DATA201 – ASSIGNMENT

PLEASE GIVE US AN A+

Mike Whitley, Ben Duke, Juan Miguel, Ryan Sheridan

STARTING OUT

PICKING A TOPIC

Movies

Research Questions

Most profitable movie genre

Most profitable movie based on budget

Profit of movies based on length

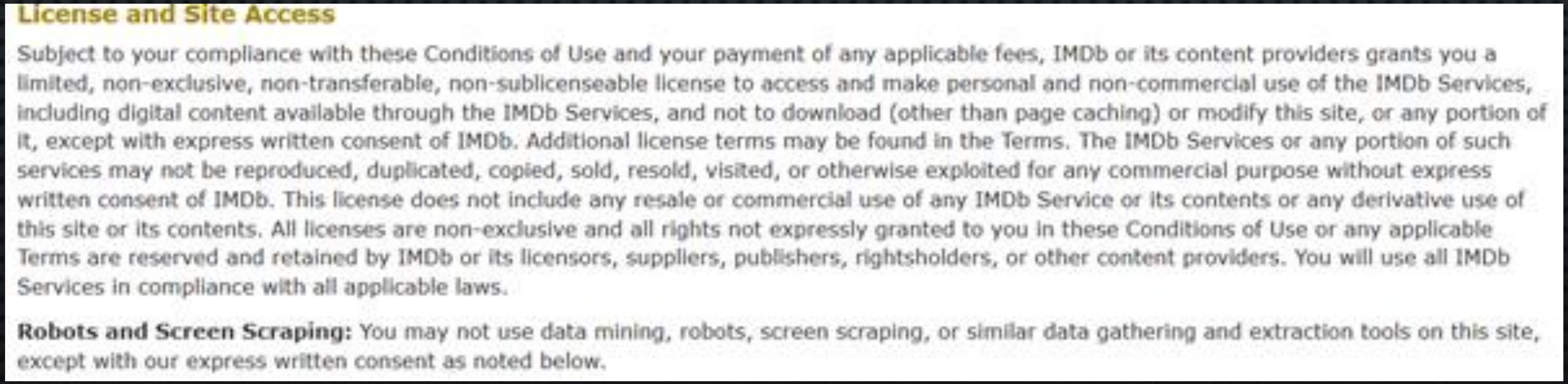
ETHICS

- [HTTP://DEON.DRIVENDATA.ORG/](http://deon.drivendata.org/)
- WEB SCRAPING
- API'S

Rules and Regulations
Rules of Conduct
Values
Ethics
Ethical Practices
Research
Moral Principles

WEB SCRAPING AND API ETHICS WITH GATHING OUR DATA

- Api avaliable at unresonable prices
- Checking terms and conditions
- Abidng by robots.txt (robots exclusion standards)



License and Site Access

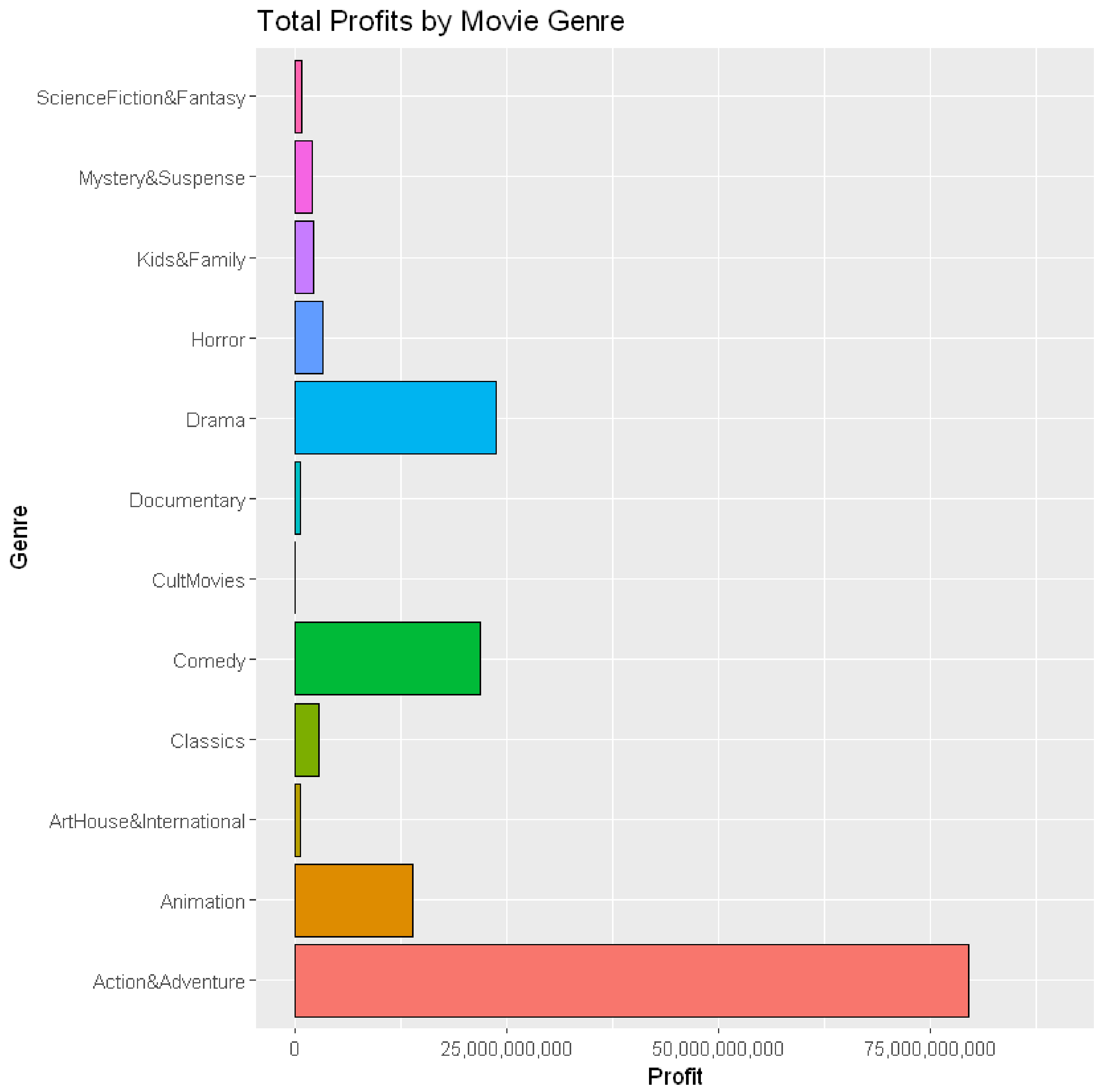
Subject to your compliance with these Conditions of Use and your payment of any applicable fees, IMDb or its content providers grants you a limited, non-exclusive, non-transferable, non-sublicenseable license to access and make personal and non-commercial use of the IMDb Services, including digital content available through the IMDb Services, and not to download (other than page caching) or modify this site, or any portion of it, except with express written consent of IMDb. Additional license terms may be found in the Terms. The IMDb Services or any portion of such services may not be reproduced, duplicated, copied, sold, resold, visited, or otherwise exploited for any commercial purpose without express written consent of IMDb. This license does not include any resale or commercial use of any IMDb Service or its contents or any derivative use of this site or its contents. All licenses are non-exclusive and all rights not expressly granted to you in these Conditions of Use or any applicable Terms are reserved and retained by IMDb or its licensors, suppliers, publishers, rightsholders, or other content providers. You will use all IMDb Services in compliance with all applicable laws.

Robots and Screen Scraping: You may not use data mining, robots, screen scraping, or similar data gathering and extraction tools on this site, except with our express written consent as noted below.

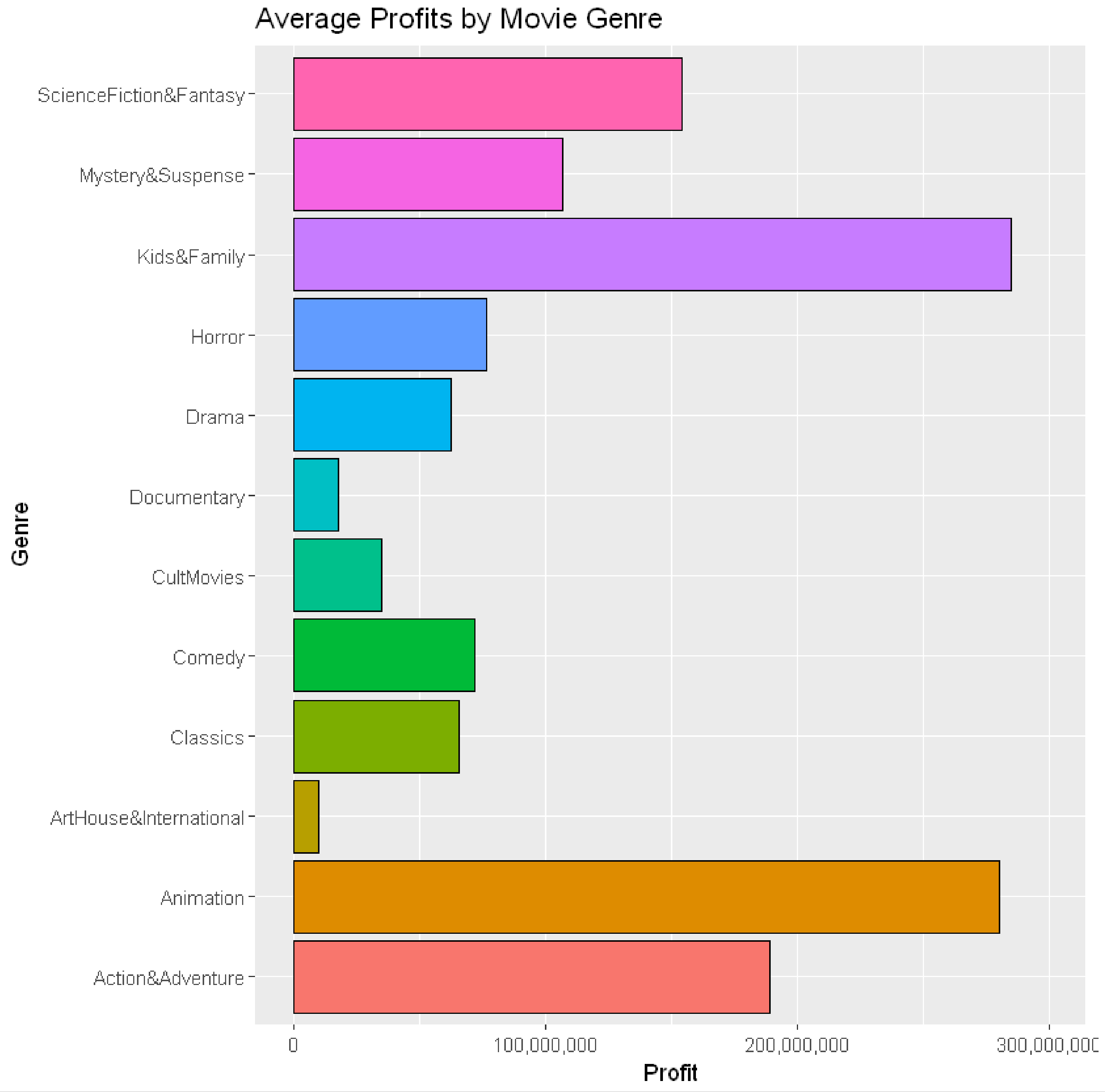
ROTTEN TOMATOES WEB SCRAPING

- Top 100 Movies for each year on the Rotten Tomatoes database
- Time Consuming
- Variable type formatting and Web errors

TOTAL PROFITS BY MOVIE GENRE



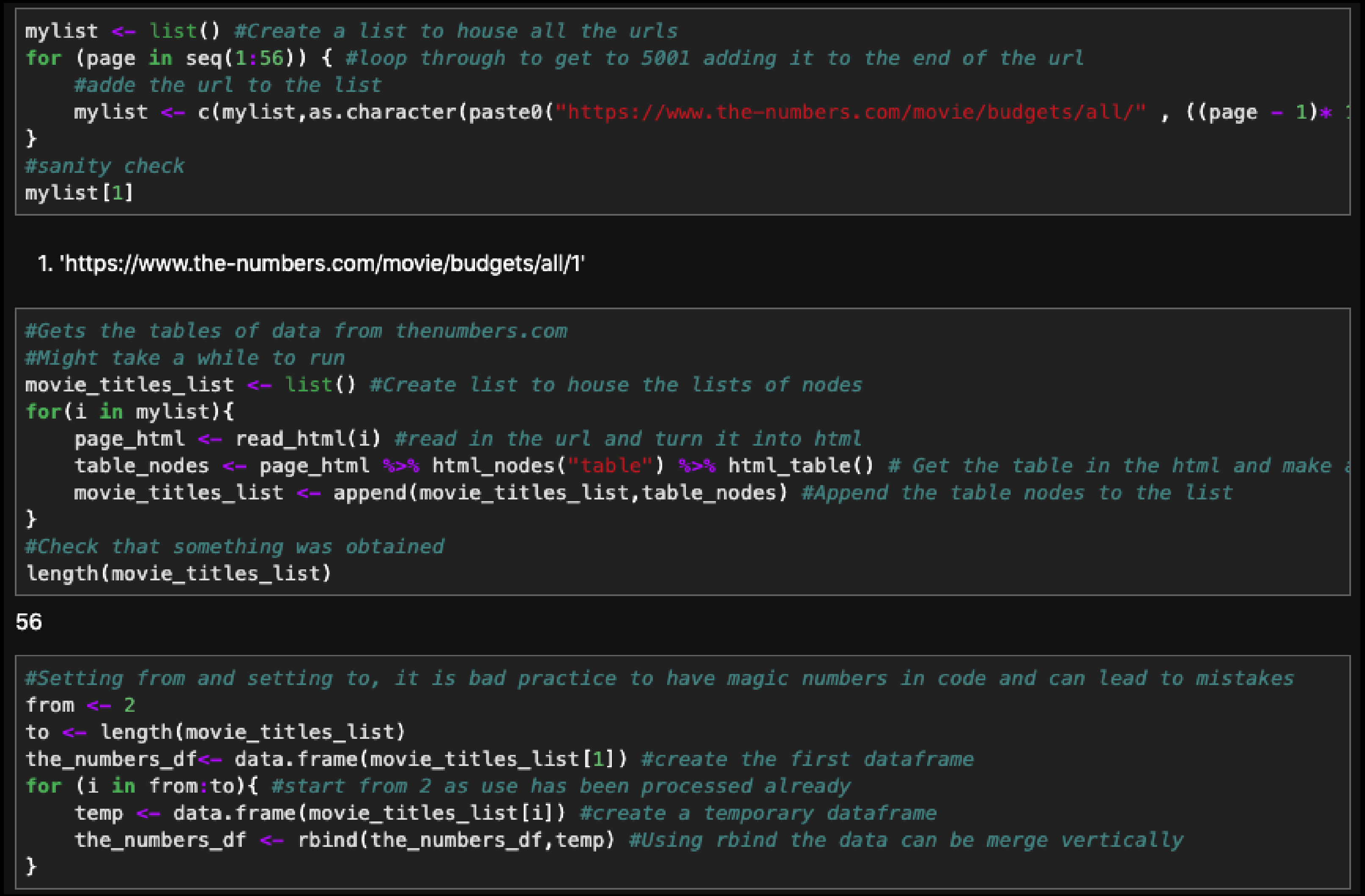
AVERAGE PROFITS BY MOVIE GENRE



ADD SOME CODE FOR WEB SCRAPING THE NUMBERS

- *TALK ABOUT WHAT YOU DID AND CHALLENGES

THE-NUMBERS.COM WEB SCRAPING



```
mylist <- list() #Create a list to house all the urls
for (page in seq(1:56)) { #loop through to get to 5001 adding it to the end of the url
    #adde the url to the list
    mylist <- c(mylist,as.character(paste0("https://www.the-numbers.com/movie/budgets/all/" , ((page - 1)*
}
#sanity check
mylist[1]
```

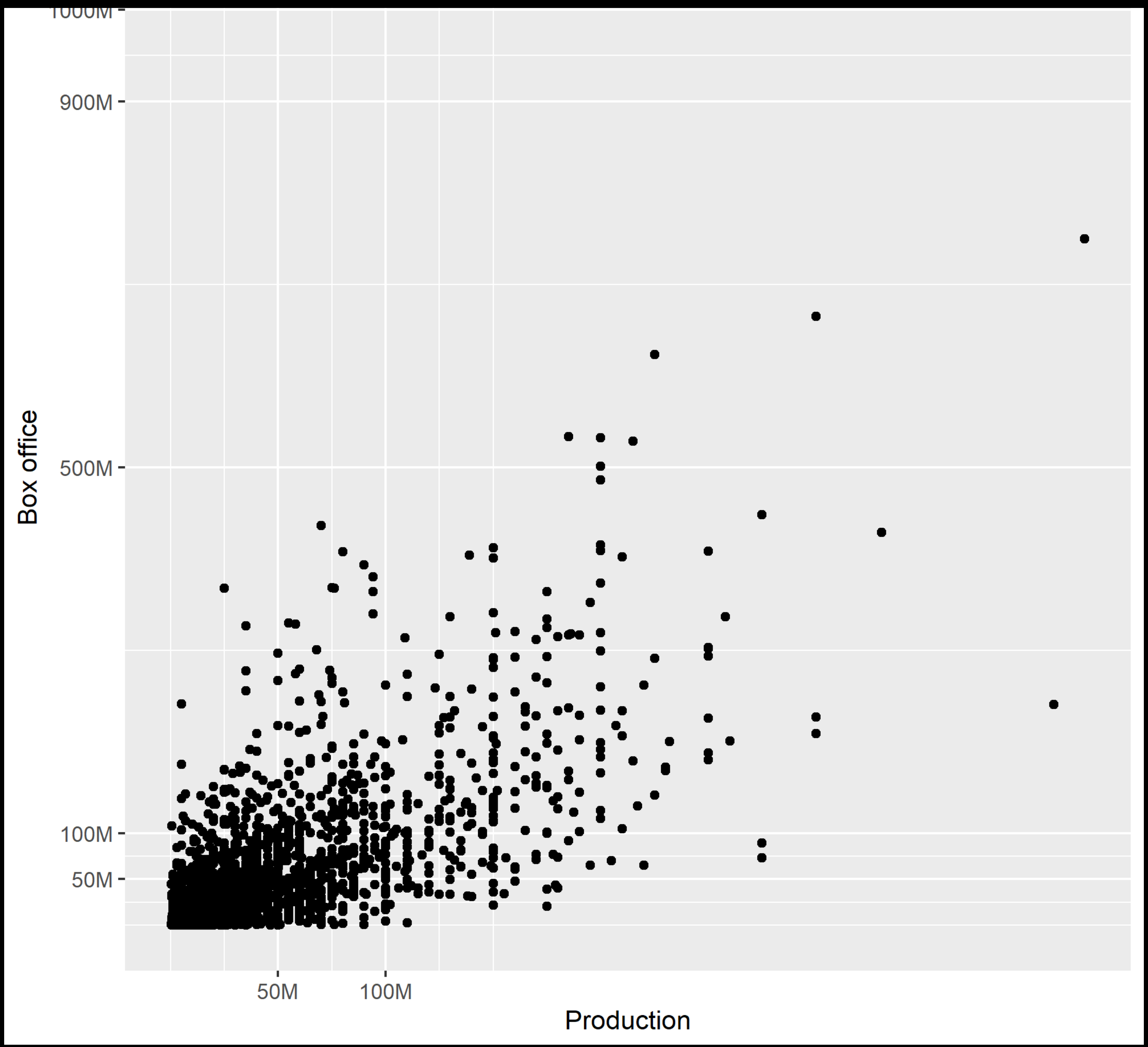
```
1. 'https://www.the-numbers.com/movie/budgets/all/1'
```

```
#Gets the tables of data from thenumbers.com
#Might take a while to run
movie_titles_list <- list() #Create list to house the lists of nodes
for(i in mylist){
    page_html <- read_html(i) #read in the url and turn it into html
    table_nodes <- page_html %>% html_nodes("table") %>% html_table() # Get the table in the html and make
    movie_titles_list <- append(movie_titles_list,table_nodes) #Append the table nodes to the list
}
#Check that something was obtained
length(movie_titles_list)
```

```
56
```

```
#Setting from and setting to, it is bad practice to have magic numbers in code and can lead to mistakes
from <- 2
to <- length(movie_titles_list)
the_numbers_df<- data.frame(movie_titles_list[1]) #create the first dataframe
for (i in from:to){ #start from 2 as use has been processed already
    temp <- data.frame(movie_titles_list[i]) #create a temporary dataframe
    the_numbers_df <- rbind(the_numbers_df,temp) #Using rbind the data can be merge vertically
}
```

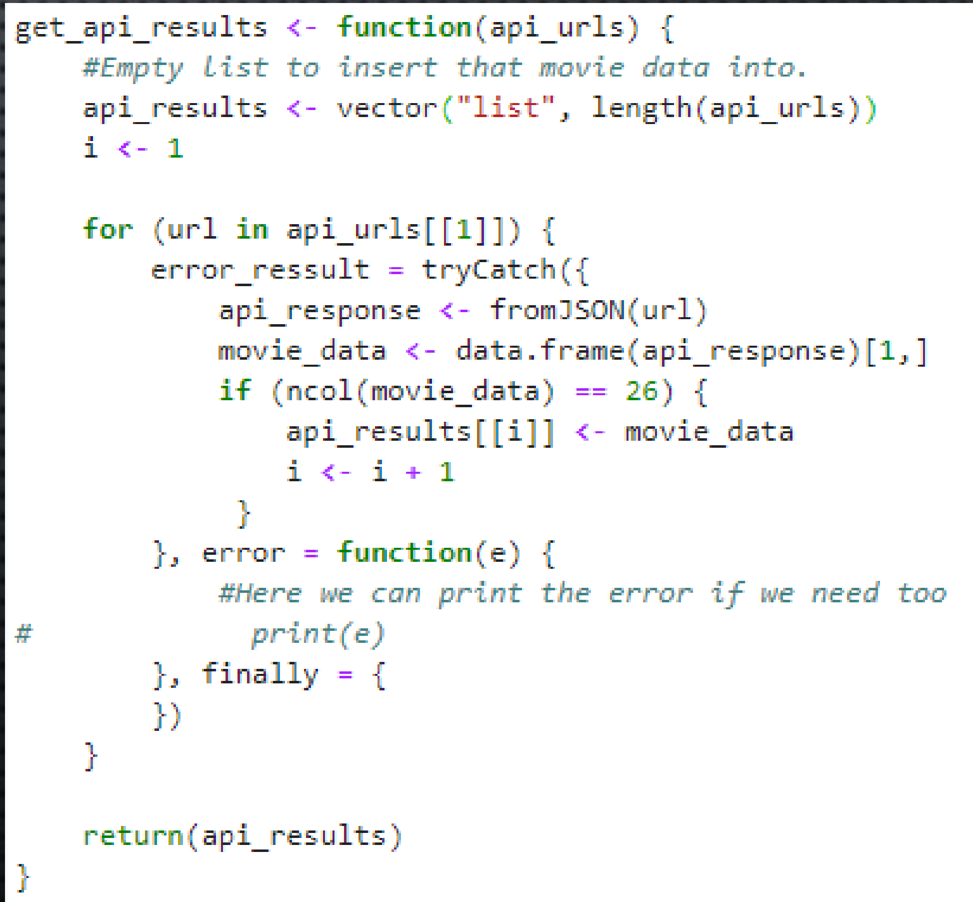
MOST PROFITABLE MOVIE BASED ON BUDGET



IMDB API

- Used the titles gathered from the 2 scrapes to search the API
- 3 Main functions used
  - Titles to Url's
  - Searching the API
  - Cleaning the JSON returned from the API

- Lots of struggles with the AP returning weird data
- Also struggled with API errors, needed to buy an API key



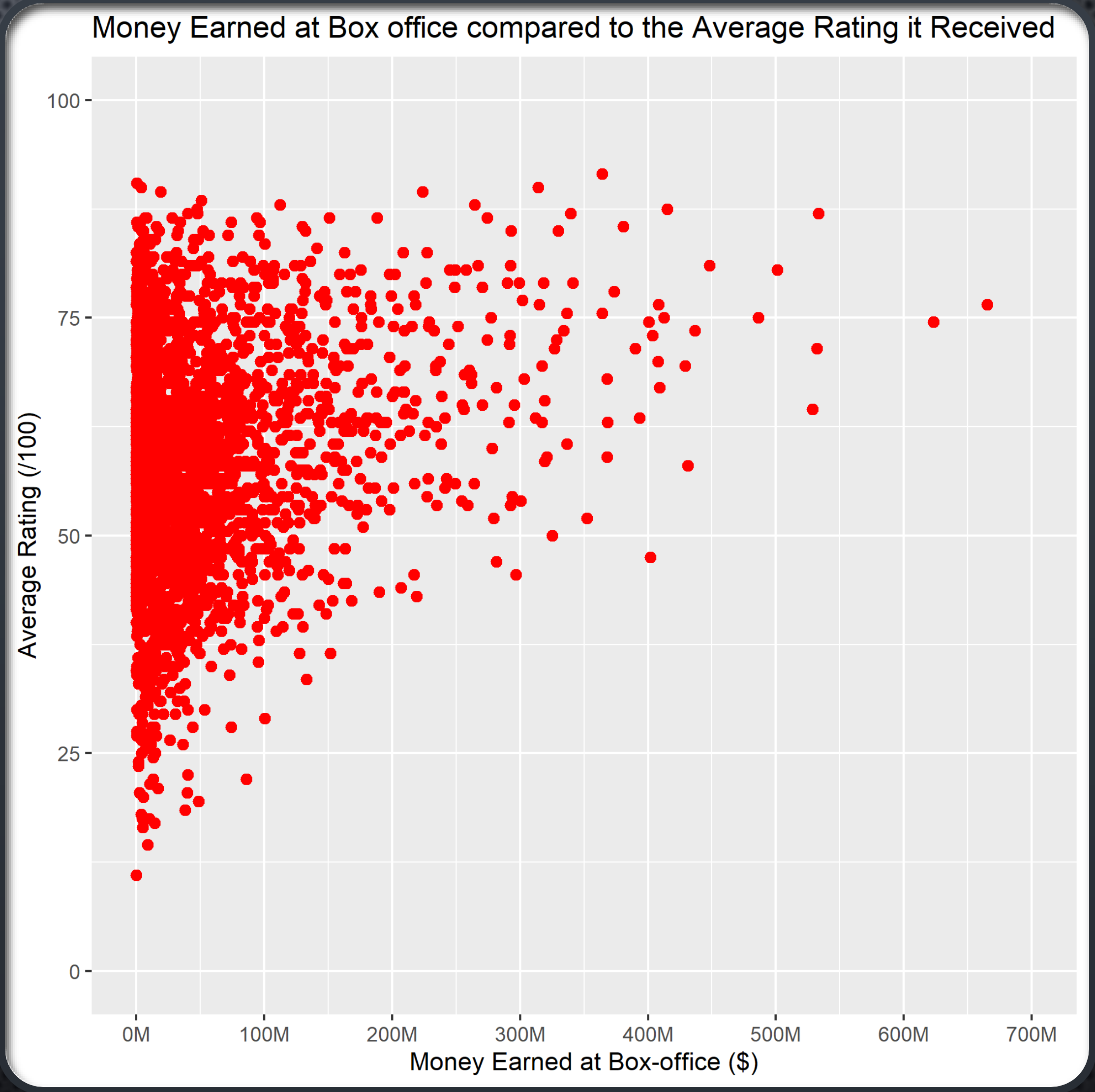
```
get_api_results <- function(api_urls) {
    #Empty list to insert that movie data into.
    api_results <- vector("list", length(api_urls))
    i <- 1

    for (url in api_urls[[1]]) {
        error_ressult = tryCatch({
            api_response <- fromJSON(url)
            movie_data <- data.frame(api_response)[1,]
            if (ncol(movie_data) == 26) {
                api_results[[i]] <- movie_data
                i <- i + 1
            }
        }, error = function(e) {
            #Here we can print the error if we need too
#             print(e)
        }, finally = {
        })
    }

    return(api_results)
}
```

AVERAGE RATING OF A MOVIE BASED ON THE MONEY EARNED AT BOX-OFFICE

- CAN'T SEE MUCH OF A CORRELATION
- MAYBE THAT SOME MOVIES ARE TOO BIG TO FAIL



THANKS FOR LISTENING, ANY QUESTIONS?

Please no questions
we're bad at public
speaking