Stat 306

Term Project: Multiple Regression Analysis of IMDB Movie Data Set

April 07, 2017

Team Members:
Sam McConnell
Kelly Ng
Luke Schwalfenberg
Richard Wang

Table of Contents

Abstract/Summary

The goal of this investigation is to form a predictive equation for Box Office Gross Earnings. The primary explanatory variables being considered for the model are: Director, Number of Critic Reviews, Duration, Lead Actor/Actress, Number of User Votes, Content Rating, Budget, IMDB Score and in some cases the number of Actor/Actress/Director's facebook likes and some other variables. After some initial analysis, it made sense to create several different subsets for this data set since there are 2399 unique director names, and several thousand different actresses and actors that star in the films examined. The subsets that were chosen for consideration included a model that does not include lead actor/actresses or directors and consists primarily of numerical data, a model which includes directors who have directed more than 3 movies and a model which includes actor/actresses appearing in more than 25 movies. To get a regression model or equation so that the homoscedasticity assumption approximately holds for the models considered, the response variable was transformed to log(BoxOfficeGrossEarnings).

Description of Data

In this case study, data was retrieved from a data set made publically available on Kaggle by user chuansun76 who had used python scripts to mine this data from Imdb. This data set includes roughly 5000 individual entries with the earliest movie dating from 1916. Roughly 50% of the dataset is from movies released after 2004. The values of Gross Revenue and Budget were in USD. Sample sizes varied depending on the model being examined. Since a number of key values were missing for data points, no sample size was as large as that of the original data set. Some of the explanatory variables examined can be found in Table 1.

Table 1: Tables of Variables That Might Explain Movie Box Office Gross

Variables	Explanation or Unit
Box Office Gross Earnings	Movie gross box office earnings in USD
Director	Categorical variable for directors
Number of Critic Reviews	Measurement of movie critical success
Lead Actor/Actress	Categorical variable for lead actor actresses
Number of User Votes	Measurement of movie popular success, the number of users who voted on Imdb
Content Rating	Content rating: G, PG, PG-13, R, X
Budget	Movie budget in USD
IMDB Score	Score on a scale of 0.0 - 10.0
Actor	Lead Actor in film

Data Analysis and Results

Initial Numerical Analysis

Summary statistic and plots of the main numerical variables can be found in Figures 1, 2 and 3 and in Tables 2 - 6. Figure 3 suggests that log(Gross Revenue) should be transformed so that the deviations from the prediction equation can be closer to homoscedastic.

Figure 1: Plot of Gross Revenue and log(Gross Revenue) versus Imdb Score

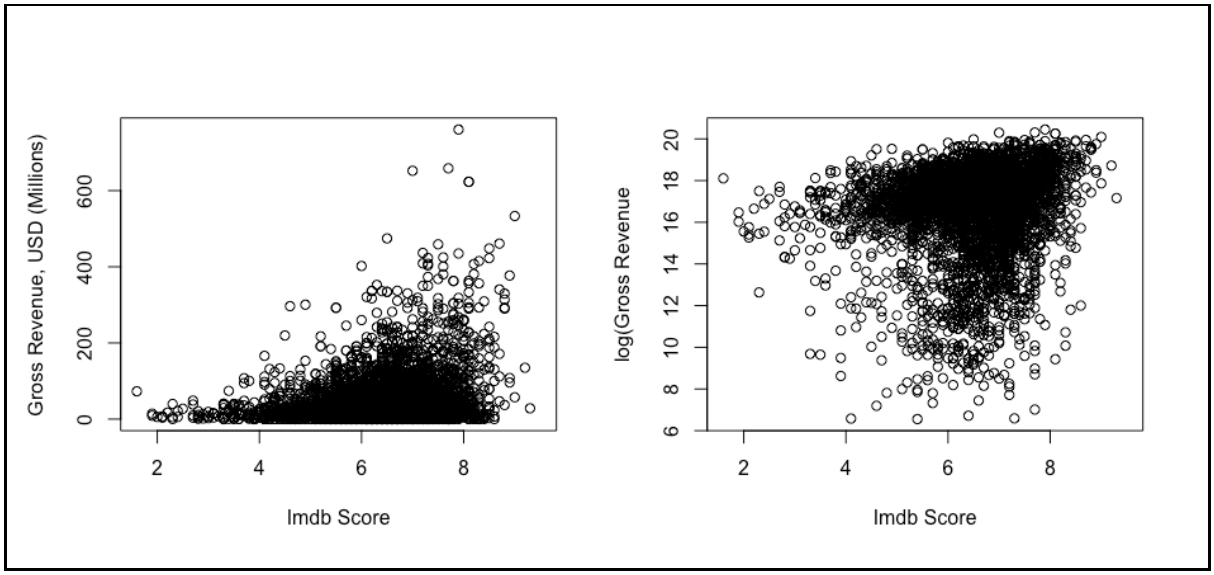


Figure 2: Box plot of log(Gross Revenue) vs. Content Rating. Far right box plot is for those films that did not have a content rating associated with them in the data set.

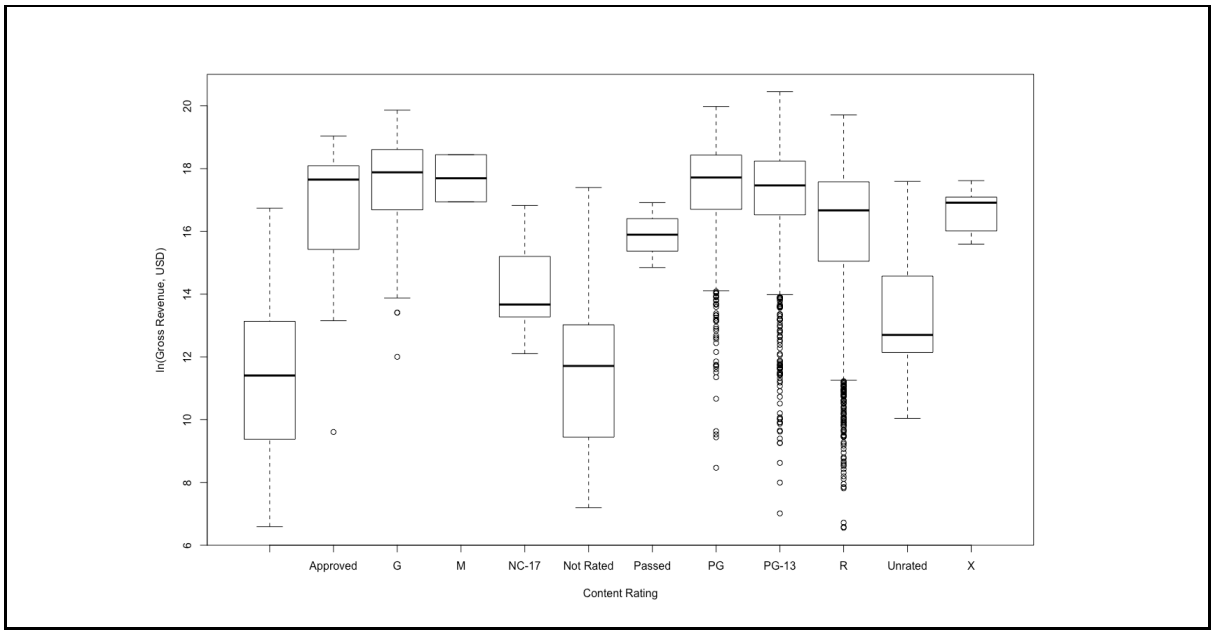
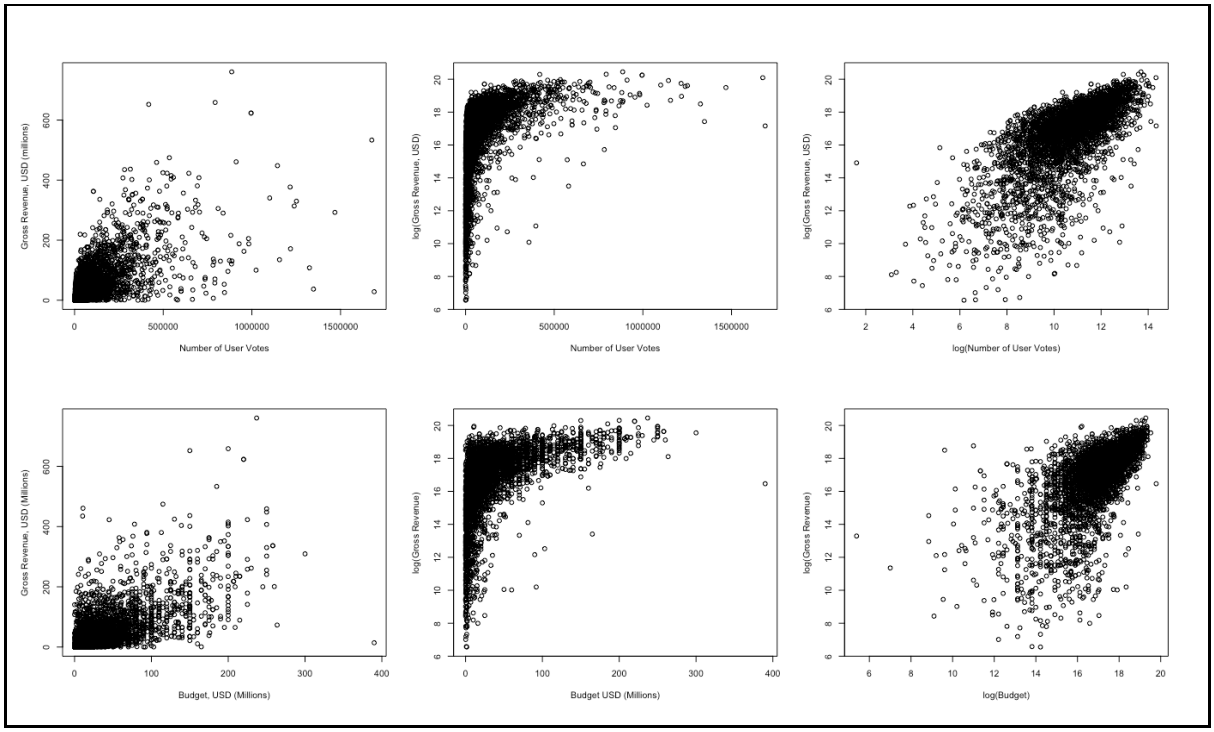


Figure 3: Plots of Gross Revenue and log(Gross Revenue) before adjusting for inflation versus the numerical explanatory variables, and plots of log(Gross Revenue) vs. logged versions of the numerical explanatory variables



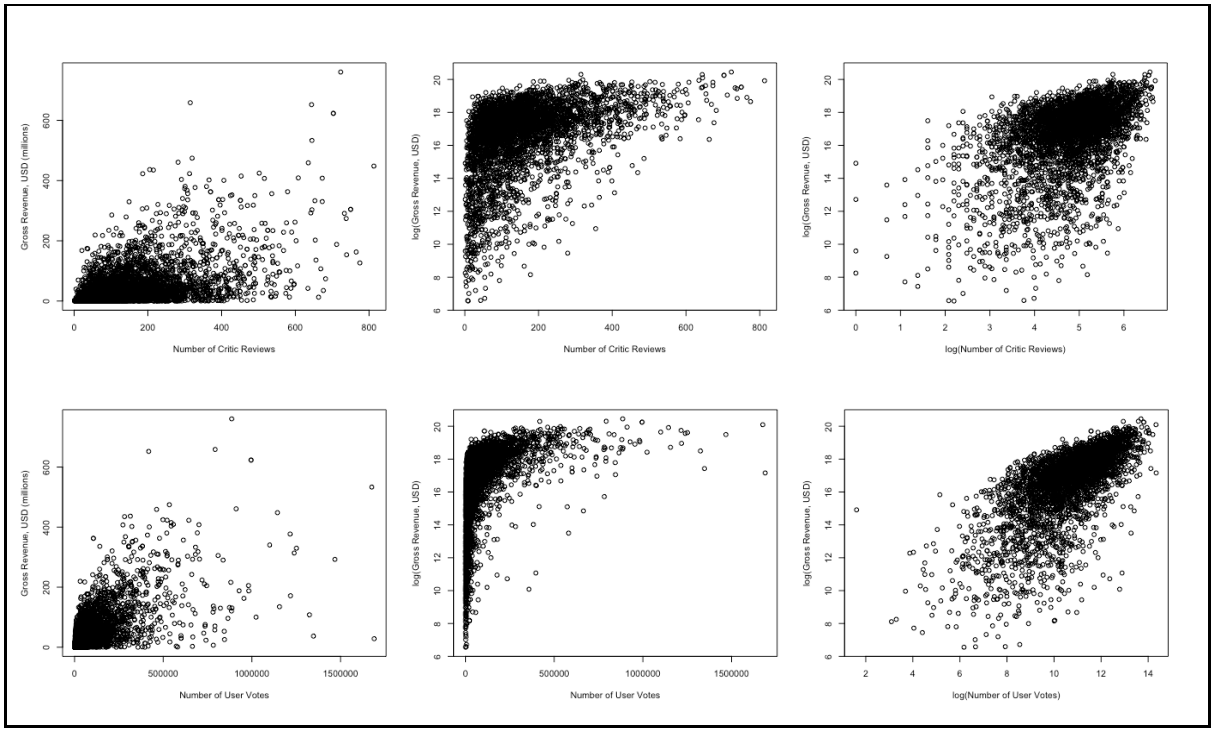


Table 2: Summary Statistics of Variables included in Data Sets

	num_critic_for_reviews	lnnum_critic_reviews	duration	director_facebook_likes	actor_3_facebook_likes
Min.	1	0	7	0	0
1st Qu.	53	3.97	93	7	165.2
Median	113	4.727	103	52	399
Mean	143.5	4.508	106.8	726.4	684.1
3rd Qu.	198	5.288	117	209	651
Max.	813	6.701	334	23000	23000

Table 3: Continuation of Summary Statistics of Variables that may be included in data sets

	actor_1_facebook_likes	gross	lngross	num_voted_users	lnnum_voted_users
Min.	0	703	6.555	5	1.609
1st Qu.	670	7164367	15.785	9808	9.191
Median	1000	28429808	17.163	37462	10.531
Mean	6951	51070645	16.555	87523	10.186
3rd Qu.	11000	65526300	17.998	102110	11.534
Max.	640000	760505847	20.45	1689764	14.34

Table 4: Continuation of Summary Statisticcs of Variables that may be included in data sets

	cast_total_facebook_likes	facenumber_in_poster	num_user_for_reviews	budget (USD)	title_year
Min.	0	0	1	218	1916
1st Qu.	1622	0	73	6500000	1999
Median	3339	1	165	20000000	2005
Mean	10290	1.384	284.4	34484957	2002
3rd Qu.	14578	2	340	45000000	2010
Max.	656730	43	5060	390000000	2016

Table 5: Continuation of Summary Statistic of Variables that may be included in data sets

	actor_2_facebook_likes	imdb_score	aspect_ratio	movie_facebook_likes
Min.	0	1.6	1.18	0
1st Qu.	329	5.8	1.85	0
Median	631	6.5	2.35	175.5
Mean	1755	6.409	2.209	7797.7
3rd Qu.	936	7.2	2.35	4000
Max.	137000	9.5	16	349000

Table 6: Sample Correlations of Select Explanatory Variables

	num_critic_for_reviews	duration	gross	num_voted_users	num_user_for_reviews	budget	imdb_score	lngross	lnnum_critic_reviews l	lnnum_voted_users
num_critic_for_reviews	1.00	0.26	0.48	0.60	0.57	0.48	0.35	0.41	0.86	0.68
duration	0.26	1.00	0.28	0.36	0.38	0.31	0.37	0.25	0.23	0.34
gross	0.48	0.28	1.00	0.64	0.55	0.65	0.24	0.62	0.41	0.56
num_voted_users	0.60	0.36	0.64	1.00	0.78	0.40	0.49	0.42	0.50	0.70
num_user_for_reviews	0.57	0.38	0.55	0.78	1.00	0.42	0.33	0.39	0.50	0.61
budget	0.48	0.31	0.65	0.40	0.42	1.00	0.05	0.49	0.42	0.45
imdb_score	0.35	0.37	0.24	0.49	0.33	0.05	1.00	0.13	0.32	0.43

lngross	0.41	0.25	0.62	0.42	0.39	0.49	0.13	1.00	0.50	0.69
lnnum_critic_reviews	0.86	0.23	0.41	0.50	0.50	0.42	0.32	0.50	1.00	0.80
lnnum_voted_users	0.68	0.34	0.56	0.70	0.61	0.45	0.43	0.69	0.80	1.00

Overall Analysis with Quantitative Variables

The goal of this analysis is to determine a regression equation based solely on quantitative explanatory variables. The best prediction equation, after residual plots, variable selection and using criteria of adjusted R^2 and cross-validated root mean square prediction error is:

$$\log(\text{adjgross}) = -14.342 - 0.438(\text{num\_critic\_for\_reviews}) - 0.045(\text{director\_facebook\_likes}) - 0.045(\text{director\_facebook\_likes}) - 0.410(\text{actor\_1\_facebook\_likes}) + 0.843(\text{num\_voted\_users}) + 0.344(\text{cast\_total\_facebook\_likes}) + 0.266(\text{num\_user\_for\_reviews}) + 0.472(\text{budget}) - 0.094(\text{imdb\_score})$$

According to the regression on quantitative variables (Tables 7 and 8), almost all the variables are significant. num_voted _users has the highest positive coefficient of 0.843, while num_critic_for_reviews appears to have the largest negative coefficient of -0.438. Therefore, in our next analyses, we will include categorical variables that incorporate directors and actors.

Table 7: Multiple Regression Summary of Quantitative Analysis

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-14.34181	0.51136	-28.047	< 2e-16	***
num_critic_for_reviews	-0.43824	0.07237	-6.056	1.74e-09	***
director_facebook_likes	-0.04461	0.02305	-1.935	0.053138	.
actor_1_facebook_likes	-0.41035	0.08652	-4.743	2.30e-06	***
num_voted_users	0.84273	0.06342	13.288	< 2e-16	***
cast_total_facebook_likes	0.34444	0.10268	3.354	0.000814	***
num_user_for_reviews	0.26582	0.07839	3.391	0.000714	***
budget	0.4724	0.03179	14.861	< 2e-16	***
imdb_score	-0.094	0.04179	-2.249	0.024631	*

Backward Elimination and Forward Selection algorithms were applied to determine the best regression model which is shown below:

Table 8: Cross Validation and Out of Sample Comparisons for Quantitative Analysis

statistic\model	Full Model	Best Model
adjusted R squared	0.5638149	0.5641097
residual SD	1.462724	1.462229
rmsepred(leave-one-out)	1.470517	1.468138
rmsepred(train/holdout)	1.479248	1.473195

Directors Analysis

This purpose of this separate analysis is to look at the dataset with a focus on directors who have more than 3 films in their filmography with respect to our dataset. The goal of this subset analysis is to find a prediction equation for predicting box office gross with a focus on directors.

Similar to the other models in this report, the response variable of adjusted gross was transformed to log(adjgross).

The best prediction equation, after residual plots, variable selection and using criteria of adjusted R^2 and cross-validated root mean square prediction error is:

$$\log(\text{adjgross}) = 7.402 - 0.134(\text{lncast\_total\_facebook\_likes}) - 0.00341(\text{lnactor1\_facebook\_likes}) - 0.004(\text{lndirector\_facebook\_likes}) + 1.08(\text{lnnum\_voted\_users}) - 0.306(\text{lnnum\_users\_for\_review}) + 0.106(\text{lnbudget}) - 0.192(\text{imdb\_score}) - 0.950(\text{content\_ratingPG-13}) - 1.06(\text{content\_ratingR}) + 0.007(\text{duration}) + 0.382(\text{BobbyFarrelly}) + 1.06(\text{BrettRatner}) + 0.776(\text{ClintEastwood}) - 0.071(\text{DavidFincher}) + 1.043(\text{JoelSchumacher}) + 1.066(\text{JohnMcTiernan}) - 0.170(\text{MartinScorsese}) + 0.143(\text{PaulWSAnderson}) + 0.654(\text{RennyHarlin}) + 1.40(\text{RichardDonner}) + 1.33(\text{RobCohen}) + 0.547(\text{RonHoward}) + 0.52(\text{ShawnLevy}) + 0.616(\text{StephenFrears}) + 1.01(\text{StevenSpielberg}) + 0.590(\text{TimBurton}) + 0.643(\text{TonyScott})$$

Note that director categorical variables are not grouped together. This is because directors act like a brand, and grouping them together would not provide any useful analysis as directors do not work together in real life.

Significant categorical variables include Steven Spielberg, Rob Cohen, Clint Eastwood and Brett Ratner. A review of their films show that their films are not only typically high grossing but very profitable.

Steven Spielberg has a high coefficient of 1.009 to the log(adjgross), Rob Cohen is 1.33, Clint Eastwood is 0.776 and Brett Ratner is 1.06.

Description of Data (Director's Analysis)

Data are collected by scraping both imdb resources and Box Office Mojo websites. Assumptions were made regarding the adjusted gross (estimated data from Box Office Mojo) which are outlined in Appendix III. The variables used in this analysis can be found in Table 9.

Table 9: Tables of variables that might explain movie box office gross for director's analysis

Variables	Explanation or unit
adjgross	Gross US domestic box office, adjusted for inflation
director_facebook_likes	Number of Facebook likes a director has
actor_1_facebook_likes	Number of Facebook likes from the lead actor
duration	Duration of the film (in minutes)
num_voted_users	Number of users who have voted for film on imdb
cast_total_facebook_likes	Number of Facebook likes received by the cast
Budget	Budget for the film
num_users_for_review	Number of uses that have reviewed the film
movie_facebook_likes content_rating director_name	Number of Facebook likes received by the film Rating for the film (e.g. PG, R, NC-17, etc.) Film directors that have made more than 3 films

This particular data set subsets the original data set such that only directors that have made more than 3 films are included. This reduces the dataset from over 5000 observations to approximately 400.

Directors are not combined into fewer categorical variables because in practical situations, directors do not work together and their contributions to the movie industry are considered singular.

Data Analysis and Results

Summary statistics are provided in Tables 10, 11, 12 and 13. Plots that justify the use of transforms for explanatory variables are given in Figure 4 and remain unchanged through this subset analysis.

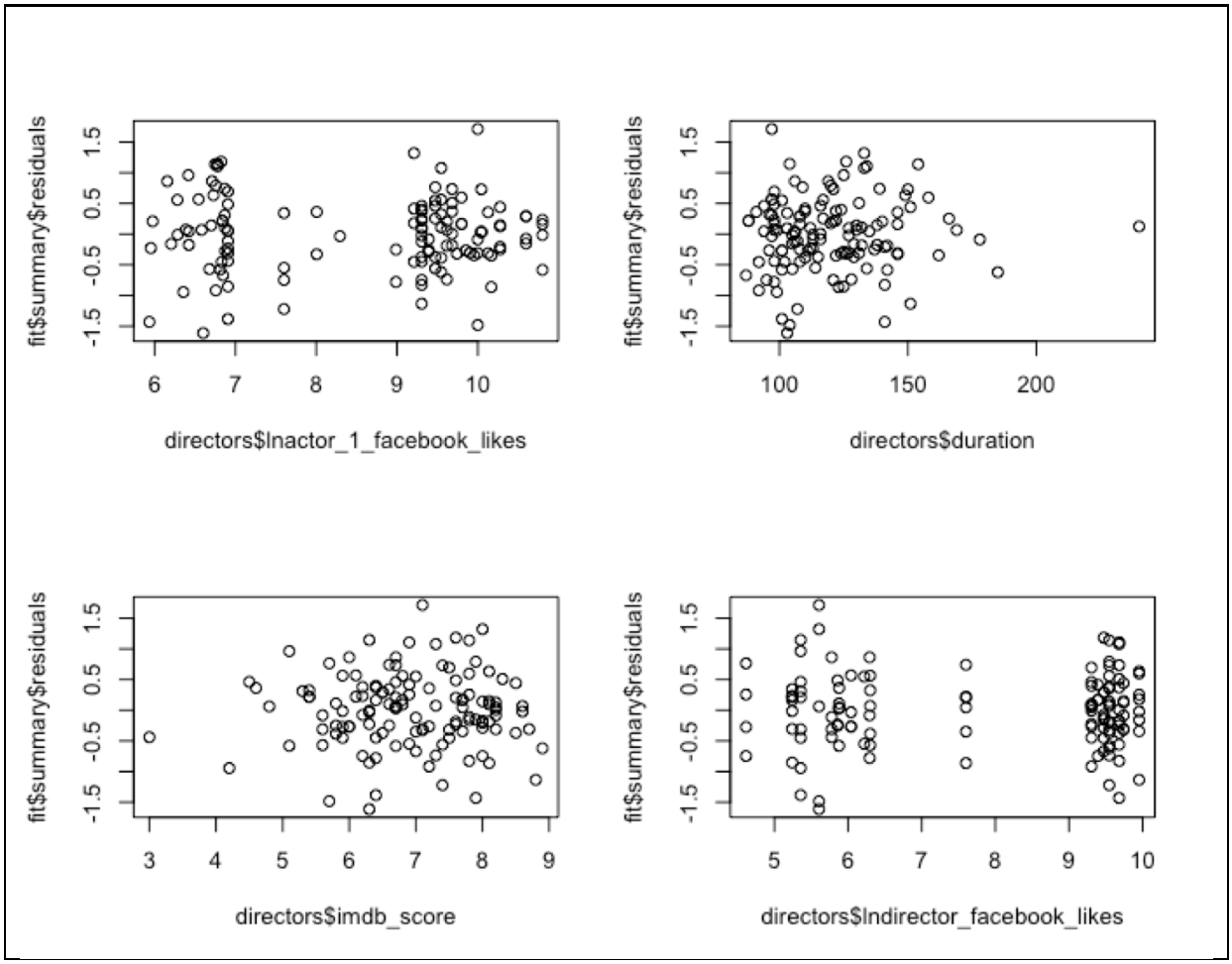
The sample correlation matrix for this analysis is in Appendix II. Note the lnadjgross is more highly correlated with lnnum_user_for_reviews (0.657) and lnmovie_facebook_likes (0.594).

We fit a multiple regression model (Table 14) with the explanatory variables given above. The adj-R^2 of this model is 0.7412 with a Residual Standard Error of 0.6683.

Residual Plots

Sample residual plots are given in Figure 4. The residuals do not suggest any curvilinear form, therefore quadratic terms or interaction terms were not added to the model.

Figure 4: Residual plots for some explanatory variables included in the director's analysis



Backward Elimination and Forward Selection techniques were used to determine other good models. The 2nd best model eliminated two explanatory variables: lntotal_facebook_likes and lndirector_facebook_likes. This model, as well as the full model, are compared via Cp statistic, adj-R^2 and error from a random holdout set (Table 15).

Table 10: Summary statistics of variables included in directors' regression analysis

	lnnum_voted_users	lncast_total_facebook_likes	lnbudget	lnnum_users_for_review	lnmovie_facebook_likes
Min.	7.243	6.252	14.08	3.258	4.883
1st Q	10.587	8.12	16.81	5.13	7.905
Median	11.789	9.483	17.5	5.807	9.547
Mean	11.644	9.177	17.5	5.826	9.079
3rd Q	12.615	10.096	18.06	6.539	10.127
Max.	14.114	11.3	19.11	7.996	11.891

Table 11: Continuation of summary statistics of variables included in director's regression analysis

	lnadjgross	lngross	lndirector_facebook_likes	lnactor_1_facebook_likes	duration
Min.	11.63	11.23	4.615	5.935	87
1st Q	17.34	17.08	5.858	6.872	103
Median	18.05	17.69	9.306	9.306	116
Mean	17.92	17.64	7.858	8.575	119.8
3rd Q	18.83	18.44	9.547	9.695	132
Max.	20.81	19.89	9.952	10.8	240

Table 12: Frequency Table for Directors with >= 3 Movies

Barry Levinson	Bobby Farrelly	Brett Ratner	Clint Eastwood	David Fincher
5	4	4	13	8
Joel Schumacher	John McTiernan	Martin Scorsese	Paul WS Anderson	Renny Harlin
3	4	6	5	9
Rob Cohen	Ron Howard	Shawn Levy	Stephen Frears	Steven Spielberg
6	6	8	3	17
Tim Burton	Tony Scott	Woody Allen		
9	4	12		

Table 13: Frequency Table for Content Rating

Approved	G	GP	M	NC-17
303	55	112	6	5
PG-13	R	TV-14	TV-G	TV-MA
1461	2118	30	10	20
Not	Rated	Passed	PG	X
7	116	9	701	13
TV-PG	TV-Y	TV-Y7	Unrated	
13	1	1	62	

Table 14: Multiple Regression Summary for director's analysis

Variable Name	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	7.402044	2.343641	3.158	0.0021	**
lncast_total_facebook_likes	-0.133733	0.227064	-0.589	0.55721	
lnactor_1_facebook_likes	-0.003454	0.180389	-0.019	0.98476	

lndirector_facebook_likes	-0.003638	0.112191	-0.032	0.97419	
lnnum_voted_users	1.075662	0.177303	6.067	2.34E-08	***
lnnum_users_for_review	-0.306319	0.222169	-1.379	0.17104	
lnbudget	0.105663	0.115545	0.914	0.36267	
imdb_score	-0.191828	0.134387	-1.427	0.15657	
content_ratingPG-13	-0.949634	0.232948	-4.077	9.18E-05	***
content_ratingR	-1.064077	0.26723	-3.982	0.00013	***
duration	0.006723	0.005051	1.331	0.18622	
director_nameBobby Farrelly	0.382473	0.591534	0.647	0.51939	
director_nameBrett Ratner	1.061384	0.509952	2.081	0.03996	*
director_nameClint Eastwood	0.775558	0.357935	2.167	0.03263	*
director_nameDavid Fincher	-0.07068	0.45463	-0.155	0.87677	
director_nameJoel Schumacher	1.043301	0.529577	1.97	0.0516	.
director_nameJohn McTiernan	1.066037	0.511246	2.085	0.0396	*
director_nameMartin Scorsese	-0.169529	0.515933	-0.329	0.74315	
director_namePaul W.S. Anderson	0.143406	0.466768	0.307	0.75931	
director_nameRenny Harlin	0.653876	0.464068	1.409	0.16193	
director_nameRichard Donner	1.396116	0.624239	2.237	0.02754	*
director_nameRob Cohen	1.329123	0.485584	2.737	0.00734	**
director_nameRon Howard	0.546761	0.406754	1.344	0.18192	
director_nameShawn Levy	0.519174	0.506346	1.025	0.30768	
director_nameStephen Frears	0.616334	0.544897	1.131	0.26072	
director_nameSteven Spielberg	1.009328	0.365303	2.763	0.00682	**
director_nameTim Burton	0.589871	0.373339	1.58	0.11727	
director_nameTony Scott	0.64321	0.458547	1.403	0.1638	

Residual standard error: 0.7493 on 100 degrees of freedom

Multiple R-squared: 0.7569, Adjusted R-squared: 0.6913

F-statistic: 11.53 on 27 and 100 DF, p-value: < 2.2e-16

Table 15: Cross Validation and Out of Sample Comparisons for Quantitative Analysis

Statistic / Model	fit1	fit2
Adj-R2	0.6913	0.6908
Res-SD	0.75	0.75
rmsepred(leave-one-out)	0.84	0.832
rmsepred(train / holdout)	1.884	3.367

The RMSE for the training/holdout is quite high. It's possible that there could be some issues with the data (the variables that were omitted for the second model reduced the RMSE by a large factor, therefore perhaps those variables should not be included in the analysis), or that simply it is very hard to predict the performance of films, and high variability is to be expected when doing an analysis within this industry.

Brief Discussion (Director's Analysis):

In conclusion, we have found a good-fitting model when the model focuses on directors as categorical variables, and residual plots that show no sign of curvilinear form. There are some directors that are significantly correlated with box office gross (i.e. Steven Spielberg, Clint Eastwood).

The sample did not include foreign films – only domestic films made in the US were used for this subset. It is possible to include genre data or actor data, but the model suffered from overfitting if the categorical data is not grouped into fewer groups. Please see the Actor's Analysis for a more comprehensive regression with consideration of actors.

Actor's Analysis

Abstract/Summary

This purpose of this separate analysis is to look at the dataset with a focus on actors who have performed in more than 25 films in their filmography with respect to our dataset. The goal is to find a prediction equation for predicting box office gross with a focus on actors.

The response variable of adjusted gross was transformed to log(adjgross), otherwise the residual plots do not seem heteroscedastic .

The best prediction equation, after residual plots, variable selection and using criteria of adjusted R^2 and cross-validated root mean square prediction error is:

$$\begin{aligned}\log(\text{adjgross}) = {} & -2.0176274 - -0.0062244\text{duration} \\ & + 0.8828473(\text{lognum\_voted\_users}) - 0.0031701(\text{num\_critic\_for\_reviews}) \\ & + 0.5978969(\text{logbudget}) + 0.1475909(\text{lognum\_user\_for\_reviews}) + 0.0233070(\text{imdb\_score}) \\ & + 0.6570164(\text{Denzel Washington}) + 0.7746052(\text{Harrison Ford}) + 0.4390425(\text{J.K. Simmons}) - \\ & 0.5567939(\text{Jason Statham}) - 0.0978200(\text{Johnny Depp}) + 0.0513779(\text{Liam Neeson}) + \\ & 0.1420526(\text{Matt Damon}) - 0.0958409(\text{Nicolas Cage}) - 0.0127797(\text{Robert De Niro}) + \\ & 0.1114163(\text{Robert Downey Jr.}) + 0.3149969(\text{Robin Williams}) - \\ & 0.1972292(\text{content\_ratingPG}) - 0.4430192(\text{content\_ratingPG-13}) - \\ & 0.8910856(\text{content\_ratingR}) + 1.3003917\ (\text{content\_ratingX})\end{aligned}$$

Actor categorical variables are not grouped together. Since an actor/actress acts as a brand, and this analysis focuses on the effect of a single main actor on the gross grouping actors/actresses them together would not provide any useful analysis. Significant categorical variables of actors include Denzel Washington, Harrison Ford and Jason Statham.

Denzel Washington has a high coefficent of 0.657, Harrison Ford is 0.775, and Jason Statham is 0.557.

Description of Data

Data are collected by scraping both imdb resources and Box Office Mojo websites. Assumptions were made regarding the adjusted gross (estimated data from Box Office Mojo) which are outlined in APPENDIX III. Variables used in this analysis can be found in Table 16.

Table 16: Table of variables that might explain movie box office ross for actor's analysis

Variables	Explanation or unit
adjgross	Gross US domestic box office, adjusted for inflation
actor_1_facebook_likes	Number of Facebook likes from the lead actor
duration	Duration of the film (in minutes)
num_voted_users	Number of users who have voted for film on imdb
cast_total_facebook_likes	Number of Facebook likes received by the cast
Budget	Budget for the film
num_critic_for_review	Number of uses that have reviewed the film
movie_facebook_likes content_rating actor_name	Number of Facebook likes received by the film Rating for the film (e.g. PG, R, NC-17, etc.) Actors who has appeared in more than 25 films

This particular data set subsets the original data set such that only actors that have appeared in more than 25 films are considered. This reduces the dataset from over 5000 observations to 356.

Actors are not combined into fewer categorical variables because we try to focus on the effect of a single actor so their contributions to the gross are considered singular.

Data Analysis and Results

Summary statistics are provided are given in Tables 17, 18 and 19. Plots that justify the use of transforms for explanatory variables are given in Figure 3 and remain unchanged through this subset analysis.

Sample Correlations

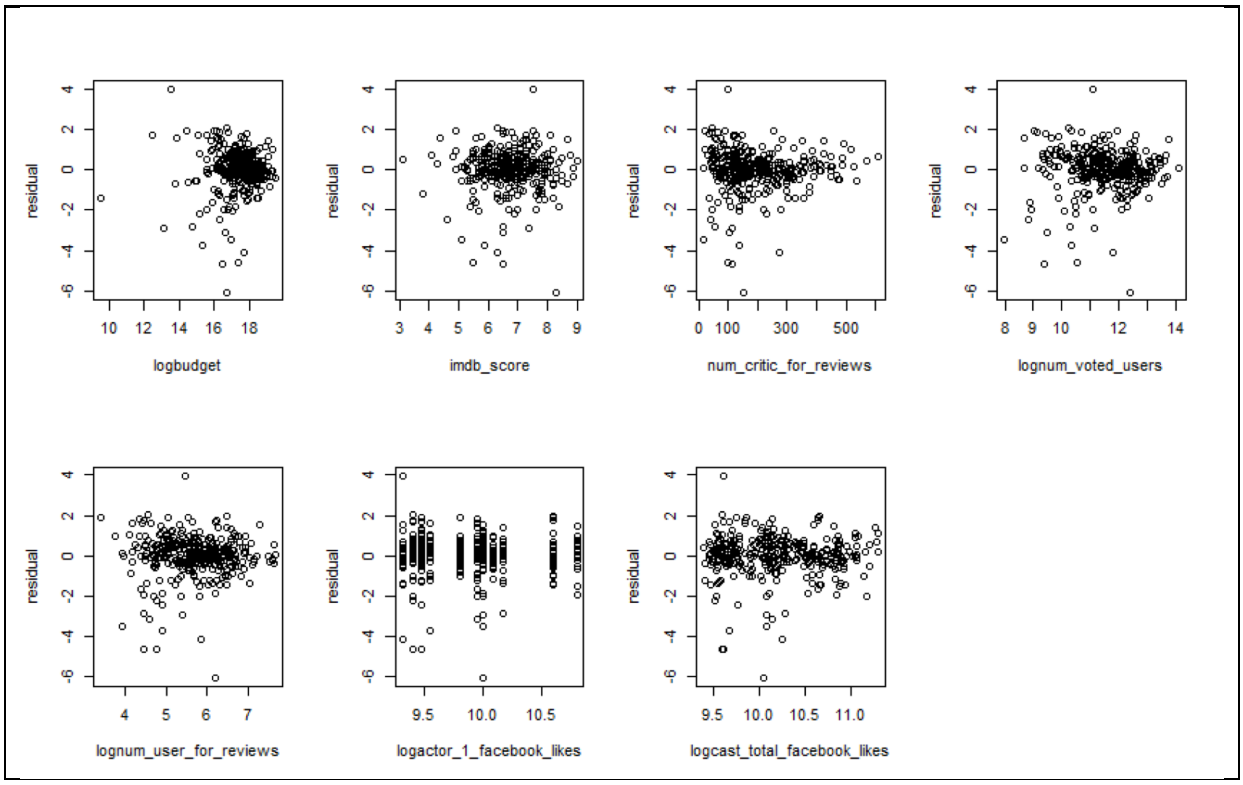
The highest correlation with the log(adjgross) appears to be the log(num_voted_users) with 0.609, log(budget) with 0.540 and log(num_user_for_reviews) with 0.517. Please see Appendix IV for the full sample correlation matrix.

We fit a multiple regression model (Table 20) with the explanatory variables given above. The adj-R^2 of this model is 0.7412 with a Residual Standard Error of 0.6683.

Residual Plots

Sample residual plots are given in Figure 5. The residuals do not suggest any curvilinear form, therefore quadratic terms or interaction terms were not added to the model.

Figure 5: Residual plots for some explanatory variables included in the actor's analysis



Backward Elimination and Forward Selection techniques were used to determine other good models. The best fit model suggested by both adj and cp value eliminated two explanatory variables: logactor_1_facebook_likes and logcast_total_facebook_likes. This model, as well as the full model, are compared via Cp statistic, adj-R2 Cross-validation root mean square error(leave-one-out) and Cross-validation root mean square error from a random holdout set (Table 21).

Table 17: Summary statistics of variables in director's regression analysis

	num_critic_for_reviews	duration	imdb_score	logadjgross	logactor_1_facebook_likes
Min:	16.0	77.0	3.100	9.709	9.306
1st Qu:	112.0	103.0	6.200	17.184	9.473
Median:	167.0	113.5	6.700	17.956	9.952

Mean:	196.6	117.3	6.722	17.686	9.903
3rd Qu:	266.2	126.0	7.300	18.594	10.106
Max:	608.0	289.0	9.000	21.047	10.800

Table 18: Summary statistics of variables included in director's regression analysis

	lognum_user_for_reviews	logcast_total_facebook_likes	lognum_voted_users	logbudget
Min:	3.401	9.384	7.979	9.547
1st Qu:	5.112	9.726	10.770	16.860
Median:	5.768	10.136	11.562	17.504
Mean:	5.710	10.188	11.503	17.390
3rd Qu:	6.260	10.596	12.299	18.081
Max:	7.694	11.300	14.097	19.519

Table 19: Frequency Table for actors involved >= 25 movies

Bruce Willis	Denzel Washington	Harrison Ford	J.K. Simmons	Jason Statham
29	30	25	31	25
Johnny Depp	Liam Neeson	Matt Damon	Nicolas Cage	Robert De Niro
39	26	28	30	42
Robert Downey Jr.	Robin Williams			
26	25			

Table 20: Multiple regression summary for actor's analysis

variable	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-2.01763	1.453236	-1.388	0.16595	
duration	-0.00622	0.002684	-2.319	0.02097	*
lognum_voted_users	0.882847	0.123172	7.168	4.93E-12	***
num_critic_for_reviews	-0.00317	0.000682	-4.648	4.84E-06	***
logbudget	0.597897	0.062404	9.581	< 2e-16	***
lognum_user_for_reviews	0.147591	0.134108	1.101	0.27189	
imdb_score	0.023307	0.096751	0.241	0.80978	
actor_1_nameDenzel Washington	0.657016	0.267544	2.456	0.01457	*
actor_1_nameHarrison Ford	0.774605	0.280838	2.758	0.00613	**
actor_1_nameJ.K. Simmons	0.439043	0.265523	1.654	0.09917	.
actor_1_nameJason Statham	-0.55679	0.281165	-1.98	0.04849	*
actor_1_nameJohnny Depp	-0.09782	0.254302	-0.385	0.70073	
actor_1_nameLiam Neeson	0.051378	0.276059	0.186	0.85247	
actor_1_nameMatt Damon	0.142053	0.279337	0.509	0.61141	
actor_1_nameNicolas Cage	-0.09584	0.263541	-0.364	0.71634	
actor_1_nameRobert De Niro	-0.01278	0.24486	-0.052	0.95841	
actor_1_nameRobert Downey Jr.	0.111416	0.282508	0.394	0.69355	
actor_1_nameRobin Williams	0.314997	0.283612	1.111	0.26751	
content_ratingPG	-0.19723	0.739032	-0.267	0.78973	
content_ratingPG-13	-0.44302	0.736631	-0.601	0.54797	
content_ratingR	-0.89109	0.737744	-1.208	0.22796	
content_ratingX	1.300392	1.053237	1.235	0.21782	

Residual standard error: 0.9982 on 334 degrees of freedom

Multiple R-squared: 0.644, Adjusted R-squared: 0.6216

F-statistic: 28.77 on 21 and 334 DF, p-value: < 2.2e-16

Table 21: Cross Validation and Out of Sample Comparisons for Actor's Analysis

Statistic \ Model	Full Model	Best Model
Adj-R2	0.556	0.558
Res-SD	1.082	1.079
rmsepred(leave-one-out)	1.103	1.097
rmsepred(train / holdout 1)	1.089	1.051
rmsepred(train / holdout 2)	1.082	1.214
rmsepred(train / holdout 3)	1.073	1.055

Brief Discussion (Actor's Analysis):

The model we found for actors is fine, there is no sign of curvilinear form. The adjusted R2 for the best model we choose is about 0.56, which is not very high, however the RMSE for train/holdout is really high, indicating that it may be hard to predict the gross of films using the data given and high variability is to be expected when doing an analysis within the film industry. One thing to be noticed, when fitting the best model selected by regsubsets, most of the variables we kept are not related to actors and only 3 of the actors shows statistical significance for the response variable.

Conclusion:

Predicting movie box office success by gross revenue is extremely difficult since the monetary success of a movie depends on a great deal of factors that cannot be easily predicted upon. In this investigation models were found could fit the data reasonably well, but were not completely accurate because movie performance varies widely.

Contributions

Team Formation:

Initially, our team formed when Kelly, Sam and Luke decided to work on a project together about trying to predict some property related to a data set on movies. Then, after the deadline for project proposals had passed, Richard had submitted a proposal for the same data set that was selected by Kelly, Sam and Luke. Dr. Joe suggested that Richard join our team and thus our team of four formed.

Authorship:

The names of the authors are listed in alphabetical order by surname.

Teamwork and contributions:

Our team equally contributed to the work done for this project. Together our team decided on the topic: that we wanted to study the effect of variables in our data set on predicting the gross revenue of movies. Initially, all of our team members helped to sort out the data and make it useful since some of the columns of the data set were difficulty to work with. Once our data was in a form that could be more easily worked with, Kelly performed the regression analysis including the directors, Richard performed the regression analysis including the actors/actresses, Sam performed the regression analysis for only the numerical explanatory of the data set and Luke created the charts and tables for the summary statistics. All team members contributed to writing the report and creating it in its final form. As well, all team members contributed to the R code used in this project. Luke and Kelly were co-team leaders that ensured the group stayed on track during the course of the project. As well, during the course of the project Luke helped bring to the attention of the other team members that we needed to consider adjusting the currencies used in this analysis for inflation. All group members were involved with planning group meetings and booking rooms to meet in to work on this project.

APPENDIX I - Director's Analysis

R-Script:

```
#setwd("./Movie_Project")
install.packages("leaps")
library(leaps)

dat = read.csv("data/moviedata_sortedby_year_split_genre.csv", stringsAsFactors =
FALSE)

# omit bad data with na.omit
options(na.action=na.omit)
options(max.print=99999)


# Adjust for inflation
year <- dat$title_year
Len  <- length(year)

for (i in 1:Len) {
  if ( dat$title_year[i] >= 1915 & dat$title_year[i] <= 1925) {
    dat$adjgross[i] <- dat$gross[i] * 40
  } else if (dat$title_year[i] > 1925 & dat$title_year[i] <= 1930) {
    dat$adjgross[i] <- dat$gross[i] * 35
  } else if (dat$title_year[i] > 1930 & dat$title_year[i] <= 1935) {
    dat$adjgross[i] <- dat$gross[i] * 30
  } else if (dat$title_year[i] > 1935 & dat$title_year[i] <= 1940) {
    dat$adjgross[i] <- dat$gross[i] * 25
  } else if (dat$title_year[i] > 1940 & dat$title_year[i] <= 1945) {
    dat$adjgross[i] <- dat$gross[i] * 21.6
  } else if (dat$title_year[i] > 1945 & dat$title_year[i] <= 1950) {
    dat$adjgross[i] <- dat$gross[i] * 14.5
  } else if (dat$title_year[i] > 1950 & dat$title_year[i] <= 1955) {
    dat$adjgross[i] <- dat$gross[i] * 11.5
  } else if (dat$title_year[i] > 1955 & dat$title_year[i] <= 1960) {
    dat$adjgross[i] <- dat$gross[i] * 10
  } else if (dat$title_year[i] > 1960 & dat$title_year[i] <= 1965) {
    dat$adjgross[i] <- dat$gross[i] * 7.5
  } else if (dat$title_year[i] > 1965 & dat$title_year[i] <= 1970) {
    dat$adjgross[i] <- dat$gross[i] * 5
  } else if (dat$title_year[i] > 1970 & dat$title_year[i] <= 1975) {
    dat$adjgross[i] <- dat$gross[i] * 4
  } else if (dat$title_year[i] > 1975 & dat$title_year[i] <= 1980) {
    dat$adjgross[i] <- dat$gross[i] * 3
  } else if (dat$title_year[i] > 1980 & dat$title_year[i] <= 1985) {
    dat$adjgross[i] <- dat$gross[i] * 2.5
  } else if (dat$title_year[i] > 1985 & dat$title_year[i] <= 1990) {
    dat$adjgross[i] <- dat$gross[i] * 2
  } else if (dat$title_year[i] > 1990 & dat$title_year[i] <= 1995) {
    dat$adjgross[i] <- dat$gross[i] * 1.8
  } else if (dat$title_year[i] > 1995 & dat$title_year[i] <= 2000) {
    dat$adjgross[i] <- dat$gross[i] * 1.5
  } else if (dat$title_year[i] > 2000 & dat$title_year[i] <= 2005) {
    dat$adjgross[i] <- dat$gross[i] * 1.3
  } else if (dat$title_year[i] > 2005 & dat$title_year[i] <= 2010) {
    dat$adjgross[i] <- dat$gross[i] * 1.1
  } else {
    dat$adjgross[i] <- dat$gross[i]
```

```
  }
}

# Use these lines to filter out by country and language
dat<-dat[(dat$country=="USA"| dat$country=="UK" | dat$country=="Canada" |
            dat$country=="New Zealand" | dat$country=="Australia" |
dat$country=="Ireland" |
            (dat$country=="Germany" & dat$language=="English") |
(dat$country=="France" & dat$language=="English") |
            (dat$country=="Italy" & dat$language=="English") |
(dat$country=="Spain" & dat$language=="English")),]


# get rid of categorical variables and gross for this analysis
drops <- c("gross", "color", "director_name", "actor_2_name", "genres",
"primary_genre", "actor_1_name", "movie_title", "actor_3_name", "plot_keywords",
"movie_imdb_link", "language", "country", "content_rating", "facenumber_in_poster",
"title_year", "aspect_ratio")
dat <- dat[ , !(names(dat) %in% drops)]

# creating regression
dat$adjgross <- log(dat$adjgross/10000000)                          # scale
adjusted gross by 10 million dollars and log
dat$num_critic_for_reviews <- log(dat$num_critic_for_reviews)       # log number of
critic reviews
dat$director_facebook_likes <- log(dat$director_facebook_likes)     # log nunber of
director likes
dat$actor_3_facebook_likes <- log(dat$actor_3_facebook_likes)       # log number of
3rd top billing actor likes
dat$actor_1_facebook_likes <- log(dat$actor_1_facebook_likes)       # log number of
1st top billing actor likes
dat$num_voted_users <- log(dat$num_voted_users)                     # log number of
user votes
dat$cast_total_facebook_likes <- log(dat$cast_total_facebook_likes) # log number of
total cast likes
dat$num_user_for_reviews <- log(dat$num_user_for_reviews)           # log number of
user reviews
dat$budget <- log(dat$budget)                                       # log budget
dat$actor_2_facebook_likes <- log(dat$actor_2_facebook_likes)       # log number of
2nd top billing actor likes
dat$movie_facebook_likes <- log(dat$movie_facebook_likes)           # log number of
facebook likes

dat <- do.call(data.frame,lapply(dat, function(x) replace(x, is.infinite(x),NA)))
dat <- na.omit(dat)

reg_quant <- lm(adjgross ~., data = dat, na.action = na.omit)
s_quant <- summary(reg_quant)

# check which model works best with forward selection  (best fit is 8|10 variables)
s.fwd <- regsubsets(adjgross ~., data=dat, method="forward", nvmax = 20)
ss.fwd <- summary(s.fwd)
which.min(ss.fwd$cp)     # 8 variables for best model
which.max(ss.fwd$adjr2)  # 10 variables for best model

# check which model works best with backward selection   (best fit is 8|10
variables)
s.bwd <- regsubsets(adjgross ~., data=dat, method="backward", nvmax = 20)
ss.bwd <- summary(s.bwd)
which.min(ss.bwd$cp)     # 8 variables for best model
which.max(ss.bwd$adjr2)  # 10 variables for best model

# the 12 explanatory variables to make the best model as identified by
forward/backward selection
keep <- c("adjgross", "num_critic_for_reviews", "director_facebook_likes",
"actor_1_facebook_likes", "num_voted_users", "cast_total_facebook_likes",
"num_user_for_reviews", "budget", "imdb_score")
```

```
# generate a new regression for the best model
datBest <- dat[keep]
regBest <- lm(adjgross ~., data = datBest)
sBest <- summary(regBest)

ls.cvrmse <- function(ls.out)
  # Compute the leave-one-out cross-validated root mean squared error of
prediction.
  # Handles missing values.
  # ls.out is a fitted regression model from lsfit or lm.
  # (c) Copyright William J. Welch 1997
{
  res.cv <- ls.out$residuals / (1.0 - ls.diag(ls.out)$hat)
  # Identify NA's and remove them.
  is.na.res <- is.na(res.cv)
  res.cv <- res.cv[!is.na.res]
  cvrmse <- sqrt(sum(res.cv^2) / length(res.cv))
  return(cvrmse)
}

n <- nrow(datBest)
set.seed(1)
# here select half of the data randomly.
index.subset1 <- sort(sample(1:n, round(n/2), replace = FALSE))

# Initially subset1 is the training set and subset2 is the hold-out set
quant.subset1 <- dat[index.subset1,]
quant.subset2 <- dat[-index.subset1,]

quantFull.subset1 <- lm(adjgross~., data = quant.subset1)

# Make predictions at the hold-out data set.
quantFull.pred1 <- predict(quantFull.subset1, quant.subset2)
quantFull.err1 <- sqrt(sum((quant.subset2$adjgross -
quantFull.pred1)^2)/length(quantFull.pred1))

# Compare to the selected model.
quantBest.subset1 <- lm(adjgross~., data = datBest[index.subset1,])
quantBest.pred1 <- predict(quantBest.subset1, datBest[-index.subset1,])
quantBest.err1 <- sqrt(sum((datBest[-index.subset1,]$adjgross -
quantBest.pred1)^2)/length(quantBest.pred1))


# ~~~~~~~~~~~~~~~~~~~~~ SUMMARY OF ANALYSIS ~~~~~~~~~~~~~~~~~~~~~ #

sBest                 # Summary statistics of the best model

sBest$adj.r.squared   # adjusted r squared of the best model
s_quant$adj.r.squared # adjusted r squared of the full model

sBest$sigma           # residual SD of the best model
s_quant$sigma         # residual SD of the full model

ls.cvrmse(regBest)    # leave-one-out cross validation of the best model
ls.cvrmse(reg_quant)  # leave-one-out cross validation of the full model

quantBest.err1        # rmse prediction of the training/holdout sets of the best
model
quantFull.err1        # rmse prediction of the training/holdout sets of the full
model


# ~~~~~~~~~~~~~~~~~~~~~ DIRECTOR ANALYSIS ~~~~~~~~~~~~~~~~~~~~~ #

data = read.csv("moviedata_sortedby_year.csv", colClasses=c("genres"="character"))
# add adjgross to data
```

```
for (i in 1:Len) {
    if ( data$title_year[i] >= 1915 & data$title_year[i] <= 1925) {
        data$adjgross[i] <- data$gross[i] * 40
    } else if (data$title_year[i] > 1925 & data$title_year[i] <= 1930) {
        data$adjgross[i] <- data$gross[i] * 35
    } else if (data$title_year[i] > 1930 & data$title_year[i] <= 1935) {
        data$adjgross[i] <- data$gross[i] * 30
    } else if (data$title_year[i] > 1935 & data$title_year[i] <= 1940) {
        data$adjgross[i] <- data$gross[i] * 25
    } else if (data$title_year[i] > 1940 & data$title_year[i] <= 1945) {
        data$adjgross[i] <- data$gross[i] * 21.6
    } else if (data$title_year[i] > 1945 & data$title_year[i] <= 1950) {
        data$adjgross[i] <- data$gross[i] * 14.5
    } else if (data$title_year[i] > 1950 & data$title_year[i] <= 1955) {
        data$adjgross[i] <- data$gross[i] * 11.5
    } else if (data$title_year[i] > 1955 & data$title_year[i] <= 1960) {
        data$adjgross[i] <- data$gross[i] * 10
    } else if (data$title_year[i] > 1960 & data$title_year[i] <= 1965) {
        data$adjgross[i] <- data$gross[i] * 7.5
    } else if (data$title_year[i] > 1965 & data$title_year[i] <= 1970) {
        data$adjgross[i] <- data$gross[i] * 5
    } else if (data$title_year[i] > 1970 & data$title_year[i] <= 1975) {
        data$adjgross[i] <- data$gross[i] * 4
    } else if (data$title_year[i] > 1975 & data$title_year[i] <= 1980) {
        data$adjgross[i] <- data$gross[i] * 3
    } else if (data$title_year[i] > 1980 & data$title_year[i] <= 1985) {
        data$adjgross[i] <- data$gross[i] * 2.5
    } else if (data$title_year[i] > 1985 & data$title_year[i] <= 1990) {
        data$adjgross[i] <- data$gross[i] * 2
    } else if (data$title_year[i] > 1990 & data$title_year[i] <= 1995) {
        data$adjgross[i] <- data$gross[i] * 1.8
    } else if (data$title_year[i] > 1995 & data$title_year[i] <= 2000) {
        data$adjgross[i] <- data$gross[i] * 1.5
    } else if (data$title_year[i] > 2000 & data$title_year[i] <= 2005) {
        data$adjgross[i] <- data$gross[i] * 1.3
    } else if (data$title_year[i] > 2005 & data$title_year[i] <= 2010) {
        data$adjgross[i] <- data$gross[i] * 1.1
    } else {
        data$adjgross[i] <- data$gross[i]
    }
}

directors <- subset(data, table(data$director_name)[data$director_name] >= 10)

directors$lngross <- log(directors$gross)
directors$lnadjgross <- log(directors$adjgross)
directors$lndirector_facebook_likes <- log(directors$director_facebook_likes)
directors$lnactor_1_facebook_likes <- log(directors$actor_1_facebook_likes)
directors$lnnum_voted_users <- log(directors$num_voted_users)
directors$lncast_total_facebook_likes <- log(directors$cast_total_facebook_likes)
directors$lnbudget <- log(directors$budget)
directors$lnnum_users_for_review <- log(directors$num_user_for_reviews)
directors$lnmovie_facebook_likes <- log(directors$movie_facebook_likes)

# cleanup bad data
directors$lnbudget[is.infinite(directors$lnbudget)] <- NA
directors$lnadjgross[is.infinite(directors$adjgross)] <- NA
directors$lnmovie_facebook_likes[is.infinite(directors$lnmovie_facebook_likes)] <-
NA
directors$lngross[is.infinite(directors$lngross)] <- NA
directors$lnnum_voted_users[is.infinite(directors$lnnum_voted_users)] <- NA
directors$lncast_total_facebook_likes[is.infinite(directors$lncast_total_facebook_l
ikes)] <- NA
directors$lndirector_facebook_likes[is.infinite(directors$lndirector_facebook_likes
)] <- NA
directors$lncast_total_facebook_likes[is.infinite(directors$lncast_total_facebook_l
ikes)] <- NA
```

```
directors$lnactor_1_facebook_likes[is.infinite(directors$lnactor_1_facebook_likes)]
<- NA

directors$lnbudget[is.nan(directors$lnbudget)] <- NA
directors$lnadjgross[is.nan(directors$adjgross)] <- NA
directors$lnmovie_facebook_likes[is.nan(directors$lnmovie_facebook_likes)] <- NA
directors$lngross[is.nan(directors$lngross)] <- NA
directors$lnnum_voted_users[is.nan(directors$lnnum_voted_users)] <- NA
directors$lncast_total_facebook_likes[is.nan(directors$lncast_total_facebook_likes)
] <- NA
directors$lndirector_facebook_likes[is.nan(directors$lndirector_facebook_likes)] <-
NA
directors$lncast_total_facebook_likes[is.nan(directors$lncast_total_facebook_likes)
] <- NA
directors$lnactor_1_facebook_likes[is.nan(directors$lnactor_1_facebook_likes)] <-
NA

# create regression model
directors <- na.omit(directors)

fit <- lm(lnadjgross ~ lncast_total_facebook_likes + lnactor_1_facebook_likes +
lndirector_facebook_likes + lnnum_voted_users + lnnum_users_for_review + lnbudget +
imdb_score + content_rating + duration + director_name, data = directors)

# install leaps package and run backward/forward selection
install.packages("leaps")
library(leaps)

# remove categorical variables as not compatible with variable selection algorithm
fitbackward <- regsubsets(lnadjgross ~ lncast_total_facebook_likes +
lnactor_1_facebook_likes + lndirector_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score + duration, data = directors, method
= "backward")
fitforward <- regsubsets(lnadjgross ~ lncast_total_facebook_likes +
lnactor_1_facebook_likes + lndirector_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score + duration, data = directors, method
= "backward")

# this process can be automated below to get the appropriate statistics for
comparison
fitbackward$summary <- summary(fitbackward) # to find cp and adjr2
fitforward$summary <- summary(fitforward) # to find cp and adjr2
maxadjr2fitbackward <- which.max(fitbackward$summary$adjr2) #model with max adjr2
value
minfitforwardcp <- which.min(fitforward$summary$cp) #model with min cp value

# calculate leave-one-out CV error
ls.cvrmse <- function(ls.out)
# Compute the leave-one-out cross-validated root mean squared error of prediction.
# Handles missing values.
# ls.out is a fitted regression model from lsfit or lm.
# (c) Copyright William J. Welch 1997
{
    res.cv <- ls.out$residuals / (1.0 - ls.diag(ls.out)$hat)
    # Identify NA's and remove them.
    is.na.res <- is.na(res.cv)
    res.cv <- res.cv[!is.na.res]
    cvrmse <- sqrt(sum(res.cv^2) / length(res.cv))
    return(cvrmse)
}

# Compare the full model and best model found by regsubsets, categoricals removed
fit <- lm(lnadjgross ~ lncast_total_facebook_likes + lnactor_1_facebook_likes +
lndirector_facebook_likes + lnnum_voted_users + lnnum_users_for_review + lnbudget +
imdb_score + content_rating + duration + director_name, data = directors)
#full model
```

```
# remove director_facebook_likes and duration due to variable selection process,
categoricals removed
fit2 <- lm(lnadjgross ~ lnactor_1_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score + content_rating + director_name,
data = directors)# removed num_users_for_review, but usually you use your best
model against your full model# find best model using forward, backward or
exhaustive methods
summary(fit2)

# Compare the full model and best model found by regsubsets, categoricals removed
fitA <- lm(lnadjgross ~ lncast_total_facebook_likes + lnactor_1_facebook_likes +
lndirector_facebook_likes + lnnum_voted_users + lnnum_users_for_review + lnbudget +
imdb_score + duration, data = directors)
#full model

# remove director_facebook_likes and duration due to variable selection process,
categoricals removed
fit2A <- lm(lnadjgross ~ lnactor_1_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score, data = directors)# removed
num_users_for_review, but usually you use your best model against your full model#
find best model using forward, backward or exhaustive methods
summary(fit2)

# Calculate the leave-one-out CV RMSE for the full model, categoricals removed
fitA.cvrmse <- ls.cvrmse(fitA)

# Calculate the leave-one-out CV RMSE for the best model via regsubsets,
categoricals removed
fit2A.cvrmse <- ls.cvrmse(fit2A)

print(c(fitA.cvrmse, fit2A.cvrmse))
# The best model has smaller cvrmse
# fit2 has a smaller cvrmse which is surprising as it has a smaller adj-R2.  But
maybe it's due to the NaNs produced

# Two-fold CV, using the Lab8 Technique
n <- nrow(directors)
set.seed(1)
id.subset1 <- sort(sample(1:n, round(n/2), replace = FALSE))

# randomly select holdout data
directors.subset1 <- directors[id.subset1,]
directors.subset2 <- directors[-id.subset1,]
fit.subset1 <- lm(lnadjgross ~ lncast_total_facebook_likes +
lnactor_1_facebook_likes + lndirector_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score + content_rating + duration +
director_name, data = directors.subset1)

# calculate a prediction based on the holdout set, using the full model
fit.pred1 <- predict(fit.subset1, data = directors.subset2)
# calculate the error
fit.err1 <- sqrt(sum((directors.subset2$lnadjgross -
fit.pred1)^2)/length(fit.pred1))

# calculate a prediction based on the 2nd fit model
fit2.subset1 <- lm(lnadjgross ~ lnactor_1_facebook_likes + lnnum_voted_users +
lnnum_users_for_review + lnbudget + imdb_score + content_rating + director_name,
data = directors.subset1)
fit2.pred1 <- predict(fit2.subset1, directors.subset2)
fit2.err1 <- sqrt(sum((directors.subset2$lnadjgross -
fit2.pred1)^2)/length(fit2.pred1))
fit2.err1

# ~~~~~~~~~~~~~~~~~~~~~ SUMMARY OF ANALYSIS ~~~~~~~~~~~~~~~~~~~~~ #

fullModel <- summary(fit)
secondBest <- summary(fit2)
```

```
print(c(fitA.cvrmse, fit2A.cvrmse))

# calculate the error
fit.err1
fit2.err1

# ~~~~~~~~~~~~~~~~~~~~~ ACTOR'S ANALYSIS ~~~~~~~~~~~~~~~~~~~~~ #

#first clean up the data
library(readr)
dat <- read_csv("~/R/movie_metadata.csv/moviedata_sortedby_year.csv")
data<-dat[(dat$country=="USA"| dat$country=="UK" | dat$country=="Canada" |
             dat$country=="New Zealand" | dat$country=="Australia" |
dat$country=="Ireland" |
             (dat$country=="Germany" & dat$language=="English") |
(dat$country=="France" & dat$language=="English") |
             (dat$country=="Italy" & dat$language=="English") |
(dat$country=="Spain" & dat$language=="English")),]
setwd(".")    # <--- trying to avoid direct path as we all have different setups
#install.packages('plyr')
#install.packages("stringr", repos='http://cran.us.r-project.org')
library(stringr)
library(plyr)
#read data from csv file sorted by year, with primary genres selected

options(na.action=na.omit)
options(max.print=99999)
#produces frewuency table output for content rating as seen on page 110:
count(data,"content_rating")

#create a vector that includes genre in the form "Action|Comedy|etc." for use in a
function used to count frequency of genres

genrevector <- data$genres
countAction = 0; countAdventure = 0; countAnimation = 0; countBiography = 0;
countComedy = 0;
countCrime = 0; countDocumentary = 0; countDrama = 0; countFamily = 0;
countFantasy = 0; countFilmNoir = 0; countGameShow = 0; countWar=0;
countHistory = 0; countHorror = 0; countMusic = 0; countMusical = 0; countMystery =
0;
countRomance = 0; countSciFi = 0; countThriller = 0; countWestern = 0; countSport =
0; countNews = 0; countRealityTV = 0;

#code to count frequency of genres
for (i in genrevector ) {
  if (grepl(i, "Action"))
    countAction = countAction + 1;
  if (grepl(i, "Adventure"))
    countAdventure = countAdventure + 1;
  if (grepl(i, "Animation"))
    countAnimation = countAnimation + 1;
  if (grepl(i, "Biography"))
    countBiography = countBiography + 1;
  if (grepl(i, "Comedy"))
    countComedy = countComedy + 1;
  if (grepl(i, "Crime"))
    countCrime = countCrime + 1;
  if (grepl(i, "Documentary"))
    countDocumentary = countDocumentary + 1;
  if (grepl(i, "Drama"))
    countDrama = countDrama + 1;
  if (grepl(i, "Family"))
    countFamily = countFamily + 1;
  if (grepl(i, "Fantasy"))
    countFantasy = countFantasy + 1;
  if (grepl(i, "Film-Noir"))
```

```
    countFilmNoir = countFilmNoir + 1;
  if (grepl(i, "Game-Show"))
    countGameShow = countGameShow + 1;
  if (grepl(i, "History"))
    countHistory = countHistory + 1;
  if (grepl(i, "Horror"))
    countHorror = countHorror + 1;
  if (grepl(i, "Music"))
    countMusic = countMusic + 1;
  if (grepl(i, "Musical"))
    countMusical = countMusical + 1;
  if (grepl(i, "Mystery"))
    countMystery = countMystery + 1;
  if (grepl(i, "Romance"))
    countRomance = countRomance + 1;
  if (grepl(i, "Sci-Fi"))
    countSciFi = countSciFi + 1;
  if (grepl(i, "Thriller"))
    countThriller = countThriller + 1;
  if (grepl(i, "Western"))
    countWestern = countWestern + 1;
  if (grepl(i, "War"))
    countWar = countWar + 1;
  if (grepl(i, "Sport"))
    countSport = countSport + 1;
  if (grepl(i, "News"))
    countNews = countNews + 1;
  if (grepl(i, "Reality-TV"))
    countRealityTV = countRealityTV + 1;
}
countAction; countAdventure; countAnimation; countBiography; countComedy ;
countCrime ; countDocumentary ; countDrama ; countFamily ;
countFantasy ; countFilmNoir ; countGameShow ;
countHistory ; countHorror ; countMusic ; countMusical ; countMystery ;
countRomance ; countSciFi ; countThriller ; countWestern ; countWar; countSport;
countNews; countRealityTV;

# Create binary vector for each genre
# Make vectors for:
#   - Action, Adventure, Animation, Biography, Comedy, Crime, Drama, Family,
Fantasy, History, Horror, Music, Musical, Mystery, Romance, Sci-fi, Thriller, War

# Given a genre and a the genres of a movie, check if a movie contains it, if it
does return 1, 0 otherwise
hasGenre <- function(genre, movieGenres) {
  regExp <- paste("(^|\\|)", genre, "(\\||$)", sep = "")
  if(grepl(regExp, movieGenres)){
    return(1)
  } else {
    return(0)
  }
}

# Given a genre and a vector of the genres of various movies, produce a vector of
TRUE/FALSE for movies that contain/don't contain the given genre
moviesHaveGenre <- function(genre_, genreVector) {
  return(sapply(genreVector, hasGenre, genre = genre_, USE.NAMES = FALSE))
}

commonGenres <- c("Action", "Adventure", "Animation", "Biography", "Comedy",
"Crime", "Drama", "Family", "Fantasy", "History", "Horror", "Music", "Musical",
"Mystery", "Romance", "Sci-Fi", "Thriller", "War")

# Given a genre, create vector of movies that have or do not have the genre (1|0),
and cbind to the dataframe

for (genre in commonGenres) {
```

```
  gVector <- moviesHaveGenre(genre, data$genres)
  data[genre] <- gVector
}


dat<-data.frame(data)
dat<-na.omit(dat)

year <- dat$title_year
Len  <- length(year)

for (i in 1:Len) {
  if ( dat$title_year[i] >= 1915 & dat$title_year[i] <= 1925) {
    dat$adjgross[i] <- dat$gross[i] * 40
  } else if (dat$title_year[i] > 1925 & dat$title_year[i] <= 1930) {
    dat$adjgross[i] <- dat$gross[i] * 35
  } else if (dat$title_year[i] > 1930 & dat$title_year[i] <= 1935) {
    dat$adjgross[i] <- dat$gross[i] * 30
  } else if (dat$title_year[i] > 1935 & dat$title_year[i] <= 1940) {
    dat$adjgross[i] <- dat$gross[i] * 25
  } else if (dat$title_year[i] > 1940 & dat$title_year[i] <= 1945) {
    dat$adjgross[i] <- dat$gross[i] * 21.6
  } else if (dat$title_year[i] > 1945 & dat$title_year[i] <= 1950) {
    dat$adjgross[i] <- dat$gross[i] * 14.5
  } else if (dat$title_year[i] > 1950 & dat$title_year[i] <= 1955) {
    dat$adjgross[i] <- dat$gross[i] * 11.5
  } else if (dat$title_year[i] > 1955 & dat$title_year[i] <= 1960) {
    dat$adjgross[i] <- dat$gross[i] * 10
  } else if (dat$title_year[i] > 1960 & dat$title_year[i] <= 1965) {
    dat$adjgross[i] <- dat$gross[i] * 7.5
  } else if (dat$title_year[i] > 1965 & dat$title_year[i] <= 1970) {
    dat$adjgross[i] <- dat$gross[i] * 5
  } else if (dat$title_year[i] > 1970 & dat$title_year[i] <= 1975) {
    dat$adjgross[i] <- dat$gross[i] * 4
  } else if (dat$title_year[i] > 1975 & dat$title_year[i] <= 1980) {
    dat$adjgross[i] <- dat$gross[i] * 3
  } else if (dat$title_year[i] > 1980 & dat$title_year[i] <= 1985) {
    dat$adjgross[i] <- dat$gross[i] * 2.5
  } else if (dat$title_year[i] > 1985 & dat$title_year[i] <= 1990) {
    dat$adjgross[i] <- dat$gross[i] * 2
  } else if (dat$title_year[i] > 1990 & dat$title_year[i] <= 1995) {
    dat$adjgross[i] <- dat$gross[i] * 1.8
  } else if (dat$title_year[i] > 1995 & dat$title_year[i] <= 2000) {
    dat$adjgross[i] <- dat$gross[i] * 1.5
  } else if (dat$title_year[i] > 2000 & dat$title_year[i] <= 2005) {
    dat$adjgross[i] <- dat$gross[i] * 1.3
  } else if (dat$title_year[i] > 2005 & dat$title_year[i] <= 2010) {
    dat$adjgross[i] <- dat$gross[i] * 1.1
  } else {
    dat$adjgross[i] <- dat$gross[i]
  }
}

# data cleaned

movie_actor<-na.omit(dat)
#only picks movies data with main actor who shows up in more than 20 movies
ac1<-subset(movie_actor, table(actor_1_name)[actor_1_name] >= 25)
ac1<-na.omit(ac1)

#take log for big data
ac1$logadjgross<-log(ac1$adjgross)
ac1$logactor_1_facebook_likes<-log(ac1$actor_1_facebook_likes)
ac1$logcast_total_facebook_likes<-log(ac1$cast_total_facebook_likes)
ac1$lognum_voted_users<-log(ac1$num_voted_users)
ac1$logbudget<-log(ac1$budget)
ac1$lognum_user_for_reviews<-log(ac1$num_user_for_reviews)
```

```
ac1$logdirector_facebook_likes<-log(ac1$director_facebook_likes)

# fit full model with actor_name as catagory variables
fit <- lm(logadjgross ~logactor_1_facebook_likes+logcast_total_facebook_likes
          +lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_revi
ews+imdb_score+actor_1_name,data=ac1)
s1<-summary(fit)

par(mfrow=c(2,4))
plot(ac1$logbudget,s1$residuals,xlab="logbudget",ylab="residual")
plot(ac1$imdb_score,s1$residuals,xlab="imdb_score",ylab="residual")
plot(ac1$num_critic_for_reviews,s1$residuals,xlab="num_critic_for_reviews",ylab="re
sidual")
plot(ac1$lognum_voted_users,s1$residuals,xlab="lognum_voted_users",ylab="residual")
plot(ac1$lognum_user_for_reviews,s1$residuals,xlab="lognum_user_for_reviews",ylab="
residual")
plot(ac1$logactor_1_facebook_likes,s1$residuals,xlab="logactor_1_facebook_likes",yl
ab="residual")
plot(ac1$logcast_total_facebook_likes,s1$residuals,xlab="logcast_total_facebook_lik
es",ylab="residual")

#box plot
par(mfrow=c(1,1))
boxplot(ac1$logadjgross ~ ac1$actor_1_name, main="actor_1_name", xlab =
"actor_1_name", ylab = "ln(Gross Revenue, USD)")

# with movie type, but doesn't make much sense
fit2 <- lm(logadjgross ~num_critic_for_reviews+logcast_total_facebook_likes +
logactor_1_facebook_likes
           + lognum_voted_users + lognum_user_for_reviews + logbudget +
             imdb_score + content_rating + Action + Adventure + Animation +
             Biography + Comedy + Crime + Drama + Family + Fantasy + History +
Horror + Music +
             Musical + Mystery + Romance + Thriller + War, data = ac1)

s2<-summary(fit2)

#install.packages("leaps")
library(leaps)
#forward and backward selection for full model without catagory variable
fitbackward <- regsubsets(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core, data = ac1, method = "backward")
fitforward <- regsubsets(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core, data = ac1, method = "forward")

fitbackward$summary <- summary(fitbackward)
fitforward$summary <- summary(fitforward)

#check adj and cp to choose the best model, adj suggests that full model is the
best, but cp suggests model with four variable is better
#fit both models suggested by adj and cp value
which.max(fitbackward$summary$adjr2)
which.min(fitbackward$summary$cp)
which.max(fitforward$summary$adjr2)
which.min(fitforward$summary$cp)

fitbest<-lm(logadjgross ~duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core, data = ac1)
fitfull <- lm(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core, data = ac1)
```

```
#summary
summaryfull<-summary(fitfull)
summarybest<-summary(fitbest)


ls.cvrmse <- function(ls.out)
{
    res.cv <- ls.out$residuals / (1.0 - ls.diag(ls.out)$hat)
    # Identify NA's and remove them.
    is.na.res <- is.na(res.cv)
    res.cv <- res.cv[!is.na.res]
    cvrmse <- sqrt(sum(res.cv^2) / length(res.cv))
    return(cvrmse)
}
cat("\nCross-validation root mean square error using ls.cvrmse function\n")
cvrmsebest<-ls.cvrmse(fitbest); cvrmsefull<-ls.cvrmse(fitfull)
cat(cvrmsebest,cvrmsefull,"\n")


#For each of the 2 models, we do 3 random holdout prediction to get the
#Cross-validation root mean square error with holdout,here we add
#back the catagory variables (actor_1_name)

# for the full model
n <- nrow(ac1)
set.seed(1)
id.subset1 <- sort(sample(1:n, round(n/2), replace = FALSE))

# randomly select holdout data
ac1.subset1 <- ac1[id.subset1,]
ac1.subset2 <- ac1[-id.subset1,]
#here we add back the category variables
fit.subseta <- lm(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset1)

# calculate a prediction based on the holdout set, using the full model
fit.preda <- predict(fit.subseta, ac1.subset2)
# calculate the error
fit.erra <- sqrt(sum((ac1.subset2$logadjgross - fit.preda)^2)/length(fit.preda))
fit.erra

#repeat for 3 times
id.subset3 <- sort(sample(1:n, round(n/2), replace = FALSE))
ac1.subset3 <- ac1[id.subset3,]
ac1.subset4 <- ac1[-id.subset3,]
fit.subsetb <- lm(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset3)

fit.predb <- predict(fit.subsetb, ac1.subset4)
# calculate the error
fit.errb <- sqrt(sum((ac1.subset4$logadjgross - fit.predb)^2)/length(fit.predb))
fit.errb

#third time
id.subset5 <- sort(sample(1:n, round(n/2), replace = FALSE))
ac1.subset5 <- ac1[id.subset5,]
ac1.subset6 <- ac1[-id.subset5,]
fit.subsetc <- lm(logadjgross
~logactor_1_facebook_likes+logcast_total_facebook_likes+duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset5)

fit.predc <- predict(fit.subsetc, ac1.subset6)
```

```
fit.errc <- sqrt(sum((ac1.subset6$logadjgross - fit.predc)^2)/length(fit.predc))
fit.errc

# do this again for 6 variable model and the category variables
id.subset1 <- sort(sample(1:n, round(n/2), replace = FALSE))

# randomly select holdout data
ac1.subset1 <- ac1[id.subset1,]
ac1.subset2 <- ac1[-id.subset1,]
#here we add back the category variables
fit.subseta <- lm(logadjgross ~duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset1)

# calculate a prediction based on the holdout set, using the full model
fit.preda <- predict(fit.subseta, ac1.subset2)
# calculate the error
fit.errd <- sqrt(sum((ac1.subset2$logadjgross - fit.preda)^2)/length(fit.preda))
fit.errd

#repeat for 3 times
id.subset3 <- sort(sample(1:n, round(n/2), replace = FALSE))
ac1.subset3 <- ac1[id.subset3,]
ac1.subset4 <- ac1[-id.subset3,]
fit.subsetb <- lm(logadjgross ~duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset3)

fit.predb <- predict(fit.subsetb, ac1.subset4)
# calculate the error
fit.erre <- sqrt(sum((ac1.subset4$logadjgross - fit.predb)^2)/length(fit.predb))
fit.erre

#third time
id.subset5 <- sort(sample(1:n, round(n/2), replace = FALSE))
ac1.subset5 <- ac1[id.subset5,]
ac1.subset6 <- ac1[-id.subset5,]
fit.subsetc <- lm(logadjgross ~duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name, data = ac1.subset5)

fit.predc <- predict(fit.subsetc, ac1.subset6)
fit.errf <- sqrt(sum((ac1.subset6$logadjgross - fit.predc)^2)/length(fit.predc))
fit.errf



#get the residual SD and adjusted R2
SD1<-summaryfull$sigma
SD2<-summarybest$sigma
adj1<-summaryfull$adj.r.squared
adj2<-summarybest$adj.r.squared



# build table
trial <-
matrix(c(adj1,SD1,cvrmsefull,fit.erra,fit.errb,fit.errc,adj2,SD2,cvrmsebest,fit.err
d,fit.erre,fit.errf), ncol=2)
colnames(trial) <- c("full model", "fit by cp")
rownames(trial) <- c("adjusted R2","residual SD","rmsepred(leave-one-
out)","rmsepred(train/holdout) 1",
"rmsepred(train/holdout) 2","rmsepred(train/holdout) 3")
trial.table <- as.table(trial)
trial.table

# Model with 4 explanatory variables is better based on leave-one-out
```

```
fitbest2<-lm(logadjgross ~duration
+lognum_voted_users+num_critic_for_reviews+logbudget+lognum_user_for_reviews+imdb_s
core+actor_1_name+content_rating, data = ac1)
summary(fitbest2)
ac2<-ac1[,c(3,4,26,48,49,50,51,52,53)]
summary(ac2)

# ~~~~~~~~~~~~~~~~~~~~~ SUMMARY OF ANALYSIS ~~~~~~~~~~~~~~~~~~~~~ #

summaryfull #summary(fitfull)
summarybest #summary(fitbest)
SD1 #summaryfull$sigma
SD2 #summarybest$sigma
adj1 #summaryfull$adj.r.squared
adj2 #summarybest$adj.r.squared

cvrmsebest

# calculate the error, training holdout
fit.erra
fit.errb

fit.errc
fit.errd

fit.erre
fit.errf
```

APPENDIX II - Sample Corr Matrix, Director's Analysis

	num_critic_f	duration	director_face	actor_3_face	actor_1_face	gross	num_voted_	cast_total_fa	facenumber_	num_user_fc	budget	title_year	actor_2_face	imdb_score	aspect_ratio	movie_faceb	adjgross	lngross	lnadjgross	lndirector_fa	lnactor_1_fa	lnnum_votec	lncast_total_	lnbudget	lnnum_users	lnmovie_facebook_likes
num_critic_f	1	0.3369725	0.5219378	0.1605909	0.2514274	0.3558677	0.4084187	0.2817229	-0.157897	0.4136619	0.4303116	0.5503261	0.2862279	0.3437031	0.1466021	0.7632567	0.3558677	0.4180714	0.4180714	0.4849606	0.2472263	0.5635719	0.2614069	0.362075	0.5604013	0.74951
duration	0.3369725	1	0.5348259	0.0452583	0.2088621	0.2411822	0.5764363	0.1846627	-0.103138	0.5159781	0.1913013	-0.068128	0.1555955	0.5640545	0.1654805	0.4173318	0.2411822	0.3277384	0.3277384	0.4753934	0.2516909	0.5197842	0.2493283	0.2795738	0.5270072	0.3691096
director_face	0.5219378	0.5348259	1	0.021013	0.1516626	0.2655802	0.5148263	0.1430016	-0.224949	0.4426446	0.1064395	0.0164267	0.164034	0.6386277	-0.010216	0.4045856	0.2655802	0.2900038	0.2900038	0.9525933	0.196895	0.5070429	0.1823767	0.0856915	0.4896327	0.4467196
actor_3_face	0.1605909	0.0452583	0.021013	1	0.4202258	0.2286248	0.1087915	0.6986845	0.2335005	0.1690198	0.2926449	0.1031983	0.6801043	-0.047531	-0.076383	0.1134819	0.2286248	0.0761927	0.0761927	0.0409872	0.3195473	0.1227549	0.4418879	0.2216486	0.1455957	0.092096
actor_1_face	0.2514274	0.2088621	0.1516627	0.4202258	1	0.2880814	0.251427	0.9145232	0.081473	0.1719246	0.4256565	0.1538439	0.5540434	0.1431531	0.0004408	0.177477	0.2880814	0.3081679	0.3081679	0.14408	0.8640041	0.3549504	0.8646538	0.3705623	0.2787477	0.2500774
gross	0.3558677	0.2411822	0.2655802	0.2286248	0.2880814	1	0.439556	0.3079895	-0.119978	0.4484577	0.3634289	-0.159334	0.25261	0.3122876	-0.081636	0.2837586	1	0.7530192	0.7530192	0.2689027	0.202554	0.5576022	0.2332994	0.2576015	0.529476	0.3725986
num_voted_	0.4084187	0.5764363	0.5148263	0.1087915	0.251427	0.439556	1	0.2526832	-0.10992	0.8240199	0.118675	-0.137316	0.2387662	0.6994907	0.0438347	0.4722358	0.439556	0.4818652	0.4818652	0.4430742	0.2987465	0.8172555	0.3065549	0.1601436	0.7744161	0.5514798
cast_total_fa	0.2817229	0.1846627	0.1430016	0.6986845	0.9145232	0.3079895	0.2526832	1	0.1166419	0.2186987	0.4334305	0.1562698	0.8126951	0.1235033	0.0017834	0.180248	0.3079895	0.2720759	0.2720759	0.1416559	0.7879688	0.3422478	0.8582217	0.362265	0.3012818	0.2439041
facenumber_	-0.157897	-0.103138	-0.224949	0.2335005	0.081473	-0.119978	-0.10992	0.1166419	1	-0.121369	0.0367227	0.0070373	0.0530212	-0.133573	0.0581085	-0.047534	-0.119978	-0.109957	-0.109957	-0.255513	0.0141001	-0.152365	0.0547777	0.0396891	-0.219776	-0.098392
num_user_fc	0.4136619	0.5159781	0.4426446	0.1690198	0.1719246	0.4484577	0.8240199	0.2186987	-0.121369	1	0.2547833	-0.058395	0.2434347	0.5137547	0.0625745	0.3542489	0.4484577	0.4642227	0.4642227	0.3928079	0.2354462	0.7135415	0.2587622	0.2685503	0.8557799	0.4595964
budget	0.4303116	0.1913013	0.1064395	0.2926449	0.4256565	0.3634289	0.118675	0.4334305	0.0367227	0.2547833	1	0.3993871	0.324407	-0.100682	0.1435388	0.1711476	0.3634289	0.3814137	0.3814137	0.0777536	0.3183493	0.2229844	0.3449185	0.8795585	0.2798578	0.2174498
title_year	0.5503261	-0.068128	0.0164267	0.1031983	0.1538439	-0.159334	-0.137316	0.1562698	0.0070373	-0.058395	0.3993871	1	0.1099293	-0.25798	0.2389375	0.378426	-0.159334	-0.012118	-0.012118	-0.012324	0.123434	0.0022429	0.1264758	0.411584	0.018724	0.3496114
actor_2_face	0.2862278	0.1555955	0.164034	0.6801043	0.5540434	0.25261	0.2387662	0.8126951	0.0530212	0.2434347	0.324407	0.1099293	1	0.1447941	0.0385918	0.1397679	0.25261	0.1956877	0.1956877	0.1638855	0.4906042	0.2883398	0.6237144	0.2457021	0.300875	0.2142503
imdb_score	0.3437031	0.5640545	0.6386277	-0.047531	0.1431531	0.3122876	0.6994907	0.1235033	-0.133573	0.5137547	-0.100682	-0.25798	0.1447941	1	-0.085484	0.3934401	0.3122876	0.3611975	0.3611975	0.634507	0.1973959	0.6955625	0.2087259	-0.063884	0.5582467	0.5412319
aspect_ratio	0.1466021	0.1654805	-0.010216	-0.076383	0.0004408	-0.081636	0.0438347	0.0017834	0.0581085	0.0625745	0.1435388	0.2389375	0.0385918	-0.085484	1	0.1046505	-0.081636	0.0713304	0.0713304	-0.113931	0.0377021	0.0730237	0.045999	0.2652683	0.104477	0.1132495
movie_faceb	0.7632567	0.4173318	0.4045856	0.1134819	0.177477	0.2837586	0.4722358	0.180248	-0.047534	0.3542489	0.1711476	0.378426	0.1397679	0.3934401	0.1046504	1	0.2837586	0.3558658	0.3558658	0.3562744	0.1738916	0.5079968	0.1869438	0.2041563	0.4566062	0.736873
adjgross	0.3558677	0.2411822	0.2655802	0.2286248	0.2880814	1	0.439556	0.3079895	-0.119978	0.4484577	0.3634289	-0.159334	0.25261	0.3122876	-0.081636	0.2837586	1	0.7530192	0.7530192	0.2689027	0.202554	0.5576022	0.2332994	0.2576015	0.529476	0.3725986
lngross	0.4180714	0.3277384	0.2900038	0.0761927	0.3081678	0.7530192	0.4818652	0.2720759	-0.109957	0.4642227	0.3814137	-0.012118	0.1956877	0.3611975	0.0713304	0.3558658	0.7530192	1	1	0.2721338	0.2429645	0.7512085	0.276473	0.4222671	0.6566744	0.5941608
lnadjgross	0.4180714	0.3277384	0.2900038	0.0761927	0.3081678	0.7530192	0.4818652	0.2720759	-0.109957	0.4642227	0.3814137	-0.012118	0.1956877	0.3611975	0.0713304	0.3558658	0.7530192	1	1	0.2721338	0.2429645	0.7512085	0.276473	0.4222671	0.6566744	0.5941608
lndirector_fa	0.4849606	0.4753934	0.9525933	0.0409872	0.14408	0.2689027	0.4430742	0.1416559	-0.255513	0.3928079	0.0777536	-0.012324	0.1638855	0.634507	-0.113931	0.3562744	0.2689027	0.2721338	0.2721338	1	0.1940549	0.4714429	0.1760538	0.0508027	0.4508376	0.4138964
lnactor_1_fa	0.2472263	0.2516909	0.196895	0.3195473	0.8640041	0.202554	0.2987465	0.7879688	0.0141001	0.2354462	0.3183493	0.123434	0.4906042	0.1973959	0.0377021	0.1738916	0.202554	0.2429645	0.2429645	0.1940549	1	0.366293	0.9644378	0.2932093	0.321659	0.2457373
lnnum_votec	0.5635719	0.5197842	0.5070429	0.1227549	0.3549504	0.5576022	0.8172555	0.3422478	-0.152365	0.7135415	0.2229844	0.0022429	0.2883398	0.6955625	0.0730237	0.5079969	0.5576022	0.7512085	0.7512085	0.4714429	0.366293	1	0.3901156	0.2912168	0.9006212	0.7695774
lncast_total_	0.2614069	0.2493283	0.1823767	0.4418879	0.8646538	0.2332994	0.3065549	0.8582217	0.0547777	0.2587622	0.3449185	0.1264758	0.6237144	0.2087259	0.045999	0.1869438	0.2332994	0.276473	0.276473	0.1760538	0.9644378	0.3901156	1	0.3191879	0.344398	0.2632386
lnbudget	0.362075	0.2795738	0.0856915	0.2216486	0.3705623	0.2576015	0.1601436	0.362265	0.0396891	0.2685503	0.8795585	0.411584	0.2457021	-0.063884	0.2652683	0.2041563	0.2576015	0.4222671	0.4222671	0.0508027	0.2932093	0.2912168	0.3191879	1	0.336354	0.2445683
lnnum_users	0.5604013	0.5270072	0.4896327	0.1455957	0.2787477	0.529476	0.7744161	0.3012818	-0.219776	0.8557799	0.2798578	0.018724	0.300875	0.5582467	0.104477	0.4566062	0.529476	0.6566744	0.6566744	0.4508376	0.321659	0.9006212	0.344398	0.336354	1	0.6718814
lnmovie_face	0.7495099	0.3691096	0.4467196	0.092096	0.2500774	0.3725986	0.5514798	0.2439041	-0.098392	0.4595964	0.2174498	0.3496114	0.2142503	0.5412319	0.1132495	0.736873	0.3725986	0.5941608	0.5941608	0.4138964	0.2457373	0.7695774	0.2632386	0.2445683	0.6718814	1

APPENDIX III – Inflation Rate Table

Inflation Rate (estimated, Box Office Mojo):

1940 - 1945: 21.6
1945 - 1950: 14.5
1950 - 1955: 11.5
1955 - 1960: 10
1960 - 1965: 7.5
1965 - 1970: 5
1970 - 1975: 4
1975 - 1980: 3
1980 - 1985: 2.5
1985 - 1990: 2
1990 - 1995: 1.8
1995 - 2000: 1.5
2000 - 2005: 1.3
2005 - 2010: 1.1

APPENDIX IV – Sample Correlation Matrix – Actor's Analysis

	num_critic_for_reviews	duration	imdb_score	logadjgross	logactor_1_facebook_likes	logcast_total_facebook_likes	lognum_voted_users	logbudget	lognum_user_for_reviews
_for_reviews	1	0.0868748	0.180724	0.3058305	-0.013211	0.0782889	0.5917467	0.3319611	0.5977337
duration	0.0868748	1	0.3533537	0.1684917	-0.120628	-0.054534	0.2714155	0.1986837	0.318658
imdb_score	0.180724	0.3533537	1	0.2983945	-0.010015	0.0160883	0.6165886	-0.151797	0.4292816
logadjgross	0.3058305	0.1684917	0.2983945	1	-0.01944	0.0342877	0.6085351	0.539068	0.5168911
ebook_likes	-0.013211	-0.120628	-0.010015	-0.01944	1	0.8667619	0.0117203	-0.053603	0.0037241
ebook_likes	0.0782889	-0.054534	0.0160883	0.0342877	0.8667619	1	0.0626535	0.0390377	0.0432845
voted_users	0.5917467	0.2714155	0.6165886	0.6085351	0.0117203	0.0626535	1	0.2574494	0.830361
logbudget	0.3319611	0.1986837	-0.151797	0.539068	-0.053603	0.0390377	0.2574494	1	0.2602615
_for_reviews	0.5977337	0.318658	0.4292816	0.5168912	0.0037241	0.0432845	0.830361	0.2602615	1