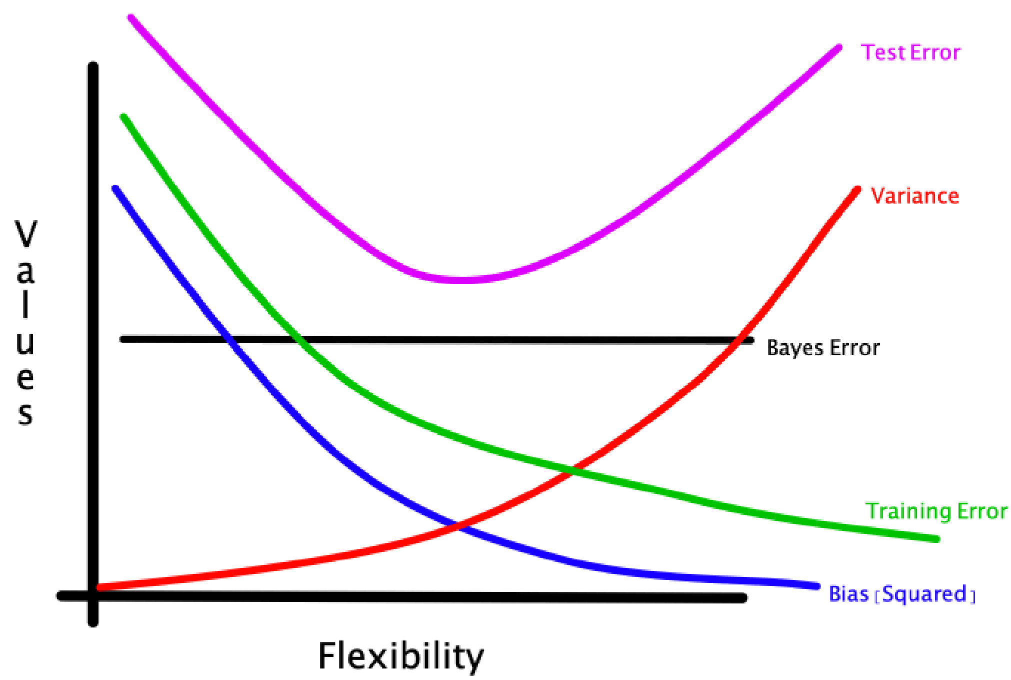
Problem Set 3

MGSC 310, Fall 2019, Professor Hersh (BEST PROFESSOR EVER!!!)

Geoffrey Hughes

9/20/2019

Question 1) ISLR Ch. 2, Problem 3



a.

b.

- Because the **Bayes Error** is the lowest possible error, it is plotted as constant (at 0).
- The **Bias (Squared)** generally tends to decrease as the flexibility level increases because the model becomes more complex and better fit, and so the bias becomes less important.
- As the flexibility of a model increases, the **variance** refers to how much the model's predictions will change with new (say test or training) data sets. So as our model increases in flexibility, the change will also increase.
- The **Training Error** minimizes as we better fit out model's function, so as the flexibility increases, the training error will decrease.
- **Test Error** also known as MSE (mean squared error) initially decreases to a minimum where the model is not over-fit or under-fit. This is where you want to be, and as the model becomes more complex, the it becomes over-fit, which in turn increases the test error (MSE). The model goes from underfit to overfit as flexibility increases, and the test error simply shows that parabolic relationship.

Question 2) What Predicts Movie Profitability?

a. Done.

b. Import movies dataset. Remove large outliers and create new variables.

```
library('tidyverse')
```

```
## -- Attaching packages ----------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts -------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
getwd()
```

```
## [1] "/Users/geoffreyhughes/Documents/MGSC_310/MGSC310/ProblemSets"
```

```
options(scipen = 10)
movies <- read.csv("/Users/geoffreyhughes/Documents/MGSC_310/MGSC310/Datasets/movie_metadata.csv")
movies <- movies %>% filter(budget < 400000000) %>% filter(content_rating != "",
content_rating != "Not Rated")
movies <- movies %>%
mutate(genre_main = unlist(map(strsplit(as.character(movies$genres),"\\|"),1)),
grossM = gross / 1000000,
budgetM = budget / 1000000,
profitM = grossM - budgetM,
cast_total_facebook_likes000s = cast_total_facebook_likes / 1000)
movies <- movies %>% mutate(genre_main = factor(genre_main) %>% fct_drop())
```

c. Split dataset into Training (80%) and Testing (20%)

```
set.seed(1861)
sample <- sample.int(n = nrow(movies), size = floor(0.8 * nrow(movies)), replace = FALSE)
movies_train <- movies[sample, ]
movies_test  <- movies[-sample, ]
```

d. How many rows in each dataset? (Test & Train)

```
dim(movies_train)
```

```
## [1] 3396   33
```

```r
dim(movies_test)
```

```
## [1] 849  33
```

- There are 4000 rows in the training dataset, and 1000 rows in the testing dataset!

e. Create a coorelation matrix, and print out variables coorelations with ProfitM. What are most strongly coorelated with ProfitM?

```r
cormat <- cor(movies_train %>% select_if(is.numeric) %>% drop_na())
print(cormat[, "profitM"])
```

```
##        num_critic_for_reviews                       duration
##                  0.2416979638                   0.1063892917
##       director_facebook_likes         actor_3_facebook_likes
##                  0.1070303835                   0.1665145723
##        actor_1_facebook_likes                          gross
##                  0.0472643456                   0.7859808359
##               num_voted_users      cast_total_facebook_likes
##                  0.4861649378                   0.0988924207
##           facenumber_in_poster           num_user_for_reviews
##                 -0.0196624052                   0.3693420547
##                        budget                     title_year
##                  0.0005501269                  -0.1205250671
##        actor_2_facebook_likes                     imdb_score
##                  0.1196320100                   0.2609408853
##                  aspect_ratio           movie_facebook_likes
##                 -0.0602042210                   0.2386962438
##                        grossM                        budgetM
##                  0.7859808359                   0.0005501269
##                       profitM cast_total_facebook_likes000s
##                  1.0000000000                   0.0988924207
```

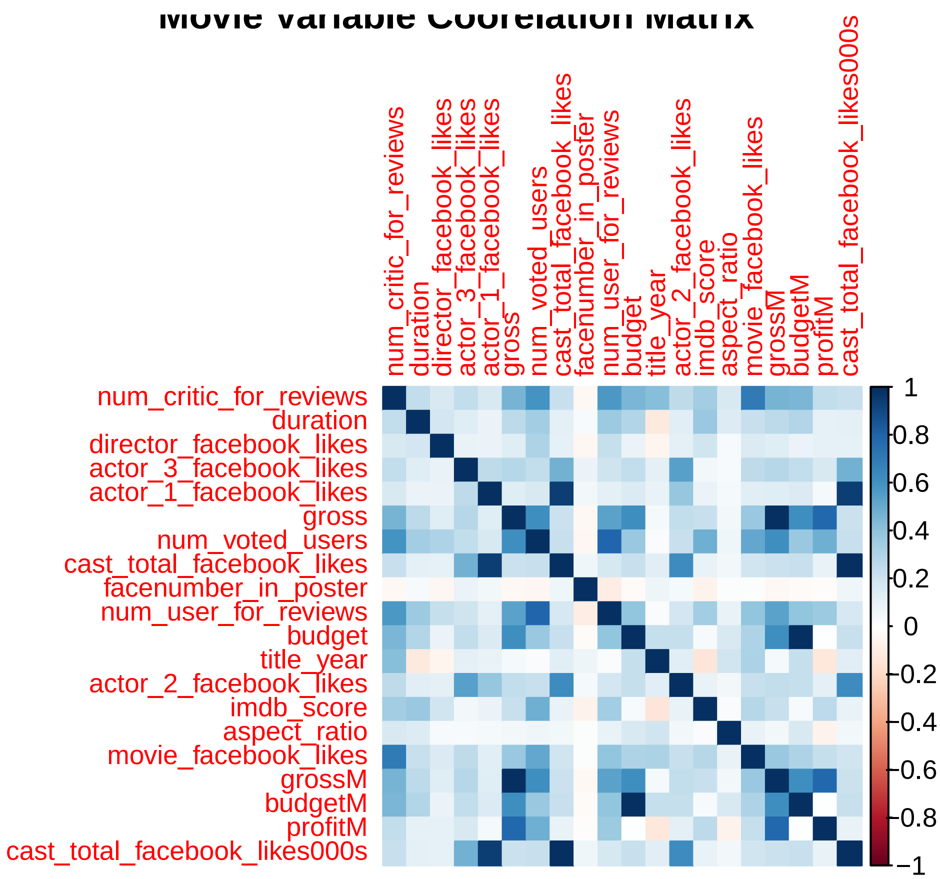
- Some of the most influential vatiables coorelated with ProfitM are, in descending order: grossM/gross, num_voted_users, num_user_for_reviews, imdb_score, num_critic_for_reviews, and movie_facebook_likes

f. *Extra Credit:* Plot the Coorelation Matrix with corrplot (I chose color)

```r
library('corrplot')
```

```
## corrplot 0.84 loaded
```

```r
corrplot(cormat, method = 'color', title = 'Movie Variable Coorelation Matrix', tl.cex = 0.8)
```



g. Regressive Model of profitM against imdb_score with training dataset

```
mod1 <- lm(profitM ~ imdb_score,
           data = movies_train)
summary(mod1)
```

```
##
## Call:
## lm(formula = profitM ~ imdb_score, data = movies_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -389.02  -26.38   -9.65   16.38  490.35
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -72.1702     5.8946  -12.24   <2e-16 ***
## imdb_score   13.3319     0.9019   14.78   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.53 on 3027 degrees of freedom
##   (367 observations deleted due to missingness)
## Multiple R-squared:  0.06732,    Adjusted R-squared:  0.06701
## F-statistic: 218.5 on 1 and 3027 DF,  p-value: < 2.2e-16
```

h. Interpretting the imdb_score coefficient

- The imdb_score coefficient is 13.3319 (magnitude), which means that **for every 1 unit change of imdb_score, the movie's profitM will, on average, change by 13.3319.** Since this coefficient is positive, that means that a positive change will elicit a positive change in profitM, and silimarly a negative change to imdb_score will elicit a negative change in profitM (on average).

i. Interpretting the imdb_score P-value

- The imdb_score's P-value is 2.2e-16, or 0.0000000000000002.
- P-value is the probability, given there is no relationship at all between the dependent and independent variables (H0), that the magnitude, or significance, (coefficient) would be this extreme or even more extreme.
- If we assume any reasonable alpha, say 0.05, or even 0.001, we can say that **there is a relationship between imdb_score and profitM**, and we reject the null hypothesis (that there is no relationship).

j. Regressive Model of profitM using imdb_score and cast_total_facebook_likes000s

```
mod2 <- lm(profitM ~ imdb_score + cast_total_facebook_likes000s,
           data = movies_train)
summary(mod2)
```

```
##
## Call:
## lm(formula = profitM ~ imdb_score + cast_total_facebook_likes000s,
##     data = movies_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -386.71  -25.96   -9.25   16.22  492.26
##
## Coefficients:
##                               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)                  -72.09743    5.87758  -12.27   < 2e-16 ***
## imdb_score                    12.95458    0.90359   14.34   < 2e-16 ***
## cast_total_facebook_likes000s   0.20710    0.04805    4.31 0.0000168 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.37 on 3026 degrees of freedom
##   (367 observations deleted due to missingness)
## Multiple R-squared:  0.07301,    Adjusted R-squared:  0.0724
## F-statistic: 119.2 on 2 and 3026 DF,  p-value: < 2.2e-16
```

k. Impact of cast_total_facebook_likes000s on profitM

- For every 1 unit change in cast_total_facebook_likes_000s, there will be, on average, a change of 0.2071 to profitM. So a positive change of +1 to cast_total_facebook_likes000s will yield (on average) that movie 207.1k more in profit.

l. Add Variable rating_simple to movies_train (G, PG, PG-13, R, Other) using fct_lump()

```r
movies_train <- movies_train %>% mutate(rating_simple =
                                    fct_lump(movies_train$content_rating, n = 4, ties.method = "ma
table(movies_train$rating_simple)
```

```
##
##     G    PG PG-13     R Other
##    92   509  1107  1559   129
```

m. Regressive Model of profitM using imdb_score, cast_total_facebook_likes000s & Interpret rating_simpleR's Coefficient

```r
mod3 <- lm(profitM ~ imdb_score + cast_total_facebook_likes000s + rating_simple,
           data = movies_train)
summary(mod3)
```

```
##
## Call:
## lm(formula = profitM ~ imdb_score + cast_total_facebook_likes000s +
##     rating_simple, data = movies_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -378.42  -24.53   -7.55   17.12  486.11
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -64.55692    8.27213  -7.804 8.19e-15 ***
## imdb_score                    14.08094    0.90685  15.527  < 2e-16 ***
## cast_total_facebook_likes000s   0.19101    0.04778   3.998 6.55e-05 ***
## rating_simplePG               -1.11971    6.29605  -0.178 0.858857
## rating_simplePG-13           -10.21472    6.01564  -1.698 0.089605 .
## rating_simpleR               -23.03335    5.95321  -3.869 0.000112 ***
## rating_simpleOther           -18.26624    9.17181  -1.992 0.046509 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.73 on 3022 degrees of freedom
##   (367 observations deleted due to missingness)
## Multiple R-squared:  0.0968, Adjusted R-squared:  0.095
## F-statistic: 53.98 on 6 and 3022 DF,  p-value: < 2.2e-16
```

- The rating_simpleR coefficient is -23.03335, which means that if the movie is rated R, it will make (on average) -23.03335 million less profit than if it was rated G (baseline for the rating_simple categorical variable).

n. Why do we not see rating_simpleG?

- This is because with categorical variables, we can only compare them against one another, as each movie is binary as to which movie category it is in. As such, we must have a baseline coefficient for rating_simple, and in this case that is G (rating_simpleG). All of the other ratings are comparing themselves to rating_simpleG, which is therefore 0 and the baseline.