#Loading the rvest, and tidyverse packages

**library(rvest)**

**library(tidyverse)**

#Specifying the url for desired website to be scraped

url <- 'http://www.imdb.com/search/title?count=100&release\_date=2016,2016&title\_type=feature'

**Webpage <- read\_html("https://www.imdb.com/search/title/?count=100&release\_date=2016,2016&title\_type=feature")**

#Using CSS selectors to scrape the rankings section

rank\_data\_html <- html\_nodes(Webpage, '.text-primary')

#Converting the ranking data to text

rank\_data <- html\_text(rank\_data\_html)

head(rank\_data)

Output - "1." "2." "3." "4." "5." "6."

#Data-Preprocessing: Converting rankings to numerical

rank\_data <- as.numeric(rank\_data)

head(rank\_data)

Output- [1] 1 2 3 4 5 6

#Using CSS selectors to scrape the title section

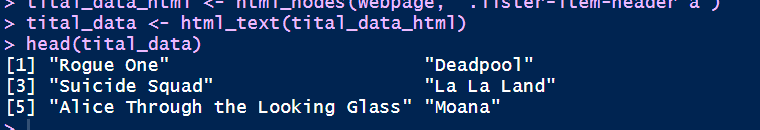
tital\_data\_html <- html\_nodes(Webpage, '.lister-item-header a')

#Converting the title data to text

tital\_data <- html\_text(tital\_data\_html)

#Let's have a look at the title

head(tital\_data)



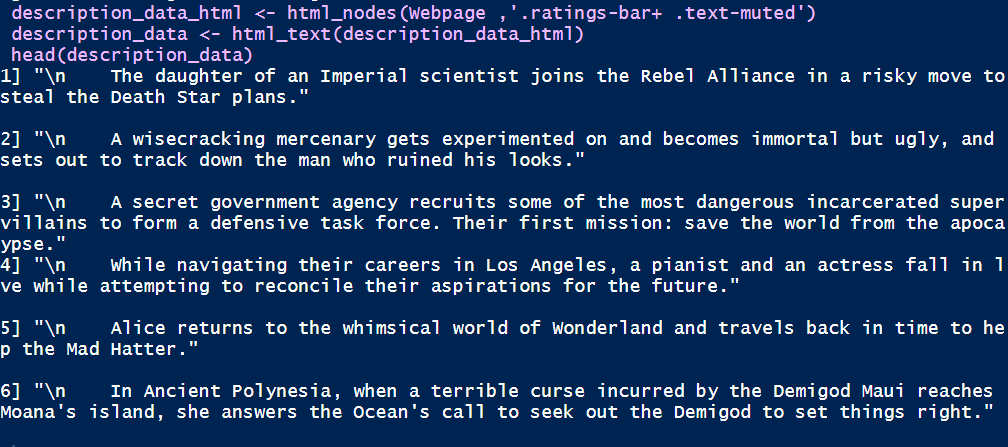
#Using CSS selectors to scrape the description section

description\_data\_html <- html\_nodes(Webpage ,'.ratings-bar+ .text-muted')

#Converting the description data to text

description\_data <- html\_text(description\_data\_html)

head(description\_data)

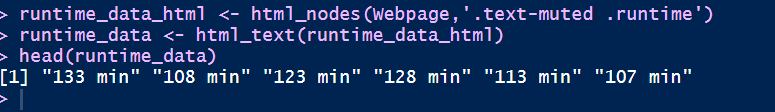


#Using CSS selectors to scrape the Movie runtime section

runtime\_data\_html <- html\_nodes(Webpage,'.text-muted .runtime')

runtime\_data <- html\_text(runtime\_data\_html)

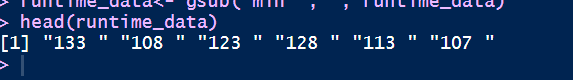
head(runtime\_data)



#Data-Preprocessing: removing mins and converting it to numerical

runtime\_data<- gsub("min" ,"", runtime\_data)

head(runtime\_data)



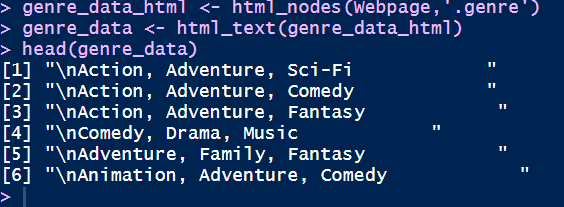
#Using CSS selectors to scrape the Movie genre section

genre\_data\_html <- html\_nodes(Webpage,'.genre')

genre\_data <- html\_text(genre\_data\_html)

#Let's have a look at the runtime

head(genre\_data)



#Data-Preprocessing: removing \n

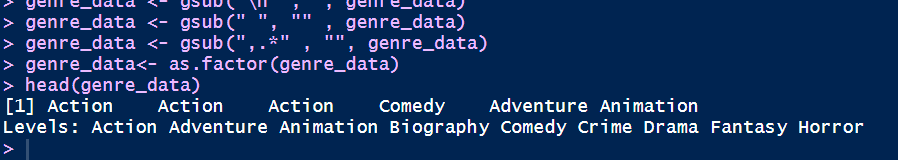
genre\_data <- gsub("\n" ,"", genre\_data)

genre\_data <- gsub(" ", "" , genre\_data)

genre\_data <- gsub(",.\*" , "", genre\_data)

genre\_data<- as.factor(genre\_data)

head(genre\_data)



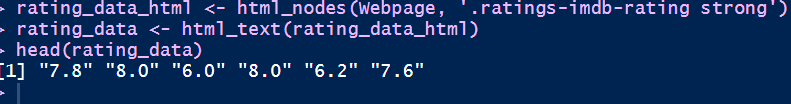
#Using CSS selectors to scrape the IMDB rating section

rating\_data\_html <- html\_nodes(Webpage, '.ratings-imdb-rating strong')

rating\_data <- html\_text(rating\_data\_html)

#Let's have a look at the ratings

head(rating\_data)

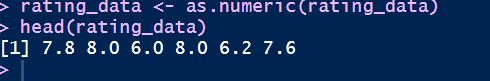


#Data-Preprocessing: converting ratings to numerical

rating\_data <- as.numeric(rating\_data)

#Let's have another look at the ratings data

head(rating\_data)



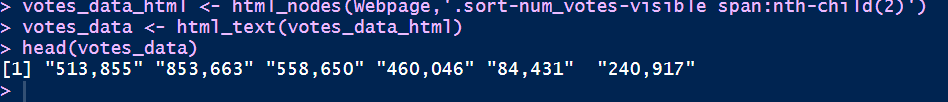
#Using CSS selectors to scrape the votes section

votes\_data\_html <- html\_nodes(Webpage,'.sort-num\_votes-visible span:nth-child(2)')

votes\_data <- html\_text(votes\_data\_html)

#Let's have a look at the votes data

head(votes\_data)

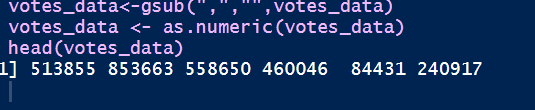


#Data-Preprocessing: removing commas

votes\_data<-gsub(",","",votes\_data)

votes\_data <- as.numeric(votes\_data)

head(votes\_data)



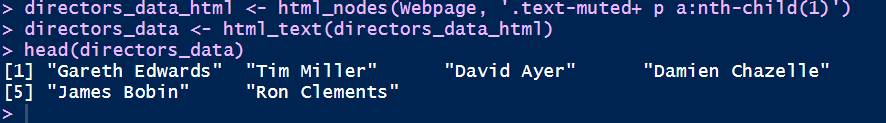
#Using CSS selectors to scrape the directors section

directors\_data\_html <- html\_nodes(Webpage, '.text-muted+ p a:nth-child(1)')

directors\_data <- html\_text(directors\_data\_html)

#Let's have a look at the directors data

head(directors\_data)



#Data-Preprocessing: converting directors data into factors

directors\_data <- as.factor(directors\_data)

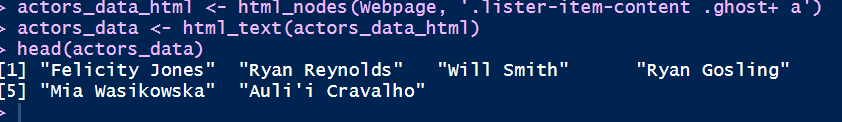
#Using CSS selectors to scrape the actors section

actors\_data\_html <- html\_nodes(Webpage, '.lister-item-content .ghost+ a')

actors\_data <- html\_text(actors\_data\_html)

#Let's have a look at the actors data

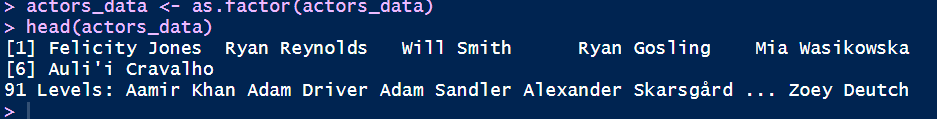
head(actors\_data)



#Data-Preprocessing: converting actors data into factors

actors\_data <- as.factor(actors\_data)

head(actors\_data)



#Using CSS selectors to scrape the metascore section

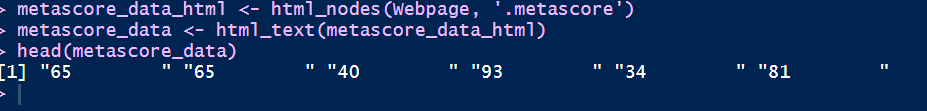
metascore\_data\_html <- html\_nodes(Webpage, '.metascore')

#Converting the runtime data to text

metascore\_data <- html\_text(metascore\_data\_html)

#Let's have a look at the metascore

head(metascore\_data)

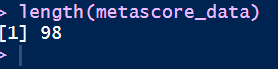


#Data-Preprocessing: removing extra space in metascore

metascore\_data <- gsub(" ", "", metascore\_data)

#Lets check the length of metascore data

length(metascore\_data)



The length of the metascore data is 98 while we are scraping the data for 100 movies. The reason this happened is that there are 2 movies that don’t have the corresponding Metascore fields.

for(i in c(44,85)) {

a <- metascore\_data[1:(i-1)]

b <- metascore\_data[i:length(metascore\_data)]

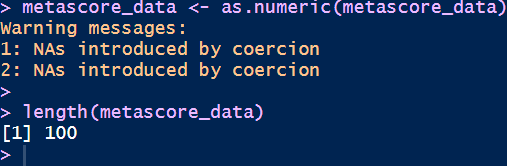
metascore\_data <- append(a,list("NA"))

metascore\_data <- append(metascore\_data,b)

}

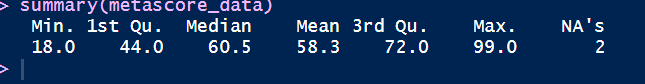
metascore\_data <- as.numeric(metascore\_data)

length(metascore\_data)



#Let's look at summary statistics

summary(metascore\_data)



Now we have successfully scraped all the 11 features for the 100 most popular feature films released in 2016. Let’s combine them to create a dataframe and inspect its structure.

#Combining all the lists to form a data frame

movies\_df = data.frame(Rank = rank\_data , Title = tital\_data,

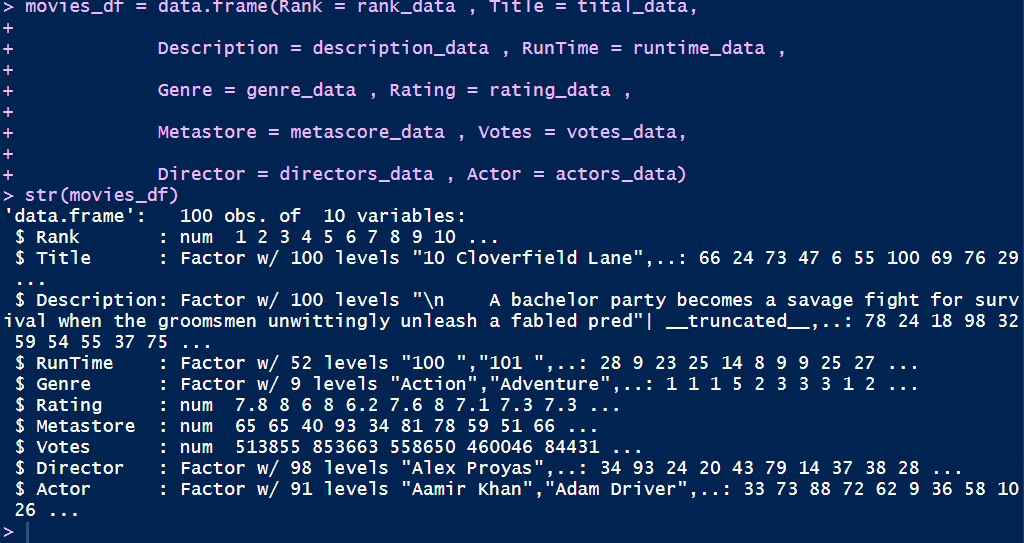
Description = description\_data , RunTime = runtime\_data ,

Genre = genre\_data , Rating = rating\_data ,

Metastore = metascore\_data , Votes = votes\_data,

Director = directors\_data , Actor = actors\_data)

str(movies\_df)

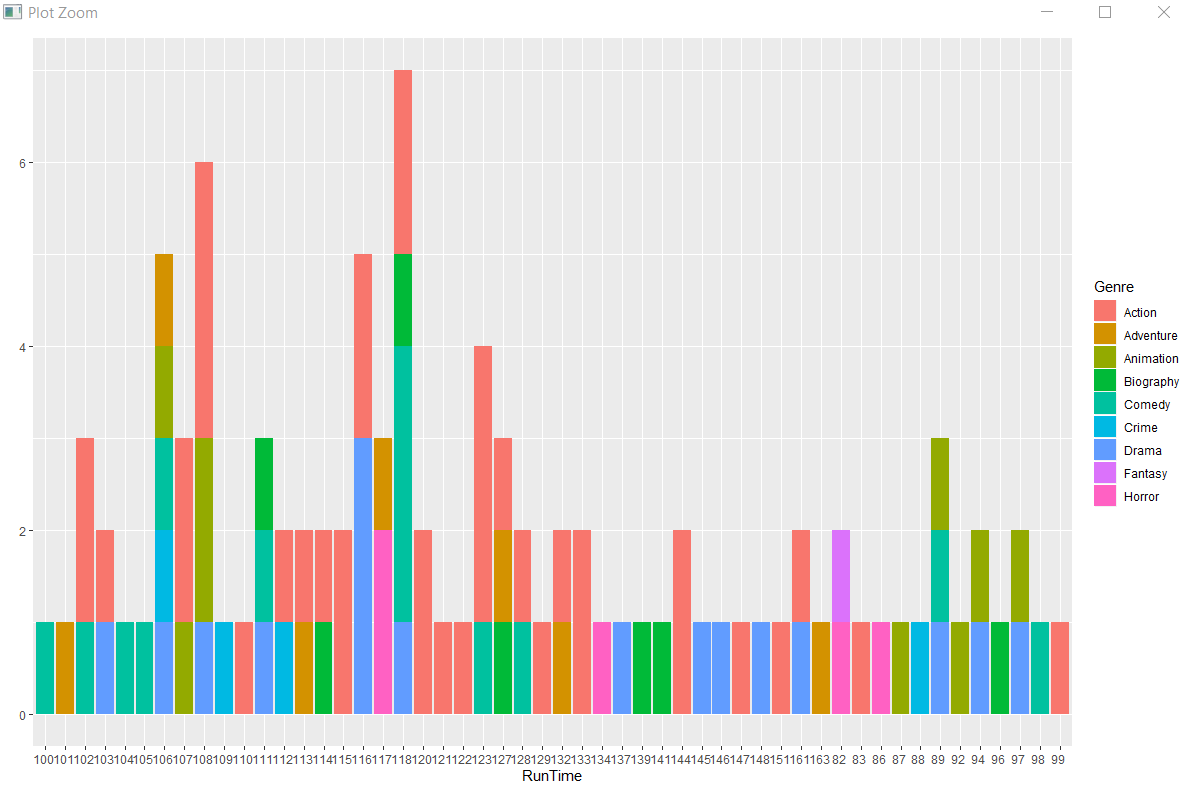


**Analyzing scraped data from the web**

Once you have the data, you can perform several tasks like analyzing the data, drawing inferences from it, training machine learning models over this data, etc. I have gone on to create some interesting visualization out of the data we have just scraped. Follow the visualizations and answer the questions given below. Post your answers in the comment section below.

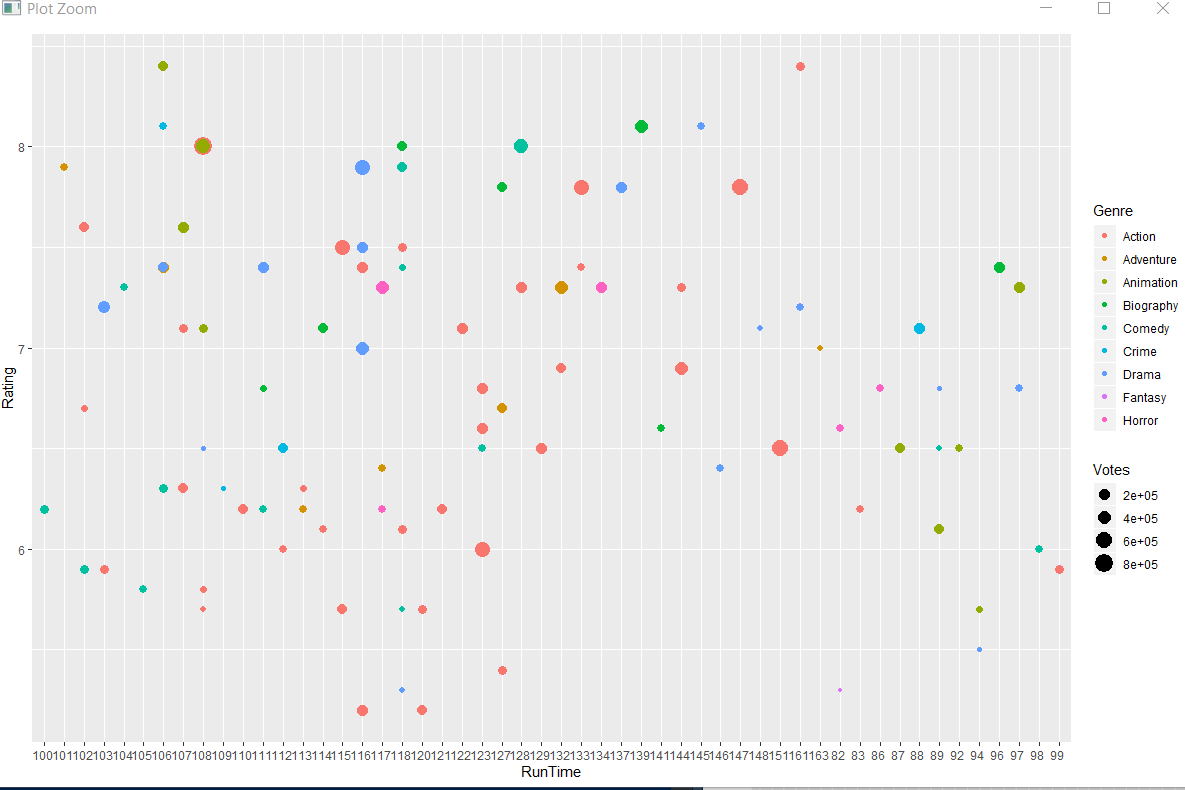
library(ggplot2)

qplot(data = movies\_df , RunTime, fill= Genre , bins = 30)



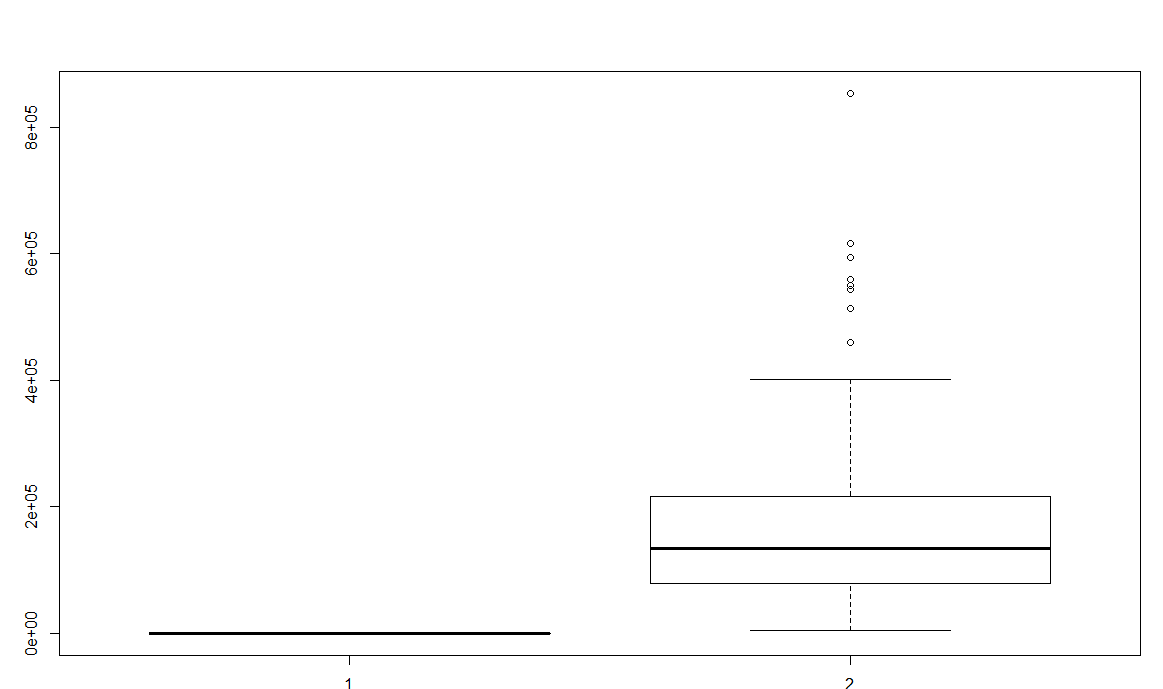
ggplot(movies\_df, aes(x= RunTime, y = Rating))+

geom\_point(aes(size = Votes, col = Genre))



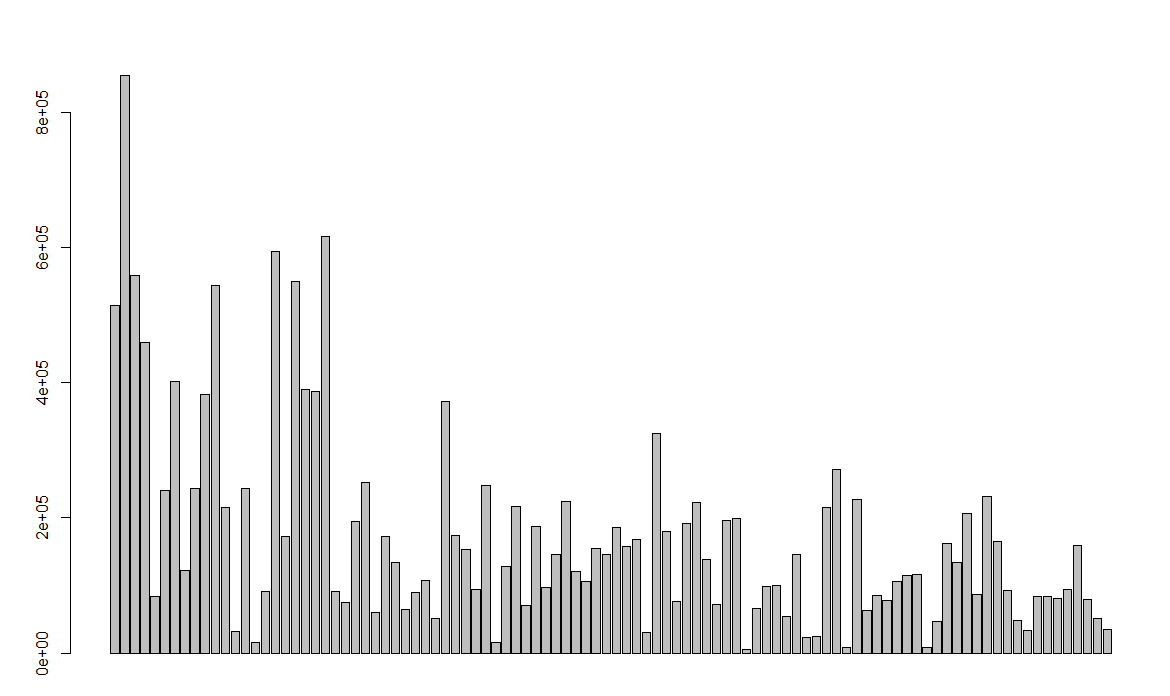
Box plot -

boxplot(movies\_df$Rating, movies\_df$Votes)



Barplot

barplot(movies\_df$Votes)



Discuss the legality of webscraping outside the scope of this data, what problems can webscraping cause?