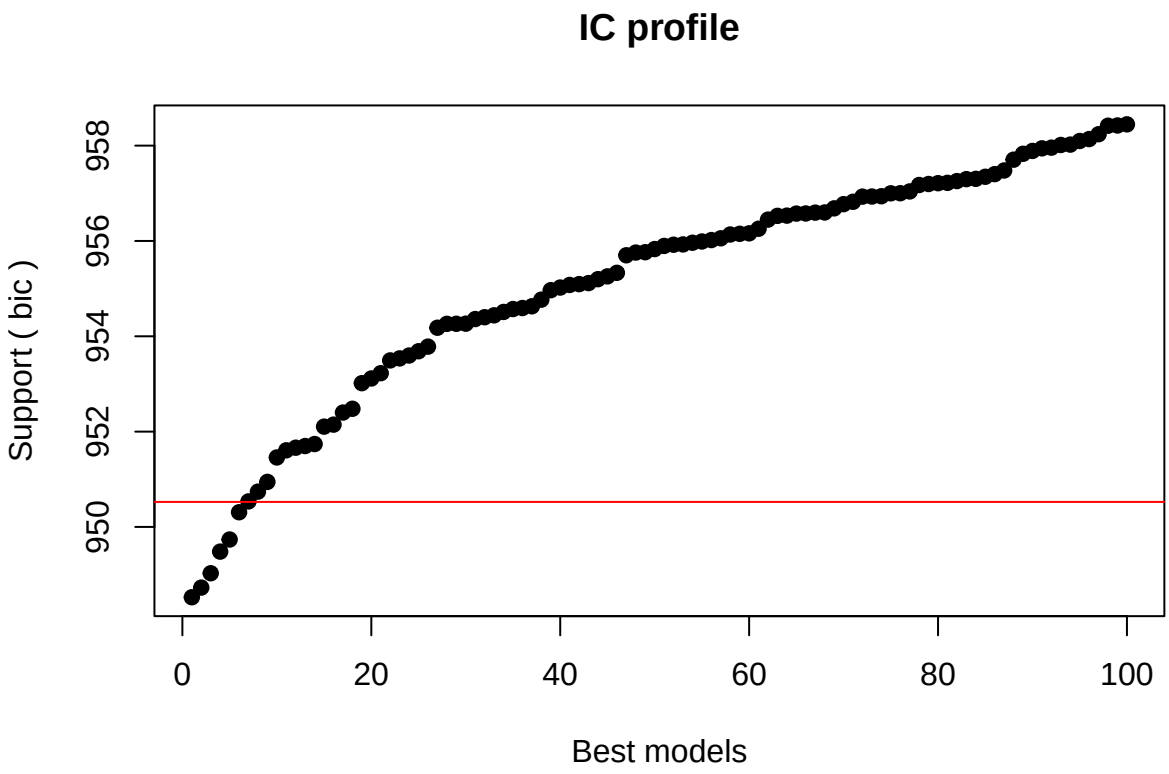
We'll now build some models to see whether we can remove any of these using a the "glmulti" packagen and an exhaustive algorithm. It's advised to refer to the documentation to ensure the correct/desired model parameters are set if you have more predictor variables.

Extract model information

```
# model results
print(g.model)
```

```
## glmulti.analysis
## Method: h / Fitting: glm / IC used: bic
## Level: 2 / Marginality: FALSE
## From 100 models:
## Best IC: 948.52516677856
## Best model:
## [1] "RelSpeed..m.s. ~ 1 + Lead.Ankle.Flexion.at.FC..deg. + Lead.Ankle.Flexion.at.FC..deg.:Step.Width
## [2] "    Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg. + "
## [3] "    Stride.Orientation:Lead.Ankle.Flexion.at.FC..deg."
## Evidence weight: 0.106373647157659
## Worst IC: 958.446091378584
## 6 models within 2 IC units.
## 62 models to reach 95% of evidence weight.
```

```
# plot the best 100 models base on information criterion (IC)
plot(g.model) # red line indicates 2 IC units from the best model
```



See which variables the best models contain (in this case, the best 4 models).

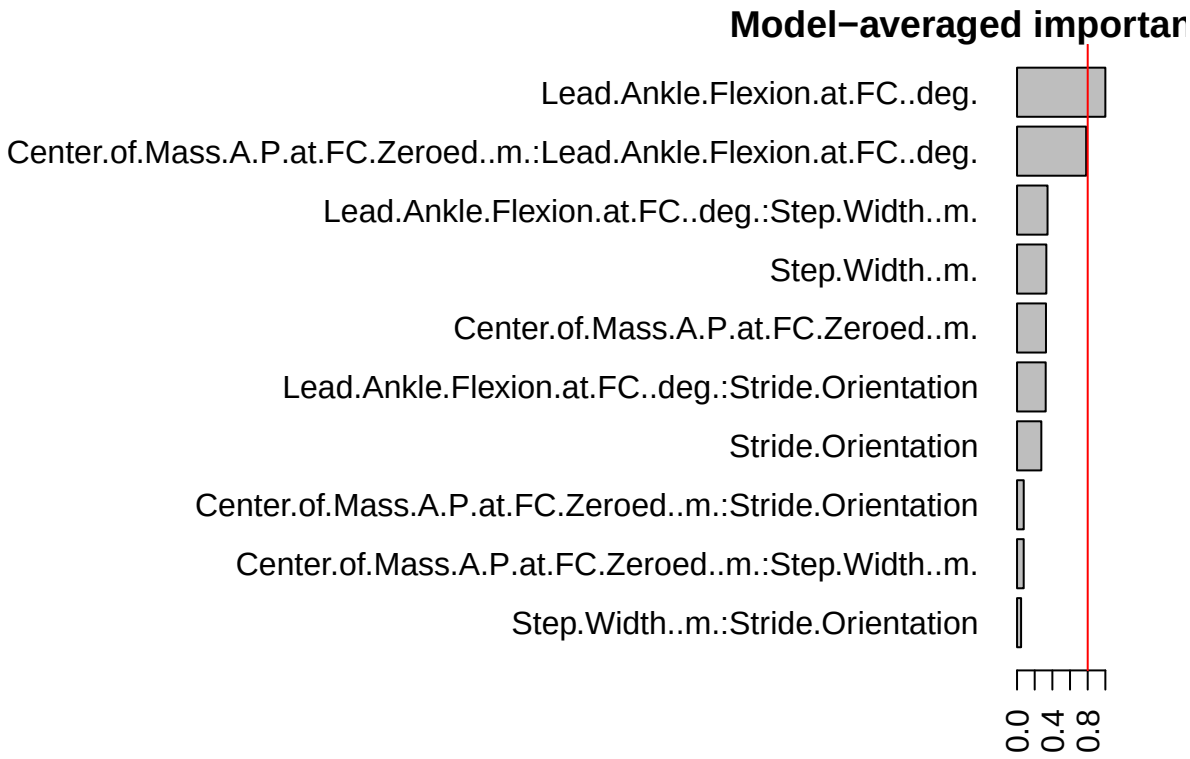
```
weightable(g.model)[1:6,] %>% # select the best 6 models
  regulartable() %>%
  autofit()
```

```
## Warning: fonts used in ‘flextable‘ are ignored because the ‘pdflatex‘ engine is
## used and not ‘xelatex‘ or ‘lualatex‘. You can avoid this warning by using the
## ‘set_flextable_defaults(fonts_ignore=TRUE)‘ command or use a compatible engine
## by defining ‘latex_engine: xelatex‘ in the YAML header of the R Markdown
## document.
```

model
RelSpeed..m.s. ~ 1 + Lead.Ankle.Flexion.at.FC..deg. + Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m. + Center.o
RelSpeed..m.s. ~ 1 + Lead.Ankle.Flexion.at.FC..deg. + Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.a
RelSpeed..m.s. ~ 1 + Step.Width..m. + Lead.Ankle.Flexion.at.FC..deg. + Stride.Orientation + Center.of.Mass.A.P
RelSpeed..m.s. ~ 1 + Step.Width..m. + Lead.Ankle.Flexion.at.FC..deg. + Center.of.Mass.A.P.at.FC.Zeroed..m.:Le
RelSpeed..m.s. ~ 1 + Lead.Ankle.Flexion.at.FC..deg. + Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m. + Center.o
RelSpeed..m.s. ~ 1 + Lead.Ankle.Flexion.at.FC..deg. + Stride.Orientation + Center.of.Mass.A.P.at.FC.Zeroed..m.

We're going to save the model importance terms as a PNG file so we can see which variables are included in 80+% of models.

```r
# adjust the plot margins
par(mar = c(5, 25, 1, 2) + 0.1)

# plot the model so it shoes in the plot viewer
plot(g.model, type = "s")
```




```r
################################################################################

# open a PNG device with specified width and height
png(filename = "Model Importance Terms.png", width = 1800, height = 1200, res = 150)

# set up the layout to include extra space for the y-axis labels
layout(matrix(1), widths = c(1.5), heights = c(1))

# adjust the plot margins
par(mar = c(5, 25, 1, 2) + 0.1)

# plot the model
plot(g.model, type = "s")

# close the PNG device
dev.off()
```

```
## pdf
##   2
```

So, it looks like [1] **Lead.Ankle.Flexion.at.FC..deg.** and its interaction with COM position in the AP ([1] **Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg.**) might be most important when it comes to biomechanics explaining pitched ball velocity! Interestingly, they're both included in the model with the highest weighting and the second strongest model. However, this second model exclusively has these two terms. This may suggest that the second strongest model may be most appropriate when trying to explain pitched ball velocity.

```
first.model <- lm(RelSpeed..m.s. ~
                    # predictors
                    Lead.Ankle.Flexion.at.FC..deg. +
                    Lead.Ankle.Flexion.at.FC..deg. +
                    # interactions
                    Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m. +
                    Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg. +
                    Stride.Orientation:Lead.Ankle.Flexion.at.FC..deg.,
                  data = ball.speed.filtered)

# summarise the final model
summary(first.model)
```

```
##
## Call:
## lm(formula = RelSpeed..m.s. ~ Lead.Ankle.Flexion.at.FC..deg. +
##     Lead.Ankle.Flexion.at.FC..deg. + Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m. +
##     Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg. +
##     Stride.Orientation:Lead.Ankle.Flexion.at.FC..deg., data = ball.speed.filtered)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.58933 -0.77920 -0.05688  0.68141  2.92182
##
## Coefficients:
##                                                                 Estimate
## (Intercept)                                                    43.787128
## Lead.Ankle.Flexion.at.FC..deg.                                 -0.062826
## Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m.                   0.452892
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.  0.008280
## Lead.Ankle.Flexion.at.FC..deg.:Stride.Orientation              -0.012180
##                                                                Std. Error
## (Intercept)                                                      0.476064
## Lead.Ankle.Flexion.at.FC..deg.                                   0.009484
## Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m.                    0.162986
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.   0.001811
## Lead.Ankle.Flexion.at.FC..deg.:Stride.Orientation                0.004631
##                                                                t value
## (Intercept)                                                     91.977
## Lead.Ankle.Flexion.at.FC..deg.                                  -6.625
## Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m.                    2.779
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.   4.572
## Lead.Ankle.Flexion.at.FC..deg.:Stride.Orientation               -2.630
##                                                                         Pr(>|t|)
## (Intercept)                                                    < 0.0000000000000002
## Lead.Ankle.Flexion.at.FC..deg.                                      0.000000000157
## Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m.                         0.00579
```

```
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.       0.000007017022
## Lead.Ankle.Flexion.at.FC..deg.:Stride.Orientation                              0.00897
##
## (Intercept)                                                           ***
## Lead.Ankle.Flexion.at.FC..deg.                                        ***
## Lead.Ankle.Flexion.at.FC..deg.:Step.Width..m.                         **
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m. ***
## Lead.Ankle.Flexion.at.FC..deg.:Stride.Orientation                     **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.061 on 306 degrees of freedom
## Multiple R-squared:  0.2415, Adjusted R-squared:  0.2316
## F-statistic: 24.36 on 4 and 306 DF,  p-value: < 0.00000000000000022
```

```
second.model <- lm(RelSpeed..m.s. ~
                     # predictors
                     Lead.Ankle.Flexion.at.FC..deg. +
                     # interactions
                     Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg.,
                     data = ball.speed.filtered)
# summarise the final model
summary(second.model)
```

```
##
## Call:
## lm(formula = RelSpeed..m.s. ~ Lead.Ankle.Flexion.at.FC..deg. +
##     Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg.,
##     data = ball.speed.filtered)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.46813 -0.73308 -0.07783  0.67104  2.97536
##
## Coefficients:
##                                                                  Estimate
## (Intercept)                                                     43.936348
## Lead.Ankle.Flexion.at.FC..deg.                                  -0.066716
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.  0.010495
##                                                                 Std. Error
## (Intercept)                                                       0.480638
## Lead.Ankle.Flexion.at.FC..deg.                                    0.009548
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.   0.001615
##                                                                 t value
## (Intercept)                                                      91.413
## Lead.Ankle.Flexion.at.FC..deg.                                   -6.987
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.   6.500
##                                                                        Pr(>|t|)
## (Intercept)                                                     < 0.0000000000000002
## Lead.Ankle.Flexion.at.FC..deg.                                       0.0000000000174
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m.      0.0000000003230
##
## (Intercept)                                                     ***
## Lead.Ankle.Flexion.at.FC..deg.                                  ***
```

```
## Lead.Ankle.Flexion.at.FC..deg.:Center.of.Mass.A.P.at.FC.Zeroed..m. ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.077 on 308 degrees of freedom
## Multiple R-squared:  0.2125, Adjusted R-squared:  0.2074
## F-statistic: 41.55 on 2 and 308 DF,  p-value: < 0.00000000000000022
```

```r
# create a table of the final model outputs, and save as a word .doc
tab_model(first.model, second.model,
          show.df = TRUE,
          show.se = TRUE,
          string.se = "SE",
          dv.labels = c("Strongest Model", "Second Strongest Model"),
          file = "Pitch Ball Velocity Models.doc")
```

Strongest Model

Second Strongest Model

Predictors

Estimates

SE

CI

p

df

Estimates

SE

CI

p

df

(Intercept)

43.79

0.48

42.85 – 44.72

<0.001

306.00

43.94

0.48

42.99 – 44.88

<0.001

308.00

Lead Ankle Flexion at FCdeg

-0.06

0.01

-0.08 – -0.04

<0.001

306.00

-0.07

0.01

-0.09 – -0.05

<0.001

308.00

Lead Ankle Flexion at FCdeg × Step Width m

0.45

0.16

0.13 – 0.77

0.006

306.00

Lead Ankle Flexion at FCdeg × Center of Mass A Pat FC Zeroed m

0.01

0.00

0.00 – 0.01

<0.001

306.00

0.01

0.00

0.01 – 0.01

<0.001

308.00

Lead Ankle Flexion at FCdeg × Stride Orientation

-0.01

0.00

-0.02 – -0.00

0.009

306.00

Observations

311

311

R2 / R2 adjusted

0.242 / 0.232

0.212 / 0.207

Our strongest model accounted for ~24% of the variance in pitched ball velocity, and our second strongest model accounted for ~21%. Since these values are so close, let's run a quick model comparison between the top two to make sure we pick the right one!

```
# model comparison
model_performance(first.model)
```

```
## # Indices of model performance
##
## AIC     |    AICc |     BIC |    R2 | R2 (adj.) |  RMSE | Sigma
## ----------------------------------------------------------------
## 926.086 | 926.363 | 948.525 | 0.242 |     0.232 | 1.052 | 1.061
```

```
model_performance(second.model)
```

```
## # Indices of model performance
##
## AIC     |    AICc |     BIC |    R2 | R2 (adj.) |  RMSE | Sigma
## ----------------------------------------------------------------
## 933.769 | 933.899 | 948.728 | 0.212 |     0.207 | 1.072 | 1.077
```

```
# compare models with performance package
compare_performance(first.model, second.model, rank = T)
```

```
## # Comparison of Model Performance Indices
##
## Name         | Model |    R2 | R2 (adj.) |  RMSE | Sigma | AIC weights
## ------------------------------------------------------------------------
## first.model  |    lm | 0.242 |     0.232 | 1.052 | 1.061 |       0.979
## second.model |    lm | 0.212 |     0.207 | 1.072 | 1.077 |       0.021
##
## Name         | AICc weights | BIC weights | Performance-Score
## ---------------------------------------------------------------
## first.model  |        0.977 |       0.525 |           100.00%
## second.model |        0.023 |       0.475 |             0.00%
```

```
# compare models with flexplot package
model.comparison(first.model, second.model)
```

```
## $statistics
##                  aic     bic bayes.factor     p   rsq adj.rsq
## first.model  926.086 948.525        1.107 0.003 0.242   0.232
## second.model 933.769 948.728        0.904       0.212   0.207
##
## $predicted_differences
##    0%   25%   50%   75%  100%
## 0.001 0.069 0.134 0.230 0.661
```

```
# # visualise model fit indices
# plot(compare_performance(first.model, second.model))
```

Model fit criteria are almost identical for both models. While the AIC and BIC are lower for the model with more terms included, the Bayes Factor suggests that the difference between them is negligible. Given that our model-averaged variable importance plot showed that **Lead.Ankle.Flexion.at.FC..deg.** and **Center.of.Mass.A.P.at.FC.Zeroed..m.:Lead.Ankle.Flexion.at.FC..deg.** were by far the most common model terms across all computed models, we will select the model that includes just these (model 2). With ~21% of explained variance in pitched ball speed, it seems that other variables other than the ones identified here are likely to better explain pitched ball velocity in baseball.