EDA

2023-11-15

Here we will be performing some basic analysis on the data to get an idea of any major problems with the dataset before we build the model. I.e. if we find any irrelevant data that we can remove this will greatly simplify things later down the line.

Firstly we create our set of years and define the test set to be the year 2023

```r
#List of years from 1968 to 2023
years<-seq(1968,2023)

#Test set year
test_year<-2023

#Loading test set
library(readr)
test_set<-read_csv(paste("tennis_wta/wta_matches_",as.character(test_year),".csv", sep=""),show_col_typ
```

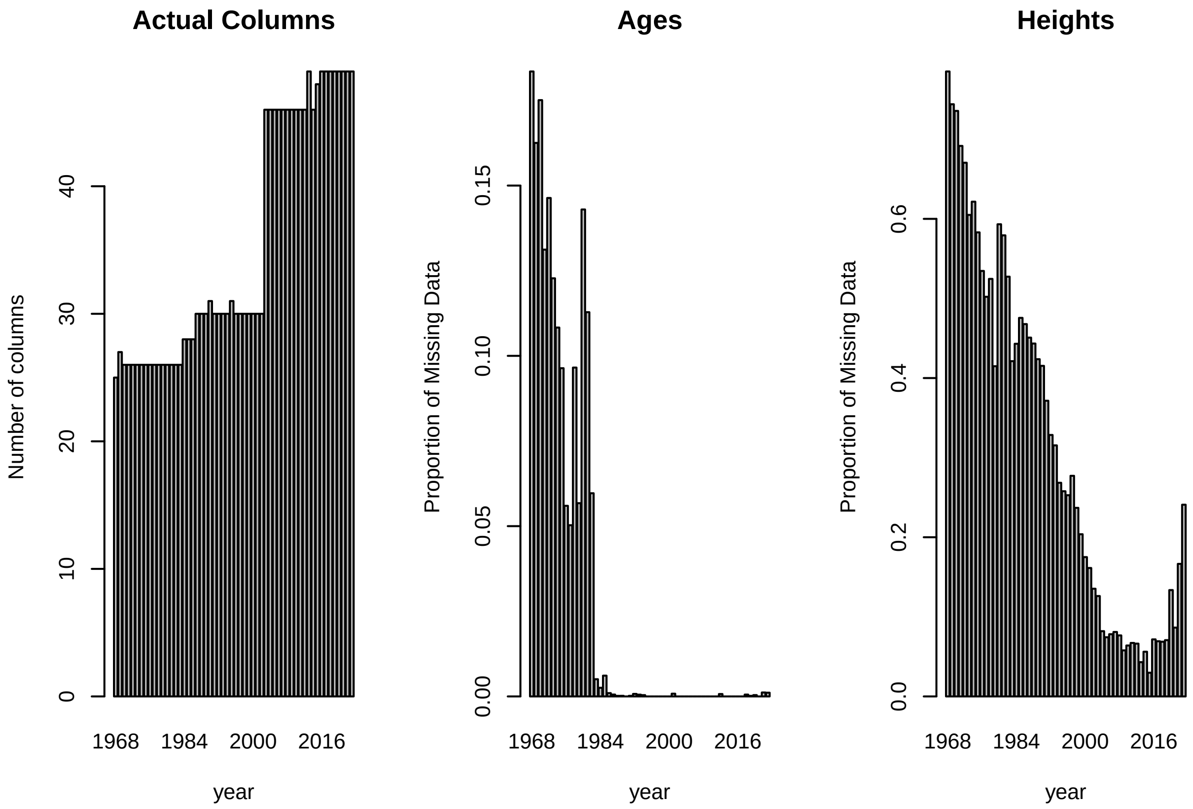
Next we want to see how much data is actually available. From briefly skimming through the datasets we saw that not all years contain the same amount of variables. Therefore we quantify the amount of variables by measuring how many columns in the data frame actually contain data. Then we will also potentially use age and height as covariates in our model, although again by quickly looking at the .csv files we can see a fair amount of missing data. Hence we also measure this and see if it changes by year.

```r
#Checking if a column is empty
containsdata<-function(column){
  if (all(is.na(column))){
    return(FALSE)
  }
  else{
    return(TRUE)
  }
}

#How many columns contain data
fullcolumns<-function(data){
  num<-0
  for (i in 1:dim(data)[2]){
    if (containsdata(data[,i])){
      num<-num+1
    }
  }
  return(num)
}
```

```r
#Checking for missing data in age
ismissing_age<-function(data){
  num_missing<-0
  for (column in c('winner_age','loser_age')){
    for (j in 1:dim(data[,column])[1]){
      if (all(is.na(data[j,column]))){
        num_missing<-num_missing+1
      }
    }
  }
  return(num_missing)
}

#Checking for missing data in height
ismissing_height<-function(data){
  num_missing<-0
  for (column in c('winner_ht','loser_ht')){
    for (j in 1:dim(data[,column])[1]){
      if (all(is.na(data[j,column]))){
        num_missing<-num_missing+1
      }
    }
  }
  return(num_missing)
}
```

By plotting the amount of variables (non-zero columns) and percentage of missing data across time, we can see that the older data is much worse than the current, which is to be expected. For instance, the data around 1970 only has about half the variables recorded as in 2023, and over half the heights of players are not recorded. We can also see that for modern data, the proportion of missing age data is negligible. As expected also the amount of height data collected increases with time, although strangely falters past the 2010s. However we will see later why this might be.

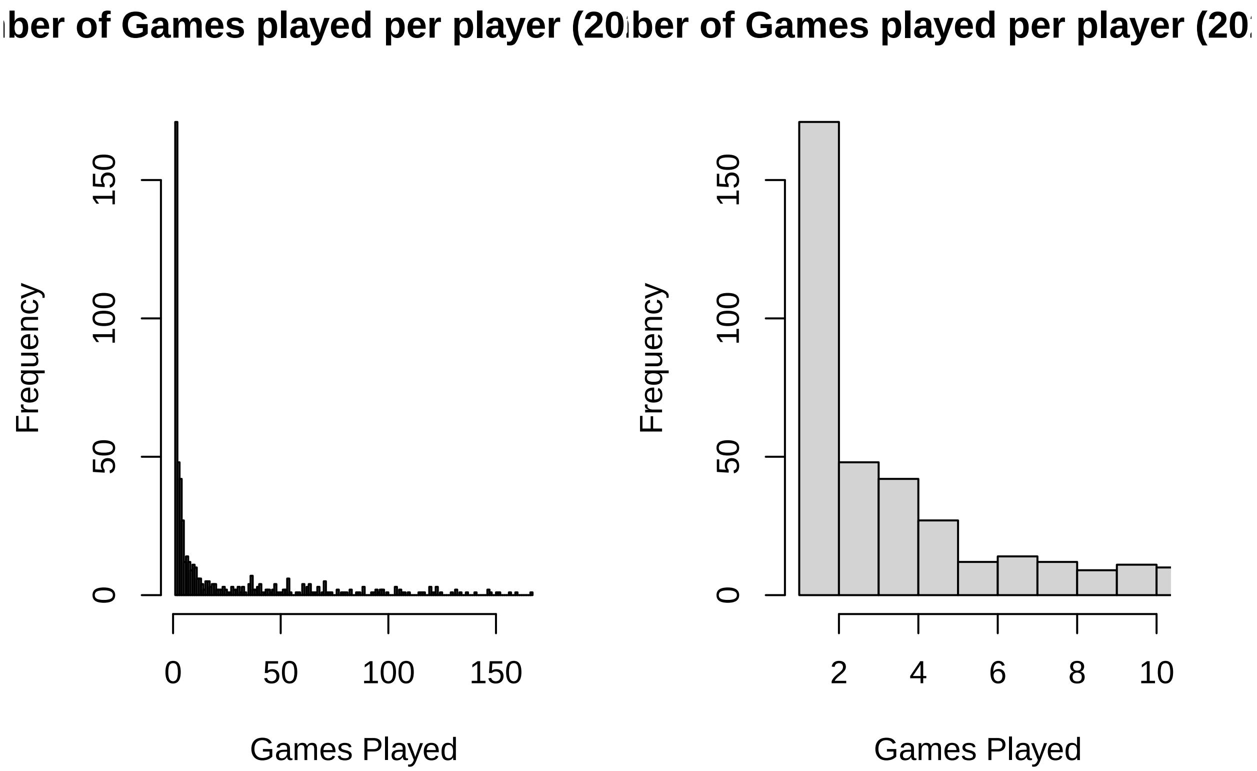
```r
#Plotting results
library(ggplot2)
par(mfrow=c(1,3))
barplot(fullcols,names.arg = years,xlab='year',ylab='Number of columns',main='Actual Columns')
barplot(mis_ages,names.arg = years,xlab='year',ylab='Proportion of Missing Data',main='Ages')
barplot(mis_hts,names.arg = years,xlab='year',ylab='Proportion of Missing Data',main='Heights')
```



Now we have justification for sticking mostly to modern data, we can start to think about which players are relevant in our model. We suspect that not all players need to necessarily be included. For instance, a rogue player that has only played one game will input very little into our model's predictive power. We can quantify this suspicion by plotting each players total number of games played from 2020-2022.

By plotting a histogram of frequencies (Left), it is immediately obvious that our worries were correct, and a large proportion of players have played an insignificant amount of games over three years. In fact by zooming in on this histogram (right), a lot of players have indeed only played one game and then never played again.

```
#Finding how many games each player has played from 2020-2022
N<-length(names)
num_games<-rep(0,N)
for (i in 1:N){
  num_games[i]<-length(which(num_names==names[i]))
}

#Plotting results
par(mfrow=c(1,2))
hist(num_games,breaks=200,xlab='Games Played',main='Number of Games played per player (2020-2022)')
hist(num_games,breaks=200,xlim=c(1,10),xlab='Games Played',main='Number of Games played per player (202
```



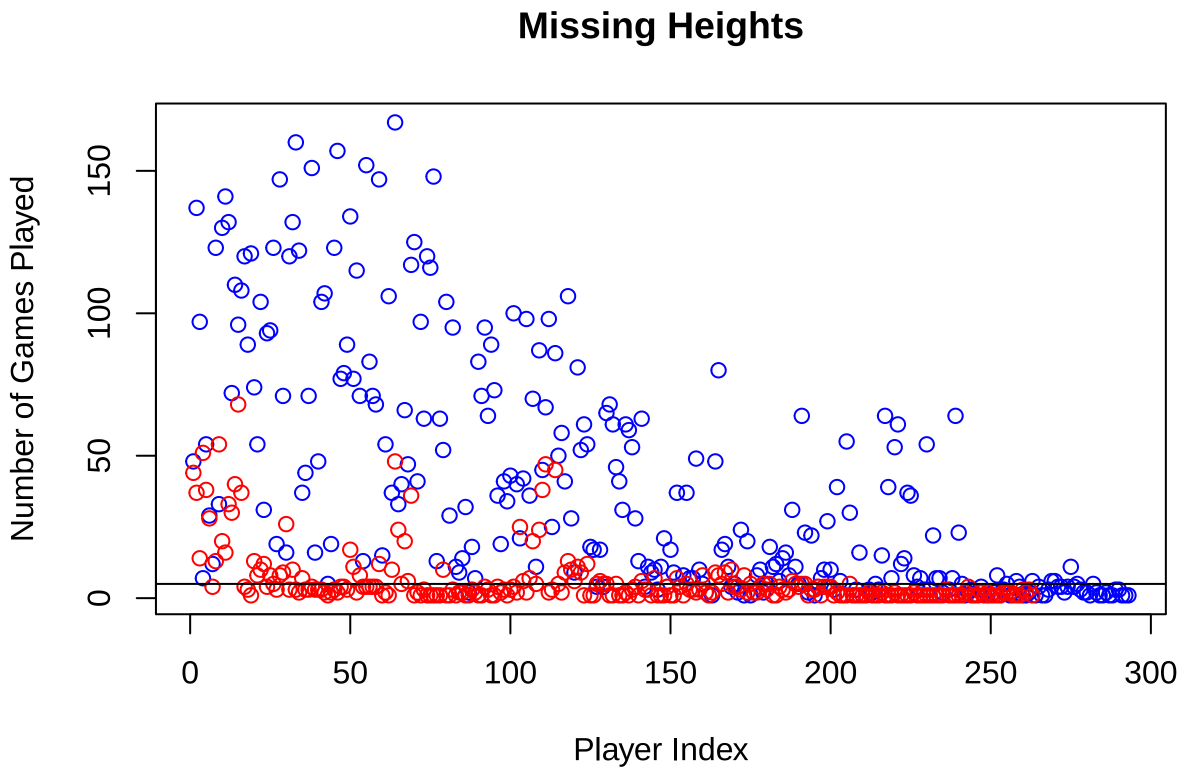
To then create a list of 'significant' players, we only include players from 2020-2022 who have played more than 5 games over those 3 years, and have then also played again in 2023 (So we can predict their performance in the test set).

```
#Players who played in the test set (2023)
test_names<-c(as.matrix(test_set[,'winner_name']),as.matrix(test_set[,'loser_name']))

#We can only keep players who played more than 5 games in the past 3 years and are still playing in
#2023
main_names<-names[num_games > 5 & names %in% test_names]
```

Using this, we can then clean the data by only including the variables we care about, and then only including players from the list of 'significant' players.

Returning to the confusing amount of missing height data from 2010 onwards, we can test whether a lot of this missing data comes from these rogue players who have only played a few games. By plotting each players number of games played, we can see that indeed the players with missing data (red) have been removed in our cleaned data, as most have played less than 5 games (black line). Hence this cleaned data has significantly less missing data than the original.

```
#Checking if a Player has a height or not
noheight<-rep(FALSE,N)
for (i in 1:N){
  name<-names[i]
  if (name %in% df1$winner_name){
    j<-min(which(df1[,'winner_name']==name))
    height<-df1[j,'winner_ht']
  }
  if (name %in% df1$loser_name){
    j<-min(which(df1[,'loser_name']==name))
    height<-df1[j,'loser_ht']
  }
  if (is.na(height)){
    noheight[i]<-TRUE
  }
}

#Plotting each player against their number of games played, with different colors for whether their hei
plot(num_games[!noheight],col='blue',xlab='Player Index',ylab='Number of Games Played',main='Missing He
points(num_games[noheight],col='red')
abline(h=5)
```



Finally, I will turn these data-cleaning functions into one function we will include in the R packages, with parameters for start year, end year, test set, and number of games played to be significant.

```
# Creating super function

TennisTidyr<-function(startyear,endyear,testset,numgames,filename){
  # Adding errors for if years will not match the dataset
  if (!(startyear > 1967 & startyear < 2023)){
    return("Error: Start year must be between 1968 and 2022")
  }
```

```r
  if (!(endyear > 1967 & endyear < 2023)){
    return("Error: End year must be between 1968 and 2022")
  }
  if (!(testset > 1967 & testset < 2024)){
    return("Error: Test set must be between 1968 and 2023")
  }
  if (startyear > endyear){
    return("Error: Start year must be less than end year")
  }
  if (endyear > testset){
    return("Error: End year must be less than test set")
  }

  #Creating blank dataset for training set, then appending each year
  train<-tibble()
  num_names<-c()
  for (i in startyear:endyear){
    data<-read_csv(paste(filename,as.character(i),".csv", sep=""),show_col_types = FALSE)
    num_names<-append(num_names,c(as.matrix(data[,'winner_name']),as.matrix(data[,'loser_name'])))
    train<-rbind(train,data)
  }
  #List of all unique names in that time period
  names<-unique(num_names)

  #Number of games played per player
  N<-length(names)
  num_games<-rep(0,N)
  for (i in 1:N){
    num_games[i]<-length(which(num_names==names[i]))
  }

  #Players who played in the test set (2023)
  test_set<-read_csv(paste(filename,as.character(testset),".csv", sep=""),show_col_types = FALSE)
  test_names<-c(as.matrix(test_set[,'winner_name']),as.matrix(test_set[,'loser_name']))

  #List of 'significant' players
  main_names<-names[num_games > numgames & names %in% test_names]

  #Only selecting columns we currently care about
  train<-train[,c('surface','winner_name','winner_hand','winner_ht','winner_age','loser_name','loser_han
  test<-test_set[,c('surface','winner_name','winner_hand','winner_ht','winner_age','loser_name','loser_h

  #Only selecting players we care about
  train<-train[train$winner_name %in% main_names,]
  train<-train[train$loser_name %in% main_names,]

  test<-test[test$winner_name %in% main_names,]
  test<-test[test$loser_name %in% main_names,]


  return(list(train,test))
}
```

```r
testdata<-TennisTidyr(2020,2022,2023,10,"tennis_wta/wta_matches_")

testdata[2]
```

```
## [[1]]
## # A tibble: 1,596 x 9
##    surface winner_name    winner_hand winner_ht winner_age loser_name loser_hand
##    <chr>   <chr>          <chr>           <dbl>      <dbl> <chr>      <chr>
##  1 Hard    Jessica Pegula R                 170       28.8 Martina T~ L
##  2 Hard    Madison Keys   R                 178       27.8 Lucia Bro~ R
##  3 Hard    Jessica Pegula R                 170       28.8 Iga Swiat~ R
##  4 Hard    Madison Keys   R                 178       27.8 Magda Lin~ R
##  5 Hard    Martina Trevi~ L                 160       29.1 Maria Sak~ R
##  6 Hard    Maria Sakkari  R                 172       27.4 Petra Mar~ R
##  7 Hard    Donna Vekic    R                 179       26.5 Despina P~ R
##  8 Hard    Iga Swiatek    R                 176       21.5 Martina T~ L
##  9 Hard    Magda Linette  R                 171       30.8 Lucia Bro~ R
## 10 Hard    Jessica Pegula R                 170       28.8 Harriet D~ R
## # i 1,586 more rows
## # i 2 more variables: loser_ht <dbl>, loser_age <dbl>
```