Applying Descriptive Statistics, Exploratory Data Analysis & Visaulizations¶

Descriptive Statistics¶

> Descriptive Statistics describes the data in a meaningfull ways

Exploratory Data Analysis¶

> location, spread and shape and inter dependencies of the data

In [8]:

```
# Set working directory
```

In [9]:

```
# Load CSV data
movies <- read.csv(
  file = "data/Movies.csv")

genres <- read.csv(
  file = "data/Genres.csv")
```

In [10]:

```
# Peek at data
head(movies)

head(genres)
```

Title	Year	Rating	Runtime	Critic.Score	Box.Office
The Whole Nine Yards	2000	R	98	45	57.3
Gladiator	2000	R	155	76	187.3
Cirque du Soleil	2000	G	39	45	13.4
Dinosaur	2000	PG	82	65	135.6
Big Momma's House	2000	PG-13	99	30	0.5
Gone in Sixty Seconds	2000	PG-13	118	24	101.0

Title	Genre	Year	Rating	Runtime	Critic.Score	Box.Office
The Whole Nine Yards	Crime	2000	R	98	45	57.3
The Whole Nine Yards	Comedy	2000	R	98	45	57.3
Cirque du Soleil	Drama	2000	G	39	45	13.4
Cirque du Soleil	Family	2000	G	39	45	13.4
Gladiator	Action	2000	R	155	76	187.3
Gladiator	Drama	2000	R	155	76	187.3

Univariate statistics for qualitative variables¶

In [11]:

```
table(movies$Rating)

table(genres$Genre)
```

```
    G    PG PG-13     R 
   93   497  1225  1423 
```

```
     Action   Adventure   Animation   Biography      Comedy       Crime 
        612         496         168         193        1281         478 
Documentary       Drama      Family     Fantasy     History      Horror 
        243        1570         230         215          86         269 
      Music     Musical     Mystery      Sci-Fi       Sport    Thriller 
        176          37         244         198         121         493 
        War     Western 
         51          20 
```

Univariate statistics for quantitative variables¶

In [12]:

```
# Analyze the location of a quantitative variable
mean(movies$Runtime)

median(movies$Runtime)

which.max(table(movies$Runtime))
```

```
104.405188387894
101
90: 27
```

Analyze the spread of a quantitative variable¶

In [13]:

```
min(movies$Runtime)

max(movies$Runtime)

range(movies$Runtime)

diff(range(movies$Runtime))

quantile(movies$Runtime)

quantile(movies$Runtime, 0.25)

quantile(movies$Runtime, 0.90)

IQR(movies$Runtime)

var(movies$Runtime)

sd(movies$Runtime)
```

```
38
219

   1. 38
   2. 219

181

0%
      38
25%
      93
50%
      101
75%
      113
100%
      219

25%: 93
90%: 126
20
284.448684842472
16.8656065660999
```

Analyze the shape of a quantitative variable¶

In [14]:

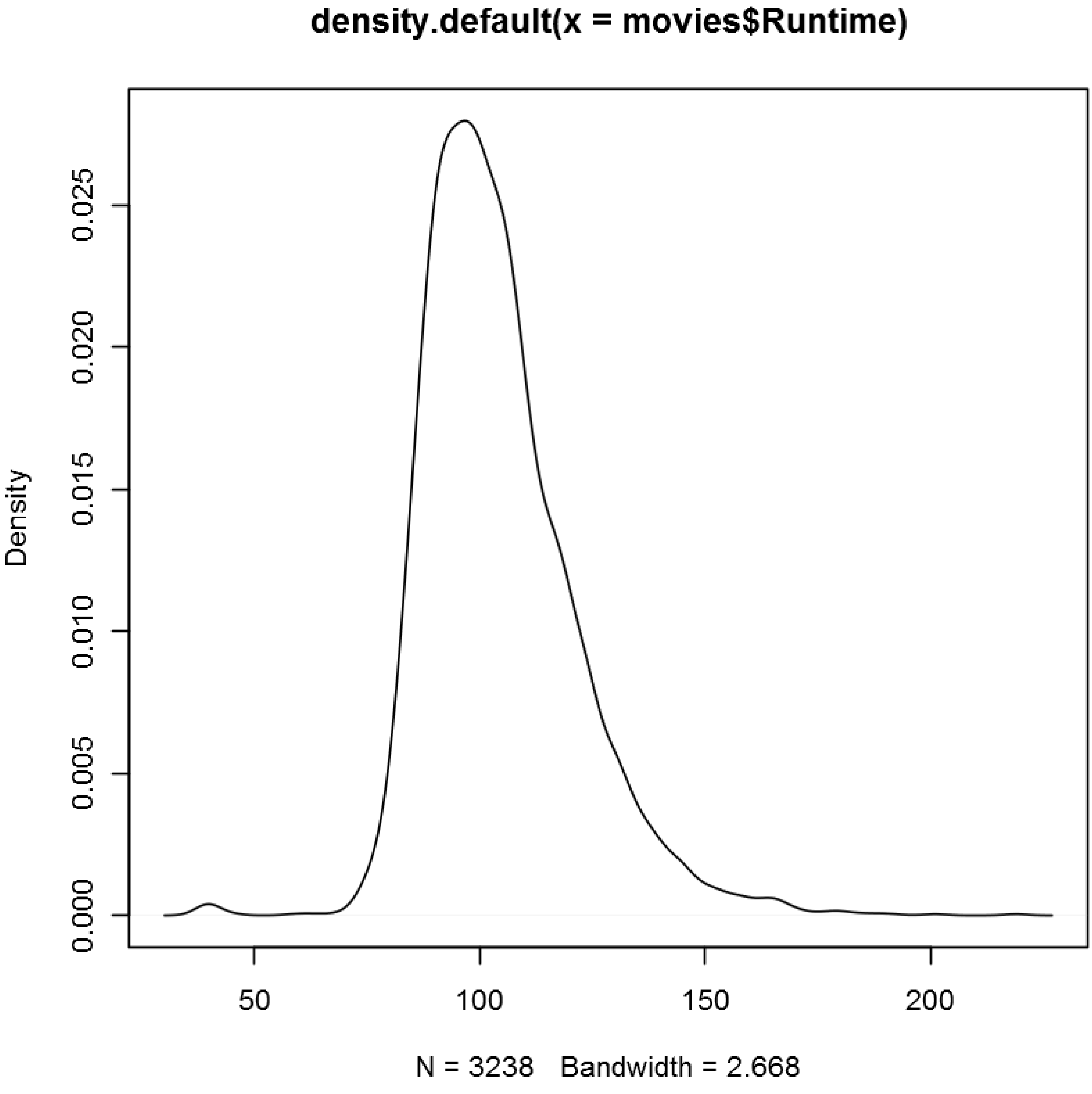
```
# install.packages("moments")
```

In [15]:

```
library(moments)
```

```
skewness(movies$Runtime)

kurtosis(movies$Runtime)

plot(density(movies$Runtime))
```

1.00778834530783
5.95635535550189



In [16]:

```
# Summarize a quantitative variable
summary(movies$Runtime)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   38.0    93.0   101.0   104.4   113.0   219.0 
```

In [17]:

```
# Bivariate statistics for two qualitative variables
table(genres$Genre, genres$Rating)

# Covarience
cov(movies$Runtime, movies$Box.Office)

cov(movies$Critic.Score, movies$Box.Office)
```

```
               G  PG PG-13   R
  Action        2  70   311 229
  Adventure    44 179   209  64
  Animation    43 111     8   6
  Biography     0  27    73  93
  Comedy       45 258   472 506
  Crime         0   9   141 328
  Documentary  27  73    78  65
  Drama        12 136   586 836
  Family       38 181    10   1
  Fantasy       6  51   115  43
  History       3  12    36  35
  Horror        0   3    71 195
  Music         5  31    81  59
  Musical       0  11    20   6
  Mystery       0   6   102 136
  Sci-Fi        0   7   119  72
  Sport         4  36    62  19
  Thriller      0   2   167 324
  War           1   0    19  31
  Western       0   4     6  10
```

381.624015269827
289.633547836202
In [18]:

```
# Analyze the location of a quantitative variable
mean(movies$Runtime)

median(movies$Runtime)

which.max(table(movies$Runtime))
```

104.405188387894
101
90: 27
In [19]:

```
# Covarience
cov(movies$Runtime, movies$Box.Office)

cov(movies$Critic.Score, movies$Box.Office)
```

381.624015269827
289.633547836202

Bivariate statistics for two quantitative variables¶

In [20]:

```
# Correlation coefficients

cor(movies$Runtime, movies$Box.Office)

cor(movies$Critic.Score, movies$Box.Office)
```

0.347747954137135
0.160832402381056

Bivariate statistics for both a qualitative and quantitative variable¶

In [21]:

```
# Bivariate statistics for both a qualitative and quantitative variable
tapply(movies$Box.Office, movies$Rating, mean)

tapply(genres$Box.Office, genres$Genre, mean)
```

```
ERROR while rich displaying an object: Error in dn[[2L]]: subscript out of bounds

Traceback:
1. FUN(X[[i]], ...)
2. tryCatch(withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr))
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler), error = outer_handler)
3. tryCatchList(expr, classes, parentenv, handlers)
4. tryCatchOne(expr, names, parentenv, handlers[[1L]])
5. doTryCatch(return(expr), name, parentenv, handler)
6. withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler)
7. mime2repr[[mime]](obj)
8. repr_markdown.numeric(obj)
9. repr_vector_generic(html_escape_names(obj), "%s. %s\n", "%s\n:   %s", 
 .     "**%s:** %s", "%s\n\n", item_uses_numbers = TRUE, escape_fun = html_escape)
10. html_escape_names(obj)
11. .escape_names(obj, "html")
12. colnames(obj)
ERROR while rich displaying an object: Error in dn[[2L]]: subscript out of bounds

Traceback:
1. FUN(X[[i]], ...)
2. tryCatch(withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler), error = outer_handler)
3. tryCatchList(expr, classes, parentenv, handlers)
4. tryCatchOne(expr, names, parentenv, handlers[[1L]])
5. doTryCatch(return(expr), name, parentenv, handler)
6. withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler)
7. mime2repr[[mime]](obj)
8. repr_latex.numeric(obj)
9. repr_vector_generic(latex_escape_names(obj), "\\item %s\n", "\\item[%s] %s\n", 
 .     "\\textbf{%s:} %s", enum_wrap = "\\begin{enumerate*}\n%s\\end{enumerate*}\n", 
 .     named_wrap = "\\begin{description*}\n%s\\end{description*}\n", 
 .     only_named_item = "\\textbf{%s:} %s", escape_fun = latex_escape)
10. latex_escape_names(obj)
11. .escape_names(obj, "latex")
12. colnames(obj)
```

G
    55.4756087311828
PG
    56.4043937625755
PG-13
    54.5613408163265
R
    22.2611764919185

```
ERROR while rich displaying an object: Error in dn[[2L]]: subscript out of bounds

Traceback:
1. FUN(X[[i]], ...)
2. tryCatch(withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler), error = outer_handler)
3. tryCatchList(expr, classes, parentenv, handlers)
4. tryCatchOne(expr, names, parentenv, handlers[[1L]])
5. doTryCatch(return(expr), name, parentenv, handler)
6. withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler)
7. mime2repr[[mime]](obj)
8. repr_markdown.numeric(obj)
9. repr_vector_generic(html_escape_names(obj), "%s. %s\n", "%s\n:   %s", 
 .     "**%s:** %s", "%s\n\n", item_uses_numbers = TRUE, escape_fun = html_escape)
10. html_escape_names(obj)
11. .escape_names(obj, "html")
```

```
12. colnames(obj)
ERROR while rich displaying an object: Error in dn[[2L]]: subscript out of bounds

Traceback:
1. FUN(X[[i]], ...)
2. tryCatch(withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler), error = outer_handler)
3. tryCatchList(expr, classes, parentenv, handlers)
4. tryCatchOne(expr, names, parentenv, handlers[[1L]])
5. doTryCatch(return(expr), name, parentenv, handler)
6. withCallingHandlers({
 .     rpr <- mime2repr[[mime]](obj)
 .     if (is.null(rpr)) 
 .         return(NULL)
 .     prepare_content(is.raw(rpr), rpr)
 . }, error = error_handler)
7. mime2repr[[mime]](obj)
8. repr_latex.numeric(obj)
9. repr_vector_generic(latex_escape_names(obj), "\\item %s\n", "\\item[%s] %s\n", 
 .     "\\textbf{%s:} %s", enum_wrap = "\\begin{enumerate*}\n%s\\end{enumerate*}\n", 
 .     named_wrap = "\\begin{description*}\n%s\\end{description*}\n", 
 .     only_named_item = "\\textbf{%s:} %s", escape_fun = latex_escape)
10. latex_escape_names(obj)
11. .escape_names(obj, "latex")
12. colnames(obj)
```

```
Action
    76.5308055555556
Adventure
    101.745110282258
Animation
    96.6033107142857
Biography
    26.5003077720207
Comedy
    40.860972863388
Crime
    34.3201418870293
Documentary
    6.26857494650206
Drama
    24.7402955573248
Family
    68.3392
Fantasy
    93.2512111627907
History
    24.1815825581395
Horror
    27.9328945762082
Music
    21.9789181818182
Musical
    37.1727756756757
Mystery
    40.3286605778689
Sci-Fi
    86.8747626262626
Sport
    27.7392404958678
Thriller
    38.5233641805274
War
    26.4742980392157
Western
    36.146105
```

In [22]:

```
# Summarize entire table
summary(movies)
```

```
                  Title           Year        Rating        Runtime
 Camp                :   2   Min.   :2000   G    :  93   Min.   : 38.0
 Frozen              :   2   1st Qu.:2004   PG   : 497   1st Qu.: 93.0
 The Other Woman     :   2   Median :2008   PG-13:1225   Median :101.0
 (500) Days of Summer:   1   Mean   :2008   R    :1423   Mean   :104.4
 (Untitled)          :   1   3rd Qu.:2011                3rd Qu.:113.0
 10 Items or Less    :   1   Max.   :2015                Max.   :219.0
 (Other)             :3229
  Critic.Score      Box.Office
 Min.   :  0.00   Min.   :  0.0002
 1st Qu.: 26.00   1st Qu.:  1.0000
 Median : 49.00   Median : 16.1000
 Mean   : 49.68   Mean   : 40.6756
 3rd Qu.: 74.00   3rd Qu.: 51.4750
 Max.   :100.00   Max.   :760.5000
```

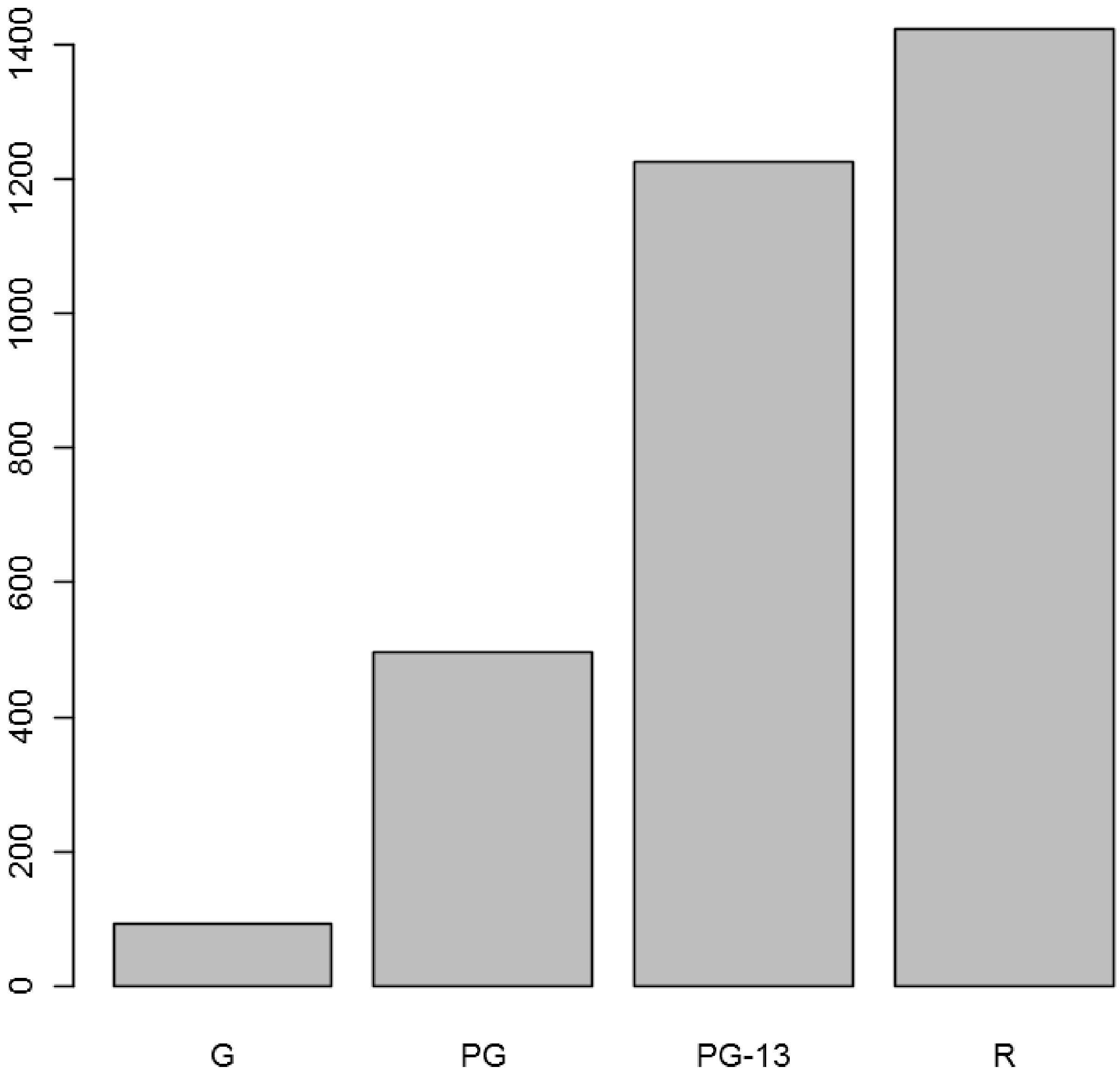
Visualizations¶

> Representing Characterstics of the data in visual ways

Univariate visualizations for a qualitiative variable¶

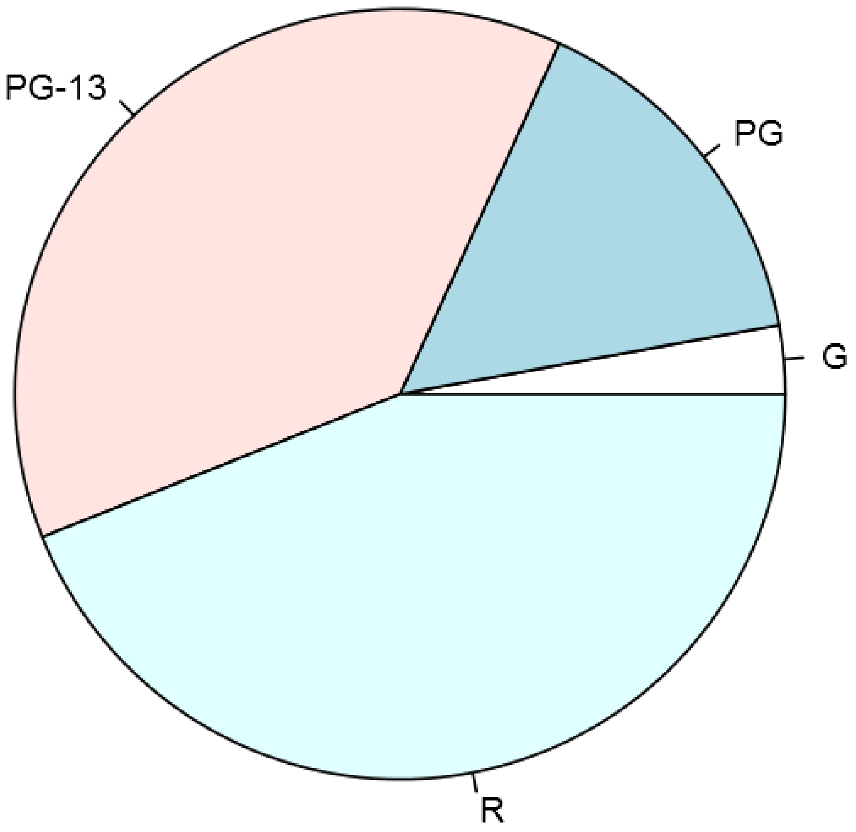
In [23]:

```
# Create a bar graph of rating observations
plot(movies$Rating)
```



In [24]:

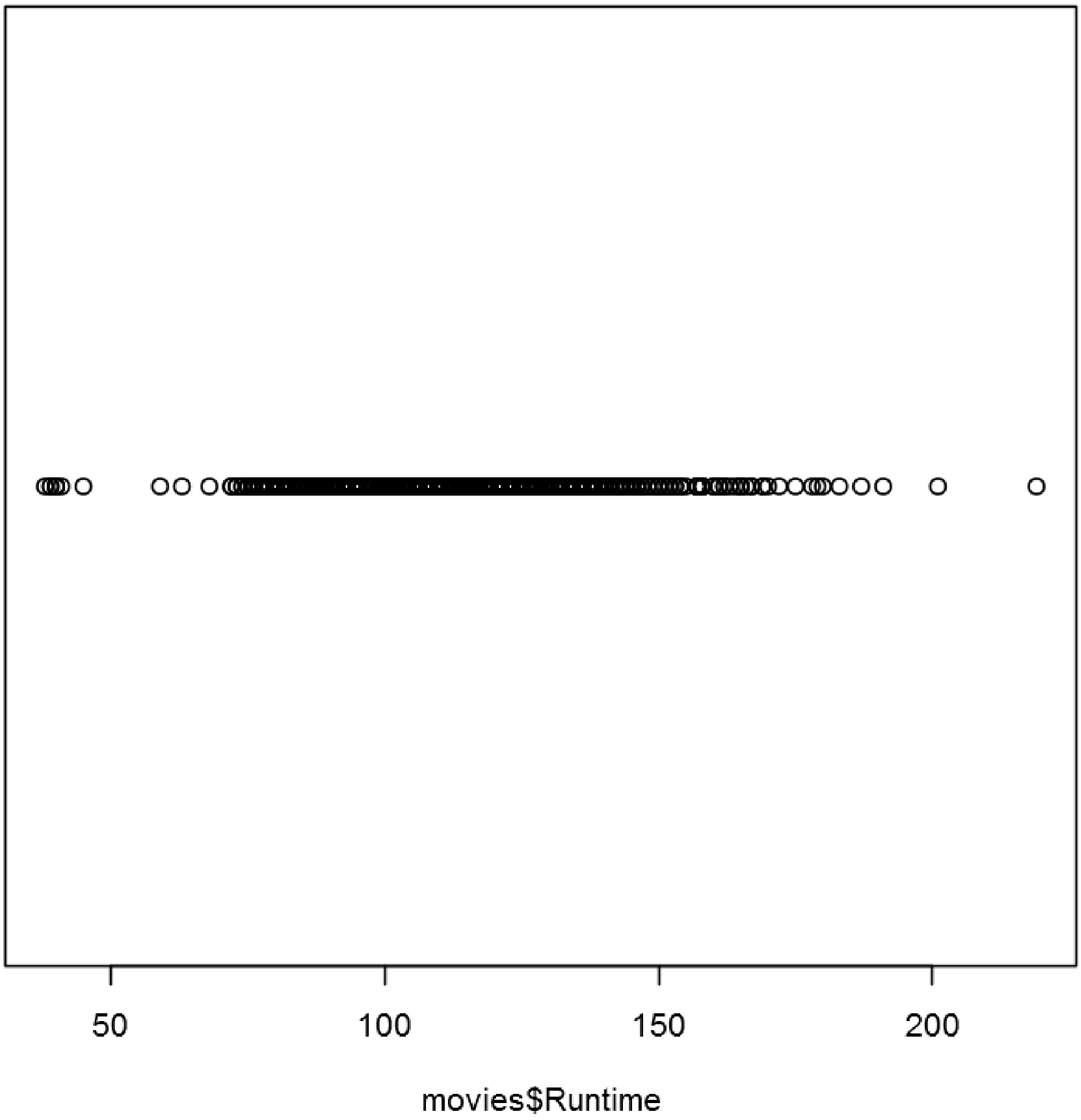
```
# Create a pie chart of rating observations
pie(table(movies$Rating))
```



Univariate visualizations of a quantitiative variable¶

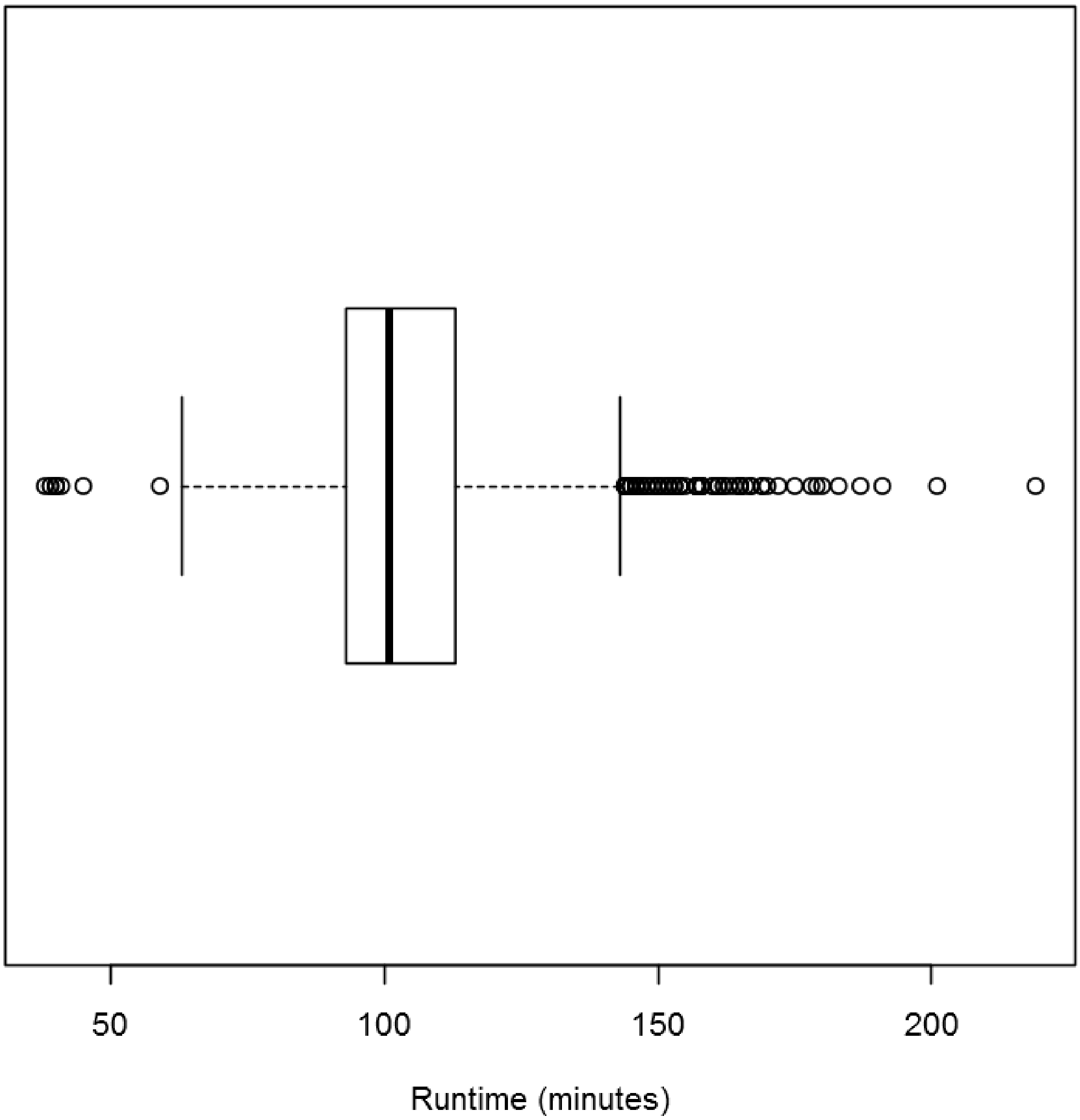
In [25]:

```
# Create a dot plot of runtime
plot(
  x = movies$Runtime,
  y = rep(0, nrow(movies)),
  ylab = "",
  yaxt = "n")
```



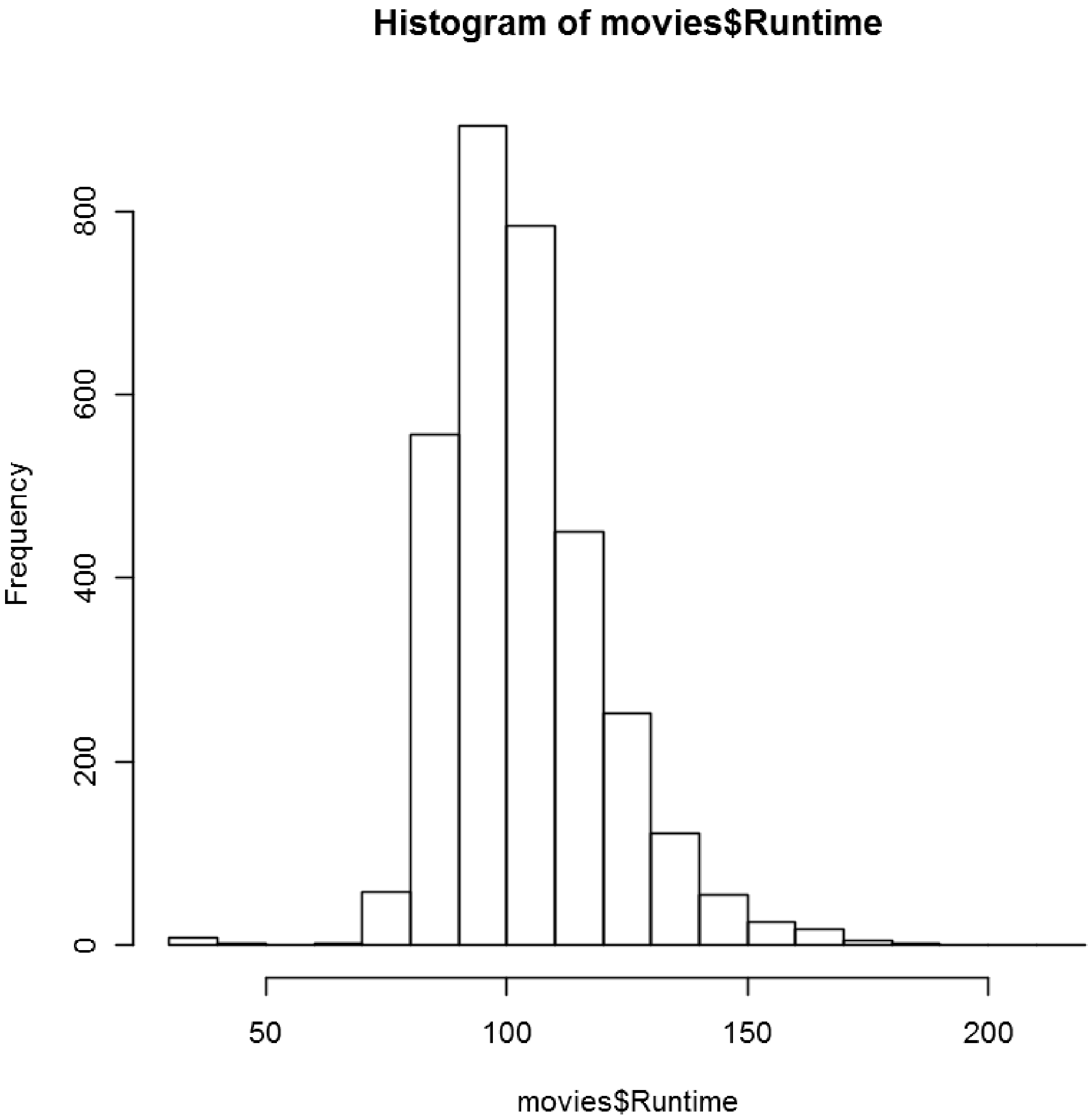
In [26]:

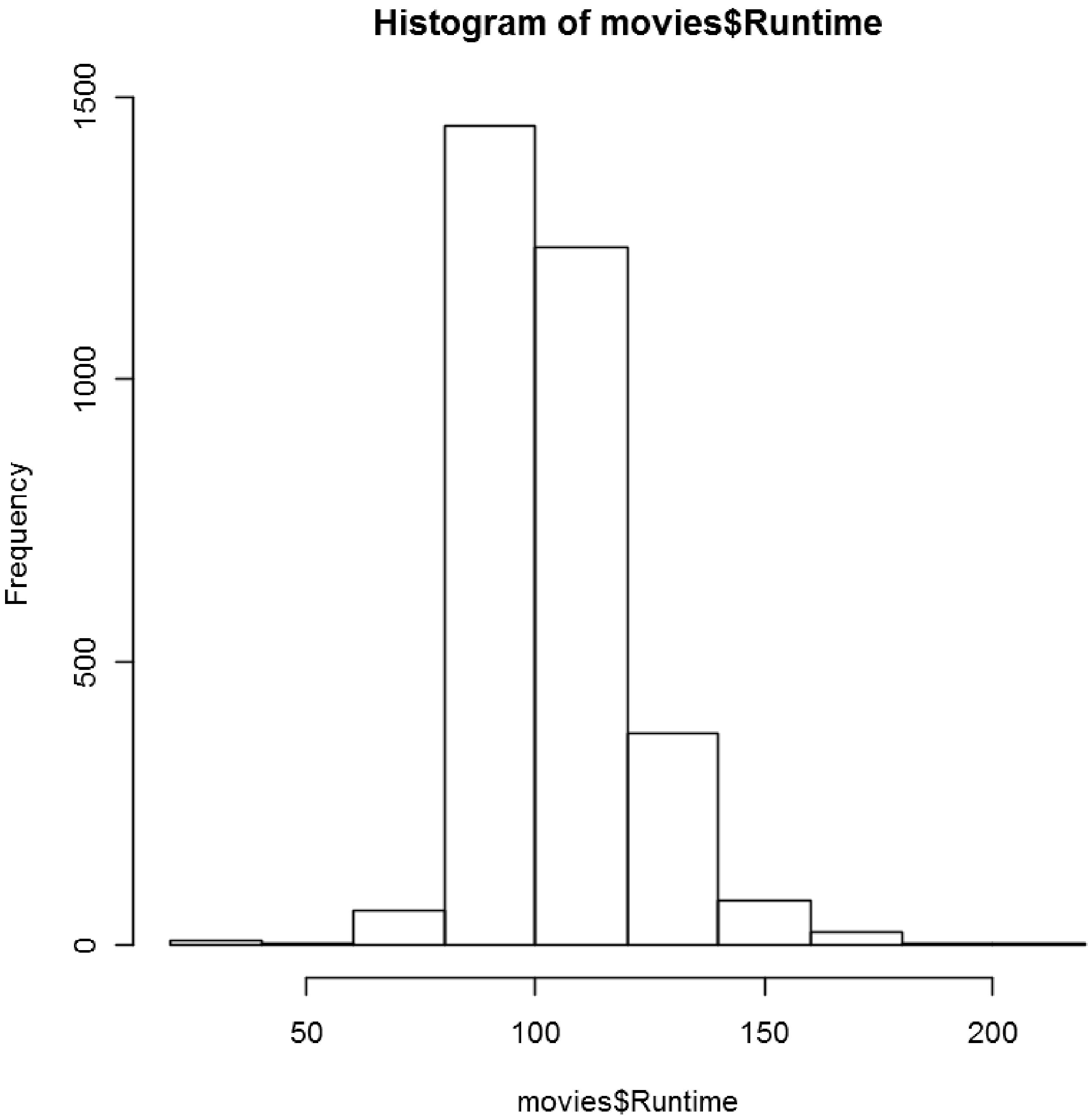
```
# Create a boxplot of runtime
boxplot(
  x = movies$Runtime,
  xlab = "Runtime (minutes)",
  horizontal = TRUE)
```

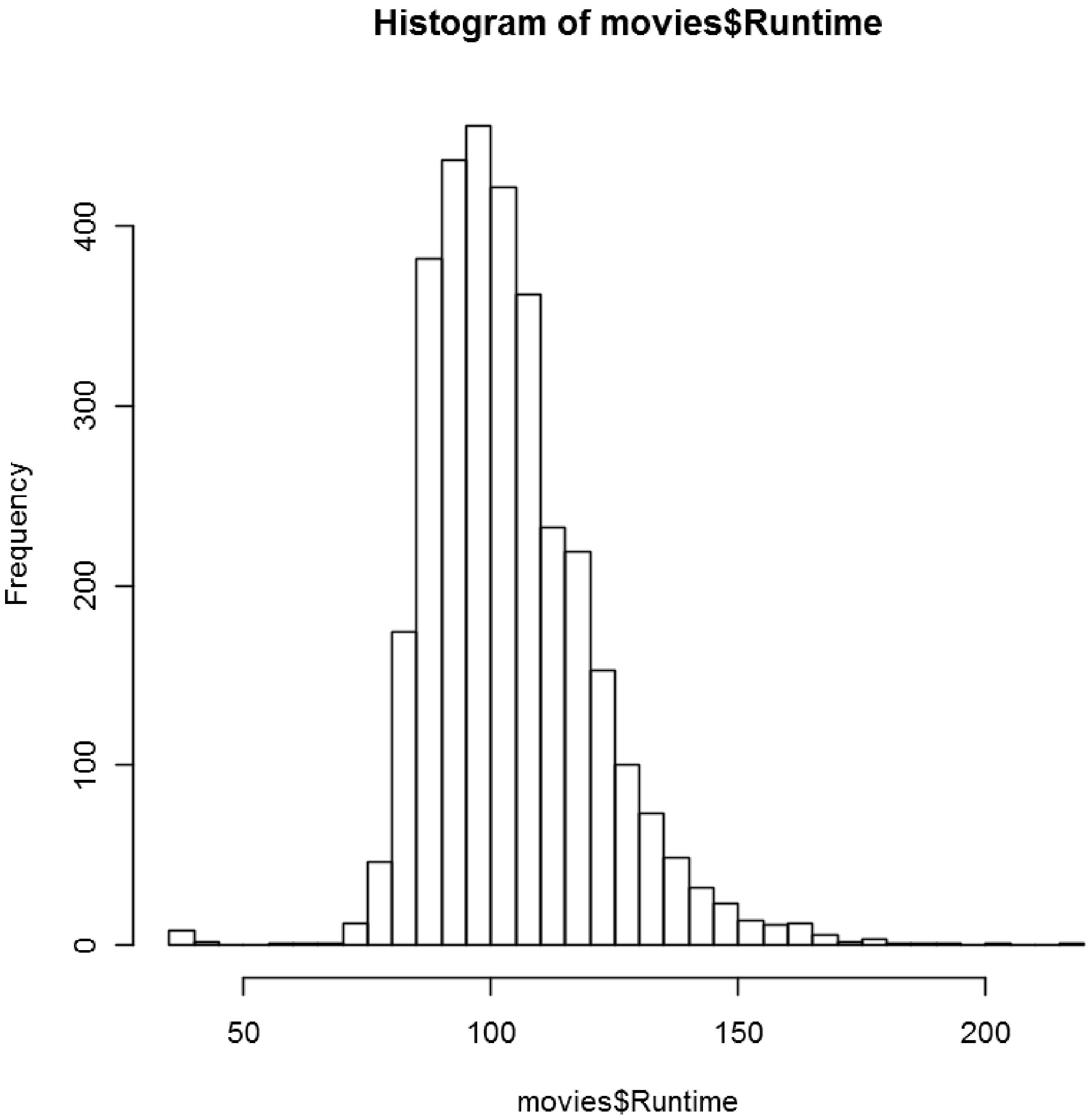


In [27]:

```
# Create a histogram of runtime
hist(movies$Runtime)

# Create a more course-grain histogram
hist(
  x = movies$Runtime,
  breaks = 10)

# Create a more fine-grain histogram
hist(
  x = movies$Runtime,
  breaks = 30)
```

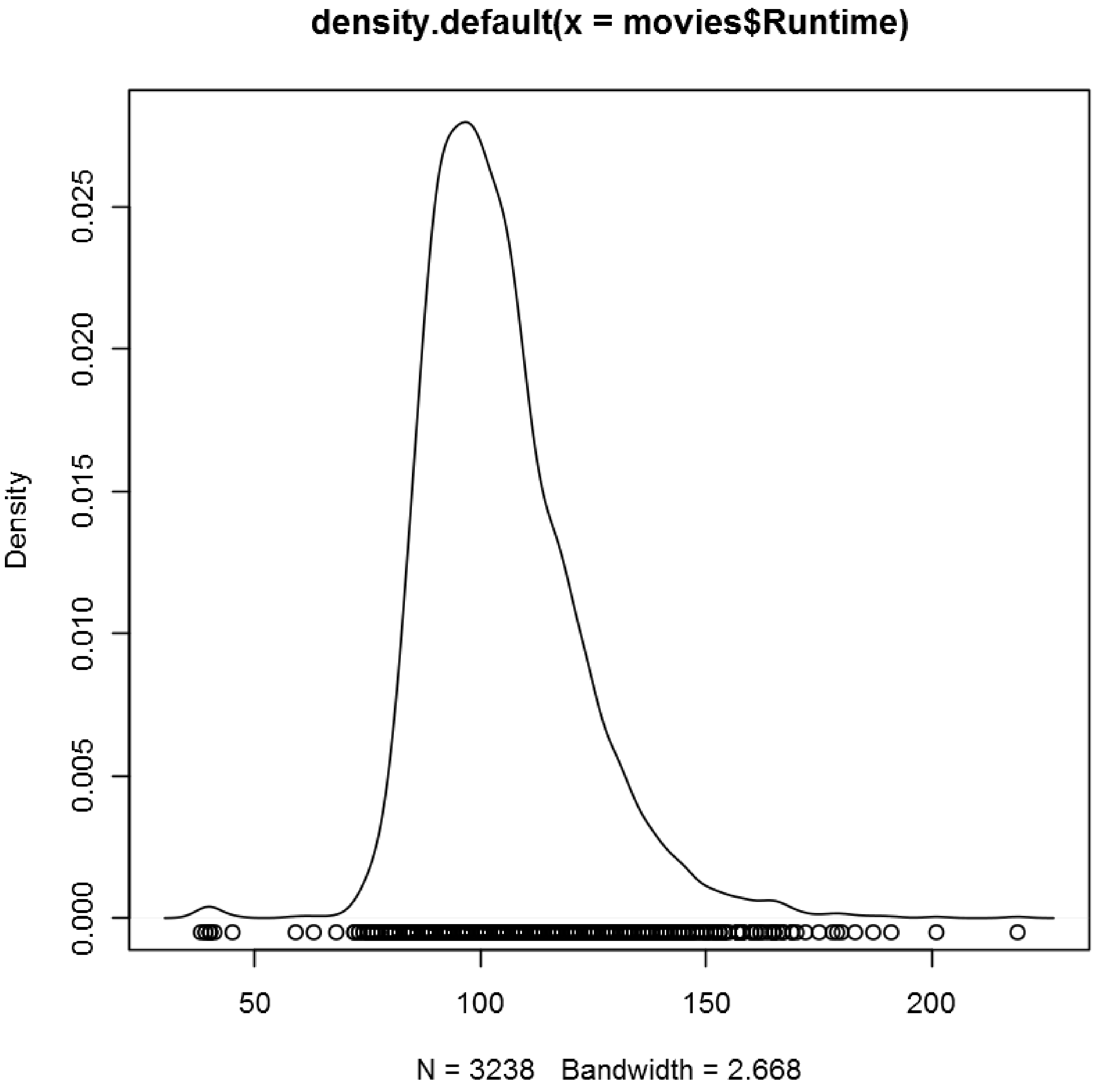

Histogram of movies$Runtime
Frequency
0
200
400
600
800
50
100
150
200
movies$Runtime


Histogram of movies$Runtime
Frequency
0
500
1000
1500
50
100
150
200
movies$Runtime



In [28]:

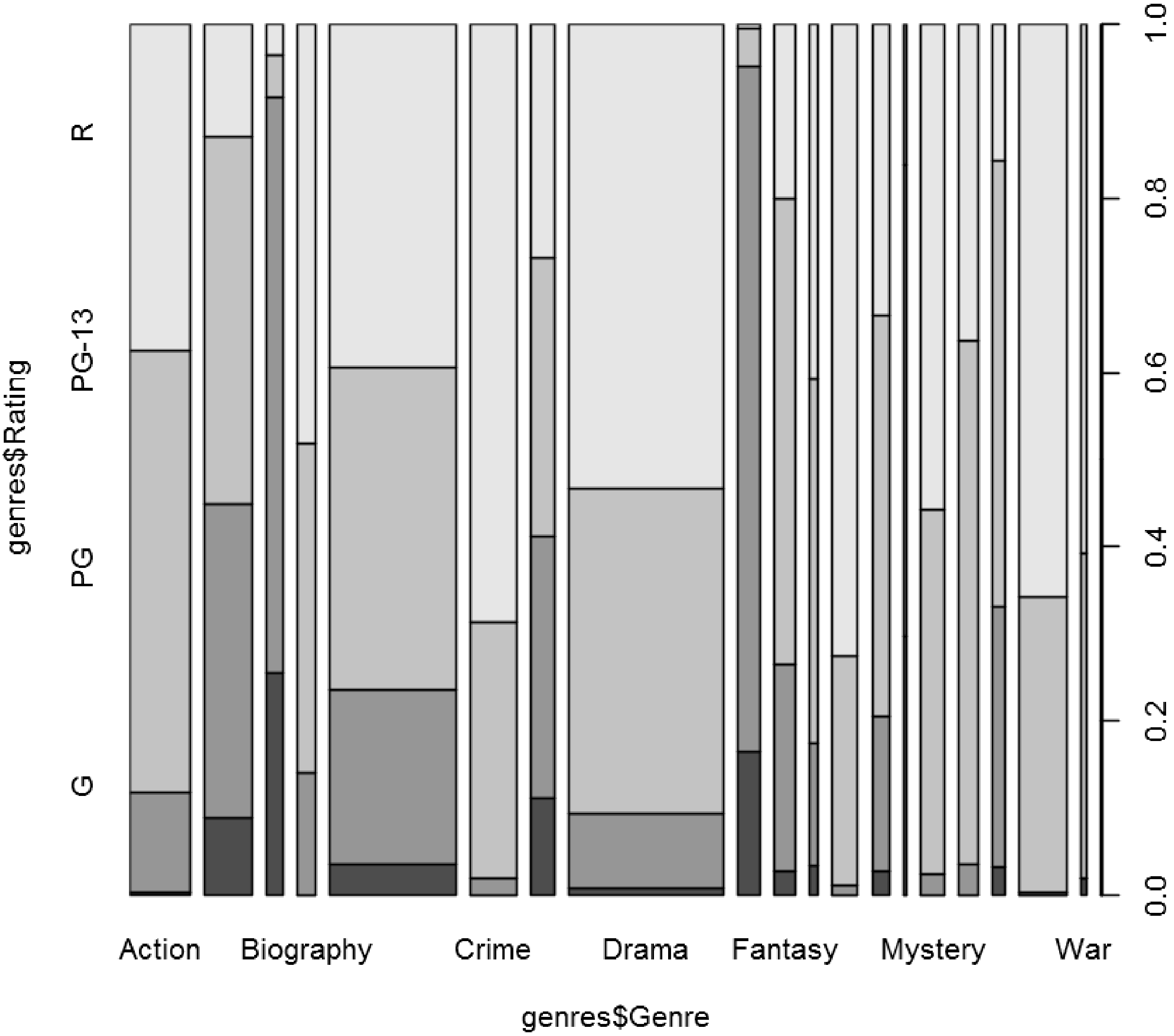
```
# Create a density plot of runtime
plot(density(movies$Runtime))

# Add dot plot to base of density plot
points(
  x = movies$Runtime,
  y = rep(-0.0005, nrow(movies)))
```

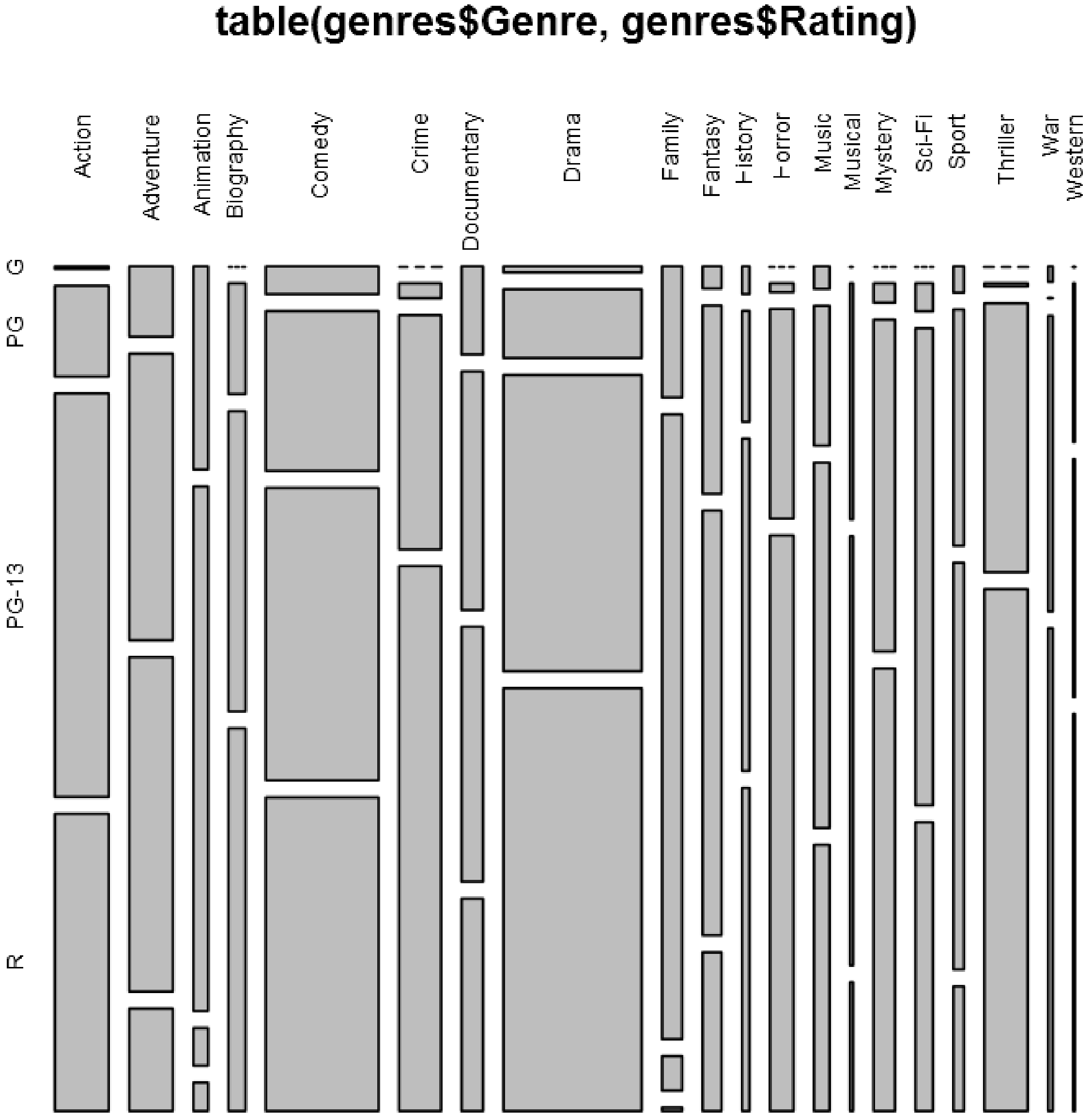


Bivariate visualizations for two qualitiative variables¶

In [29]:

```
# Create a spineplot of genre and rating
spineplot(
  x = genres$Genre,
  y = genres$Rating)

# Create a mosaic plot of genre and rating
mosaicplot(
  x = table(
    genres$Genre,
    genres$Rating),
  las = 3)
```

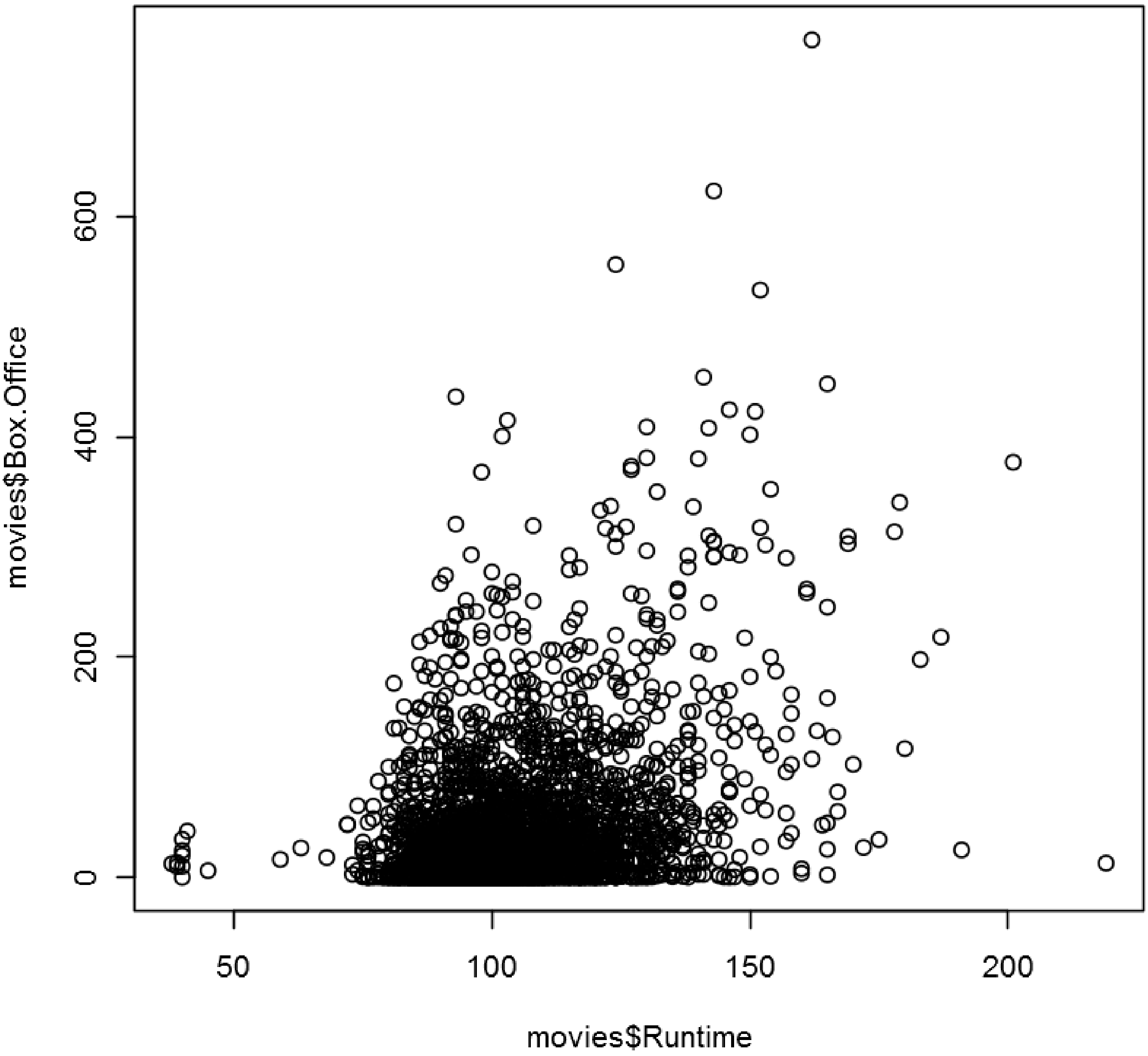

genres$Rating
G
PG
PG-13
R
0.0
0.2
0.4
0.6
0.8
1.0
Action
Biography
Crime
Drama
Fantasy
Mystery
War
genres$Genre

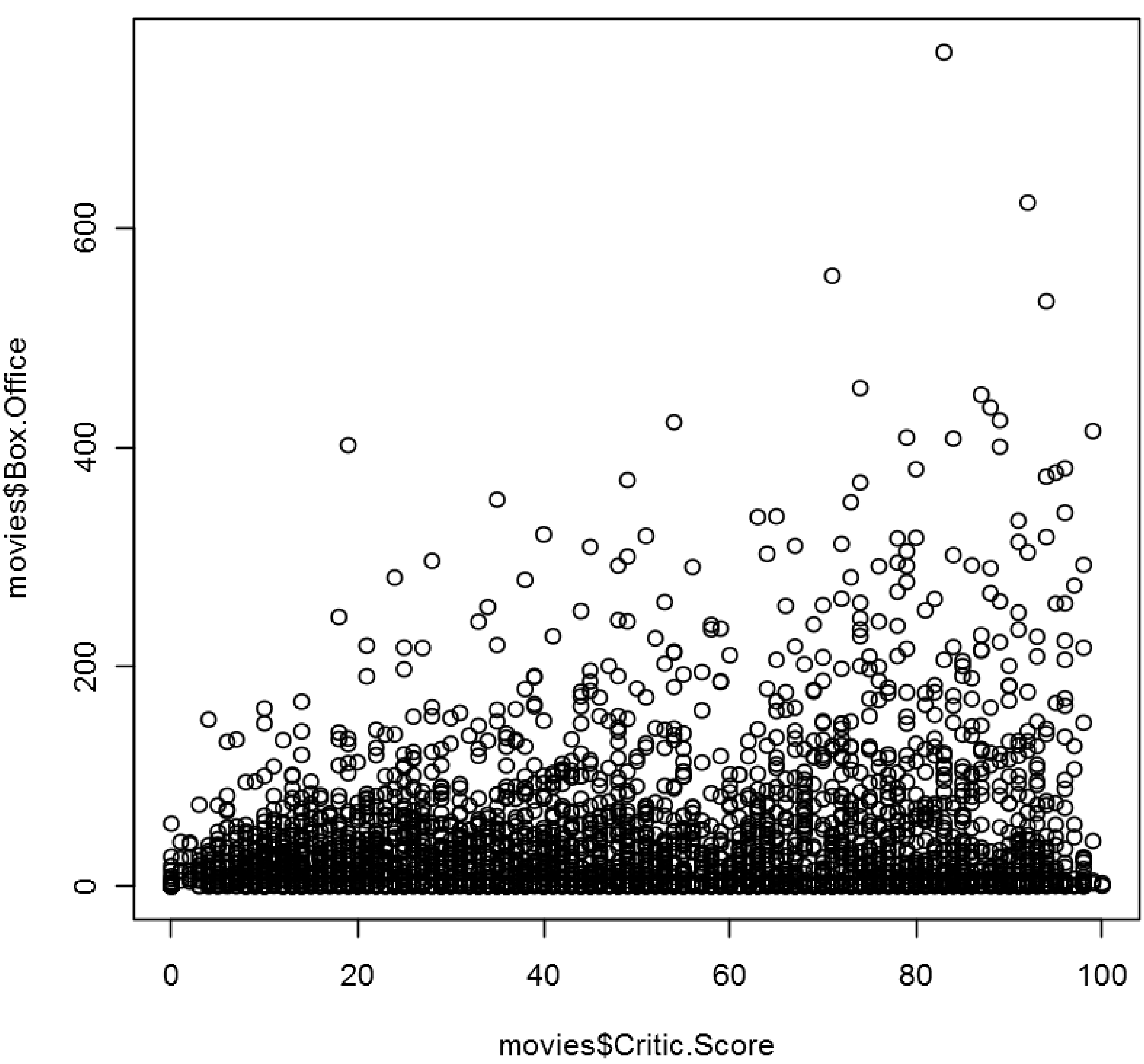


Bivariate visualizations for two quantitiative variables¶

In [30]:

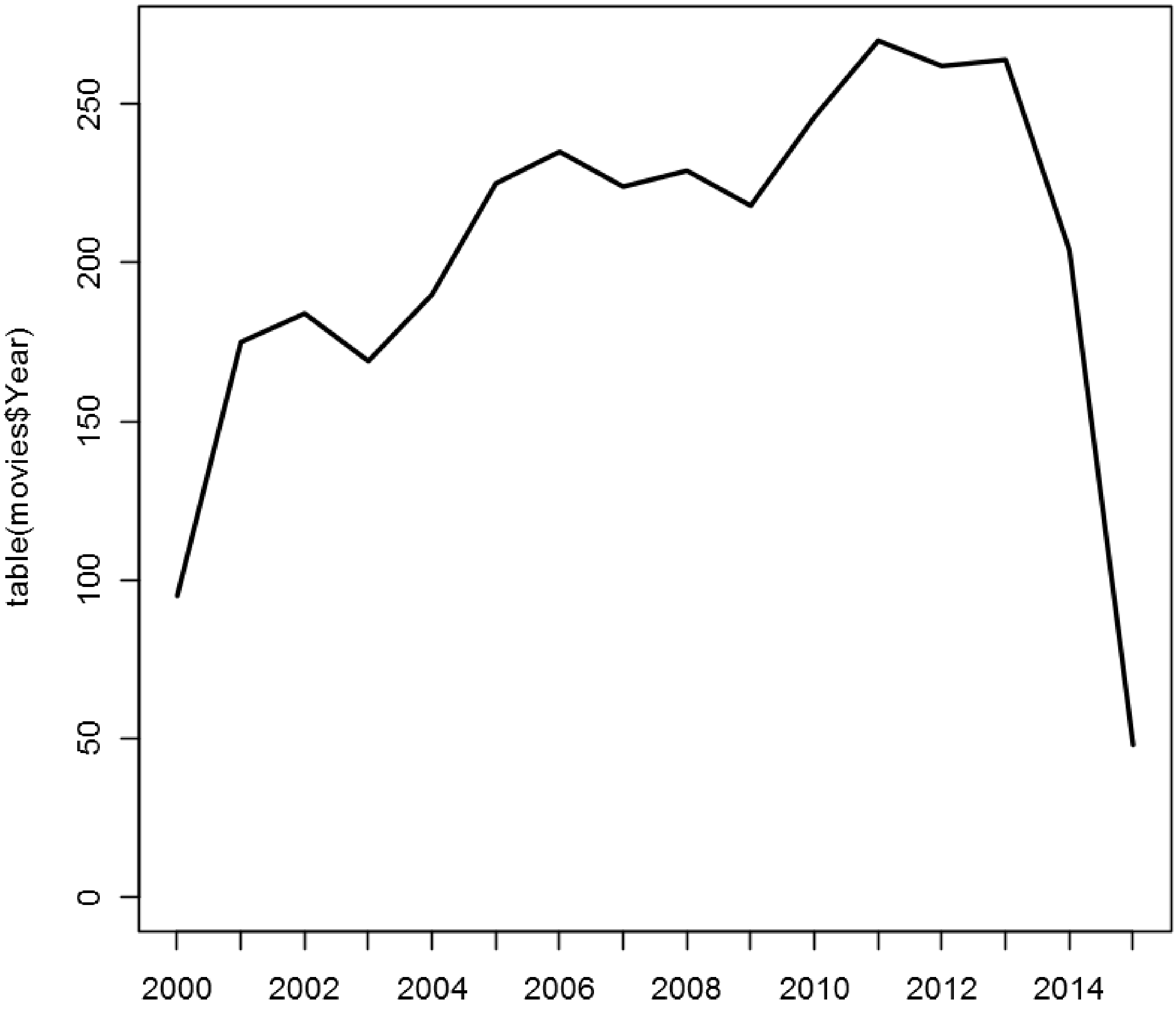
```
# Create a scatterplot of runtime and box office
plot(
  x = movies$Runtime,
  y = movies$Box.Office)

# Create a scatterplot of critic score and box office
plot(
  x = movies$Critic.Score,
  y = movies$Box.Office)
```


movies$Box.Office
0
200
400
600
50
100
150
200
movies$Runtime



In [31]:

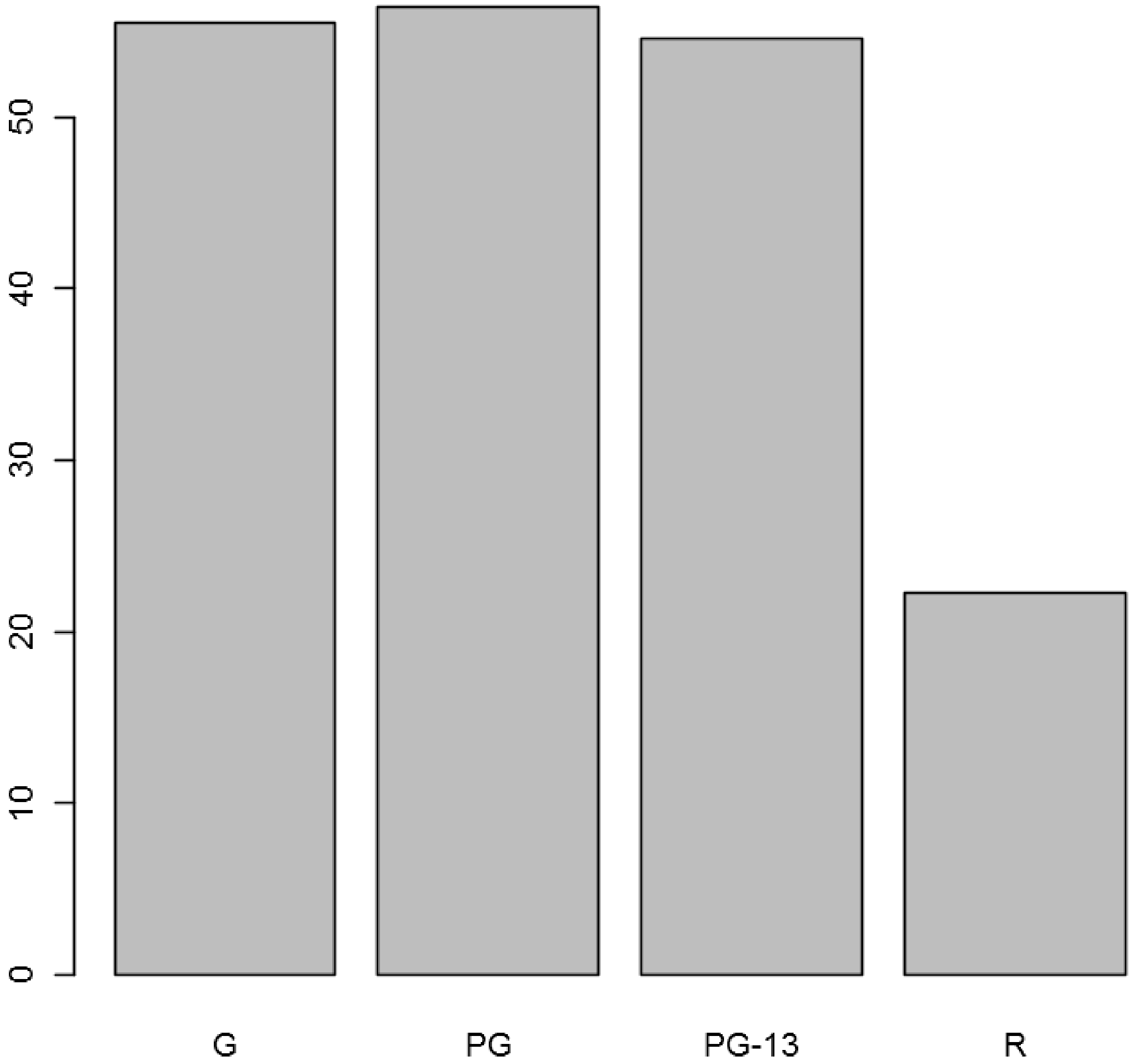
```
# Plot a line graph of count of movies by year
plot(
  x = table(movies$Year),
  type = "l")
```

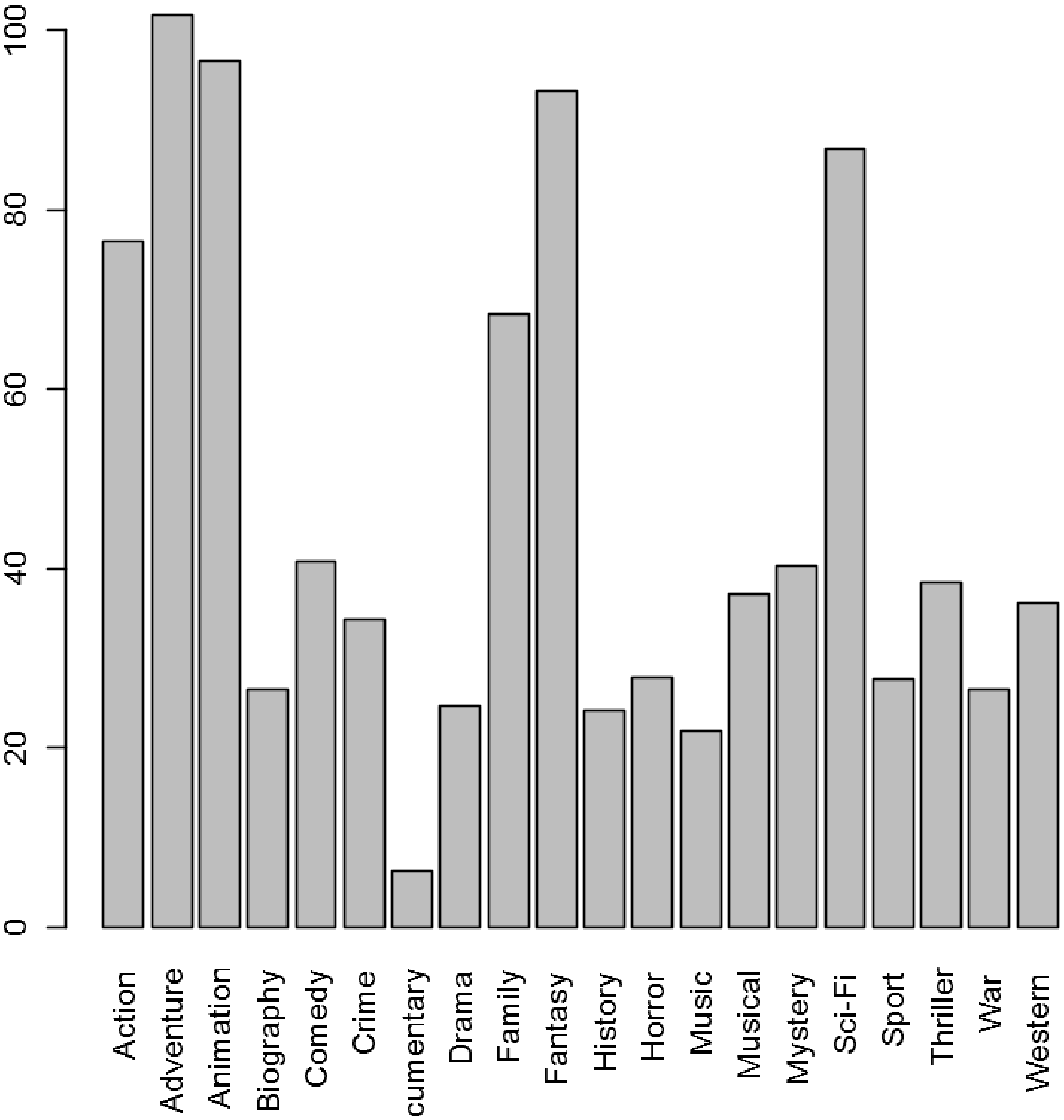


Bivariate visualizations for both a qualitiative and quantitiative variable¶

In [32]:

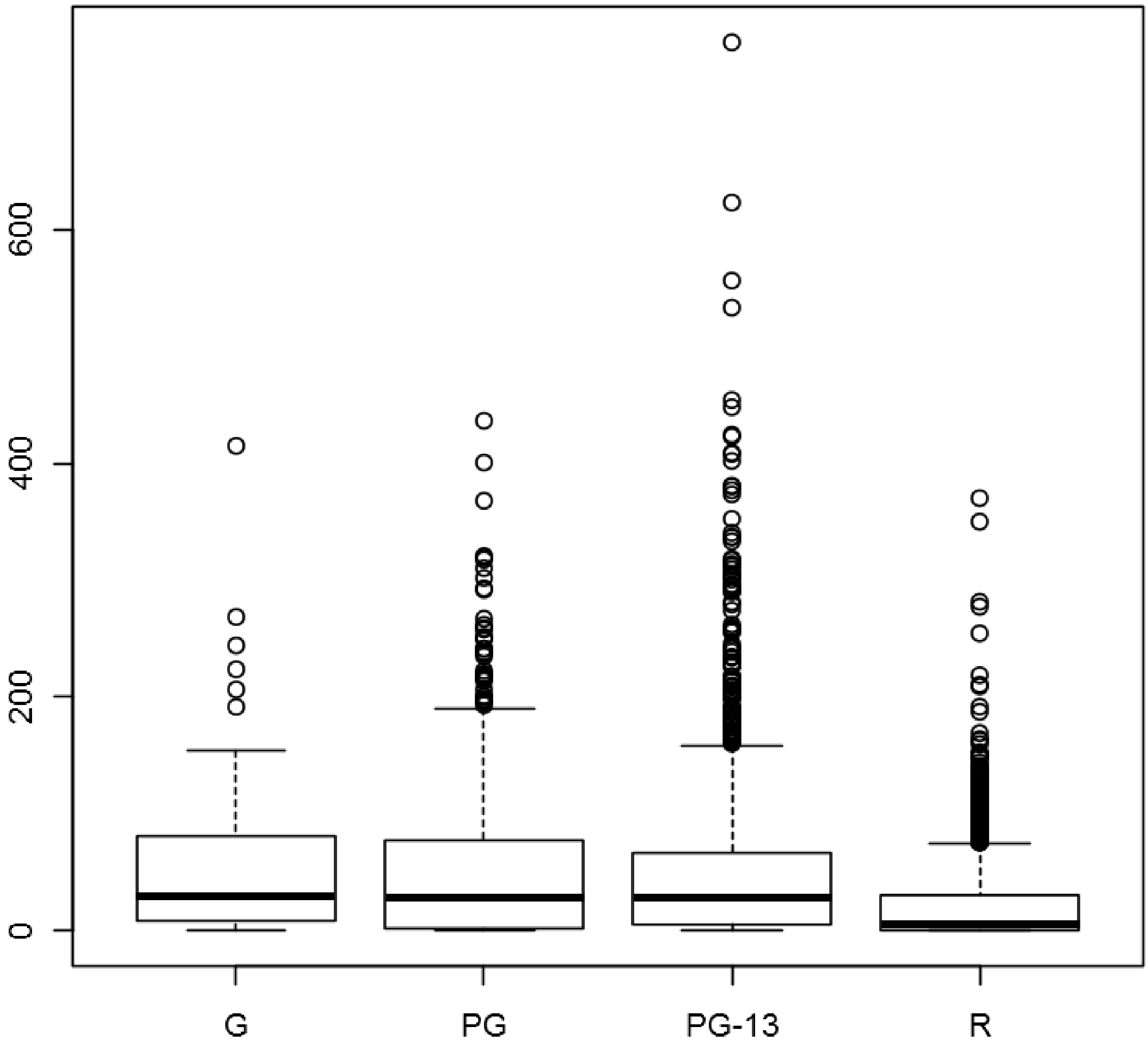
```
# Create a bar graph of average box office by rating
barplot(tapply(movies$Box.Office, movies$Rating, mean))

# Create a bar graph of average box office by genre
barplot(
  height = tapply(genres$Box.Office, genres$Genre, mean),
  las = 3)
```


0
10
20
30
40
50
G
PG
PG-13
R



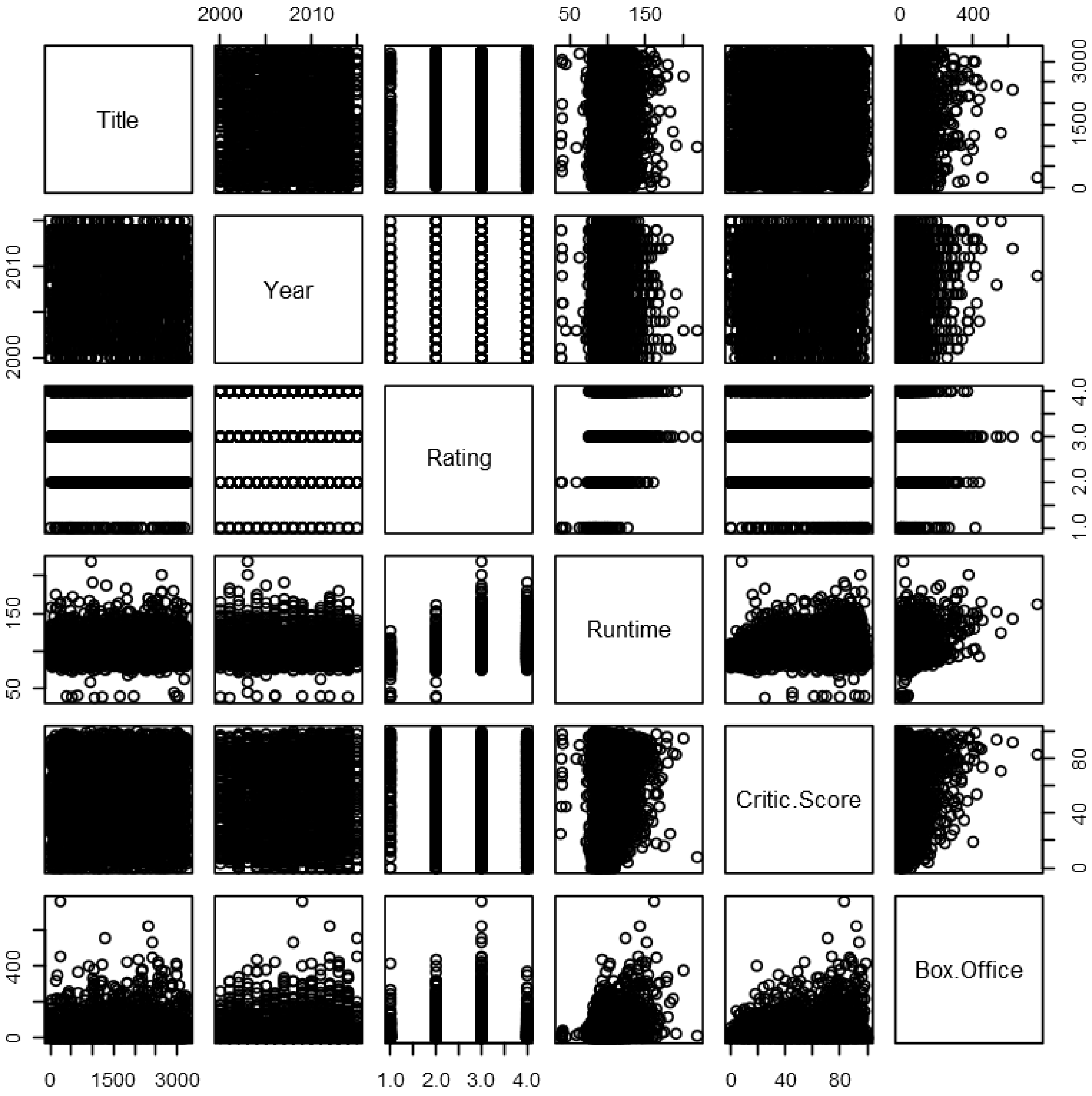
In [33]:

```
# Plot bivariate box plots of box office by rating
plot(
  x = movies$Rating,
  y = movies$Box.Office)
```



In [34]:

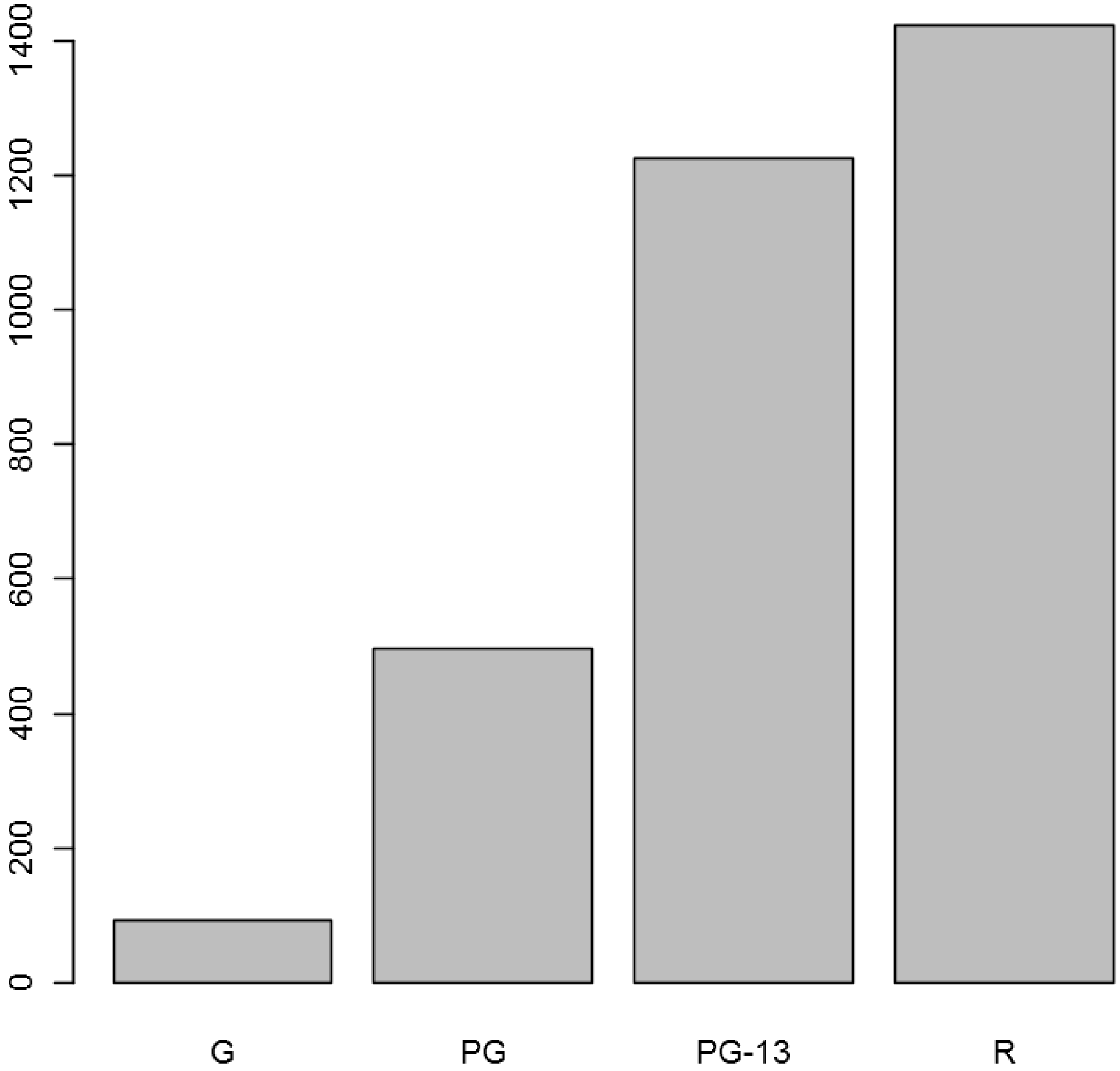
```
# Summarizing an entire table

# Create a scatterplot matrix
plot(movies)
```

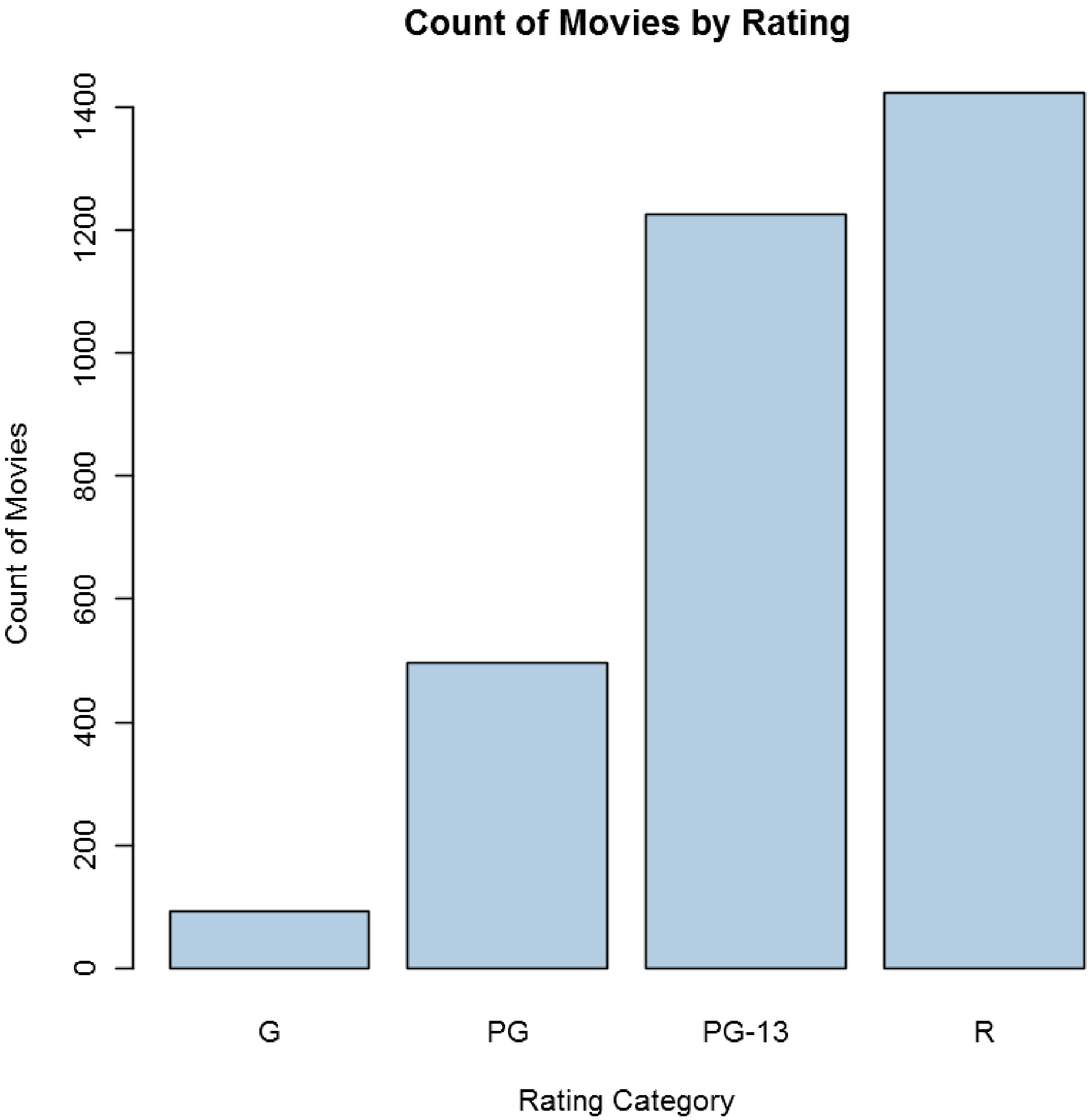


Cleaning up data visualizations¶

In [35]:

```
# Create a bar chart with defaults
plot(movies$Rating)

# Clean up the bar chart
plot(
  x = movies$Rating,
  main = "Count of Movies by Rating",
  xlab = "Rating Category",
  ylab = "Count of Movies",
  col = "#b3cde3")

# View help for plots and parameters
?plot
?par
```


0
200
400
600
800
1000
1200
1400
G
PG
PG-13
R


Count of Movies by Rating
Count of Movies
0
200
400
600
800
1000
1200
1400
G
PG
PG-13
R
Rating Category