# How Can I Get Better at Golf?

Will Lavey

2024-05-20

```
knitr::opts_chunk$set(echo = TRUE)

pacman::p_load(tidyverse, ggplot2, dplyr, caret, rpart, rpart.plot, MASS, psych, string
r, vtable, randomForest)


# This confusion matrix design was taken from stack overflow.
# https://stackoverflow.com/questions/23891140/r-how-to-visualize-confusion-matrix-using
-the-caret-package
draw_confusion_matrix <- function(cm) {

  total <- sum(cm$table)
  res <- as.numeric(cm$table)

  # Generate color gradients. Palettes come from RColorBrewer.
  greenPalette <- c("#f7fdff","#cff2ff","#ace7fc","#75daff","#4cc8f5","#37b1de","#2295b
f","#107aa1","#015a7a")
  redPalette <- c("#FFF5F0","#FEE0D2","#FCBBA1","#FC9272","#FB6A4A","#EF3B2C","#CB181
D","#A50F15","#67000D")
  getColor <- function (greenOrRed = "blue", amount = 0) {
    if (amount == 0)
      return("#FFFFFF")
    palette <- greenPalette
    if (greenOrRed == "red")
      palette <- redPalette
    colorRampPalette(palette)(100)[10 + ceiling(90 * amount / total)]
  }

  # set the basic layout
  layout(matrix(c(1,1,2)))
  par(mar=c(2,2,2,2))
  plot(c(100, 345), c(300, 450), type = "n", xlab="", ylab="", xaxt='n', yaxt='n')
  title('CONFUSION MATRIX', cex.main=2)

  # create the matrix
  classes = colnames(cm$table)
  rect(150, 430, 240, 370, col=getColor("blue", res[1]))
  text(195, 435, "Has Not Won", cex=1.2)
  rect(250, 430, 340, 370, col=getColor("red", res[3]))
  text(295, 435, "Has Won", cex=1.2)
  text(125, 370, 'Predicted', cex=1.3, srt=90, font=2)
  text(245, 450, 'Actual', cex=1.3, font=2)
  rect(150, 305, 240, 365, col=getColor("red", res[2]))
  rect(250, 305, 340, 365, col=getColor("blue", res[4]))
  text(140, 400, "Has Not Won", cex=1.2, srt=90)
  text(140, 335, "Has Won", cex=1.2, srt=90)

  # add in the cm results
  text(195, 400, res[1], cex=1.6, font=2, col='white')
  text(195, 335, res[2], cex=1.6, font=2, col='white')
  text(295, 400, res[3], cex=1.6, font=2, col='white')
  text(295, 335, res[4], cex=1.6, font=2, col='white')
```

```
  # add in the specifics
  plot(c(100, 0), c(100, 0), type = "n", xlab="", ylab="", main = "DETAILS", xaxt='n', y
axt='n')
  text(10, 85, names(cm$byClass[1]), cex=1.2, font=2)
  text(10, 70, round(as.numeric(cm$byClass[1]), 3), cex=1.2)
  text(30, 85, names(cm$byClass[2]), cex=1.2, font=2)
  text(30, 70, round(as.numeric(cm$byClass[2]), 3), cex=1.2)
  text(50, 85, names(cm$byClass[5]), cex=1.2, font=2)
  text(50, 70, round(as.numeric(cm$byClass[5]), 3), cex=1.2)
  text(70, 85, names(cm$byClass[6]), cex=1.2, font=2)
  text(70, 70, round(as.numeric(cm$byClass[6]), 3), cex=1.2)
  text(90, 85, names(cm$byClass[7]), cex=1.2, font=2)
  text(90, 70, round(as.numeric(cm$byClass[7]), 3), cex=1.2)

  # add in the accuracy information
  text(30, 35, names(cm$overall[1]), cex=1.5, font=2)
  text(30, 20, round(as.numeric(cm$overall[1]), 3), cex=1.4)
  text(70, 35, names(cm$overall[2]), cex=1.5, font=2)
  text(70, 20, round(as.numeric(cm$overall[2]), 3), cex=1.4)
}
```

# Introduction

While brainstorming ideas for this final project, I thought to myself: How can I leverage the computer to create something practical and valuable for myself? What aspects in my life do I want to learn more about and improve upon?

And after some consideration, I landed on golf. How can I get better at golf?

Golf is a seemingly infinitely difficult sport filled with so many complex physics concepts to disturb your perfect strike and yet so many simple mental tricks to help you get better. What does the computer have to say about what makes one a great golfer? Is it your ability to hit a baby draw 200 yards with your 7 iron? Is it your ability to minimize three putting? Data may not be able to perfectly capture what determines the perfect golf game, but hopefully can point us in the right direction.

# Data Descriptions

The data to be used are two PGA Tour data sets. One is data from each tournament in the seasons between 2015 and 2022. The other is all players' aggregate statistics over the entirety of the 2017 season. One of the most important metrics in golf is called Strokes Gained. The number relies on the average number of shots a player takes to get the ball in the cup from one's current position on the hole, which is data that has been accumulated over time. If a player hits an unusually good shot from a given position, their Strokes Gained index will be positive, and vice versa.

## Golf Scoring Explained

In golf, par is the average, or recommended number of strokes to get the ball in the hole. A course will typically have a par of 72, which consists of a variety of par 4's, par 3's and par 5's. Birdie is one under par (eg. 3 strokes on a par 4) and bogey is one over par (eg. 6 strokes on a par 5). The scoring in golf relates to how far over or

under par (average) you are. The typical scores depend on the skill of golfer and the difficulty of course. PGA Tour professionals can typically shoot around 12 under par on a good day. Many metrics in the data sets relate to these scoring values.

```
# Tournament level data PGA Tour 2015-2022
tourney_level_2015_2022 <- read.csv("/Users/Will/Documents/College/Junior/Spring 2024/ST
AT218/Datasets/ASA All PGA Raw Data - Tourn Level.csv")

# Player level data PGA Tour 2017
player_level_2017 <- read.csv("/Users/Will/Documents/College/Junior/Spring 2024/STAT218/
Datasets/PGATOUR_data2.csv")
```

```
# Remove unnused columns
tourney_clean <- tourney_level_2015_2022
tourney_clean <- subset(tourney_clean, select = -c(Unnamed..2, Unnamed..3, Unnamed..4,
                                                   hole_DKP, hole_SDP, hole_FDP,
                                                   streak_DKP, streak_SDP, streak_FDP,
                                                   finish_DKP, finish_SDP, finish_FDP,
                                                   total_DKP, total_SDP, total_FDP))

# Make pos 157 if missed cut. There are a max of 156 players allowed in a PGA tournament
tourney_clean$pos[is.na(tourney_clean$pos)] <- 157

# Remove NAs
tourney_clean <- na.omit(tourney_clean)
```

```
player <- player_level_2017

# Remove NAs
player <- na.omit(player)

# Create 'won' column for if they have won a tournament
player <- player |>
  mutate(won = case_when(NUMBER_OF_WINS == 0 ~ 0,
                         NUMBER_OF_WINS > 0 ~ 1))

player$FAIRWAYS_HIT = as.numeric(player$FAIRWAY_HIT_.)
player$TOTAL_DRIVES <- as.numeric(gsub(",", "", player$TOTAL_DRIVES))

# Scaled data used for SVM
player_scale <- player |>
  mutate(across(-all_of(c("won", "Player")), scale))
```

# Data Summaries

```
st(tourney_clean, out = "return")[, c(1, 3, 4, 5, 8)]
```

```
##         Variable      Mean Std. Dev.  Min       Max
## 1  tournament.id 249366480 194567360 2232 401353273
## 2      player.id     80107    576079    5   4845309
## 3       hole_par       223        71   70       292
## 4        strokes       221        67   66       313
## 5       n_rounds       3.1      0.99    1         4
## 6       made_cut      0.58      0.49    0         1
## 7            pos        89        64    1       999
## 8          purse       7.6         2  3.5        20
## 9         season      2019       2.2 2015      2022
## 10        no_cut     0.058      0.23    0         1
## 11       sg_putt     -0.12       1.1   -6       4.4
## 12        sg_arg    -0.041      0.73 -6.4       3.2
## 13        sg_app      -0.1       1.1 -9.2       4.7
## 14        sg_ott    -0.046      0.81 -7.7       2.8
## 15        sg_t2g     -0.19       1.6  -14       6.3
## 16      sg_total     -0.31         2  -14       8.5
```

```
st(player, out = "return")[, c(1, 3, 4, 5, 8)]
```

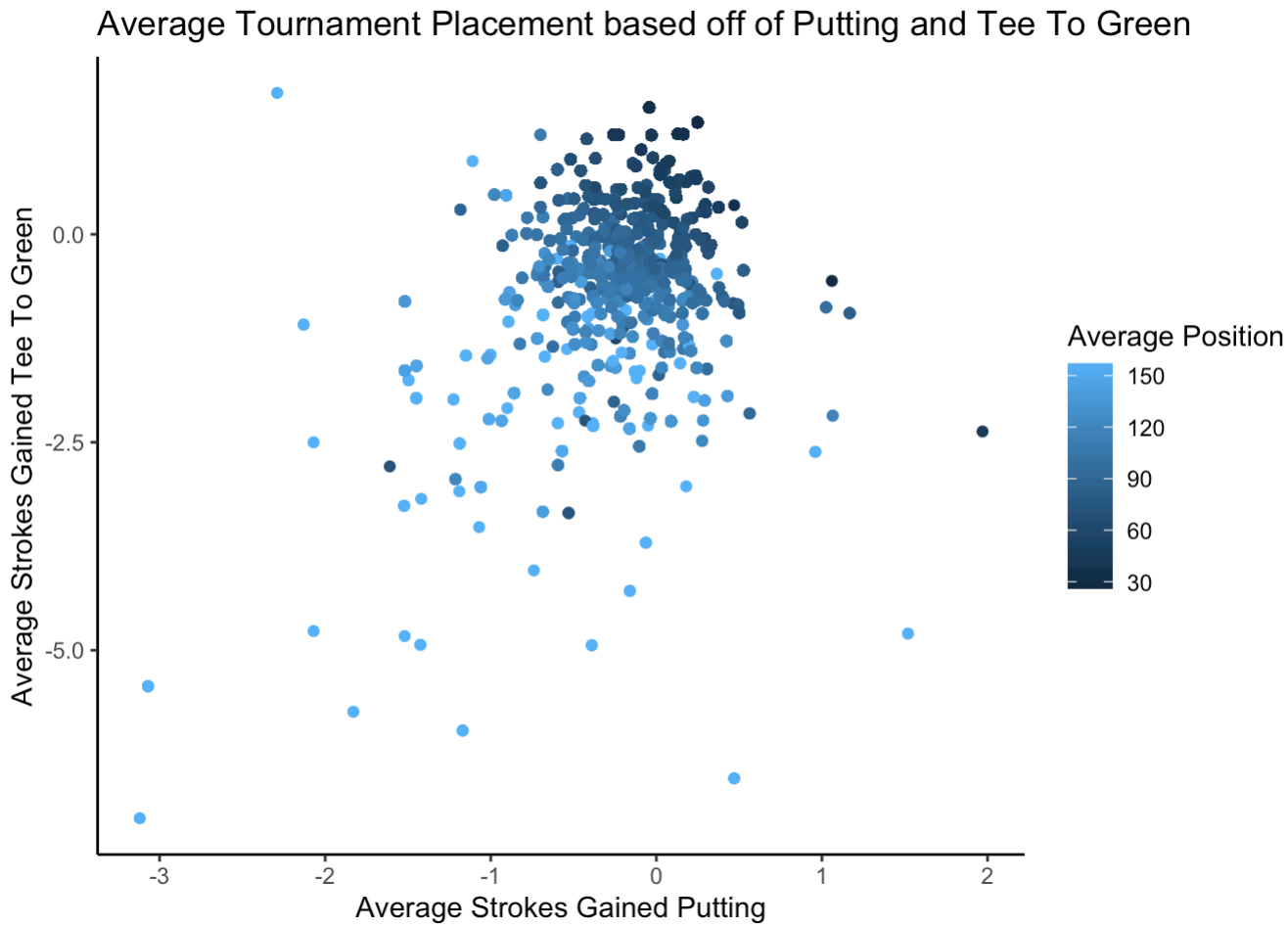
```
##                    Variable  Mean Std. Dev.   Min  Max
## 1              EVENTS_PLAYED    24       4.2    15   32
## 2                     POINTS   721       608    10 3289
## 3             NUMBER_OF_WINS  0.23      0.57     0    3
## 4         NUMBER_OF_TOP_Tens   2.6       2.2     0   11
## 5         POINTS_BEHIND_LEAD  4896       608  2328 5607
## 6              ROUNDS_PLAYED    78        14    46  110
## 7       SG_PUTTING_PER_ROUND  0.05      0.31 -0.75 0.86
## 8           TOTAL_SG.PUTTING     3        18   -43   60
## 9            MEASURED_ROUNDS    60        14    30   92
## 10      AVG_Driving_DISTANCE   297       8.2   278  320
## 11             UP_AND_DOWN_.    59       3.5    44   67
## 12             PAR_OR_BETTER   257        53   136  392
## 13                MISSED_GIR   438        87   227  635
## 14             FAIRWAY_HIT_.    61       5.1    41   74
## 15              FAIRWAYS_HIT    61       5.1    41   74
## 16         POSSIBLE_FAIRWAYS  1014       202   531 1454
## 17                   GIR_RANK    98        56     1  195
## 18     GOING_FOR_GREEN_IN_2.    56       9.1    28   78
## 19              ATTEMPTS_GFG   124        35    38  227
## 20          NON.ATTEMPTS_GFG    97        31    35  228
## 21   RTP.GOING_FOR_THE_GREEN   -68        22  -128  -22
## 22 RTP.NOT_GOING_FOR_THE_GRN  -6.8       9.9   -51   27
## 23                 HOLE_OUTS    11       4.4     1   25
## 24               SAND_SAVE.     50       5.8    34   66
## 25           NUMBER_OF_SAVES    62        16    23  108
## 26         NUMBER_OF_BUNKERS   124        29    57  193
## 27             TOTAL_O.U_PAR    40        13    15   74
## 28               Three_PUTT.   2.9      0.64   1.5  5.1
## 29             TOTAL_3_PUTTS    39        11    12   73
## 30              SG_PER_ROUND  0.13      0.75  -3.6    2
## 31                    SG.OTT  0.04      0.42  -1.6    1
## 32                    SG.APR 0.059      0.39  -1.6 0.99
## 33                    SG.ARG 0.031      0.22 -0.92 0.63
## 34              DRIVES_320..    11         7   1.2   34
## 35     TOTAL_DRIVES_FOR_320.    92        62     8  307
## 36              TOTAL_DRIVES   859       196   420 1344
## 37           ROUGH_TENDNECY.    29       3.9    17   43
## 38               TOTAL_ROUGH   244        60    93  428
## 39           FAIRWAY_BUNKER.   6.1       1.1   3.1  9.5
## 40     TOTAL_FAIRWAY_BUNKERS    52        15    18   94
## 41       AVG_CLUB_HEAD_SPEED   114       3.9   105  125
## 42          FASTEST_CH_SPEED   118       4.2   109  129
## 43          SLOWEST_CH_SPEED   110       3.8   102  119
## 44            AVG_BALL_SPEED   170       5.8   157  182
## 45        FASTEST_BALL_SPEED   174       5.8   160  188
## 46        SLOWEST_BALL_SPEED   164       6.1   146  177
## 47          AVG_SMASH_FACTOR   1.5     0.014   1.4  1.5
## 48                HIGHEST_SF   1.5    0.0059   1.5  1.5
## 49                 LOWEST_SF   1.4     0.027   1.3  1.5
## 50          AVG_LAUNCH_ANGLE    11       1.4   7.1   15
## 51       LOWEST_LAUNCH_ANGLE     7       1.6   1.2   11
```

```
## 52     STEEPEST_LAUNCH_ANGLE    15       1.6    11   19
## 53             AVG_SPIN_RATE  2634       210  2127 3346
## 54         HIGHEST_SPIN_RATE  5087      1368  2819 9640
## 55          LOWEST_SPIN_RATE  1737       220  1400 2314
## 56             AVG_HANG_TIME   6.3      0.25   5.5  6.9
## 57     LONGEST_ACT.HANG_TIME   7.6      0.33   6.8  8.7
## 58    SHORTEST_ACT.HANG_TIME   3.2       1.2   0.5  5.1
## 59        AVG_CARRY_DISTANCE   278        10   250  303
## 60    LONGEST_CARRY_DISTANCE   305        12   272  338
## 61   SHORTEST_CARRY_DISTANCE   245        13   193  276
## 62                AVG_SCORE    71      0.82    69   75
## 63            TOTAL_STROKES  5324      1006  3261 7515
## 64             TOTAL_ROUNDS    75        14    45  107
## 65             MAKES_BOGEY.    16       2.1    12   28
## 66              BOGEYS_MADE   222        42   123  330
## 67             HOLES_PLAYED  1358       260   810 1926
## 68                      AGE    33       5.6    21   49
## 69                      won  0.17      0.38     0    1
```

These two data sets have a variety of features, however both contain information on Strokes Gained. The player level data has some interesting metrics on how well a player got out of the sand, if they made risky moves and went for the green in two shots instead of three, and whether they made it into the hole from off the green in two strokes. The tournament level data may be more useful to look at what helps a player on a given day, whereas the aggregated player level data may better pick up on season-long trends.

# Exploratory Graph

```
tourney_clean |>
  group_by(player.id) |>
  mutate(avg_pos = mean(pos),
         avg_sg_putt = mean(sg_putt),
         avg_sg_t2g = mean(sg_t2g)) |>
  ggplot() +
  geom_point(aes(x = avg_sg_putt, y = avg_sg_t2g, color = avg_pos)) +
  labs(x = "Average Strokes Gained Putting",
       y = "Average Strokes Gained Tee To Green",
       title = "Average Tournament Placement based off of Putting and Tee To Green",
       color = "Average Position") +
  theme_classic()
```




It is without a doubt that better putting and golfing to the putting surface correlate with better scores, and therefore tournament placement.

# Random Forest Models

## Tournament Model Implementation

Let's now look at what best indicates good tournament placement. I have elected to use a random forest, as I am curious about variable importance.

```
tourney_train <- tourney_clean |>
  filter(pos != 157) |>
  sample_n(1000)

rf1 <- train(pos ~ sg_putt + sg_t2g + sg_arg + sg_ott + sg_app,
             data = tourney_train,
             importance = TRUE,
             method = "rf")
```

### Results

```
rf1
```

```
## Random Forest
##
## 1000 samples
##    5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 1000, 1000, 1000, 1000, 1000, 1000, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##   2     33.31501  0.3462000  9.350178
##   3     35.23890  0.3139767  9.309661
##   5     38.78841  0.2589949  9.517779
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 2.
```

```
varImp(rf1)
```

```
## rf variable importance
##
##         Overall
## sg_putt  100.00
## sg_t2g    87.28
## sg_app    65.38
## sg_arg    64.01
## sg_ott     0.00
```

## Analysis

As we can see from the model's results, we received a mean absolute error of around 10. In the context of a golf tournament, this is not so bad as one stroke can sometimes be the difference between 12th place and 40th place.

What is interesting, is that the most important variable by far is Strokes Gained Putting. This tells me crucial information about what is most important for me to practice! While it may not be as glorious as going to the range and smashing my driver as far as possible, it is proven effective. It's advice that everyone hears all the time, but no one likes to do it. Now, obviously, I am not a PGA Tour professional, but for them to place the best, they have to score the best, and to place the best requires excellent putting.

# Player Model Implementation

Now let's take a look at player level data, aggregate across the entire 2017 season. There are quite a few more metrics here which could prove important. The two models used predict if they have won or not and the number of top ten finishes. The reason for separating the models these ways is to look at what indicates someone winning a given weekend, versus what indicates a more consistent player who places in the top ten more often. I opted to remove many of the key features of the data set as I felt they were unfairly indicative of the results we see in tournaments.

## Has Won

```
rf_has_won <- train(factor(won) ~ . -NUMBER_OF_TOP_Tens -NUMBER_OF_WINS -Player -POINTS
-POINTS_BEHIND_LEAD -GIR_RANK -AVG_SCORE -SG_PER_ROUND -MAKES_BOGEY. -BOGEYS_MADE -TOTAL
_STROKES -HOLES_PLAYED -TOTAL_ROUNDS -EVENTS_PLAYED -ROUNDS_PLAYED -MEASURED_ROUNDS -TOT
AL_DRIVES,
                  data = player,
                  importance = TRUE,
                  classification = TRUE,
                  method = "rf")

rf_confusion <- randomForest(factor(won) ~ . -NUMBER_OF_TOP_Tens -NUMBER_OF_WINS -Player
-POINTS -POINTS_BEHIND_LEAD -GIR_RANK -AVG_SCORE -SG_PER_ROUND -MAKES_BOGEY. -BOGEYS_MAD
E -TOTAL_STROKES -HOLES_PLAYED -TOTAL_ROUNDS -EVENTS_PLAYED -ROUNDS_PLAYED -MEASURED_ROU
NDS -TOTAL_DRIVES,
                  data = player,
                  importance = TRUE,
                  classification = TRUE)

t <- varImp(rf_has_won)
t <- t$importance
t <- t |> subset(select = -`0`)
colnames(t) <- "Has Won Importance"
t <- t |>
  arrange(-`Has Won Importance`) |>
  head(10)
```

## Top Tens

```
rf_toptens <- train(NUMBER_OF_TOP_Tens ~ . -NUMBER_OF_WINS -won -Player -POINTS -POINTS_
BEHIND_LEAD -GIR_RANK -AVG_SCORE -SG_PER_ROUND -MAKES_BOGEY. -BOGEYS_MADE -TOTAL_STROKES
-HOLES_PLAYED -TOTAL_ROUNDS -EVENTS_PLAYED -ROUNDS_PLAYED -MEASURED_ROUNDS -TOTAL_DRIVE
S,
                    data = player,
                    importance = TRUE,
                    method = "rf")

d <- varImp(rf_toptens)
d <- d$importance
colnames(d) <- "Top Tens Importance"
d <- d |>
  arrange(-`Top Tens Importance`) |>
  head(10)
```

## Results

```
rf_has_won
```

```
## Random Forest
##
## 194 samples
##  69 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 194, 194, 194, 194, 194, 194, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.8543304  0.1327213
##   27    0.8303138  0.1796275
##   52    0.8122194  0.1586154
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
rf_toptens
```

```
## Random Forest
##
## 194 samples
##  69 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 194, 194, 194, 194, 194, 194, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##    2    1.700990  0.4601371  1.322865
##   27    1.605388  0.4799848  1.232750
##   52    1.621688  0.4569432  1.244946
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 27.
```

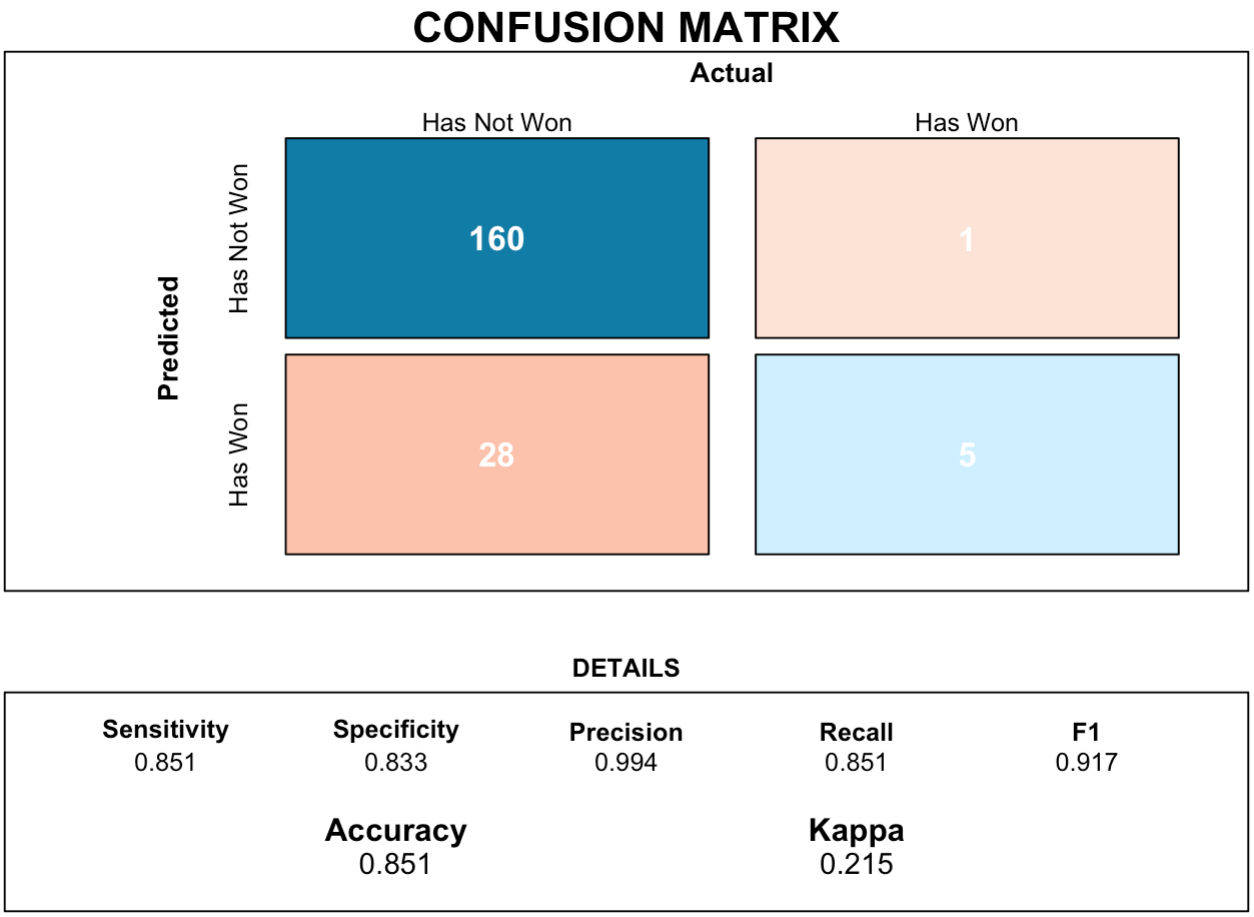
```
#| layout-ncol: 2
#| tbl-cap: "Tables"
#| tbl-subcap: ["A table", "Another table"]

kable(list(t, d))
```

| | Has Won Importance | | Top Tens Importance |
|---|---|---|---|
| SG.OTT | 100.00000 | SG.APR | 100.00000 |
| DRIVES_320.. | 65.96614 | SG.OTT | 54.70644 |
| AVG_Driving_DISTANCE | 63.03587 | NUMBER_OF_SAVES | 51.19082 |
| AVG_CARRY_DISTANCE | 61.27268 | UP_AND_DOWN_. | 49.90869 |
| PAR_OR_BETTER | 59.56676 | SHORTEST_CARRY_DISTANCE | 45.97618 |
| LONGEST_CARRY_DISTANCE | 58.90846 | POSSIBLE_FAIRWAYS | 45.54139 |
| GOING_FOR_GREEN_IN_2. | 56.74779 | TOTAL_DRIVES_FOR_320. | 44.78145 |
| TOTAL_O.U_PAR | 55.90327 | TOTAL_SG.PUTTING | 42.26637 |
| FASTEST_BALL_SPEED | 52.67159 | SG_PUTTING_PER_ROUND | 40.36505 |
| TOTAL_DRIVES_FOR_320. | 51.69456 | DRIVES_320.. | 37.82795 |

```
cm1 <- confusionMatrix(data = rf_confusion$confusion[,1:2], reference = player$won)

draw_confusion_matrix(cm1)
```



## Analysis

The performance of the models is okay. The Kappa value of the classification of if a player has won or not indicates some slight agreement among the predictions. Alternatively, the Mean Absolute Error of the number of top tens a player had that season is rather good, with it being able to predict within about one or two tournament placements.

What these two models reveal is that on Tour, among other factors, your ability to get off the tee well (SG.OTT) and hit the ball far (AVG_Driving_DISTANCE) are crucial. These two metrics go more or less hand in hand, as SG.OTT is in part calculated by how far you hit the ball with your driver.

It's interesting to note the difference between these models. A more consistent golfer that places in the top ten often is a player whose approach game (SG.APR) shines. Approach shots are typically between 80 and 200 yards out, and are a defining factor on a hole's score. Additionally, we see importance among variables like how well they get themselves out of tricky situations (NUMBER_OF_SAVES) and whether they can chip the ball close when just off the green (UP_AND_DOWN_.).

On the other hand, a player that has won on tour can be categorized by a riskier player. The model favors players that focus on distance. Players that drive the ball far (DRIVES_320..) and go for the green in tough situations (RTP.GOING_FOR_THE_GREEN) are players that have won a tournament.

## What can I take from these models?

That said, I am not a professional golfer, and I know for a fact I cannot hit the golf ball with my driver 320 yards, nor hit every fairway. While these things are important to consider, it's even more important to take into consideration that these are what help you win. On the PGA Tour. The most prestigious of golf leagues.

I am not quite trying to go out and win a PGA Tournament, I'm more interested in a few more pars and the occasional birdie, but this information is important to keep in mind regardless.

# SVM Model

Let's split our data into the players that have won, and the players that have not won on tour to see if we can make a prediction based on their season metrics. To do this, we will use a SVM model (Support Vector Machine). Although this won't necessarily aid me in improving my golf game because we are unable to extract variable importance from an SVM, it could be useful if you were to be looking at this data mid-season who has not won, but likely will. According to this data, about $\frac{1}{3}$ of golfers on the PGA tour won once or more in the 2017 season.

## How an SVM works

A Support Vector Machine works by first, plotting each player on a high dimensional graph, where each different axis represents one of the features within the data set that we have selected to use for our model.

The algorithm then attempts to split the data points into two different groups with a hyper plane; think of this as a line in 2-Dimensional graphs, or a plane in a 3-Dimensional graph. In this case, the hyper plane is splitting the data into golfers that have won on Tour in 2017, and golfers that have not.

The golfers that are closest to this hyper plane according to their statistics are called the Support Vectors, and the algorithm aims to maximize the distance between these closest golfers plotted points and the hyper plane. This is achieved through mathematical processes. Here, I used a linear kernel which uses a straight hyperplane (like a straight 2-D line) instead of a more complex function (like a hyperbolic 2-D line) to separate the two groups.

# SVM Implementation

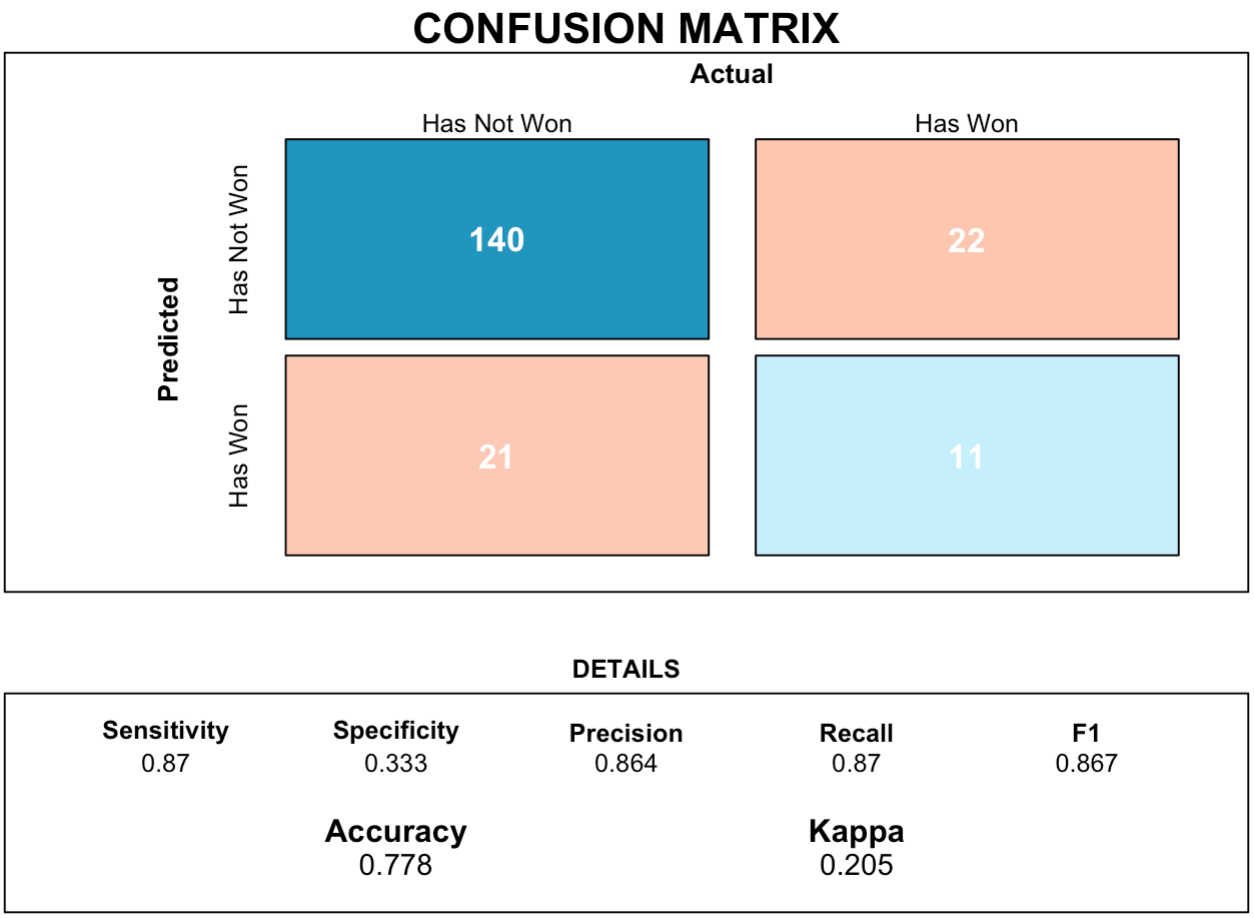
```
train_control<- trainControl(method = "cv", number = 5, savePredictions = TRUE)

svm1 <- train(factor(won) ~ . -Player -NUMBER_OF_WINS -NUMBER_OF_TOP_Tens -POINTS -POINT
S_BEHIND_LEAD -GIR_RANK -AVG_SCORE -SG_PER_ROUND -MAKES_BOGEY. -BOGEYS_MADE,
              data = player_scale,
              trControl = train_control,
              method = "svmLinear")

preds <- svm1$pred
cm <- confusionMatrix(data = preds$pred, reference = preds$obs)
```

# Results

```
draw_confusion_matrix(cm)
```




# Analysis

The final confusion matrix of this model shows us similar results to the Kappa value of the random forest above. Although boasting a solid accuracy of around 0.75, the Kappa value tells us there is slight to little agreement among the prediction variables and the prediction.

It makes sense that we cannot make great predictions about whether a player has won a tournament or not. Two players could play almost identically up to the very final putt, and one will always come out on top. One stroke separates so much in the professional scene.

Although these results are rather lacking in their ability to predict which golfers have won tournaments, they do tell me a lot about what's important in golf. On the surface it may appear to be a numbers game, but in reality it's so much more. The fact that we cannot accurately separate the good from the great based on a multitude of numbers reveals to me so much about what is important for success. It's beyond the raw statistics.

A couple key parts of a player's game that cannot be expressed in these raw statistics are 1. their mentality, and 2. their course management. A player's course management does display itself within some Strokes Gained numbers, but that is only half of the result, as execution is another factor. The fact that we cannot accurately predict if someone has won emphasizes the importance of these two components of the game. This aligns with what makes the best golfers the best. Players that remain level headed, remain consistent and players that are able to reset after a bad shot typically score better.

# Final Takeaways

Taking into account all of these results (and their contexts) tells me that I need to focus on putting, accuracy off the tee, and possibly most importantly mentality. While I am not a professional golfer, these are the stepping stones to get there.