Problem Set 3

Jadyn Gonzalez

9/20/2019

IMDB's Top 5000 Movies Problem Set

Data Exploration

- a) Lets import our data set

```
setwd("~/Documents/MGSC310")
getwd()
```

```
## [1] "/Users/jgonzalez/Documents/MGSC310"
```

```
movies = read.csv("movie_metadata.csv")
```

```
  and display the dimensions of the data.
```

```
dim(movies)
```

```
## [1] 5043   28
```

- b) Displaying the names of the attributes in the data set.

```
names(movies)
```

```
##  [1] "color"                     "director_name"
##  [3] "num_critic_for_reviews"    "duration"
##  [5] "director_facebook_likes"   "actor_3_facebook_likes"
##  [7] "actor_2_name"              "actor_1_facebook_likes"
##  [9] "gross"                     "genres"
## [11] "actor_1_name"              "movie_title"
## [13] "num_voted_users"           "cast_total_facebook_likes"
## [15] "actor_3_name"              "facenumber_in_poster"
## [17] "plot_keywords"             "movie_imdb_link"
## [19] "num_user_for_reviews"      "language"
## [21] "country"                   "content_rating"
## [23] "budget"                    "title_year"
## [25] "actor_2_facebook_likes"    "imdb_score"
## [27] "aspect_ratio"              "movie_facebook_likes"
```

- c) Let's see how many missing values there are. Then we'll remove them.

```r
sum(is.na(movies$budget))
```

```
## [1] 492
```

```r
movies = movies[!is.na(movies$budget),]
```

```
Now we'll display the new dimensions.
```

```r
dim(movies)
```

```
## [1] 4551   28
```

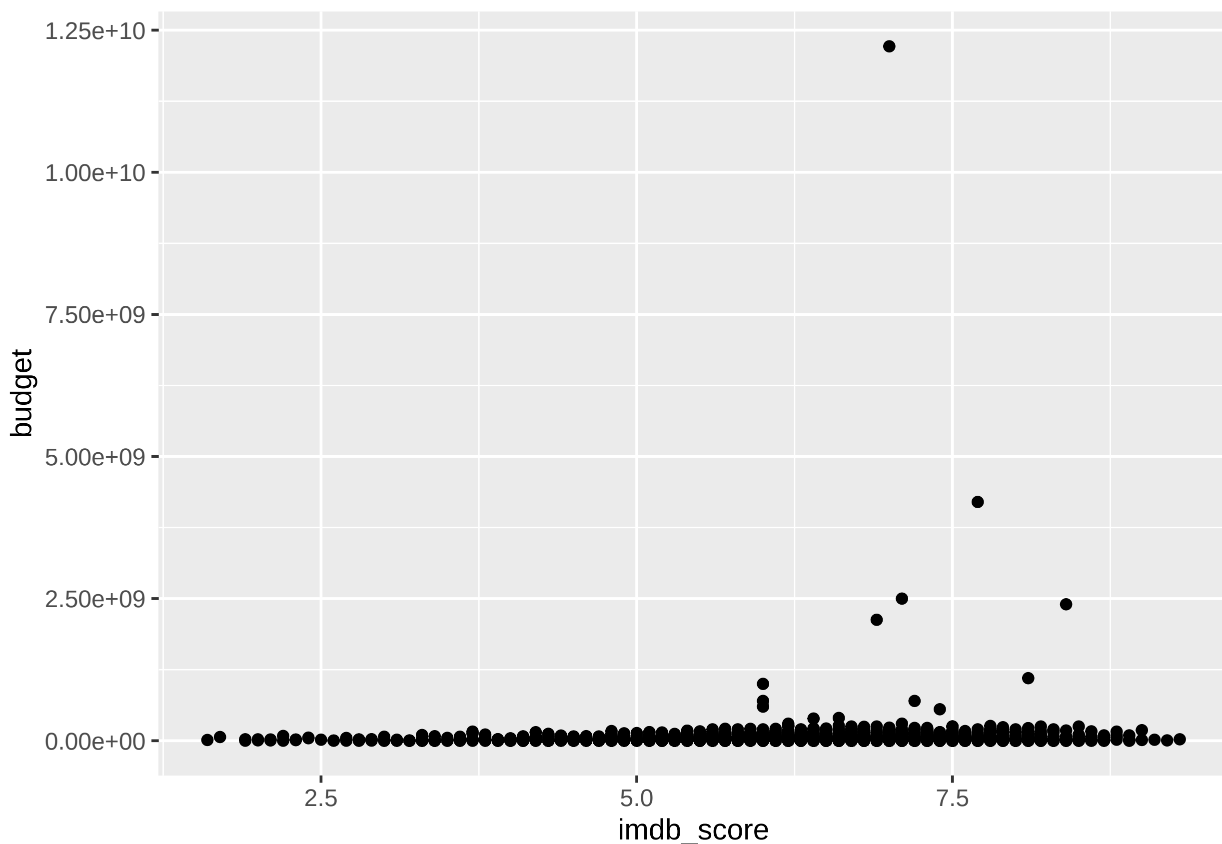
- d) We'll use length() and unique() to count the unique directors in the data.

```r
length(unique(movies$director_name))
```

```
## [1] 2175
```

- e) Plot of imdb_score and budget using ggplot2.

```r
library(ggplot2)
ggplot(movies, aes(x = imdb_score,y = budget)) + geom_point()
```



- f) Looks like we had some outliers. We'll remove any movie with a budget over $400 million.

```
movies = movies[movies$budget<400000000,]
```

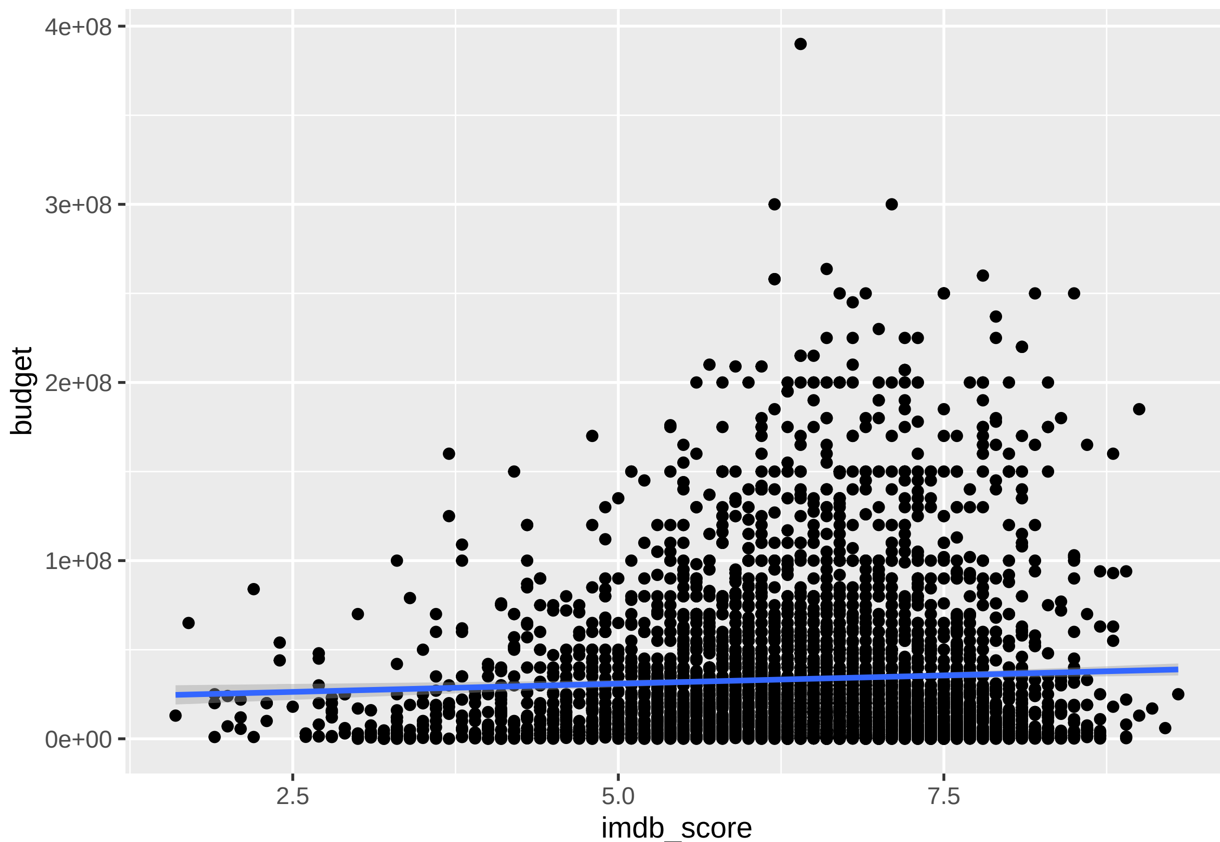
```
Here's how many unique films we have in the data.
```

```
length(unique(movies$movie_title))
```

```
## [1] 4423
```

- g) Let's plot a trendline with imdb_score and budget.

```
ggplot(movies, aes(x = imdb_score,y = budget)) + geom_point() + geom_smooth(method = "lm")
```



From this there doesn't seem to be a strong correlation but let's confirm.

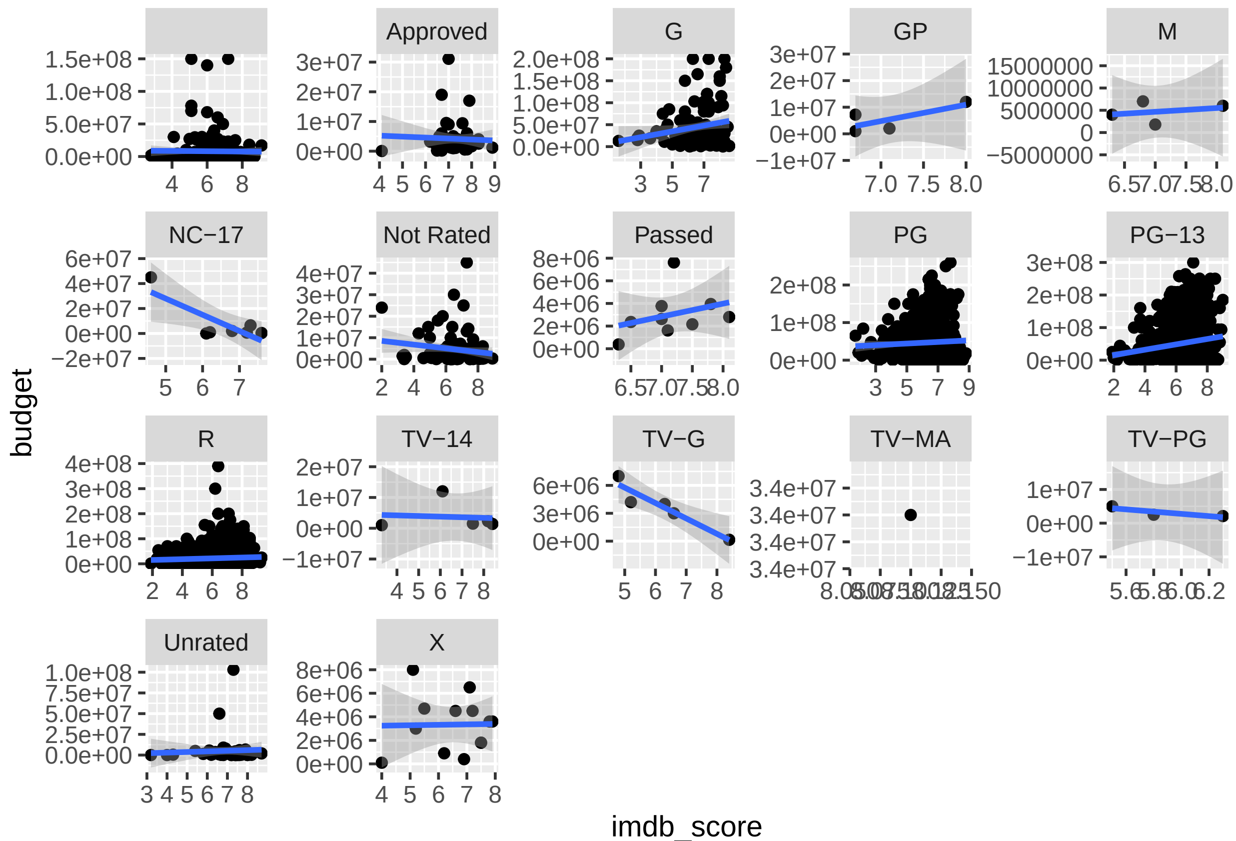
```
cor(movies$imdb_score, movies$budget)
```

```
## [1] 0.04933916
```

```
Yes, there is not a strong correlation between the two.
```

- h) Lets make some subplots of imdb_score and budget around content rating.

```
ggplot(movies, aes(x = imdb_score,y = budget)) + geom_point() + facet_wrap(~content_rating, scales = "f
```



It looks like Unrated and NC-17 films have the highest realationships.

Data Manipulation

Creating simplified budget and gross columns as well as a main genre column.

```
movies$grossM = movies$gross/1e+6
movies$budgetM = movies$budget/1e+6

movies$genre_main = do.call('rbind',strsplit(as.character(movies$genres), '|', fixed=TRUE))[,1]
```

```
## Warning in rbind(c("Action", "Adventure", "Fantasy", "Sci-Fi"),
## c("Action", : number of columns of result is not a multiple of vector
## length (arg 2)
```

- a) Let's create a new profit column as well as a Return on Investment variable

```
movies$profitM = movies$gross - movies$budget
movies$ROI = movies$profitM/movies$budget
```

- b) Displaying the mean ROI of movies.

```r
mean(movies$ROI, na.rm = TRUE)
```

```
## [1] 5.273088
```

- c) Here's a histogram of ROI.

```r
hist(movies$ROI, breaks = 10)
```

We see that some outliers in excess of about 1000 are throwing the histogram extremely off.

- d) We need to remove these extreme values from the data.

```r
sum(movies$ROI > 10, na.rm = TRUE)
```

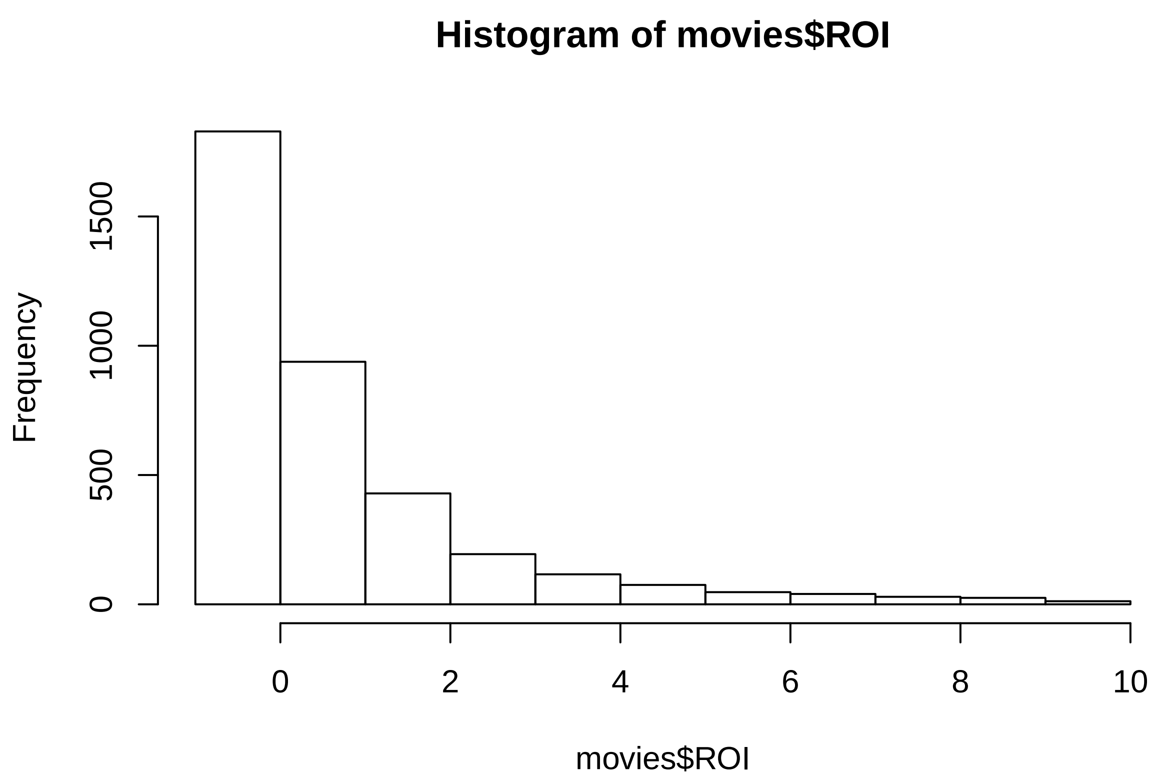
```
## [1] 145
```

```r
movies = movies[!movies$ROI > 10,]
dim(movies)
```

```
## [1] 4394   33
```

- e) It looks like fixing the outliers solved the problem.

```r
hist(movies$ROI, breaks = 10)
```

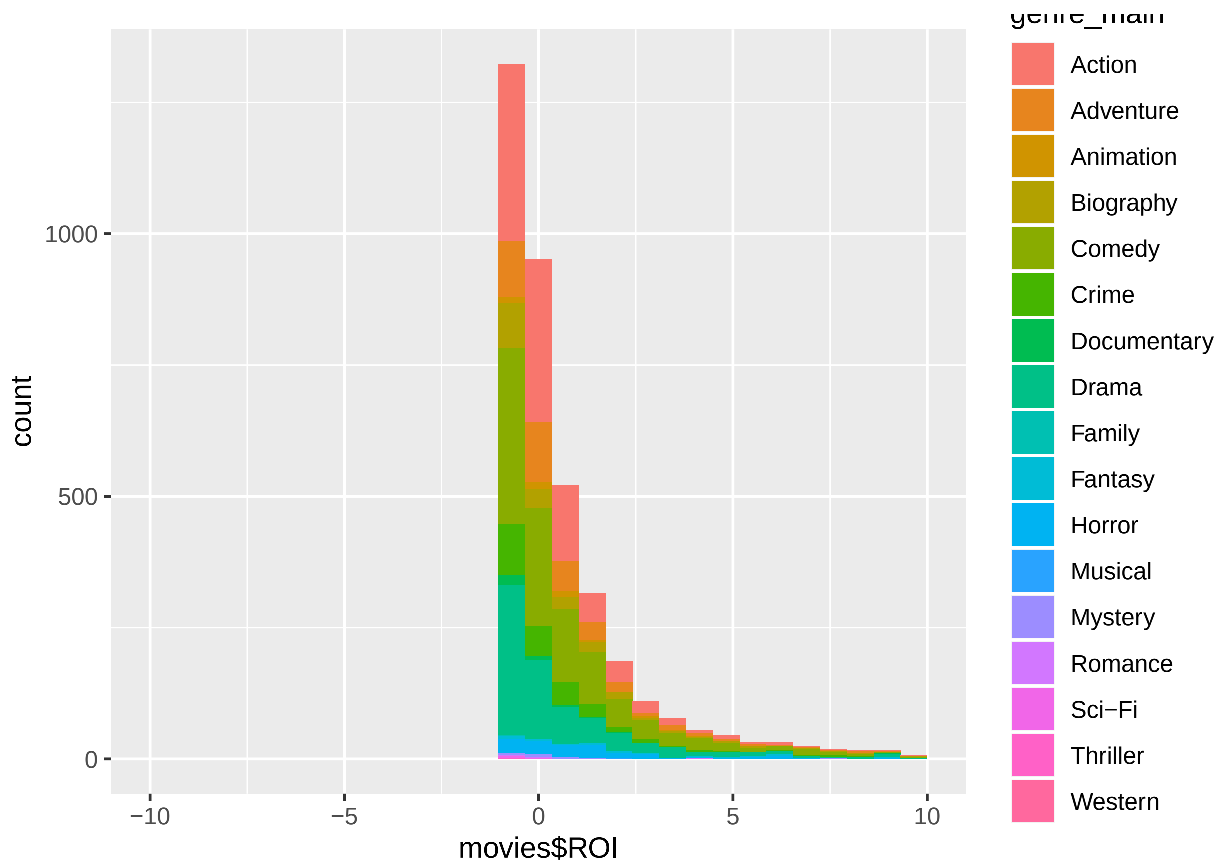


Here we use ggplot2 for the histogram.

```r
ggplot(movies, aes(x = movies$ROI, fill = genre_main)) + geom_histogram() + xlim(-10,10)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 660 rows containing non-finite values (stat_bin).
```



- f) Here's a summary of the ROI by main genre.

```
library(doBy)
movieSummary = summaryBy(ROI ~ genre_main, data = movies)
movieSummary
```

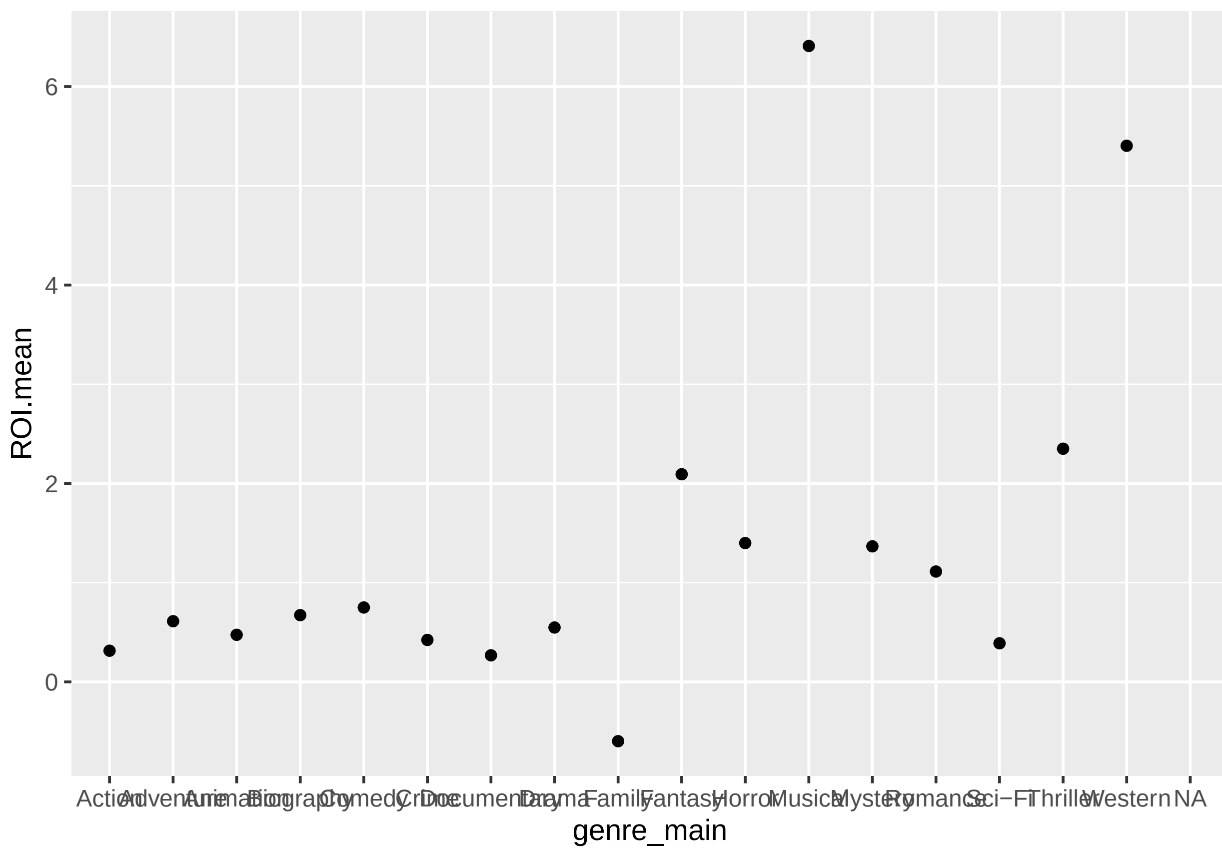
```
##     genre_main   ROI.mean
## 1       Action  0.3146972
## 2    Adventure  0.6117778
## 3    Animation  0.4749139
## 4    Biography  0.6730581
## 5       Comedy  0.7502510
## 6        Crime  0.4230916
## 7  Documentary  0.2681136
## 8        Drama  0.5484959
## 9       Family -0.5971447
## 10     Fantasy  2.0929081
## 11      Horror  1.3994674
## 12     Musical  6.4089710
## 13     Mystery  1.3665859
## 14     Romance  1.1126902
## 15      Sci-Fi  0.3892234
## 16    Thriller  2.3503454
## 17     Western  5.4029778
## 18        <NA>         NA
```

```
It looks like Fantasy, Musical, THriller, and Western have the highest mean ROI.
```

- g) Let's plot our results.

```r
ggplot(movieSummary, aes(y = ROI.mean, x = genre_main)) + geom_point()
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



Simple Linear Regression

- a) First we'll split the data 80:20 into a training and test set.

```r
set.seed(42)
sampleSize = floor(0.8*nrow(movies))
trainIndex = sample(seq_len(nrow(movies)),size = sampleSize)

train = movies[trainIndex,]
test = movies[-trainIndex,]
```

- b) Displaying the dimensions.

```r
dim(train)
```

```
## [1] 3515   33
```

```r
dim(test)
```

```
## [1] 879  33
```

- c) Now we'll regress profit against imdb_score.

```r
mod1 = lm(profitM ~ imdb_score,train)
summary(mod1)
```

```
##
## Call:
## lm(formula = profitM ~ imdb_score, data = train)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -308524875  -25077928   -9094722   14259930  494107863
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -67126324    5768315  -11.64   <2e-16 ***
## imdb_score   12218267     884038   13.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 50610000 on 2981 degrees of freedom
##   (532 observations deleted due to missingness)
## Multiple R-squared:  0.06022,    Adjusted R-squared:  0.0599
## F-statistic:   191 on 1 and 2981 DF,  p-value: < 2.2e-16
```

- d) Displaying the coefficients.

```r
coef(mod1)
```

```
## (Intercept)  imdb_score
##   -67126324    12218267
```

```
The coefficients represent our beta values for our linear equation.
Beta1 would be 12,218,267 which shows that for every unit of imdb_score,
profit is expected to increase by $12,218,267.
```