DataVisualization_523Midterm

Yunxuan

11/18/2017

Rules

Be sure to carefully read and understand all of the rules for this exam contained in the `README.md` file. If you have any questions please ask me or one of the TAs.

Data

For this assignment you will be working with a data from the 2015 Formula 1 season. The data was downloaded from ergast.com in the form of a single large JSON file which contains information on the results of all 19 races from the 2015 season. Your repo should contain both a prettified version of the original json file (`f1.json`) as well as an Rdata binary file (`f1.Rdata`) which can be read in using

```
load(file="f1.Rdata")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library(purrr)
library(tidyr)
library(ggplot2)
```

The data is structured as a list of lists of lists of lists and so on, it is up to you to look at the data and figure out how it is structured and how best to get at the information you want. There is no need to create a tidy data frame of the entire data set, you are free to manipulate and encode the data in anyway you see fit.

Task 1 - 10 pts

Briefly describe the structure of the `f1` object, in particular you should address what information is contained in each level of the list of lists as well as comment on any interesting or unusual features of these data.
Answer:
This object only contains on element, "MR Data", which consists of 7 elements:
xmlns
series
url
limit offset
total
Race-Table
These elements contain information about this giant list (the origin, the download link, and other general information). All other elements except race-table are 1-character length.

RaceTable, on the other hand, is a list containing information of F1 races. In this list, there are two elements: season, which indicates the season of races in this dataset; Races, which contains all races happening in 2015 season.
Races is a list of 19 elemenet, indicating that there were 19 races in total in season 2015.
Within Races, each element (i.e., each single race), contains the following elements:
Season
Round
url
raceName
Circuit - list of 4: the place of circuit, the name of the circuit, the wiki introduction of the circuit, and the location (which itself is another list of 4) (longitude,altitude,city,country)
date
time
results.
Results is a list of elements, with size equal to # of cars participating into the race. Each element is a list of elements that contians information about a specific car. Each element has following sub-elements:
number - car number
position - rank of the result
position text
points - points earned from this race
driver - a list of 8 elements that contains driver's information about id, permanent number, code, intro url, name, date of birth, and nationality. constructor - a list of 4 elemetns that contains the car's information about constructor, its name, its nationality and intro url.
grid
laps
status - whether the game s finished or not.
time - a list of 2 that contains time information, one in miliseconds, one in standard form.
fastest lap - a list of 4 that contains:
rank and time for the fastest lap,
time: list of 1 that contains time cost
average: list of 2 that contain the speed and the unit of the speed, on average

Task 2 - 30 pts

Using these data construct a table showing the World Drivers' Championship standings for this F1 season. This table should resemble the results available on Wikipedia. Your data frame should also have the same 21 columns: Driver name, finishing position for all 19 races, and their overall points total for the season. Failure to finish for any reason (did not start, did not finish, disqualified, etc.) should be coded as an `NA`. Race finishes and points total should all have an integer type. Your data frame should be sorted by points total, but you do not need to include any additional logic to handle ties.
Write-up:
First of all, I create a dataframe tha contains all the information we may need in part 2 and 3.
First, I create [dd], a dataframe that contains the general race information for 19 races. Then I use slice functions to replicate [dd] by numbers of results in each race, which becomes [half1].
We then extract a list of all races, and then use a for-loop to extract the result list for each race into and store the list in [temp]. For each race (i.e., each [temp], we use map functions to extract information from [temp], and store these information into [half2].
Combining [half1] and [half2], we get [final], the dataframe that contains all info we need.
In order to finish the goal of question 2, we first extract driver's name, positions, and points from [final] to calculate the total points earned by each driver, stored in [drivers]. Then we select driver's name, positions, and race rounds, and use spread() to spread the information into a wide dataframe, [driver_spread]. We finally combine these two dataframe via merge(), and sort by the total points in descending order.

```r
#data tidying - create a dataframe to help get the object dataframe desired.
races<-map(f1,"RaceTable")%>%map(.,"Races")#extract races
races<-races[[1]]

dd<-data.frame(
  round=map_chr(races,"round"),
  race_name=map_chr(races,"raceName"),
  data=map_chr(races,"date")
)
num_results <- map(races, "Results") %>% map_int(length)#replicate the race information by the # of res

half_1 <- dd %>%slice(rep(1:n(), num_results))
half_2<-matrix(data=NA, nrow=sum(num_results),ncol=6)
colnames(half_2)=c("position","first_name","last_name","points","status","constructor")

ii<-0
for(i in 1:length(races)){ #1:19, 19 races in total
  #for each race, extract the results and all information we want in the results.
    temp<-map(races[i],"Results")
    poi<-unlist(map(temp[[1]],"points"))
    f_name<-unlist(map(temp[[1]],"Driver")%>%map(.,"givenName"))
    l_name<-unlist(map(temp[[1]],"Driver")%>%map(.,"familyName"))
    pos<-unlist(map(temp[[1]],"position"))
    stat<-unlist(map(temp[[1]],"status"))
    cons<-unlist(map(temp[[1]],"Constructor")%>%map(.,"name"))

  #use a for-loop to inset the results to the dataset one row each time.
  for (j in 1: num_results[i]) {
    half_2[ii+j,]<-c(pos[j],f_name[j],l_name[j],poi[j],stat[j],cons[j])
  }
  ii<-ii+num_results[i]
}

colnames(half_2)=c("position","first_name","last_name","points","status","constructor")
half_22<-as.data.frame((half_2))

final<-cbind(half_1,half_22)
final<-final%>%
  mutate(full_name=paste(first_name,last_name))
final<-as.data.frame(final)

final$points<-as.numeric(as.character(final$points))
final$position<-as.numeric(as.character(final$position))
#1-st dataframe done




#the dataset for part 2
names<-final%>%select(race_name,round)%>%#name of 19 races
  group_by(round)%>%
  unique%>%
  arrange(as.numeric(as.character(round)))
```

```r
drivers<-final%>%#drivers' total points in season 2015
  select(full_name,points)%>%
  group_by(full_name)%>%
  summarize(total=sum(points))%>%
  arrange(desc(total))

driver_spread<-final%>%mutate(round= factor(round, levels=unique(round)))%>%
  select(full_name,position,round)%>%
  spread(round,position)
#The use of spread is inspired by:
#https://rpubs.com/bradleyboehmke/data_wrangling
#https://stackoverflow.com/questions/35103523/tidyr-spread-sorting-inconsistencies

driver_standings<-merge(driver_spread,drivers,by="full_name")#Mergs() inspired by http://www.statmethod
driver_standings<-driver_standings%>%
  arrange(desc(total))
colnames(driver_standings)=c("driver",as.character(names$race_name),"total_points")

knitr::kable(driver_standings)
```

driver	Australian Grand Prix	Malaysian Grand Prix	Chinese Grand Prix	Bahrain Grand Prix	Spanish
Lewis Hamilton	1	2	1	1	
Nico Rosberg	2	3	2	3	
Sebastian Vettel	3	1	3	5	
Kimi Räikkönen	12	4	4	2	
Valtteri Bottas	18	5	6	4	
Felipe Massa	4	6	5	10	
Daniil Kvyat	16	9	19	9	
Daniel Ricciardo	6	10	9	6	
Sergio Pérez	10	13	11	8	
Nico Hülkenberg	7	14	20	13	
Romain Grosjean	14	11	7	7	
Max Verstappen	13	7	17	18	
Felipe Nasr	5	12	8	12	
Pastor Maldonado	15	16	18	15	
Carlos Sainz	9	8	13	19	
Jenson Button	11	17	14	20	
Fernando Alonso	NA	18	12	11	
Marcus Ericsson	8	19	10	14	
Alexander Rossi	NA	NA	NA	NA	
Kevin Magnussen	17	NA	NA	NA	
Roberto Merhi	NA	15	16	17	
Will Stevens	NA	20	15	16	

Task 3 - 30 pts

Using these data construct a table showing the World Constructors' Championship standings for this F1 season. Your data frame should have 10 rows, one for each team/constructor, and 21 columns: team name, 19 columns that contain the points earned for each race, and team overall points total. Note that a team can have multiple drivers, so the reported points should be to total of all points scored for that race. It would be a good idea to compare your computed points total to those given in the table here.

Write up:
This is pretty similar to building the dataset in part 2.
We first extract constructors, points to calculate the total points earned by each driver, stored in [constructor]. Then we select constructors, points, and race rounds from [final], and use spread() to spread the information into a wide dataframe, [cons_spread]. We finally combine these two dataframe via merge(), and sort by the total points in descending order.

```
constructor<-final%>%# the constructors' total points in season 2015
  select(constructor,points)%>%
  group_by(constructor)%>%
  summarize(total=sum(points))

cons_spread<-final%>%mutate(round=factor(round, levels=unique(round)))%>%
  select(constructor,points,round)%>%
  group_by(constructor,round)%>%
  summarize(total=sum(points))%>%
  spread(round,total)

constructor_standings<-merge(cons_spread,constructor,by="constructor")%>%arrange(desc(total))
colnames(constructor_standings)=c("constructor",as.character(names$race_name),"total_points")

constructor_standings<-as.data.frame(constructor_standings)

knitr::kable(constructor_standings)
```

constructor	Australian Grand Prix	Malaysian Grand Prix	Chinese Grand Prix	Bahrain Grand Prix	Spanish
Mercedes	43	33	43	40	
Ferrari	15	37	27	28	
Williams	12	18	18	13	
Red Bull	8	3	2	10	
Force India	7	0	0	4	
Lotus F1	0	0	6	6	
Toro Rosso	2	10	0	0	
Sauber	14	0	5	0	
McLaren	0	0	0	0	
Manor Marussia	NA	0	0	0	

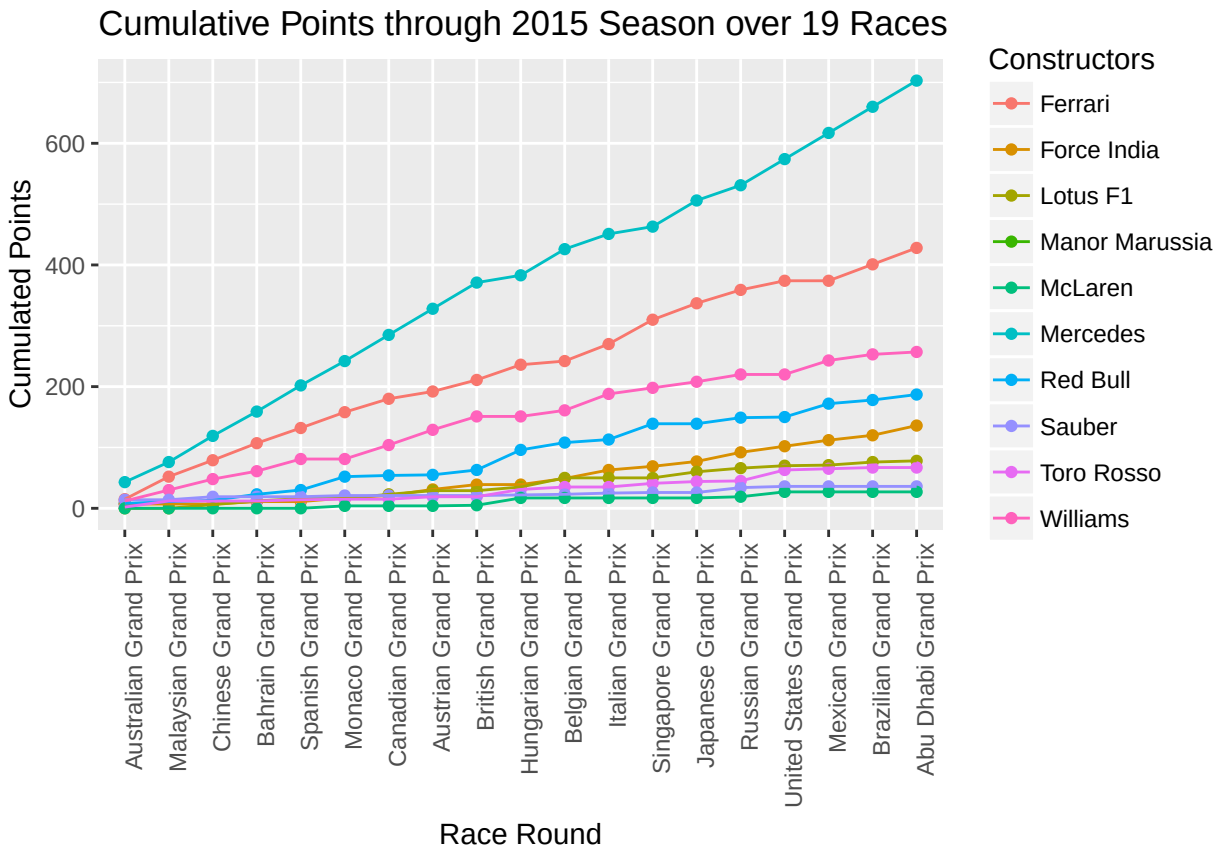
Task 4 - 30 pts

Using the data frame created in task 3, construct a visualization comparing the 10 teams that shows their *cumulative* points earned throughout the 2015 season. This plot should have cumulative points on the y-axis and race on the x-axis with team/constructor identified by color or some other aesthetic.
Write up: This is pretty similar to what we have done in hw3. First, we transform the dataset in part 3 into a dataset with three columns: constructor, round, points. Then we use ggplot, and group by the constructor to plot the data. Cumulative points can be calculated by cumsum().

```
modi_df<-matrix(NA,nrow=190,ncol=3)#modified the dataframe in part 3 for visualization
j<-constructor_standings$constructor
modi_df[,1]<-rep(as.character(j),19)#similar steps as in hw3
modi_df[,2]<-c(sapply(c(1:19),function(x){rep(x,10)}))
modi_df[,3]<-c(apply(constructor_standings[,2:20],2,print))
```

```
modi_df1<-as.data.frame(modi_df)
colnames(modi_df1)<-c("constructor","round","points")
modi_df1$points<-as.numeric(as.character(modi_df1$points))
```

```
visualization_constructors<-modi_df1%>%
  group_by(constructor)%>%
  mutate(cumsum=cumsum(points))%>%
  ggplot(., aes(x=as.numeric(as.character(round)),color=as.factor(constructor)))+geom_point(aes(y=cumsu
  geom_line(aes(y=cumsum))+
  labs(x="Race Round",y="Cumulated Points",colour="Constructors",title="Cumulative Points through 2015 S
visualization_constructors
```




```
#the way to do cumsum first is inspired by https://stackoverflow.com/questions/30441406/start-multiple-
#the use of scale_x_discrete: https://rstudio-pubs-static.s3.amazonaws.com/3364_d1a578f521174152b46b19d
#on rotating x-axis label: https://stackoverflow.com/questions/1330989/rotating-and-spacing-axis-labels
```