Tennis Tournament Classification - J. Finelli, July 2021

Project Overview

In this project, we attempt to build machine learning-based classification models that can accurately identify the tournament and gender affiliations of point-by-point data from two separate major tennis tournaments: the US Open, and Wimbledon.

Below, we use **logistic regression**, **K-nearest neighbors**, **classification trees**, and **random forest** models in order to classify groups of points (games and sets) by tournament, by gender, and by both at once. In the end, our effort proves successful: our best models achieve over 95% accuracy in classifying data by tournament, over 80% accuracy by gender, and 70% accuracy when attempting to do both at once.

Tennis tournaments, venues, and playing styles

The professional tennis calendar is highlighted each year by the four "major" tournaments: the Australian Open in January, Roland Garros in May (the "French Open"), The Championships, Wimbledon in July ("Wimbledon"), and the United States Open in September (the "US Open").

These four major tournaments are defined not only by their historic venues, but especially by their unique playing surfaces, which in turn reward specific styles of play. For example, the French Open in Paris is played on a soft red clay which is famous for slowing down the ball as it bounces, creating long back-and-forth rallies that favor players in peak physical shape.

Wimbledon, contested annually at the All England Lawn Tennis & Croquet Club outside of London, is famous for its grass courts. Tennis played on grass is a unique experience: the ball bounces low and fast (it "skips" and "skids"), making the ball difficult to return and often not worth running far to chase. As a result, players often prefer tactics aimed at ending points quickly.

The US Open, held in Flushing Meadows (Queens) New York, is played on a traditional hard court surface which produces average conditions for ball speed: faster than the French Open, slower than Wimbledon.

At the same time, women's and men's professional tennis have evolved to favor slightly different styles of play as well. For example, the serve (by which a player initiates the point, fully within his/her control) has become a weapon particularly in the top echelons of the men's game.

The Match Charting Project

"Match charting" is an exercise which documents the characteristics of tennis points across several elements (serve speed, player positioning, rally length, etc.). This data can then be complied across various dimensions (by game, set, gender, tournament round etc.) in order to facilitate research.

Thanks to the Match Charting Project[1] and analyst Jeff Sackmann in particular, nearly a decade's worth of detailed point-by-point data from each of the four major tournaments has been compiled and is available for public consumption and research purposes on Sackmann's GitHub repository[2]. He requests acknowledgement, which we gladly give him here.

[1] http://www.tennisabstract.com/charting/meta.html
[2] https://github.com/JeffSackmann/tennis_slam_pointbypoint

Methods and Analysis

Available data fields

Match charting efforts give us access to a wide range of information about each point, including:

- Status fields, such as match ID, point winner, and game & set scores before and after the point
- Cumulative counts, such as total points, games, and sets won thus far by each player
- Action descriptions, such as serve speed, distance covered, and rally length (in shots and in time)
- Outcome binaries, such as net presence, aces, double faults, "winner" shots, and unforced errors

Not every variable is successfully charted for every point, and thus in this project we select data sets that were, on the whole, the most complete of those available to us.

Data preparation and cleaning:

We begin by accessing the data files for all points in singles play (for both men and women) from the US Open in 2017 and Wimbledon in 2021. GitHub URLs are identified in the attached code, but are not shown here for brevity and formatting. We choose these two specific tournament-year combinations because they contain the most complete information about the distance traveled by each player during each point, and we suspect this data will be useful.

```
#download raw data files
download.file(url_us17,"us17_points")
download.file(url_wm21,"wm21_points")

#create data frames
us17points <- read_csv("us17_points")
wm21points <- read_csv("wm21_points")
tennispoints <- rbind(us17points,wm21points)
```

The list of available fields is impressive, even though not all fields are populated.

```
##  [1] "match_id"           "ElapsedTime"        "SetNo"
##  [4] "P1GamesWon"         "P2GamesWon"         "SetWinner"
##  [7] "GameNo"             "GameWinner"         "PointNumber"
## [10] "PointWinner"        "PointServer"        "Speed_KMH"
## [13] "Rally"              "P1Score"            "P2Score"
## [16] "P1Momentum"         "P2Momentum"         "P1PointsWon"
## [19] "P2PointsWon"        "P1Ace"              "P2Ace"
## [22] "P1Winner"           "P2Winner"           "P1DoubleFault"
## [25] "P2DoubleFault"      "P1UnfErr"           "P2UnfErr"
## [28] "P1NetPoint"         "P2NetPoint"         "P1NetPointWon"
## [31] "P2NetPointWon"      "P1BreakPoint"       "P2BreakPoint"
## [34] "P1BreakPointWon"    "P2BreakPointWon"    "P1FirstSrvIn"
## [37] "P2FirstSrvIn"       "P1FirstSrvWon"      "P2FirstSrvWon"
## [40] "P1SecondSrvIn"      "P2SecondSrvIn"      "P1SecondSrvWon"
## [43] "P2SecondSrvWon"     "P1ForcedError"      "P2ForcedError"
## [46] "History"            "Speed_MPH"          "P1BreakPointMissed"
## [49] "P2BreakPointMissed" "ServeIndicator"     "Serve_Direction"
## [52] "Winner_FH"          "Winner_BH"          "ServingTo"
## [55] "P1TurningPoint"     "P2TurningPoint"     "ServeNumber"
## [58] "WinnerType"         "WinnerShotType"     "P1DistanceRun"
## [61] "P2DistanceRun"      "RallyCount"
```

Next, we use information embedded in the match_ID field to label each row as belonging to a specific tournament and a specific gender. We also build a tournament-gender combination variable so we can attempt to distinguish between both at once.

```
#create tournament column using text from match_id
  tennispoints <- tennispoints %>%
    mutate(Tournament=ifelse(str_sub(match_id,6,6)=="u","US17","WM21"))

  #create gender column using match_id (key = first digit of #)
  tennispoints <- tennispoints %>%
    mutate(Gender=ifelse(str_sub(match_id,-4,-4)==1,"M","W"))

  #create tournament_gender column using paste
  tennispoints <- tennispoints %>%
    mutate(TGComb=paste(Tournament, Gender, sep="_"))
```

We continue our data preparation by adding binary columns for certain point characteristics (ace, double fault, net presence, etc.) that combine the existing player-specific binaries. Because we will be looking at big-picture trends by tournament and by gender, there's no need to distinguish, for example, whether it was player A or player B who hit an unforced error in a given point.

```
  #create combined Y/N cols for add.l point features (ace, net pt, etc.)
  tennispoints <- tennispoints %>%
    mutate(UfeYN=P1UnfErr+P2UnfErr,
           NetYN=P1NetPoint+P2NetPoint,
           WnrYN=P1Winner+P2Winner,
           AceYN=P1Ace+P2Ace,
           DfYN=P1DoubleFault+P2DoubleFault,
           BrkYN=P1BreakPointWon+P2BreakPointWon,
           CombDist=P1DistanceRun+P2DistanceRun)
```

The remaining data cleaning work involves removing a small number of blank rows, substituting averages where certain fields (such as distance covered) are unexpectedly zero, and converting from metric units in one case. We also use existing point-, game-, and set-winner information to determine and assign critical PointID, GameID, and SetID values for each row respectively. Full code is available separately.

Having cleaned the table of point-by-point data, we can now use `group_by` and `summarize` to generate summary tables organized by game (best of 7 points) and by set (first to six games). These two tables will ultimately fuel the bulk of our analysis. For brevity, we show code only for the by-game table here.

```
#create core data table by game
  tpbygame <- tennispoints %>% group_by(GameID) %>%
    summarize(
      Tournament=first(Tournament),
      Gender=first(Gender),
      TGComb=paste(Tournament, Gender, sep="_"),
      Set=first(SetID),
      Points=n(),
      Margin=ifelse(Points<4,4,ifelse(Points==4,4,ifelse(Points==5,3,2))),
      AvgRC=sum(RallyCount)/Points, AvgCDist=sum(Comb2Dist)/Points,
      AcePct=sum(AceYN)/Points,
      WnrPct=sum(WnrYN)/Points,
      DfPct=sum(DfYN)/Points,
      UfePct=sum(UfeYN)/Points,
      NetPct=sum(NetYN)/Points,
      BrkYN=sum(BrkYN))
```

Importance of aggregation

With the creation of these two new tables, our statistics are now aggregated by game and by set, which gives us a better chance of classifying the points in question. For example, knowing that a single point resulted in an "ace" (when the receiver is unable to make any contact whatsoever with a serve) might not help us classify it as belonging to a women's or men's point, as there are plenty of aces in both women's and men's play. However, knowing that a game featured, for instance, 4 aces out of its 5 points played might be a bit more informative. Examining this type of data at the set level will be significantly more informative still.

We are also able to introduce a few new variable columns to our tables, such as the winning margin for each game, and whether a game qualified as a "break" (where the server does not win the game). We look forward to testing the importance of these variables.

Data partitioning

Our last data preparatory step is to partition each of our core aggregation tables for model training and testing purposes.

We begin by setting aside 10% of each table as a `validation` test set, against which we will confirm the results of our best models at the end of the process. We name the remaining large portion of each table as `*_trim`, which we will use for model training before final validation.

```
#set seed
set.seed(7, sample.kind="Rounding")

#for "by game" tables
test_index <- createDataPartition(y = tpbygame$TGComb,
                                  times = 1, p = 0.1,
                                  list = FALSE)
tpbg_validation <- tpbygame[test_index,]
tpbg_trim <- tpbygame[-test_index,]

#for "by set" tables
test_index <- createDataPartition(y = tpbyset$TGComb,
                                  times = 1, p = 0.1,
                                  list = FALSE)
tpbs_validation <- tpbyset[test_index,]
tpbs_trim <- tpbyset[-test_index,]
```

Finally, we separate each `trim` table into traditional `train` and `test` sets, reserving 90% of each `trim` set for training to give us the best chance for successful classification.

```
#create train and test sets from the trim set by game
set.seed(5, sample.kind="Rounding")
test_index <- createDataPartition(y = tpbg_trim$TGComb,
                                  times = 1, p = 0.1,
                                  list = FALSE)
tpbg_test <- tpbg_trim[test_index,]
tpbg_train <- tpbg_trim[-test_index,]

#create train and test sets from the trim set
test_index <- createDataPartition(y = tpbs_trim$TGComb,
                                  times = 1, p = 0.1,
                                  list = FALSE)
tpbs_test <- tpbs_trim[test_index,]
tpbs_train <- tpbs_trim[-test_index,]
```

Data exploration and visualization

With our data now prepared for analysis, we can look to see which point attributes, when aggregated, will best help us classify data by tournament, by gender, and by both at once.

We first look broadly at the simple averages of each predictor when data is grouped by tournament. Right away, we see that the mean rally count (i.e. length of rally) was 25% higher at the US Open (just over 4 shots / point) vs. at Wimbledon (under 3), and that players ran ~30% further per point at the US Open as well. These predictors might seem to be highly correlated by definition, but we will discover they are not necessarily.

```
#simple table by tournament
  TournaTable <- tennispoints %>%
  group_by(Tournament) %>%
  summarize(AvgRC=sum(RallyCount)/n(),
            AvgCDist=sum(Comb2Dist)/n(),
            AcePct=sum(AceYN)/n(),
            WnrPct=sum(WnrYN)/n(),
            DfPct=sum(DfYN)/n(),
            UfePct=sum(UfeYN)/n(),
            NetPct=sum(NetYN)/n())
  TournaTable %>% knitr::kable()
```

Tournament	AvgRC	AvgCDist	AcePct	WnrPct	DfPct	UfePct	NetPct
US17	4.08	21.0	0.063	0.314	0.045	0.355	0.176
WM21	2.93	15.8	0.070	0.316	0.042	0.302	0.201

When this data is grouped by gender instead, we see that distinctions appear in a new set of predictors. For example, we see that aces are nearly twice as common in the men's game (8.4%) vs. the women's game (4.2%), and that men play at the net about 25% more frequently than women do (21% vs. 16%). These findings are credible intuitively, and they give us additional confidence in the data.

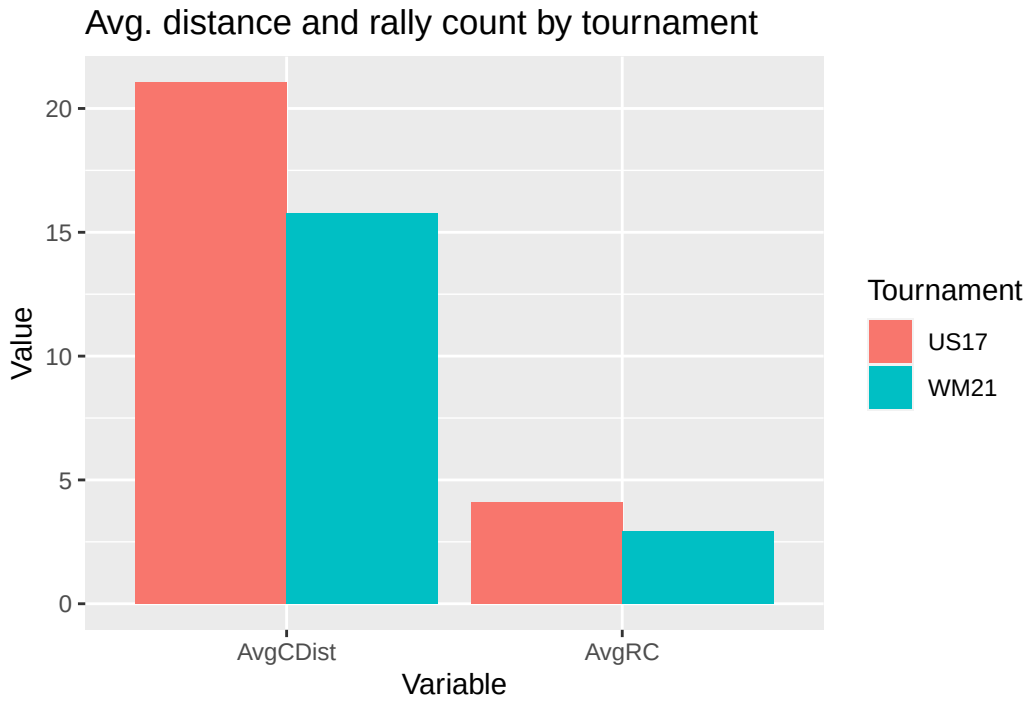
Gender	AvgRC	AvgCDist	AcePct	WnrPct	DfPct	UfePct	NetPct
M	3.3	17.6	0.084	0.328	0.037	0.300	0.212
W	3.5	18.2	0.043	0.297	0.052	0.357	0.161

We also examine a table grouped by the combined "tg_comb" variable, showing us all four categories of matches at once. We see some of the same trends, albeit with finer distinctions, though the net attack rate of 22.6% for men at Wimbledon stands out.

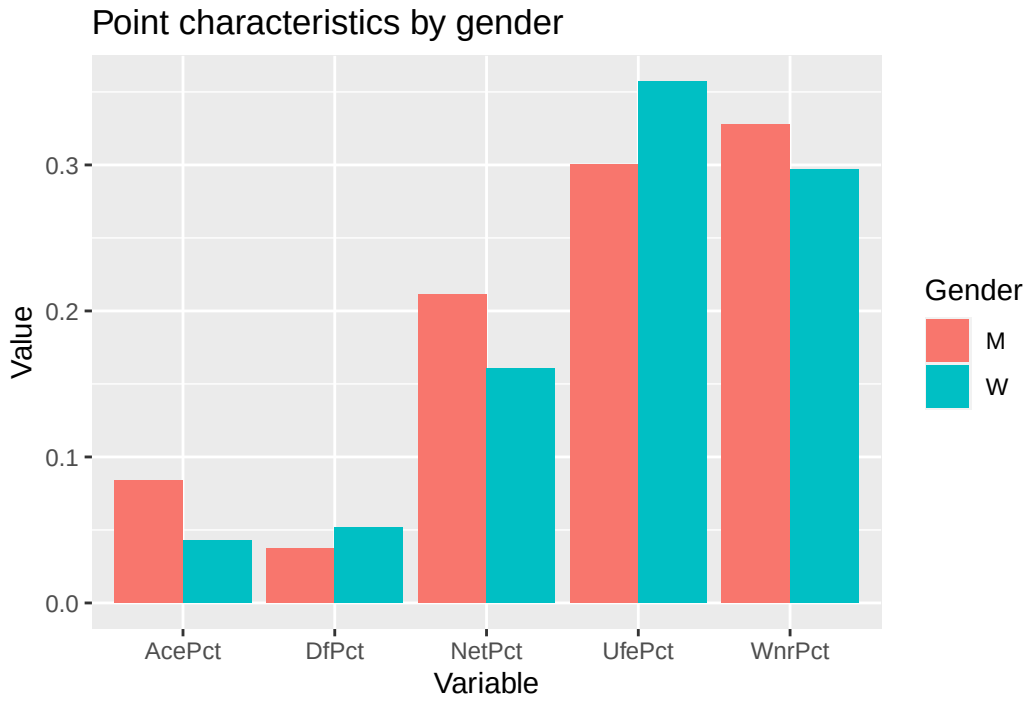
Also noticeable is that some metrics clearly group by tournament (e.g. rally count, distance), while others clearly group by gender (e.g. ace rate). We suspect these contrasting groupings will allow us to predict both tournament and gender together.

TGComb	AvgRC	AvgCDist	AcePct	WnrPct	DfPct	UfePct	NetPct
US17_M	3.94	20.9	0.082	0.327	0.041	0.331	0.187
US17_W	4.28	21.3	0.038	0.297	0.050	0.387	0.161
WM21_M	2.93	15.6	0.085	0.328	0.035	0.282	0.226
WM21_W	2.94	16.0	0.046	0.298	0.053	0.335	0.161

Using bar charts, we can see a few of these distinctions more clearly. Specifically, we see the tournament-based gap in average rally count and distance covered:

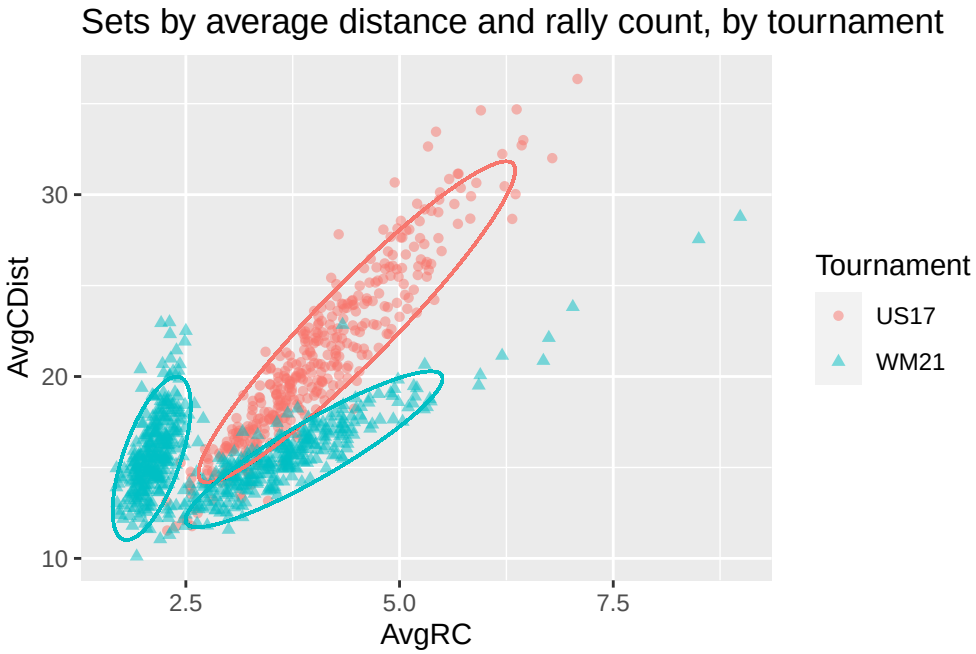


and, we see the gender-based gap in ace %, net attack rate, and unforced error %.



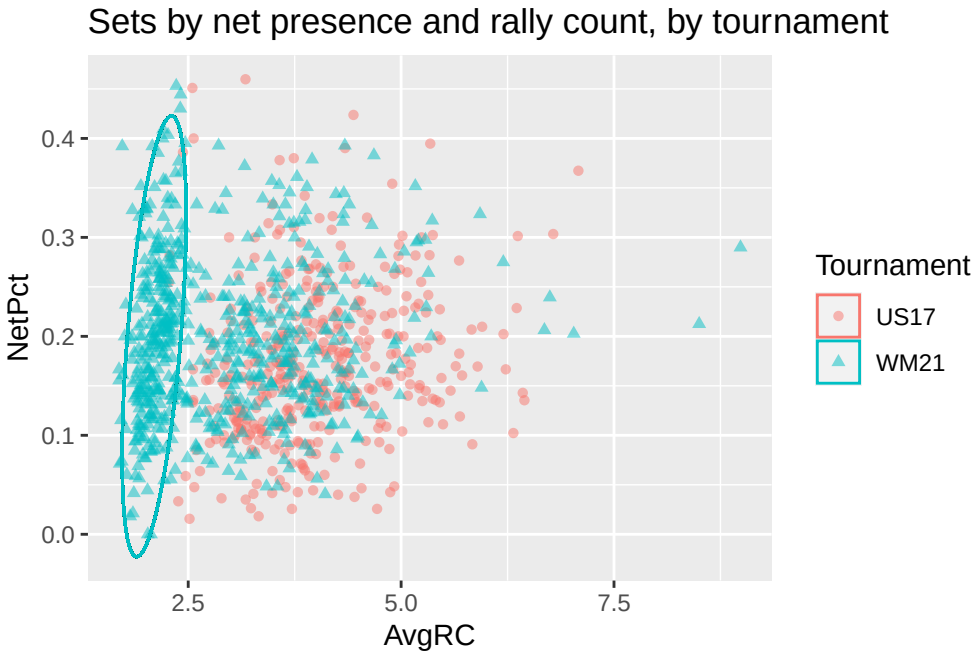
Tournament-based distinctions

We next try to articulate better the tournament-based gaps in rally count and distance covered. The graph below (data grouped by set) helps us visualize them together.



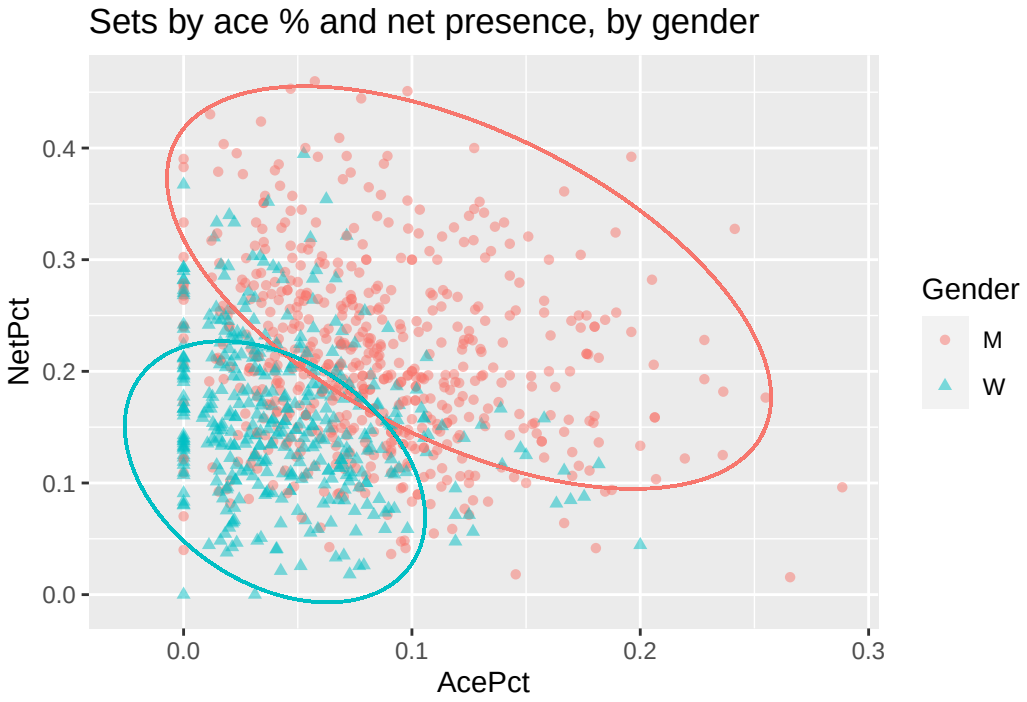
Here, we notice that sets at Wimbledon (circled in green) fall into two visually distinct categories: those with low average rally totals but higher per-stroke distances covered, and those with longer rallies but lower per-stroke distances. This could be the result of the "serve-and-volley" style of play, popular at Wimbledon, in which the server runs up to the net right after serving, covering distance but shortening rally count.

Sure enough, when plotting rally count and net attack rate together, we can see that this serve-and-volley pattern is indeed popular at Wimbledon and stands out visually. This will likely help us as we classify.



Gender-based distinctions

We see a similarly helpful result when we explore gender-based distinctions in plotting sets by ace and net attack rates. As is visually clear, the sets with the highest net attack and ace rates (and especially both together, circled in red below) are almost exclusively men's sets.



Modeling approach

With these distinctions having been observed and visualized, we are optimistic that machine learning models, especially when applied to data aggregated at the set level, will be able to classify data by tournament and by gender with at least some success.

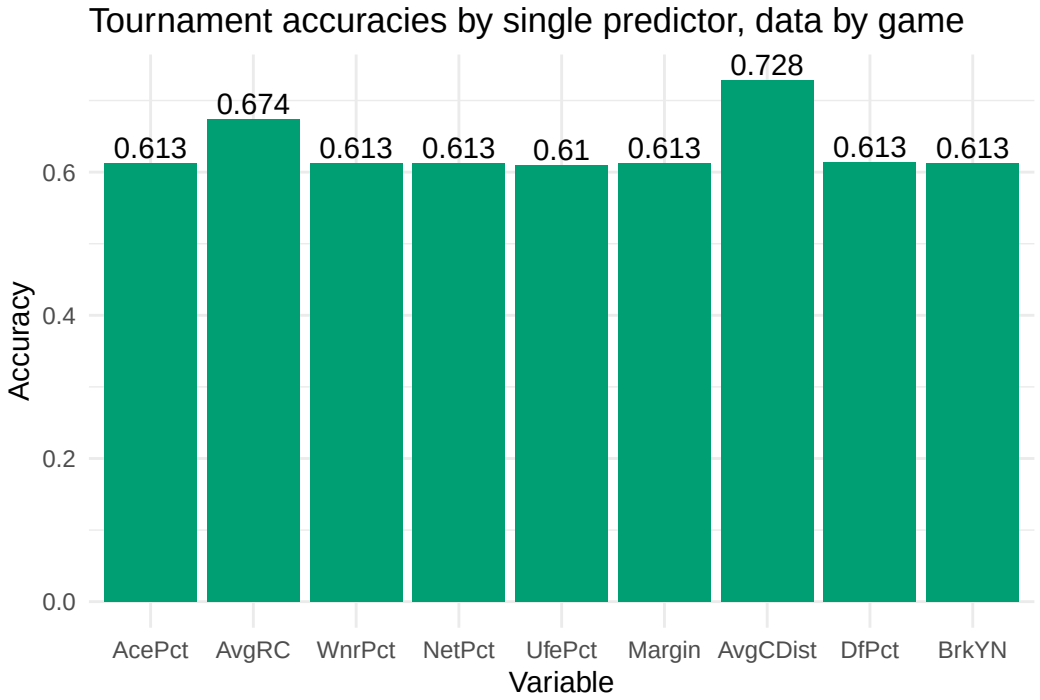
Throughout our modeling, we will measure success by the **overall accuracy** metric. Classification model results contain many statistics as we know, and while we will certainly be interested, for example, in sensitivity and specificity measurements, overall accuracy will be our standard.

As a starting point, and in order to establish a baseline result for future context, we build guessing models that are successful only by random chance.

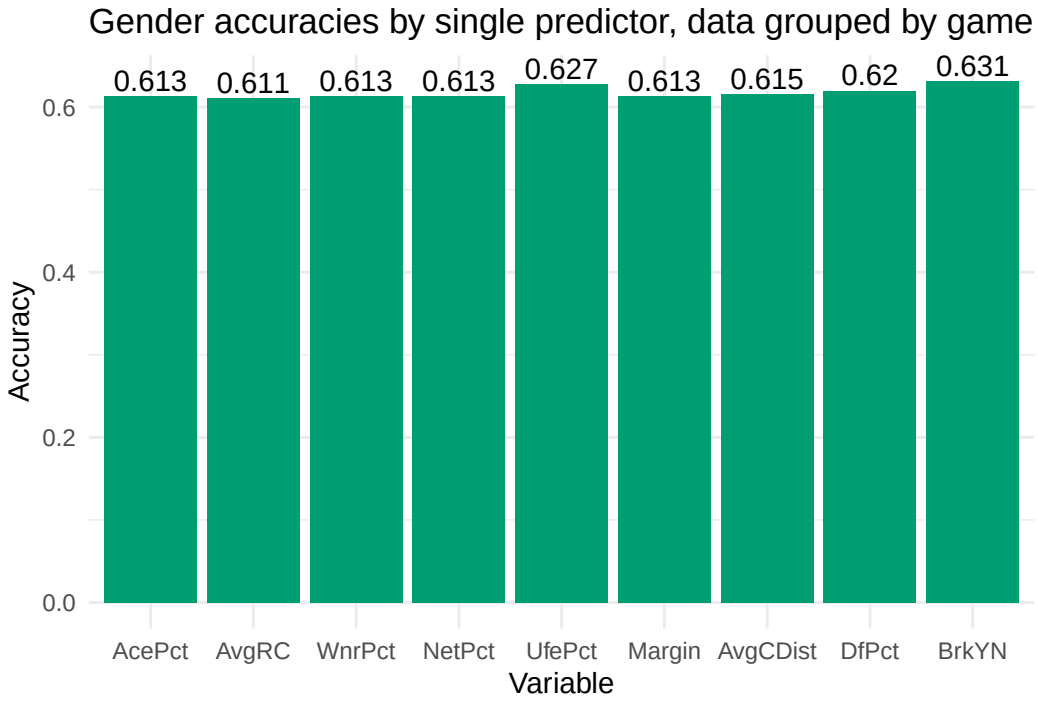
Next, we build logistic regression models which will use the most powerful predictors of the many available. To confirm which variables we should prioritize, however, we first use `sapply` to examine the success of logistic regression on each single predictor by itself. Here are the results when using data aggregated by game for both tournament and gender predictions. Code displayed once for convenience.

```
#sapply w/ tournament by game
  predictors <- c("AcePct","AvgRC","WnrPct","NetPct","UfePct",
                  "Margin","AvgCDist","DfPct","BrkYN")
  tpbg_train <- tpbg_train %>% mutate(t = as.numeric(Tournament == "US17"))
  tpbg_test$t <- factor(tpbg_test$Tournament)
  accs <- sapply(predictors, function(p){
    form <- reformulate(p, response = "t")
    fit_glm <- glm(form,
                   data=tpbg_train,
```

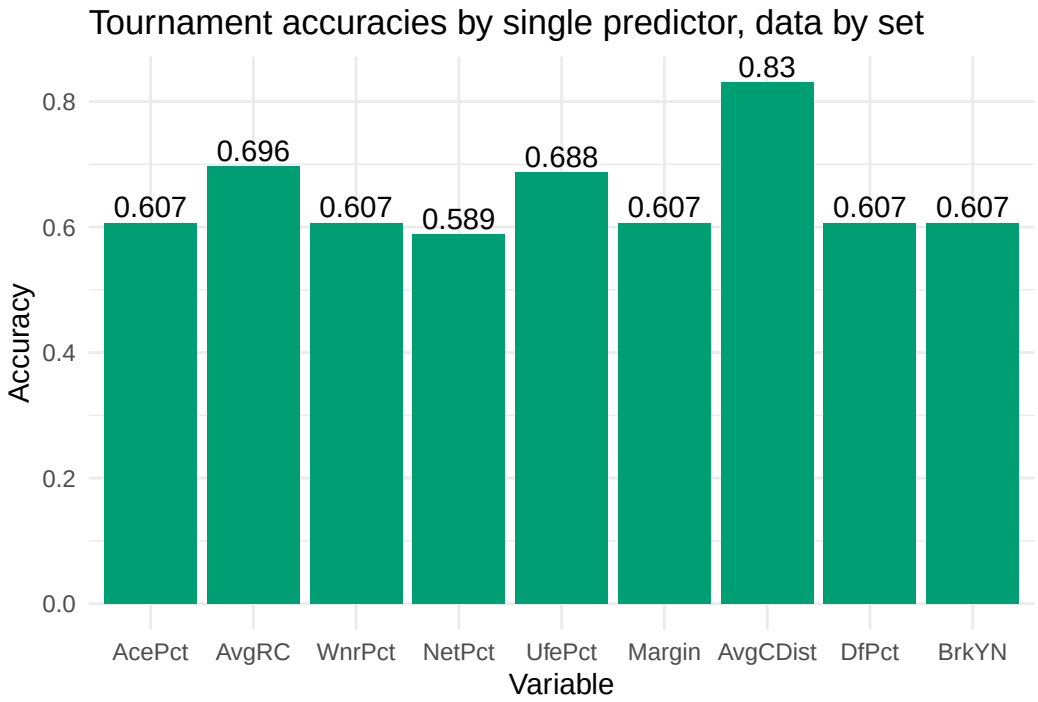
```
                  family = "binomial")
  p_hat_glm <- predict(fit_glm, tpbg_test, type="response")
  y_hat_glm <- factor(ifelse(p_hat_glm > 0.5, "US17", "WM21"))

  return(confusionMatrix(y_hat_glm, tpbg_test$t)$overall["Accuracy"])
})
accs <- stack(accs)[2:1]
colnames(accs) <- c("Variable","Accuracy")
rnl <- levels(accs$Variable)
levels(accs$Variable) <- substr(rnl,1,nchar(rnl)-9)
accs %>%
  ggplot(aes(Variable, Accuracy)) +
  geom_bar(stat="identity", fill="#009E73")+
  ggtitle("Tournament accuracies by single predictor, data by game") +
  geom_text(aes(label=round(Accuracy,3)), position=position_dodge(width=0.9),
            vjust=-0.25) +
  theme_minimal()
```

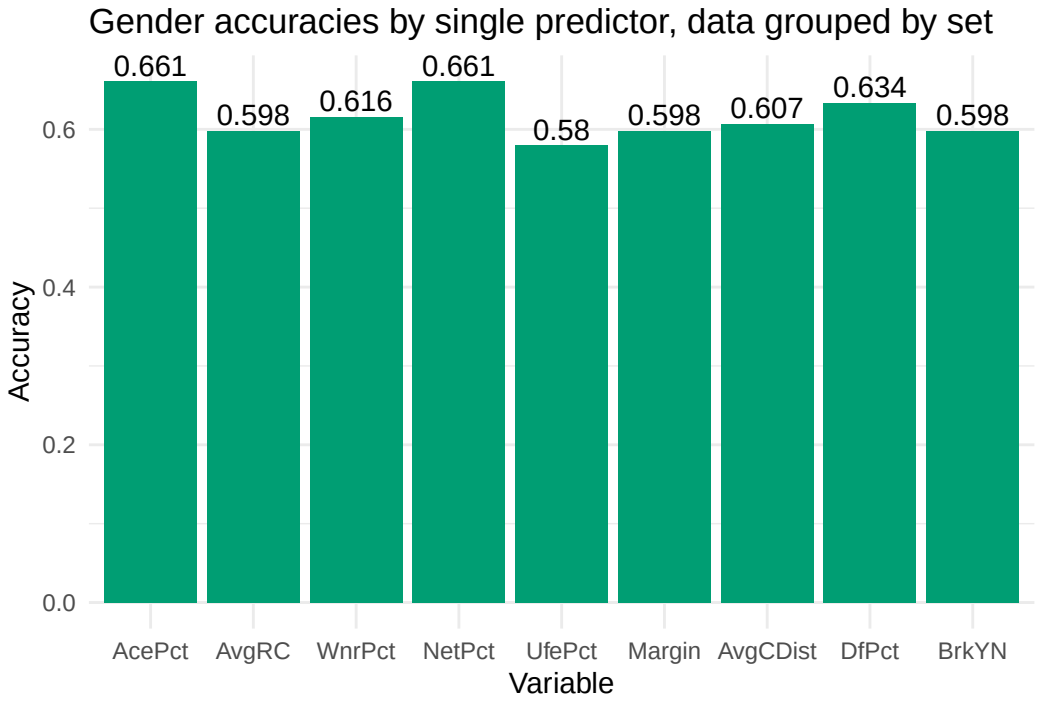


Consistent with earlier findings, we see that for tournament-based classification, rally count and distance covered both move the needle. We next repeat the same exercise for gender distinction, and we find that distinctions are hard to notice, though we see them a bit in unforced error rate and whether the player serving won the game in question.



Hoping to see more clear results, we next examine these same charts when data is aggregated by set rather than by game. As we would expect, the predictive ability of single variables become both stronger and more readily distinguished from one-another.





These last two charts in particular help confirm that rally count and distance are most useful when classifying by tournament, and that ace and net attack percents are most helpful when classifying by gender. We will use these findings not only when selecting variables for our logistic regression models, but potentially when using other approaches as well.

Moving beyond logistic regression, we next develop k-nearest neighbor models to see if they can more successfully digest and interpret our several predictors. We are particularly interested to see if knn models can do a good job of classifying data by tournament and gender together (i.e. our "tgcomb" category).

Finally, we explore the power of classification trees and their extension, random forests, as we round out our modeling exercise. Since we can already identify the handful of predictors likely to be most important to classification, we believe that both trees and forests modeling is achievable with available computing power.

Results

Guessing

Our first modeling effort is classification by guessing, i.e. using random chance. We begin by guessing tournament affiliation when data is grouped by game. We use the `sample` function and compare results against the tournament classification in the `tpbg_test` set. Not surprisingly, our accuracy hovers near 50% for a two-class outcome.

```
#guess tournament by game
  y_hat_guess <- factor(sample(c("US17","WM21"),nrow(tpbg_test),replace=TRUE))
  #check results
  guess_cm <- confusionMatrix(y_hat_guess,factor(tpbg_test$Tournament))
  tgg <- round(as.numeric(guess_cm$overall["Accuracy"]),2)
  tgg
```

```
## [1] 0.51
```

We proceed to guess outcomes for gender and tournament/gender combination, and for completeness we evaluate guessing models for our three "by-set" test sets as well. We are not surprised when those three

"by-set" accuracy rates diverge a bit from 50%, as the "by set" data sets contain roughly 1/10th as many rows as their "by-game" counterpart test sets. Also, as expected, the accuracy result for the four-class outcome of our combined tournament/gender variable hovers near 25%.

We create a table to hold our results, and we populate the first row with the results of our guessing models.

```
#create a data frame to keep track of all results, add first result to it
  acc_results <- data.frame(Method = character(),
                            TmtByG = character(),
                            GenByG = character(),
                            TgByG = character(),
                            TmtByS = character(),
                            GenByS = character(),
                            TgByS = character())
  acc_guess <- c("Guess",tgg, ggg, tggg, tsg, gsg, tgsg)
  acc_results[nrow(acc_results) + 1,] = acc_guess
  acc_results %>% knitr::kable()
```

Method	TmtByG	GenByG	TgByG	TmtByS	GenByS	TgByS
Guess	0.51	0.49	0.24	0.55	0.46	0.22

Result table column key:

TmtByG: Classifying *tournament*, predictor data grouped by *game*
GenByG: Gender, by game
TgByG: Combined tournament/gender, by game
TmtByS: Tournament, data by set
GenByS: Gender, by set
TgByS: Combined tournament/gender, by set

Logistic regression

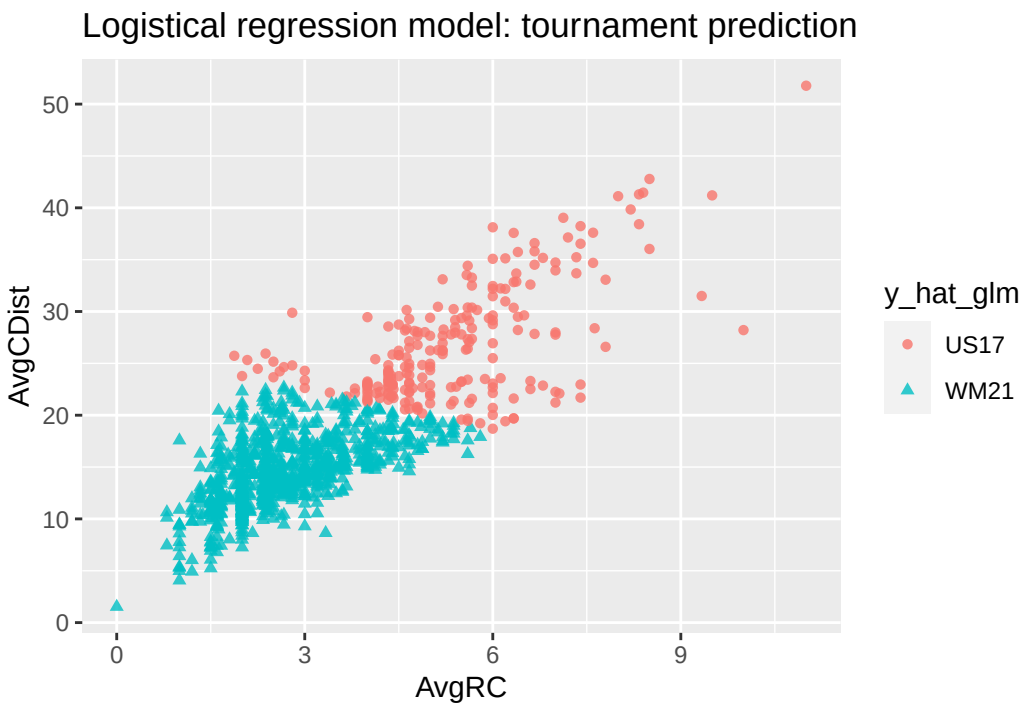
Earlier, we evaluated logistic regression models using each predictor one at a time (bar graphs above). Per that analysis, we will use `AvgRC` (rally count) and `AvgCDist` (distance) as our logistic regression predictors to classify by tournament. When we do so, our accuracy jumps to nearly 75%.

```
#logistic regression of tournament by game on top two variables
  #assign 1 if tournament is US17
  tpbg_train <- tpbg_train %>% mutate(t = as.numeric(Tournament == "US17"))
  tpbg_test$t <- factor(tpbg_test$Tournament)
  #fit logistic model on two predictors to start
  fit_glm <- glm(t ~ AvgCDist + AvgRC,
                 data=tpbg_train,
                 family = "binomial")
  #generate predictions, #assign binary outcome based on %
  p_hat_glm <- predict(fit_glm, tpbg_test, type="response")
  y_hat_glm <- factor(ifelse(p_hat_glm > 0.5, "US17", "WM21"))
  #check results
  tglr <-
  as.numeric(round(confusionMatrix(y_hat_glm, tpbg_test$t)$overall["Accuracy"],2))
  tglr
```

```
## [1] 0.74
```

Looking closer, we can examine how this first model classified each game's worth of points, plotted on axes reprenting the two variables it used. Compare this plot to the first scatter plot shown above (top of page 7).

```
#show model guess
  tpbg_test %>% ggplot(aes(AvgRC, AvgCDist, color = y_hat_glm, shape = y_hat_glm)) +
    geom_point(alpha = .8) +
    ggtitle("Logistical regression model: tournament prediction")
```



Above, we see the anticipated linear border between those games it classified as being from the US Open (longer rallies, greater distances covered), and those it assigned to Wimbldeon. It will be interesting to see if other models can improve upon this somewhat crude linear divide.

Next, for classifying by gender, we use `AcePct` and `NetPct` predictors, having seen that that ace percentages and net play rates were the most important predictors along the gender axis. The resulting accuracy rate of 64% is likewise a meaningful improvement on guessing.

Further, applying these same models to input data is grouped by set instead, we again see notable accuracy gains vs. their performance in classifying data grouped by game (to 80% and 77% respectively). This phenomenon is becoming less and less of a surprise.

Method	TmtByG	GenByG	TgByG	TmtByS	GenByS	TgByS
Guess	0.51	0.49	0.24	0.55	0.46	0.22
LogReg	0.74	0.64	-	0.8	0.77	-

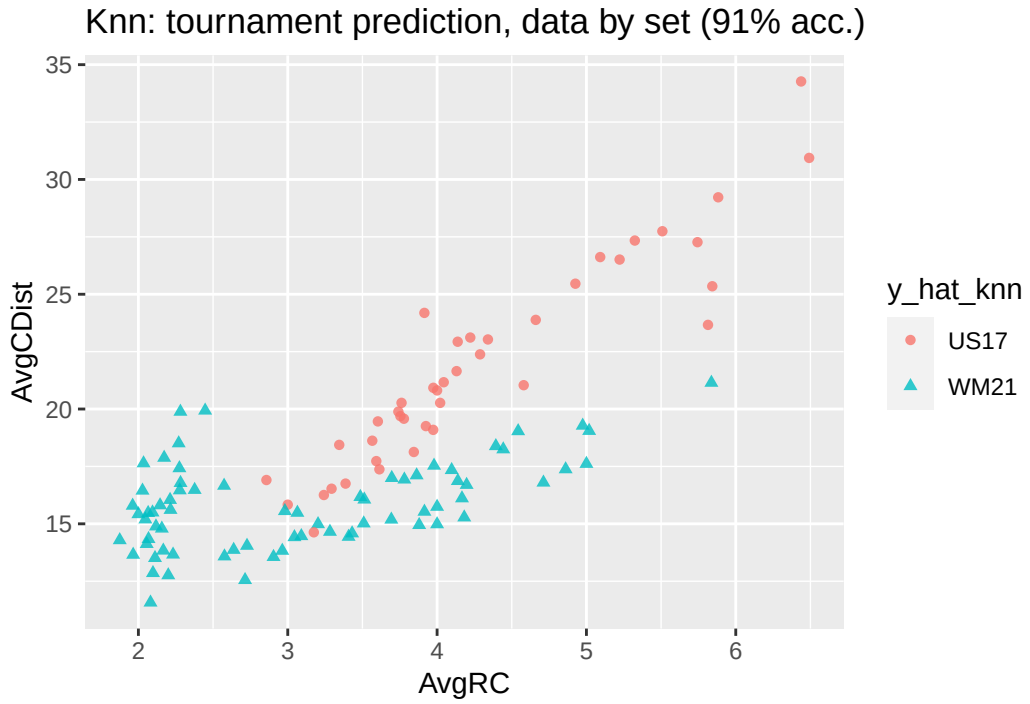
Finally, we note that we do not attempt to use logistic regression to classify data by the four-outcome combined tournament/game variable, as logistic regression only works to predict binary outcomes.

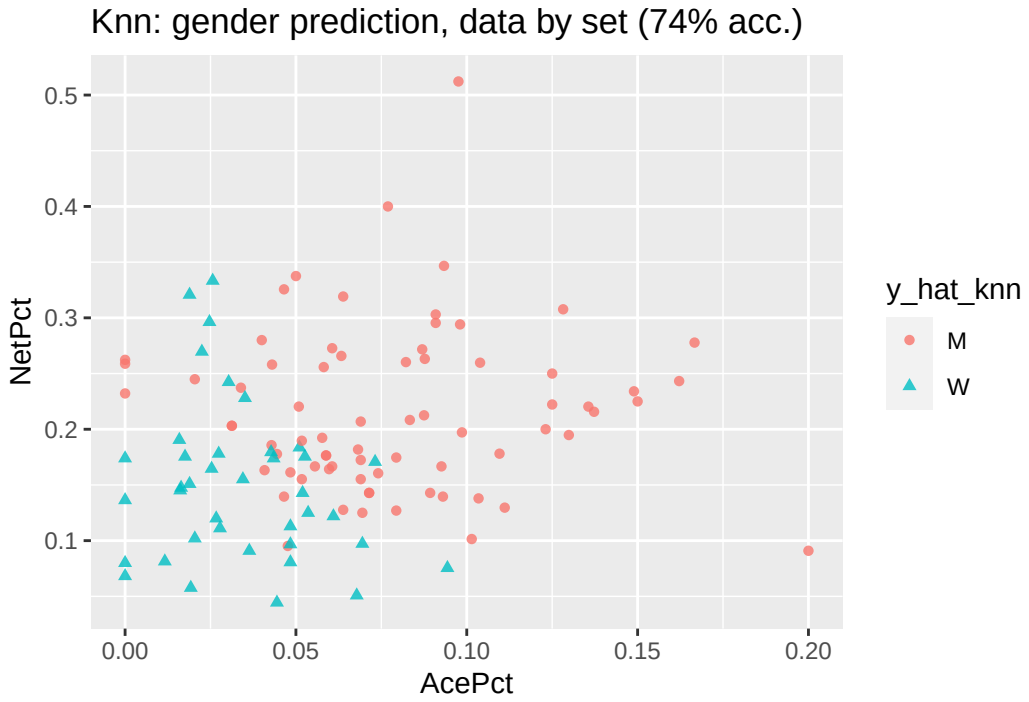
K-nearest neighbor models

We build k-nearest neighbor models to classify our data in a more nuanced, non-linear fashion, hopefully producing greater accuracy. We also hope that knn models can succeed in classifying data by tournament-gender combination.

The results appear to bear out our hopes, especially when we examine the plots (below) produced from prediction from data when grouped by set. Note: for brevity, only code for tournament classification is shown.

```
# knn tournament by set
  # cols 8 and 9 are still two best predictors: AvgRC and AvgCDist
  x <- tpbs_train[,8:9]
  y <- factor(tpbs_train$Tournament)
  knn_fit <- knn3(x, y)
  y_hat_knn <- predict(knn_fit, tpbs_test[,8:9], type = "class")
  tskn <- confusionMatrix(data = y_hat_knn, reference = (tpbs_test$t))$overall["Accuracy"]
  #pick best k
  ks <- seq(5, 100, 5)
  x <- tpbs_train[,8:9]
  y <- factor(tpbs_train$Tournament)
  accuracy <- map_df(ks, function(k){
    knn_fit <- knn3(x, y, k = k)
    y_hat_knn <- predict(knn_fit, tpbs_test[,8:9], type = "class")
    cm_test <- confusionMatrix(y_hat_knn, reference = (tpbs_test$t))
    test_error <- cm_test$overall["Accuracy"]
    tibble(test = test_error)
  })
  tskn <- round(max(accuracy$test),2)
  #graph prediction
  tpbs_test %>% ggplot(aes(AvgRC, AvgCDist, color = y_hat_knn, shape = y_hat_knn)) +
    geom_point(alpha = .8) +
    ggtitle("Knn: tournament prediction, data by set (91% acc.)")
```





We can see that both sets of predictions, plotted above, now have more nuanced borders than those generated by the logistic regression model. Accuracy for by-set tournament data jumps from 80% to over 90%.

Knn models also show some promise in classifying data by tournament-gender combination. Here, we use our four most potent predictors when building a knn model to classify by tournament and gender together. The resulting 62% accuracy rate is clearly superior to our baseline guess accuracy of 25%.

```
# knn for tg_comb by set
  # Use AvgRC, AvgCDist, AcePct, NetPct
  x <- tpbs_train[,c(8,9,10,14)]
  y <- factor(tpbs_train$TGComb)
  knn_fit <- knn3(x, y)
  y_hat_knn <- predict(knn_fit, tpbs_test[,c(8,9,10,14)], type = "class")
  #pick best k
  ks <- seq(5, 100, 5)
  x <- tpbs_train[,c(8,9,10,14)]
  y <- factor(tpbs_train$TGComb)
  accuracy <- map_df(ks, function(k){
    knn_fit <- knn3(x, y, k = k)
    y_hat_knn <- predict(knn_fit, tpbs_test[,c(8,9,10,14)], type = "class")
    cm_test <- confusionMatrix(y_hat_knn, reference = factor(tpbs_test$TGComb))
    test_error <- cm_test$overall["Accuracy"]
    tibble(test = test_error)
  })
  tgskn <- round(max(accuracy$test),2)
  tgskn
```

```
## [1] 0.62
```

Last, while knn models focused on data grouped by game out-perform their logistic regression counterparts, grouping data by set again produces even better accuracy rates across the board. As such, we will focus on using only grouped-by-set data going forward. Full results from knn modeling are below.

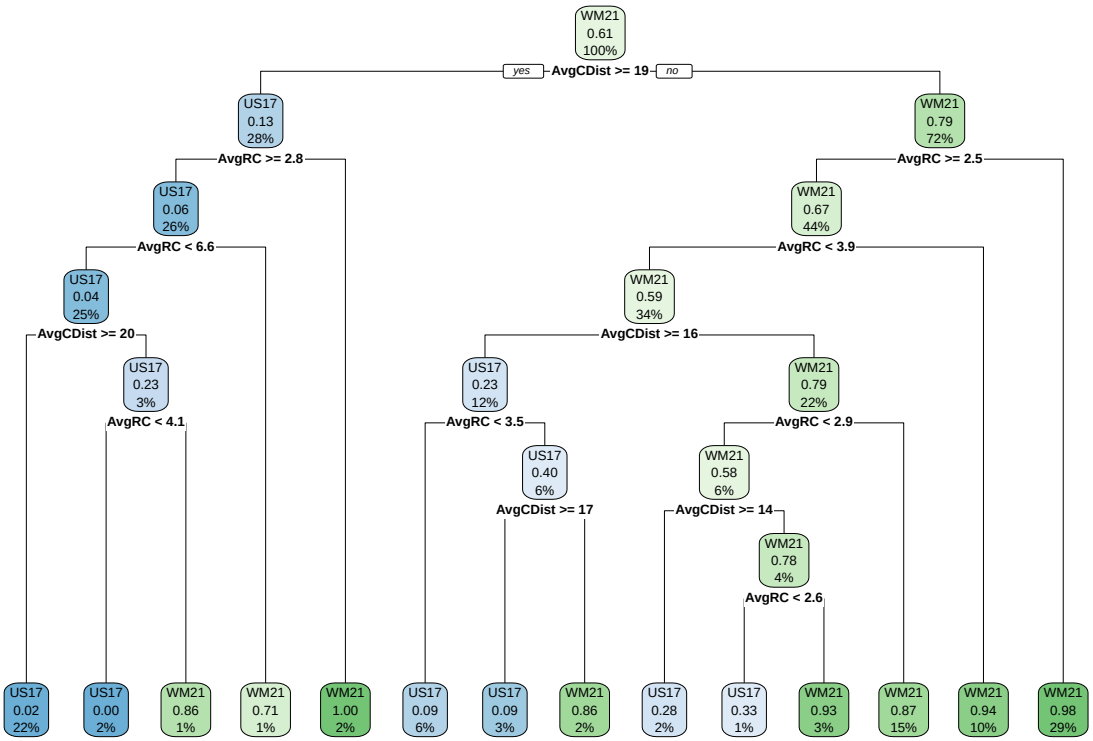
Method	TmtByG	GenByG	TgByG	TmtByS	GenByS	TgByS
Guess	0.51	0.49	0.24	0.55	0.46	0.22
LogReg	0.74	0.64	-	0.8	0.77	-
Knn	0.79	0.61	0.5	0.94	0.73	0.62

Recursive partitioning

Next, we examine classification trees built by recursive partitioning models. These trees have the advantage of being interpretable, so we will be able to verify that the results are credible. As mentioned above, we now use our grouped-by-set data only, as it has consistently yielded higher accuracy rates.

Our first classification tree aims to predict tournament affiliation, and our model produces a 90% accuracy rate. From the resultant tree diagram below, we see that the model predicts Wimbledon with high accuracy when the distances traveled are short, and if not, when the rally counts are short (right side of each branch). These conditions align with the fast bounces and serve-and-volley play common on grass.

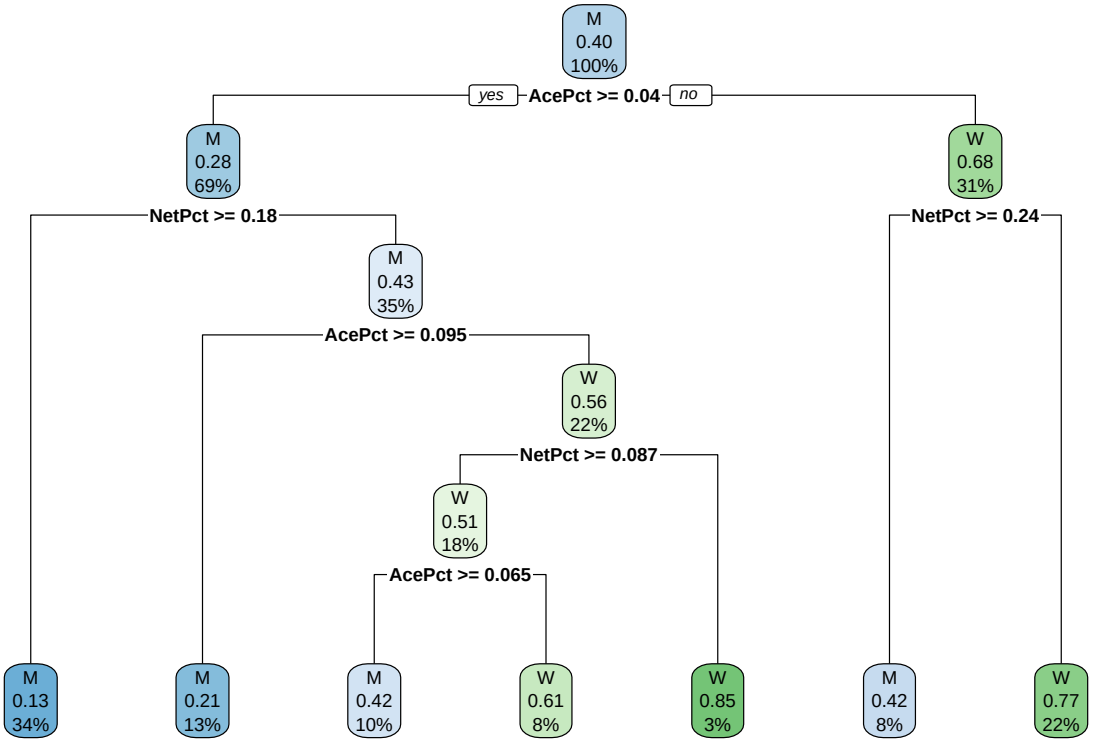
```
#classification tree tournament by set, with plot
  y <- tpbs_train$Tournament
  train_rpart <- train(Tournament ~ AvgRC + AvgCDist,
                       method = "rpart",
                       tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
                       data=tpbs_train)
  cpbest <- train_rpart$bestTune
  #once tuned, time for rpart with graphs
  fit_rpart <- rpart(Tournament ~ AvgRC + AvgCDist,
                     data=tpbs_train,
                     cp=cpbest,
                     method = "class")
  rpart.plot(fit_rpart)
```



```
y_hat <- predict(fit_rpart, tpbs_test, type="class")
tsct <- confusionMatrix(factor(y_hat), factor(tpbs_test$Tournament))$overall["Accuracy"]
tsct <- round(as.numeric(tsct),2)
tsct
```

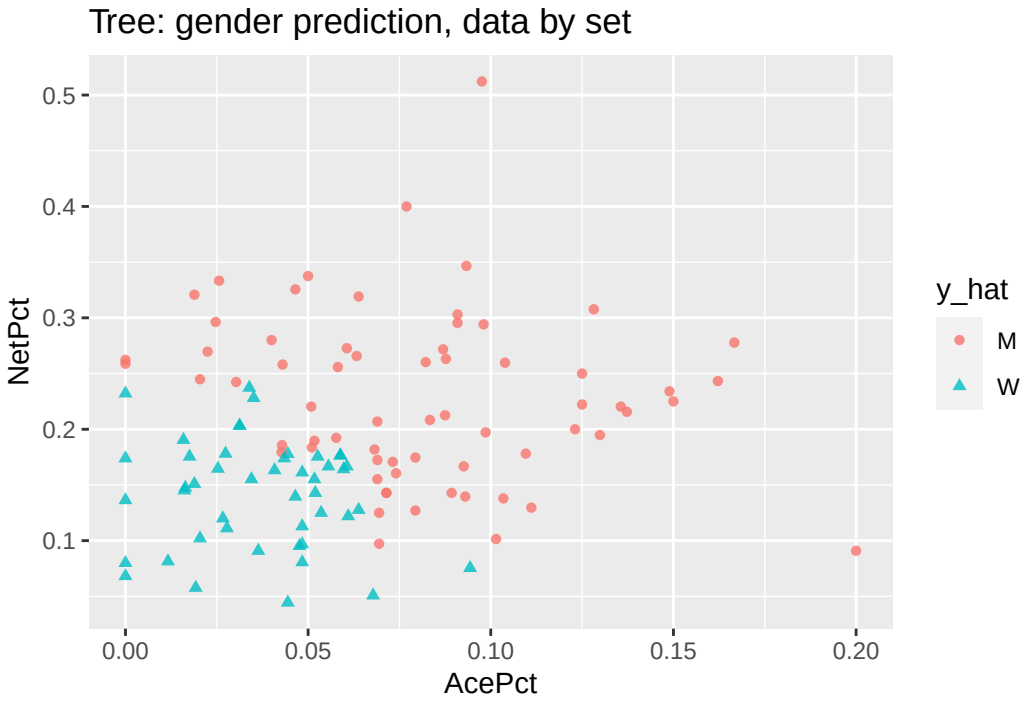
```
## [1] 0.9
```

The classification tree generated for predicting gender is also quite helpful: we clearly see branches leading to predictions of women's data whenever ace percents and net play percents are relatively lower (right side of each branch).

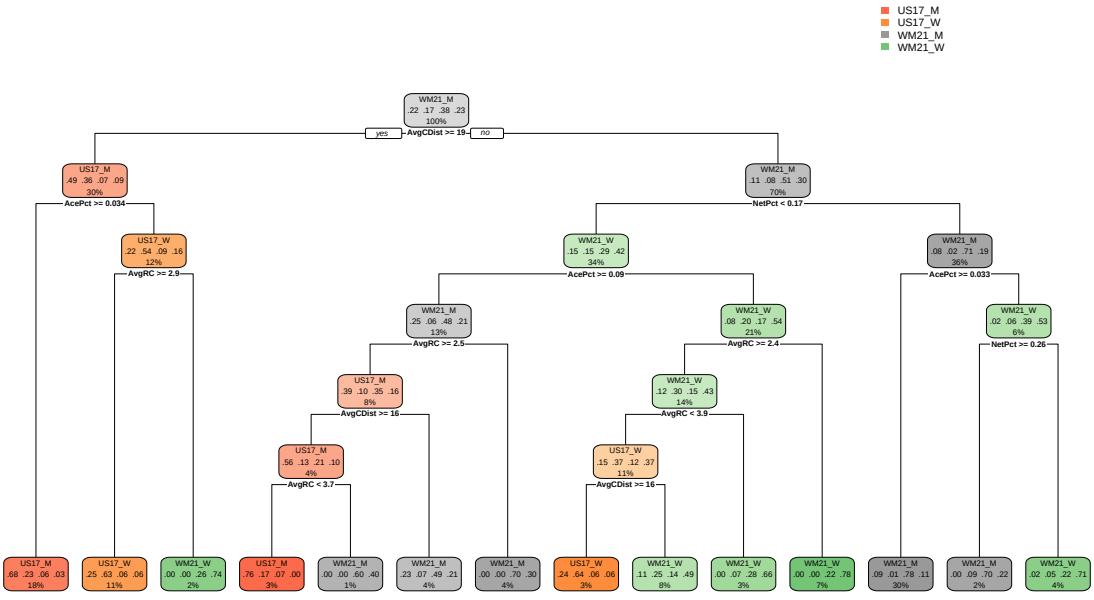



```
## [1] 0.72
```

The resulting plot (below) of gender prediction for this classification tree shows the results we would therefore expect: predictions of male for the larger percentages of aces and of net play.



Recursive partitioning also does a strong job of classifying by tournament and gender simultaneously, with an accuracy rate in the high 60s. We expose the model to our four strongest predictors all at once, and we see, as one example, that the model predicts Wimbledon men's data most frequently when distances covered are low, net attack rates are high, and ace rates are also high (third box from the bottom right). All of these characteristics align with earlier findings.



Overall, classification trees perform essentially as well as knn models, and especially well with combined tournament-gender predictions. What's more, we can make easier sense of their results.

Method	TmtByG	GenByG	TgByG	TmtByS	GenByS	TgByS
Guess	0.51	0.49	0.24	0.55	0.46	0.22
LogReg	0.74	0.64	-	0.8	0.77	-
Knn	0.79	0.61	0.5	0.94	0.73	0.62
CTree	-	-	-	0.9	0.72	0.68

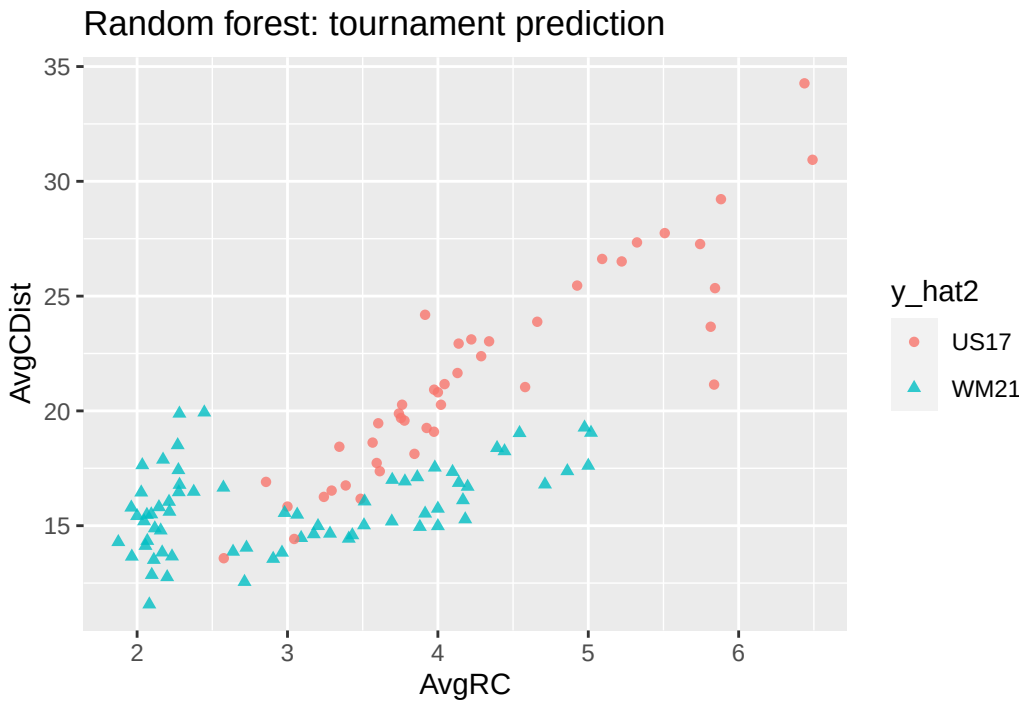
Random forests

We explore random forest models as our final modeling approach. Thanks to repeated bootstrap sampling, random forest models are more robust to adjustments in input data compared to classification trees and more stable in their predictions generally.

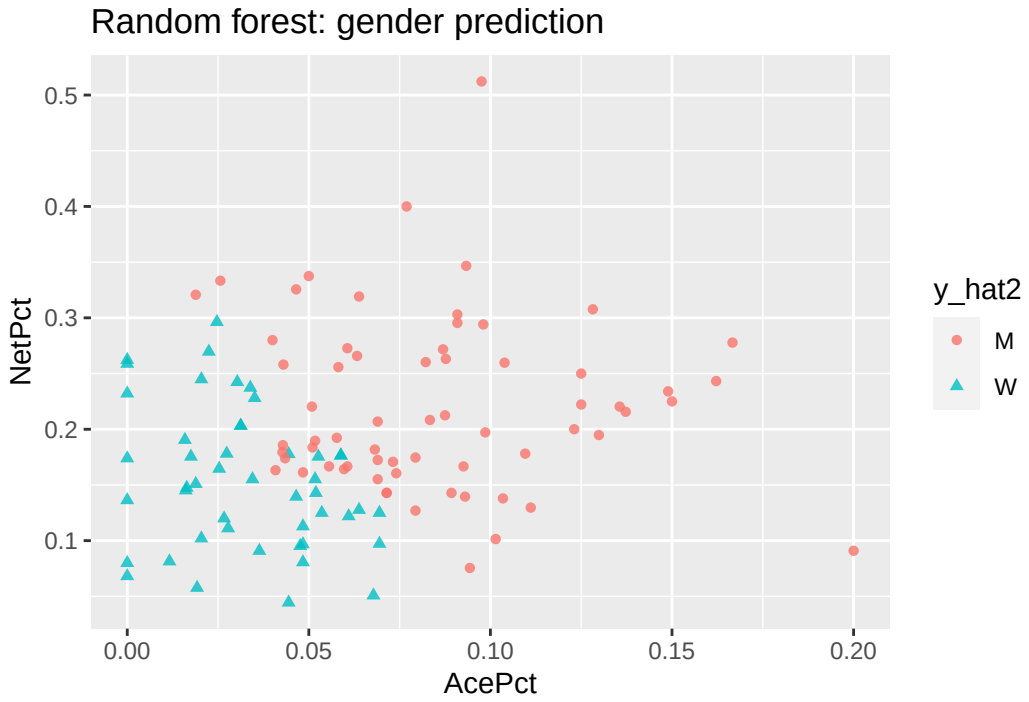
Our random forest model for tournament prediction (again, looking only at data grouped by set) does a similarly strong job of distinguishing Wimbledon data from US Open data. The visual below of predicted tournament is well familiar by now, as is its associated accuracy rate in the low 90s.

```
#tuned version of tournament by set
  train_rft <- train(Tournament ~ AvgCDist + AvgRC,
                     method = "Rborist",
                     tuneGrid = data.frame(predFixed = 2, minNode = c(3, 25)),
                     data=tpbs_train)
  y_hat2 <- predict(train_rft, tpbs_test)
  tsrf <- confusionMatrix(y_hat2, factor(tpbs_test$Tournament))$overall["Accuracy"]
  tsrf <- round(as.numeric(tsrf),2)
  tsrf
```

```
## [1] 0.93
```



The random forest gender prediction model yields an strong accuracy rate just under 75%, with a prediction plot to match.



Finally, with an accuracy rate of 70%, random forest modeling does the best job that we have seen so far of predicting tournament and gender combination. We suspect that bootstrap aggregation helps to increase the effective amount of data available to the model for training.

```
#tuned version of TGComb by set
  train_rft <- train(TGComb ~ AvgRC + AvgCDist + AcePct + NetPct,
                     method = "Rborist",
                     tuneGrid = data.frame(predFixed = 2, minNode = c(3, 25)),
                     data=tpbs_train)
  y_hat2 <- predict(train_rft, tpbs_test)
  tgsrf <- confusionMatrix(y_hat2, factor(tpbs_test$TGComb))$overall["Accuracy"]
  tgsrf <- round(as.numeric(tgsrf),2)
  tgsrf
```

```
## [1] 0.7
```

Overall, random forest model accuracy results stack up quite well: they are slightly stronger than classification tree results, they are comparable with knn model accuracy rates generally, and are the best when predicting tournament-gender combination.

Method	TmtByG	GenByG	TgByG	TmtByS	GenByS	TgByS
Guess	0.51	0.49	0.24	0.55	0.46	0.22
LogReg	0.74	0.64	-	0.8	0.77	-
Knn	0.79	0.61	0.5	0.94	0.73	0.62
CTree	-	-	-	0.9	0.72	0.68
RForest	-	-	-	0.93	0.73	0.7

Validation testing

As our last step, in order to verify our results, we re-run each of our best performing models using the `validation` testing set we partitioned at the start of the project. Accordingly, we also combine the current `train` and `test` sets back into their original `trim` set for richer model training.

For classifying tournament and tournament-gender combination, we note that the knn, tree, and random forest models all produced closely comparable accuracy rates. So, we will test all 3 models against the validation set for those two classification challenges to see which performs best. For classifying gender, the logistical regression model proved to be the strongest thus far (78% accuracy), so we will select that model.

Below is one example of code for validation testing: in this case, our classification tree model predicting tournament using the `trim` set to train and the `validation` set for final testing. The associated prediction chart is supplied as well.

We see that our initial results are largely confirmed by the validation tests, with the exception of an notable improvement in the knn model predicting gender (from 73% to 82%). In fact, the knn model benefitted so much from the additional training data here that it overtook the logistic regression model as the best for predicting gender.

```
#Classification tree vs. validation set
  #tournament by set
  y <- tpbs_trim$Tournament
  train_rpart <- train(Tournament ~ AvgRC + AvgCDist,
                       method = "rpart",
                       tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
                       data=tpbs_trim)
  cpbest <- train_rpart$bestTune
  #once tuned, time for rpart with graphs
  fit_rpart <- rpart(Tournament ~ AvgRC + AvgCDist,
                     data=tpbs_trim,
                     cp=cpbest,
                     method = "class")
  # rpart.plot(fit_rpart) (excluded visually here)
  y_hat <- predict(fit_rpart, tpbs_validation, type="class")
  cm <- confusionMatrix(factor(y_hat), factor(tpbs_validation$Tournament))
  tsctv <- cm$overall["Accuracy"]
  tsctv <- round(as.numeric(tsct),2)
```

Final validation set accuracy rates are displayed here.

Method	TmtByS	GenByS	TgByS
LogReg	-	0.8	-
Knn	0.97	0.82	0.54
CTree	0.9	0.72	0.68
RForest	0.93	0.73	0.7

Conclusion

Summary

This project aimed to build machine learning models to classify raw point-by-point tennis data by tournament affiliation (Wimbledon vs. US Open), by gender affiliation (women's vs. men's play), or by both at the same time. In the end, we found that knn models could predict tournament classification with better than 95% accuracy, and that random forest models could predict gender and tournament-gender combination with better than 70% accuracy at least.

Impact

One helpful impact of this project is simply increased credibility in the content of the underlying data. If the data had resulted in non-believable findings (for example, that rallies consistently last longer on grass), that would have negatively impacted the ability of others to use this data set. We are glad to have, at minimum, helped to show that the Match Charting Project produces generally credible data.

Limitations:

This study was limited to an extent by the amount of raw data it examined. We found an unfortunate absence of reliable "distance covered" measurement in the majority of the available data sets, and thus chose to exclude them. The fact that our data came from different calendar years could have also had some impact on the results, though we are confident that there have not been such drastic changes to patterns of overall top-tier play in recent years (the author is an active tennis coach).

We were also limited, potentially, by the accuracy of the team of match charters whose judgements about certain outcomes (whether a missed shot qualifies as an "unforced error") are baked into the data they produce. We have no reason to doubt their entries, but no realistic way to confirm them, either.

Future Work

Regarding future work, it would be interesting to see if the patterns observed here were also found within much larger samples of point-by-point data available from these major tournaments, even if distance-traveled metrics had to be excluded. It would also be interesting to explore how match statistics change as tournaments progress and only the best-performing players advance into each successive round.

Acknowledgements

We gratefully acknowledge the Match Charting Project and Jeff Sackmann's efforts in compiling a GitHub repository of point-by-point data from the last decade's worth of tennis grand slam tournaments, available at: https://github.com/JeffSackmann/tennis_slam_pointbypoint. We also acknowledge EdX Data Science course professor Rafael Irizarry, and his thorough textbook, available online at: https://rafalab.github.io/dsbook/.