NYC Marathon

Nicholas Jacob

2024-12-09

NYC Marathon 2024

I really wanted to construct the project one more time during the current semester for MATH 3583 Applied Stats. I figured I could express some new coding techniques I have learned over the last four years and hopefully get a little better at R.

For this project, I am using the NYC Marathon results from 2024. Here are the first six finishers. Be careful about printing too much of the data at any one time as it makes your report unreadable. I'll include some of my code but often, I'll just make the outputs available when appropriate.

```
df = read.csv("NYCMarathon2024.csv")
head(df)
```

```
##   runnerId firstName bib age gender              city countryCode
## 1 41771195      Abdi   7  35      M           Nijmegen         NLD
## 2 41775746     Evans   3  35      M           Kapsabet         KEN
## 3 41766254    Albert   2  30      M          Kapkitony         KEN
## 4 41763160   Tamirat   1  33      M        Addis Ababa         ETH
## 5 41757406  Geoffrey   6  31      M Kapchorwa District         KEN
## 6 41772970    Conner  10  27      M              Provo         USA
##   stateProvince iaaf overallPlace overallTime pace genderPlace ageGradeTime
## 1                NED            1     2:07:39 4:53           1         6:57
## 2             -  KEN            2     2:07:45 4:53           2         7:03
## 3                KEN            3     2:08:00 4:53           3         8:00
## 4                ETH            4     2:08:12 4:54           4         8:02
## 5             -  KEN            5     2:08:50 4:55           5         8:50
## 6            UT  USA            6     2:09:00 4:56           6         9:00
##   ageGradePlace ageGradePercent racesCount
## 1             1           96.86          4
## 2             2           96.79          2
## 3             3           96.06          5
## 4             4           96.03          4
## 5             6           95.44          5
## 6             7           95.31          2
```

I first want to give a little check on the quality of the data. I do this with a large block of code that I have hidden away but show the results only.

Table 1: Descriptive Summary of Numeric Variables

variable	n	missing	missing_pct	unique	unique_pct	mean	min	Q1	median	Q3	max	sd
runnerId	55524	0	0.00	55524	100.00	4.2e+07	4.2e+07	4.2e+07	4.2e+07	4.2e+07	4.2e+07	16059
bib	55524	12	0.02	55513	99.98	3.3e+04	1.0e+00	1.7e+04	3.3e+04	4.9e+04	6.7e+04	18932
age	55524	0	0.00	70	0.13	4.0e+01	0.0e+00	3.0e+01	3.9e+01	4.8e+01	8.8e+01	12
overallPlace	55524	0	0.00	55524	100.00	2.8e+04	1.0e+00	1.4e+04	2.8e+04	4.2e+04	5.6e+04	16029
genderPlace	55524	0	0.00	30696	55.28	1.4e+04	1.0e+00	6.9e+03	1.4e+04	2.1e+04	3.1e+04	8287
ageGradePlace	55524	0	0.00	30697	55.29	1.4e+04	0.0e+00	6.9e+03	1.4e+04	2.1e+04	3.1e+04	8287
ageGradePercent	55524	0	0.00	5603	10.09	5.2e+01	0.0e+00	4.5e+01	5.2e+01	5.9e+01	9.7e+01	11
racesCount	55524	0	0.00	315	0.57	1.3e+01	1.0e+00	1.0e+00	2.0e+00	1.2e+01	1.4e+03	30

Table 2: Descriptive Summary of Categorical Variables

variable	n	missing	miss_pct	unique	unique_pct	mode	mode_freq	least common	freq
firstName	55524	0	0.00	12463	22.45	Michael	595	A. AlLEXANDER	1
gender	55524	0	0.00	4	0.01	M	30692		12
city	55524	0	0.00	10604	19.10	New York	10355	(Select Country)	1
countryCode	55524	0	0.00	137	0.25	USA	37695	ABW	1
stateProvince	55524	13	0.02	2351	4.23	NY	21372	— Please Select —	1
iaaf	55524	1	0.00	161	0.29	USA	33361	AHO	1
overallTime	55524	0	0.00	14838	26.72	3:42:24	17	10:00:53	1
pace	55524	0	0.00	966	1.74	9:05	230	17:46	1
ageGradeTime	55524	0	0.00	3600	6.48	0:00	132	30:01:00	3

If you go back and look at the code, you'll see that is a lot of work to make this pretty table. You are welcome to use this code but with so much going on, it is difficult to debug...

There are a couple of issues I am noticing in the current code. First off, `countryCode` and `iaaf` look very similar but the summary statistics are different so not sure what is going on with that. Not a huge issue. The big issue I see is that the `overallTime` is being interpreted as a categorical field. This will need to be fixed!

Data Cleaning

I like to do my data cleaning in the programming language I am using. It will make it so that you don't need to touch or change the data at all and you should be able to recreate what you did right after you load the data. This way it will work for all other parts of the project.

Let's talk for a moment about my process for solving this problem. I think I have done this before but I don't remember exactly how to do it. First I take a guess at what this would be called. I know it isn't a time but a time difference, so I search google with "r time difference". At first I get a bit distracted by the first entry but eventually I see the ETH Zurich site. This one is the R manual. There I see that there is a data class called `difftime`. I skim the top parts and go to the example which seems to do what I want. You see below that it prints the winners time in hours as I had hoped!

```
as.difftime(df$overallTime)[1]
```

```
## Time difference of 2.13 hours
```

I can also convert that the seconds if I wanted to. The `[1]` is limiting the printout just to the very first entry.

```
as.difftime(df$overallTime, units = "secs")[1]
```

```
## Time difference of 7659 secs
```

So I will mutate the time columns so that we can continue our analysis.

```
df <- df %>% mutate(
  overallTime = as.difftime(overallTime)
)
```

Dang it! That worked for te overall time but not for `pace` nor `ageGradeTime`. Since I am not sure what `ageGradeTime` is, I'll just not use it in any analysis. I'll recompute pace by taking the `overallTime` and dividing by the length of the Marathon (26.2 miles) Notice how content knowledge is important to dealing with your data?

```
df <- df %>% mutate(
  pace = ms(pace), #using lubridate and convert og pace
  pace2 = overallTime/26.2 # just divide by length.  Gives pace in hours
)
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `pace = ms(pace)`.
## Caused by warning in `.parse_hms()`:
## ! Some strings failed to parse
```

Well, I could get the division to work but I couldn't seem to get it in an hour format. I left it here as called `pace2`. I did find the tidy version for dealing with dates and times called `lubridate`. This is a library I had to add to the beginning of my document. Now, `pace` has minutes and seconds.

Okay so your data will have some cleaning that is needed. You'll need to start early! Cleaning is a pain and you may need some of my help to get it in a good format. Once the data is clean, you shouldn't need to do that again!

Exploratory Data Analysis

While I already did some of the EDA in my data summary table, I should show some easier to understand code for this too. Here is summary of the times.

```
summary(as.numeric(df$overallTime))
```

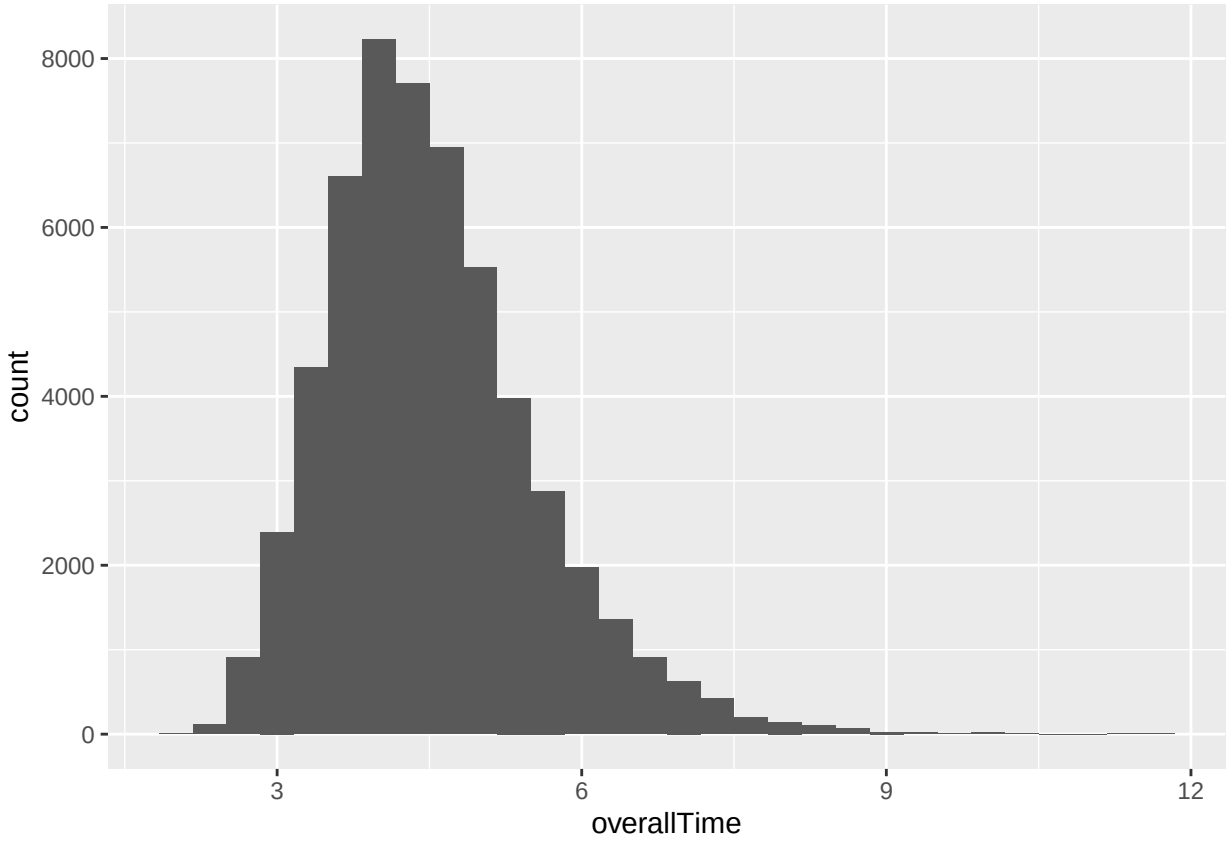
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.13    3.81    4.39    4.53    5.09   11.80
```

I am missing standard deviation, so I add that inline showing that the standard deviation, 1.03.

I add the graphical displays using the `ggplot2` package (I am a big fan!)

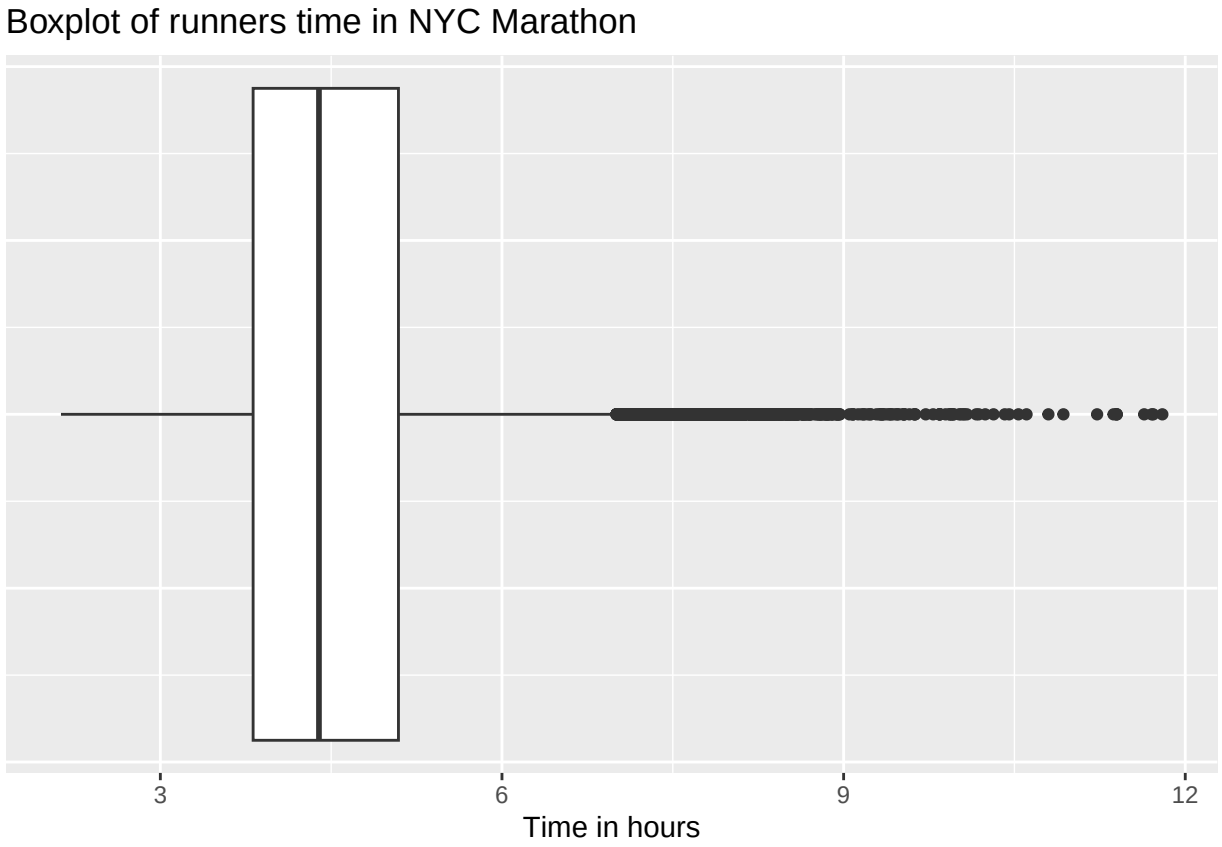
```
ggplot(data = df, aes(x = overallTime))+
  geom_histogram()
```

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = df, aes(x = overallTime))+
  geom_boxplot() +
  labs(x = "Time in hours",
       title = "Boxplot of runners time in NYC Marathon")+
  theme(axis.text.y=element_blank(),axis.ticks.y=element_blank())
```

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.
```



For the categorical variable, we can do some similarly straight forward coding

```
table(df$countryCode)[1:10]#this limited me to 10 entries
```

```
##
##     ABW AGO AND ANT ARE ARG ARM AUS AUT
##  12   1   1   3  59  51 185   1 710 107
```

12 blanks is a little odd. I can do this another way that may give you a better look.

```
df %>% group_by(countryCode)%>%
  summarise(count = n()) %>%
  mutate( freq_pct = count/length(df$runnerId)*100) %>%
  head()
```

```
## # A tibble: 6 x 3
##   countryCode count freq_pct
##   <chr>       <int>    <dbl>
## 1 ""             12  0.0216
## 2 "ABW"           1  0.00180
## 3 "AGO"           1  0.00180
## 4 "AND"           3  0.00540
## 5 "ANT"          59  0.106
## 6 "ARE"          51  0.0919
```

For the two-way table, I'll repeat with two methods. Here is the base method, again restricting so I don't show too much

```
table(df$countryCode,df$iaaf)[1:10,1:10]
```

```
##
##           AFG AHO ALB ALG AND ANG ANT ARG ARM
##        12   0   0   0   0   0   0   0   0   0
##   ABW   0   0   0   0   0   0   0   0   0   0
##   AGO   0   0   0   0   0   0   1   0   0   0
##   AND   0   0   0   0   0   0   0   0   0   0
##   ANT   0   0   1   0   0   0   0   1   0   0
##   ARE   0   0   0   0   0   0   0   0   0   0
##   ARG   0   0   0   0   0   0   0   0 174   0
##   ARM   0   0   0   0   0   0   0   0   0   1
##   AUS   0   0   0   0   0   0   0   0   0   0
##   AUT   0   0   0   0   0   0   0   0   0   0
```

As for the tidy version:

```
df %>% count(countryCode,iaaf)%>%
  head()
```

```
##   countryCode iaaf  n
## 1                  12
## 2         ABW  ARU  1
## 3         AGO  ANG  1
## 4         AND  AUS  1
## 5         AND  ESP  1
## 6         AND  GBR  1
```

This doesn't look like a two-way but it has the same data. (I couldn't find the method to change it to a table quickly so I am happy as is.)

Hypothesis Testing

You can get pretty crazy with your hypotheses. I just ask that you explore something of interest based on your content knowledge. For me, I am interested in playing around with names. I am going to test if people named Nicholas have a different average time in the race from those named Micheal (our mode). I'll state this formally as

$$H_0: \quad \mu_{Nicholas} = \mu_{Micheal}$$
$$H_a: \quad \mu_{Nicholas} \neq \mu_{Micheal}$$

To continue, I'll need to subset my data. This is actually not too hard but has some quirks. Using the $ to get to the variable in the dataset, I ask it be logically equal to what I am looking for.

```
head(df$firstName == "Nicholas")
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
```

This gives a bunch of True/False. To get the data from that, we pass it into the dataframe

```
head(df[df$firstName == "Nicholas",]) #163 is still too many to print them all!
```

```
##     runnerId firstName  bib age gender           city countryCode stateProvince
## 241 41788330  Nicholas 1364  24      M           Reno         USA            NV
## 382 41780288  Nicholas  604  24      M        Memphis         USA            TN
## 404 41772035  Nicholas  693  28      M       New York         USA            NY
## 534 41790532  Nicholas 1866  29      M Salt Lake City         USA            UT
## 541 41758256  Nicholas  773  34      M       New York         USA            NY
```

```
## 662 41764548  Nicholas 4526  29      M     Park Ridge         USA            IL
##     iaaf overallPlace overallTime   pace genderPlace ageGradeTime ageGradePlace
## 241  USA          241  2.58 hours 5M 55S         220     35:05:00           472
## 382  USA          382  2.66 hours  6M 5S         354     39:27:00           794
## 404  USA          404  2.67 hours  6M 7S         376     40:04:00           846
## 534  USA          534  2.72 hours 6M 13S         498     42:55:00          1075
## 541  USA          541  2.72 hours 6M 14S         505     42:35:00          1049
## 662  USA          662  2.75 hours 6M 18S         616     44:56:00          1271
##     ageGradePercent racesCount        pace2
## 241            79.3          1 0.0987 hours
## 382            77.1          1 0.1014 hours
## 404            76.8         19 0.1018 hours
## 534            75.5          1 0.1036 hours
## 541            75.6         37 0.1037 hours
## 662            74.5          6 0.1049 hours
```

Lastly, I get the data I want by taking the subsetted data and asking for that variable with the $ sign again. Now I want the `overallTime`. I've dropped this into my `t.test` to preform the test.

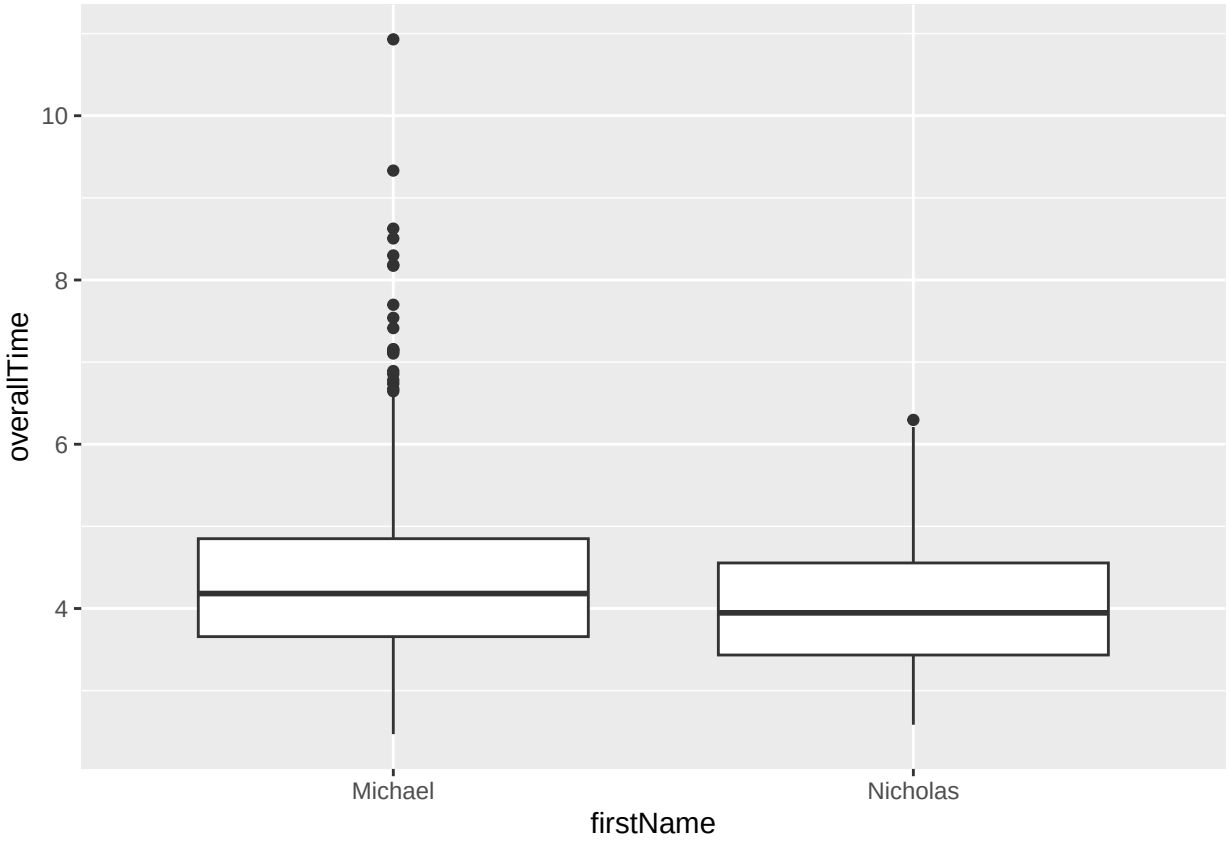
```
ttest <-t.test(as.numeric(df[df$firstName == "Nicholas",]$overallTime),
        as.numeric(df[df$firstName == "Michael",]$overallTime))

ttest
```

```
##
##  Welch Two Sample t-test
##
## data:  as.numeric(df[df$firstName == "Nicholas", ]$overallTime) and as.numeric(df[df$firstName == "M
## t = -4, df = 322, p-value = 0.00007
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.474 -0.164
## sample estimates:
## mean of x mean of y
##      4.05      4.37
```

The results above are printed but we can access the data in other ways too. We can say that this test gave us a p-value of 0.

So we see these are different but we should look at some visualizations to confirm.

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.
```


10
8
6
4
overallTime
Michael
Nicholas
firstName