Problem Set 4 Movie Profitability

Jadyn Gonzalez

9/26/2019

Movie Profitability

We'll contiunue working with the previous movies dataset from last week. - a) First we need set our working directory then import the data.

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

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

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

- b) Next, using the given code, we will generate some new simplified variables. First we need to remove missing values.

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

```
 Next we'll simplify some variables and create a new profit variable.
```

```
movies = movies[movies$budget<4e+8,]
movies$grossM = movies$gross/1e+6
movies$budgetM = movies$budget/1e+6
movies$profitM = movies$grossM-movies$budgetM
movies$cast_total_facebook_likes000s = movies$cast_total_facebook_likes / 1000
```

```
 Finally we need to split our data into a training and test set.
```

```
set.seed(2019)
train_indx = sample(1:nrow(movies), 0.8 * nrow(movies), replace=FALSE)
train = movies[train_indx, ]
test = movies[-train_indx, ]
```

- c) Here are the number of rows for our train and test set. Train:

```
nrow(train)
```

```
## [1] 3103
```

```
 Test:
```

```
nrow(test)
```

```
## [1] 776
```

- d) Now, using the given code, we'll generate a correlation matrix from the numeric values in the data set and display it against profit.

```
nums = sapply(movies, is.numeric) # names of numeric variables
cormat = cor(movies[,nums], use="complete.obs")
print(cormat[,"profitM"])
```

```
##        num_critic_for_reviews                      duration
##                    0.24353361                    0.09423033
##       director_facebook_likes        actor_3_facebook_likes
##                    0.10485194                    0.17831580
##        actor_1_facebook_likes                         gross
##                    0.05850519                    0.78438560
##               num_voted_users     cast_total_facebook_likes
##                    0.50043953                    0.11507040
##          facenumber_in_poster          num_user_for_reviews
##                   -0.02128043                    0.38106102
##                        budget                    title_year
##                    0.02352410                   -0.11615920
##        actor_2_facebook_likes                    imdb_score
##                    0.12969431                    0.25215121
##                  aspect_ratio          movie_facebook_likes
##                   -0.05979073                    0.22941383
##                        grossM                       budgetM
##                    0.78438560                    0.02352410
##                       profitM cast_total_facebook_likes000s
##                    1.00000000                    0.11507040
```

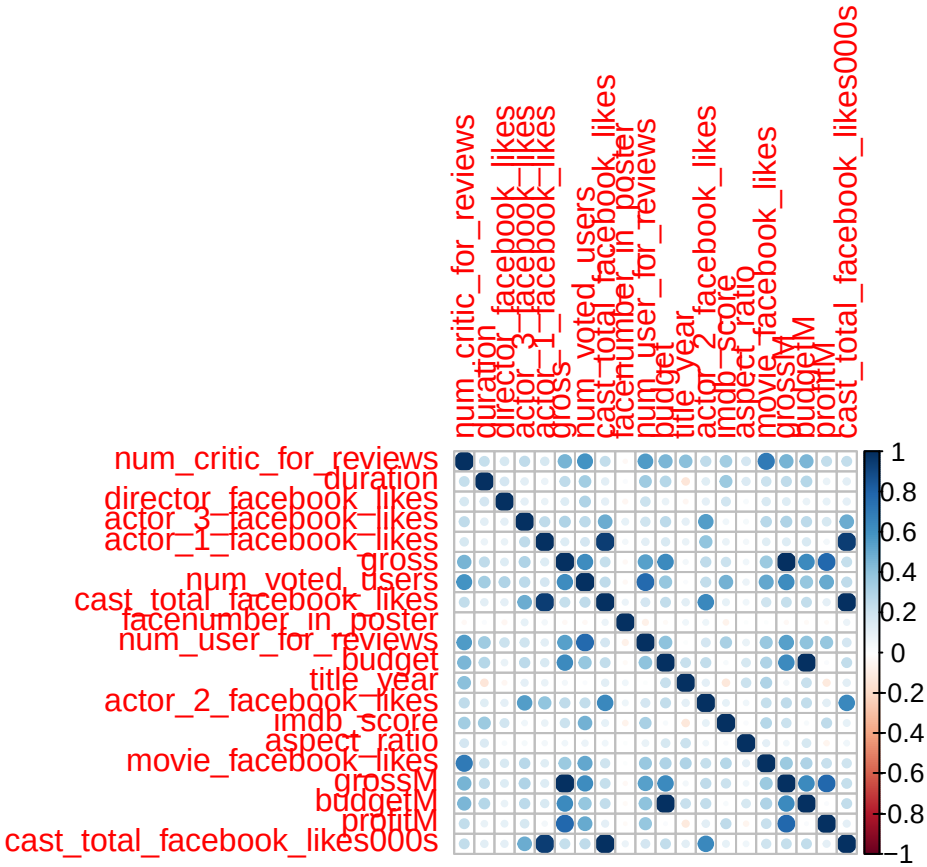
From this it looks like gross profit has a strong positive correlation and num_user_for_reviews has a moderate positive correlation. num_voted_users also has a moderate-strong poitive correlation with profit.

- e) Let's plot the correlation matrix.

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(cormat)
```



- f) Now we'll regress profit agains imdb_score and cast_total_facebook_likes000s.

```
mod1 = lm(profitM ~ imdb_score + cast_total_facebook_likes000s, data = train)
summary(mod1)
```

```
##
## Call:
## lm(formula = profitM ~ imdb_score + cast_total_facebook_likes000s,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -384.16  -25.27   -8.76   14.49  495.64
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -68.64310    5.77272 -11.891  < 2e-16 ***
## imdb_score                    12.01315    0.88830  13.524  < 2e-16 ***
## cast_total_facebook_likes000s  0.33117    0.05769   5.741 1.03e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.06 on 3100 degrees of freedom
## Multiple R-squared:  0.07082,    Adjusted R-squared:  0.07022
## F-statistic: 118.1 on 2 and 3100 DF,  p-value: < 2.2e-16
```

- g) The coefficient for cast facebook likes, rounded, is 0.33. This means that we expect to see an increase of profit by $0.33M for every 1000 cast facebook likes.
- h) The p-values are as follows: imdb_score - p = < 2e-16 cast_total_facebook_likes000s - p = 1.03e-08. P-value is defined as the probability of finding extreme values when H0 is true. Since both of the p-values are low we can reject H0 for both and say that both are significant.
- i) For variables to be statistically significant at the 95% confidence level, their p-value would have to be less than 0.05. WIth imdb_score having a very low p-value less than 0.05, it is likely that changes in imdb_score are related to changes in profit.
- j) R^2 is a meause of how good our model fits with the data. Since our R^2 is very low (0.07) it appears as if our model is not a good fit for the data. Adjusted R^2 is R^2 that has been adjusted for the added complexity of additional predictors in our model. It increases if the added predictor explains more of the data than expected and decreases when it explains less than expected. Our adjusted R^2 decreased so it appears as if our added predictor did not add any more explaination for our data.
- k) The F-stat gives us indication of joint significance of our predictors, basically it tells us if our coefficients improve our model. Since our F-stat was greater than 10, we can assume that our linear model is a better fit for the data than the constant (intercept only) model.
- l) Lets check the length of our residuals and make sure it is equal to our train set length.

```
length(mod1$residuals)
```

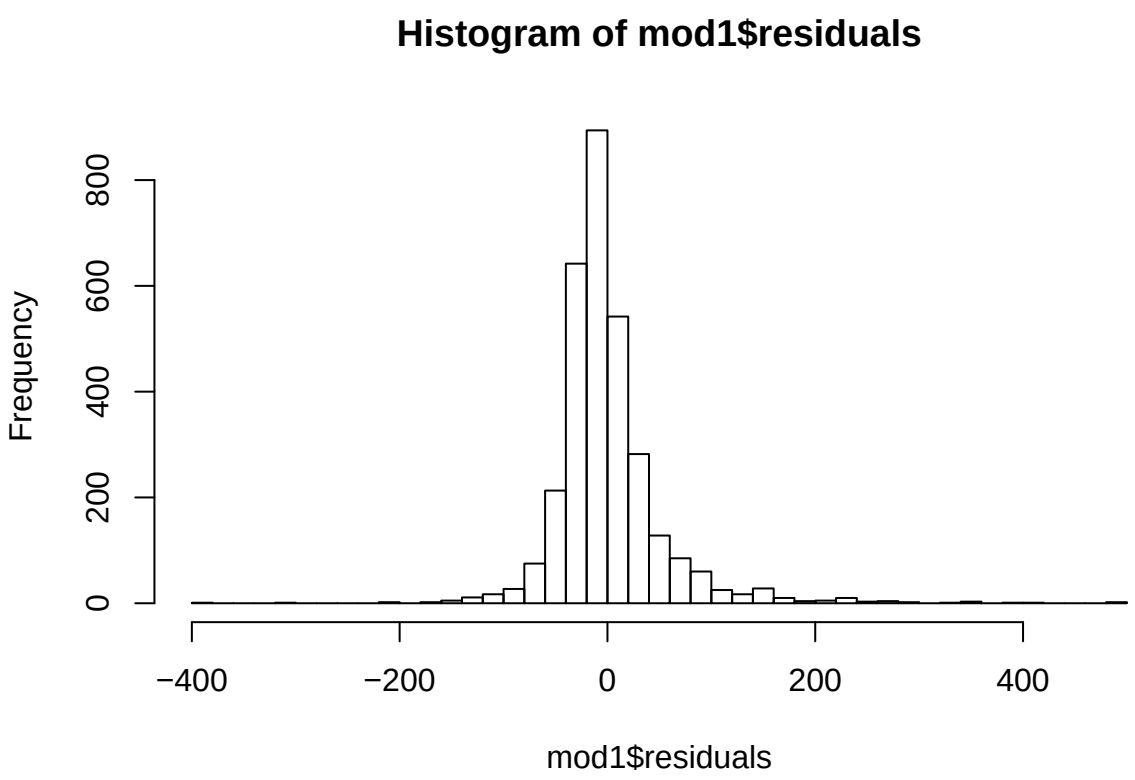
```
## [1] 3103
```

```
#we can compare the length of any column
length(train$profitM)
```

```
## [1] 3103
```

- m) Now we'll examine the histogram of the residuals.

```
hist(mod1$residuals, breaks = 40)
```



It appears to be a normal distribution.

- n) Calculating R^2 manually.

```
r2 = function(x,y0,y1)
{
  rss =  sum(((x + y0*train$imdb_score + y1*train$cast_total_facebook_likes000s) - train$profitM)^2)
  tss = sum((train$profitM - mean(train$profitM))^2)
  return(1 - rss/tss)
}

r2(coef(mod1)[1], coef(mod1)[2], coef(mod1)[3])
```

```
## [1] 0.07081586
```

This appears to be correct as our model gave an R^2 of the same value.