DATA607: HW2

Neil Shah

2/6/2020

Overview:

In this notebook we will be use relational databases (SQL) and R on survey data. Specifically we will collect, wrangle and then analyze survey data relating to 6 popular movies from a random group of individuals. The overall goal here is to sucessfully pull survey data into a SQL database and then into R, but see if we can uncover any trends. I will also compare my results to the IMDB database.

Experimental Methodology

I will perform the following steps in order

- Survey Design
- Distribute Survey
- Collect Results into a SQL Database/port to R
- Initial Analysis in R
- IMDB Comparison Analysis
- Conclusions
- Recommendations
- Improvements

Survey Design

I created a survey via Google Forms available here (https://forms.gle/YthZSBA92WJ8YLai9) with the following movies.

1. The Irishman
2. The Joker
3. Parasite
4. Once Upon a Time in Hollywood
5. Roma
6. Into the Spiderverse.

Movies were selected from a cursory search of top movies from 2018 onward via Google, strictly chosen by myself.

I proposed a linear scale for each movie from 0 to 5 as required input/question for each participant. I specified that 1 would be defined as “least enjoyed” and 5 “most enjoyed” with 3 being “neutral”. I included 0 as “have not seen” to ensure that all possibilities were covered.

Distibution of Survey

I chose to create a link to said survey and disseminated through 5 WhatsApp groups that I am part of–the groups did not have overlapping members and in sum the potential sample size was 36 unique participants.

I gave a 48 hour cut off time for responses–with the survey being closed on Friday, February 6th 2020 at 12:00 AM EST.

Collect Results into SQL and then R

After the survey closed, the results were exported as a .csv file and imported into SQL.

```
CREATE TABLE data
(id INT NOT NULL AUTO_INCREMENT,
Irishman VARCHAR(255) NOT NULL,
Joker VARCHAR(255) NOT NULL,
Parasite VARCHAR(255) NOT NULL,
Hollywood VARCHAR(255) NOT NULL,
Roma VARCHAR(255) NOT NULL,
Spiderman INT NOT NULL,
 PRIMARY KEY (id));

 LOAD DATA LOCAL INFILE 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\moviedata.csv'
 INTO TABLE data FIELDS TERMINATED BY ','
 LINES TERMINATED BY '\n' IGNORE 1 ROWS
 (id, Irishman, Joker, Parasite, Hollywood, Roma, Spiderman);
```

To interface between MySQL and R I used the package ‘RMySQL’

```
install.packages("RMySQL")
library(RMySQL)
Loading required package: DBI

>  con <- dbConnect(MySQL(),    user='root',    password='password',   dbname='movies', host='lo
calhost')
> con
<MySQLConnection:0,0>
```

Fetching a query

```
> moviequery <- dbSendQuery(con, "SELECT * FROM data")
> dbFetch(moviequery)
   id Irishman Joker Parasite Hollywood Roma Spiderman
1   1        0     2        5         3    0         5
2   2        0     0        0         4    0         4
3   3        4     4        0         0    0         5
4   4        3     0        0         0    1         4
5   5        0     0        0         0    0         0
6   6        0     0        0         0    0         0
7   7        2     5        0         0    4         4
8   8        3     3        0         4    0         5
9   9        0     0        5         0    5         0
10 10        0     0        0         2    1         5
11 11        0     4        5         2    0         3
12 12        1     5        0         0    0         4
13 13        0     5        0         0    0         3
14 14        0     5        5         5    0         0
15 15        0     0        0         0    0         5
16 16        0     4        0         0    0         5
17 17        0     0        0         0    0         4
18 18        5     0        5         5    0         5
```

Another way to demonstrate that I can send queries

```
> dbFetch(dbSendQuery(con, "SELECT Joker FROM data"))
   Joker
1      2
2      0
3      4
4      0
5      0
6      0
7      5
8      3
9      0
10     0
11     4
12     5
13     5
14     5
15     0
16     4
17     0
18     0
```

Initial analysis in R

To analyze the survey and to make my life easier, I'll load these results into a dataframe.

```
> dbClearResult(dbListResults(con)[[1]])
[1] TRUE
> moviequery <- dbSendQuery(con, "SELECT * FROM data")
> df <- dbFetch(moviequery)
> df
   id Irishman Joker Parasite Hollywood Roma Spiderman
1   1        0     2        5         3    0         5
2   2        0     0        0         4    0         4
3   3        4     4        0         0    0         5
4   4        3     0        0         0    1         4
5   5        0     0        0         0    0         0
6   6        0     0        0         0    0         0
7   7        2     5        0         0    4         4
8   8        3     3        0         4    0         5
9   9        0     0        5         0    5         0
10 10        0     0        0         2    1         5
11 11        0     4        5         2    0         3
12 12        1     5        0         0    0         4
13 13        0     5        0         0    0         3
14 14        0     5        5         5    0         0
15 15        0     0        0         0    0         5
16 16        0     4        0         0    0         5
17 17        0     0        0         0    0         4
18 18        5     0        5         5    0         5
> dbListResults(con)
[[1]]
<MySQLResult:0,0,10>

>
> dbDisconnect(con)
[1] TRUE
Warning message:
Closing open result sets
```

Finally-for the fun analysis.

```
> head(df)
  id Irishman Joker Parasite Hollywood Roma Spiderman
1  1        0     2        5         3    0         5
2  2        0     0        0         4    0         4
3  3        4     4        0         0    0         5
4  4        3     0        0         0    1         4
5  5        0     0        0         0    0         0
6  6        0     0        0         0    0         0
> nrows(df)
Error in nrows(df) : could not find function "nrows"
> NROW(df)
[1] 18
```

My survey had a total of 18 observations (survey results)–since I already specified that 0 indicated an unseen movie [this was suprisingl smart on my behalf].

First I'll calculate the rating based on the sum of the rating values divided by the non-zero value.

For example the Irishman:

```
> sum(df$Irishman)/ave(df$Irishman, FUN = function(x) sum(x!=0))[1]
[1] 3
```

Also I thought it would interesting to tally the non-watch (or 0s) count for each movie–Once again using the Irishman.

```
> (NROW(df$Irishman))-(ave(df$Irishman, FUN = function(x) sum(x!=0))[1])
[1] 12
```

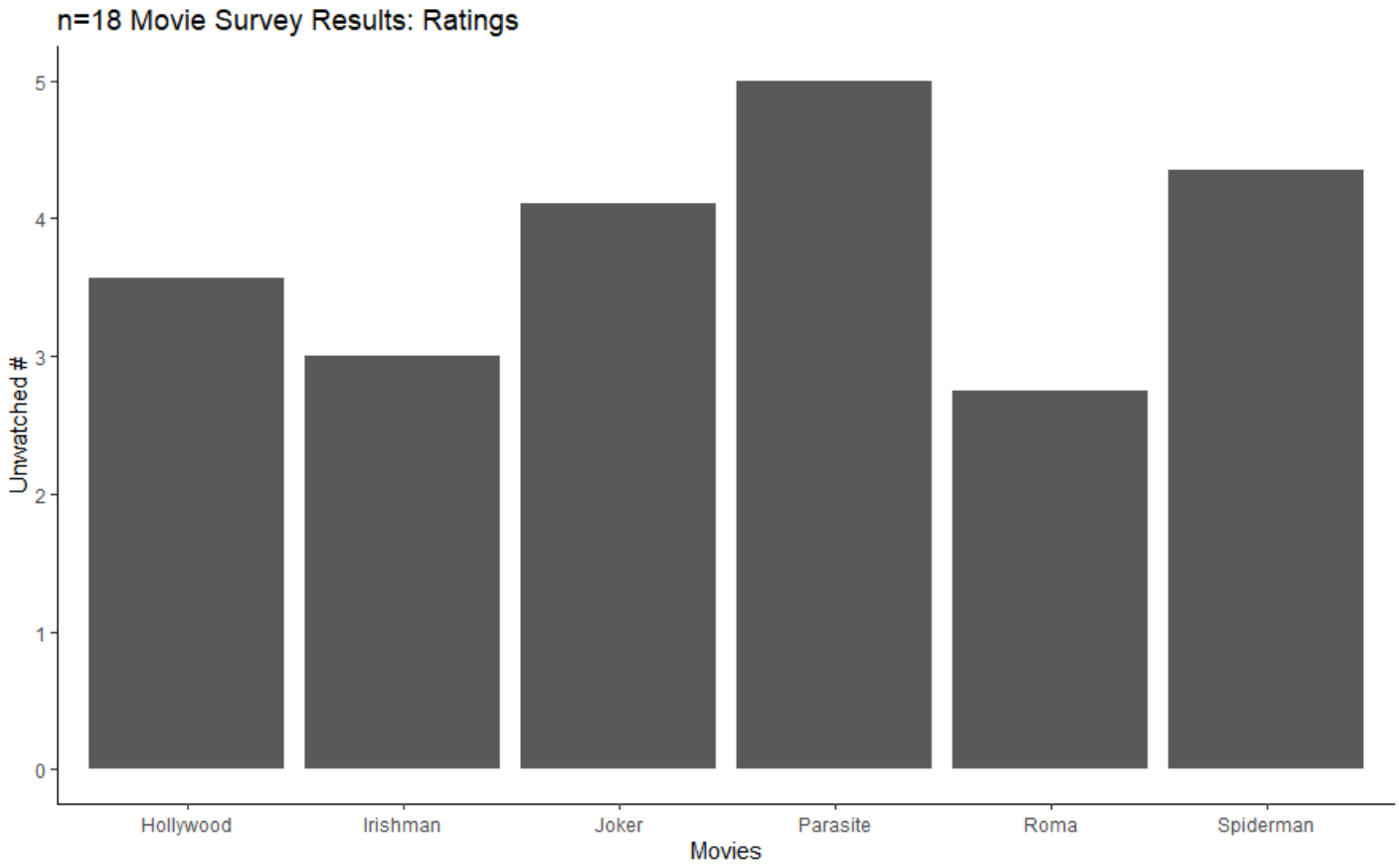
Dividing this result by the number of observations (18) would give us a metric of non-watch percentage.

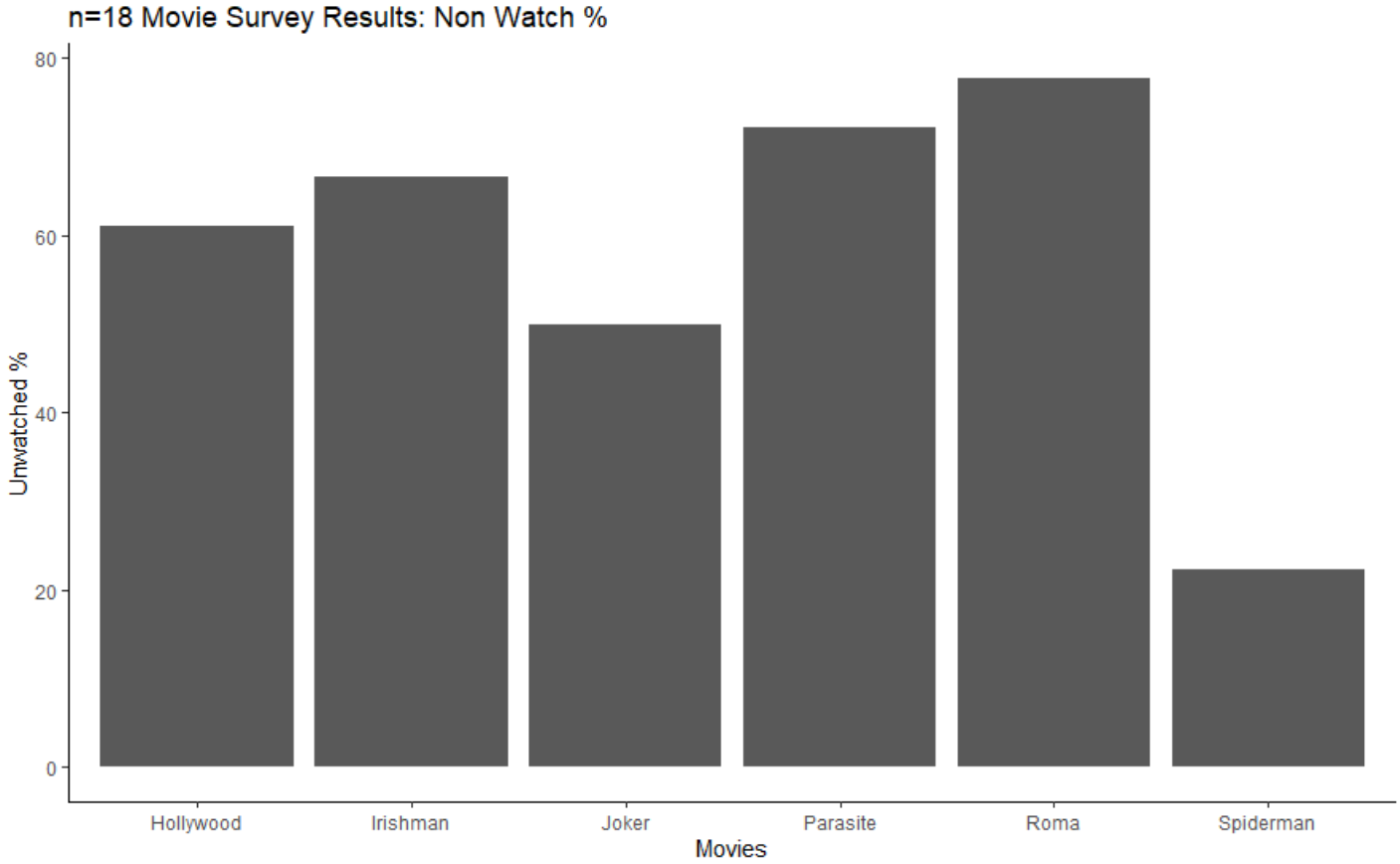
Collecting these all in a dataframe…

```
> surveyratings <- c((sum(df$Irishman)/ave(df$Irishman, FUN = function(x) sum(x!=0))[1]),(sum(df
$Joker/ave(df$Joker, FUN = function(x) sum(x!=0))[1])),(sum(df$Parasite)/ave(df$Parasite, FUN =
function(x) sum(x!=0))[1]),(sum(df$Hollywood)/ave(df$Hollywood, FUN = function(x) sum(x!=0))
[1]),(sum(df$Roma)/ave(df$Roma, FUN = function(x) sum(x!=0))[1]),(sum(df$Spiderman)/ave(df$Spide
rman, FUN = function(x) sum(x!=0))[1]))
> surveyratings
[1] 3.000000 4.111111 5.000000 3.571429 2.750000 4.357143

> surveynonwatch <-c(((NROW(df$Irishman))-(ave(df$Irishman, FUN = function(x) sum(x!=0))[1])),
((NROW(df$Joker))-(ave(df$Joker, FUN = function(x) sum(x!=0))[1])),((NROW(df$Parasite))-(ave(df
$Parasite, FUN = function(x) sum(x!=0))[1])),((NROW(df$Hollywood))-(ave(df$Hollywood, FUN = func
tion(x) sum(x!=0))[1])),((NROW(df$Roma))-(ave(df$Roma, FUN = function(x) sum(x!=0))[1])),((NROW
(df$Spiderman))-(ave(df$Spiderman, FUN = function(x) sum(x!=0))[1])))
> surveynonwatch
[1] 12  9 13 11 14  4
> surveynonwatchpct <-(surveynonwatch/18)*100
> surveynonwatchpct
[1] 66.66667 50.00000 72.22222 61.11111 77.77778 22.22222

> surveyDF <- data.frame('Movies'=titles,'SurveyRating'=surveyratings,'Nonwatch'=surveynonwatc
h,'NonWatch%'=surveynonwatchpct)
> surveyDF
     Movies SurveyRating Nonwatch NonWatch.
1  Irishman     3.000000       12  66.66667
2     Joker     4.111111        9  50.00000
3  Parasite     5.000000       13  72.22222
4 Hollywood     3.571429       11  61.11111
5      Roma     2.750000       14  77.77778
6 Spiderman     4.357143        4  22.22222
```

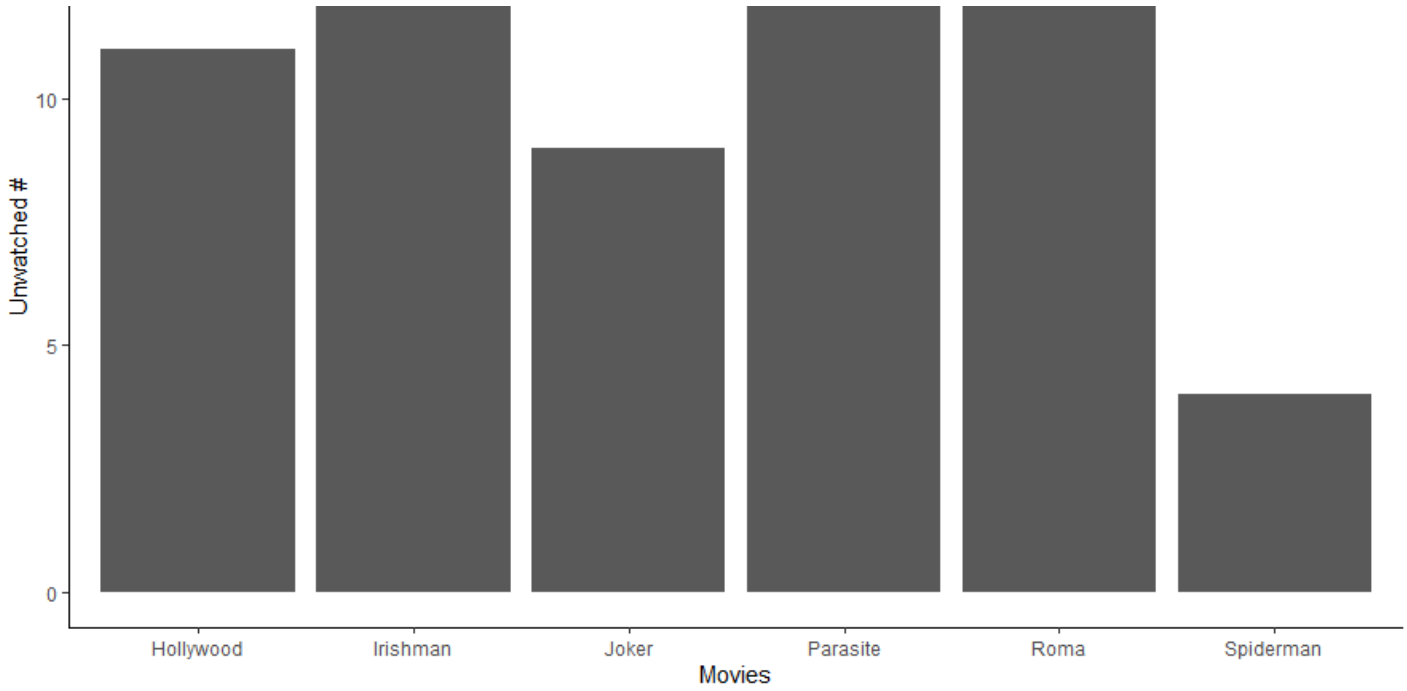
Putting it all together in nice Data: Ink ratio form

```
> ggplot(surveyDF,aes(y=surveyDF$NonWatch.,x=surveyDF$Movies))+geom_bar(position="dodge", stat
="identity") + labs(x='Movies',y='Unwatched %',title='n=18 Movie Survey Results: Non Watch Perce
nt') + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
+ panel.background = element_blank(), axis.line = element_line(colour = "black"))

> ggplot(surveyDF,aes(y=surveyDF$NonWatch,x=surveyDF$Movies))+geom_bar(position="dodge", stat="i
dentity") + labs(x='Movies',y='Unwatched #',title='n=18 Movie Survey Results: Non Watch #') + th
eme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
+ panel.background = element_blank(), axis.line = element_line(colour = "black"))
> ggplot(surveyDF,aes(y=surveyDF$SurveyRating,x=surveyDF$Movies))+geom_bar(position="dodge", sta
t="identity") + labs(x='Movies',y='Unwatched #',title='n=18 Movie Survey Results: Ratings') + th
eme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
+ panel.background = element_blank(), axis.line = element_line(colour = "black"))
```


n=18 Movie Survey Results: Ratings
Unwatched #
0
1
2
3
4
5
Hollywood
Irishman
Joker
Parasite
Roma
Spiderman
Movies


n=18 Movie Survey Results: Non Watch %
Unwatched %
0
20
40
60
80
Hollywood
Irishman
Joker
Parasite
Roma
Spiderman
Movies


n=18 Movie Survey Results: Non Watch #



So here are the some intitial thoughts I can draw from my survey

1. Parasite was rated the highest movie (5/5), followed by Spiderman, The Joker, Once Upon a Time in Hollywood, the Irishman and finally Roma.
2. However–save for Spiderman, most of the participants (over 50%) have not seen any of the movies! While Spiderman has the 2nd highest rating it was the most seen movie!

IMDB Comparision Analysis

IMDB (https://www.imdb.com/?ref_=nv_home) is a popular database that contains granular data about movies, such as director, genre, cast and ratings. I thought it would be interesting to use this database to map metadata within my survey, and see if there were any other trends we could discover.

I consulted the API documentation (https://cran.r-project.org/web/packages/imdbapi/index.html), registered for an API key and then installed the library.

```
library('imdbapi')
#Setting key to save me time
key =*****[intentionally blurred out by myself]
```

The IMDBI can search it's database via Title or a unique ID. While I could use the titles from my dataframe as an argument for the title search, I felt it would be easier (it's only 6 movies) to hard code the title IDs for easier search. I consulted IMDB url to generate my title ID keys.

```
JokerID='tt7286456'
> RomaID='tt6155172'
> SpidermanID='tt4633694'
> HollywoodID='tt7131622'
> IrishmanID='tt1302006'
> ParasiteID='tt6751668'
```

Building IMDB Database

Let's look at an example of an IMDB pull–I will store the metadata within a dataframe for each movie.

```
RomaDF <- find_by_id(RomaID,api_key=key)
> head(RomaDF)
# A tibble: 2 x 25
  Title Year  Rated Released   Runtime Genre Director Writer Actors Plot  Language Country Award
s Poster Ratings
  <chr> <chr> <chr> <date>     <chr>   <chr> <chr>    <chr>  <chr>  <chr> <chr>    <chr>   <chr>
<chr>  <list>
1 Roma  2018  R     2018-11-21 135 min Drama Alfonso~ Alfon~ Yalit~ A ye~ Spanish~ Mexico  Won 3
~ https~ <named~
2 Roma  2018  R     2018-11-21 135 min Drama Alfonso~ Alfon~ Yalit~ A ye~ Spanish~ Mexico  Won 3
~ https~ <named~
# ... with 10 more variables: Metascore <chr>, imdbRating <dbl>, imdbVotes <dbl>, imdbID <chr>,
Type <chr>,
#   DVD <date>, BoxOffice <chr>, Production <chr>, Website <chr>, Response <chr>
>
```

Repeating this for all the other movies

```
HollywoodDF <- find_by_id(HollywoodID,api_key=key)
> IrishDF <- find_by_id(IrishmanID,api_key=key)
> ParasiteDF <- find_by_id(ParasiteID,api_key=key)
> SpidermanDF <- find_by_id(SpidermanID,api_key=key)
```

Awesome–now let's look dig into the granularity to see what data is displayed.

```
> RomaDF$Awards
[1] "Won 3 Oscars. Another 238 wins & 198 nominations." "Won 3 Oscars. Another 238 wins & 198 no
minations."
> RomaDF$Ratings
[[1]]
[[1]]$Source
[1] "Internet Movie Database"

[[1]]$Value
[1] "7.7/10"


[[2]]
[[2]]$Source
[1] "Metacritic"

[[2]]$Value
[1] "96/100"


> RomaDF$Genre
[1] "Drama" "Drama"
> RomaDF$Runtime
[1] "135 min" "135 min"
```

As we can see, the IMDB database has a wealth of information! For this project I am going to do a comparasion of my survey's ratings and IMDB; specifically when looking at genre and runtime.

If we look at the IMDB rating

```
> RomaDF$Ratings[[1]][2]
$Value
[1] "7.7/10"
```

It's based of a 1-10 point scale–while I used a 1-5 point. I'll transform the IMDB rating to a comparative rating by dividing by 2.0.

TO do this I'll do the following:

1. Pull the IMDB rating from the Ratings column
2. Split the string across the '/'
3. Convert the string to a float
4. Divide by 2

In action in one line of code

```
> RomaRating <- as.numeric(strsplit((RomaDF$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0
> RomaRating
[1] 3.85
```

Repeating the above for each movie and making a vector called ratings.

```
> ratings <- c((as.numeric(strsplit((IrishDF$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0),(as.numer
ic(strsplit((JokerDF$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0),(as.numeric(strsplit((ParasiteDF
$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0),(as.numeric(strsplit((HollywoodDF$Ratings[[1]][2]$Val
ue),'/')[[1]][1])/2.0),(as.numeric(strsplit((RomaDF$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0),(a
s.numeric(strsplit((SpidermanDF$Ratings[[1]][2]$Value),'/')[[1]][1])/2.0))
> ratings
[1] 4.00 4.30 4.30 3.90 3.85 4.20
```

The previous method can be also be used to create vectors for genre–note i'll only be taking the first genre definer.

```
> genre <- c((strsplit(IrishDF$Genre[1],',')[[1]][1]),(strsplit(JokerDF$Genre[1],',')[[1]][1]),
(strsplit(ParasiteDF$Genre[1],',')[[1]][1]),(strsplit(HollywoodDF$Genre[1],',')[[1]][1]),(strspl
it(RomaDF$Genre[1],',')[[1]][1]),(strsplit(SpidermanDF$Genre[1],',')[[1]][1]))
>
> genre
[1] "Biography" "Crime"     "Comedy"    "Comedy"    "Drama"     "Animation"
```

Finally the same for runtimes

```
> runtimes <- c((strsplit(IrishDF$Runtime[1],' ')[[1]][1]),(strsplit(JokerDF$Runtime[1],' ')
[[1]][1]),(strsplit(ParasiteDF$Runtime[1],' ')[[1]][1]),(strsplit(HollywoodDF$Runtime[1],' ')
[[1]][1]),(strsplit(RomaDF$Runtime[1],' ')[[1]][1]),(strsplit(SpidermanDF$Runtime[1],' ')[[1]]
[1]))
> runtimes
[1] "209" "122" "132" "161" "135" "117"
```

Great–now combining these all in one IMDB dataframe.

```
> IMDBDF <-data.frame('Movies'=titles,'Run Time'=runtimes,'IMDBRatings'=ratings,'Genre'=genre)
> IMDBDF
     Movies Run.Time IMDBRatings     Genre
1  Irishman      209        4.00 Biography
2     Joker      122        4.30     Crime
3  Parasite      132        4.30    Comedy
4 Hollywood      161        3.90    Comedy
5      Roma      135        3.85     Drama
6 Spiderman      117        4.20 Animation
```

Now lets add our data to the IMDB data.

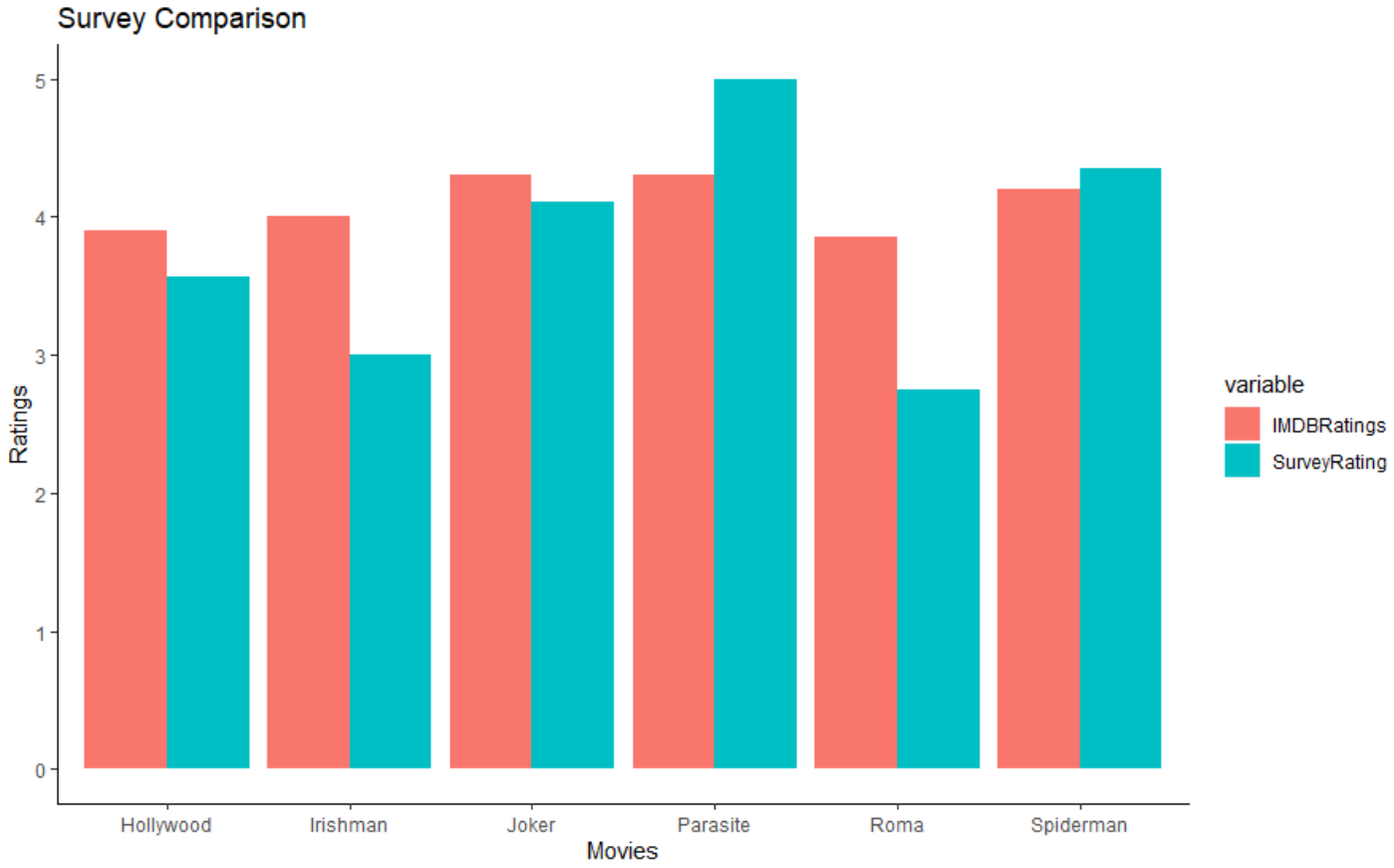
```
> IMDBDF$SurveyRating <- surveyDF$SurveyRating
> IMDBDF
     Movies Run.Time IMDBRatings     Genre SurveyRating
1  Irishman      209        4.00 Biography     3.000000
2     Joker      122        4.30     Crime     4.111111
3  Parasite      132        4.30    Comedy     5.000000
4 Hollywood      161        3.90    Comedy     3.571429
5      Roma      135        3.85     Drama     2.750000
6 Spiderman      117        4.20 Animation     4.357143
```

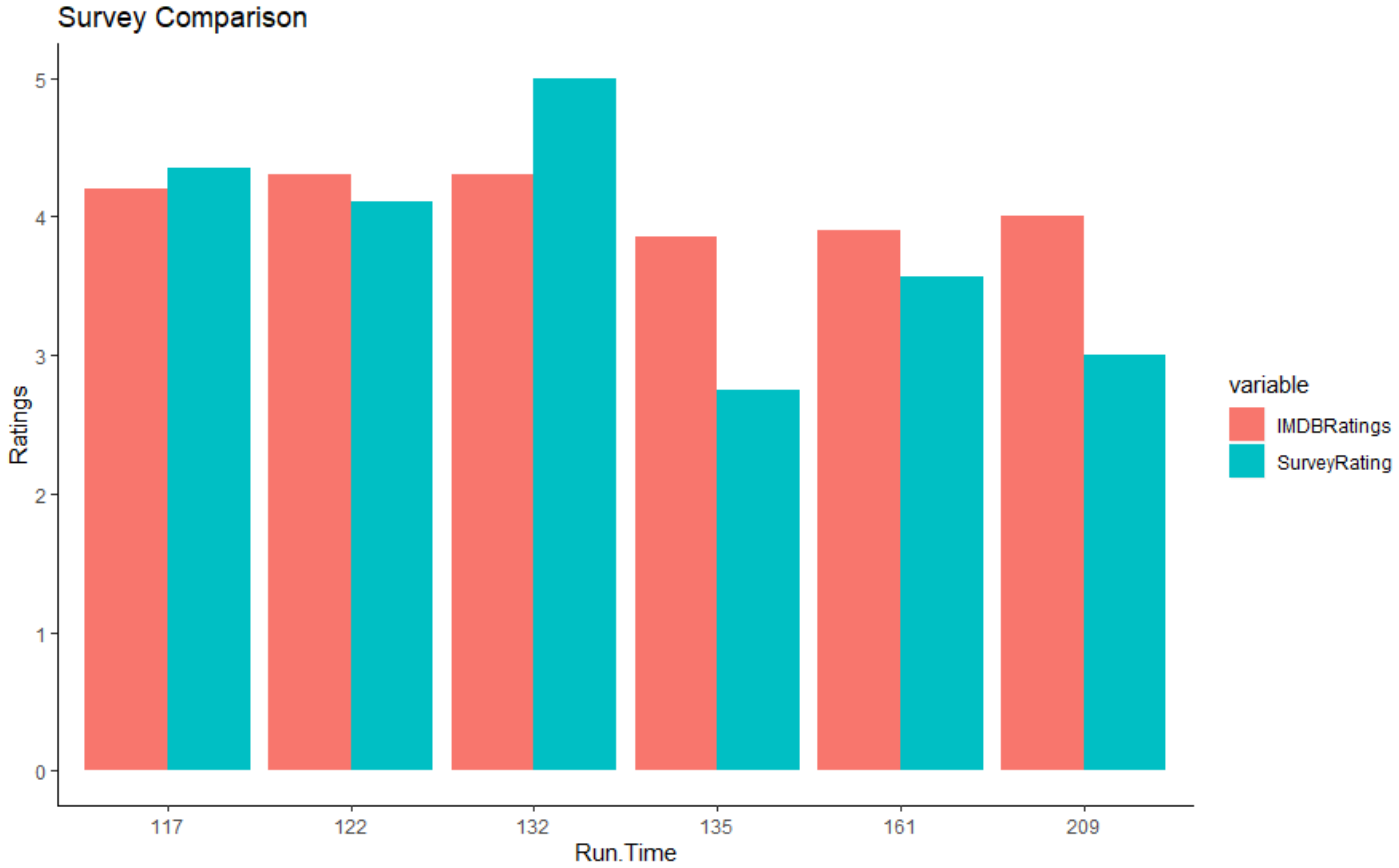
To compare my Survey to IMDB side by side I will reshape my dataframe.

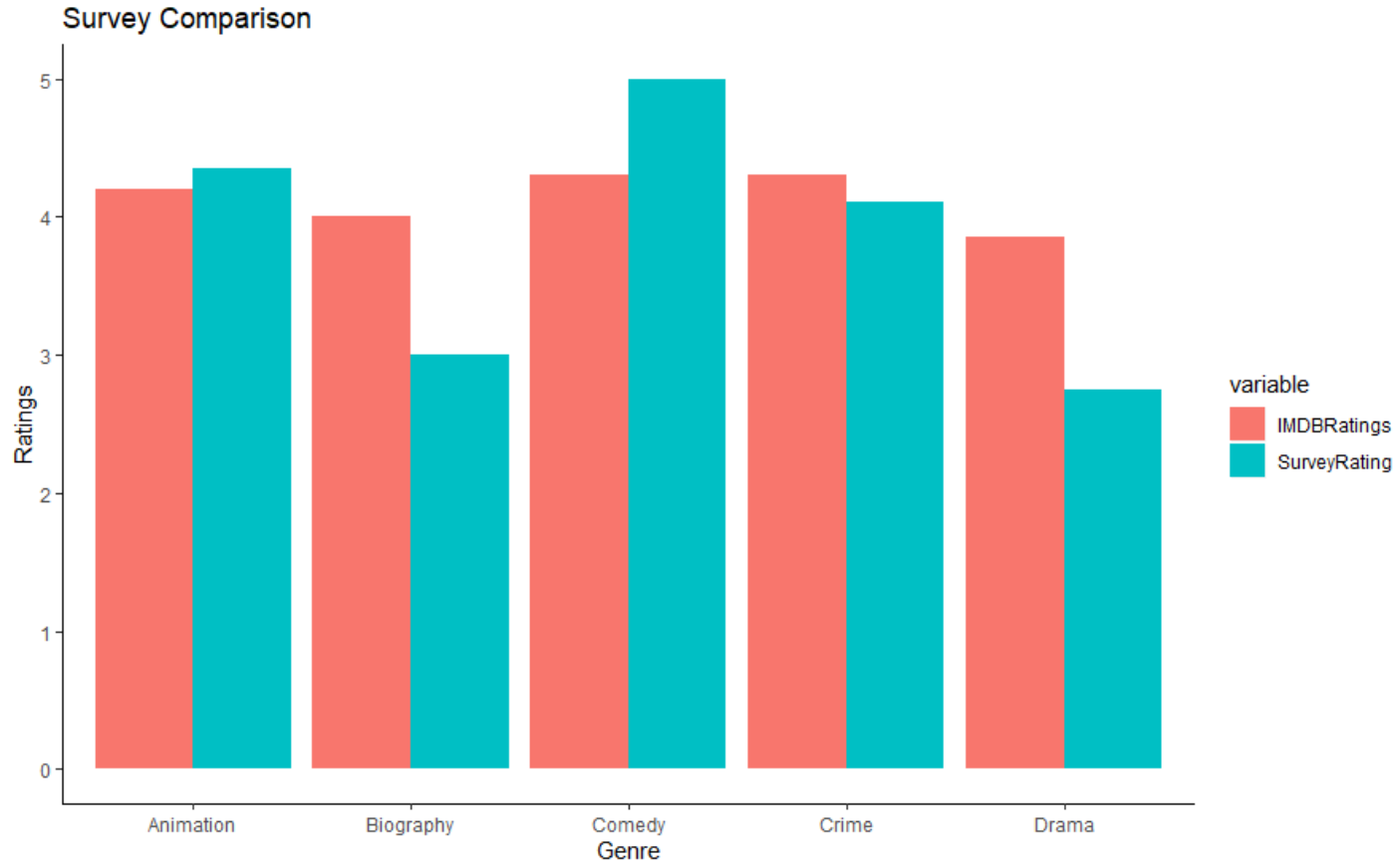
```
> IMDBDFshaped <-melt(IMDBDF)
Using Movies, Run.Time, Genre as id variables
```

Finally plotting all–I decided to keep color for visibility

```
> ggplot(IMDBDFshaped,aes(Genre,value,fill=variable))+
+      geom_bar(stat="identity",position="dodge") +labs(title='Survey Comparison', y='Ratings')+
theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), panel.background =
element_blank(), axis.line = element_line(colour = "black"))
> ggplot(IMDBDFshaped,aes(Run.Time,value,fill=variable))+
+      geom_bar(stat="identity",position="dodge") +labs(title='Survey Comparison', y='Ratings')+
theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), panel.background =
element_blank(), axis.line = element_line(colour = "black"))
> ggplot(IMDBDFshaped,aes(Movies,value,fill=variable))+
+      geom_bar(stat="identity",position="dodge") +labs(title='Survey Comparison', y='Ratings')+
theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), panel.background =
element_blank(), axis.line = element_line(colour = "black"))
```


Survey Comparison
Ratings
0
1
2
3
4
5
Hollywood
Irishman
Joker
Parasite
Roma
Spiderman
Movies
variable
IMDBRatings
SurveyRating


Survey Comparison
Ratings
0
1
2
3
4
5
117
122
132
135
161
209
Run.Time
variable
IMDBRatings
SurveyRating



Genre Comparison

Conclusions

1. IMDB users ranked via rating Parasite and Joker tied, followed by Spiderman, Irishman, Once Upon a time in Hollywood and finally Roma last.
2. My survey ranked via rating Parasite, Spiderman, Joker, Once Upon a time in Hollywood, the Irishman and finally Roma last.
3. IMDB/My Survey–both agreed on their love for Parasite–however my Survey rated the movie on a higher scale (this can be due to the smaller size.) Similiary both Parasite and Spiderman were ranked higher by my Survey than by IMDB.
4. Both surveys ranked Roma last–however my Survey disproportionaly did not like it (larger spread between ratings)
5. When looking at length (runtime) making an impact–longer movies received lower ratings (the 161 runtime and 209) however the 135 runtime movie (Roma) was universally unliked despite it's midrange lenght.
6. Spiderman and the Joker has the most narrow rating spread–they were both ranked similarly (and within top half of movies)
7. My Survey group enjoys Comedies the most followed by Animation, Crime, Biography and then Drama. IMDB users had a similar profile.
8. Don't watch Roma?

Improvements

1. **Unique identifers:** To elminate the possibility of multiple-sampling, a unique identifier (email address) could be used for each survey participant. While this might increase the non-response rate (individauls who want to remain anonymous), it also elimninates the possibility of multiple entries.
2. **Random selection of movies:** I chose the movies randomly but there can be internal biases–perhaps I chose movies that only I saw subconciously. I propose next time that a list of popular movies (metric could be rating, boxoffice values or weeks on top list), inserted into an array and then 6 are randomly chosen to sample. This could eliminate any selection bias .
3. **Random selection of genres/medium:** Similar to #2–perhaps using top movies from each genre would be a better way to gain diverisity/sample said movies. In addition including Netflix, Amazon Prime and other media would be interesting as well.
4. **More metrics on movie user:** Additional of questions on frequency of movies watched or favorite genre could reveal more themes or trends.
5. **Alternative Samplng:** Since I sent the survey through my own WhatsApp group, only friends and families saw said survey. If resources were available an internet poll or through CUNY could yield a large and diverse sample.