Hw11a - Movie Data: scraping and graphing

Nikhil Kotecha

11/18/2017

Source: https://www.analyticsvidhya.com/blog/2017/03/beginners-guide-on-web-scraping-in-r-using-rvest-with-hands-on-kno

Scraping Movie Data from IMDB

Scrapping movie data from imdb. Doing some pre-processing - e.g. looking at title data, ratings, genre, actors / actresses

Pre-processing

Doing some pre-processing - e.g. cleaning title data, ratings, genre, actors / actresses - to amke it easier for analysis.

```
#Using CSS selectors to scrap the rankings section
rank_data_html <- html_nodes(webpage,'.text-primary')

#Converting the ranking data to text
rank_data <- html_text(rank_data_html)

#Let's have a look at the rankings
head(rank_data)
```

```
## [1] "1." "2." "3." "4." "5." "6."
```

```
#Data-Preprocessing: Converting rankings to numerical
rank_data<-as.numeric(rank_data)

#Let's have another look at the rankings
head(rank_data)
```

```
## [1] 1 2 3 4 5 6
```

```
#Using CSS selectors to scrap the title section
title_data_html <- html_nodes(webpage,'.lister-item-header a')

#Converting the title data to text
title_data <- html_text(title_data_html)

#Let's have a look at the title
head(title_data)
```

```
## [1] "Split"         "Sing"          "Bad Moms"      "Suicide Squad"
## [5] "Moana"         "LBJ"
```

```
#Using CSS selectors to scrap 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)
```

```
#Let's have a look at the description data
head(description_data)
```

```
## [1] "\nThree girls are kidnapped by a man with a diagnosed 23 distinct personalities. They must try t
## [2] "\nIn a city of humanoid animals, a hustling theater impresario's attempt to save his theater wit
## [3] "\nWhen three overworked and under-appreciated moms are pushed beyond their limits, they ditch th
## [4] "\nA secret government agency recruits some of the most dangerous incarcerated super-villains to
## [5] "\nIn Ancient Polynesia, when a terrible curse incurred by the Demigod Maui reaches Moana's islan
## [6] "\nThe story of U.S. President Lyndon Baines Johnson from his young days in West Texas to the Whi
```

```
#Data-Preprocessing: removing '\n'
description_data<-gsub("\n","",description_data)

#Let's have another look at the description data
head(description_data)
```

```
## [1] "Three girls are kidnapped by a man with a diagnosed 23 distinct personalities. They must try to
## [2] "In a city of humanoid animals, a hustling theater impresario's attempt to save his theater with
## [3] "When three overworked and under-appreciated moms are pushed beyond their limits, they ditch thei
## [4] "A secret government agency recruits some of the most dangerous incarcerated super-villains to fo
## [5] "In Ancient Polynesia, when a terrible curse incurred by the Demigod Maui reaches Moana's island
## [6] "The story of U.S. President Lyndon Baines Johnson from his young days in West Texas to the White
```

```
#Using CSS selectors to scrap the Movie runtime section
runtime_data_html <- html_nodes(webpage,'.text-muted .runtime')

#Converting the runtime data to text
runtime_data <- html_text(runtime_data_html)

#Let's have a look at the runtime
head(runtime_data)
```

```
## [1] "117 min" "108 min" "100 min" "123 min" "107 min" "98 min"
```

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

runtime_data<-gsub(" min","",runtime_data)
runtime_data<-as.numeric(runtime_data)

#Let's have another look at the runtime data
head(rank_data)
```

```
## [1] 1 2 3 4 5 6
```

```
#Using CSS selectors to scrap the Movie genre section
genre_data_html <- html_nodes(webpage,'.genre')

#Converting the genre data to text
genre_data <- html_text(genre_data_html)

#Let's have a look at the runtime
head(genre_data)
```

```
## [1] "\nHorror, Thriller            "
## [2] "\nAnimation, Comedy, Family            "
## [3] "\nComedy            "
```

```
## [4] "\nAction, Adventure, Fantasy            "
## [5] "\nAnimation, Adventure, Comedy            "
## [6] "\nBiography, Drama            "
```

```
#Data-Preprocessing: removing \n
genre_data<-gsub("\n","",genre_data)

#Data-Preprocessing: removing excess spaces
genre_data<-gsub(" ","",genre_data)

#taking only the first genre of each movie
genre_data<-gsub(",.*","",genre_data)

#Convering each genre from text to factor
genre_data<-as.factor(genre_data)

#Let's have another look at the genre data
head(genre_data)
```

```
## [1] Horror    Animation Comedy    Action    Animation Biography
## 9 Levels: Action Adventure Animation Biography Comedy Crime ... Thriller
```

```
#Using CSS selectors to scrap the IMDB rating section
rating_data_html <- html_nodes(webpage,'.ratings-imdb-rating strong')

#Converting the ratings data to text
rating_data <- html_text(rating_data_html)

#Let's have a look at the ratings
head(rating_data)
```

```
## [1] "7.3" "7.1" "6.2" "6.2" "7.6" "6.0"
```

```
#Data-Preprocessing: converting ratings to numerical
rating_data<-as.numeric(rating_data)

#Let's have another look at the ratings data
head(rating_data)
```

```
## [1] 7.3 7.1 6.2 6.2 7.6 6.0
```

```
#Using CSS selectors to scrap the votes section
votes_data_html <- html_nodes(webpage,'.sort-num_votes-visible span:nth-child(2)')

#Converting the votes data to text
votes_data <- html_text(votes_data_html)

#Let's have a look at the votes data
head(votes_data)
```

```
## [1] "226,440" "85,952"  "79,514"  "445,169" "164,454" "617"
```

```
#Data-Preprocessing: removing commas
votes_data<-gsub(",","",votes_data)

#Data-Preprocessing: converting votes to numerical
votes_data<-as.numeric(votes_data)
```

```
#Let's have another look at the votes data
head(votes_data)
```

```
## [1] 226440  85952  79514 445169 164454    617
```

```
#Using CSS selectors to scrap the directors section
directors_data_html <- html_nodes(webpage,'.text-muted+ p a:nth-child(1)')

#Converting the directors data to text
directors_data <- html_text(directors_data_html)

#Let's have a look at the directors data
head(directors_data)
```

```
## [1] "M. Night Shyamalan" "Garth Jennings"     "Jon Lucas"
## [4] "David Ayer"         "Ron Clements"       "Rob Reiner"
```

```
#Data-Preprocessing: converting directors data into factors
directors_data<-as.factor(directors_data)

#Using CSS selectors to scrap the actors section
actors_data_html <- html_nodes(webpage,'.lister-item-content .ghost+ a')

#Converting the gross actors data to text
actors_data <- html_text(actors_data_html)

#Let's have a look at the actors data
head(actors_data)
```

```
## [1] "James McAvoy"         "Matthew McConaughey"  "Mila Kunis"
## [4] "Will Smith"           "Auli'i Cravalho"      "Jennifer Jason Leigh"
```

```
#Data-Preprocessing: converting actors data into factors
actors_data<-as.factor(actors_data)
```

Let's look at Metadata

```
#Using CSS selectors to scrap 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)
```

```
## [1] "62        " "59        " "60        " "40        " "81        "
## [6] "54        "
```

```
#Data-Preprocessing: removing extra space in metascore
metascore_data<-gsub(" ","",metascore_data)

#Lets check the length of metascore data
length(metascore_data)
```

```
## [1] 96
```

```
# adding some NA's to clean data
########for (i in c(39,73,80,89)){

#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)

#}

#Data-Preprocessing: converting metascore to numerical
metascore_data<-as.numeric(metascore_data)

#Let's have another look at length of the metascore data

length(metascore_data)
```

```
## [1] 96
```

```
for (i in c(39,73,80,89)){

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)

}

#Data-Preprocessing: converting metascore to numerical
metascore_data<-as.numeric(metascore_data)
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
#Let's have another look at length of the metascore data

length(metascore_data)
```

```
## [1] 100
```

```
#Let's look at summary statistics
summary(metascore_data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   21.00   47.00   60.00   59.41   71.25   99.00       4
```

Let's take a look at movie earnings (gross)

```
#Using CSS selectors to scrap the gross revenue section
gross_data_html <- html_nodes(webpage,'.ghost~ .text-muted+ span')

#Converting the gross revenue data to text
gross_data <- html_text(gross_data_html)

#Let's have a look at the votes data
head(gross_data)
```

```
## [1] "$138.14M" "$270.33M" "$113.26M" "$325.10M" "$248.76M" "$100.55M"
```

```
#Data-Preprocessing: removing '$' and 'M' signs
gross_data<-gsub("M","",gross_data)

gross_data<-substring(gross_data,2,6)

#Let's check the length of gross data
length(gross_data)
```

```
## [1] 87
```

```
#Filling missing entries with NA
for (i in c(39,49,52,57,64,66,73,76,77,80,87,88,89)){
#17
a<-gross_data[1:(i-1)]

b<-gross_data[i:length(gross_data)]

gross_data<-append(a,list("NA"))

gross_data<-append(gross_data,b)

}

#Data-Preprocessing: converting gross to numerical
gross_data<-as.numeric(gross_data)
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
#Let's have another look at the length of gross data
length(gross_data)
```

```
## [1] 100
```

```
summary(gross_data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.18   18.25   57.64   99.41  125.80  532.10      13
```

11 Features for the 100 most popular feature films released in 2016.

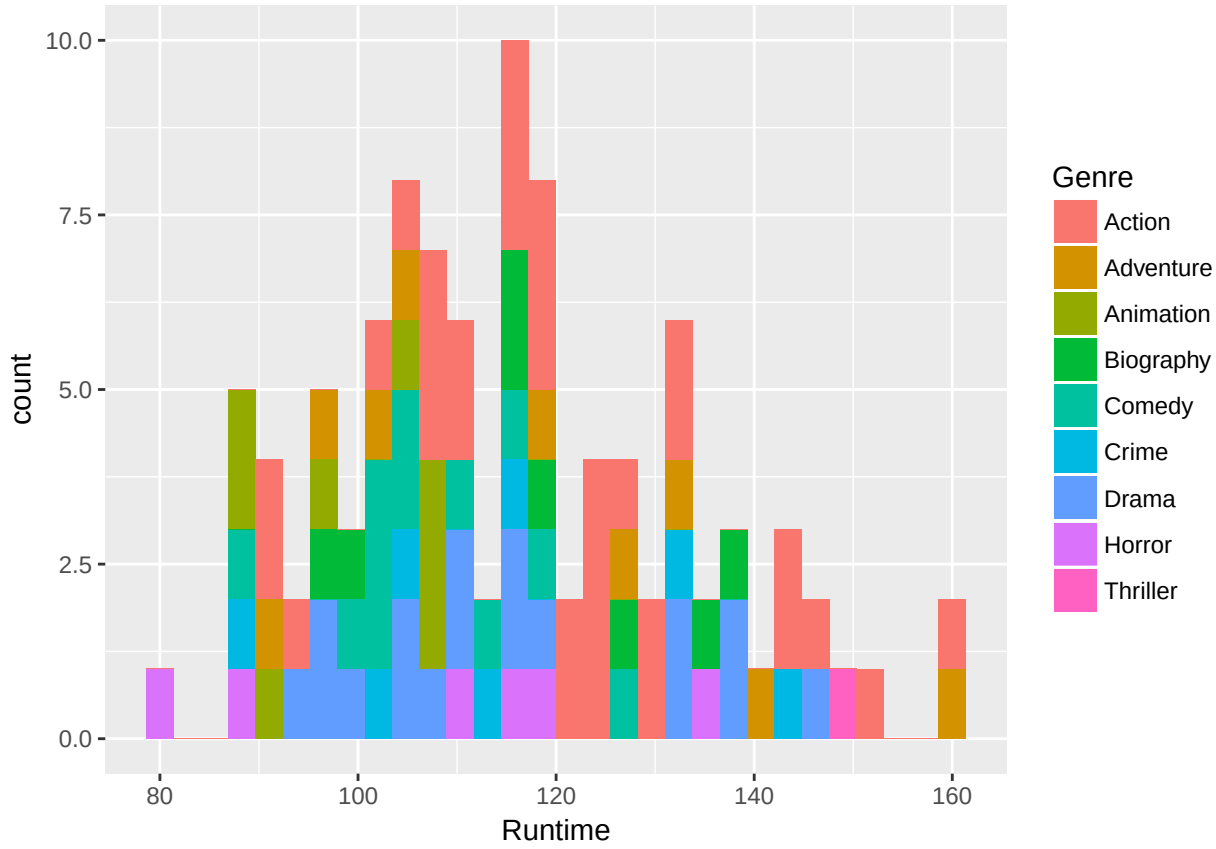
```
#Combining all the lists to form a data frame
movies_df<-data.frame(Rank = rank_data, Title = title_data,

Description = description_data, Runtime = runtime_data,

Genre = genre_data, Rating = rating_data,

Metascore = metascore_data, Votes = votes_data,

Director = directors_data, Actor = actors_data)

#Structure of the data frame

str(movies_df)
```

```
## 'data.frame':    100 obs. of  11 variables:
##  $ Rank               : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ Title              : Factor w/ 100 levels "10 Cloverfield Lane",..: 74 72 12 76 59 50 10 28 69 19
##  $ Description        : Factor w/ 100 levels "A blind woman's relationship with her husband changes
##  $ Runtime            : num  117 108 100 123 107 98 116 115 133 147 ...
##  $ Genre              : Factor w/ 9 levels "Action","Adventure",..: 8 3 5 1 3 4 7 1 1 1 ...
##  $ Rating             : num  7.3 7.1 6.2 6.2 7.6 6 8 7.5 7.9 7.9 ...
##  $ Metascore          : num  62 59 60 40 81 54 81 72 65 75 ...
##  $ Votes              : num  226440 85952 79514 445169 164454 ...
##  $ Gross_Earning_in_Mil: num  138 270 113 325 249 ...
##  $ Director           : Factor w/ 98 levels "Aisling Walsh",..: 57 33 48 21 80 76 26 82 31 5 ...
##  $ Actor              : Factor w/ 88 levels "Aamir Khan","Addison Timlin",..: 36 55 59 86 8 38 4 10
```

Let's plot

Ugly first attempt to get a sense of the data: Based on the above data, which movie from which Genre had the longest runtime?

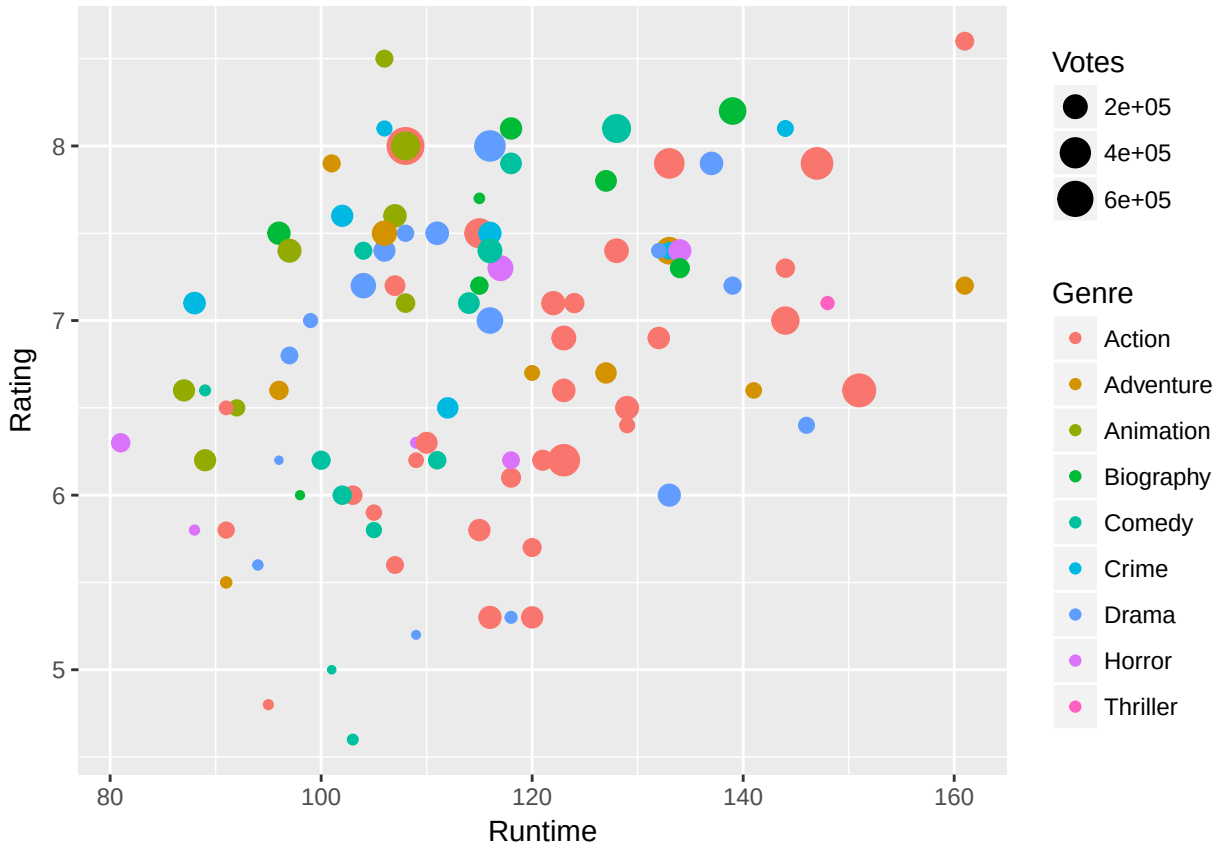
```
library('ggplot2')

qplot(data = movies_df,Runtime,fill = Genre,bins = 30)
```




```
#stack, color, break apart
#scatter, uncolor points
```

Try a scatter plot: In the Runtime of 130-160 mins, which genre has the highest votes?

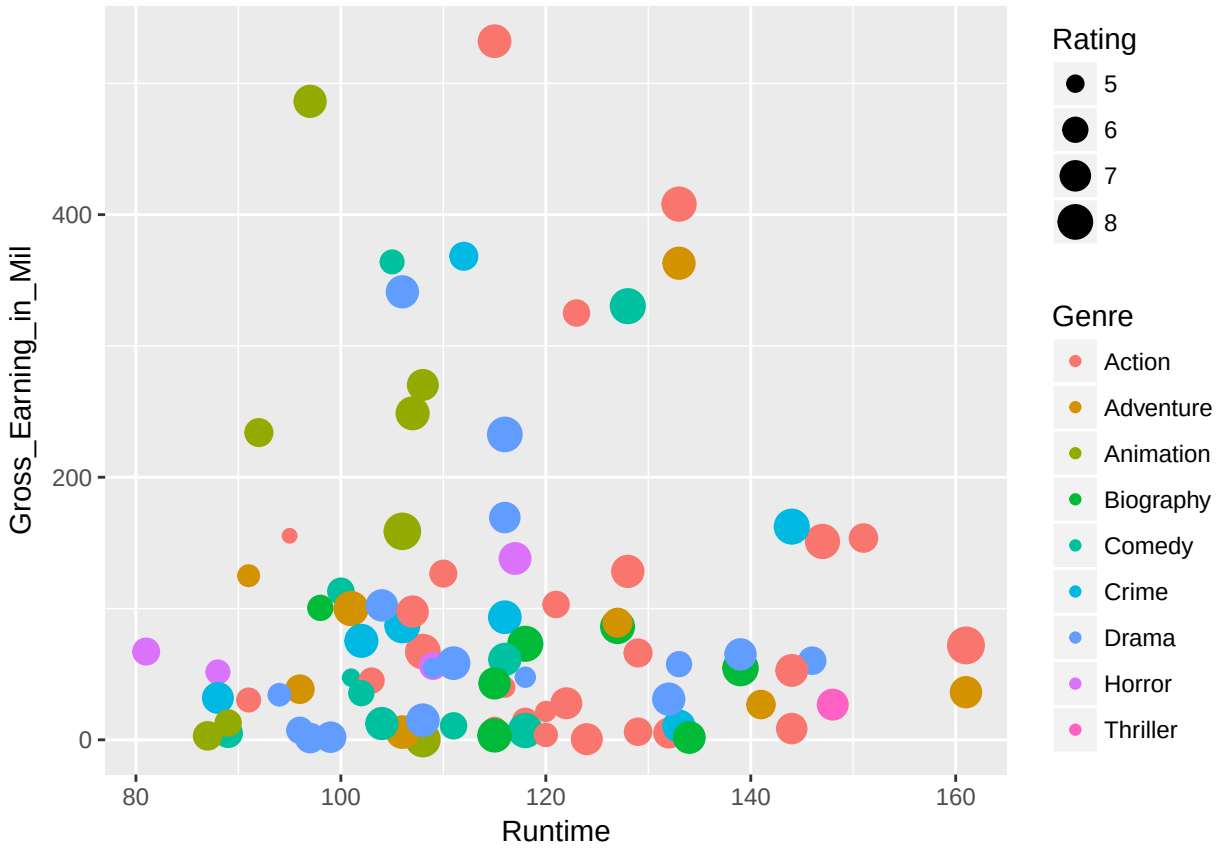
```
ggplot(movies_df,aes(x=Runtime,y=Rating))+
geom_point(aes(size=Votes,col=Genre))
```



Across all genres which genre has the highest average gross earnings in runtime 100 to 120.

```
ggplot(movies_df,aes(x=Runtime,y=Gross_Earning_in_Mil))+
geom_point(aes(size=Rating,col=Genre))
```

```
## Warning: Removed 13 rows containing missing values (geom_point).
```


Gross_Earning_in_Mil
400
200
0
80
100
120
140
160
Runtime
Rating
5
6
7
8
Genre
Action
Adventure
Animation
Biography
Comedy
Crime
Drama
Horror
Thriller