

# Case\_Study\_4\_Code

Brian Gaither, Sean Mcwhirter, Andrew Mejia, Sabrina Purvis

2021-02-01 07:55:58

## Contents

Functions . . . . .	4
<b>The gen_Table function will parse through the table of 20 records and 15 observations and parse the table into its own data frame from xml2 read)html function and then use the pipe operator to use rvest nodes to find the table structure</b>	5
then the fucntion will insert the metadata for the query parameters of year, division, section, page, source link and sex	5
This function will use all available cores on machine	6
It will process the years in parrell.	6
This code has been adapted from	6
<a href="https://github.com/ngupta23/ds7333_qtw/blob/master/case_study_2/submission_Kannan_Moro_Gupta/code/CS2_ETL.Rmd">https://github.com/ngupta23/ds7333_qtw/blob/master/case_study_2/submission_Kannan_Moro_Gupta/code/CS2_ETL.Rmd</a>	6
<a href="https://cran.r-project.org/web/packages/doSNOW/doSNOW.pdf">https://cran.r-project.org/web/packages/doSNOW/doSNOW.pdf</a>	6
<a href="https://stackoverflow.com/questions/36794063/r-foreach-from-single-machine-to-cluster">https://stackoverflow.com/questions/36794063/r-foreach-from-single-machine-to-cluster</a>	6
<a href="https://cran.r-project.org/web/packages/progress/progress.pdf">https://cran.r-project.org/web/packages/progress/progress.pdf</a>	6
<a href="https://www.r-bloggers.com/2013/08/the-wonders-of-foreach/">https://www.r-bloggers.com/2013/08/the-wonders-of-foreach/</a>	6
The purpose of this function is to transform the raw tables from the scrape . . . . .	7
<a href="https://stackoverflow.com/questions/50040968/convert-a-duration-hms-to-seconds">https://stackoverflow.com/questions/50040968/convert-a-duration-hms-to-seconds</a> . . . . .	7
<a href="https://stackoverflow.com/questions/10835908/is-there-a-way-to-convert-mmss-00-to-seconds-00">https://stackoverflow.com/questions/10835908/is-there-a-way-to-convert-mmss-00-to-seconds-00</a> . . . . .	7
<a href="https://stackoverflow.com/questions/24173194/remove-parentheses-and-text-within-from-strings-in-r">https://stackoverflow.com/questions/24173194/remove-parentheses-and-text-within-from-strings-in-r</a> . . . . .	7

Perform Scrape . . . . .	8
Caution takes a long time without hexacore machine . . . . .	8
Load tables from file . . . . .	9
Preview of raw table scrapes . . . . .	9
Perform the table transformation and remove metadata columns and other columns . . . . .	10
Preview of the transformed table . . . . .	10
Preview of NA columns . . . . .	11
Create a DF containing the NA counts of each feature . . . . .	12
Plots of prominent NA columns . . . . .	12
We will remove NAs and only focus on completed records and remove NAs for further processing and reindex the table. . . . .	14
Final dataframe metadata . . . . .	14
Generate plots of quick EDA for ages across years . . . . .	15
Conclusion / Recommendations . . . . .	43

```
library(tidyverse)
```

```
## -- Attaching packages ----- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     vforcats 0.5.0
```

```
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()
```

```
library(hms)
library(rvest)
```

```
## Loading required package: xml2
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:purrr':
##
```

```
##     pluck
```

```
## The following object is masked from 'package:readr':
##
```

```
library(lubridate)

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:hms':
##
##     hms

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(foreach)
```

```
##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
library(stringr)
library(iterators)
library(progress)
library(doParallel)
```

```
## Loading required package: parallel
```

```
library(doSNOW)
```

```
## Loading required package: snow
```

```
##
## Attaching package: 'snow'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, clusterSplit, makeCluster, parApply,
##     parCapply, parLapply, parRapply, parSapply, splitIndices,
##     stopCluster
```

```
library(dplyr)
library(states)
```

```
##
## Attaching package: 'states'
```

```

## The following object is masked from 'package:readr':
##
##     parse_date

library(ggplot2)
library(ggthemes)
library(SiZer)
library(plotly)

##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout

years = c(1999:2012)
division = 'Overall+Women'
section = '10M'
sex = 'W'

```

**Functions** #The gen\_Link function will generates the link with the query parameters for the searchable database

```

gen_Link = function(year,division,section,page=1,sex){

  paste0('http://www.cbballtimeresults.org/performances',
    '?division=',division,'&page=',page,
    '&section=',section, '&sex=',sex,
    '&utf8=%E2%9C%93', '&year=',year)
  #, '?utf8=%E2%9C%93&section=',section
  #, '&year=',year, '&division=',division, '&page=', page)

}

```

The gen\_Table function will parse through the table of 20 records and 15 observations

and parse the table into its own data frame from xml2 read)html function and then use

the pipe operator to use rvest nodes to find the table structure

then the fucntion will insert the metadata for the query parameters of year, division, section, page, source link and sex

```
gen_Table = function(year,division,section,page, sex){

  #use gen_link function to get link to page
  genlink=gen_Link(year,division,section,page=page, sex=sex)

  #read the page, and grab to 'table' tag
  single_table = xml2::read_html(genlink) %>%
    rvest::html_nodes("table") %>%
    rvest::html_table(fill=TRUE)

  #get the table and add metadata for the query parameters
  table_out = single_table[[1]] %>%
    mutate(year=year, divisionTitle=division, section=section, page=page, source=genlink, sex = sex)

}
```

This function will use all available cores on machine

It will process the years in parallel.

This code has been adapted from

[https://github.com/ngupta23/ds7333\\_qtw/blob/master/case\\_study\\_2/submission\\_Kannan\\_Moro\\_Gupta/code/CS2\\_ETL.Rmd](https://github.com/ngupta23/ds7333_qtw/blob/master/case_study_2/submission_Kannan_Moro_Gupta/code/CS2_ETL.Rmd)

<https://cran.r-project.org/web/packages/doSNOW/doSNOW.pdf>

<https://stackoverflow.com/questions/36794063/r-foreach-from-single-machine-to-cluster>

<https://cran.r-project.org/web/packages/progress/progress.pdf>

<https://www.r-bloggers.com/2013/08/the-wonders-of-foreach/>

```
scrapeTables = function(years,division,section, sex, max_itr = 500){

  library(progress)
  library(doParallel)
  library(doSNOW)

  #Initialize Parallel Process to detect number of cores
  #https://cran.r-project.org/web/packages/doSNOW/doSNOW.pdf
  #https://stackoverflow.com/questions/36794063/r-foreach-from-single-machine-to-cluster

  #Generate and register initial clusters based on cores, otherwise, this is a long process
  cl = makeCluster(detectCores())
  doSNOW::registerDoSNOW(cl)

  #Generate progress bar for the parallel loop based on number of years
  #https://cran.r-project.org/web/packages/progress/progress.pdf
  progBar = progress::progress_bar$new(total = length(years),format='[:bar] :percent :eta')
  progress = function(n) progBar$tick()

  #Initialize a parallel loop per each year
  #Initialize tableRaw as empty to loop to be populated as a table dataframe from gen_Table function
  tableRaw=NULL

  #tableRaw will now use for each using years as the iterator to use .combine to rbind
  #.export will do the gen_Table and gen_Link functions simultaneously and
```

```

#options.snow will show the progress bar
# the %dopar% will process all the years simultaneously.
#https://www.r-bloggers.com/2013/08/the-wonders-of-foreach/
tableRaw = foreach(y=years
                    ,.combine=rbind,.export=c('gen_Table','gen_Link')
                    ,.options.snow = list(progress=progress)) %dopar%
{

  library(foreach)
  library(dplyr)
  #initialize isCompleted variable as FALSE for bool conditions to see if loop has been completed
  isCompleted=FALSE

  #Initiate loop since most pages are 487, we will only loop for the iterations for max_itr
  tableRaw=foreach(p=c(1:max_itr),.combine=rbind) %do%
    if(!isCompleted) {
      message('getting year:',y, ' page:',p,appendLF = F)
      #get the table of the current page
      table = gen_Table(year=y
                         ,division=division
                         ,section=section
                         ,page=p
                         ,sex=sex)
      message(' rows:',nrow(table))
      isCompleted = nrow(table)==0 #if there is record, we are at the last page, no need to read .
      return(table)
    }
    return(tableRaw)
  }
  #Deactivate the cluster of cores
  stopCluster(cl)
  #save the raw data to rda format for later processing based on gender
  saveRDS(tableRaw,file=paste0('CB',sex,'tableRaw.rds'))

  return(tableRaw)
}

```

The purpose of this function is to transform the raw tables from the scrape

<https://stackoverflow.com/questions/50040968/convert-a-duration-hms-to-seconds>

<https://stackoverflow.com/questions/10835908/is-there-a-way-to-convert-mmss-00-to-seconds-00>

<https://stackoverflow.com/questions/24173194/remove-parentheses-and-text-within-from-strings-in-r>

```

tableTransform =function(data_df, cols_to_remove=NULL){
  dataDF = data_df %>%
    #Separate Home town into separate columns
    separate(col = 'Hometown', c('HomeTown', 'HomeState'), sep = ',', extra = 'merge', remove = TRUE, f
  #Separate PiS/TiS into separate columns

```

```

separate(col='PiS/TiS',c('PiS','TiS'),sep='\\/'
         ,extra='drop',remove=TRUE) %>%
#Separate PiD/TiD into seperate columns
separate(col='PiD/TiD',c('PiD','TiD'),sep='\\/'
         ,extra='drop',remove=TRUE) %>%
#Trim the casted upper HomeTown strings of whitespace
mutate(HomeTown = toupper(trimws(HomeTown))
       , HomeState = toupper(trimws(HomeState))
       , HomeTown = ifelse(HomeTown %in% c('NR', '', NULL), NA, toupper(trimws(HomeTown)))
       , HomeState = ifelse(HomeState %in% c('NR', '', NULL), NA, toupper(trimws(HomeState)))
#Check if HomeState is in state.abb or DC and return USA else return the HomeTown as the Country
       , HomeCountry = ifelse(HomeState %in% c(state.abb, "DC"), "USA", HomeTown)
#Remove White Space
       , PiS = ifelse(trimws(PiS) %in% c('NR', '', NULL), NA, trimws(PiS))
       , TiS = ifelse(trimws(TiS) %in% c('NR', '', NULL), NA, trimws(TiS))
       , PiD = ifelse(trimws(PiD) %in% c('NR', '', NULL), NA, trimws(PiD))
       , TiD = ifelse(trimws(TiD) %in% c('NR', '', NULL), NA, trimws(TiD))
       , Division = ifelse(trimws(Division) %in% c('NR', '', NULL), NA, trimws(Division))
# Normalize Time to seconds and minutes
       , RawTime = strptime(Time, format='%H:%M:%S')
       , RawTime_S = RawTime$hour * 3600 + RawTime$min * 60 + RawTime$sec
       , RawTime_M = as.numeric(RawTime_S)/60
       , RawPace = strptime(Pace, format = "%M:%OS")
       , RawPace_S = RawPace$min * 60 + RawPace$sec
#Normalize Age where 'NR' as NA
       , Age = ifelse(Age %in% c("NR"), NA, Age)
#Cast Variables as appropriate dtypes
       , Age = as.numeric(Age)
       , RawTime_S = as.numeric(RawTime_S)
       , RawPace_S = as.numeric(RawPace_S)
       , RawPace_M = as.numeric(RawPace_S)/60
       , year = as.factor(year)
#Remove the (<Sex>) from the names
       , Name = str_replace(Name, " \s*\|\|([^\|]+|\|)", '')
)
#Remove columns we do not want
dataDF = dataDF %>% select (-all_of(c(cols_to_remove)))

return(dataDF %>% select(c( Age, year,  HomeTown, HomeState, HomeCountry, RawTime_S, RawTime_M, RawPace_S)))
}

```

## Perform Scrape

Caution takes a long time without hexacore machine

```

#Women_table = scrapeTables(years=years,division = division,section=section,sex = sex, max_itr = 500)

#men_table = scrapeTables(years=years,division = 'Overall+Men',section=section,sex = "M", max_itr = 500)

```

## Load tables from file

```
mens_table <- readRDS("/media/andrew/Seagate Backup Plus Drive/Documents/School/HomeWork/QTW/DS7333/CAS...  
womens_table <- readRDS("/media/andrew/Seagate Backup Plus Drive/Documents/School/HomeWork/QTW/DS7333/C...
```

## Preview of raw table scrapes

```
head(womens_table, n = 10)
```

```
##          Race           Name Age   Time Pace PiS/TiS Division PiD/TiD  
## 1 1999 10M      Jane Omoro (W) 26 0:53:37 5:22  1/2358 W2529 1/559  
## 2 1999 10M      Jane Ngotho (W) 29 0:53:38 5:22  2/2358 W2529 2/559  
## 3 1999 10M Lidiya Grigoryeva (W) NR 0:53:40 5:22  3/2358 NR     NR  
## 4 1999 10M      Eunice Sagero (W) 20 0:53:55 5:24  4/2358 W2024 1/196  
## 5 1999 10M      Alla Zhilyayeva (W) 29 0:54:08 5:25  5/2358 W2529 3/559  
## 6 1999 10M      Teresa Wanjiku (W) 24 0:54:10 5:25  6/2358 W2024 2/196  
## 7 1999 10M      Elana Viazova (W) 38 0:54:29 5:27  7/2358 W3539 1/387  
## 8 1999 10M      Gladys Asiba (W)  NR 0:54:50 5:29  8/2358 NR     NR  
## 9 1999 10M      Nnenna Lynch (W) 27 0:55:39 5:34  9/2358 W2529 4/559  
## 10 1999 10M     Margaret Kagiri (W) 30 0:55:43 5:34 10/2358 W3034 1/529  
##          Hometown year divisionTitle section page  
## 1          Kenya 1999 Overall+Women    10M    1  
## 2          Kenya 1999 Overall+Women    10M    1  
## 3          Russia 1999 Overall+Women   10M    1  
## 4          Kenya 1999 Overall+Women   10M    1  
## 5          Russia 1999 Overall+Women   10M    1  
## 6          Kenya 1999 Overall+Women   10M    1  
## 7          Ukraine 1999 Overall+Women  10M    1  
## 8          Kenya 1999 Overall+Women   10M    1  
## 9 Concord, MA 1999 Overall+Women   10M    1  
## 10         Kenya 1999 Overall+Women  10M    1  
##  
## 1 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 2 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 3 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 4 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 5 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 6 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 7 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 8 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 9 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
## 10 http://www.cballtimeresults.org/performances?division=Overall+Women&page=1&section=10M&sex=W&utf8=  
##          sex  
## 1      W  
## 2      W  
## 3      W  
## 4      W  
## 5      W  
## 6      W
```

```
## 7   W
## 8   W
## 9   W
## 10  W
```

Perform the table transformation and remove metadata columns and other columns

```
cols_for_remove = c("divisionTitle", "source", "Pace", "Time", "RawTime", "RawPace", "page", "Race", "s
mens_table_T = tableTransform(mens_table, cols_to_remove = all_of(cols_for_remove))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with 'NA' in 3 rows [42230,
## 69995, 69996].
```

```
## Warning: Expected 2 pieces. Missing pieces filled with 'NA' in 31 rows [1207,
## 2797, 7583, 9206, 9255, 10991, 12255, 13046, 14812, 15842, 21756, 21819, 22093,
## 22709, 23342, 23680, 25014, 25424, 25443, 25750, ...].
```

```
womens_table_T = tableTransform(womens_table, cols_to_remove = all_of(cols_for_remove))
```

```
## Warning: Expected 2 pieces. Missing pieces filled with 'NA' in 20 rows [3, 8,
## 17, 2176, 7135, 7766, 8777, 9680, 10831, 18391, 18399, 18981, 19694, 20735,
## 21480, 22189, 28388, 29223, 38455, 47506].
```

Preview of the transformed table

```
head(womens_table_T, n = 10)
```

	Age	year	HomeTown	HomeState	HomeCountry	RawTime_S	RawTime_M	RawPace_S
## 1	26	1999	KENYA	<NA>	KENYA	3217	53.61667	322
## 2	29	1999	KENYA	<NA>	KENYA	3218	53.63333	322
## 3	NA	1999	RUSSIA	<NA>	RUSSIA	3220	53.66667	322
## 4	20	1999	KENYA	<NA>	KENYA	3235	53.91667	324
## 5	29	1999	RUSSIA	<NA>	RUSSIA	3248	54.13333	325
## 6	24	1999	KENYA	<NA>	KENYA	3250	54.16667	325
## 7	38	1999	UKRAINE	<NA>	UKRAINE	3269	54.48333	327
## 8	NA	1999	KENYA	<NA>	KENYA	3290	54.83333	329
## 9	27	1999	CONCORD	MA	USA	3339	55.65000	334
## 10	30	1999	KENYA	<NA>	KENYA	3343	55.71667	334
	RawPace_M	sex	section	PiS	TiS	PiD	TiD	
## 1	5.366667	W	10M	1	2358	1	559	
## 2	5.366667	W	10M	2	2358	2	559	
## 3	5.366667	W	10M	3	2358	<NA>	<NA>	
## 4	5.400000	W	10M	4	2358	1	196	
## 5	5.416667	W	10M	5	2358	3	559	
## 6	5.416667	W	10M	6	2358	2	196	
## 7	5.450000	W	10M	7	2358	1	387	
## 8	5.483333	W	10M	8	2358	<NA>	<NA>	
## 9	5.566667	W	10M	9	2358	4	559	
## 10	5.566667	W	10M	10	2358	1	529	

```

dim(womens_table_T)

## [1] 75866    15

summary(womens_table_T)

##      Age          year       HomeTown      HomeState
##  Min.   : 7.00   2012   : 9727   Length:75866   Length:75866
##  1st Qu.:27.00  2011   : 9030   Class  :character  Class  :character
##  Median  :32.00  2010   : 8853   Mode   :character  Mode   :character
##  Mean    :33.85  2009   : 8323
##  3rd Qu.:39.00  2008   : 6395
##  Max.   :87.00  2007   : 5532
##  NA's   :20     (Other):28006
##      HomeCountry      RawTime_S      RawTime_M      RawPace_S
##  Length:75866      Min.   : 3104   Min.   : 51.73   Min.   : 310.0
##  Class  :character  1st Qu.: 5319   1st Qu.: 88.65   1st Qu.: 532.0
##  Mode   :character  Median  : 5849   Median  : 97.48   Median  : 585.0
##                  Mean    : 5893   Mean    : 98.22   Mean    : 589.4
##                  3rd Qu.: 6418   3rd Qu.:106.97   3rd Qu.: 642.0
##                  Max.   :10651   Max.   :177.52   Max.   :1065.0
##
##      RawPace_M      sex        section      PiS
##  Min.   : 5.167   Length:75866   Length:75866   Length:75866
##  1st Qu.: 8.867   Class  :character  Class  :character  Class  :character
##  Median  : 9.750   Mode   :character  Mode   :character  Mode   :character
##  Mean    : 9.823
##  3rd Qu.:10.700
##  Max.   :17.750
##
##      TiS           PiD         TiD
##  Length:75866   Length:75866   Length:75866
##  Class  :character  Class  :character  Class  :character
##  Mode   :character  Mode   :character  Mode   :character
##
##      ##
```

### Preview of NA columns

```

columns = c("Age", "year", "HomeTown", "HomeState", "HomeCountry", "RawTime_S", "RawTime_M", "RawPace_S")

womens_table_T_na = womens_table_T %>% filter_at(vars(all_of(columns)), any_vars(is.na(.)))

head(womens_table_T_na, n = 10)

##      Age year HomeTown HomeState HomeCountry RawTime_S RawTime_M RawPace_S
## 1    26 1999    KENYA      <NA>      KENYA      3217  53.61667      322
## 2    29 1999    KENYA      <NA>      KENYA      3218  53.63333      322

```

```

## 3 NA 1999 RUSSIA <NA> RUSSIA 3220 53.66667 322
## 4 20 1999 KENYA <NA> KENYA 3235 53.91667 324
## 5 29 1999 RUSSIA <NA> RUSSIA 3248 54.13333 325
## 6 24 1999 KENYA <NA> KENYA 3250 54.16667 325
## 7 38 1999 UKRAINE <NA> UKRAINE 3269 54.48333 327
## 8 NA 1999 KENYA <NA> KENYA 3290 54.83333 329
## 9 30 1999 KENYA <NA> KENYA 3343 55.71667 334
## 10 NA 1999 LANCASTER PA USA 3576 59.60000 358
## RawPace_M sex section PiS Tis PiD TiD
## 1 5.366667 W 10M 1 2358 1 559
## 2 5.366667 W 10M 2 2358 2 559
## 3 5.366667 W 10M 3 2358 <NA> <NA>
## 4 5.400000 W 10M 4 2358 1 196
## 5 5.416667 W 10M 5 2358 3 559
## 6 5.416667 W 10M 6 2358 2 196
## 7 5.450000 W 10M 7 2358 1 387
## 8 5.483333 W 10M 8 2358 <NA> <NA>
## 9 5.566667 W 10M 10 2358 1 529
## 10 5.966667 W 10M 17 2358 <NA> <NA>

```

```
dim(womens_table_T_na)
```

```
## [1] 262 15
```

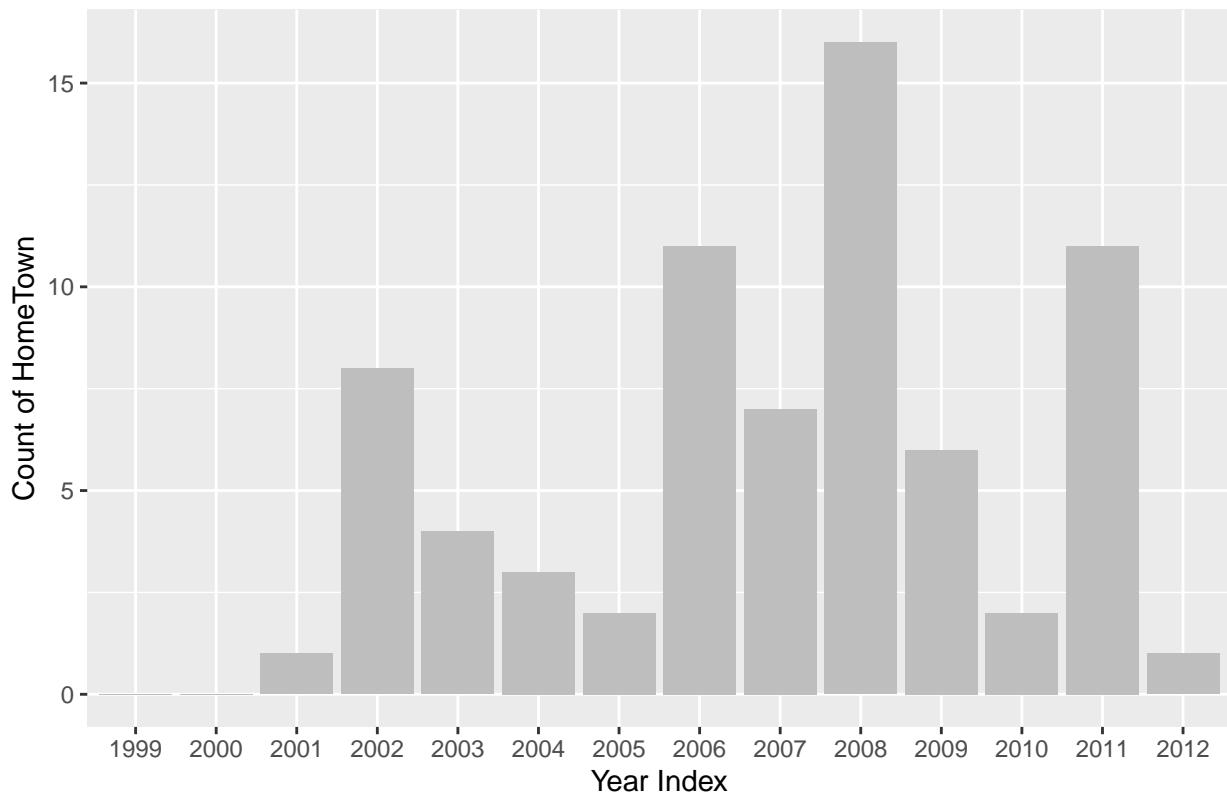
Create a DF containing the NA counts of each feature

```
na_df = data.frame(rowsum(is.na(womens_table_T)), womens_table_T$year)
na_df = cbind(year_idx = rownames(na_df), na_df)
```

Plots of prominent NA columns

```
p1_na = ggplot(na_df, aes(x=year_idx, y=HomeTown)) + geom_bar(stat = "identity", fill = "grey") + labs(
```

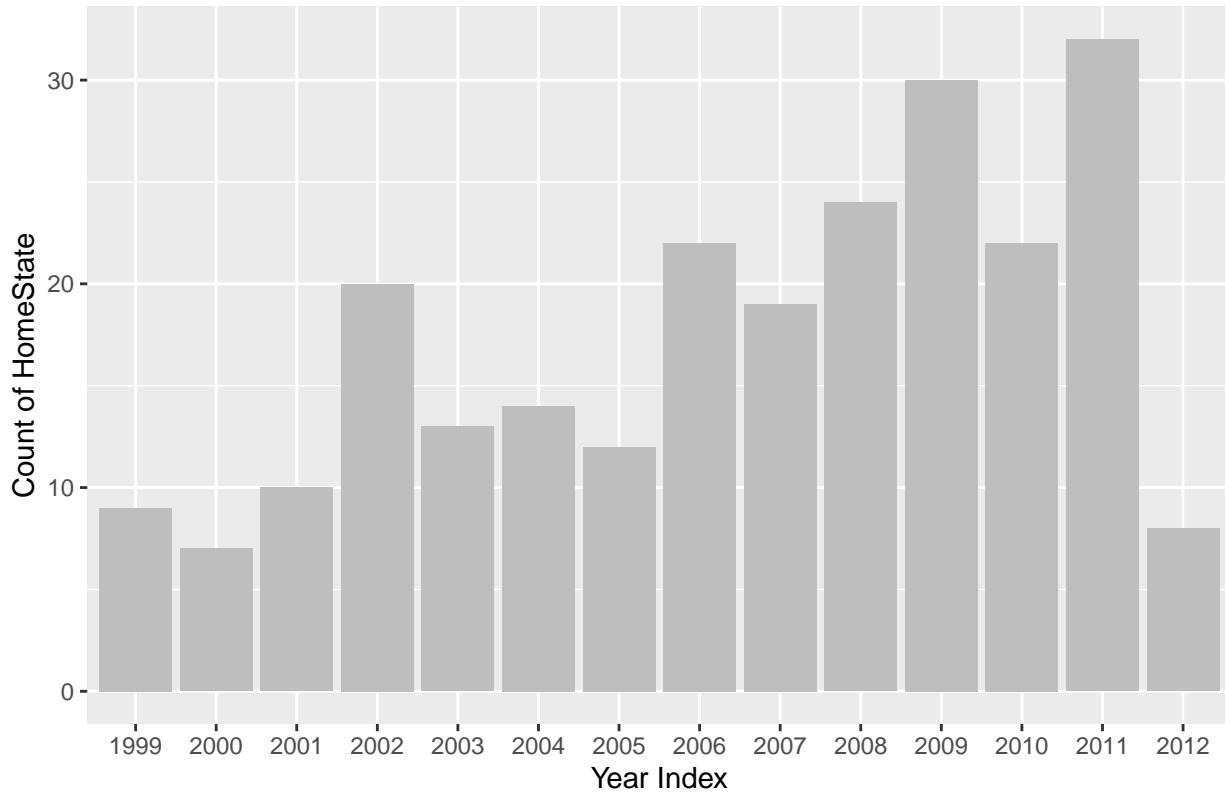
### Counts of missing HomeTown Records by Year



```
p2_na = ggplot(na_df, aes(x=year_idx, y=HomeState)) + geom_bar(stat = "identity", fill = "grey") + labs
```

p2\_na

## Counts of missing HomeState Records by Year



We will remove NAs and only focus on completed records and remove NAs for further processing and reindex the table.

```
womens_table_T = womens_table_T[complete.cases(womens_table_T), ]
row.names(womens_table_T) = NULL
```

Final dataframe metadata

```
head(womens_table_T, n = 10)
```

```
##   Age year     HomeTown HomeState HomeCountry RawTime_S RawTime_M RawPace_S
## 1 27 1999      CONCORD      MA       USA    3339  55.65000    334
## 2 30 1999      EUGENE       OR       USA    3373  56.21667    337
## 3 37 1999  BLOOMINGTON      MN       USA    3443  57.38333    344
## 4 39 1999  ALBUQUERQUE      NM       USA    3444  57.40000    344
## 5 32 1999  CHAPEL HILL      NC       USA    3471  57.85000    347
## 6 30 1999  WASHINGTON       DC       USA    3485  58.08333    349
## 7 31 1999      COLUMBIA      MD       USA    3516  58.60000    352
## 8 25 1999  ALEXANDRIA      VA       USA    3582  59.70000    358
## 9 38 1999 SILVER SPRING      MD       USA    3588  59.80000    359
## 10 31 1999  ROCKVILLE      MD       USA    3630  60.50000    363
```

```

##      RawPace_M sex section PiS  TiS PiD TiD
## 1 5.566667    W     10M   9 2358   4 559
## 2 5.616667    W     10M  11 2358   2 529
## 3 5.733333    W     10M  12 2358   2 387
## 4 5.733333    W     10M  13 2358   3 387
## 5 5.783333    W     10M  14 2358   3 529
## 6 5.816667    W     10M  15 2358   4 529
## 7 5.866667    W     10M  16 2358   5 529
## 8 5.966667    W     10M  18 2358   5 559
## 9 5.983333    W     10M  19 2358   4 387
## 10 6.050000   W     10M  20 2358   6 529

```

```

Descriptions = c('Participant Age', 'Year of race', 'Participant Home Town', 'Participant Home State',
womensTableInfo = data.frame(Feature = names(womens_table_T), Description = Descriptions , Type = sapply(
womensTableInfo

```

	Feature	Description	Type
## 1	Age	Participant Age	double
## 2	year	Year of race	integer
## 3	HomeTown	Participant Home Town	character
## 4	HomeState	Participant Home State	character
## 5	HomeCountry	Participant Home Country	character
## 6	RawTime_S	Participant's Time in Seconds	double
## 7	RawTime_M	Participant's Time in Minutes	double
## 8	RawPace_S	Participants Mile Pace in Seconds	double
## 9	RawPace_M	Participants Mile Pace in Minutes	double
## 10	sex	Gender	character
## 11	section	Participant Race	character
## 12	PiS	Position in Sex	character
## 13	TiS	Total in Sex	character
## 14	PiD	Position in Division	character
## 15	TiD	Total in Division	character

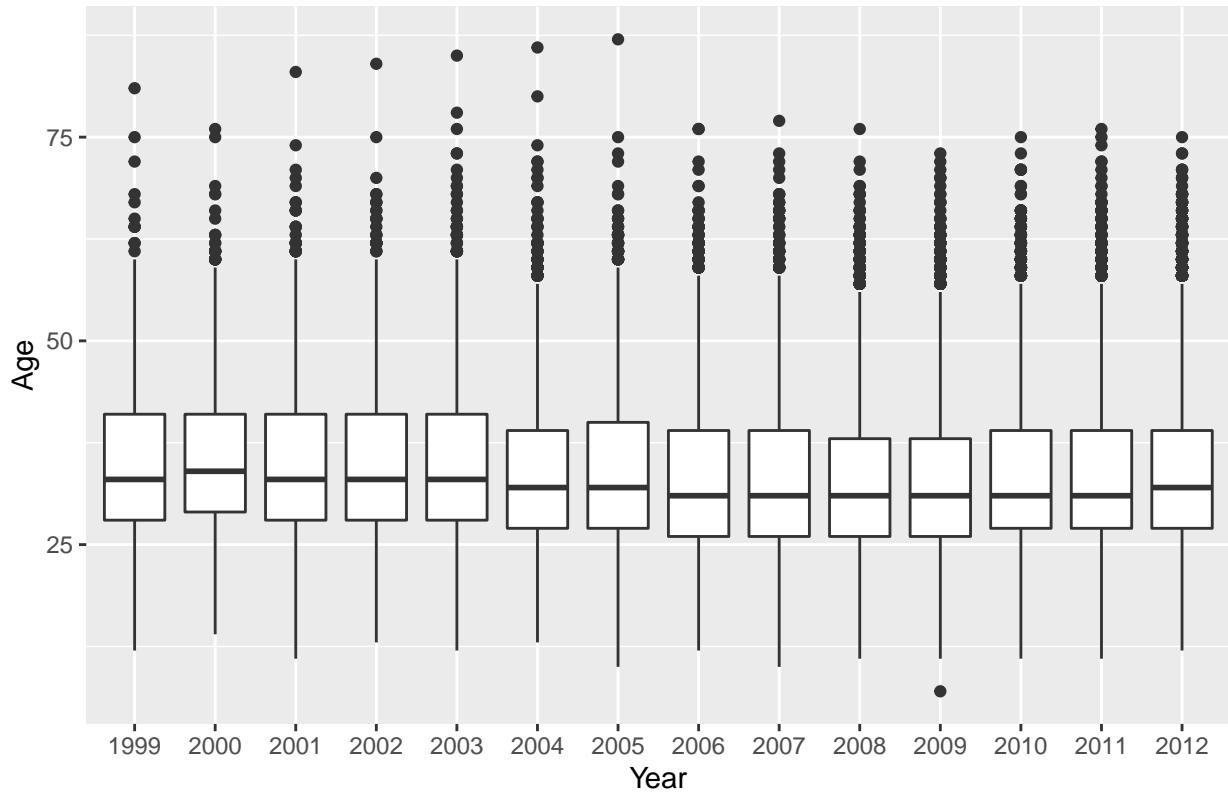
Generate plots of quick EDA for ages accross years

```

plot_data_age = ggplot(womens_table_T, aes(x = year, y = Age)) + geom_boxplot() + labs(title = "Distribution of Age by Year")
plot_data_age

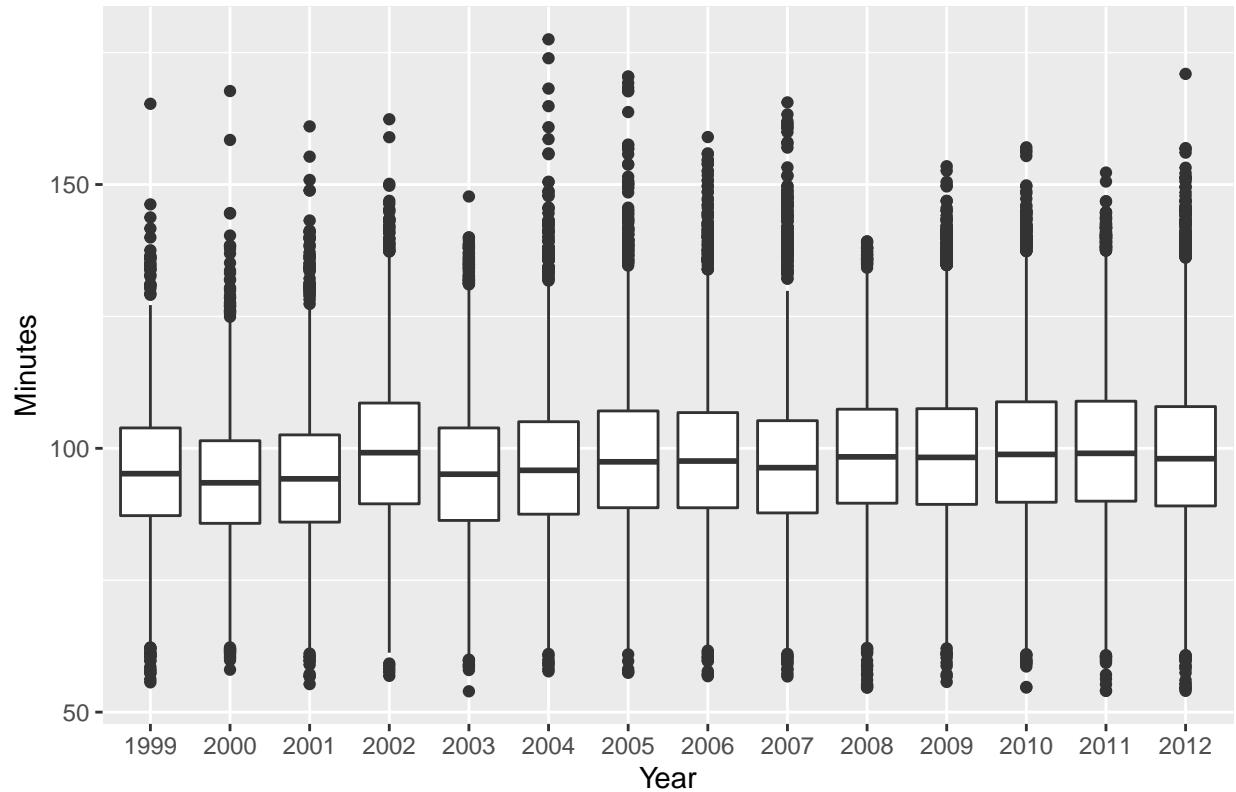
```

## Distribution of Women Participants Age by Year



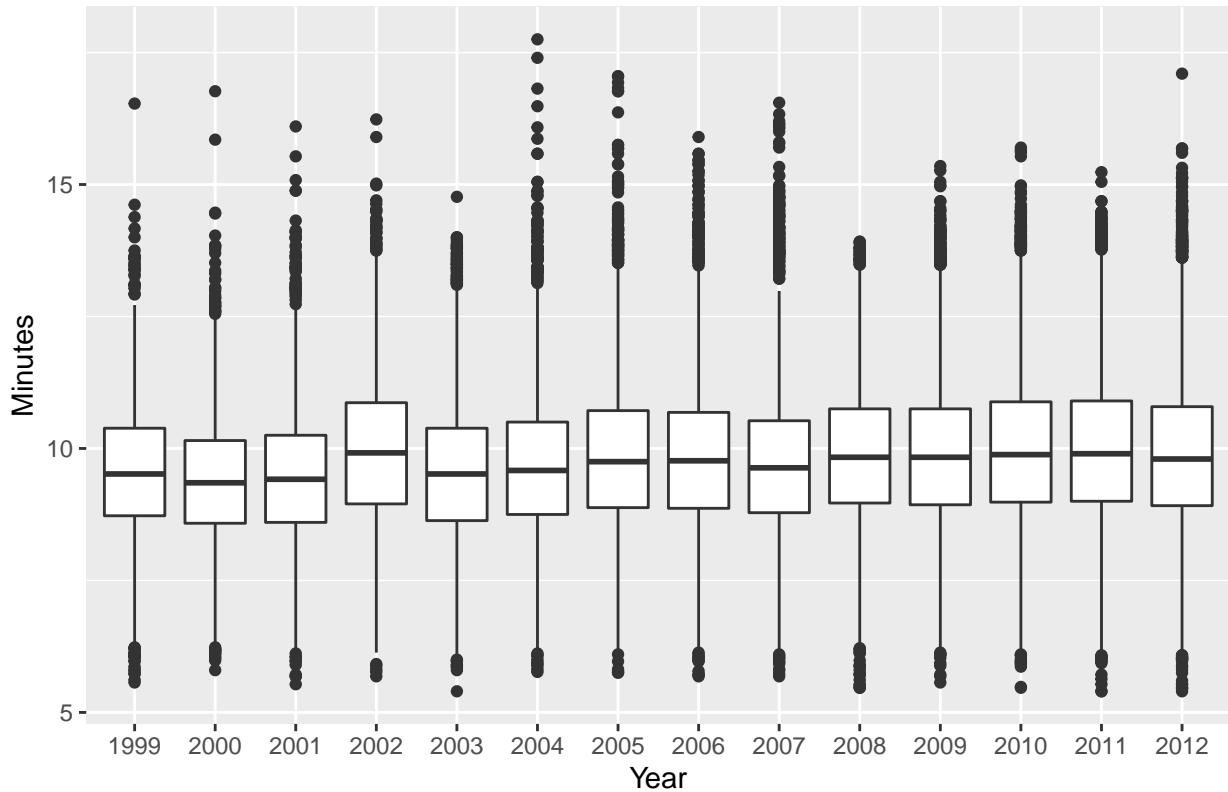
```
plot_data_time = ggplot(womens_table_T, aes(x = year, y = RawTime_M)) + geom_boxplot() + labs(title = "Women's Age Distribution by Year")  
plot_data_time
```

## Distribution of Women Raw Time by Year



```
plot_data_Pace = ggplot(womens_table_T, aes(x = year, y = RawPace_M)) + geom_boxplot() + labs(title = "Distribution of Women Raw Time by Year")  
  
plot_data_Pace
```

## Distribution of Women Raw Pace by Year



```
BRIAN GAITHER #####
Creating an age bin column for analysis
```

```
womens_table_T$AgeBin = cut(womens_table_T$Age, breaks=c(0,5,15,25,35,45,55,65,75,85,95),labels=c("1-5"
```

Creating a column to bin the pace in minutes for later analysis

```
womens_table_T$PaceBin = cut(womens_table_T$RawPace_M, breaks=c(0,5,5.5,6,6.5,7,7.5,8,8.5,9,9.5,10,10.5),
                                labels=c("1-5","5.1-5.5","5.5-6","6-6.5","6.5-7","7-7.5","7.5-8","8-8.5","8.5-9","9-9.5"))
```

write out the dataframe for later retrieval

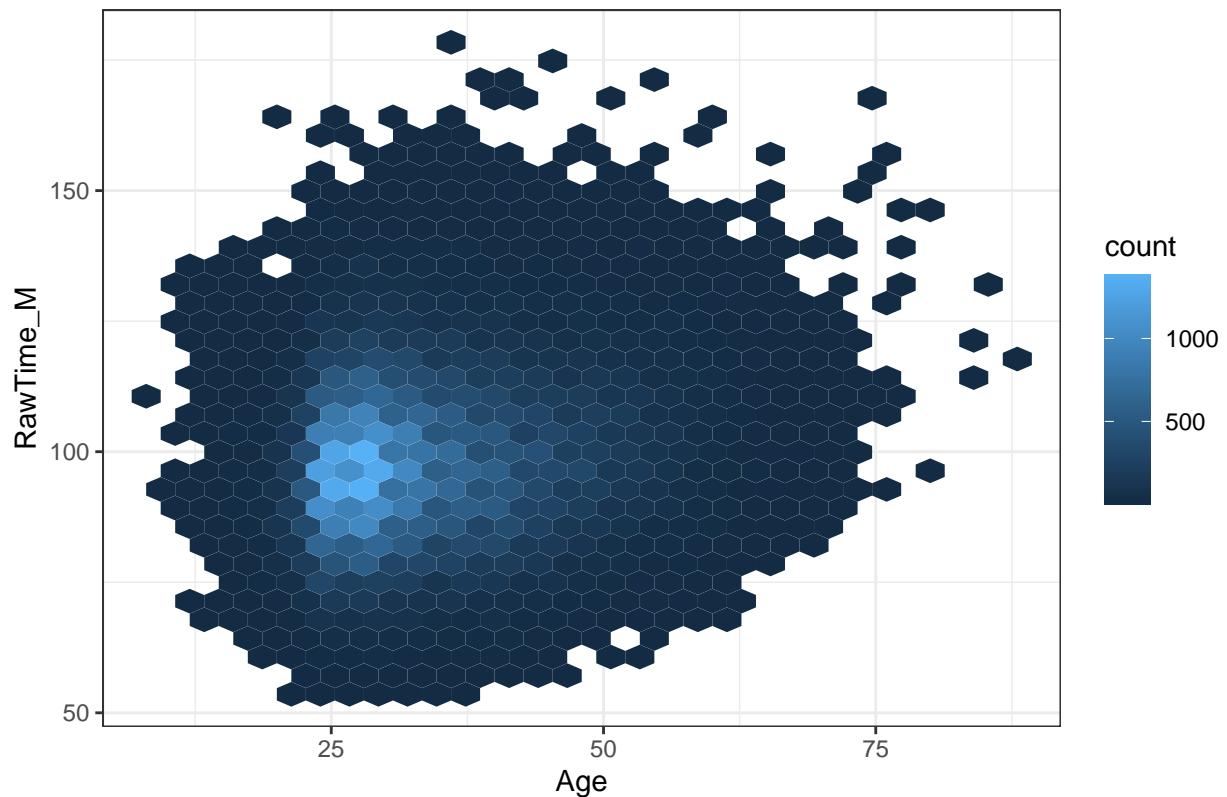
```
#write.csv(womens_table_T,"C:/Users/blgai/OneDrive/Documents/School/SMU/Spring 2021/Quantifying the Wor
```

read the data into dataframe

```
#womens_table_T = read.csv("C:/Users/blgai/OneDrive/Documents/School/SMU/Spring 2021/Quantifying the Wor
```

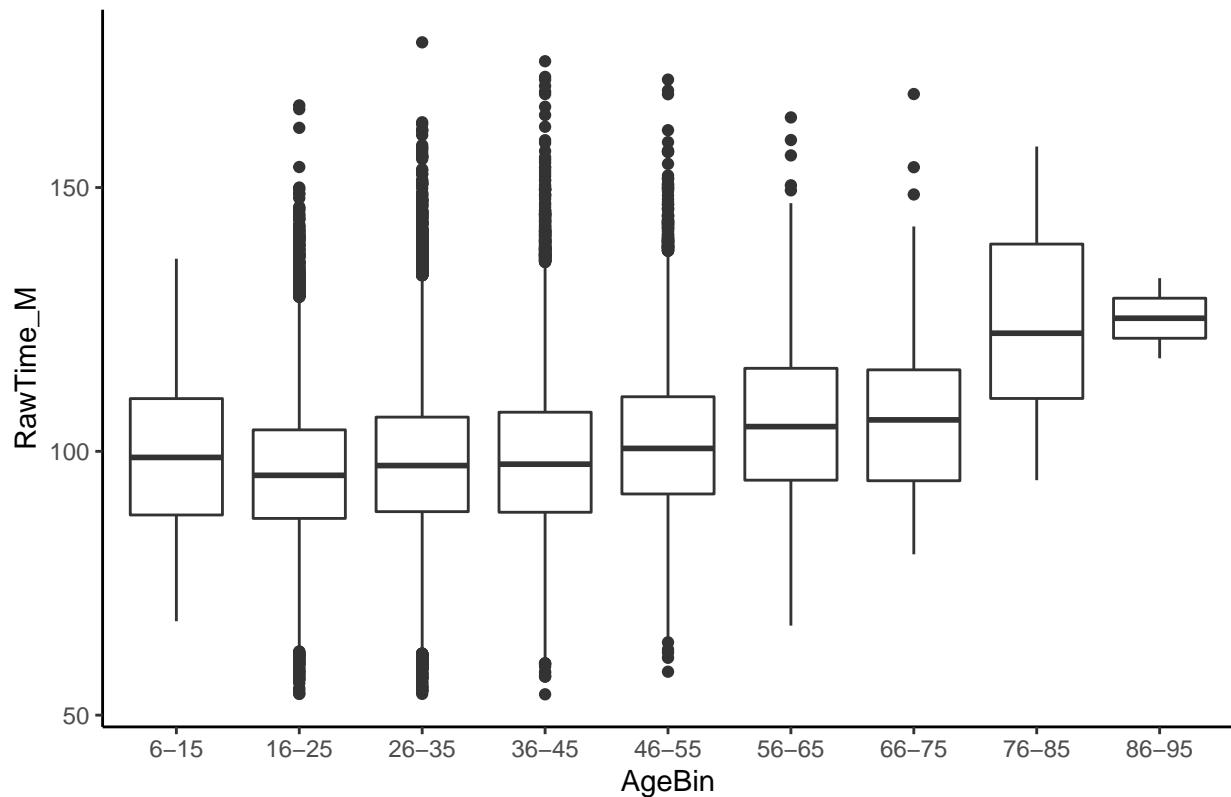
```
ggplot(womens_table_T, aes(x = Age, y=RawTime_M)) +
  geom_hex() +
  theme_bw() +
  labs(title="Scatter Plot of Female Runners: Age vs Time in Minutes")
```

Scatter Plot of Female Runners: Age vs Time in Minutes



```
f = ggplot(womens_table_T)
f + geom_boxplot(mapping = aes(x=AgeBin, y=RawTime_M)) + theme_classic() + labs(title="Boxplot of Age G")
```

Boxplot of Age Group by Running Time (Minutes)



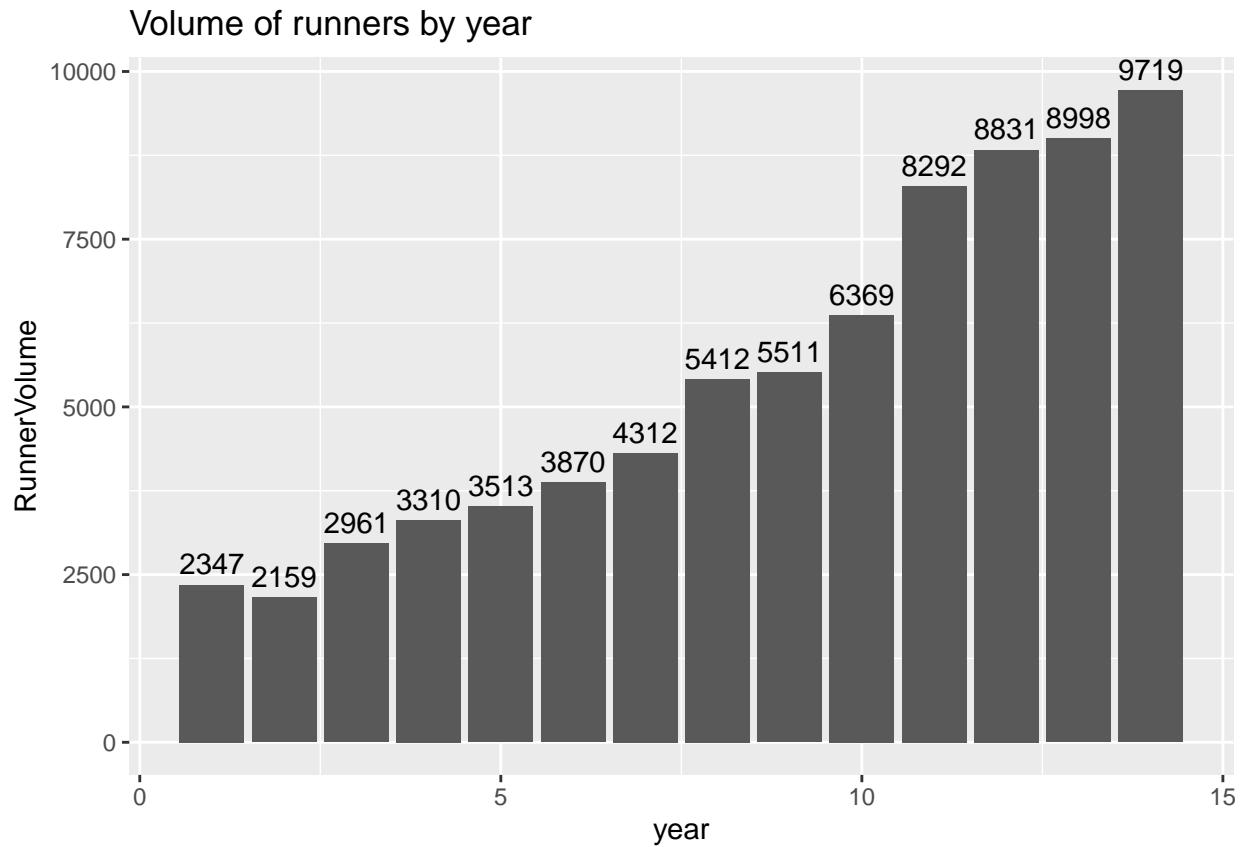
Let's examine overall runner volume

```
dfRunVol = womens_table_T %>% group_by(year) %>% tally()
dfRunVol$year = as.integer(dfRunVol$year)
names(dfRunVol)[2] <- "RunnerVolume"
head(dfRunVol)
```

```
## # A tibble: 6 x 2
##   year  RunnerVolume
##   <int>     <int>
## 1     1      2347
## 2     2      2159
## 3     3      2961
## 4     4      3310
## 5     5      3513
## 6     6      3870
```

Plotting out volume of runners by year

```
ggplot(dfRunVol, aes(x=year, y=RunnerVolume)) +
  geom_col() +
  labs(title="Volume of runners by year") +
  geom_text(aes(label = RunnerVolume), vjust = -0.5)
```



let's see how well we can predict the volume of racers in the 15th year

```
lmRunVol = lm(RunnerVolume ~ year, data = dfRunVol)
lmRunVol$coefficients
```

```
## (Intercept)      year
##   800.1978    613.3451
```

Below we see we have a statistically significant slope and intercept

```
summary(lmRunVol)
```

```
##
## Call:
## lm(formula = RunnerVolume ~ year, data = dfRunVol)
##
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -809.30 -511.97   94.27  329.17  933.46 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 800.20     339.09    2.36   0.0361 *  
## year        613.35     39.82   15.40 2.87e-09 *** 
## ---
```

```

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 600.7 on 12 degrees of freedom
## Multiple R-squared:  0.9518, Adjusted R-squared:  0.9478
## F-statistic: 237.2 on 1 and 12 DF,  p-value: 2.872e-09

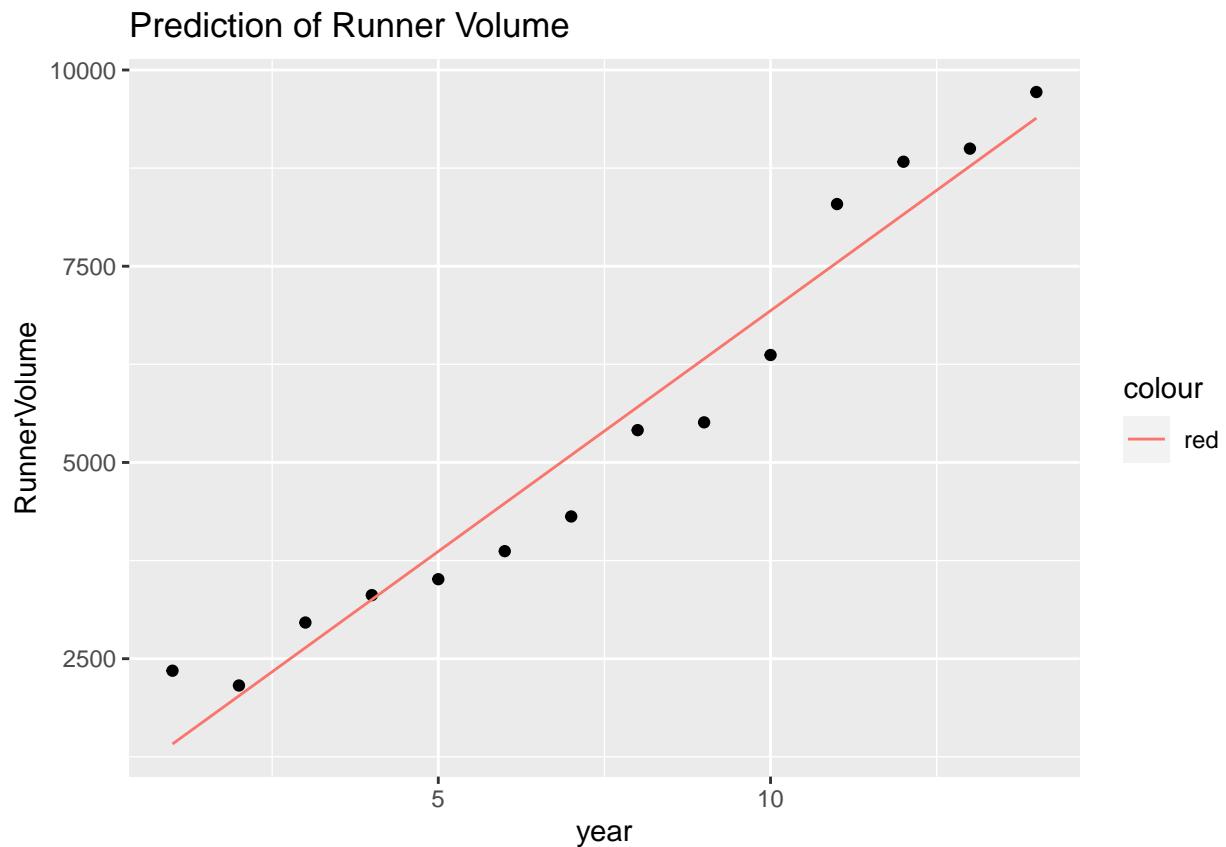
```

See how well our line fits

```

preds = predict(lmRunVol)
ggplot(dfRunVol, aes(x=year, y=RunnerVolume)) + geom_point() + geom_line(dfRunVol, mapping = aes(x=year

```



How many runners are predicted to attend in 2013 (year 15)

```

y_hat = 800.5 + (15*613.4)

print(paste0("Predicted volume of runners in 2013 is: ", y_hat))

```

```

## [1] "Predicted volume of runners in 2013 is: 10001.5"

```

95% confidence intervals

```

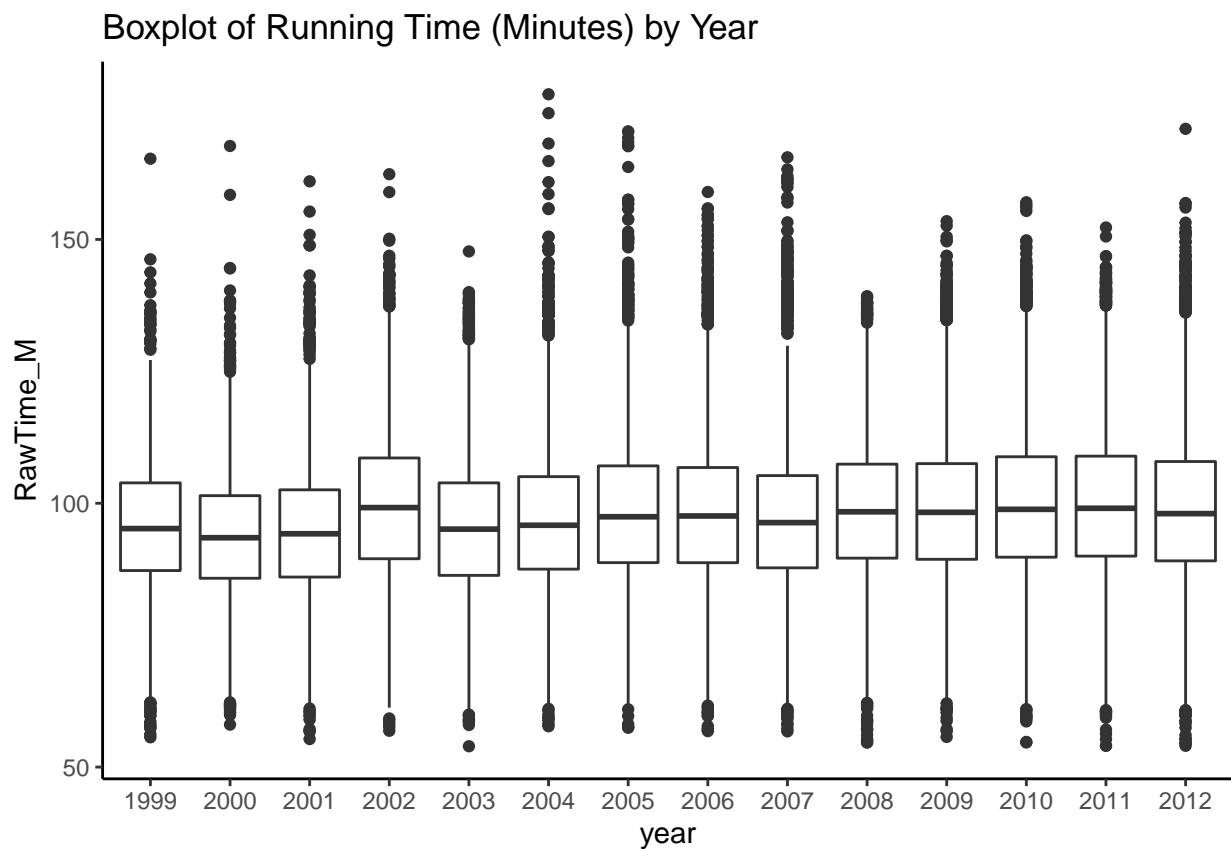
dfNew = data.frame(year=15)
predict(lmRunVol, newdata = dfNew, interval = 'confidence')

```

```
##      fit      lwr      upr
## 1 10000.37 9261.564 10739.18
```

we can see below that the IQR of running times throughout the years has remained relatively constant. That means that

```
f = ggplot(womens_table_T)
f + geom_boxplot(mapping = aes(x=year, y=RawTime_M)) + theme_classic() + labs(title="Boxplot of Running
```

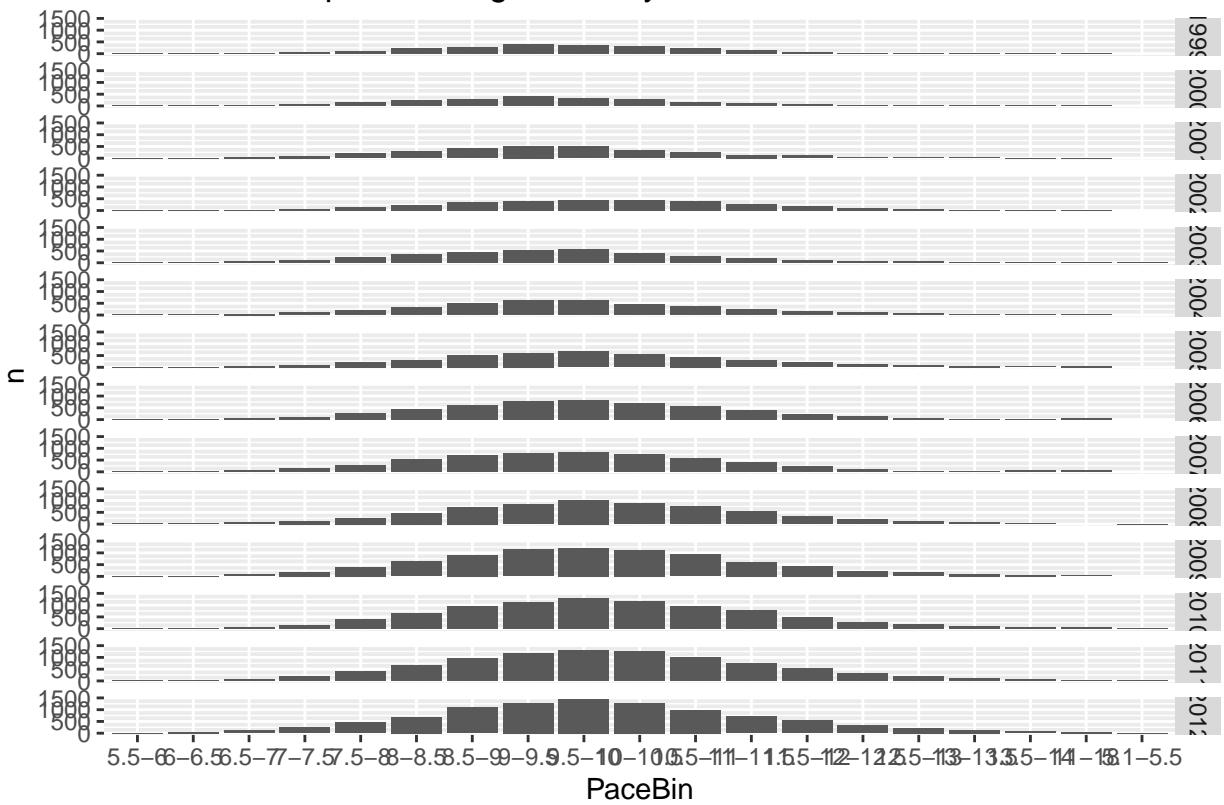


Let's look at the volume of runners by pace bin. We can see that the majority of runners are between 8-11 minute pace. This means that a large swell of runners will be running through the course together and race support must have enough volunteers to support the increase of runners as they make their way through the course.

```
womens_table_T$PaceBin <- as.character(womens_table_T$PaceBin)
womens_table_T$PaceBin <- factor(womens_table_T$PaceBin, levels=unique(womens_table_T$PaceBin))

ggplot(womens_table_T %>% group_by(year, PaceBin) %>% tally()) +
  geom_col(mapping = aes(x=PaceBin, y=n)) +
  facet_grid(vars(year)) +
  labs(title="Distribution of pace throughout the years")
```

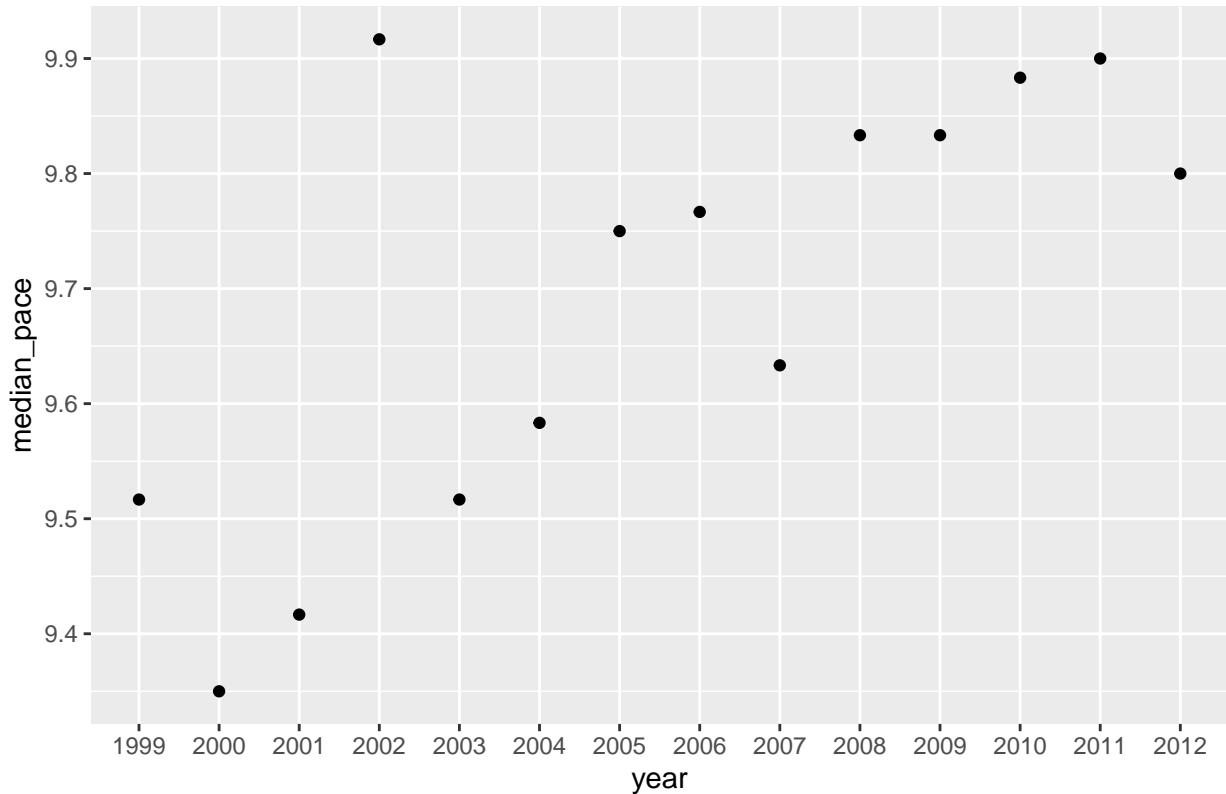
## Distribution of pace throughout the years



Let's take a look at the median pace per year.

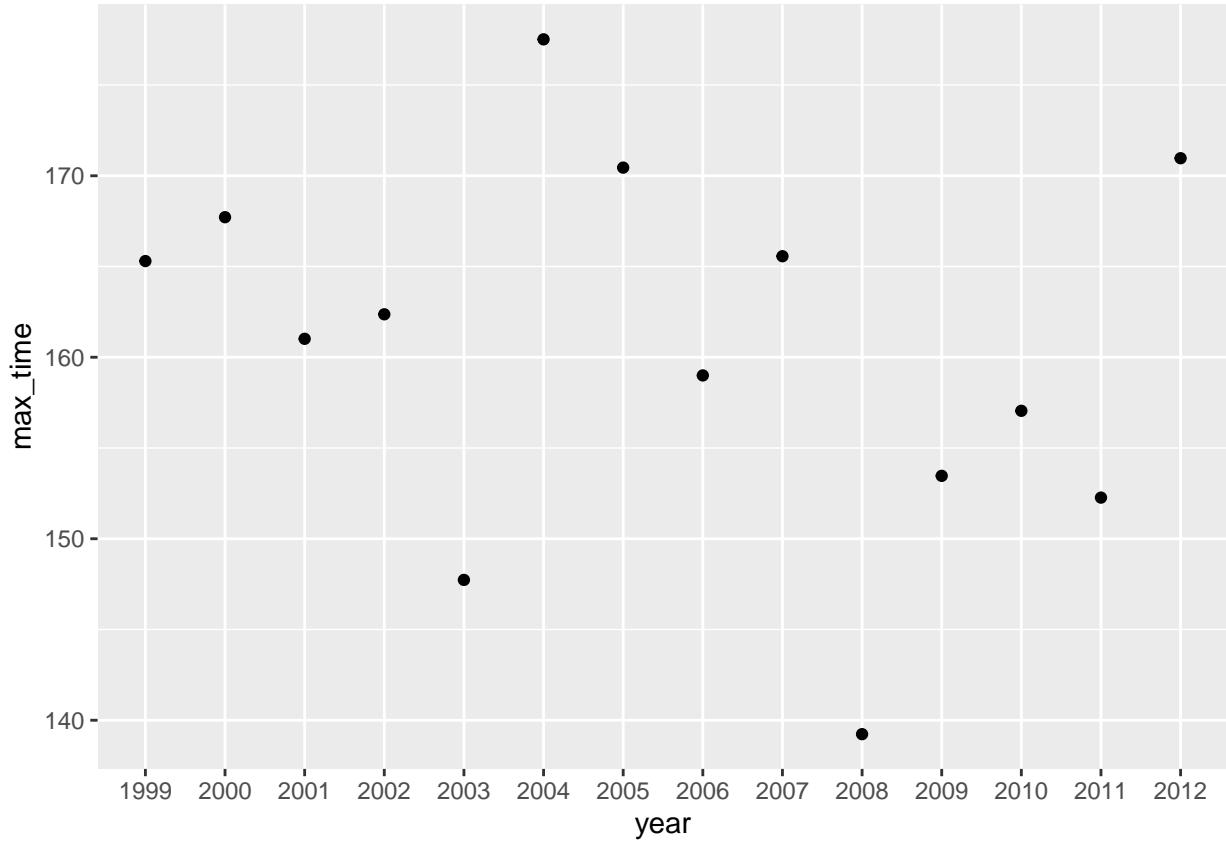
```
womens_table_T %>% group_by(year) %>% dplyr::summarise(median_pace = median(RawPace_M)) %>% ggplot(aes(  
## `summarise()` ungrouping output (override with `groups` argument)
```

## Median Pace Per Year



If we look at the max times throughout the years, there isn't a consistent pattern observable, we'll revisit this after looking more at the age bins further

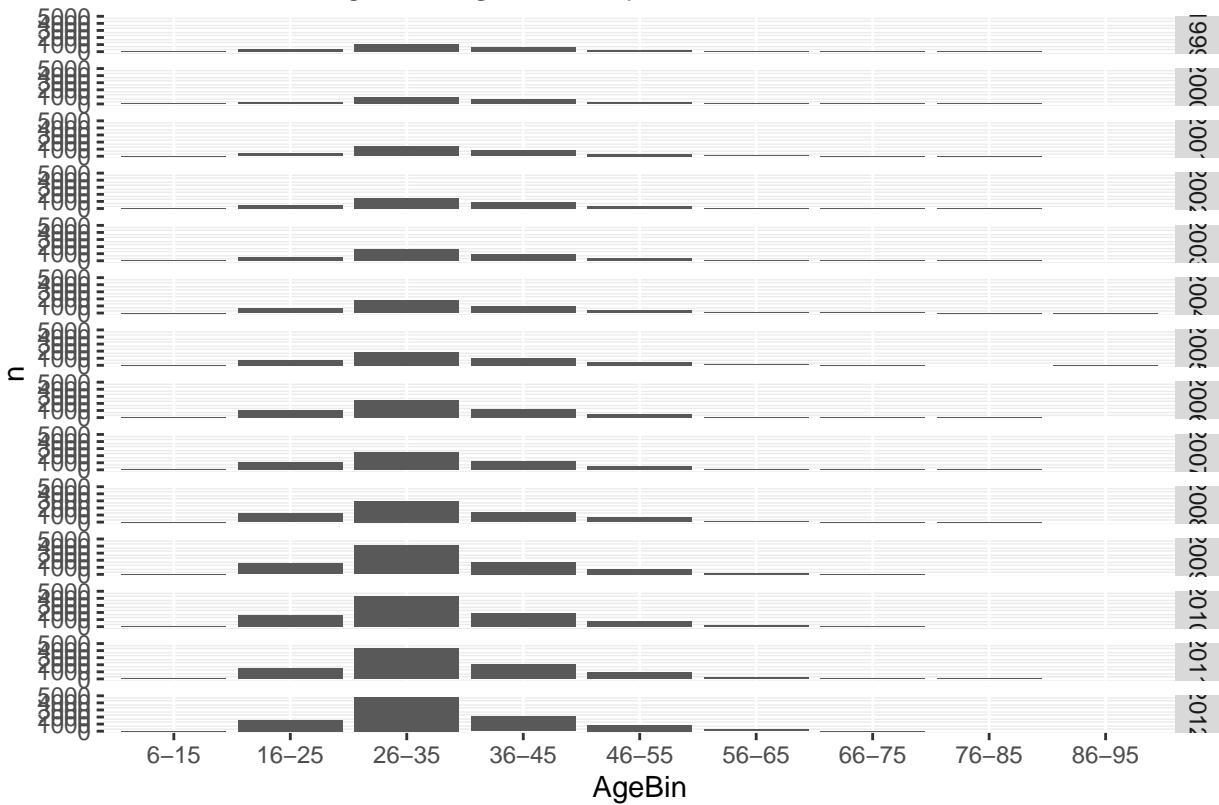
```
womens_table_T %>% group_by(year) %>% dplyr::summarise(max_time = max( RawTime_M )) %>% ggplot(aes(x=yea  
## `summarise()` ungrouping output (override with `.`groups` argument)
```



we see below that there has been a steady increase in 26-35 year old runners over the years

```
ggplot(womens_table_T %>% group_by(year, AgeBin) %>% tally()) +
  geom_col(mapping = aes(x=AgeBin, y=n)) +
  facet_grid(vars(year)) +
  labs(title="Distribution of age throughout the years")
```

## Distribution of age throughout the years



Let's take a closer look at the volume of runners in the 26-35 age category

```
dfRunAgeBinVol = womens_table_T %>% group_by(year, AgeBin) %>% dplyr::summarise(AgeBinVolume = n())

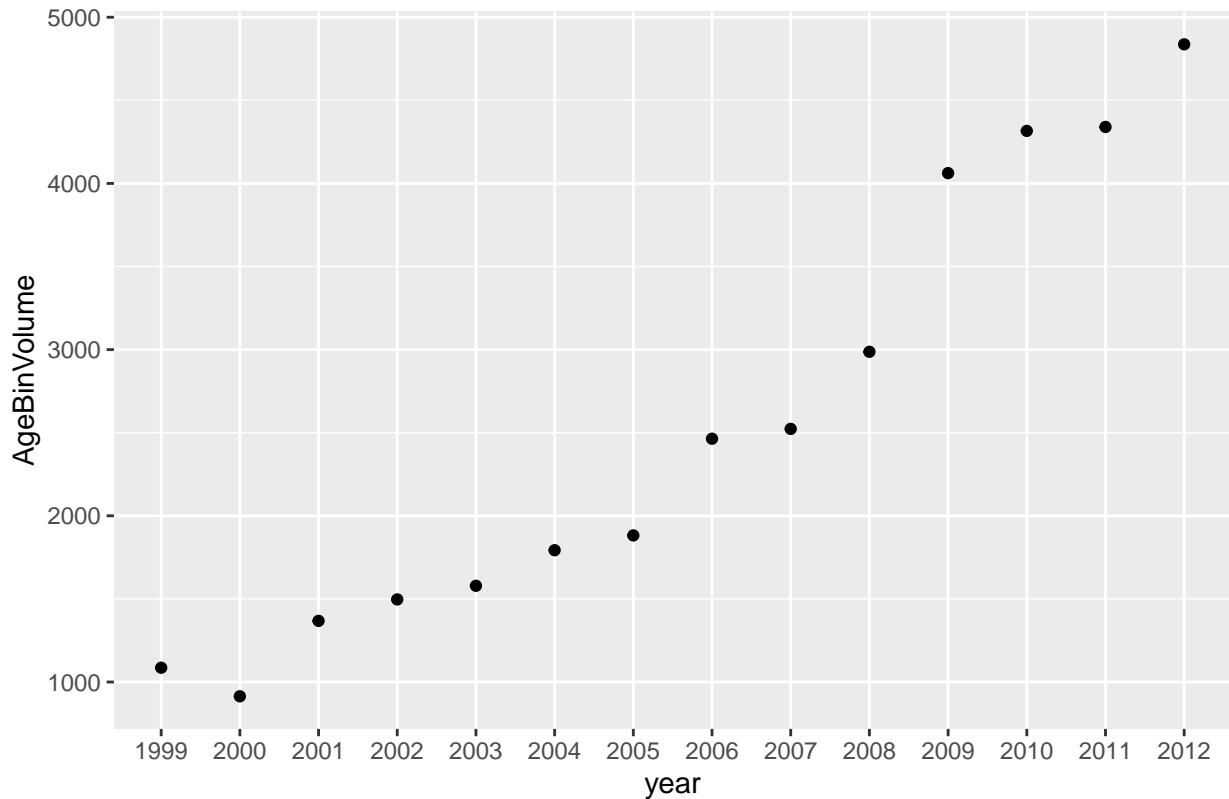
## `summarise()` regrouping output by 'year' (override with `.`groups` argument)

head(dfRunAgeBinVol)

## # A tibble: 6 x 3
## # Groups:   year [1]
##   year   AgeBin AgeBinVolume
##   <fct> <fct>     <int>
## 1 1999   6-15        5
## 2 1999   16-25      286
## 3 1999   26-35     1086
## 4 1999   36-45      649
## 5 1999   46-55      273
## 6 1999   56-65       43

ggplot(dfRunAgeBinVol[dfRunAgeBinVol$AgeBin == '26-35',]) +
  geom_point(mapping = aes(x=year, y=AgeBinVolume)) +
  #facet_wrap(vars(year)) +
  labs(title="Increase in volume of 26-35 age group throughout the years")
```

### Increase in volume of 26–35 age group throughout the years



```
dfRunAgeBinVol$year = as.numeric(as.character(dfRunAgeBinVol$year))
dfRunAgeBinVol[dfRunAgeBinVol$AgeBin == '26-35',]
```

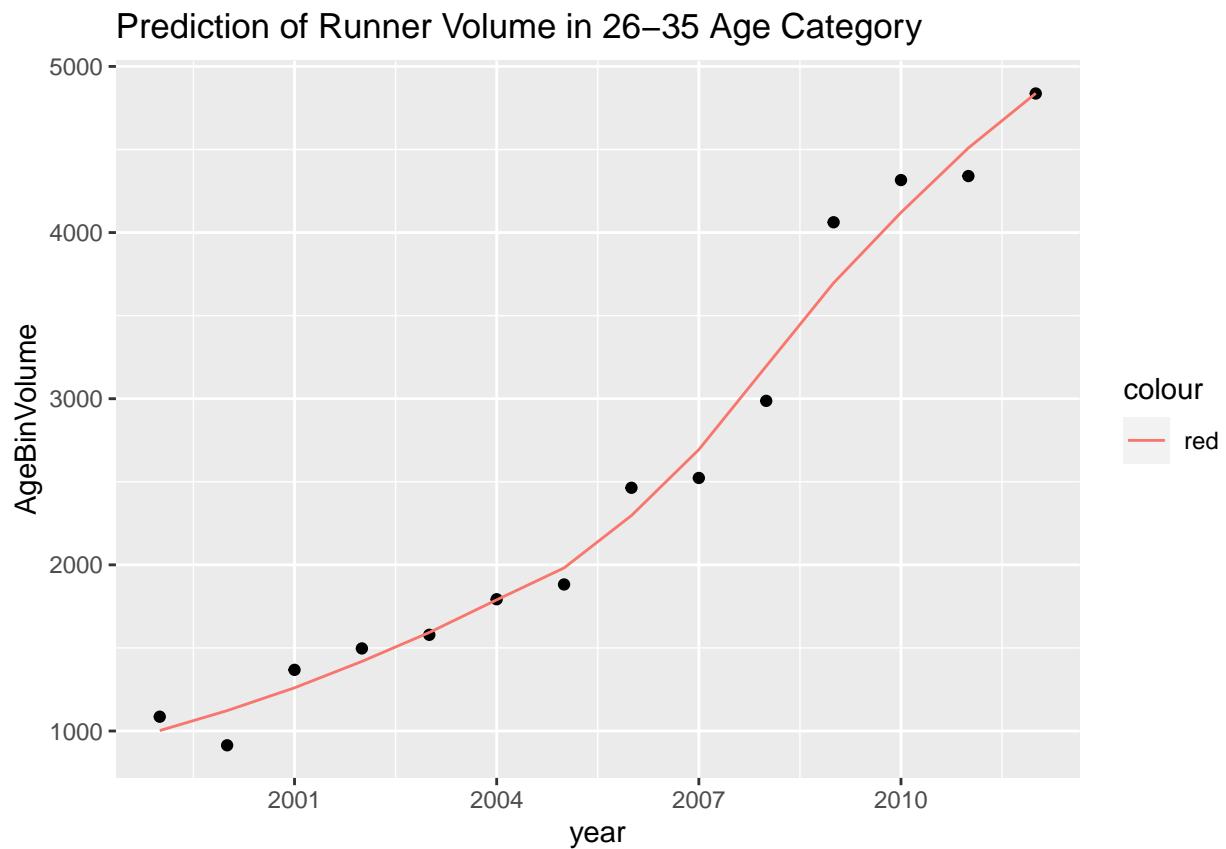
```
## # A tibble: 14 x 3
## # Groups:   year [14]
##   year AgeBin AgeBinVolume
##   <dbl> <fct>     <int>
## 1 1999 26-35      1086
## 2 2000 26-35      914
## 3 2001 26-35     1368
## 4 2002 26-35     1497
## 5 2003 26-35     1579
## 6 2004 26-35     1793
## 7 2005 26-35     1882
## 8 2006 26-35     2464
## 9 2007 26-35     2523
## 10 2008 26-35     2987
## 11 2009 26-35     4062
## 12 2010 26-35     4316
## 13 2011 26-35     4340
## 14 2012 26-35     4837
```

Let's trend the volume of 26-35 year old runners using LOESS

```

df26 = dfRunAgeBinVol[dfRunAgeBinVol$AgeBin == '26-35',]
loessAge26 = loess(AgeBinVolume ~ year, data=df26)
preds = predict(loessAge26)
ggplot(df26, aes(x=year, y=AgeBinVolume)) +
  geom_point() +
  geom_line(df26, mapping = aes(x=year, y=preds, col = "red")) +
  ggtitle("Prediction of Runner Volume in 26-35 Age Category")

```



Let's now try to predict the volume of 26-35 year olds using a piece wise linear regression model

```

pw.model = piecewise.linear(df26$year, df26$AgeBinVolume, middle=1, CI=TRUE, sig.level = 0.05)
pw.model

```

```

## [1] "Threshold alpha: 2005.0000564959"
## [1] ""
## [1] "Model coefficients: Beta[0], Beta[1], Beta[2]"
## (Intercept)          x          w
## -311648.2551    156.3910   276.1541
##      Change.Point Initial.Slope Slope.Change Second.Slope
## 2.5%       2003.315     81.99722    192.6359    380.4800
## 97.5%      2007.554    242.51225    579.9987    684.4968

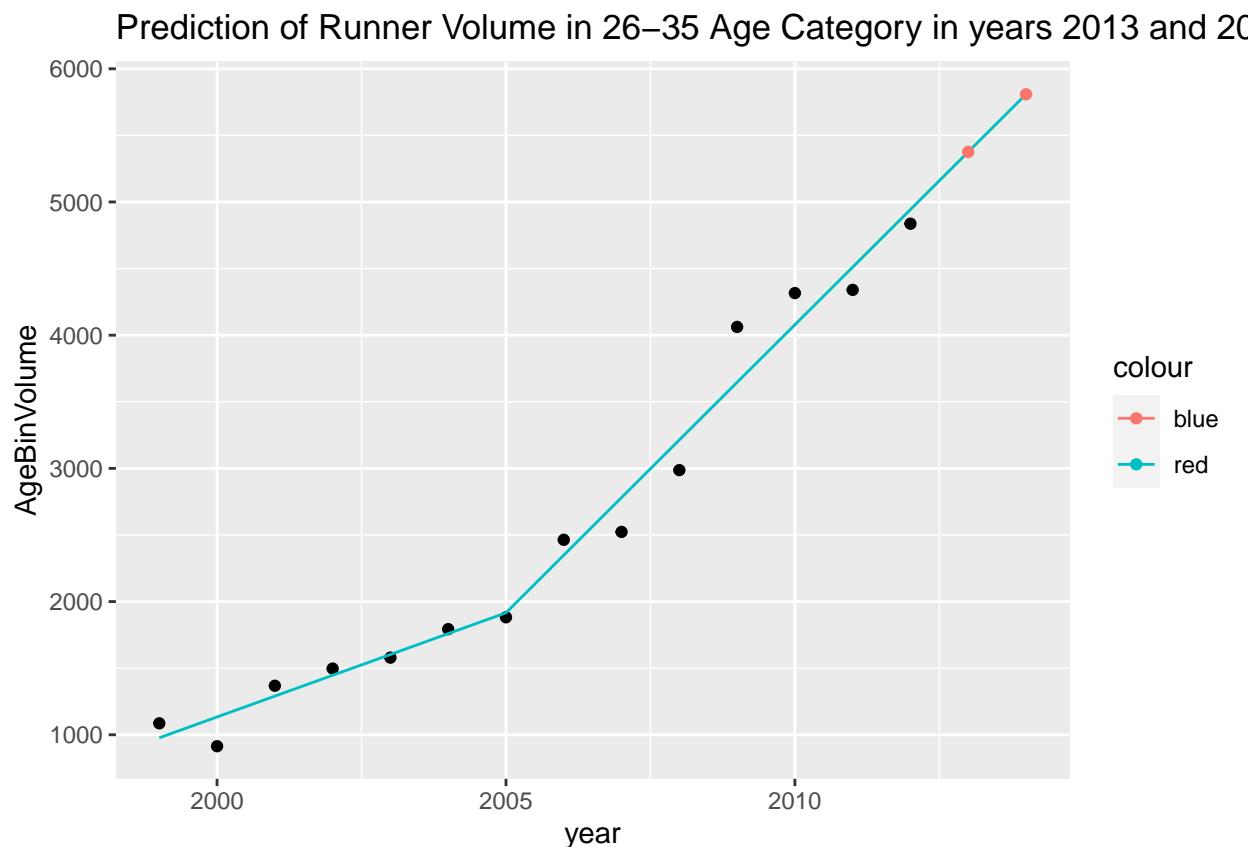
```

Now that we have a piece wise linear model, let's predict the volume of 26-35 year olds in years 2013 and 2014

```

preds = predict(pw.model, c(1999:2014))
predYears = c(1999:2014)
dfPreds = data.frame(year = predYears, predAgeBinVolume = predict(pw.model,predYears))
ggplot(df26, aes(x=year, y=AgeBinVolume)) +
  geom_point() +
  geom_line(dfPreds, mapping = aes(x=year, y=predAgeBinVolume, col = "red")) +
  geom_point(dfPreds[dfPreds$year %in% c(2013:2014), ], mapping = aes(x=year, y=predAgeBinVolume, col =
  ggttitle("Prediction of Runner Volume in 26-35 Age Category in years 2013 and 2014")

```



The actual predicted values for years 2013 and 2014. This is what we want to share with the race management team

```
predict(pw.model, c(2013:2014))
```

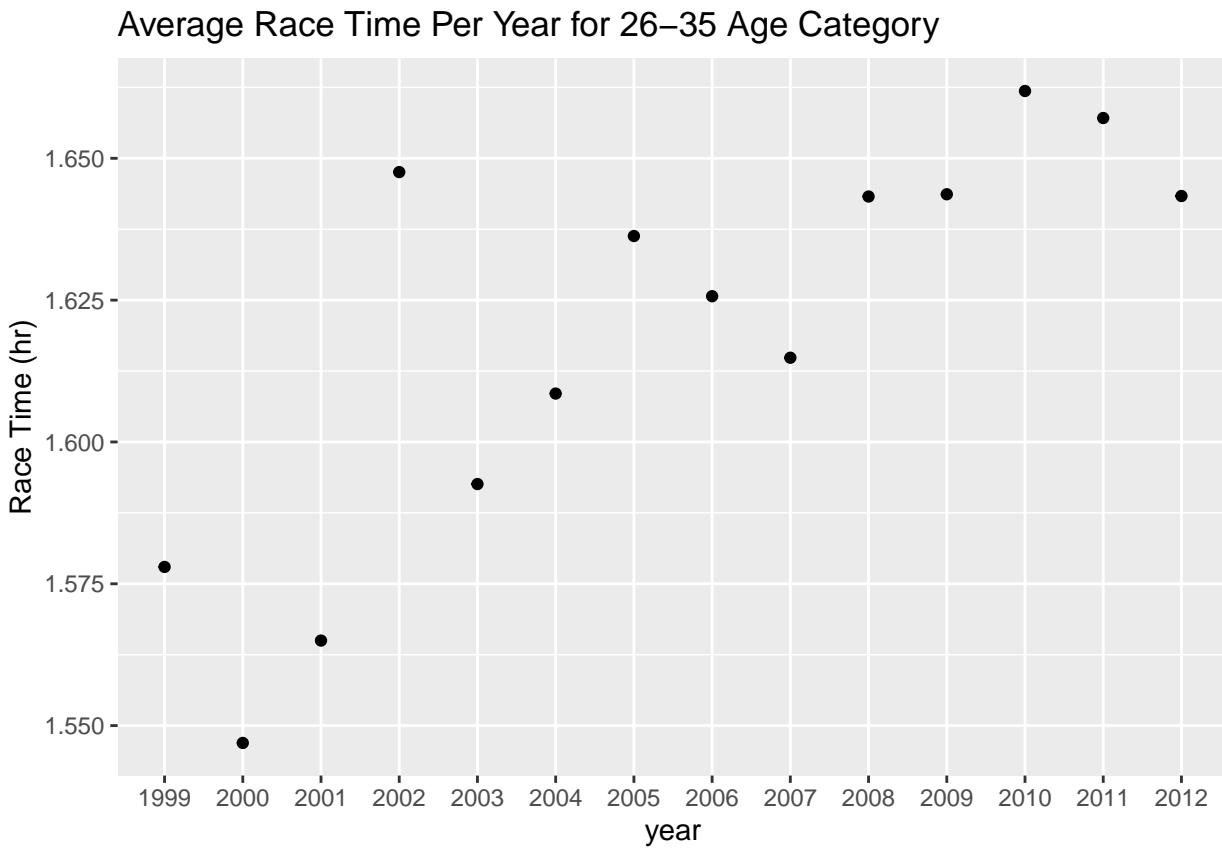
```
## [1] 5376.135 5808.681
```

Let's look at average times of age 25-36 year olds

```
dfAvgTimes26 = womens_table_T %>% group_by(year, AgeBin) %>% dplyr::summarise(avgTime = mean(RawTime_M))

## `summarise()` regrouping output by 'year' (override with '.groups' argument)
```

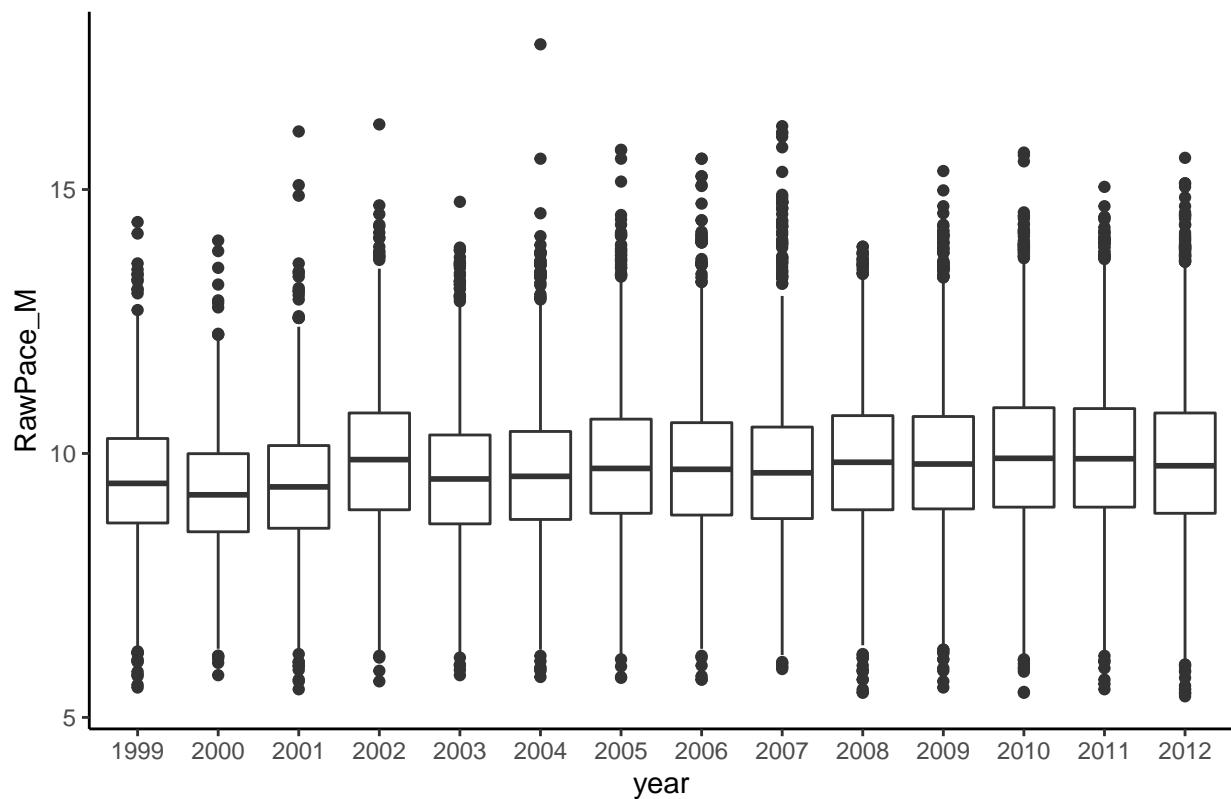
```
dfAvgTimes26 = dfAvgTimes26[dfAvgTimes26$AgeBin == '26-35',]
dfAvgTimes26 %>% ggplot(aes(x=year, y=avgTime/60)) + geom_point() + ggttitle("Average Race Time Per Year")
```



let's get a look at the pace of 26-35 year olds. over the years, they've held a pretty consistent pace. 50% are in the 9-10 minute pace group.

```
f = ggplot(womens_table_T %>% filter(womens_table_T$AgeBin == "26-35"))
f + geom_boxplot(mapping = aes(x=year, y=RawPace_M)) + theme_classic() + labs(title="Boxplot of Pace (Miles/Hour) by Year for 26-35 Age Group")
```

Boxplot of Pace (Minutes) by Year for 26–35 aged runners



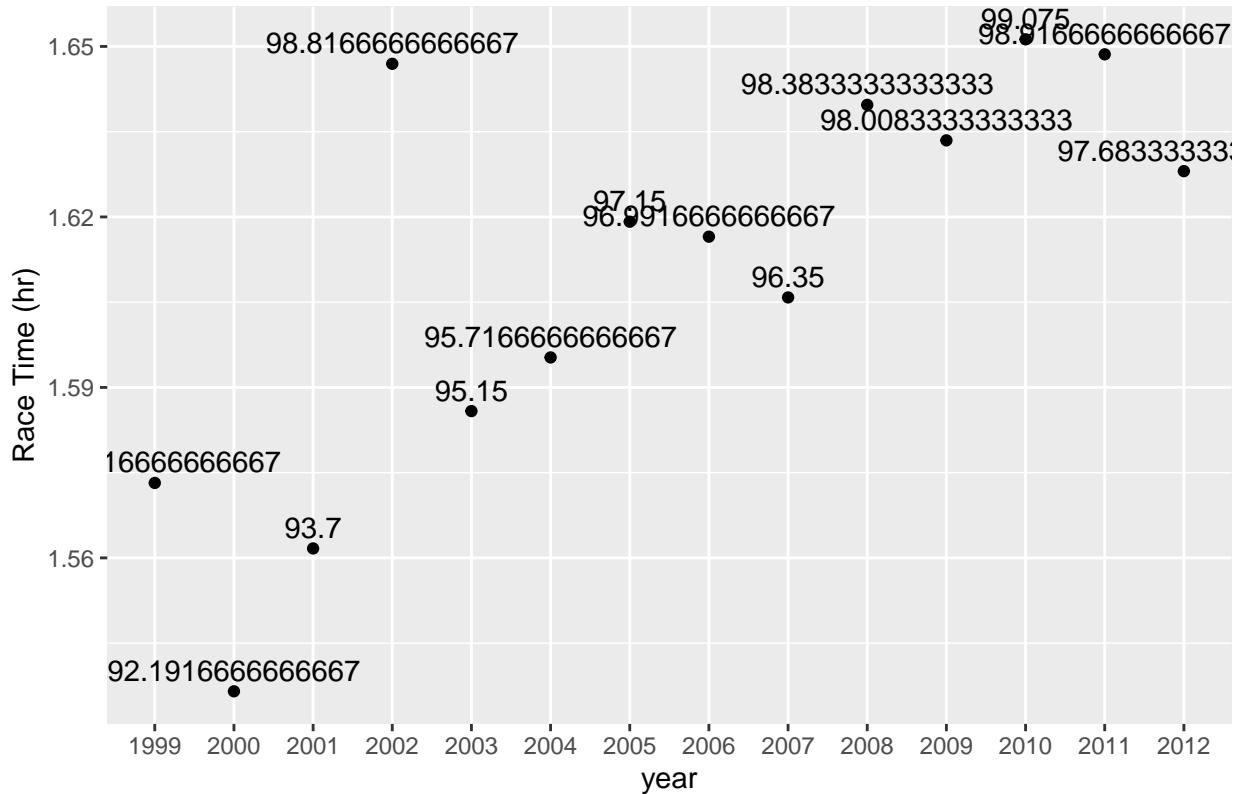
Let's look at median times of age 25-36 year olds

```
dfMedTimes26 = womens_table_T %>% group_by(year, AgeBin) %>% dplyr::summarise(medTime = median(RawTime_M))

## `summarise()`' regrouping output by 'year' (override with '.groups' argument)

dfMedTimes26 = dfMedTimes26[dfMedTimes26$AgeBin == '26-35',]
dfMedTimes26 %>% ggplot(aes(x=year, y=medTime/60)) + geom_point() + ggtitle("Median Race Time Per Year")
geom_text(aes(label = medTime), vjust = -0.5)
```

## Median Race Time Per Year for 26–35 Age Category

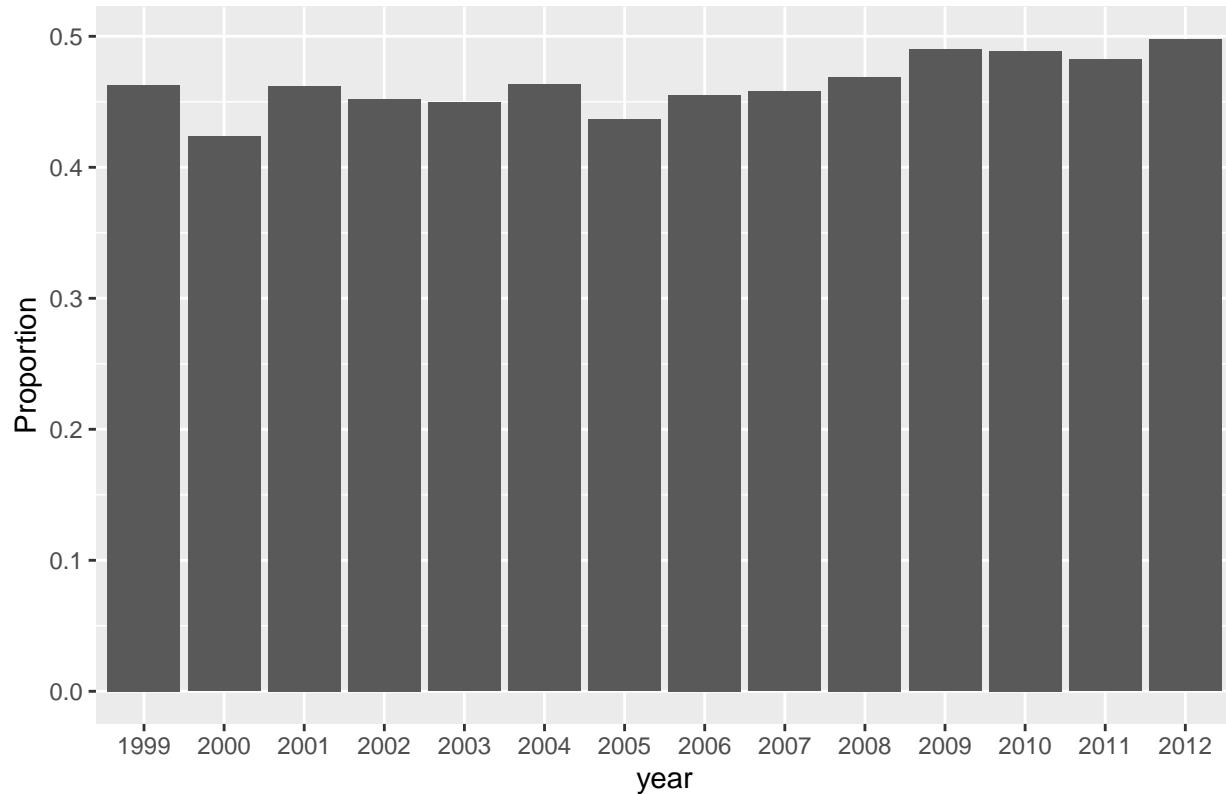


Let's look at the proportion the 26-35 year olds make up of the total runners over the years. As of 2012, this age group makes up about 50% of all runners.

```
dfAgg = womens_table_T %>% filter(womens_table_T$AgeBin == "26-35") %>% group_by(year, AgeBin) %>% dplyr::summarise(TotalRunners = n(), Proportion = AgeBinVolume / TotalRunners)

ggplot(dfAgg, aes(x=year, y=Proportion)) +
  geom_col() +
  labs(title="Proportion of 26-35 year old runners by year")
```

## Proportion of 26–35 year old runners by year

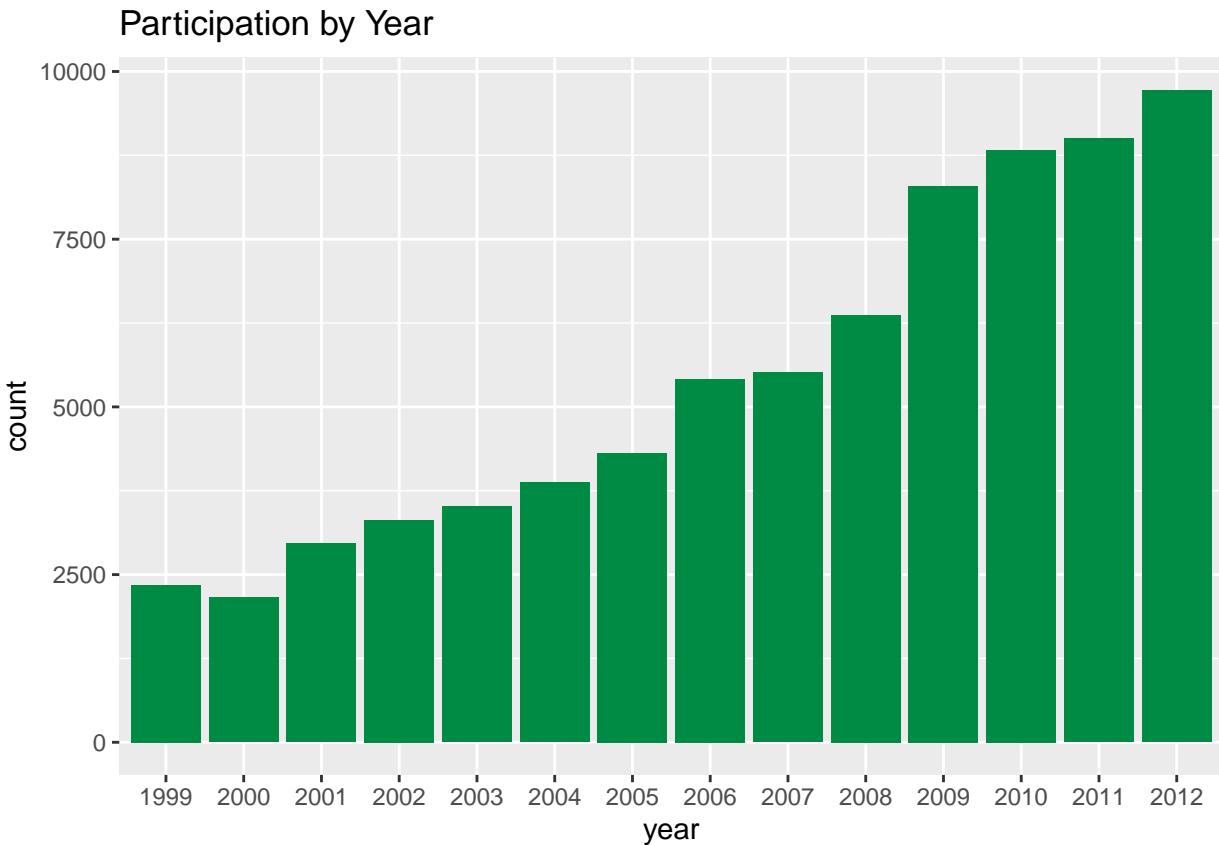


So to summarize, it's expected that 50% of the runners will be in the 26-35 year old age group in 2013. The median race time will be 97-98 minutes or approximately 9.7 minute pace.

Sean's Visuals

```
#Participation by year
#https://stackoverflow.com/questions/26114525/how-to-count-how-many-values-per-level-in-a-given-factor
library(dplyr)
womens_table_T %>% group_by(year) %>% summarise(cnt = Age) %>% ggplot(aes(x = year)) + geom_bar(fill = "#333333", color = "white", size = 1)
```

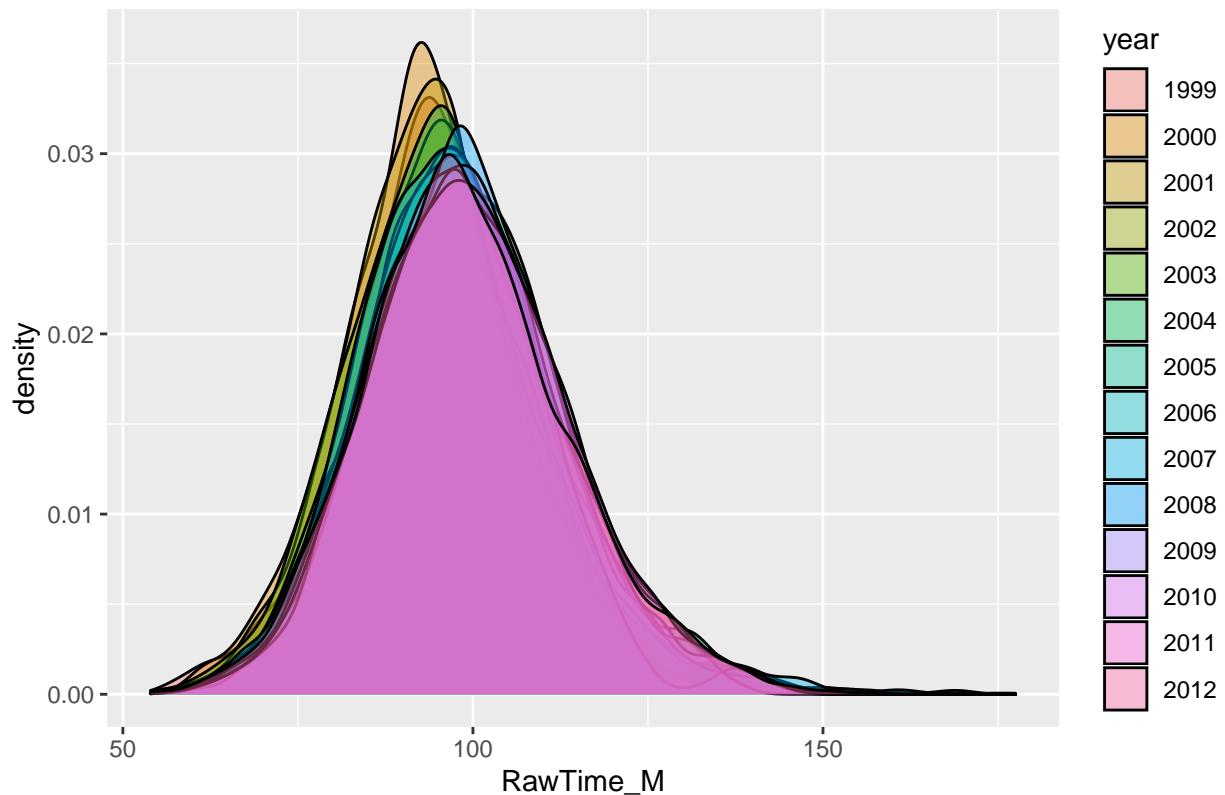
```
## `summarise()` regrouping output by 'year' (override with '.groups' argument)
```



```
#Doesn't work--Division not in the final DF
#Division count by year
#ggplot(womens_table_T, aes(Division)) + geom_bar() + facet_wrap(~year) + theme(axis.text = element_text()

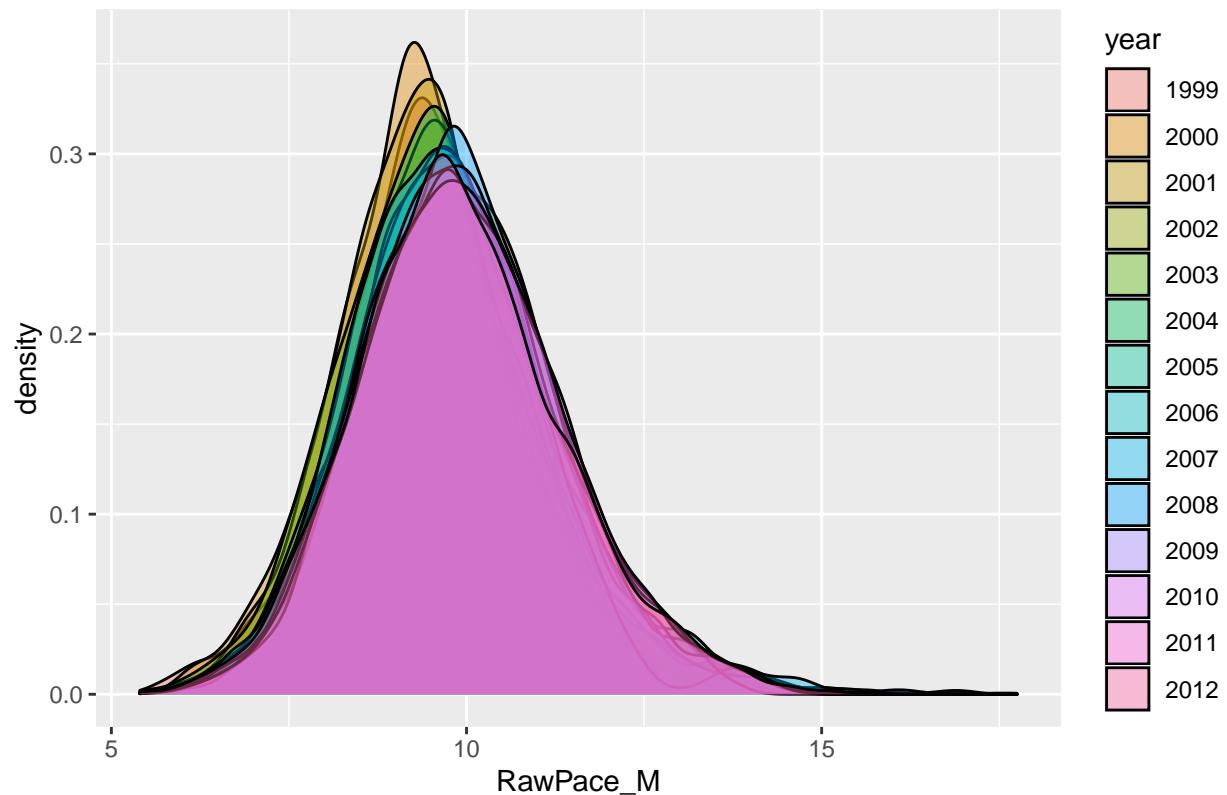
#Pace by year
ggplot(data=womens_table_T, aes(x=RawTime_M)) + geom_density(aes(fill=year), alpha=0.4) + labs(title = "Pace by Year")
```

## Female Time by Year



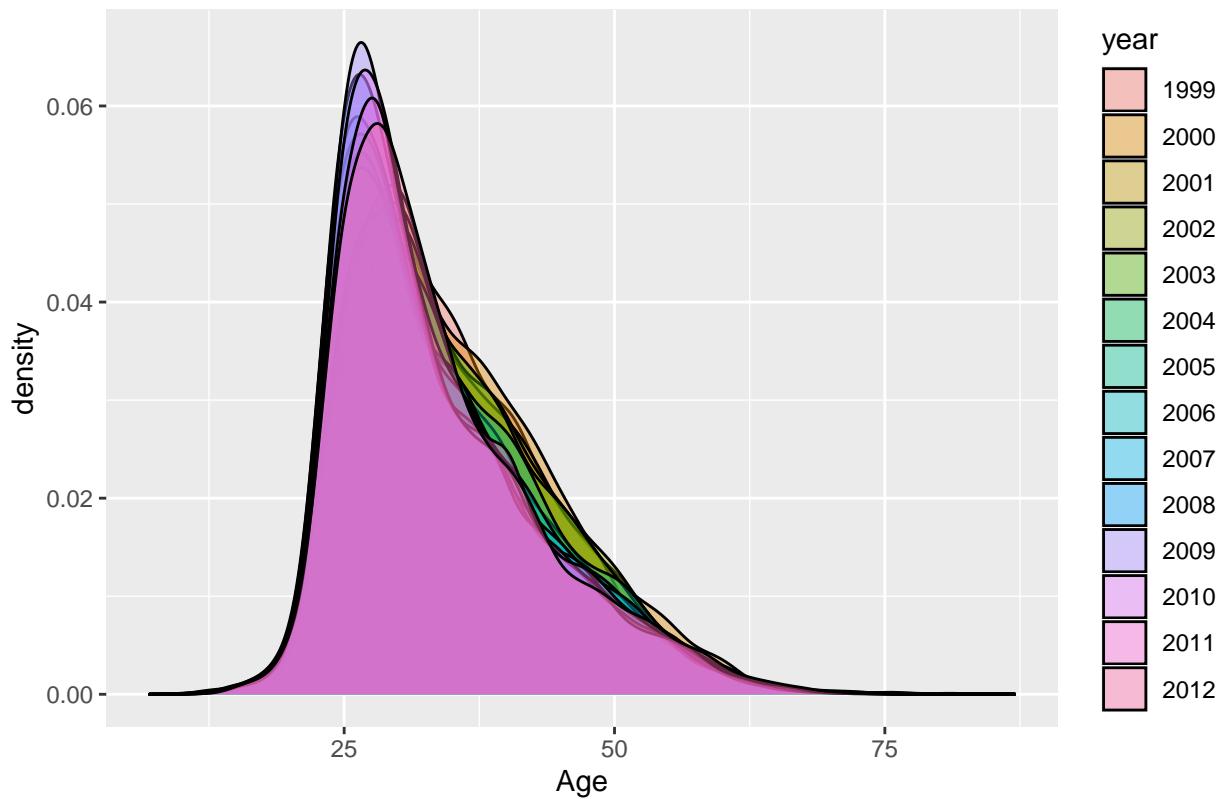
```
#Pace by year
ggplot(data=womens_table_T, aes(x=RawPace_M)) + geom_density(aes(fill=year), alpha=0.4)+ labs(title = "Pace by year")
```

## Female Pace Distributions by Year



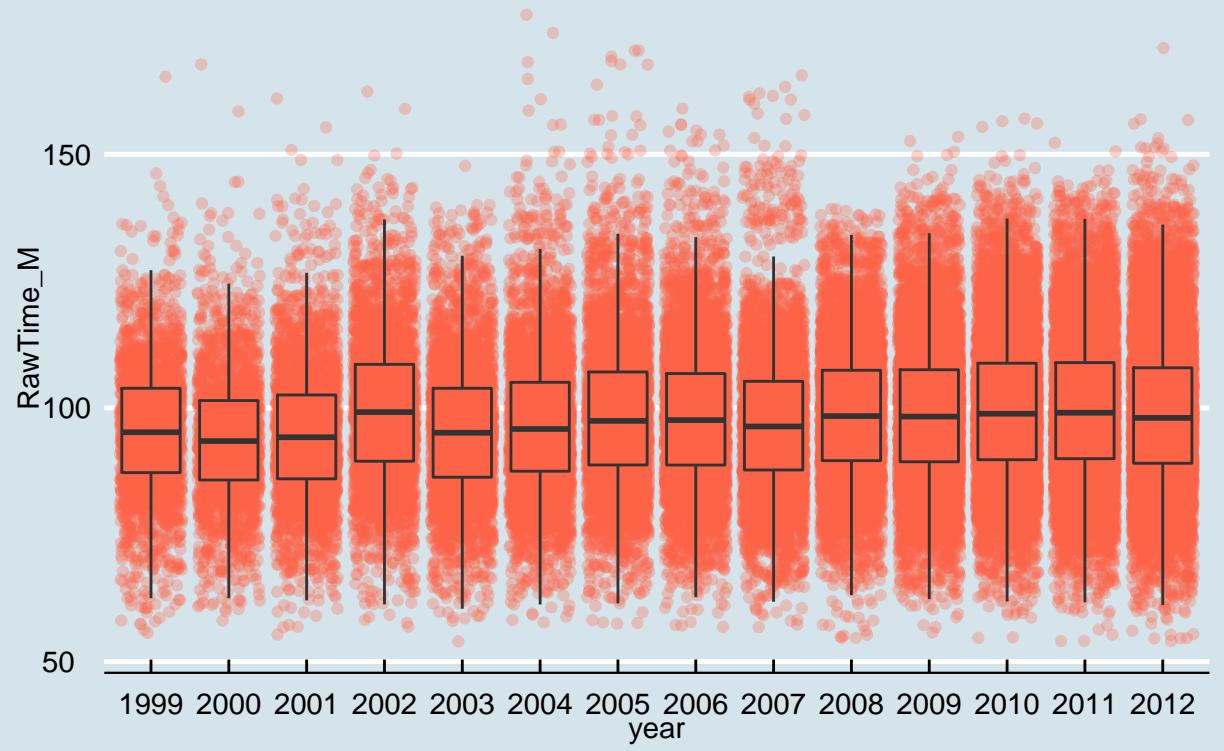
```
#Age by year  
#https://stackoverflow.com/questions/14623348/how-to-use-facet-to-create-multiple-density-plot-in-ggplot2  
ggplot(data=womens_table_T, aes(x=Age)) + geom_density(aes(fill=year), alpha=0.4) + labs(title = "Female")
```

## Female Age Distributions by Year



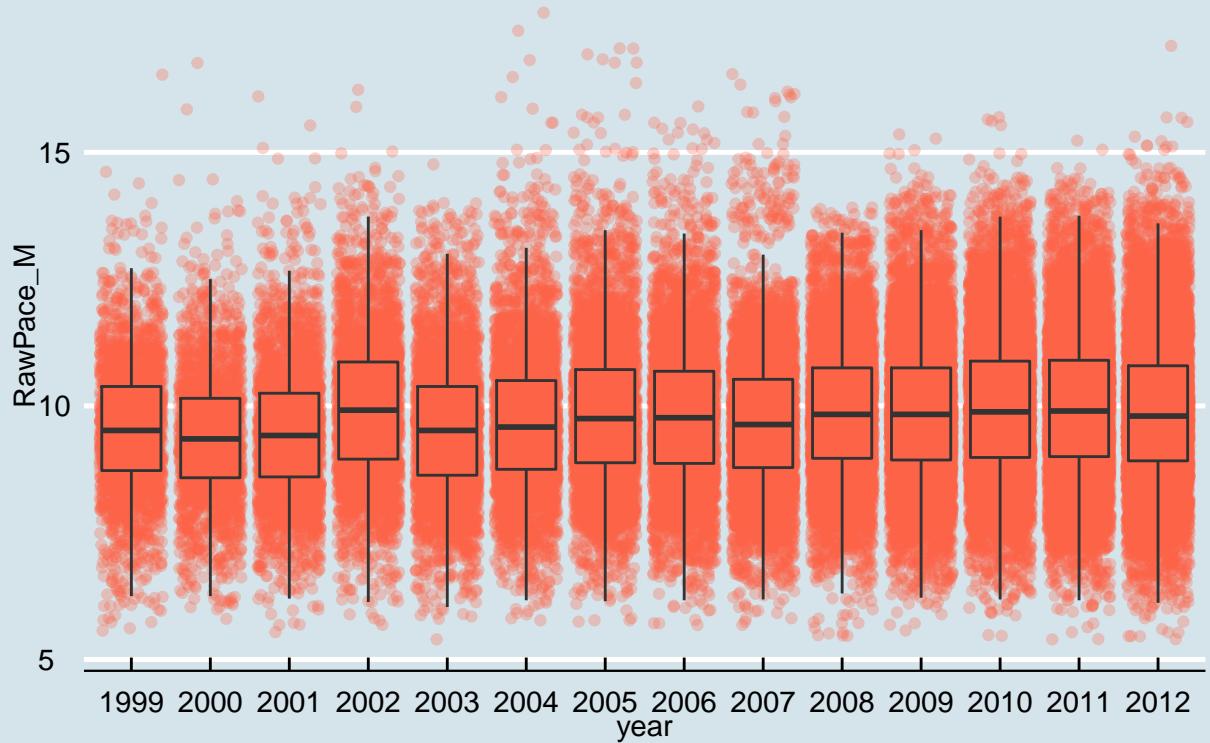
```
#Time by year
ggplot(data = womens_table_T, mapping = aes(x = year, y = RawTime_M)) +
  geom_jitter(alpha = 0.3, color = "tomato") + geom_boxplot(alpha = 0) + theme_economist() + labs(t
```

## Female Time by Year



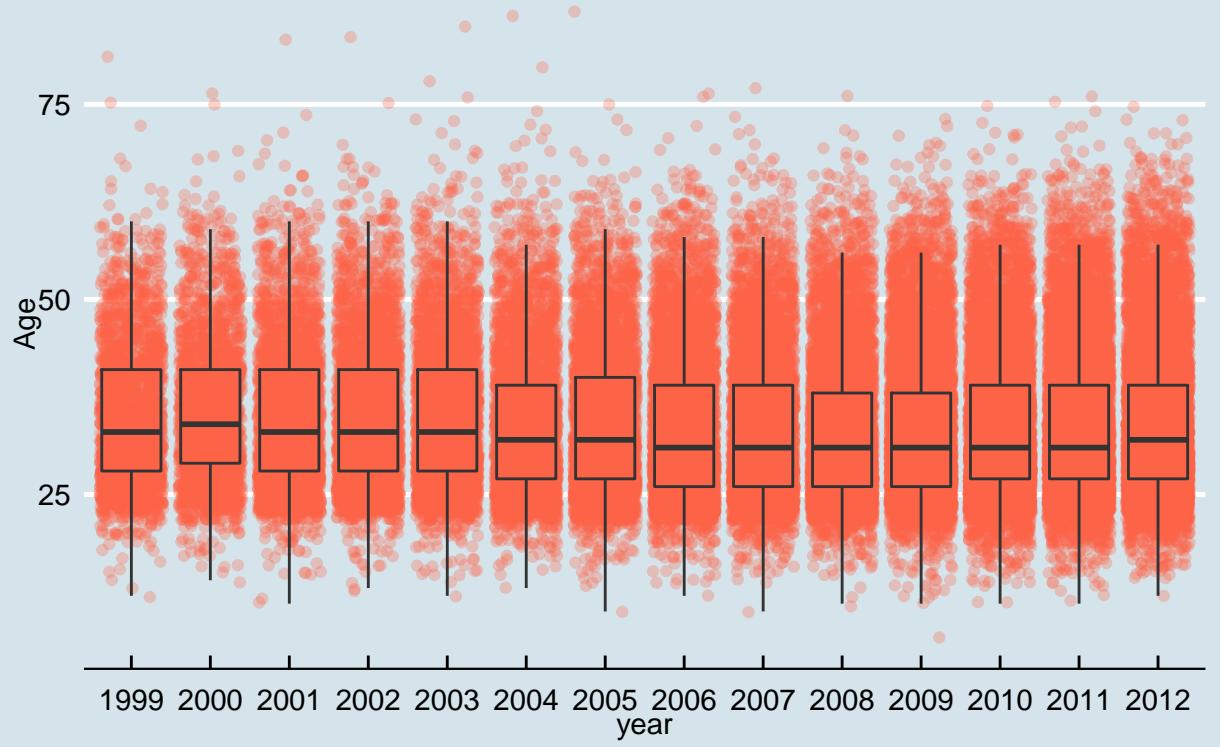
```
#Pace by year
ggplot(data = womens_table_T, mapping = aes(x = year, y = RawPace_M)) +
  geom_jitter(alpha = 0.3, color = "tomato") + geom_boxplot(alpha = 0) + theme_economist() + labs(t
```

## Female Pace by Year



```
library(ggplot2)
library(ggthemes)
#Age by year
ggplot(data = womens_table_T, mapping = aes(x = year, y = Age)) +
  geom_jitter(alpha = 0.3, color = "tomato") + geom_boxplot(alpha = 0) + theme_economist() + labs(t
```

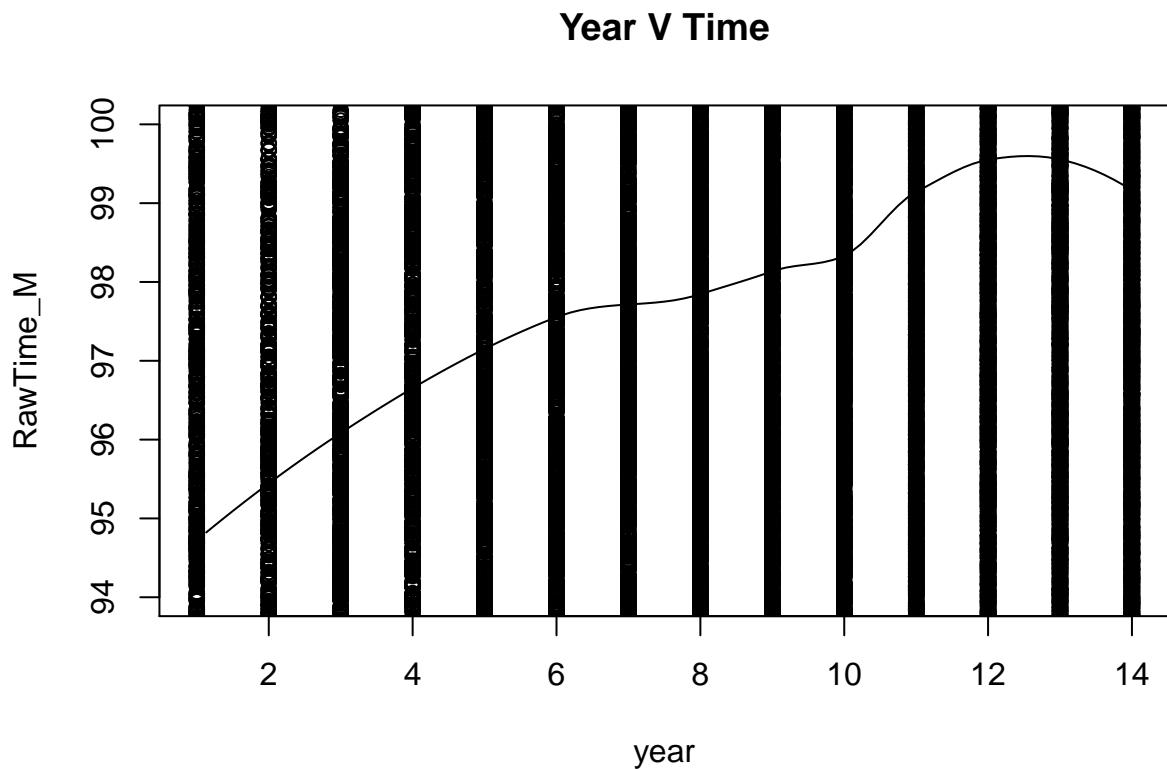
## Female Age by Year



```
#####Sabrina add on#####
#####
```

When we consider how things have changed, we were asked to look at the average racer time. For that, we are going to first consider a LOESS prediction. Because this is a nonparametric method, and will weight itself relative to time, we feel like this should give indication of whether or not the average runner is getting faster or slower.

```
#Predict LOESS on time
womens_table_T$RawTime_M = as.numeric(womens_table_T$RawTime_M) #need data to be numeric for LOESS
womens_table_T$year = as.numeric(womens_table_T$year) #need data to be numeric for LOESS
plot(RawTime_M~year, ylim = c(94,100), data=womens_table_T, main="Year V Time")
out <- loess(RawTime_M~year, data=womens_table_T)
curve(predict(out, newdata=data.frame(year = x)), add=TRUE)
```



As shown above average time actually increased across the 14 year window, but did show a downturn in 2011 and 2012. This indicates that the finishers may in fact be speeding back up again.

Our next objective is to determine if anywhere along the way the average time has sectioned off in a way that indicates multiple trends. For that, we will run a changepoint evaluation on the average.

```
#Run a piecewise fit next
plotdata = womens_table_T %>%
  group_by(year) %>%
  summarise(average = mean(RawTime_M), .groups = "keep")

plotdata$average = as.integer(plotdata$average)
plotdata$average = as.ts(plotdata$average)

library(changepoint)

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##   as.Date, as.Date.numeric

## Successfully loaded changepoint package version 2.2.2
## NOTE: Predefined penalty values changed in version 2.2. Previous penalty values with a postfix 1 i
```

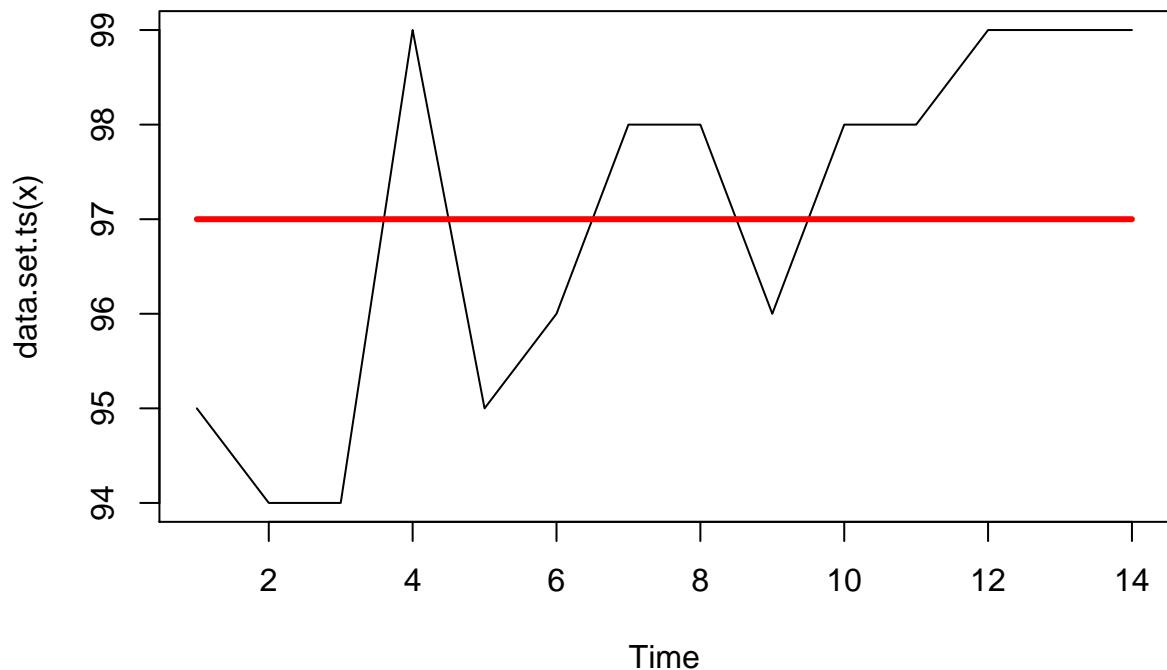
```

dis.bs <- cpt.meanvar(plotdata$average, test.stat = "Poisson", method = "BinSeg")
cpts.ts(dis.bs)

## numeric(0)

plot(dis.bs, cpt.width = 3)

```



You can see that this produced a single average. This could be appropriately stating that nothing has statistically changed. More likely, we do not have enough data points to find clean break points.

### Conclusion / Recommendations

As mentioned in previous sections, the average age of the female runner has decreased in the last 14 years. The average race time of participants has slowed, and the distribution has formed a wider, more even presentation. From this, we conclude that participation in this race has become more popular, and likely as much a social event as a competitive one. As you look to advertise in the future, you should consider expansion of advertising into social media platforms, where you are likely to connect the most effectively with the demographic described above.

As with any expansion, planning must be taken into consideration for an enjoyable event. In the 14 years we evaluated, your race participation size in females quadrupled. We predict you will see over 10,000 female participants in the next year's event. You will need to consider the logistics in organizing to that scale. You should anticipate needing 10% more staff on race day. Volunteers to pass out water, hand out medals, and manage bag checks are one part of that increase. You'll also need additional police to help maintain crowd control. If you provide pace runners, you should consider that in some of your highest volume pace groups,

you may need an additional runner per assigned segment to ensure your participants are able to manage their positions.

Your site provides contact information for hotel accommodations booked at group rates. In the last four years, out of state participation has doubled (based on excluding DC, VA, and MD). Knowing that we expect participation to be higher next year, you should factor this into any hotel negotiations for the next event. You must also ensure that you have sufficient parking, or shuttle service in place if you unable to allow participants and fans to park directly adjacent to the starting line.

```
# looking at counts with division splits too
plotdata = womens_table_T %>% filter(HomeState != "VA") %>% filter(HomeState != "MD") %>% filter(HomeSt
  group_by(year, HomeState) %>%
  summarise(count = n())

## `summarise()` regrouping output by 'year' (override with `.`groups` argument)

p = plotdata %>%
  ggplot(aes(x = year, y = count, fill = HomeState)) +
  geom_bar(stat = "identity", position = "stack")
ggplotly(p)
```

Finally, we recommend that you have contingencies for the unexpected. A myriad of situations could cause this event to be delayed, postponed and even cancelled. Coordinating 10,000 people converging on Washington DC is no small feat, and that is only your female participation. Should you encounter a situation that makes it unsafe to gather, you need to have options available to participants. If you entertain hosting a virtual run in lieu of gathering in person, you must consider the supply chain impacts of needing to distribute race swag and medals. Ordering well in advance, and in bulk since you know your expected participation, will allow you the time needed to handle everyone's participation memorabilia one extra time.