ECON 412 Project 2

Siyan Chen, Zeyi Pan, Minxuan Wang, Huafeng Zhang

UCLA Spring 2018

1 The Goal

For this project, we would like to construct a best fit model to predict for a binary response variable, i.e. in multiple regression part, we tried to extract important features which can be used as explanatory variables and fit a linear regression of IMDB score and these features; In the following parts, we switched the IMDB score into binary response variable and try to determine whether the IMDB score of a movie is distributed as an excellent-scored (above 7 points) or a normal-and-below-scored (below 7 points) one, with various predictors, which are used to explain the variation in the response variable. We are going to fit 5 different models to our dataset and recommend the best fit model based upon the estimations. The five models are multiple regression model, multiple logistic regression model, Ridge regression model, LASSO model, and Decision Tree model. At last, we would like to extract important features and to figure out which features are significant for determination of the IMDB score of a movie.

2 Data Acquisition

2.1 Data Description

The dataset used in our project is retrieved from Kaggle website. It contains 28 variables for 5043 movies, spanning 100 years in 66 countries. 2399 unique director names and thousands of cast names are recorded. The "imdb_score" which quantify the quality of the movie is used as the response variable, while other 27 variables are possible predictors. The numerical "imdb_score" is used in the multiple regression process. We turn it into a binary variable when conducting the further analysis. The original dataset has been replaced in Kaggle, here's the link for the original dataset from Dataworld:

`https://data.world/data-society/imdb-5000-movie-dataset`

2.2 Data Cleaning

Since the raw dataset is quite noisy, we clean the dataset before further analysis. First, we remove the duplicate observations. The genres of each observation have multiple records, we only keep the first one for analysis.

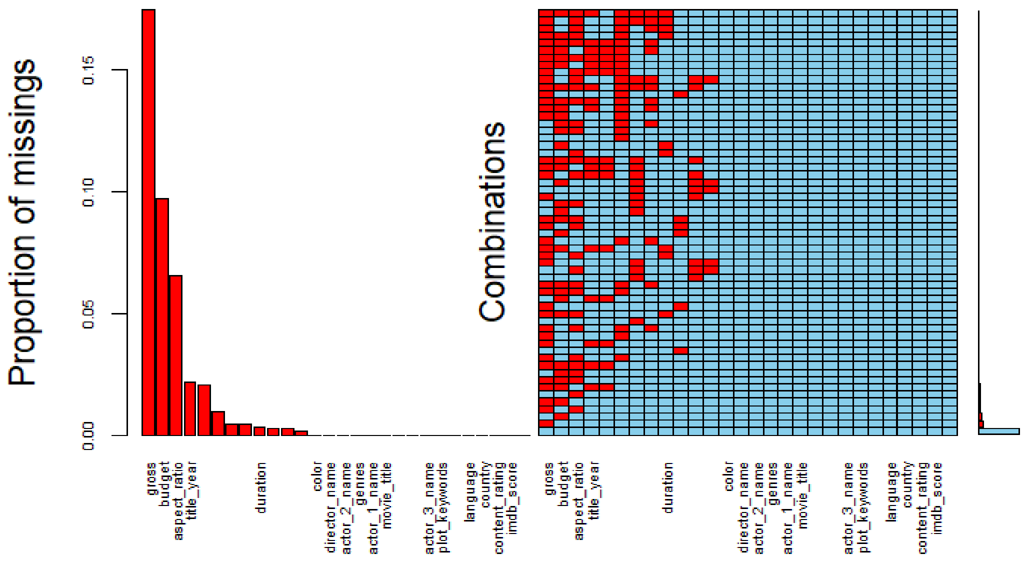
In our project, we use the primary class as the movie's genre. To be specific, If a movie falls into action, adventure and fantacy movie simultaneously, we think this movie is action movie. There are other researchers take all the genres into account.

We use the heatmap (Figure 1) to visualize the missing values in our dataset. From the heat map we can observe that the gross missing value is more than 15%. Especially, the variables "budget", "aspect_ratio", "title_year" contains rather large among of missing values. We need to consider this issue in the following analysis. In some predictors, certain values domain the observations. These values can be considered as consistent to simplify our analysis. Under this principle, we remove "color", "language" and "aspect_ratio".

In some predictors, certain values domain the observations. These values can be considered as consistent to simplify our analysis. Under this principle, we remove "color", "language" and "aspect_ratio".

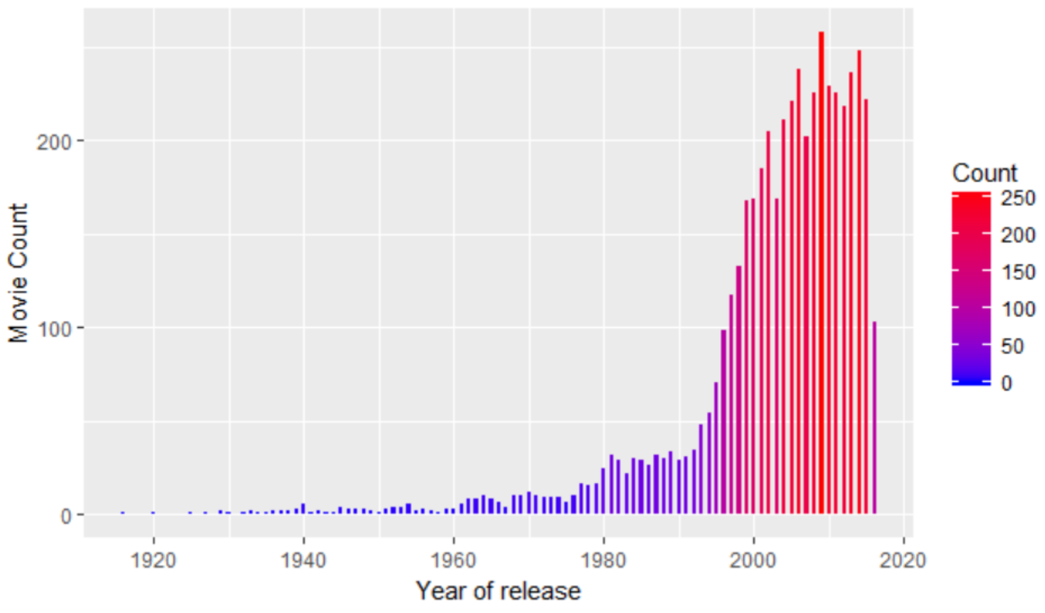
In the predictor "country", most of them are movies from the USA. However, we assume that country would have an impact on our prediction, so we change the country into two categories, i.e. the USA and others.

Figure 1: Missing Values Heatmap



During further selection, we notice that the dataset only contains a few movies before 1980 (shown in Figure 2), and they often receive higher scores. In this case, we remove observations before 1980 as outliers. Also, variables like "name of the director", "name of casts”, and ”keywords in the poster” can not be used as predictors, so we remove them from our dataset as well. We also find that ”Facebook like” is 0 for some movie but it’s not actually represent the true social media reaction for the movie. It might be due to the fact that some movie does not have their own Facebook pages. At last, we replace the value ”0” with the mean value of the whole dataset.

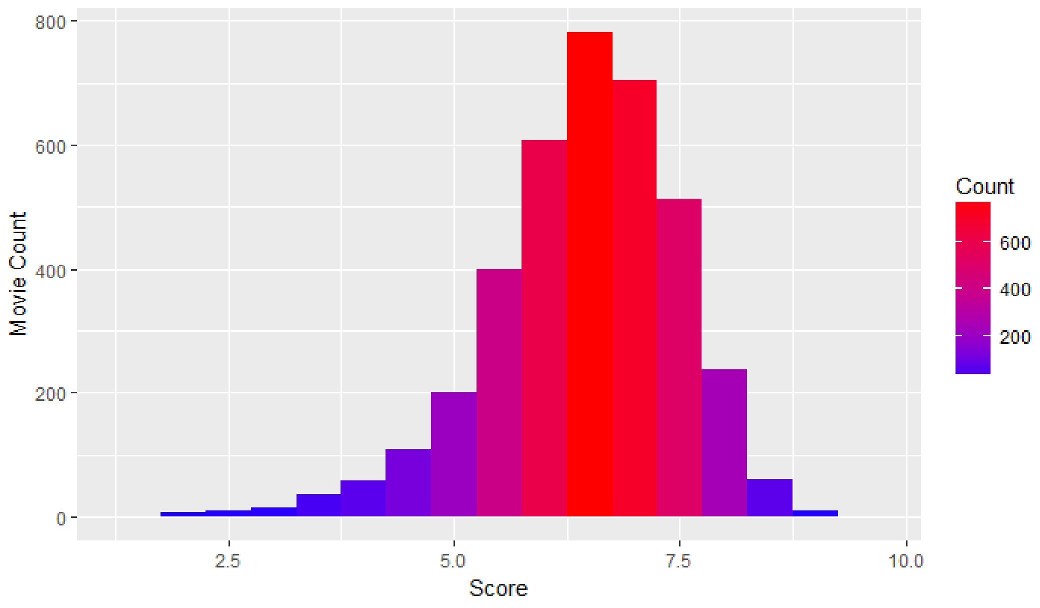
Figure 2: Histogram of Movie Released



After omitting the missing values for the second time, we basically finish our preprocessing process. Figure 3 shows the overall distribution of the IMDB scores.

```
See R code and results in "project 2.rmd" file.
```

Figure 3: Histogram of Movie IMDB Score



3 Multiple Regression Model

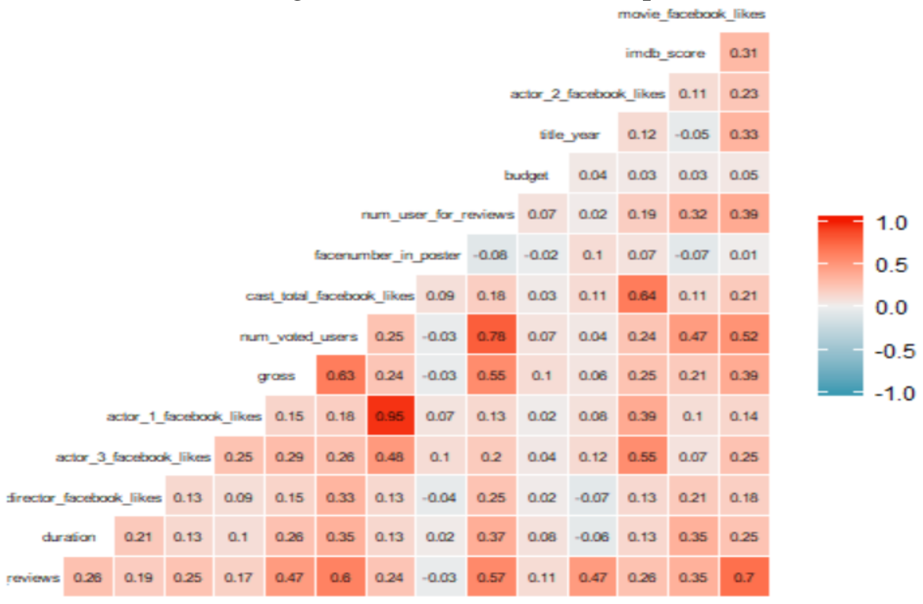
In this section, we are going to predict the movie score, so we keep the movie score as numeric.

3.1 Feature Engineer

We use correlation matrix to figure out the relationship between variables. In general, if the correlation coefficient is greater than 0.7, the two variables are considered significantly related.

From the correlation matrix, we find that the correlation coefficient between actor 1's facebook likes and cast total facebook likes is 0.95, and the correlation coefficient between actor 2's facebook likes and cast total likes is 0.64. In addition, number of voted users and number of user for reviews and number critic for reviews have high correlation. We use stepwise regression to minimize the influence of multi-collinearity on parameter estimation.

Figure 4: Correlation Heatmap



Also, When we have a large number of variables X in the model there will generally be many that have little or no effect on Y. Leaving these variables in the model makes it harder to see the “big picture”. The model would be easier to interpret by removing the unimportant variables. We can also use stepwise to make our model simple. To be specific, we built a null model and then as we added variables into the model, We use AIC as criteria to find out the best one.

3.2 Contruct Model

From the boxplot for genres, we find action, adventure, comedy and drama has a lot of outliers. From the boxplot for year, we find the mean movie score varies across years.

Figure 5: Boxplot of IMDB Score and Genres

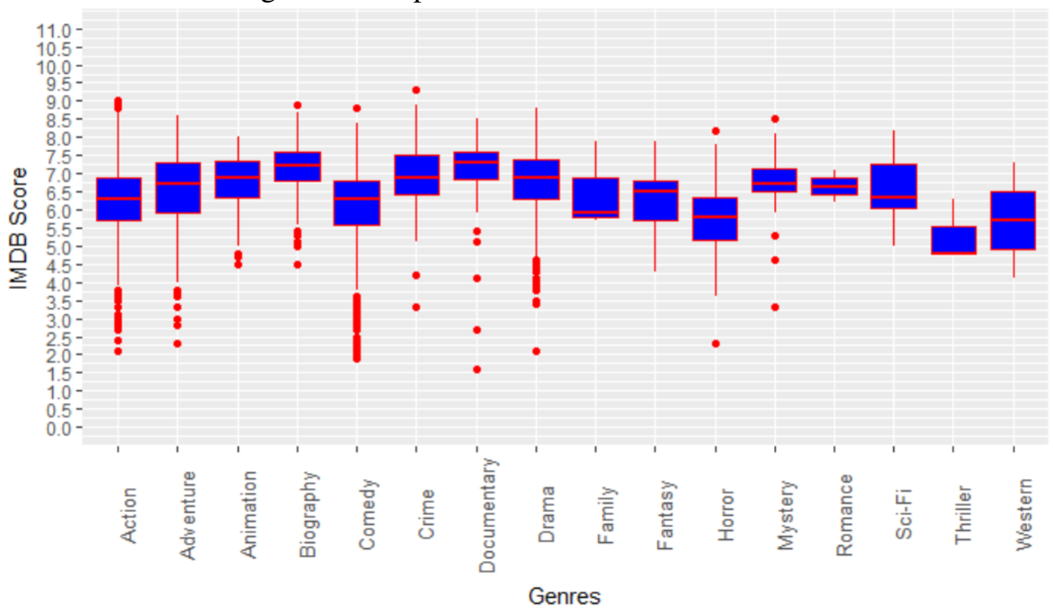
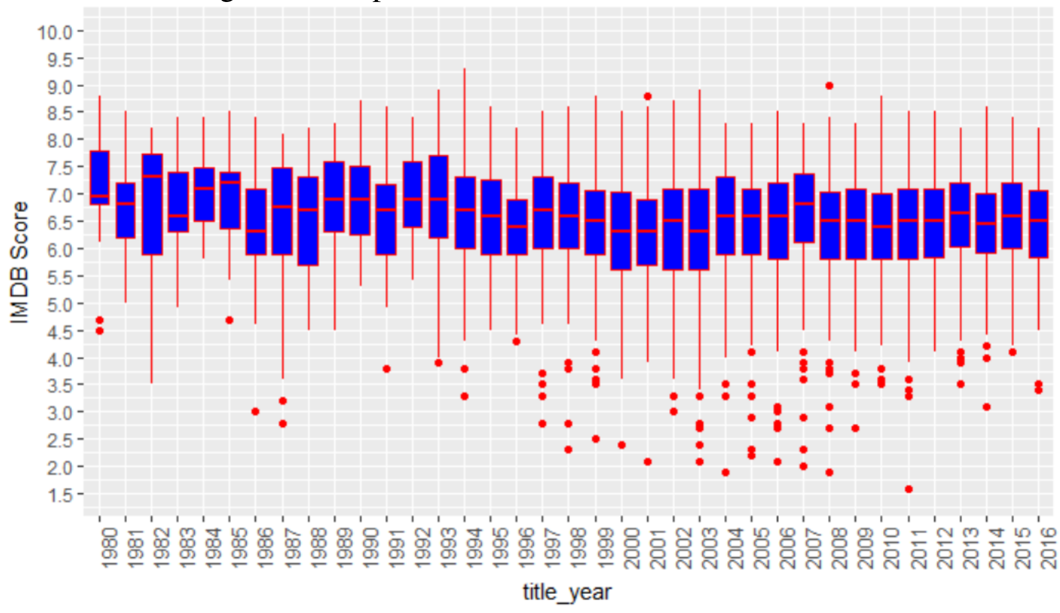


Figure 6: Boxplot of IMDB Score and Released Year



3.2.1 Setup an Original Model

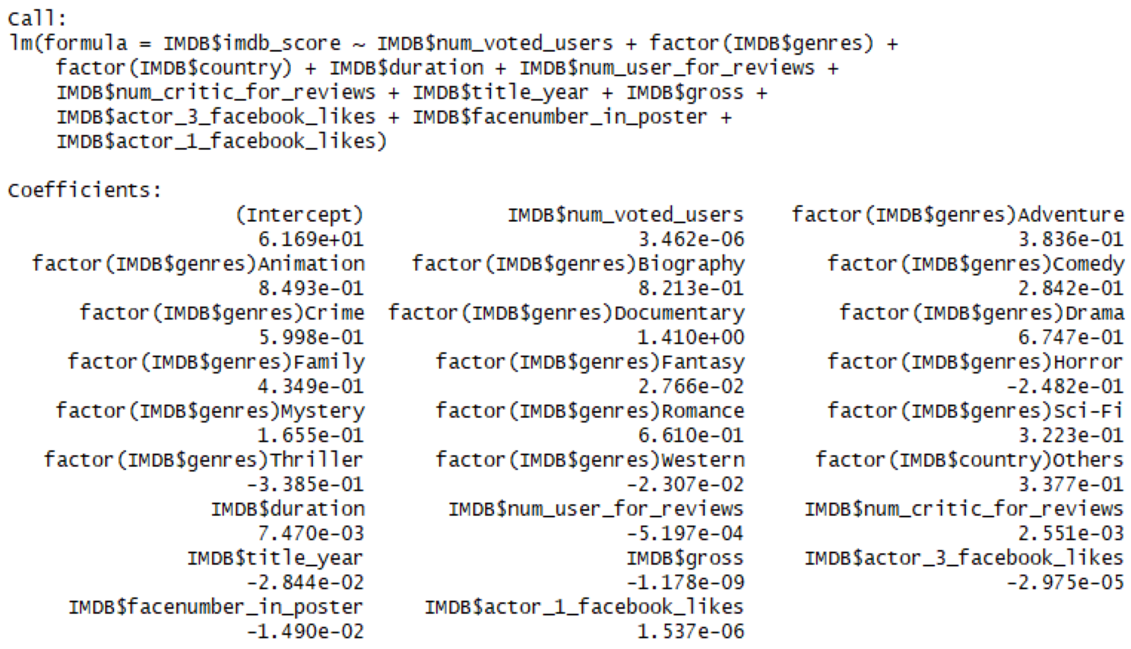
We use stepwise regression method to figure out the best model. Firstly, we built a null model and a full model with all features in.

```
null <- lm(IMDB$imdb_score ~ 1) # set null model
null <- lm(IMDB$imdb_score ~ .) # set full model
```

Then as we added variables into the model, we use AIC as criteria to find out the best one.

```
step(null, scope = list(lower = null, upper = full1),
direction = 'forward')
```

Finally we get the multiple regression model as follows:



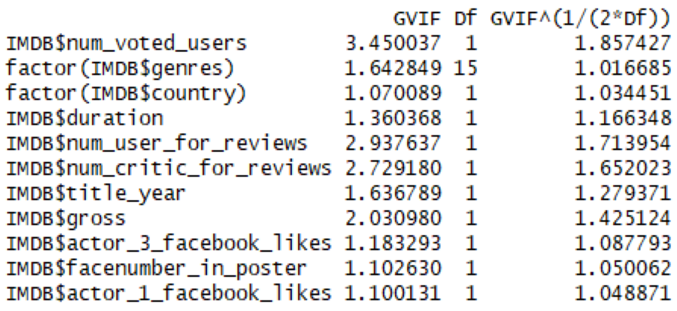
```
Call:
lm(formula = IMDB$imdb_score ~ IMDB$num_voted_users + factor(IMDB$genres) +
    factor(IMDB$country) + IMDB$duration + IMDB$num_user_for_reviews +
    IMDB$num_critic_for_reviews + IMDB$title_year + IMDB$gross +
    IMDB$actor_3_facebook_likes + IMDB$facenumber_in_poster +
    IMDB$actor_1_facebook_likes)

Coefficients:
                (Intercept)          IMDB$num_voted_users   factor(IMDB$genres)Adventure
                  6.169e+01                     3.462e-06                      3.836e-01
  factor(IMDB$genres)Animation   factor(IMDB$genres)Biography     factor(IMDB$genres)Comedy
                  8.493e-01                     8.213e-01                      2.842e-01
      factor(IMDB$genres)Crime  factor(IMDB$genres)Documentary      factor(IMDB$genres)Drama
                  5.998e-01                     1.410e+00                      6.747e-01
     factor(IMDB$genres)Family     factor(IMDB$genres)Fantasy     factor(IMDB$genres)Horror
                  4.349e-01                     2.766e-02                     -2.482e-01
    factor(IMDB$genres)Mystery     factor(IMDB$genres)Romance     factor(IMDB$genres)Sci-Fi
                  1.655e-01                     6.610e-01                      3.223e-01
   factor(IMDB$genres)Thriller     factor(IMDB$genres)Western    factor(IMDB$country)Others
                 -3.385e-01                    -2.307e-02                      3.377e-01
              IMDB$duration      IMDB$num_user_for_reviews   IMDB$num_critic_for_reviews
                  7.470e-03                    -5.197e-04                      2.551e-03
            IMDB$title_year                     IMDB$gross   IMDB$actor_3_facebook_likes
                 -2.844e-02                    -1.178e-09                     -2.975e-05
     IMDB$facenumber_in_poster     IMDB$actor_1_facebook_likes
                 -1.490e-02                     1.537e-06
```

Theoretically, we should also consider high order terms and interaction terms. For example, explore whether the number of voted users can affect the score by affecting the actor's Facebook likes. It is possible to see a large correlation between some of the features in the correlation coefficient figure, which may be allowed to be represented by interaction terms.

3.2.2 Diagonastic

Then, we checked multicollinearity and it turned out no multicollinearity existed (the VIF of all the variables are below 10).



```
                                GVIF Df GVIF^(1/(2*Df))
IMDB$num_voted_users        3.450037  1        1.857427
factor(IMDB$genres)         1.642849 15        1.016685
factor(IMDB$country)        1.070089  1        1.034451
IMDB$duration               1.360368  1        1.166348
IMDB$num_user_for_reviews   2.937637  1        1.713954
IMDB$num_critic_for_reviews 2.729180  1        1.652023
IMDB$title_year             1.636789  1        1.279371
IMDB$gross                  2.030980  1        1.425124
IMDB$actor_3_facebook_likes 1.183293  1        1.087793
IMDB$facenumber_in_poster   1.102630  1        1.050062
IMDB$actor_1_facebook_likes 1.100131  1        1.048871
```

Accordingly, in our model, we have number of voted users, genres, country, duration, number of users for reviews, number of critic for reviews, title year, gross movie box, actor 1's facebook likes, face number in poster and actor 3's facebook like.

From the plot of residual vs fitted values, we can see the points cluster and are not randomly distributed around the ”0” reference line. In addition, the QQ plot indicates the residuals are not normally distributed. Therefore, the model we get may not be a good one. The possible reason may be that this method has the i.i.d. and normal distribution assumption. Moreover, as this model also have 11 independent variables, we could conduct ridge regression and LASSO later.

Figure 7: Residual vs Fitted plot

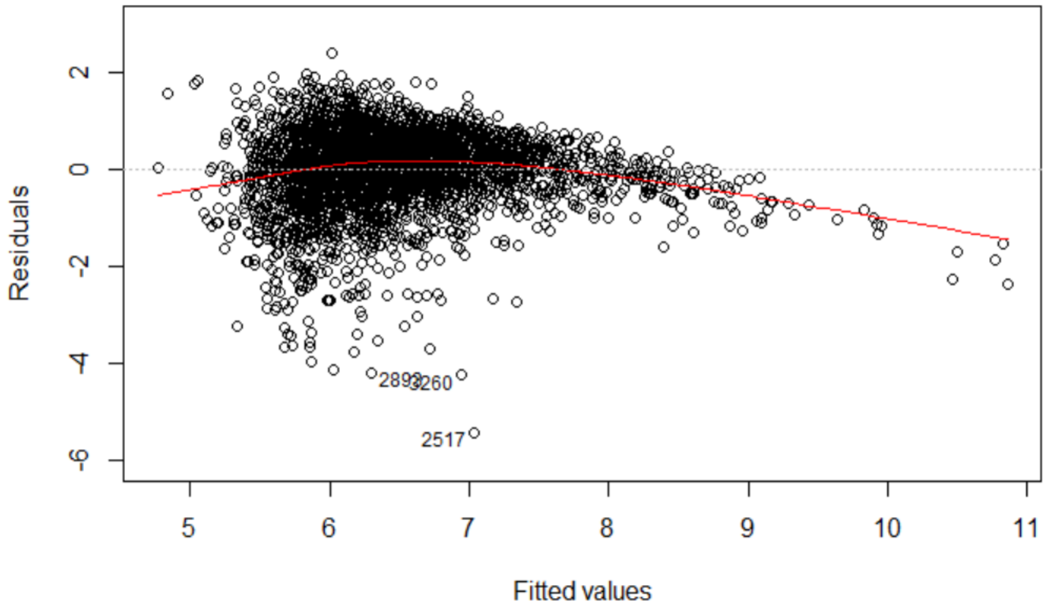
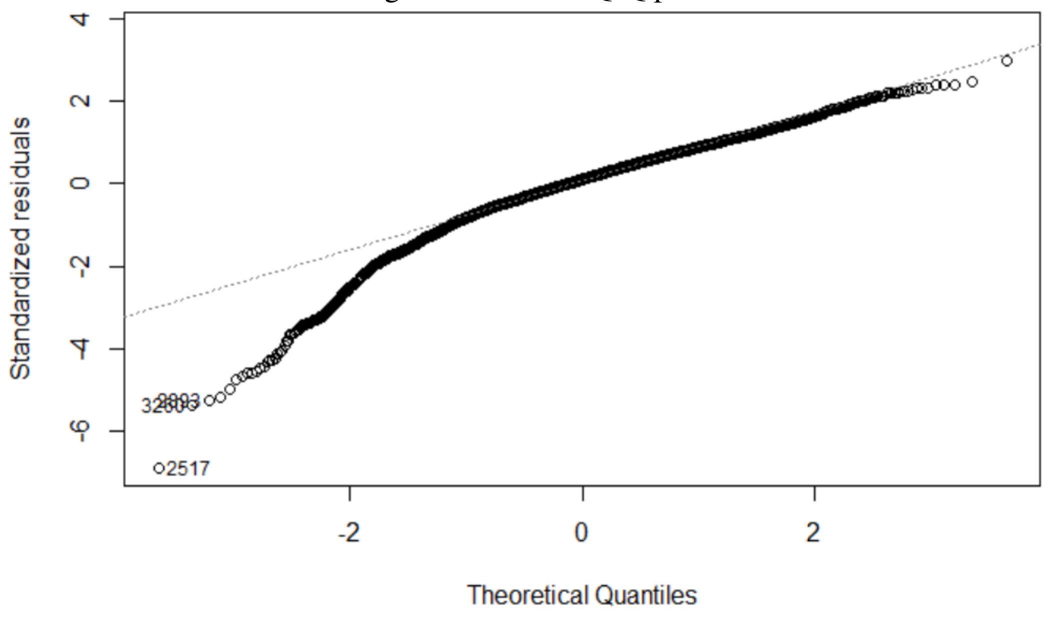


Figure 8: Normal Q-Q plot



4 Multiple Logistic Regression Model

In this section, rather than modeling the response Y directly, logistic regression models output the probability that Y belongs to a particular category. Therefore, for our data, the logistic regression models output the probability of whether the movie is a ”good” one. For example, the probability of movie being ”good” given the number of facebook likes can be written as

$$Pr(good = 1|the\ number\ of\ facebook\ likes)$$

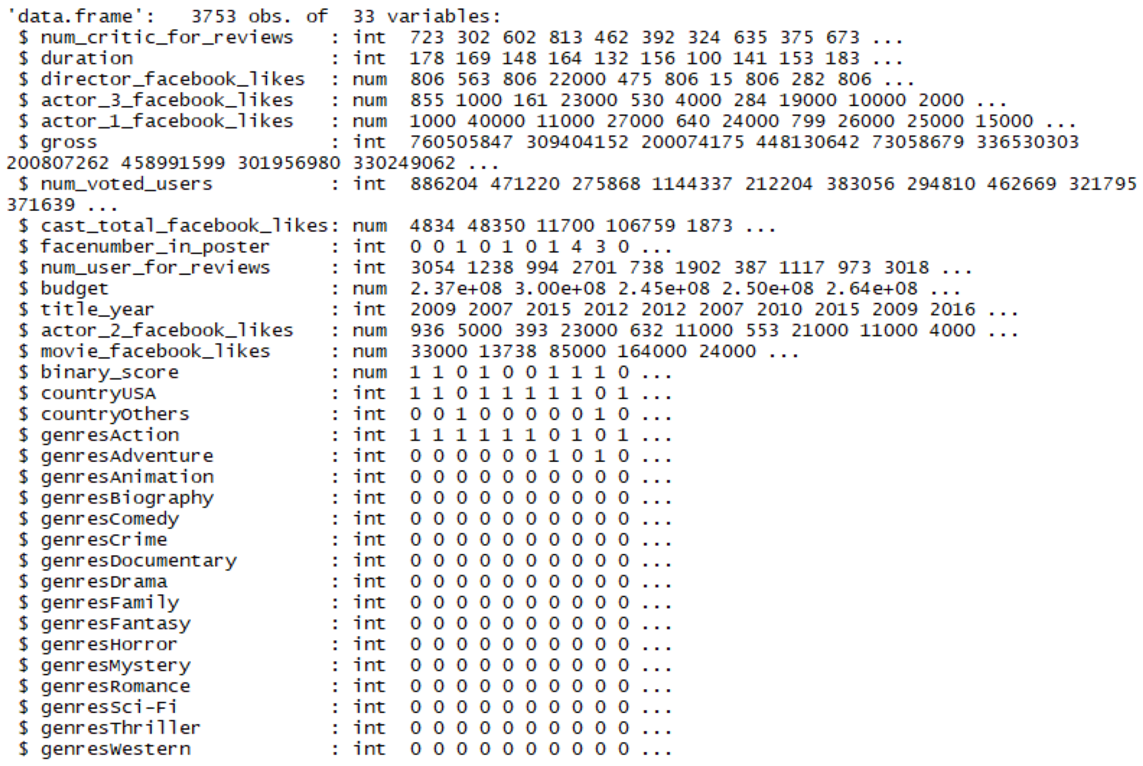
where the values range from 0 to 1. Then for any given value of the number of facebook likes, a prediction can be made for ”good” movie. For example, we could predict that movie is ”good” for any movie whose predicted probability of being ”good” is > 7. Logistic regression uses the logistic function fitted by maximum likelihood. The logistic function will always produce an S-shaped curve, so regardless of the value of x, we will always return a sensible prediction. To interpret the model we can rearrange the equation so that we return the odds. Odds are traditionally used instead of probabilities in horse-racing, since they relate more naturally to the correct betting strategy. Taking the log of the equation simplifies it further. Now it looks similar to what we have seen in linear regression.

$$log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

The left hand side is now the log-odds or logit. Instead of linear regression where one unit change in the response variable Y results in one unit change of X, in logistic regression, increasing X by one unit changes the log odds by 1. Since the probability response is not a straight line, the amount of change one unit has across values of X changes depending on the value of X. However, if 1 is positive, then increasing X will result in an increase in probability, and vice-versa.

4.1 Switch IMDB Score into Binary Response Variable

Sometimes, we don't need to know the exact score for a certain movie. The only thing we want to know is a rough judgment of the movie's quality, such as "good" or "bad". As movie score is not a binary variable, we classified it by setting thresholds. We consider movies have the score lower than 7 as bad movies and those with score higher than 7 as good ones, and assign "0" and "1" to the classes separately. We can see, in our dataset, we have 1249 "good movies" and 2504 "bad movies". Also we notice that in logistic regression we have to switch categorical variables into dummy variables. After re-edit the dataset, we get our data like this:

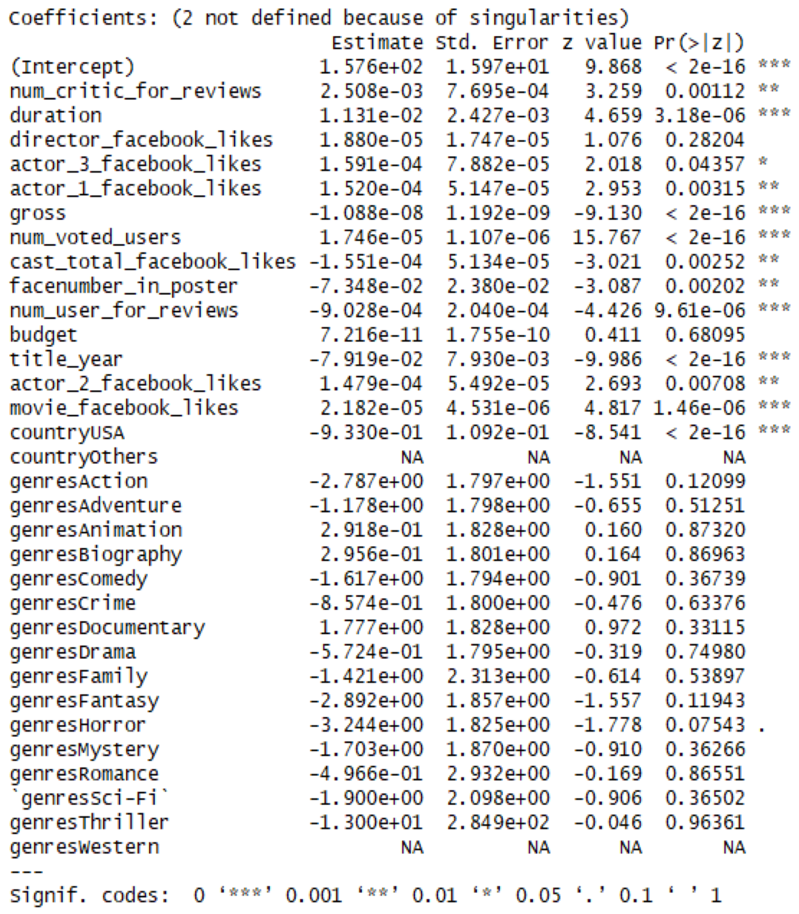


```
'data.frame':   3753 obs. of  33 variables:
 $ num_critic_for_reviews   : int  723 302 602 813 462 392 324 635 375 673 ...
 $ duration                 : int  178 169 148 164 132 156 100 141 153 183 ...
 $ director_facebook_likes  : num  806 563 806 22000 475 806 15 806 282 806 ...
 $ actor_3_facebook_likes   : num  855 1000 161 23000 530 4000 284 19000 10000 2000 ...
 $ actor_1_facebook_likes   : num  1000 40000 11000 27000 640 24000 799 26000 25000 15000 ...
 $ gross                    : int  760505847 309404152 200074175 448130642 73058679 336530303
200807262 458991599 301956980 330249062 ...
 $ num_voted_users          : int  886204 471220 275868 1144337 212204 383056 294810 462669 321795
371639 ...
 $ cast_total_facebook_likes: num  4834 48350 11700 106759 1873 ...
 $ facenumber_in_poster     : int  0 0 1 0 1 0 1 4 3 0 ...
 $ num_user_for_reviews     : int  3054 1238 994 2701 738 1902 387 1117 973 3018 ...
 $ budget                   : num  2.37e+08 3.00e+08 2.45e+08 2.50e+08 2.64e+08 ...
 $ title_year               : int  2009 2007 2015 2012 2012 2007 2010 2015 2009 2016 ...
 $ actor_2_facebook_likes   : num  936 5000 393 23000 632 11000 553 21000 11000 4000 ...
 $ movie_facebook_likes     : num  33000 13738 85000 164000 24000 ...
 $ binary_score             : num  1 1 0 1 0 0 1 1 1 0 ...
 $ countryUSA               : int  1 1 0 1 1 1 1 1 0 1 ...
 $ countryOthers            : int  0 0 1 0 0 0 0 0 1 0 ...
 $ genresAction             : int  1 1 1 1 1 1 0 1 0 1 ...
 $ genresAdventure          : int  0 0 0 0 0 0 1 0 1 0 ...
 $ genresAnimation          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresBiography          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresComedy             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresCrime              : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresDocumentary        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresDrama              : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresFamily             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresFantasy            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresHorror             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresMystery            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresRomance            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresSci-Fi             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresThriller           : int  0 0 0 0 0 0 0 0 0 0 ...
 $ genresWestern            : int  0 0 0 0 0 0 0 0 0 0 ...
```

`See R code and results in "project 2.rmd" file.`

4.2 Logistic Regression Model

```
model2 <- glm(binary_score ~ ., data = IMDB_2, family =
"binomial")
```



```
Coefficients: (2 not defined because of singularities)
                          Estimate Std. Error z value Pr(>|z|)    
(Intercept)              1.576e+02  1.597e+01   9.868  < 2e-16 ***
num_critic_for_reviews   2.508e-03  7.695e-04   3.259  0.00112 ** 
duration                 1.131e-02  2.427e-03   4.659 3.18e-06 ***
director_facebook_likes  1.880e-05  1.747e-05   1.076  0.28204    
actor_3_facebook_likes   1.591e-04  7.882e-05   2.018  0.04357 *  
actor_1_facebook_likes   1.520e-04  5.147e-05   2.953  0.00315 ** 
gross                   -1.088e-08  1.192e-09  -9.130  < 2e-16 ***
num_voted_users          1.746e-05  1.107e-06  15.767  < 2e-16 ***
cast_total_facebook_likes -1.551e-04  5.134e-05  -3.021  0.00252 ** 
facenumber_in_poster    -7.348e-02  2.380e-02  -3.087  0.00202 ** 
num_user_for_reviews    -9.028e-04  2.040e-04  -4.426 9.61e-06 ***
budget                   7.216e-11  1.755e-10   0.411  0.68095    
title_year              -7.919e-02  7.930e-03  -9.986  < 2e-16 ***
actor_2_facebook_likes   1.479e-04  5.492e-05   2.693  0.00708 ** 
movie_facebook_likes     2.182e-05  4.531e-06   4.817 1.46e-06 ***
countryUSA              -9.330e-01  1.092e-01  -8.541  < 2e-16 ***
countryOthers                   NA         NA      NA       NA    
genresAction            -2.787e+00  1.797e+00  -1.551  0.12099    
genresAdventure         -1.178e+00  1.798e+00  -0.655  0.51251    
genresAnimation          2.918e-01  1.828e+00   0.160  0.87320    
genresBiography          2.956e-01  1.801e+00   0.164  0.86963    
genresComedy            -1.617e+00  1.794e+00  -0.901  0.36739    
genresCrime             -8.574e-01  1.800e+00  -0.476  0.63376    
genresDocumentary        1.777e+00  1.828e+00   0.972  0.33115    
genresDrama             -5.724e-01  1.795e+00  -0.319  0.74980    
genresFamily            -1.421e+00  2.313e+00  -0.614  0.53897    
genresFantasy           -2.892e+00  1.857e+00  -1.557  0.11943    
genresHorror            -3.244e+00  1.825e+00  -1.778  0.07543 .  
genresMystery           -1.703e+00  1.870e+00  -0.910  0.36266    
genresRomance           -4.966e-01  2.932e+00  -0.169  0.86551    
`genresSci-Fi`          -1.900e+00  2.098e+00  -0.906  0.36502    
genresThriller          -1.300e+01  2.849e+02  -0.046  0.96361    
genresWestern                   NA         NA      NA       NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The NA occurs in countryOthers because we consider it as the baseline of dummy. With other conditions unchanged, the log odds of movie being ”good” will increase 2.182×10^{-5} as movie facebook likes increase 1 unit.The log odds of movie being ”good” will increase 2.508×10^{-3} as number critic for reviews increase 1 unit. The log odds of movie being ”good” will increase 1.131×10^{-2} as duration increase 1 unit. The log odds of movie being ”good” will increase 1.591×10^{-4} as actor 3’s facebook likes increase 1 unit. The log odds of movie being ”good” will increase 1.520×10^{-4} as actor 1’s facebook likes increase 1 unit. The log odds of movie being ”good” will decrease 1.088×10^{-8} as gross ticket box increase 1 unit. The log odds of movie being ”good” will increase 1.746×10^{-5} as number of voted users increase 1 unit.

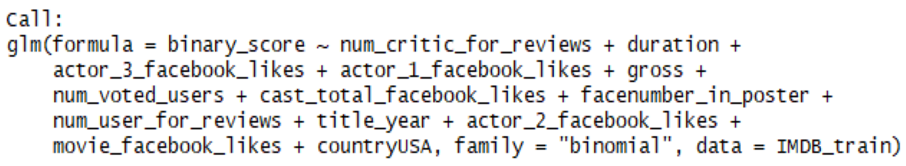
The result of multiple Logistic regression shows that the genre of movie doesn’t affect the movie score. However, the result may differ due to different classification methods as we mentioned before. For the people involved in a movie, the director variable is not significant which means people focus more on actors rather than directors. Moreover, whether a movie is good or bad is not related with its budget, which means we can still make good movies with limited budget.

4.3 Model Evaluation

To evaluate the performance of our logistic model, we split the data into a train set (0.75) and test set (0.25).

```
split <- sample.split(IMDB_2$binary_score, SplitRatio = 0.75)
#get training and test data
IMDB_train <- subset(IMDB_2, split == TRUE)
IMDB_test <- subset(IMDB_2, split == FALSE)
```

In each set, We conduct the Logistic regression with the significant variables and use it to predict whether a movie is "good" or "bad". Then, we plot the ROC curve to assist on selection of threshold and calculate the accuracy of this model, and repeat in test set using this threshold.

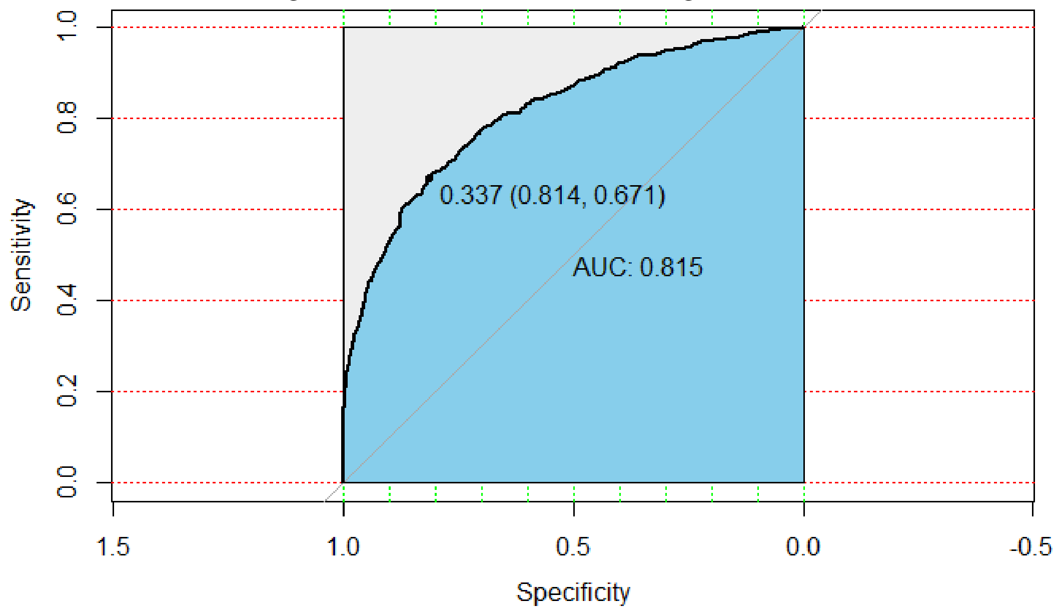


```
Call:
glm(formula = binary_score ~ num_critic_for_reviews + duration +
    actor_3_facebook_likes + actor_1_facebook_likes + gross +
    num_voted_users + cast_total_facebook_likes + facenumber_in_poster +
    num_user_for_reviews + title_year + actor_2_facebook_likes +
    movie_facebook_likes + countryUSA, family = "binomial", data = IMDB_train)
```

Receiver Operating Characteristic(ROC) summarizes the model's performance by evaluating the trade offs between true positive rate (sensitivity) and false positive rate(1 - specificity). For plotting ROC, it is advisable to assume $p > 0.5$ since we are more concerned about success rate. ROC summarizes the predictive power for all possible values of $p > 0.5$. The area under curve (AUC), referred to as index of accuracy(A) or concordance index, is a perfect performance metric for ROC curve. Higher the area under curve, better the prediction power of the model. The ROC of a perfect predictive model has True-Positive Rate equals 1 and False-Positive Rate equals 0. This curve will touch the top left corner of the graph.

Figure 9 is the ROC curve for training dataset, The best threshold is 0.337, that is we consider movies with probability larger than 0.337 as "good" movies and those with probability smaller than 0.337 as "bad" ones. The accuracy of this model is 76.6%.

Figure 9: ROC Curve for Training Dataset



The confusion matrix of train set is as follows:

```
    FALSE TRUE
  0  1527  351
  1   308  629
[1] "Accuracy of Multiple Logistic Model is 0.765896980461812"
```

We apply the threshold=0.337 we get from training dataset to test dataset. The accuracy of this model in test set is 76.9%.

```
    FALSE TRUE
  0   507  119
  1    98  214
[1] "Accuracy of Multiple Logistic Model is 0.76865671641791"
```

5 Ridge Regression Model

In the first step when constructing the regression model, we find that there might be high levels of correlations among variables, i.e. the effects of multicollinearity might exist here. Then, besides the stepwise regression method we use previously, we can apply Ridge regression here.

Ridge regression is used for independent variables with multicolinearity. In addition, with many predictors, fitting the full model without penalization will result in large prediction intervals, and LS regression estimator may not uniquely exist. Ridge regression places a particular form of constraint on the parameters (β's): $\hat{\beta}_{ridge}$ is chosen to minimize the penalized sum of squares:

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

which is equivalent to minimization of $\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2$ subject to, for some $c > 0$, $\sum_{j=1}^{p} \beta_j^2 < c$, i.e. constraining the sum of the squared coefficients.

Therefore, ridge regression puts further constraints on the parameters, β_j's, in the linear model. In this case, what we are doing is that instead of just minimizing the residual sum of squares we also have a penalty term on the β_j's. This penalty term is λ (a pre-chosen constant) times the squared norm of the β vector. As λ increases, the standardized coefficients shrinks towards zero. If the β_j's take on large values, the optimization function is penalized. We would prefer to take smaller β_j's, or β_j's that are close to zero to drive the penalty term small.

Unlike linear regression, Ridge regression cannot auto-process categorical variables. So we need to first convert the categorical variables into dummy variables. In order to construct the new dataset, we need to add the initial "imbd_score" as a new variable to the dataset "IMDB_2". Next, as we try the full linear model, we find the existence of high level of multicollinearity. Then, we run Ridge regression for our dataset.

```
See R code and results in "project 2.rmd" file.
```

5.1 Running of Ridge Regression

5.1.1 Use the "lm.ridge" Function in the "MASS" Package

```
model3 <- lm.ridge(imdb_score ~ . - binary_score, IMDB_3,
lambda = seq(0,0.1,0.001))

modified HKB estimator is 3.171416e-25
modified L-W estimator is 40.23315
smallest value of GCV  at 0.1
```

The result shows that the optimal λ here is 0.1.

5.1.2 Use the "linearRidge()" Function to Auto-select parameters

```
model4 <- linearRidge(imdb_score ~ . - binary_score, data = IMDB_3)
```

```
Ridge parameter: 0.02150855, chosen automatically,
computed using 22 PCs
```

The result shows that the optimal λ changes to 0.022. There have been significant changes for the coefficients of each variable.

5.1.3 Use the "train" Function in the "caret" Package

We adjust and optimize the parameters with the "train" function by setting up the cross-validation and parameter optimization range. Here, we apply 10-fold cross-validation.

```
trainx1 <- IMDB_3[, -c(15, 34)]
trainy1 <- IMDB_3[, 34]
ctrl1 <- trainControl(method = "cv", number = 10)
ridgeGrid <- data.frame(.lambda=seq(0, .1, length = 10))
set.seed(100)
ridgeRegTune <- train(trainx1, trainy1, method = "ridge",
tuneGrid = ridgeGrid, trControl = ctrl1, preProc = c("center","scale"))
```

```
RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was lambda = 0.
```

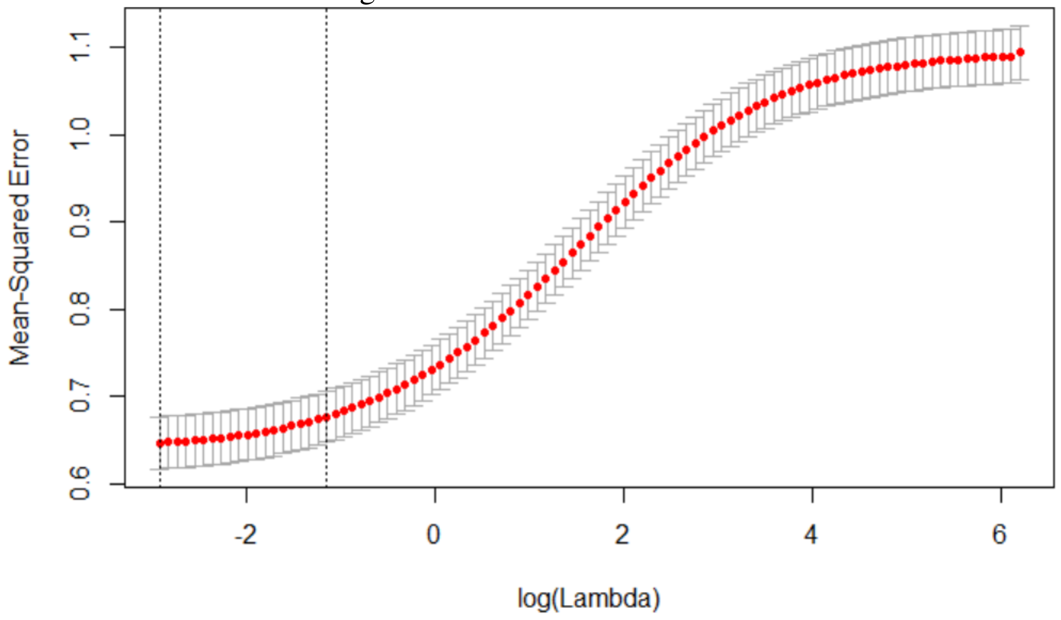
Now the optimal λ is 0.

5.1.4 Use the "cv.glmnet" Function in the "glmnet" Package

```
set.seed(1)
x <- model.matrix(imdb_score ~ . - binary_score, IMDB_3)[, -1]
y <- IMDB_3$imdb_score
cv_fit <- cv.glmnet(x, y, alpha = 0)
bestlam <- cv_fit$lambda.min
predict(cv_fit, type = 'coefficients', s = bestlam)
model5 <- cv_fit$glmnet.fit
```

The optimal λ here is 0.054. We find that there are differences in the optimal λ when applying different methods, but the λ values are all close to 0. Figure 10 shows out cross-validation fit across different $log\lambda$ values.

Figure 10: Cross-Validation Fit



5.2 Fitting Precision

We use the model in the previous ”cv.glmnet” as our ridge regression model:

```
Call:
lm(formula = IMDB$imdb_score ~ IMDB$num_voted_users + factor(IMDB$genres) +
    factor(IMDB$country) + IMDB$duration + IMDB$num_user_for_reviews +
    IMDB$num_critic_for_reviews + IMDB$title_year + IMDB$gross +
    IMDB$actor_3_facebook_likes + IMDB$facenumber_in_poster +
    IMDB$actor_1_facebook_likes)
```

```
R^2 of Ridge Regressioni Model is 0.420843173190629
R^2 of Ridge Regressioni Model is 0.4239
```

The result in R shows that the R^2 value of Ridge regression model is 0.4208, and the R^2 value of our first line regression model is 0.4239. Both models have relatively low R^2 values and that of the Ridge regression model is smaller. Next, we are fitting a LASSO Model to see whether it is going to be a better fit.

6 LASSO Model

Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of muticollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination. The acronym “LASSO” stands for Least Absolute Shrinkage and Selection Operator.

The LASSO estimates the β by minimizing the $|\beta|$:

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

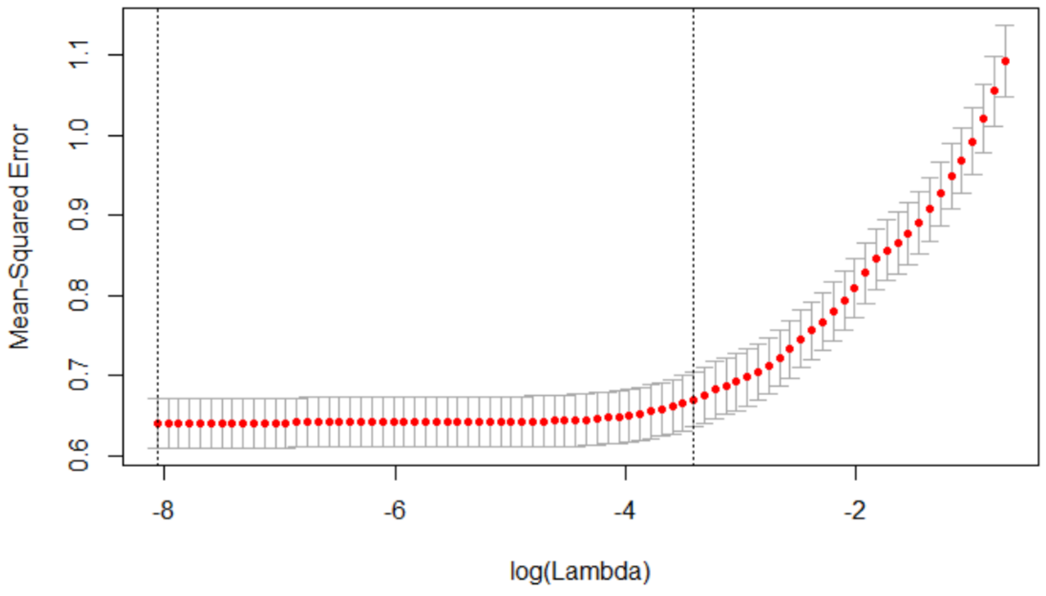
which is equivalent to minimization of $\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2$ subject to, for some $c > 0$, $\sum_{j=1}^{p} |\beta_j| < c$, i.e. constraining absolute value of the coefficients.

6.1 LASSO Model

```
cv_fitlas <- cv.glmnet(x, y, alpha = 1)
bestlam_las <- cv_fitlas$lambda.min
```

It is the cross-validation curve (red dotted line), and upper and lower standard deviation curves along the λ sequence (error bars). Two selected λs are indicated by the vertical dotted lines. lambda.min is the value of λ that gives minimum mean cross-validated error. Here, out optiaml λ is 0.0003157762.

Figure 11: Cross-Validation Curve of LASSO



Predictions can be made based on the fitted cv.glmnet object.

6.2 The Accuracy of LASSO Model

```
R^2 of LASSO Model is 0.366134134694741
```

The R^2 of LASSO model is 0.3661 which is smaller than 0.4239 from linear regression model and 0.4208 from Ridge model. It is not a better fit.

7 Decision Tree Model

Decision Tree Model is another way to make prediction in a regression problem by dividing the predictor space into distinct regions. The dividing criteria can be represent as a tree like graph or model. Each node of the decision tree stands for one attribute for decision making, while each branch represent the outcome of the test. The tree is build by finding the attribute which dividing the predictor space into subsets have minimize variance within group, and continue doing the partition for every region. Then for every new observation falls in a particular region, we can make the same prediction as the region. Comparing with other algorithm, decision tree model is more intuitive in representing the decision making process. However, the algorithm completed when the subset at a node has all the same value of the target variable, or when splitting no longer adds value to the predictions. In practical use, we need to balance the accuracy and complexity by pruning the tree.

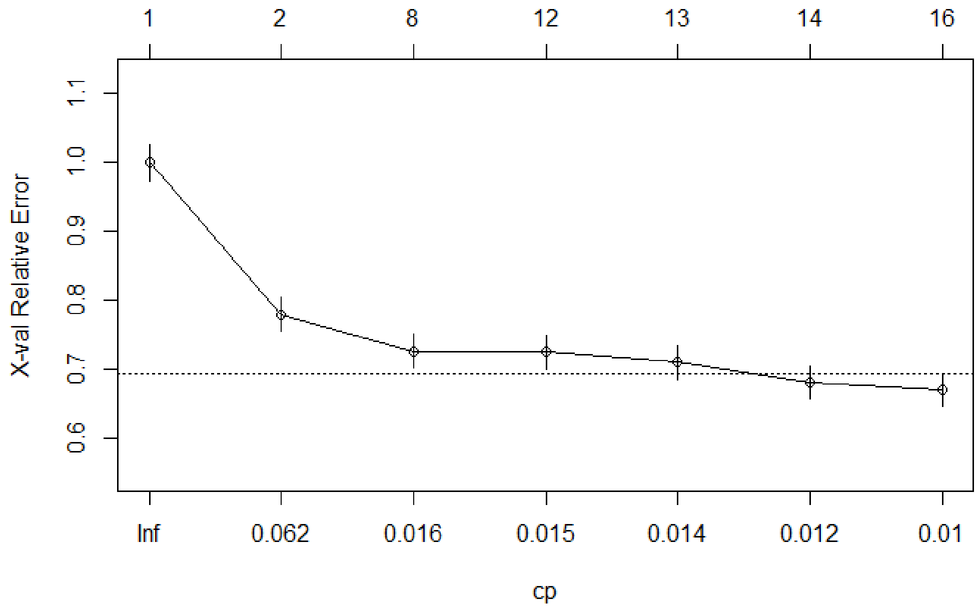
7.1 Running the Decision Tree Model

We conduct the decision tree model on the dataset for the logistic regression, i.e. our response variable is binary. In this dataset, ”0” stands for movie have the score lower than 7 and ”1” stands for the movie with score hight than 7. We grow a classification tree for the categorical response variable.

7.2 Pruning the Decision Tree

A large decision tree might overfit the training data and act poorly with test data. We need to prune the decision tree to balance the bias and variance. Pruning the decision tree means that reduce the size of tree by removing sections that provide limited power for classification. We follow the minimal cost complexity pruning (MCCP) to prune our decision tree. The goal is to get the tree with enough accuracy when the level of complexity is minimized. The minimal cost complexity is defined as: $R_\alpha(T) = R(T) + \alpha|\tilde{T}|$. Where α is the complexity parameter, i.e. CP. We can plot the CP values to see which one minimizes the cross-validation error. The plot shows the cross validation error against different CP value. We choose the $CP = 0.01$ with minimum error and build the new tree based on this CP value.

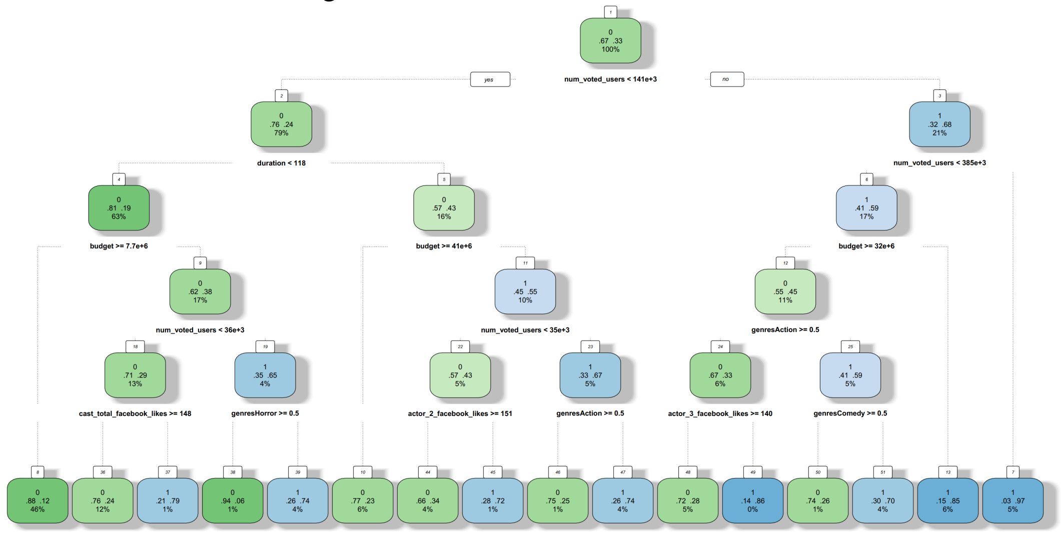
Figure 12: CP Values of Tree



```
model8 <- prune(model7, cp =
model7$cptable[which.min(model7$cptable[,"xerror"]),"CP"])
```

After pruning the tree, we get our Decision Tree model.

Figure 13: Decision Tree of Train set



The pruned decision tree based on our training dataset is plotted above. The first node shows all the observations in our training dataset which 67% of the observations belongs to the ”0” category. The algorithm classified the group as ”0” category. We can also observe that the algorithm determine the first partition criterion should be the predictor ”num_voted_users”. For any observation, with number of voted users less than 141000 should be classified as one group while observation higher than this value would be assigned to another group. This criterion is able to maximize the difference between groups and minimize the difference within groups. The second node shows the group with number of voted users lower than 141000 which takes 79% of all the observation. Among this group, 76% observation is labeled as ”0” while 24% is labeled as ”1”, so the algorithm predict this group as ”0”. Node 3 shows the observations with number of voted users value higher than the partition criteria. The algorithm continues to split the group at Node 2 and Node 3 to grow the tree. With six layers of partition, the algorithm divides the predictor space into 16 regions.

7.3 Predictive Performance

We get the confusion matrix of our decision tree model, from which we know the accuracy is 81.95%.

```
   pred_train
       0    1
  0 1741  137
  1  371  566
```

```
Accuracy of Decision Tree Model(Train) is 0.819538188277087
```

Then we repeat this model in test set and get the confusion matrix, from which we know the accuracy is 79.42%.

```
   pred_test
      0   1
  0 562  64
  1 129 183
```

```
Accuracy of Decision Tree Model(Test) is 0.794243070362473
```

8 Conclusion and Future work

For all the regression model, the R-squared of all our models are small (0.4239 in linear model, 0.4208 in Ridge model and 0.3661 in LASSO model). A potential explanation for this is that we fail to include the most important variables in our model. If we use R-squared as a criteria to choose a model, the multiple linear model will be a better choice.

For all the classification models, the accuracy for multiple Logistic model and decision tree are 0.7659 and 0.7942. Therefore, the decision tree is better than multiple Logistic model. And as the accuracy is close to 0.8, the decision tree model is of high quality.

Our analysis can still be improved with more sophisticated methods. In our project, the prediction is conducted with multiple regression, logistic regression, Ridge regression, LASSO and Decision Tree. We switch our response variables into binary form in later models. The accuracy of our prediction can be improved if we divide the response variable into multiple categories. Non-binary logistic regression, Ridge regression, LASSO and Decision Tree might return a better prediction. Also, the single Decision Tree we build is facing the problem of high variance. By introducing the bagging and boosting strategy and build the Random Forests model, the accuracy of our prediction would also have considerable improvement. Finally, there is still another method for prediction which is not used in our project, K-NN method for instance. We can include these methods in our analysis process to get a better outcome in the future.