Modeling_Katrina_Cleaned

Katrina Truebebach

March 18, 2019

```
rm(list = ls())
```

```
pdf('~/DS5110/project/output/model_plots.pdf')
```

Load cleaned data

```
load(file = '~/DS5110/data/proj_cleaned_dta.RData')
```

```
# need to drop years before 1980: too sparse
  # most of those years have 1 or 0 observations. If including year in the model, we aren't getting any
# only necessary when including year in model, but hard to compare different models then b/c data diffe
train <- train %>% filter(as.integer(as.character(year)) >= 1980)
valid <- valid %>% filter(as.integer(as.character(year)) >= 1980)
test <- test %>% filter(as.integer(as.character(year)) >= 1980)

# calculate logs
train <- train %>%
  mutate_at(vars('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
         'imdb_score', 'real_gross'), funs(log = log10(.)))
valid <- valid %>%
  mutate_at(vars('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
         'imdb_score', 'real_gross'), funs(log = log10(.)))
test <- test %>%
  mutate_at(vars('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
         'imdb_score', 'real_gross'), funs(log = log10(.)))

# label variables for graphing
label(train$real_budget) <- 'Real Budget ($)'
label(train$real_gross) <- 'Real Gross Revenue ($)'
label(train$real_budget_log) <- 'Log Real Budget ($)'
label(train$real_gross_log) <- 'Log Real Gross Revenue ($)'
label(train$imdb_score) <- 'IMDB Score'
label(train$imdb_score_log) <- 'Log IMDB Score'
label(train$content_rating) <- 'Content Rating'
label(train$year) <- 'Year'
```

Write Functions to Automate

Write function to automate stepwise
Note: not using the step() function because can't fit and find RMSE on different datasets (train, valid)

```
# function to automate each step of stepwise variable selection
# df_vars is the dataset with only the relevant variables
# var_lst is the list of variables that are in the base model
# formula is the formula with those variables besides the y variable
step_wise_step <- function(df_vars, var_lst = NULL, formula = NULL) {
  # if first step
  if (length(var_lst) == 0) {
```

```
    # rmse with each variable against real_gross
    rmse_vars <- sapply(names(df_vars), function(var) {
      # rmse of model
      rmse(lm(as.formula(str_c('real_gross_log ~', var)), data = train), data = valid)
    })
  # if > first step: exclude variables from var_lst from data and include in model formula
  } else {
      rmse_vars <- sapply(names(df_vars %>% select(-var_lst)), function(var) {
      # rmse of model
      rmse(lm(as.formula(str_c('real_gross_log ~', formula, ' + ', var)),
              data = train), data = valid)
      })
  }
  # return the name and value of the genre that resulted in the lowest RMSE
  return(rmse_vars[which.min(rmse_vars)])
}

# function to loop through each step wise loop
# adding optional starting vars and formula in case want to build off of an existing formula
step_wise_loop <- function(df_vars, starting_vars = NULL, starting_formula = NULL) {
  # list to store min RMSE from each step in
  rmse_lst <- c()

    # first step: no genre_lst or formula (default values NULL)
  min_rmse_var <- step_wise_step(df = df_vars, var_lst = starting_vars, formula = starting_formula)
  print(min_rmse_var)

    # add to list of genres, formula, and min RMSE list
  var_lst <- c(starting_vars, names(min_rmse_var))
  formula <- str_c(starting_formula, '+', names(min_rmse_var))
  rmse_lst <- c(rmse_lst, min(min_rmse_var))

  # if have starting variables, take those out of the number we are iterating through
  if (!is.null(starting_vars)) {
    df_vars_seq <- df_vars %>% select(-starting_vars)
  } else {
    df_vars_seq <- df_vars
  }
  # loop through until have considered every variable
  for (i in seq(1:(ncol(df_vars_seq)-1))) {
    print(i)
    # step
    min_rmse_var <- step_wise_step(df = df_vars, var_lst = var_lst, formula = formula)
    print(min_rmse_var)

    # add to lists
    var_lst <- c(var_lst, names(min_rmse_var))
    formula <- str_c(formula, ' + ', names(min_rmse_var))
    rmse_lst <- c(rmse_lst, min(min_rmse_var))
  }
  return(rmse_lst)
}
```

Function to graph the residuals from a model against all potential variables (included and excluded)

```r
gr_resid <- function(mod, df = train) {
  # graph residuals
  # get log versions of variables since residuals are log: same scale
  df_resid <- df %>%
    add_residuals(mod, 'lresid') %>%
    mutate_at(vars('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
           'imdb_score'), funs(log = log10(.)))

  # graph each against log residual: continuous
  lapply(c('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
           'imdb_score'), function(var) {
    print(df_resid %>%
      ggplot() +
      geom_point(aes_string(str_c(var, '_log'), y = 'lresid'))) +
      labs(x = label(df_resid[var]))
  })

  # categorical
  # can't log categorical variables
  lapply(c('content_rating', 'year', 'total_oscars_actor', 'total_oscars_director', all_genre_vars), fu
    print(df_resid %>%
      filter(!is.na(!!rlang::sym(var))) %>%
      ggplot() +
      geom_jitter(aes_string(var, 'lresid'), alpha = .3)) +
      labs(x = label(df_resid[var]))
  })

}
```

Fit Model with Genre Variables vs Real Revenue

Start with genre because it looks like it has a strong relationship with revenue (different genres have different average revenues) and because this was our original hypothesis.
However, based on our EDA, the average revenue of some genres vs not genre is almost the same. Thus we want to start by determining which genres should be included by fitting a model of only genre variables.

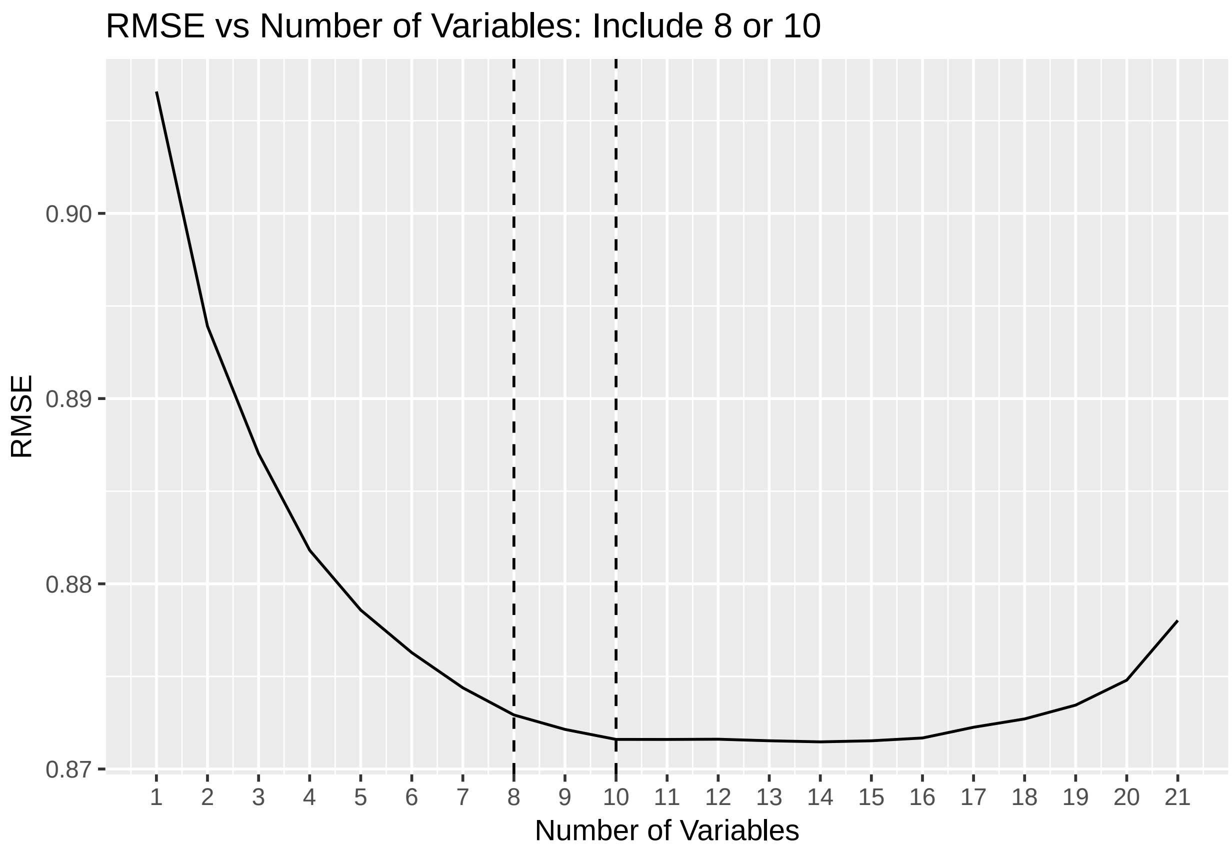
Note this is not as simple as a categorical varible because one movie can have multiple genres (adventure, action, comedy).

Step Wise Selection

Dependent variable is log(real_gross). Makes model look better *and* a lot of the relationships with other variables are more linear with log, so we will need to use this as y variable in the main model.

```r
# version of train set with just genre columns to loop through
all_genre_vars <- c('Action', 'Adventure', 'Animation', 'Biography', 'Comedy', 'Crime', 'Documentary',
                    'Drama', 'Family', 'Fantasy', 'History', 'Horror', 'Music', 'Musical', 'Mystery',
                    'Romance', 'SciFi', 'Sport', 'Thriller', 'War', 'Western')

train_genre_only <- train %>% select(all_genre_vars)

# step wise implement
# return list of all min RMSE from each step -> graph
rmse_lst <- step_wise_loop(df = train_genre_only)
```

```r
# graph RMSE at each step
fit_rmse <- tibble(nvar = 1:length(rmse_lst),
                   rmse = rmse_lst)
ggplot(fit_rmse, aes(x = nvar, y = rmse)) + geom_line() +
  scale_x_continuous(breaks = seq(1, length(rmse_lst), by = 1)) +
  geom_vline(xintercept = 8, linetype = 'dashed') +
  geom_vline(xintercept = 10, linetype = 'dashed') +
  labs(x = 'Number of Variables', y = 'RMSE',
       title = 'RMSE vs Number of Variables: Include 8 or 10')
```




```r
# after var 8, decreases too small or increase (debatably 10?)

# model based off of step wise
mod_genre <- lm(real_gross_log ~ Adventure + Action + Family + Mystery + Romance + Drama + History + Doc
mod_genre10 <- lm(real_gross_log ~ Adventure + Action + Family + Mystery + Romance + Drama + History + I


summary(mod_genre)
```

```
##
## Call:
## lm(formula = real_gross_log ~ Adventure + Action + Family + Mystery +
##     Romance + Drama + History + Documentary, data = train)
##
## Residuals:
## Log Real Gross Revenue ($)
```

```
##     Min      1Q  Median      3Q     Max
## -4.0859 -0.3211  0.1680  0.5636  1.7697
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.21040    0.04041 178.418  < 2e-16 ***
## Adventure1    0.25572    0.05959   4.291 1.87e-05 ***
## Action1       0.37493    0.05337   7.025 2.99e-12 ***
## Family1       0.46184    0.06733   6.859 9.43e-12 ***
## Mystery1      0.19739    0.07021   2.811 0.004985 **
## Romance1      0.10130    0.04886   2.073 0.038297 *
## Drama1       -0.23074    0.04398  -5.247 1.73e-07 ***
## History1      0.45334    0.12298   3.686 0.000234 ***
## Documentary1 -1.10142    0.13044  -8.444  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8582 on 1842 degrees of freedom
## Multiple R-squared:  0.1648, Adjusted R-squared:  0.1612
## F-statistic: 45.44 on 8 and 1842 DF,  p-value: < 2.2e-16
```

```r
rmse(mod_genre, data = valid)
```

```
## [1] 0.8729182
```

```r
# two additional variables Musical and War are insignificant and RMSE goes from .873 to .872
summary(mod_genre10)
```

```
##
## Call:
## lm(formula = real_gross_log ~ Adventure + Action + Family + Mystery +
##     Romance + Drama + History + Documentary + Musical + War,
##     data = train)
##
## Residuals:
## Log Real Gross Revenue ($)
##     Min      1Q  Median      3Q     Max
## -4.0873 -0.3245  0.1659  0.5635  1.7803
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.21185    0.04047 178.215  < 2e-16 ***
## Adventure1    0.25575    0.05958   4.292 1.86e-05 ***
## Action1       0.36445    0.05378   6.776 1.66e-11 ***
## Family1       0.46793    0.06792   6.890 7.65e-12 ***
## Mystery1      0.20093    0.07027   2.859  0.00429 **
## Romance1      0.10540    0.04894   2.154  0.03140 *
## Drama1       -0.23707    0.04420  -5.364 9.18e-08 ***
## History1      0.38804    0.13002   2.984  0.00288 **
## Documentary1 -1.11351    0.13064  -8.524  < 2e-16 ***
## Musical1     -0.06322    0.13019  -0.486  0.62730
## War1          0.19858    0.12895   1.540  0.12375
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.8581 on 1840 degrees of freedom
## Multiple R-squared:  0.166,  Adjusted R-squared:  0.1615
## F-statistic: 36.63 on 10 and 1840 DF,  p-value: < 2.2e-16
```

```r
rmse(mod_genre10, data = valid)
```

```
## [1] 0.8715984
```

```r
# list of these variables for future use
genre_xvar <- c('Adventure', 'Action', 'Family', 'Mystery',
                'Documentary', 'Drama', 'History', 'Romance')
```

```r
stargazer(mod_genre, type = 'latex', out = 'C:/Users/Gimli/Documents/DS5110/project/output/genre_mod.te
          no.space = T, nobs = T, dep.var.caption = '', align = T, dep.var.labels = '', title = 'Genre 
          covariate.labels = c('Adventure', 'Action', 'Family', 'Mystery', 'Romance', 'Drama', 'History
          notes.align = 'l', column.sep.width = "-1pt")
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Wed, Apr 10, 2019 - 6:10:17 PM
## % Requires LaTeX packages: dcolumn
## \begin{table}[!htbp] \centering
##   \caption{Genre Model Summary \\ Response Variable: Log Real Gross Revenue}
##   \label{}
## \begin{tabular}{@{\extracolsep{-1pt}}lD{.}{.}{-3} }
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## \\[-1.8ex] & \multicolumn{1}{c}{} \\
## \hline \\[-1.8ex]
##  Adventure & 0.256^{***} \\
##   & (0.060) \\
##   Action & 0.375^{***} \\
##   & (0.053) \\
##   Family & 0.462^{***} \\
##   & (0.067) \\
##   Mystery & 0.197^{***} \\
##   & (0.070) \\
##   Romance & 0.101^{**} \\
##   & (0.049) \\
##   Drama & -0.231^{***} \\
##   & (0.044) \\
##   History & 0.453^{***} \\
##   & (0.123) \\
##   Documentary & -1.101^{***} \\
##   & (0.130) \\
##   Constant & 7.210^{***} \\
##   & (0.040) \\
##  \hline \\[-1.8ex]
## Observations & \multicolumn{1}{c}{1,851} \\
## R$^{2}$ & \multicolumn{1}{c}{0.165} \\
## Adjusted R$^{2}$ & \multicolumn{1}{c}{0.161} \\
## Residual Std. Error & \multicolumn{1}{c}{0.858 (df = 1842)} \\
## F Statistic & \multicolumn{1}{c}{45.445$^{***}$ (df = 8; 1842)} \\
## \hline
## \hline \\[-1.8ex]
## \textit{Note:}  & \multicolumn{1}{l}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
```

```
## \end{tabular}
## \end{table}
```

This model selection by and large makes sense. All included variables are significant at some level. However, according to Qiang's graphs in EDA, some of the included genres do not make a real difference to real_gross. Especially History. Also, some genres that look like they would make a signficant difference are not included. For example, Animation.

Thoughts:

- There are a few genres that define almost all of the movies (For example, almost 80% of the movies are either Adventure, Action, Romance, or Drama). Thus, the relationship between revenue and some genres can be explained by other generes. For example, 93 out of 101 Animation movies are also Family. So Animation's effect on revenue may already by captured by Family, which is included in the model.
- On the flip side, History is included even though it seems to have a negligable effect on revenue based on the EDA bar graphs. I don't have a great explanation for this other than it was close to the cutoff RMSE for being included. 53 out of 55 History movies are also Drama. So unclear why included.

```r
train %>% filter(Animation == 1, Family == 1) %>% count() # 93
train %>% filter(Animation == 1) %>% count() # 101

train %>% filter(History == 1) %>% count() # 52
train %>% filter(History == 1, Drama == 1) %>% count() # 51
```

Residuals graph

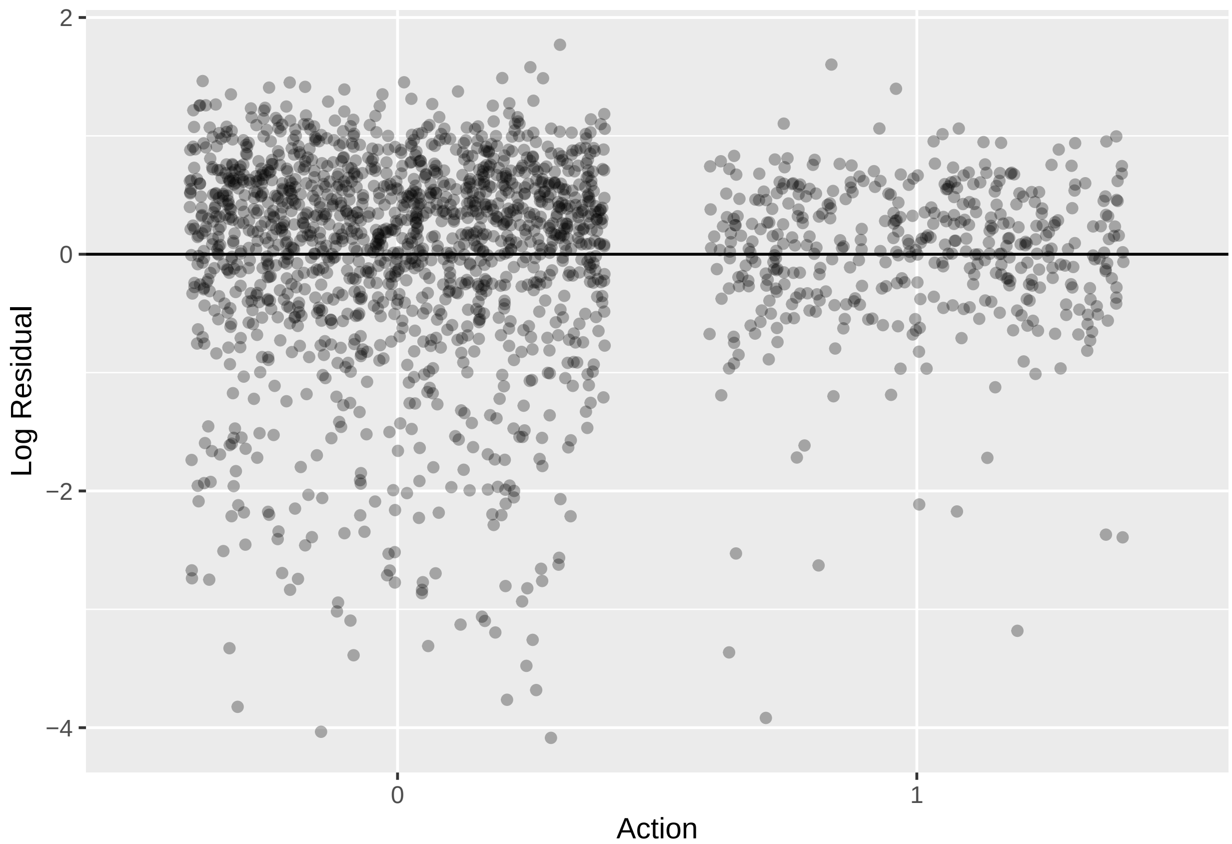
I like this geom jitter view better. Can see individual points. Most movies have some outliers where actual makes less money than predicted based on the included genres. But tricky because movies are multiple genres. So could be because that movie is also another genre that makes less money. Bulk of observations around zero.

```r
train_resid <- train %>%
  add_residuals(mod_genre, 'lresid')
```

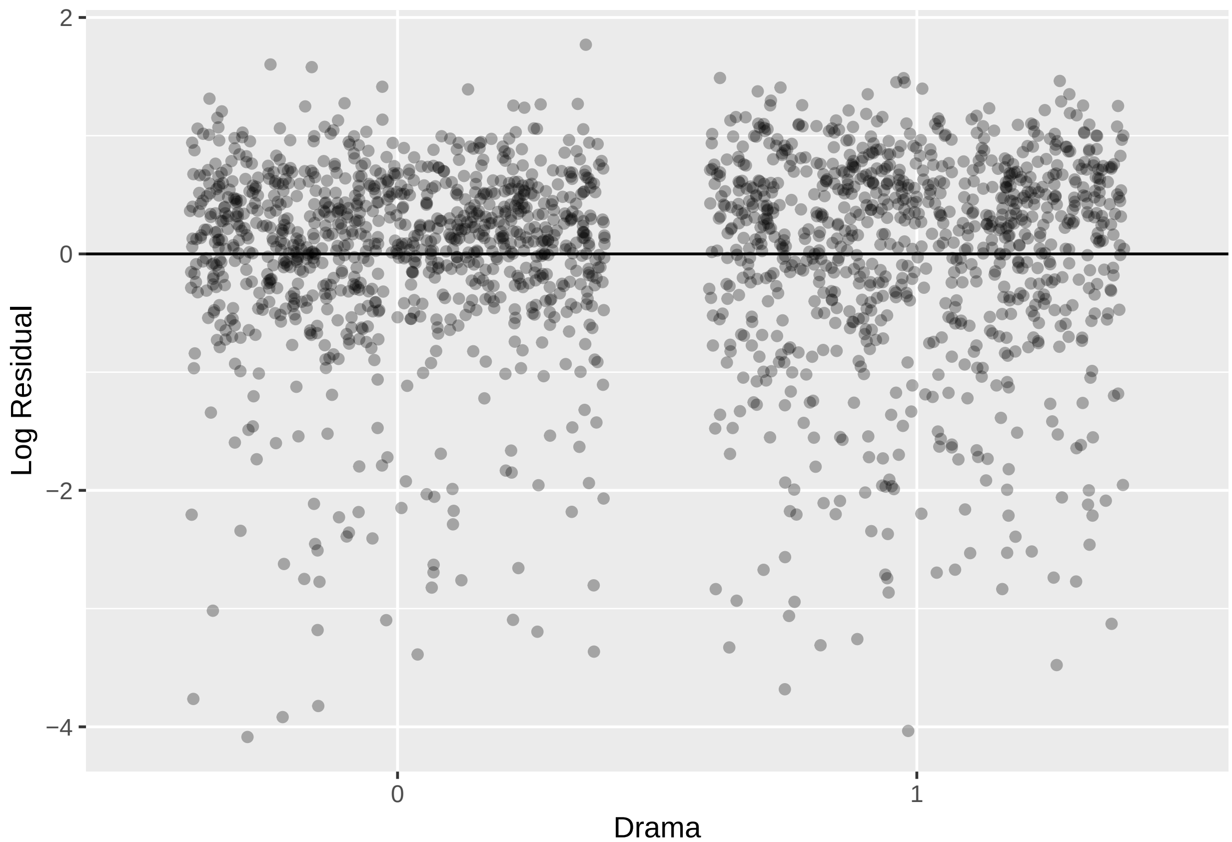
```r
# graph residuals against each variable included in the model
# most look random except Adventure
lapply(genre_xvar, function(var) {
  train_resid %>%
    ggplot() +
    geom_jitter(aes_string(var, y = 'lresid'), alpha = .3)
})
```

```r
# graph residuals against each genre not included in the model
lapply(names(train_genre_only %>% select(-genre_xvar)), function(var) {
  train_resid %>%
    ggplot() +
    geom_jitter(aes_string(var, y = 'lresid'), alpha = .3)
})
```

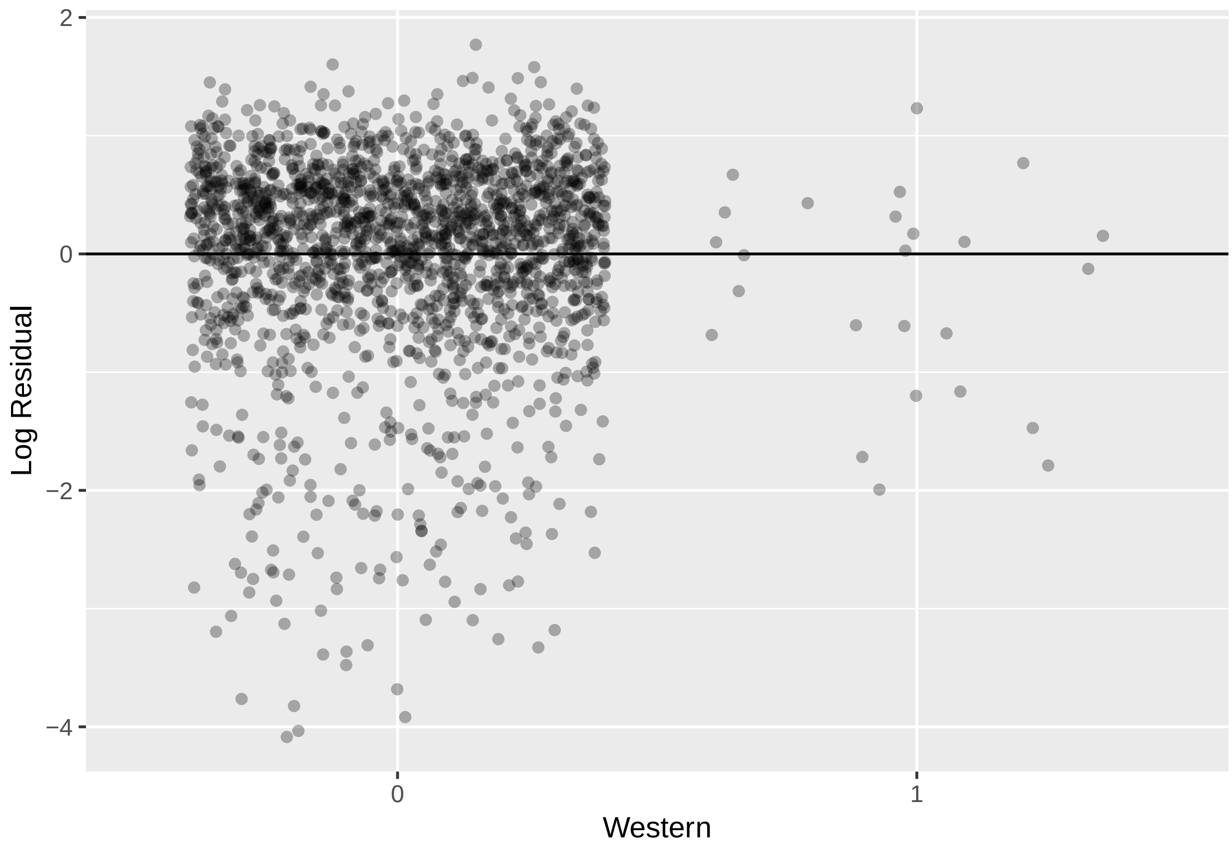
Display a couple of plots for presentation: some diversity. But show all around 0



2
0
−2
−4
Log Residual
0
1
Action

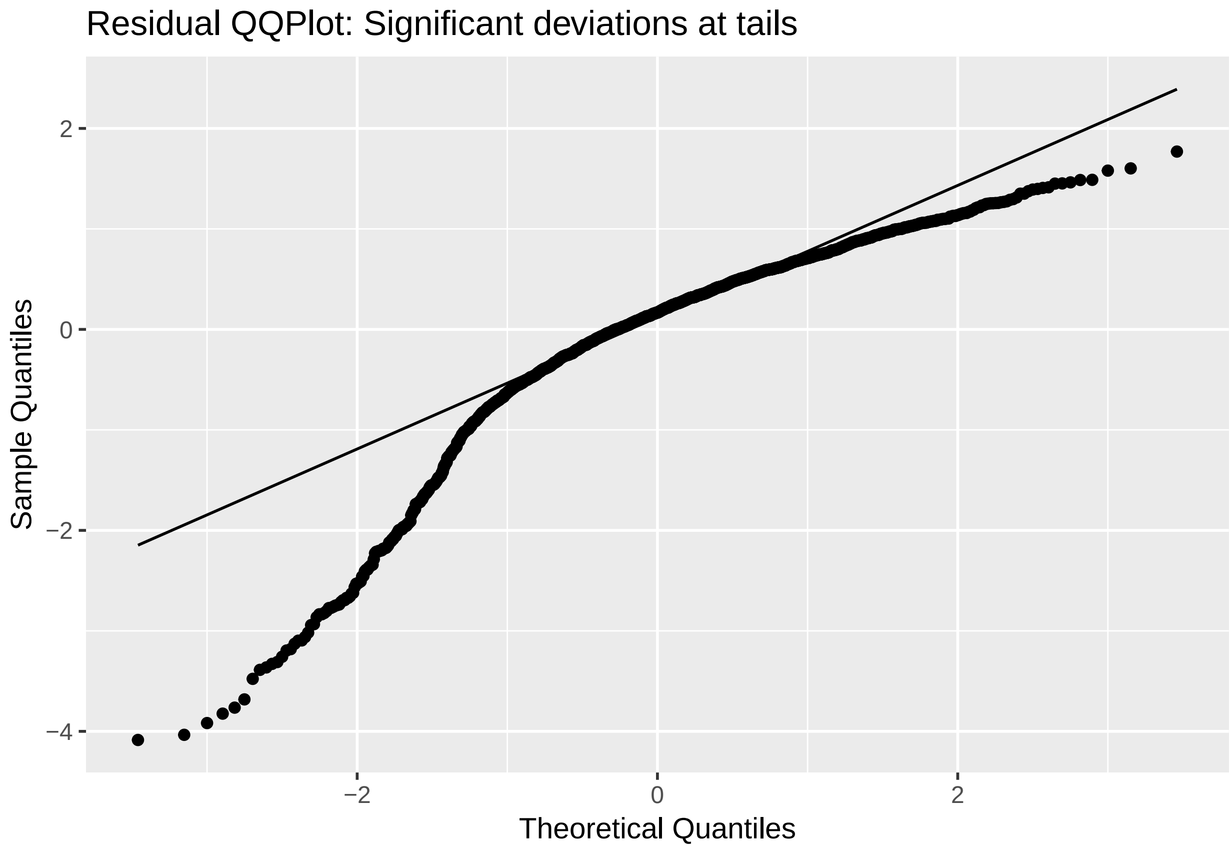


2
0
−2
−4
Log Residual
0
1
Drama



QQ-Plot Not great.

```
# residuals themselves are NOT normally distributed
# qq plot s
train_resid %>% ggplot(aes(sample = lresid)) +
  geom_qq() +
  geom_qq_line() +
  labs(title = 'Residual QQPlot: Significant deviations at tails',
       x = 'Theoretical Quantiles', y = 'Sample Quantiles')
```



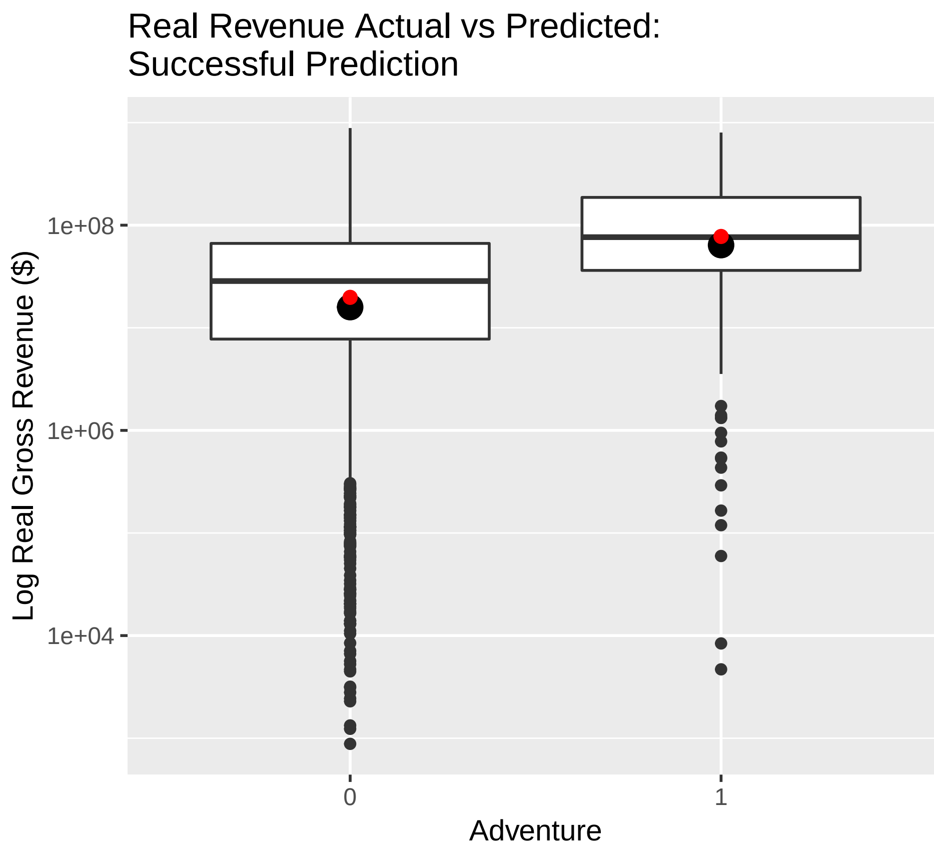
Plot Predictions

Plot prediction for mean real revenue against each genre included in the model.
AND genres not included in the model: still pretty good with some exceptions. Evidence that these genres are not useful for prediction/are covered by other genres.

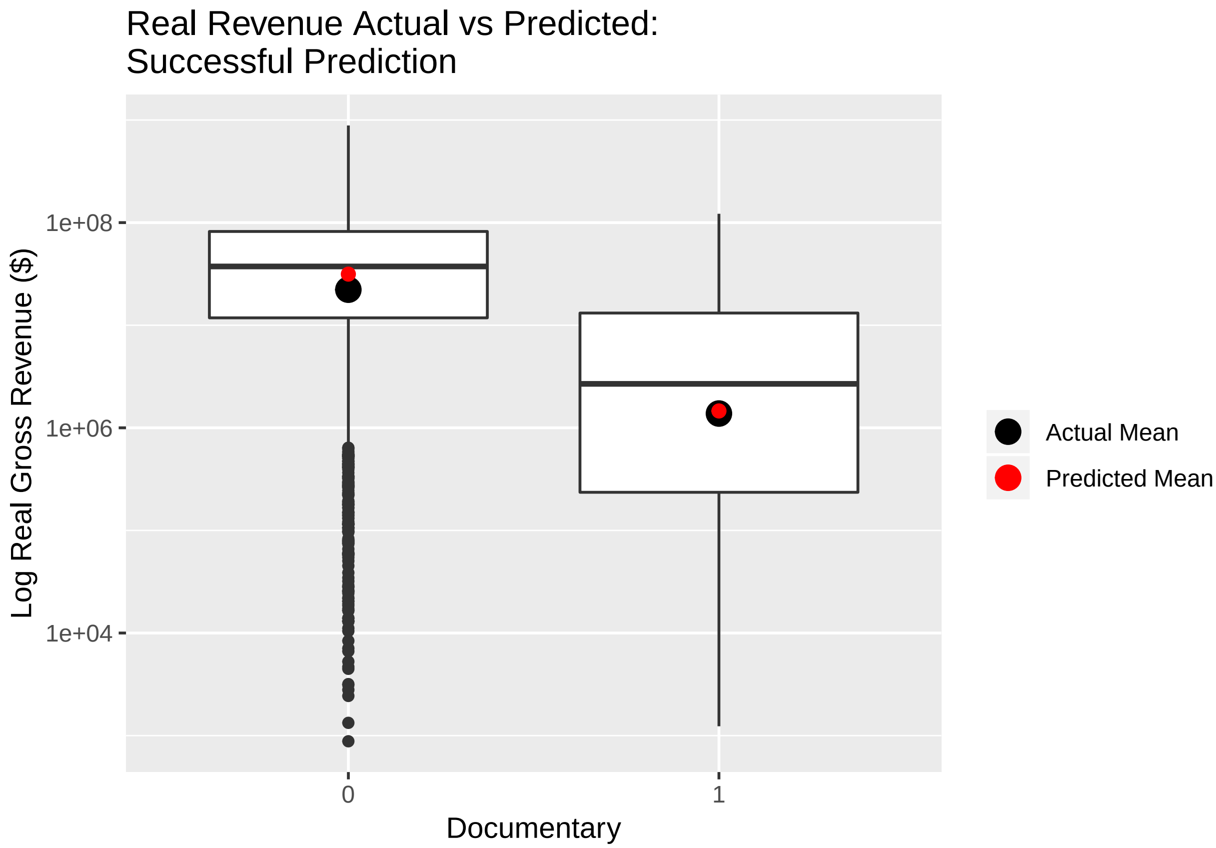
```
train_pred <- train %>%
  add_predictions(mod_genre, 'lpred') %>%
  mutate(pred = 10^lpred)

lapply(genre_xvar, function(var) {
  train_pred %>%
    ggplot(aes_string(x = var)) +
    geom_boxplot(aes(y = real_gross)) +
    # include mean
    stat_summary(aes(y = real_gross), fun.y = mean, geom = 'point', size = 4) +
    geom_point(data = train_pred %>% group_by(!!rlang::sym(var)) %>% summarise(mean = mean(pred)),
               aes(y = mean), color = 'red', size = 2) +
    scale_y_log10() +
    labs(y = 'Log Real Gross Revenue ($)', title = 'Real Revenue Actual vs Predicted\n Successful Predi
})
```

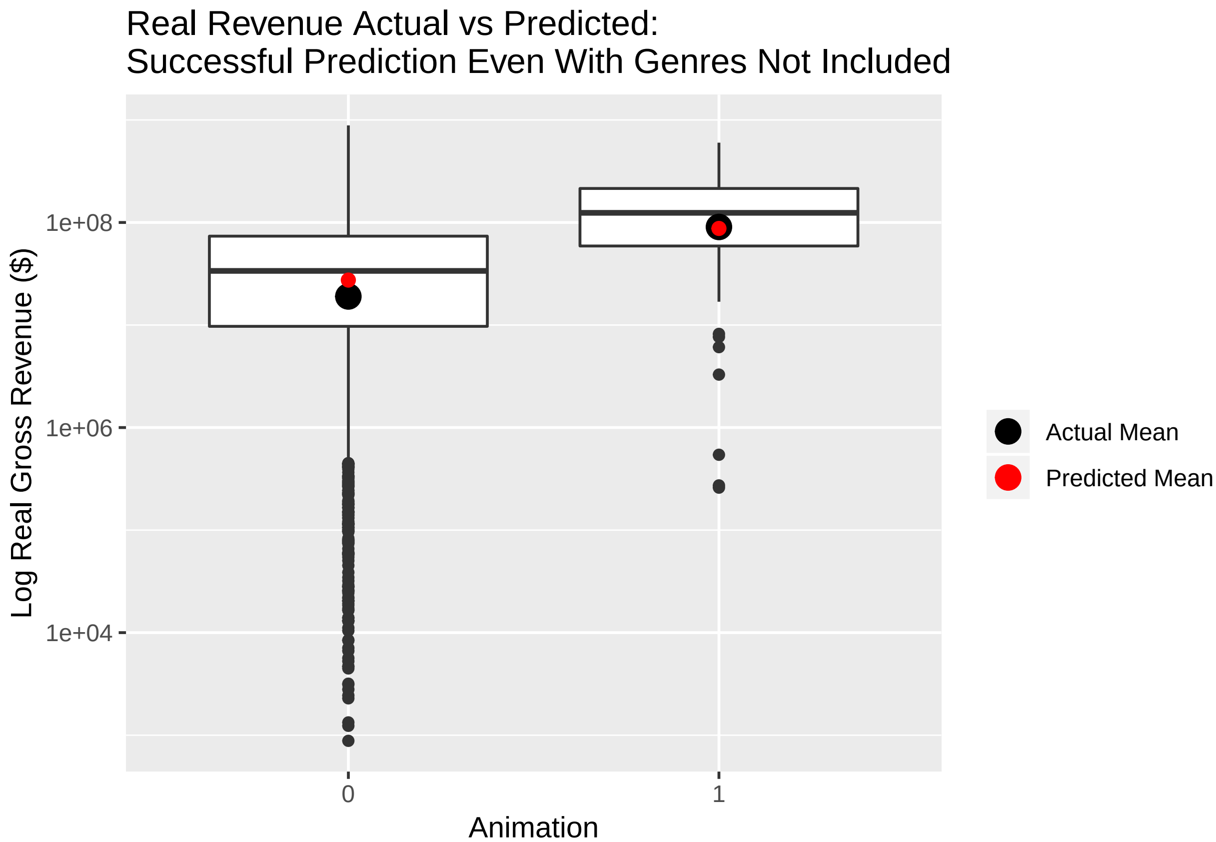
```
# predictions against other genres
lapply(names(train_genre_only %>% select(-genre_xvar)), function(var) {
  train_pred %>%
    ggplot(aes_string(x = var)) +
    geom_boxplot(aes(y = real_gross)) +
```

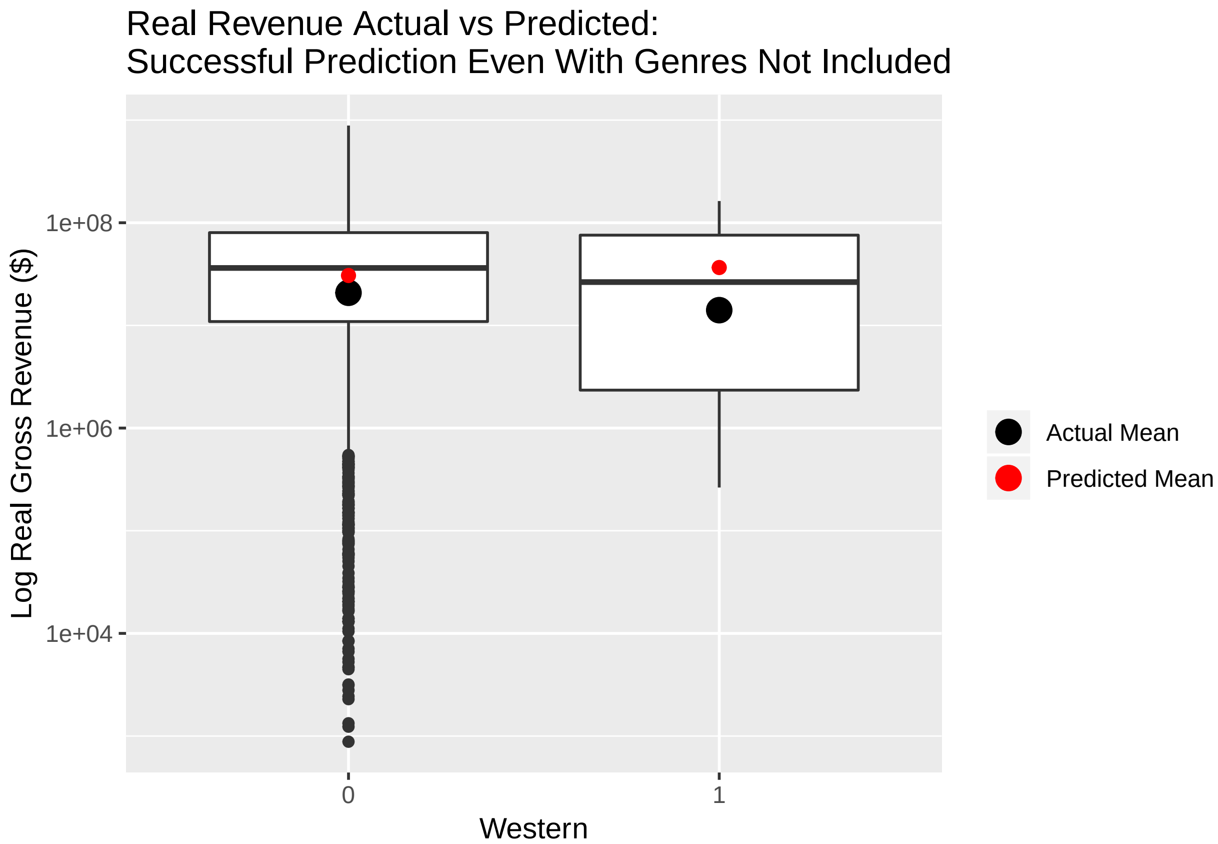
```
    # include mean
    stat_summary(aes(y = real_gross), fun.y = mean, geom = 'point', size = 4) +
    geom_point(data = train_pred %>% group_by(!!rlang::sym(var)) %>% summarise(mean = mean(pred)),
               aes(y = mean), color = 'red', size = 2) +
    scale_y_log10() +
    labs(y = 'Log Real Gross Revenue ($)', title = 'Real Revenue Actual vs Predicted:\nSuccessful Predi
})
```



Display a couple of plots for presentation.


Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
0
1
Documentary
Actual Mean
Predicted Mean

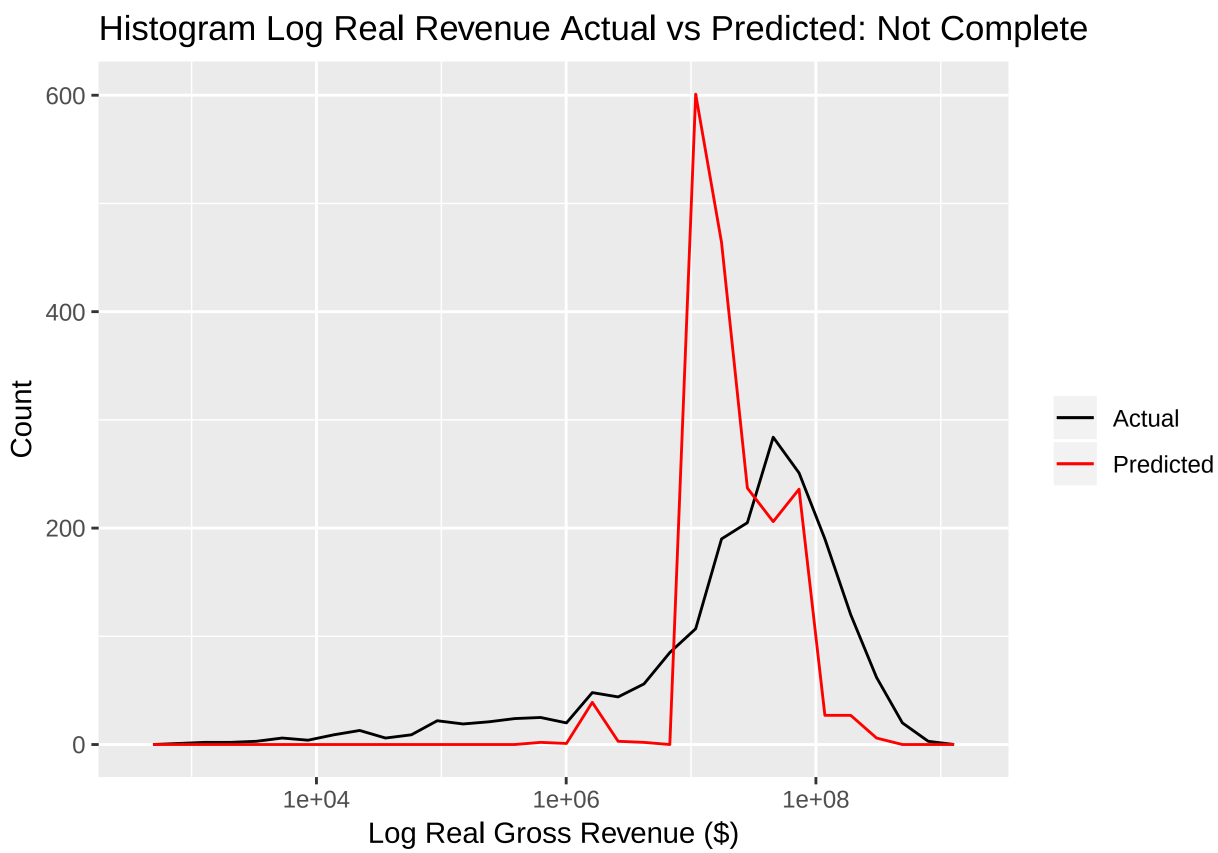

Real Revenue Actual vs Predicted:
Successful Prediction Even With Genres Not Included
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
0
1
Animation
Actual Mean
Predicted Mean



Overall predictions: clearly not enough to just specify genres

```
train %>%
  add_predictions(mod_genre, 'lpred') %>%
  mutate(pred = 10^lpred) %>%
  ggplot() +
  geom_freqpoly(aes(x = real_gross, color = 'Actual')) +
  geom_freqpoly(aes(x = pred, color = 'Predicted')) +
  scale_x_log10() +
  labs(x = 'Log Real Gross Revenue ($)', y = 'Count', title = 'Histogram Log Real Revenue Actual vs Pre
  scale_color_manual(name = '', values = c(Actual = 'black', Predicted = 'red'))
```

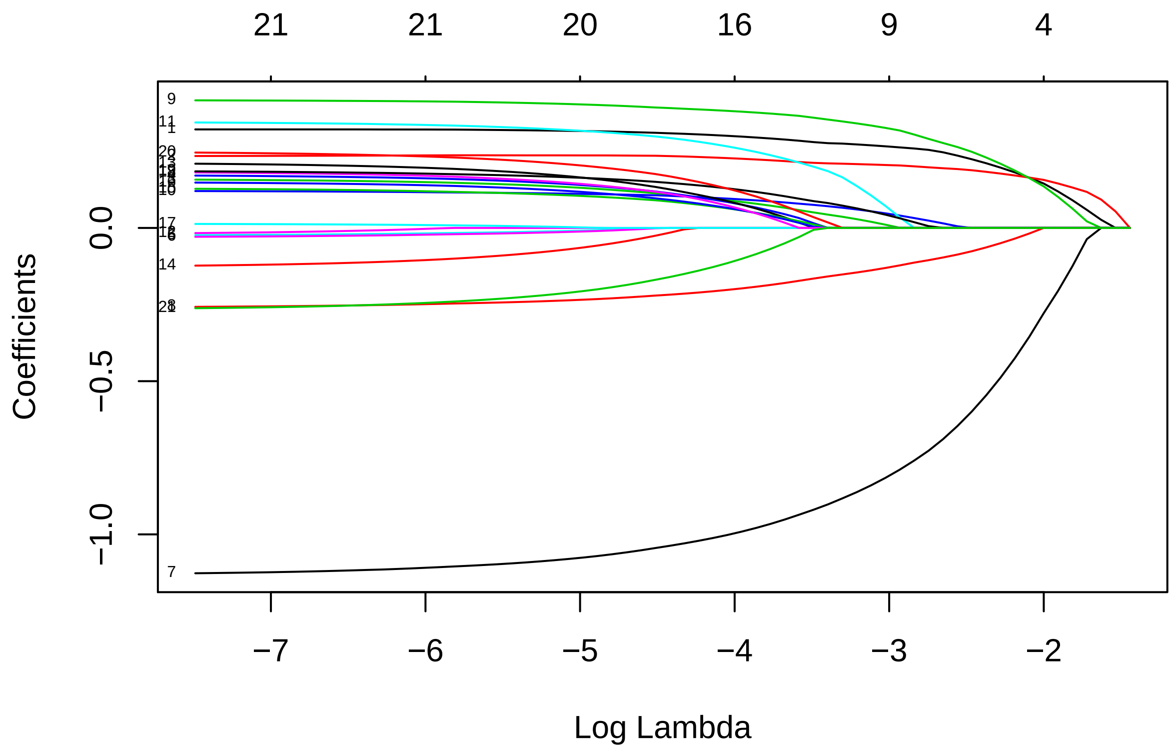
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



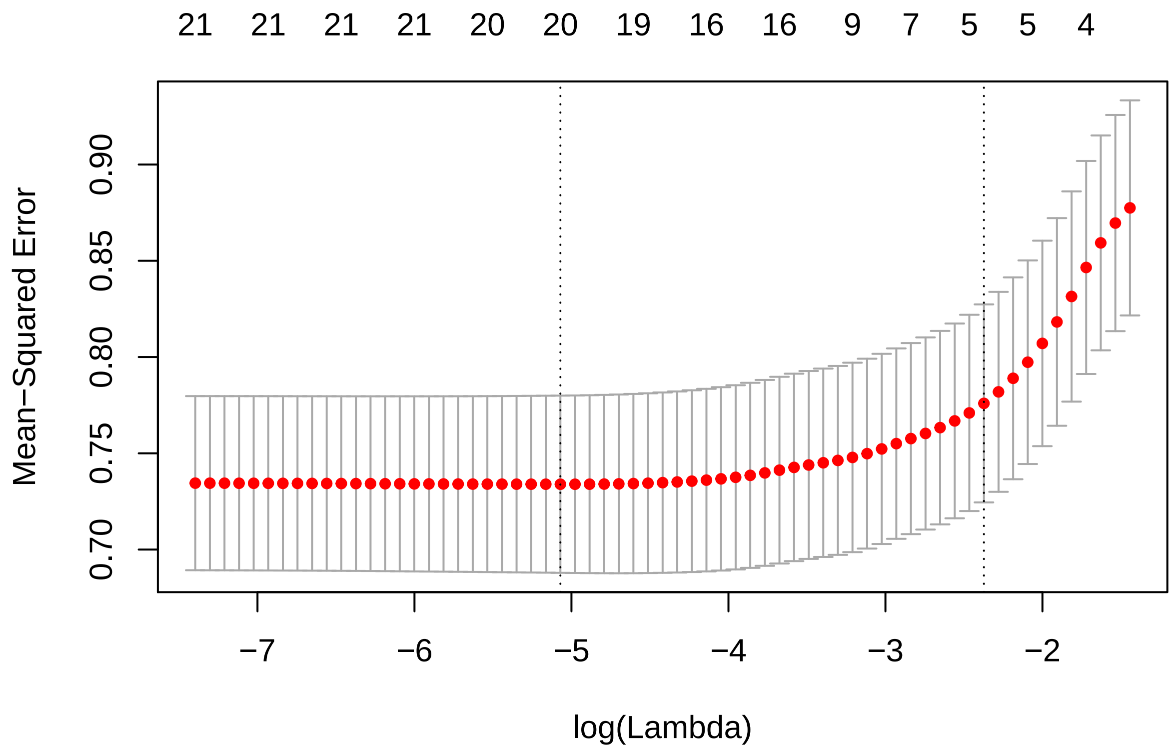
Glmnet: sparse

Try to eliminate most of the genre variables using glmnet and see if results are similar to stepwise. Can't do statistical tests, so not useful for analysis, but can use to aid justification.

```
# matrix of x and y variables
x <- as.matrix(train_genre_only %>% mutate_all(funs(as.numeric(as.character(.)))))
y <- as.matrix(train$real_gross_log)

# glmnet process form class
mod_sparse <- glmnet(x, y, family = 'gaussian')
plot(mod_sparse, xvar = 'lambda', label = TRUE)
```




```
mod_sparse <- cv.glmnet(x, y)
plot(mod_sparse)
```




```r
coef(mod_sparse, s = 'lambda.min') # use min lambda
```

```
## 22 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  7.158319467
## Action       0.316866270
## Adventure    0.236784610
## Animation    0.133081838
## Biography    0.142672310
## Comedy      -0.009843859
## Crime       -0.012866133
## Documentary -1.080069143
## Drama       -0.236062771
## Family       0.404262630
## Fantasy      0.111824039
## History      0.319273704
## Horror       .
## Music        0.168382172
## Musical     -0.069374568
## Mystery      0.106229872
## Romance      0.120292330
## SciFi        0.002684535
## Sport        0.146848534
## Thriller     0.165198422
## War          0.207283391
## Western     -0.211255187
```

```r
coef(mod_sparse, s = 'lambda.1se') # use most sparse
```

```
## 22 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  7.24955074
## Action       0.21115913
## Adventure    0.18312646
## Animation    .
## Biography    .
## Comedy       .
## Crime        .
## Documentary -0.54632201
## Drama       -0.06351752
## Family       0.22984009
## Fantasy      .
## History      .
## Horror       .
## Music        .
## Musical      .
## Mystery      .
## Romance      .
## SciFi        .
## Sport        .
## Thriller     .
## War          .
## Western      .
```

Use genre model as a base

Next, we started with the genre model and added in additional variables: clear that genre is not sufficient. Several steps:

- Plot other variables (budget, facebook likes, total number of Oscars, imdb score, number of oscars, content rating, year) against the residuals from the genre model.
  - All of these have somewhat non-random relationships with residuals. Especially budget and IMDB score. Movies with higher budgets make more revenue than predicted by genre (positive residual)
- Based on these non-random relationships, used them all in step wise, but started with the genre variables from the genre model as a base.
  - Log budget and IMDB score had the strongest relationships with log revenue in EDA. Facebook likes and Oscars had especially weak relationships. Thus the final variables included are not surprising.
- Looked at new residuals of included and excluded relationships. All fairly random.
- Looked at predictions for each individual variable. Good job even for variables not included in the model.
  - Note that these predictions are not simple lines based on one coefficient because each movie's predicted revenue is based on several variables.

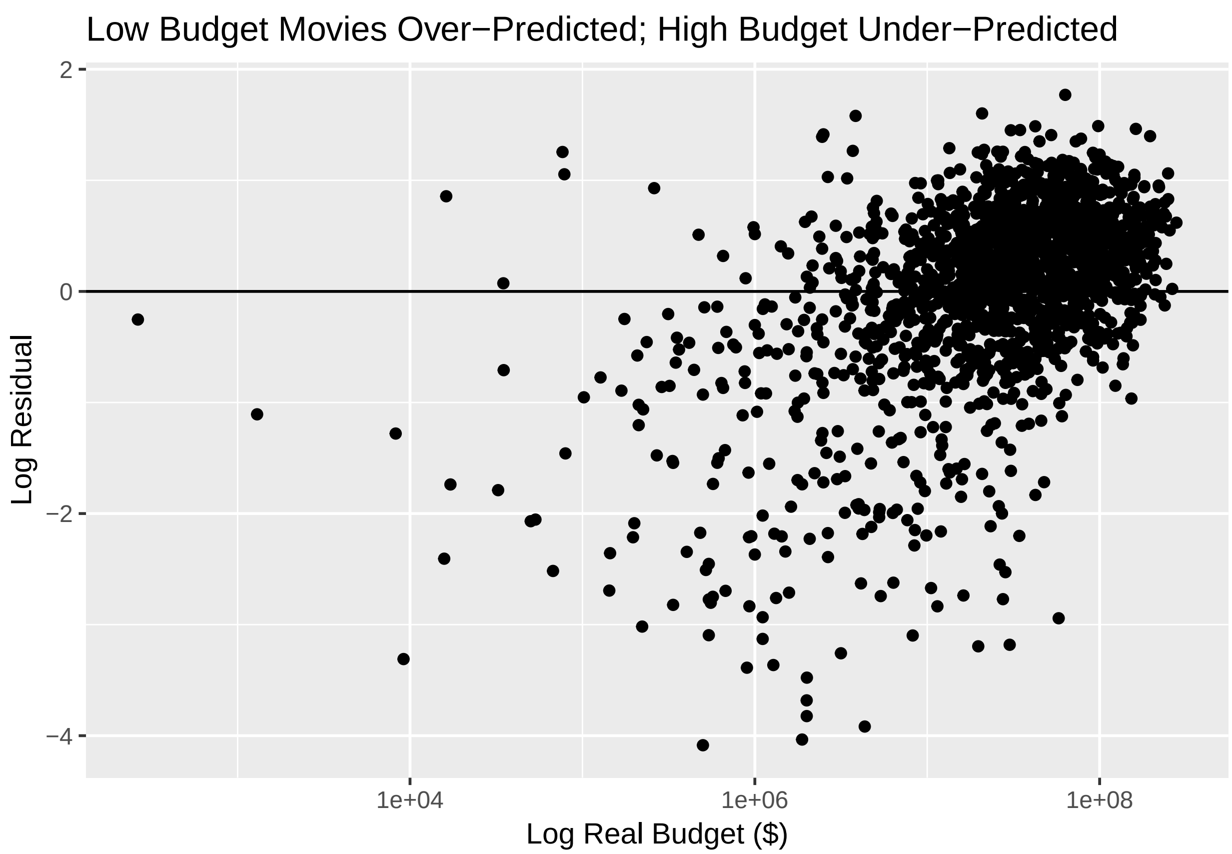
  - Year isn't great. But in most years, other factors would be more important. But some years, such as recessions, it is very important which is why it is important to include.
- Looked at predictions overall. Looks much better.

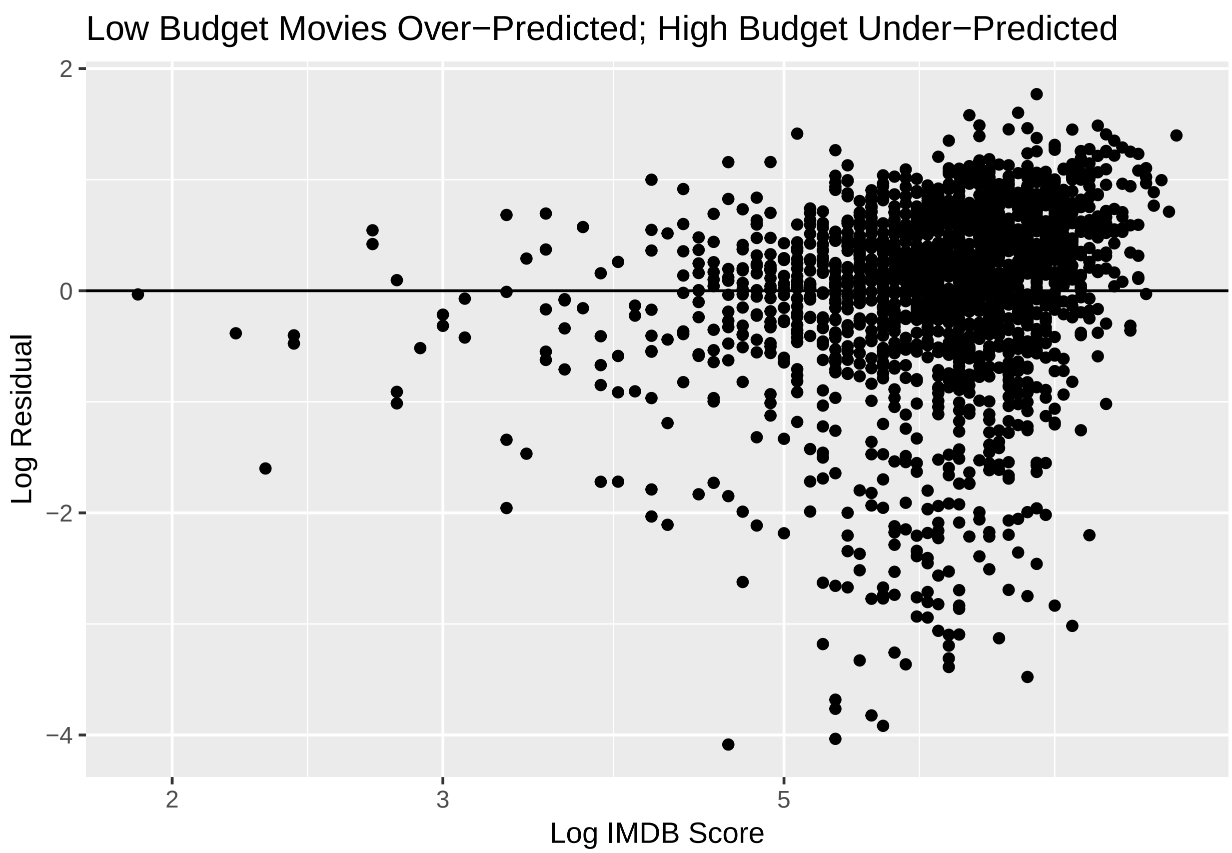
Overall, this is a pretty good model. Prediction is pretty good. Residuals are more normal. A lot of the variables are not significant, but they are significant when considered together in anova.
However, still not convinced this is the best model. I worry that it is overfitted and the genre variables are

not necessay. Maybe it is not a valid assumption that we should start with genre, especially since a lot did come in as insignificant. Perhaps these effects are being captured by other variables, so we should not start with the best assumption that we include genre.

```
train_resid <- train %>%
  add_residuals(mod_genre, 'lresid')

# graph each against log residual: continuous (log scale)
lapply(c('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes',
         'imdb_score'), function(var) {
  train_resid %>%
    ggplot() +
    geom_point(aes_string(str_c(var, '_log'), y = 'lresid'))
    #labs('Low Budget Movies Over-Predicted; High Budget Under-Predicted')
})
```

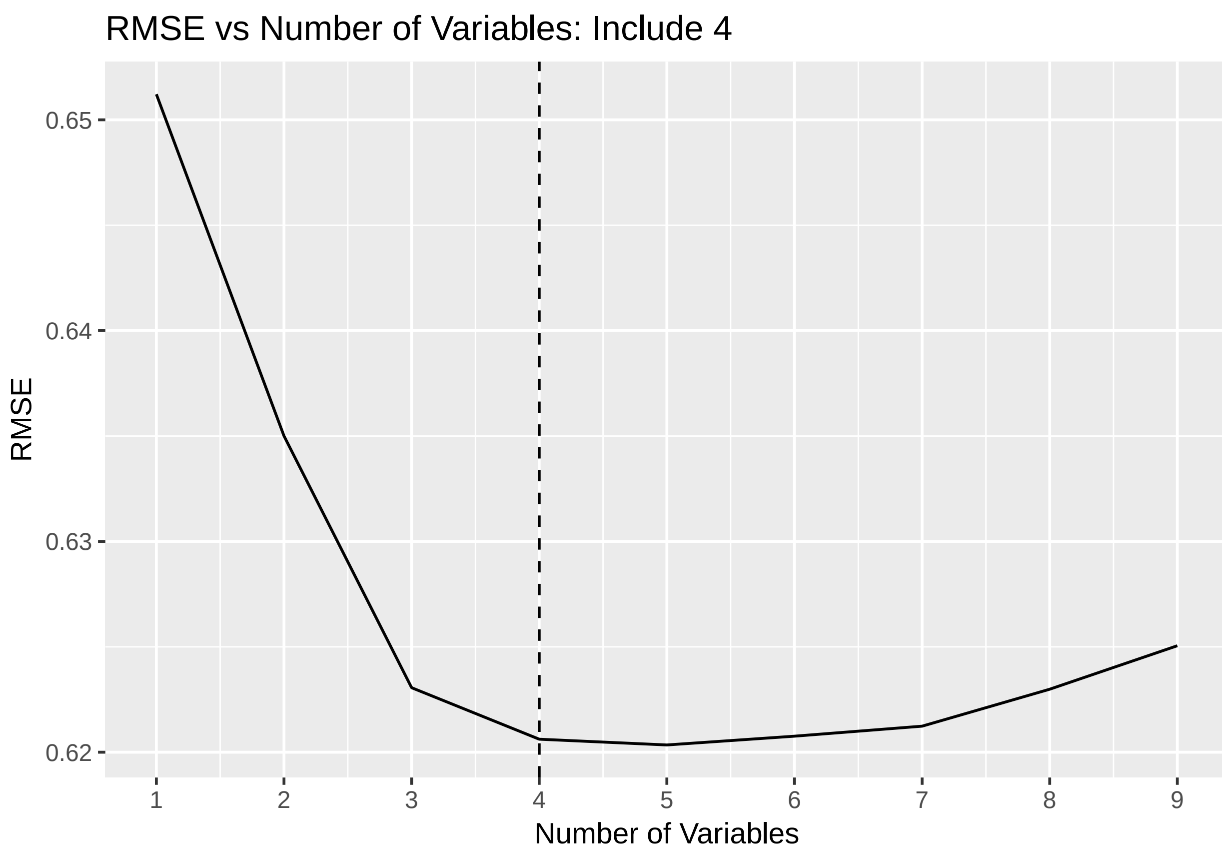
```
# categorical
# can't log categorical variables
lapply(c('content_rating', 'year', 'total_oscars_actor', 'total_oscars_director'), function(var) {
  train_resid %>%
    filter(!is.na(!!rlang::sym(var))) %>%
    ggplot() +
    geom_jitter(aes_string(var, 'lresid'), alpha = .3)
})
```






```r
# For some log, need to turn -Inf from log(0) to NA
train <- train %>%
  mutate_at(vars(contains('log')), funs(ifelse(is.infinite(.), NA, .)))
valid <- valid %>%
  mutate_at(vars(contains('log')), funs(ifelse(is.infinite(.), NA, .)))

# starting formula: genre
starting_formula = 'Adventure + Action + Family + Mystery + Documentary + Drama + History + Romance'

# stepwise starting with genre
rmse_lst <- step_wise_loop(df = train %>% select(genre_xvar, content_rating, real_budget, year,
                                                 total_oscars_actor, total_oscars_director,
                                                 imdb_score_log, real_budget_log,
                                                 director_facebook_likes_log,
                                                 cast_total_facebook_likes_log),
                           starting_vars = genre_xvar,
                           starting_formula = starting_formula)
```

```r
# graph RMSE vs number of variables
fit_rmse <- tibble(nvar = 1:length(rmse_lst),
                   rmse = rmse_lst)
ggplot(fit_rmse, aes(x = nvar, y = rmse)) + geom_line() +
  scale_x_continuous(breaks = seq(1, length(rmse_lst), by = 1)) +
  geom_vline(xintercept = 4, linetype = 'dashed') +
  labs(x = 'Number of Variables', y = 'RMSE',
       title = 'RMSE vs Number of Variables: Include 4')
```




```r
# after var 4, decreases too small or increase

# model with extra 4 variables
mod_all <- lm(real_gross_log ~ Adventure + Action + Family + Mystery +
                  Documentary + Drama + History + Romance  +
                  real_budget_log + imdb_score_log + content_rating + year,
          data = train)

summary(mod_all)
```

```
##
## Call:
## lm(formula = real_gross_log ~ Adventure + Action + Family + Mystery +
##     Documentary + Drama + History + Romance + real_budget_log +
##     imdb_score_log + content_rating + year, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4485 -0.2238  0.0878  0.3407  3.3377
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -0.016486   0.394298  -0.042  0.96665
## Adventure1            -0.142890   0.045332  -3.152  0.00165 **
## Action1               -0.011306   0.040640  -0.278  0.78090
## Family1                0.311938   0.079689   3.914 9.43e-05 ***
```

```
## Mystery1             0.042585   0.051695   0.824  0.41019
## Documentary1         0.163354   0.141606   1.154  0.24884
## Drama1              -0.257951   0.034509  -7.475 1.24e-13 ***
## History1            -0.017739   0.089809  -0.198  0.84345
## Romance1             0.015177   0.037044   0.410  0.68208
## real_budget_log      0.812038   0.029073  27.931  < 2e-16 ***
## imdb_score_log       2.160658   0.206270  10.475  < 2e-16 ***
## content_ratingNC-17 -0.036289   0.275191  -0.132  0.89510
## content_ratingPG    -0.035081   0.119686  -0.293  0.76948
## content_ratingPG-13  0.139695   0.137054   1.019  0.30822
## content_ratingR     -0.052452   0.136983  -0.383  0.70184
## year1981            -0.209854   0.368926  -0.569  0.56955
## year1982             0.182927   0.339404   0.539  0.58998
## year1983             0.247796   0.384325   0.645  0.51917
## year1984             0.500267   0.329092   1.520  0.12866
## year1985             0.288020   0.339707   0.848  0.39665
## year1986             0.065028   0.324926   0.200  0.84140
## year1987             0.172547   0.316476   0.545  0.58568
## year1988             0.147460   0.310528   0.475  0.63494
## year1989             0.279267   0.305019   0.916  0.36002
## year1990             0.062553   0.306795   0.204  0.83846
## year1991             0.044401   0.310206   0.143  0.88620
## year1992            -0.007279   0.315891  -0.023  0.98162
## year1993             0.025560   0.307311   0.083  0.93372
## year1994            -0.222206   0.301314  -0.737  0.46095
## year1995            -0.050133   0.293633  -0.171  0.86445
## year1996            -0.217635   0.286206  -0.760  0.44712
## year1997            -0.171164   0.286195  -0.598  0.54988
## year1998            -0.272722   0.285619  -0.955  0.33980
## year1999            -0.270022   0.282260  -0.957  0.33889
## year2000            -0.173448   0.283525  -0.612  0.54078
## year2001            -0.287595   0.281570  -1.021  0.30721
## year2002            -0.262480   0.281560  -0.932  0.35135
## year2003            -0.250251   0.282916  -0.885  0.37653
## year2004            -0.266357   0.282475  -0.943  0.34585
## year2005            -0.255937   0.281463  -0.909  0.36332
## year2006            -0.346753   0.281963  -1.230  0.21895
## year2007            -0.347511   0.283316  -1.227  0.22015
## year2008            -0.374356   0.281500  -1.330  0.18375
## year2009            -0.371088   0.280547  -1.323  0.18611
## year2010            -0.402398   0.281138  -1.431  0.15253
## year2011            -0.279867   0.283485  -0.987  0.32367
## year2012            -0.184045   0.281991  -0.653  0.51406
## year2013            -0.120662   0.281346  -0.429  0.66807
## year2014            -0.154492   0.283282  -0.545  0.58558
## year2015            -0.267933   0.285375  -0.939  0.34793
## year2016            -0.176387   0.301510  -0.585  0.55862
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6071 on 1668 degrees of freedom
##   (132 observations deleted due to missingness)
## Multiple R-squared:  0.495,  Adjusted R-squared:  0.4799
## F-statistic: 32.71 on 50 and 1668 DF,  p-value: < 2.2e-16
```

```r
rmse(mod_all, data = valid)
```

```
## [1] 0.6206177
```

```r
# when consider the factors as one variable, they are significant
anova(mod_all)
```

```
## Analysis of Variance Table
##
## Response: real_gross_log
##                   Df Sum Sq Mean Sq  F value    Pr(>F)
## Adventure          1  75.64   75.64 205.2075 < 2.2e-16 ***
## Action             1  24.10   24.10  65.3761 1.178e-15 ***
## Family             1  32.07   32.07  86.9917 < 2.2e-16 ***
## Mystery            1   2.40    2.40   6.5082 0.0108265 *
## Documentary        1   7.70    7.70  20.8945 5.210e-06 ***
## Drama              1  14.60   14.60  39.6072 3.960e-10 ***
## History            1   5.95    5.95  16.1431 6.134e-05 ***
## Romance            1   1.86    1.86   5.0542 0.0246965 *
## real_budget_log    1 347.64  347.64 943.0834 < 2.2e-16 ***
## imdb_score_log     1  46.38   46.38 125.8221 < 2.2e-16 ***
## content_rating     4   8.49    2.12   5.7600 0.0001328 ***
## year              36  35.96    1.00   2.7101 2.504e-07 ***
## Residuals       1668 614.85    0.37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# number of observations
nobs(mod_all)
```

```
## [1] 1719
```

```r
stargazer(mod_all, type = 'latex', out = 'C:/Users/Gimli/Documents/DS5110/project/output/all_mod.tex',
          no.space = T, nobs = T, dep.var.caption = '', align = T, dep.var.labels = '',
          title = 'Extended Genre Model Summary \\\\ Response Variable: Log Real Gross Revenue',
          omit = c('1981', '1982', '1983', '1984', '1985', '1986', '1987', '1988', '1989', '1990', '199
          '1994', '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005
          '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016'),
          covariate.labels = c('Adventure', 'Action', 'Family', 'Mystery', 'Documentary', 'Drama', 'His
                               'Log Real Budget', 'Log IMDB Score', 'NC17', 'PG', 'PG13', 'R'),
          notes = 'Year dummies omitted', notes.align = 'l', column.sep.width = "-1pt")
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Wed, Apr 10, 2019 - 6:10:32 PM
## % Requires LaTeX packages: dcolumn
## \begin{table}[!htbp] \centering
##   \caption{Extended Genre Model Summary \\ Response Variable: Log Real Gross Revenue}
##   \label{}
## \begin{tabular}{@{\extracolsep{-1pt}}lD{.}{.}{-3} }
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## \\[-1.8ex] & \multicolumn{1}{c}{} \\
## \hline \\[-1.8ex]
##  Adventure & -0.143^{***} \\
##   & (0.045) \\
```

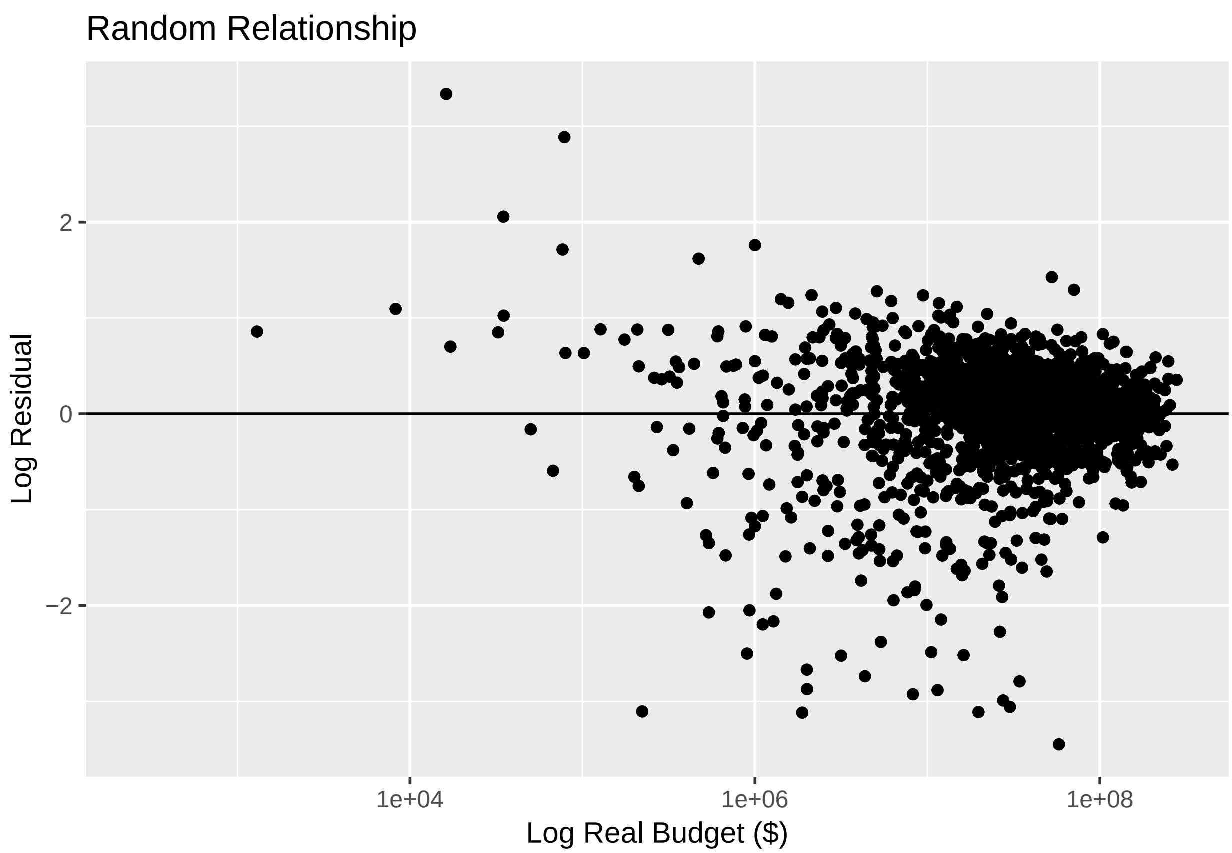
```
##   Action & -0.011 \\
##   & (0.041) \\
##   Family & 0.312^{***} \\
##   & (0.080) \\
##   Mystery & 0.043 \\
##   & (0.052) \\
##   Documentary & 0.163 \\
##   & (0.142) \\
##   Drama & -0.258^{***} \\
##   & (0.035) \\
##   History & -0.018 \\
##   & (0.090) \\
##   Romance & 0.015 \\
##   & (0.037) \\
##   Log Real Budget & 0.812^{***} \\
##   & (0.029) \\
##   Log IMDB Score & 2.161^{***} \\
##   & (0.206) \\
##   NC17 & -0.036 \\
##   & (0.275) \\
##   PG & -0.035 \\
##   & (0.120) \\
##   PG13 & 0.140 \\
##   & (0.137) \\
##   R & -0.052 \\
##   & (0.137) \\
##   Constant & -0.016 \\
##   & (0.394) \\
##  \hline \\[-1.8ex]
## Observations & \multicolumn{1}{c}{1,719} \\
## R$^{2}$ & \multicolumn{1}{c}{0.495} \\
## Adjusted R$^{2}$ & \multicolumn{1}{c}{0.480} \\
## Residual Std. Error & \multicolumn{1}{c}{0.607 (df = 1668)} \\
## F Statistic & \multicolumn{1}{c}{32.706$^{***}$ (df = 50; 1668)} \\
## \hline
## \hline \\[-1.8ex]
## \textit{Note:}  & \multicolumn{1}{l}{$^{*}$p$<$0.1; $^{**}$p$<$0.05; $^{***}$p$<$0.01} \\
##  & \multicolumn{1}{l}{Year dummies omitted} \\
## \end{tabular}
## \end{table}
```

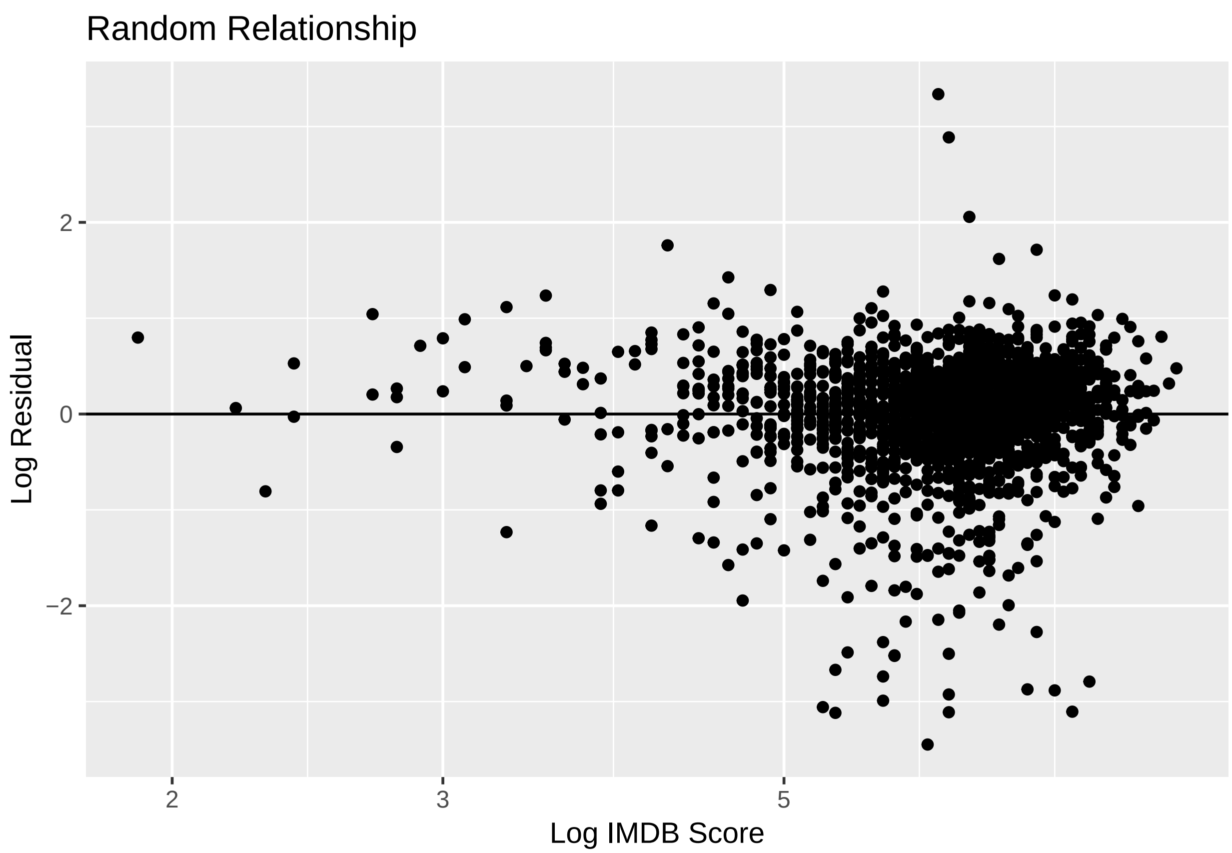
```r
train_labels <- train %>%
  rename(`Log Real Budget ($)` = real_budget_log,
         `Content Rating` = content_rating,
         `Log IMDB Score` = imdb_score_log,
         Year = year)
mod_all_labels <- lm(real_gross_log ~ Adventure + Action + Family + Mystery +
                   Documentary + Drama + History + Romance  +
                   `Log Real Budget ($)` + `Log IMDB Score` + `Content Rating` + Year,
          data = train_labels)
stargazer(anova(mod_all_labels), type = 'latex', out = 'C:/Users/Gimli/Documents/DS5110/project/output/a
          no.space = T, nobs = T, dep.var.caption = '', align = T, dep.var.labels = '',
          title = 'Extended Genre Model Anova \\\\ Response Variable: Log Real Gross Revenue', summary =
```

```
##
```

```
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Wed, Apr 10, 2019 - 6:10:33 PM
## % Requires LaTeX packages: dcolumn
## \begin{table}[!htbp] \centering
##   \caption{Extended Genre Model Anova \\ Response Variable: Log Real Gross Revenue}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## \multicolumn{1}{c}{} & \multicolumn{1}{c}{Df} & \multicolumn{1}{c}{Sum Sq} & \multicolumn{1}{c}{Mean
## \hline \\[-1.8ex]
## \multicolumn{1}{c}{Adventure} & 1 & 75.643 & 75.643 & 205.208 & 0 \\
## \multicolumn{1}{c}{Action} & 1 & 24.099 & 24.099 & 65.376 & 0 \\
## \multicolumn{1}{c}{Family} & 1 & 32.066 & 32.066 & 86.992 & 0 \\
## \multicolumn{1}{c}{Mystery} & 1 & 2.399 & 2.399 & 6.508 & 0.011 \\
## \multicolumn{1}{c}{Documentary} & 1 & 7.702 & 7.702 & 20.895 & 0.00001 \\
## \multicolumn{1}{c}{Drama} & 1 & 14.600 & 14.600 & 39.607 & 0 \\
## \multicolumn{1}{c}{History} & 1 & 5.951 & 5.951 & 16.143 & 0.0001 \\
## \multicolumn{1}{c}{Romance} & 1 & 1.863 & 1.863 & 5.054 & 0.025 \\
## \multicolumn{1}{c}{`Log Real Budget (\$)`} & 1 & 347.635 & 347.635 & 943.083 & 0 \\
## \multicolumn{1}{c}{`Log IMDB Score`} & 1 & 46.380 & 46.380 & 125.822 & 0 \\
## \multicolumn{1}{c}{`Content Rating`} & 4 & 8.493 & 2.123 & 5.760 & 0.0001 \\
## \multicolumn{1}{c}{Year} & 36 & 35.963 & 0.999 & 2.710 & 0.00000 \\
## \multicolumn{1}{c}{Residuals} & 1,668 & 614.850 & 0.369 &  &  \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

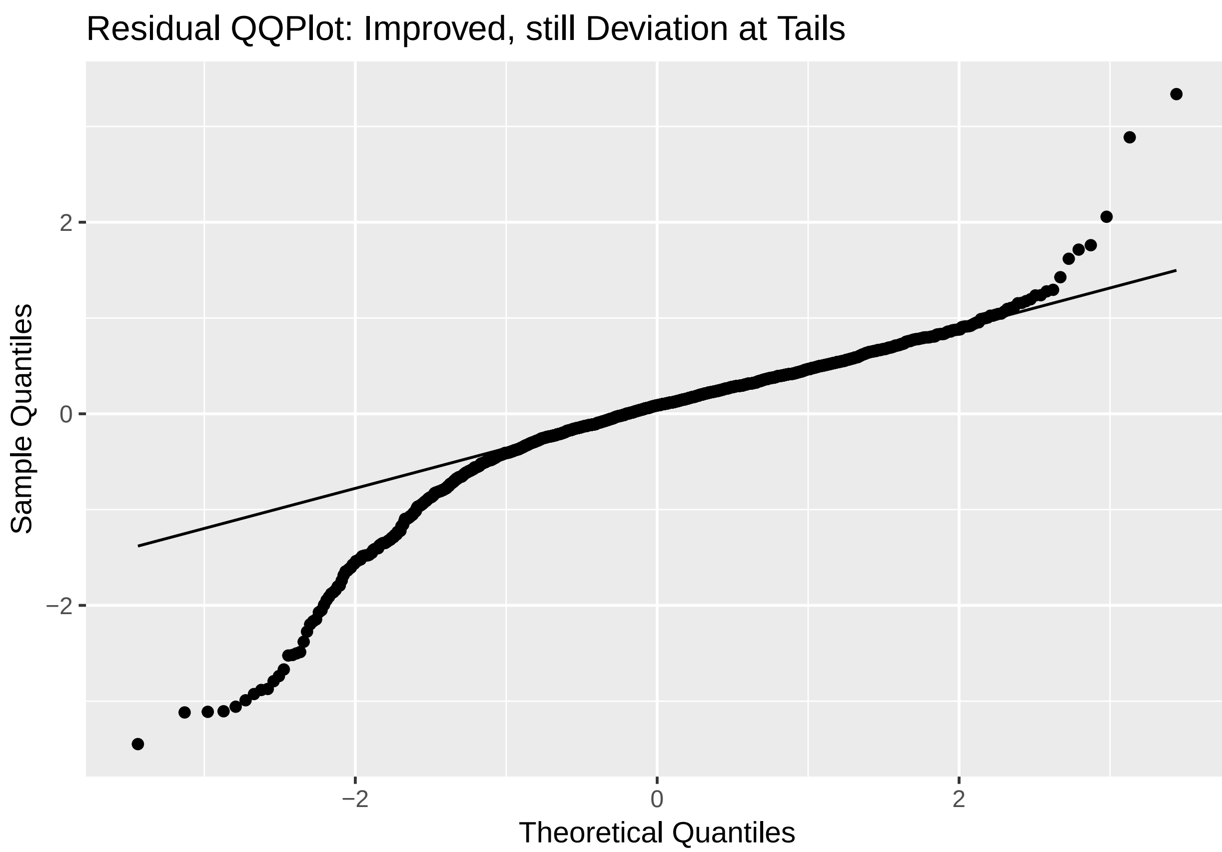
```r
gr_resid(mod_all)
```


Random Relationship
2
0
−2
Log Residual
1e+04
1e+06
1e+08
Log Real Budget ($)



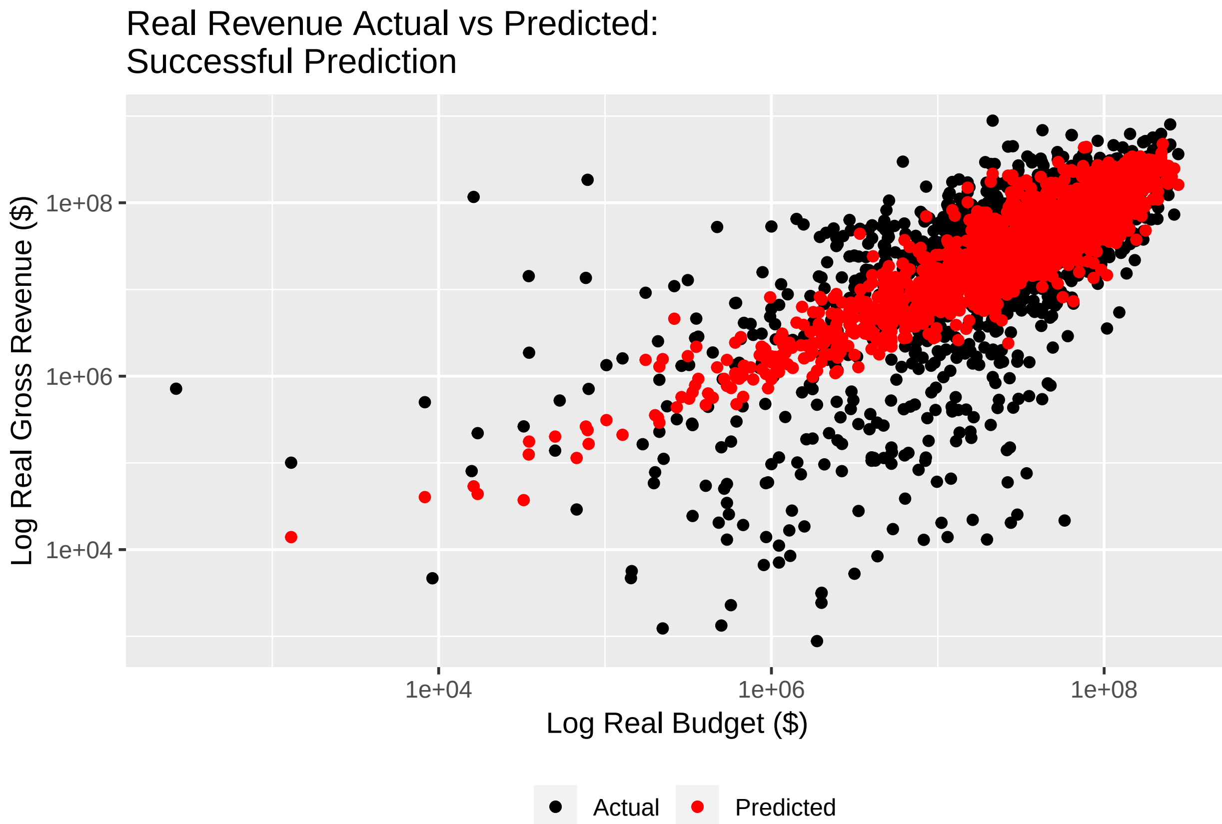

```
train %>%
  add_residuals(mod_all,'lresid') %>%
  ggplot(aes(sample = lresid)) +
    geom_qq() +
    geom_qq_line() +
    labs(title = 'Residual QQPlot: Improved, still Deviation at Tails',
       x = 'Theoretical Quantiles', y = 'Sample Quantiles')
```

```
## Warning: Removed 132 rows containing non-finite values (stat_qq).

## Warning: Removed 132 rows containing non-finite values (stat_qq_line).
```

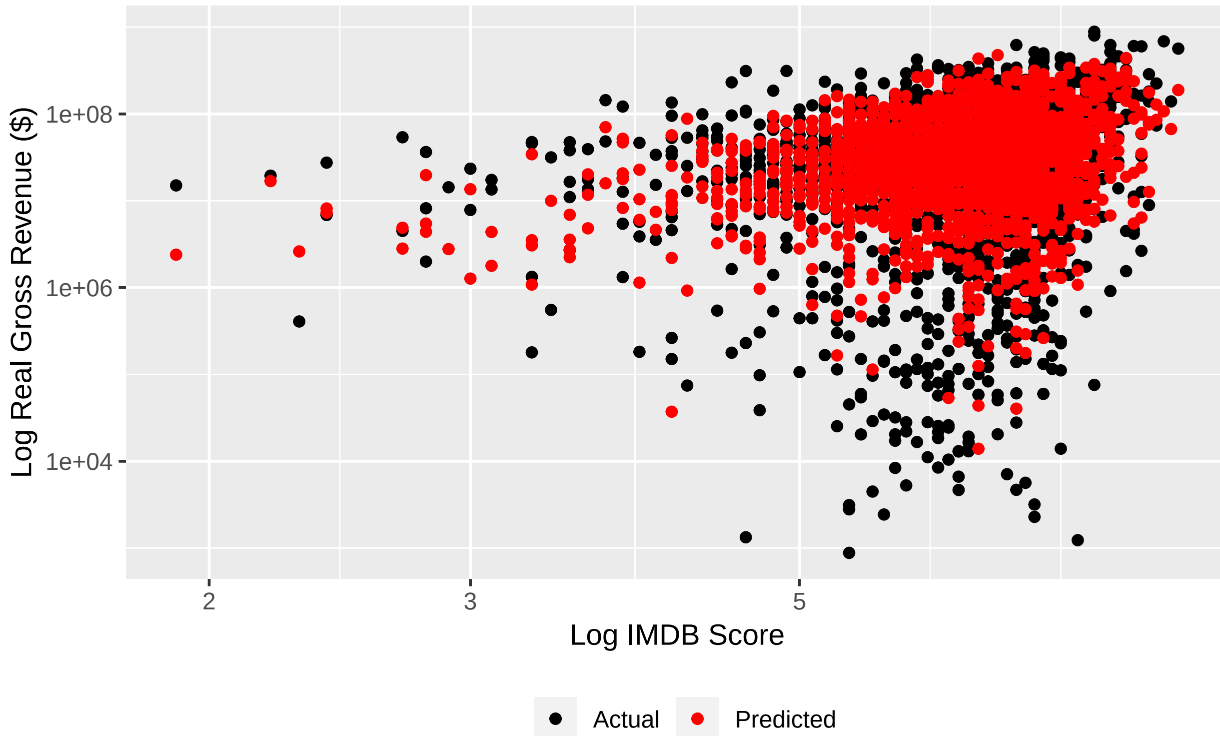


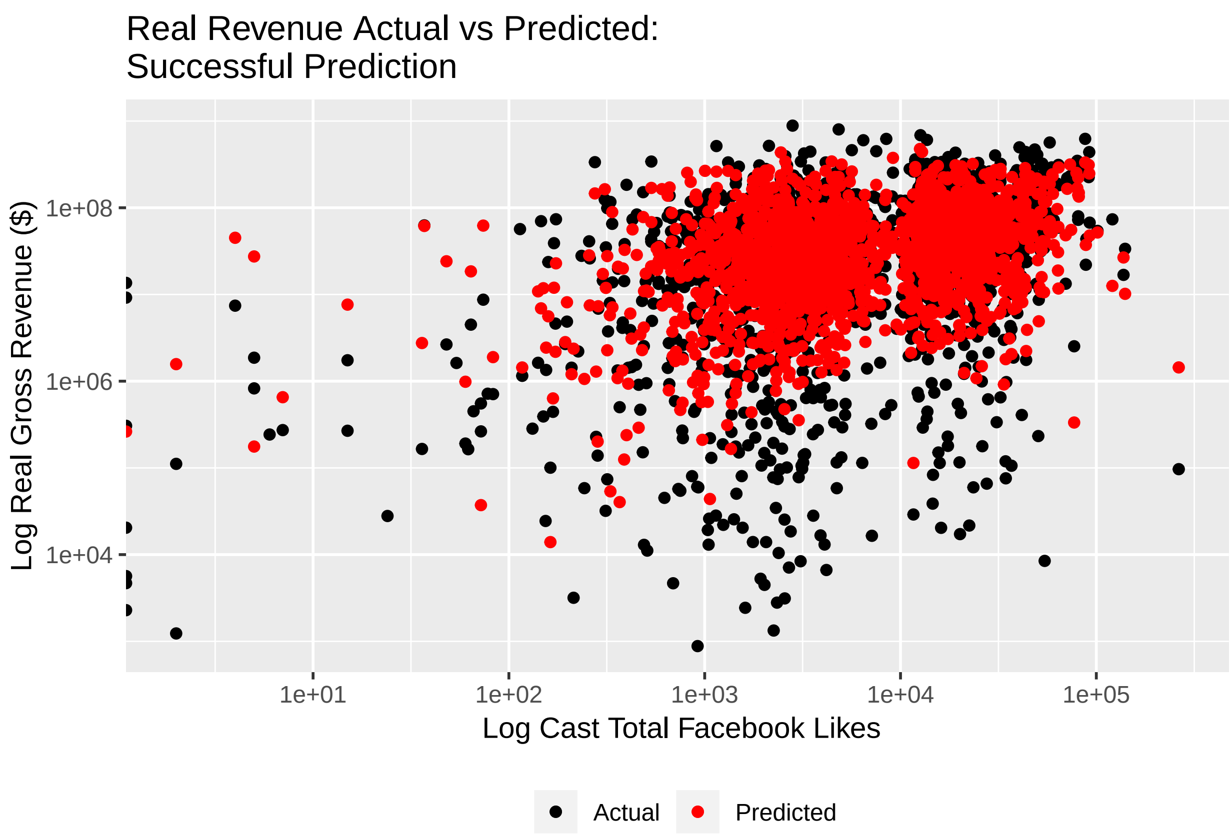

```
train_pred_all <- train %>%
  add_predictions(mod_all, 'lpred') %>%
  mutate(pred = 10^lpred)

# point because many factors. won't just be a single coefficient determining
  # SEE BOOK where did prediction based on a bunch of variables
lapply(c('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes', 'imdb_score'), function
    train_pred_all %>%
    ggplot(aes_string(x = var)) +
    geom_point(aes(y = real_gross, color = 'Actual')) +
    geom_point(aes(y = pred, color = 'Predicted')) +
    scale_y_log10() + scale_x_log10()
})
```

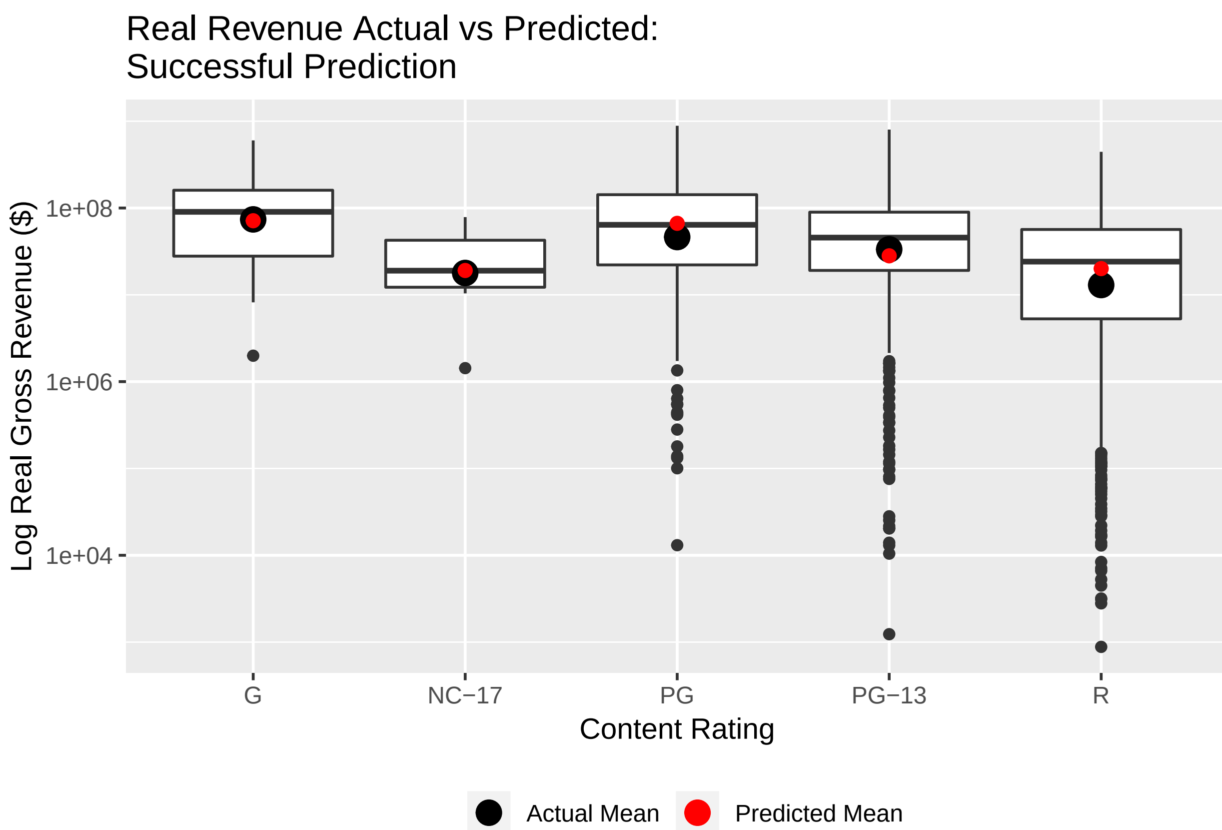
```
# predictions against other genres
lapply(c('content_rating', 'year', 'total_oscars_actor', 'total_oscars_director', all_genre_vars), func
  train_pred_all %>%
    ggplot(aes_string(x = var)) +
    geom_boxplot(aes(y = real_gross)) +
    # include mean
    stat_summary(aes(y = real_gross), fun.y = mean, geom = 'point', size = 4) +
    geom_point(data = train_pred %>% group_by(!!rlang::sym(var)) %>% summarise(mean = mean(pred)),
               aes(y = mean), color = 'red', size = 2) +
    scale_y_log10()
})
```

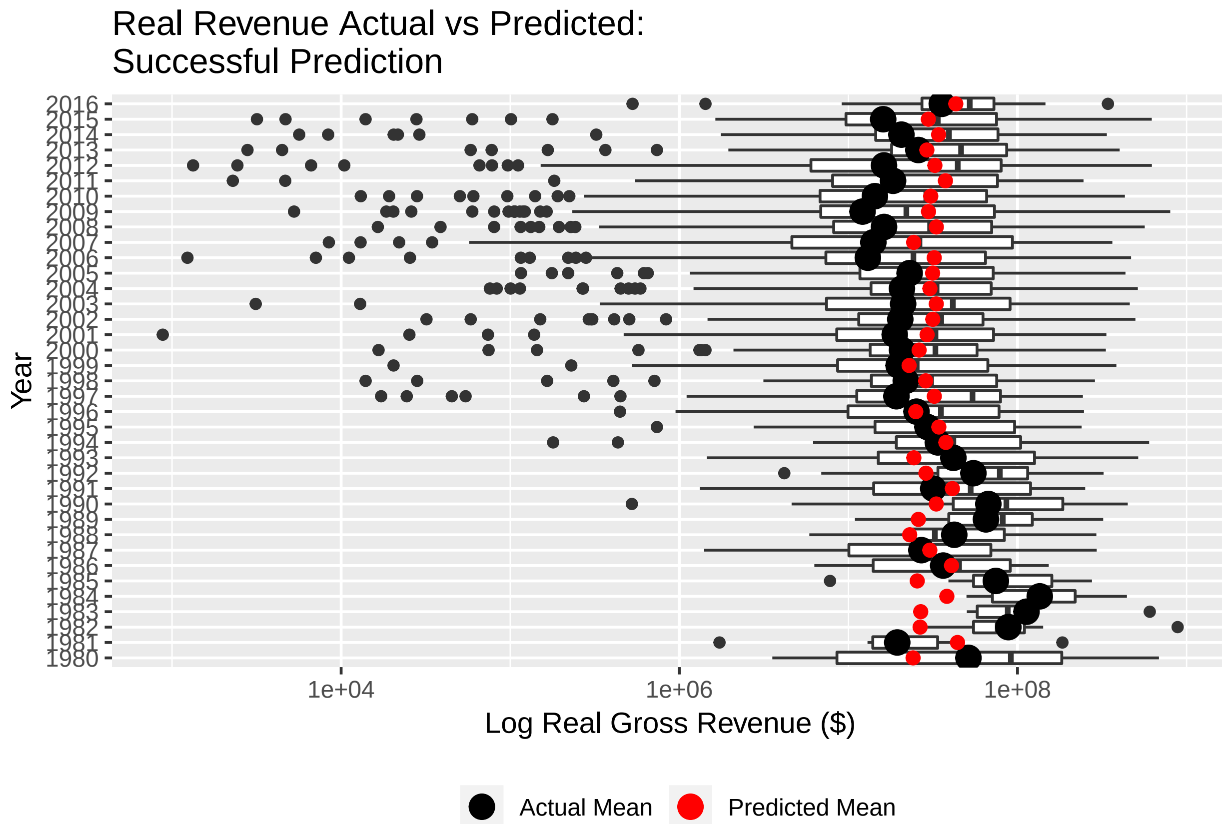

Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
1e+04
1e+06
1e+08
Log Real Budget ($)
Actual
Predicted

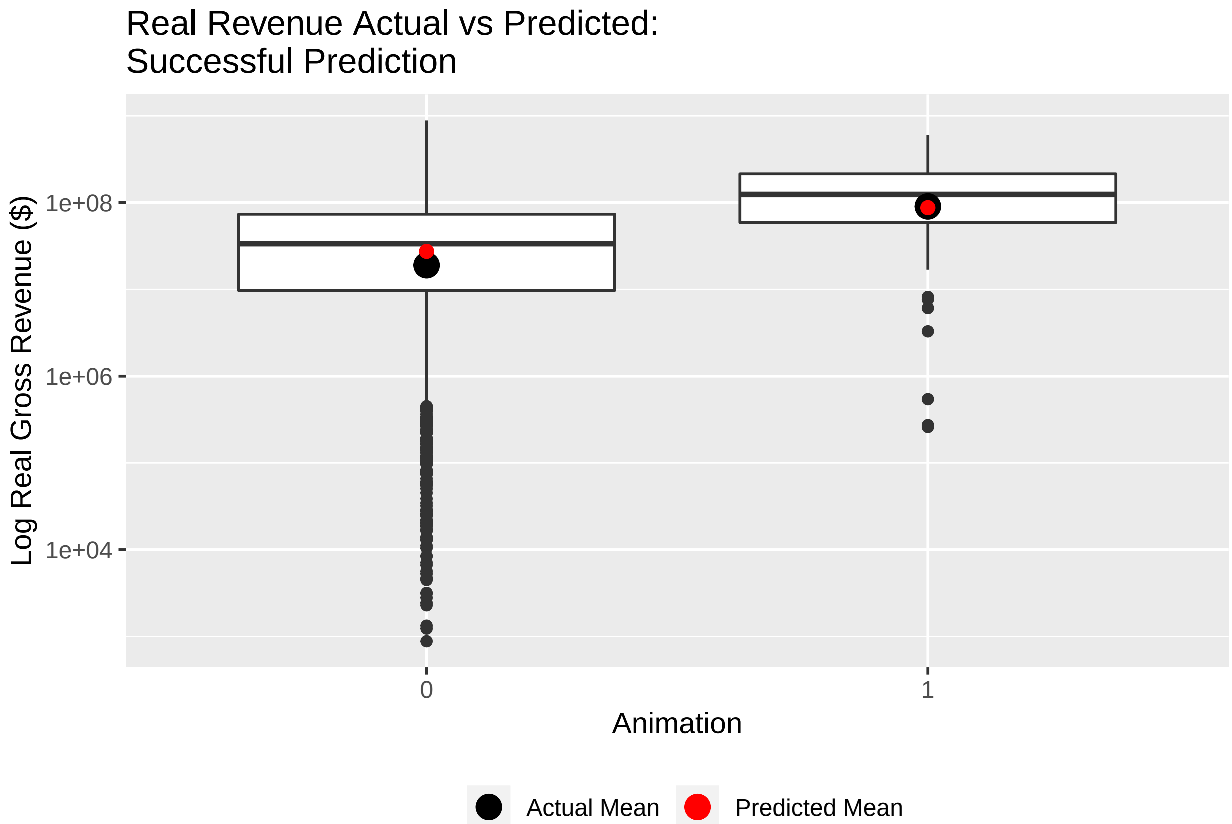
Real Revenue Actual vs Predicted:
Successful Prediction




Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
1e+01
1e+02
1e+03
1e+04
1e+05
Log Cast Total Facebook Likes
Actual
Predicted

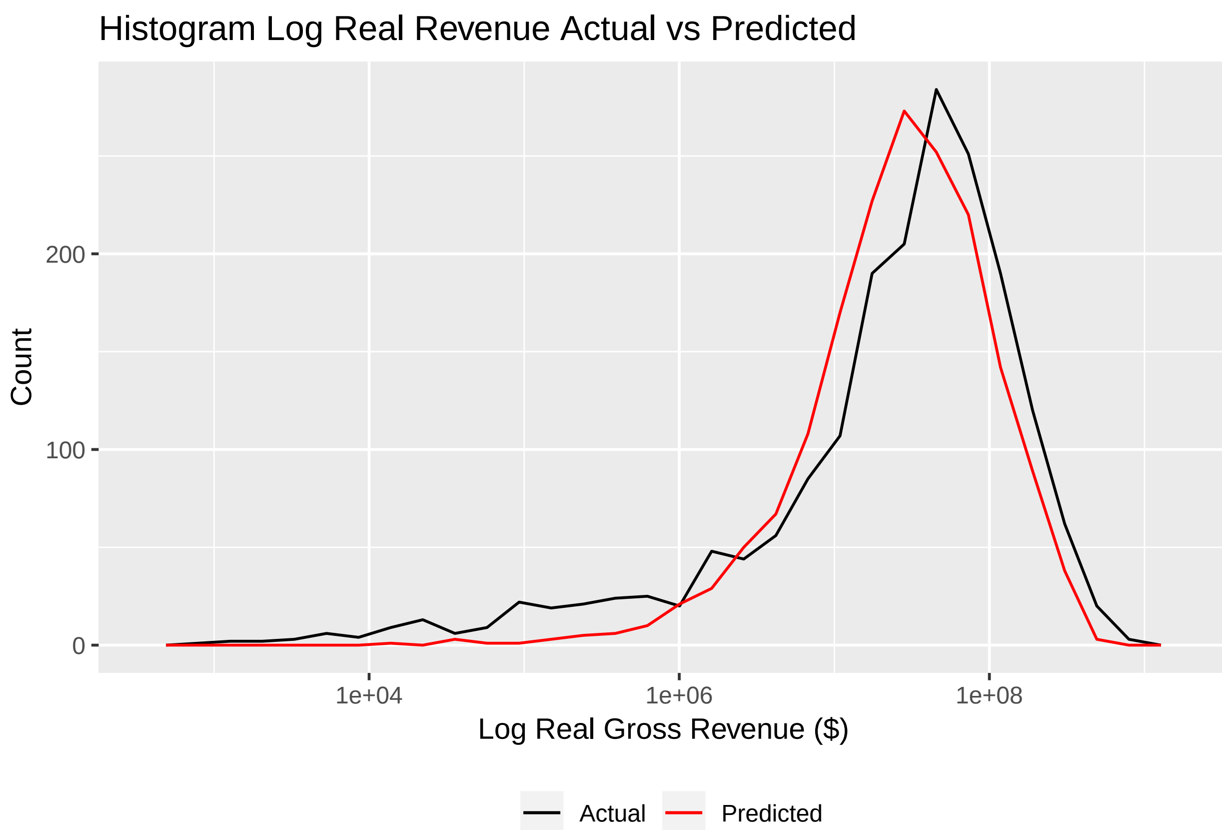

Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
G
NC−17
PG
PG−13
R
Content Rating
Actual Mean
Predicted Mean


Real Revenue Actual vs Predicted:
Successful Prediction
Year
Log Real Gross Revenue ($)
1e+04
1e+06
1e+08
Actual Mean
Predicted Mean




```
train %>%
  add_predictions(mod_all, 'lpred') %>%
  mutate(pred = 10^lpred) %>%
  ggplot() +
  geom_freqpoly(aes(x = real_gross, color = 'Actual')) +
  geom_freqpoly(aes(x = pred, color = 'Predicted')) +
  scale_x_log10() +
  theme(legend.position="bottom") +
  labs(x = 'Log Real Gross Revenue ($)', y = 'Count', title = 'Histogram Log Real Revenue Actual vs Pre
  scale_color_manual(name = '', values = c(Actual = 'black', Predicted = 'red'))
```

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

## Warning: Removed 132 rows containing non-finite values (stat_bin).
```



Fit Model From Scratch

Log budget and log revenue have the strongest and most linear relationship basd on EDA. Also it is very logical that they would be related. Start with budget.

- Plot other variables against the residuals from the budget model.
- Non-random: year, IMDB score, content rating. All of these are somewhat subtle. Budget does a pretty good job of estimating this relationship by itself. However, high IMDB scores do have positive residuals, implying that actual > predicted revenue. Some years have skewed residuals. G movies also have positive residuals.
  + All genres have random relationships with residual. Do not need to include genre.
- Include the variables with non-random residuals (year, IMDB score, content rating) in a step-wise selection process. All included based off of step-wise.

- Look at new residuals of included and excluded relationships. All fairly random.
- Look at predictions for each individual variable. Good job even for variables not included in the model.
  - Year isn't great again.
- Look at predictions overall. Looks good.

Not shown here, but I also fit a step wise from absolute scratch. Result was the same four variables plus Comedy and Mystery.
Also did step wise where all of the genre variables had to be included together. Got the same four variables and no genre.

```
mod_simple <- lm(real_gross_log ~ real_budget_log, data = train)
summary(mod_simple)
```

```
##
## Call:
## lm(formula = real_gross_log ~ real_budget_log, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4222 -0.2401  0.0963  0.3667  3.5856
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.63498    0.18025   3.523 0.000438 ***
## real_budget_log  0.91363    0.02437  37.486  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6755 on 1747 degrees of freedom
##   (102 observations deleted due to missingness)
## Multiple R-squared:  0.4458, Adjusted R-squared:  0.4455
## F-statistic:  1405 on 1 and 1747 DF,  p-value: < 2.2e-16
```

```r
rmse(mod_simple, data = valid)
```

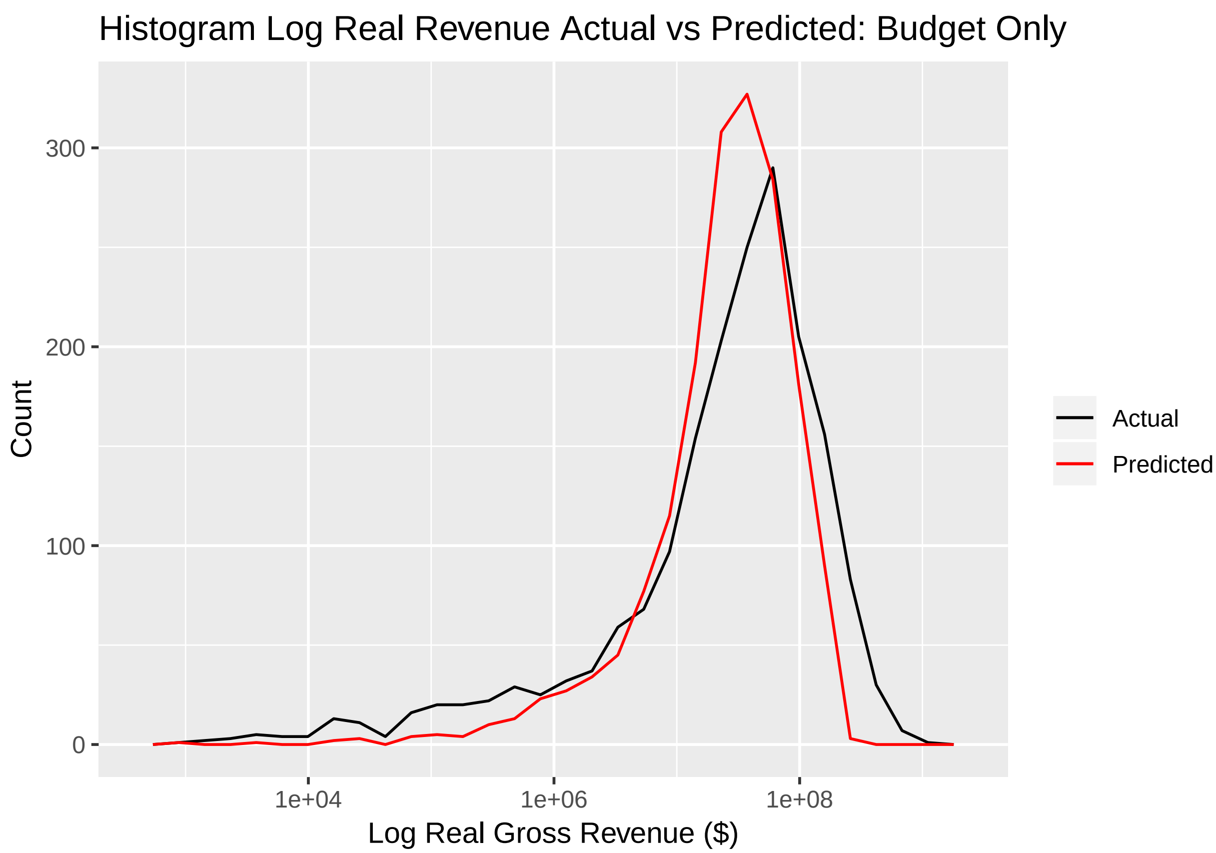
```
## [1] 0.6497192
```

```r
gr_resid(mod_simple)
```

Take a quick look at the predictions from this very simple model with just budget. Decent, but was better with more variables.

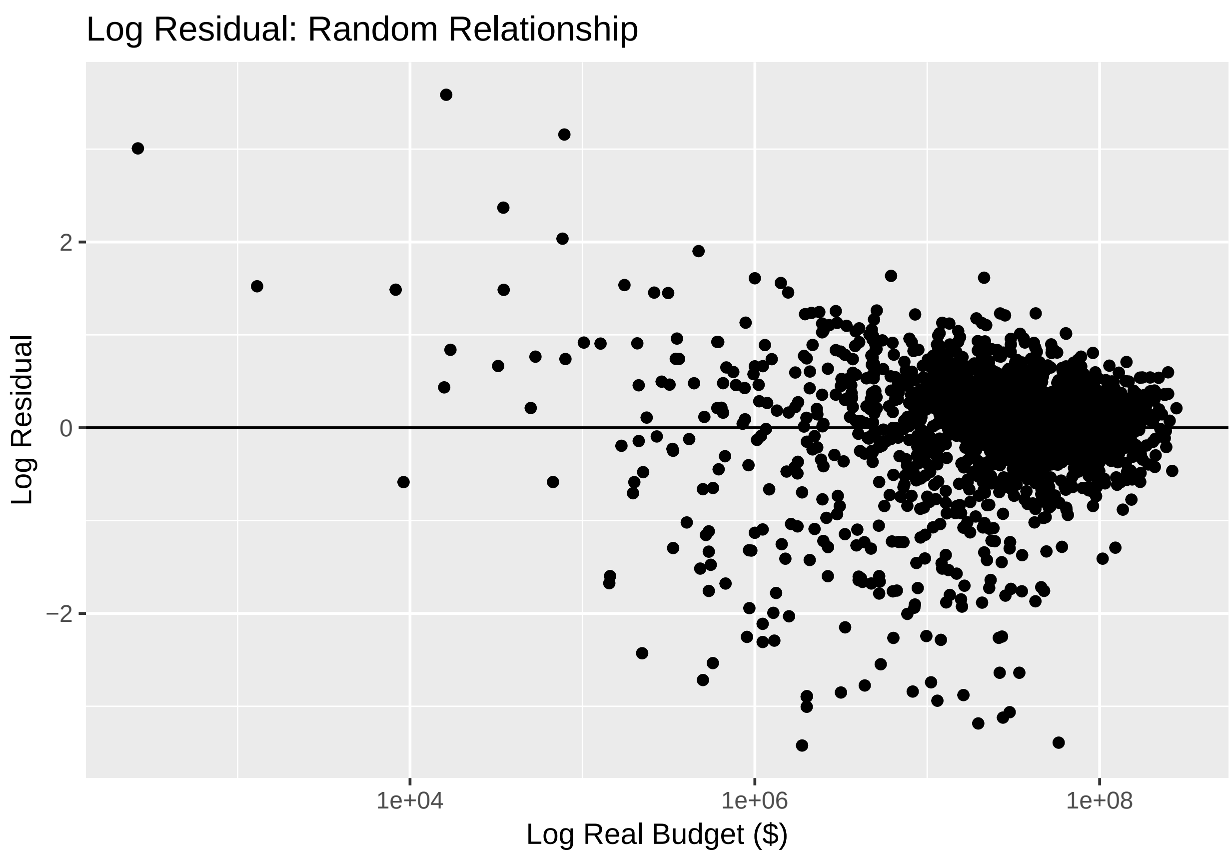
```r
train %>%
  add_predictions(mod_simple, 'lpred') %>%
  mutate(pred = 10^lpred) %>%
  ggplot() +
  geom_freqpoly(aes(x = real_gross, color = 'Actual')) +
  geom_freqpoly(aes(x = pred, color = 'Predicted')) +
  scale_x_log10() +
  labs(x = 'Log Real Gross Revenue ($)', y = 'Count', title = 'Histogram Log Real Revenue Actual vs Pre
  scale_color_manual(name = '', values = c(Actual = 'black', Predicted = 'red'))
```

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

## Warning: Removed 102 rows containing non-finite values (stat_bin).
```

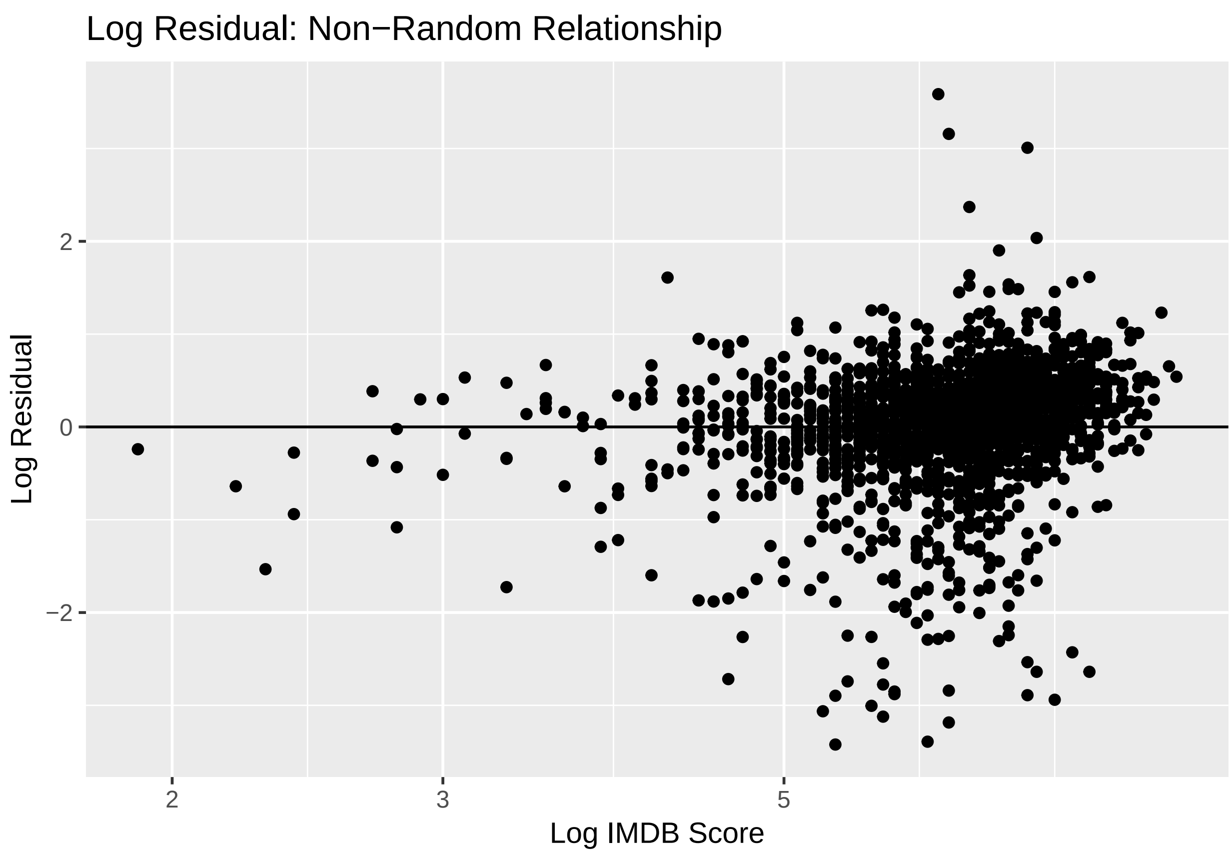



```r
train_resid_simple <- train %>%
  add_residuals(mod_simple, 'lresid')

gr_con <- function(v, title, xtitle) {
  lapply(v, function(var) {
    train_resid_simple %>%
      ggplot() +
      geom_point(aes_string(var, y = 'lresid')) +
      geom_hline(aes(yintercept = 0)) +
      labs(y = 'Log Residual', title = title, x = xtitle) +
      scale_x_log10()
  })
}

gr_cat <- function(v, title, xtitle, flip = F) {
  lapply(v, function(var) {
    gr <- train_resid_simple %>%
      filter(!is.na(!!rlang::sym(var))) %>%
      ggplot() +
      geom_jitter(aes_string(var, y = 'lresid'), alpha = .3) +
      geom_hline(aes(yintercept = 0)) +
      labs(y ='Log Residual', title = title, x = xtitle)
    if (flip) {
      gr + coord_flip()
    } else{
      gr
```

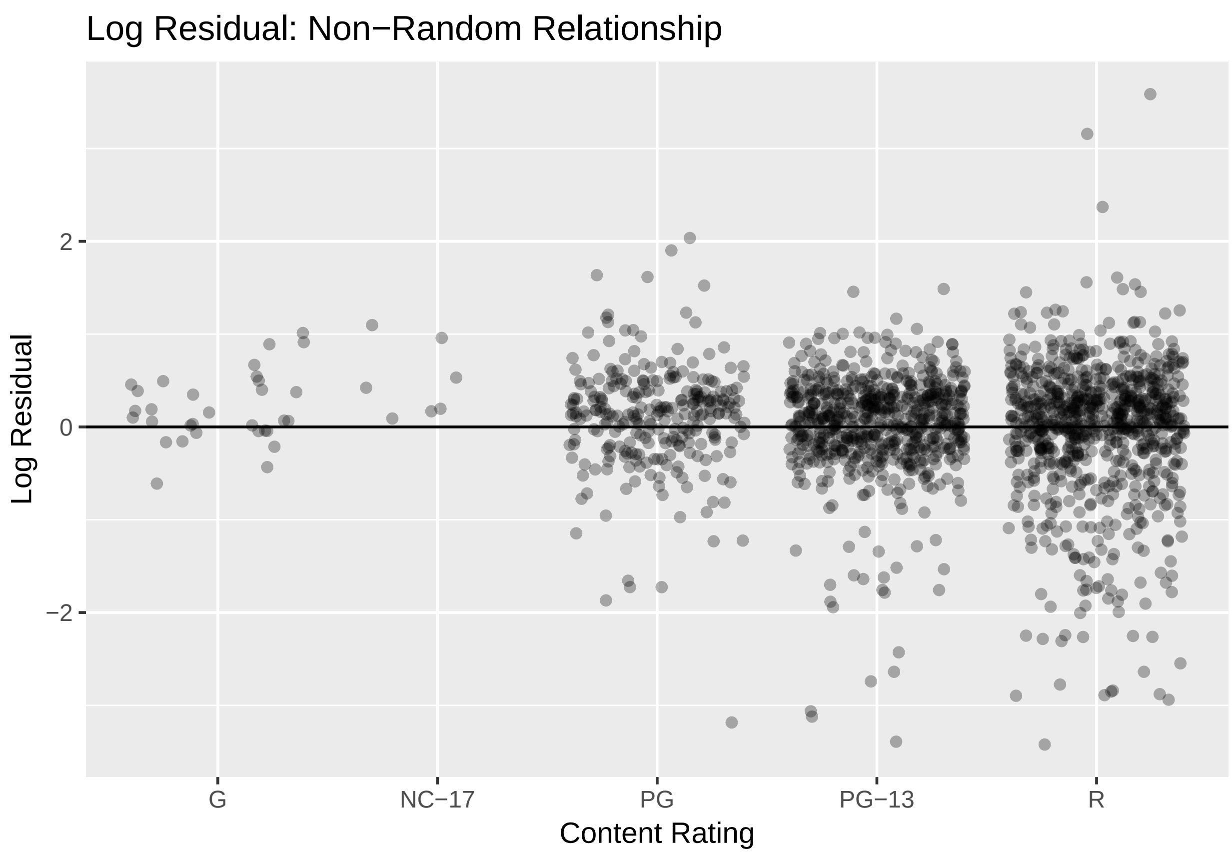
```
    }
  })
}

gr_con('real_budget', 'Log Residual: Random Relationship', 'Log Real Budget ($)')
```



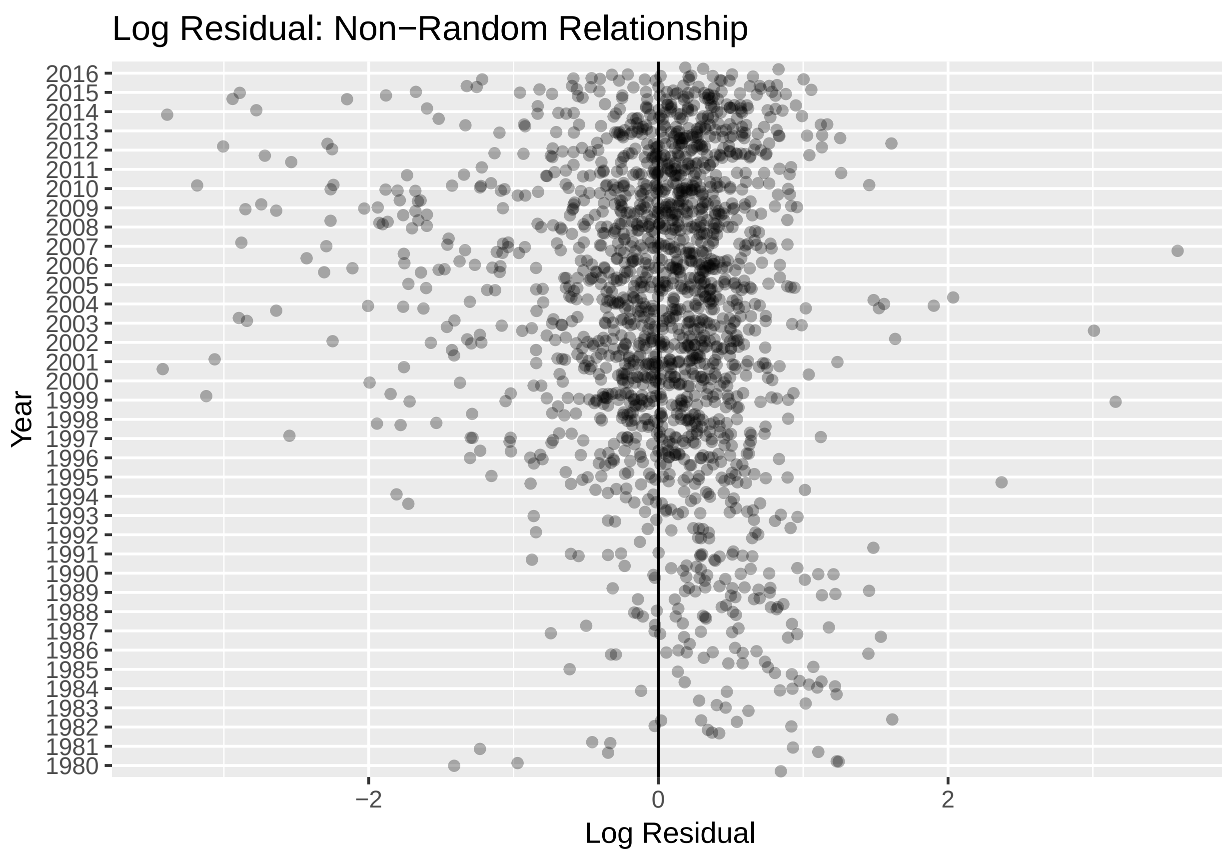

```
gr_con('imdb_score', 'Log Residual: Non-Random Relationship', 'Log IMDB Score')
```



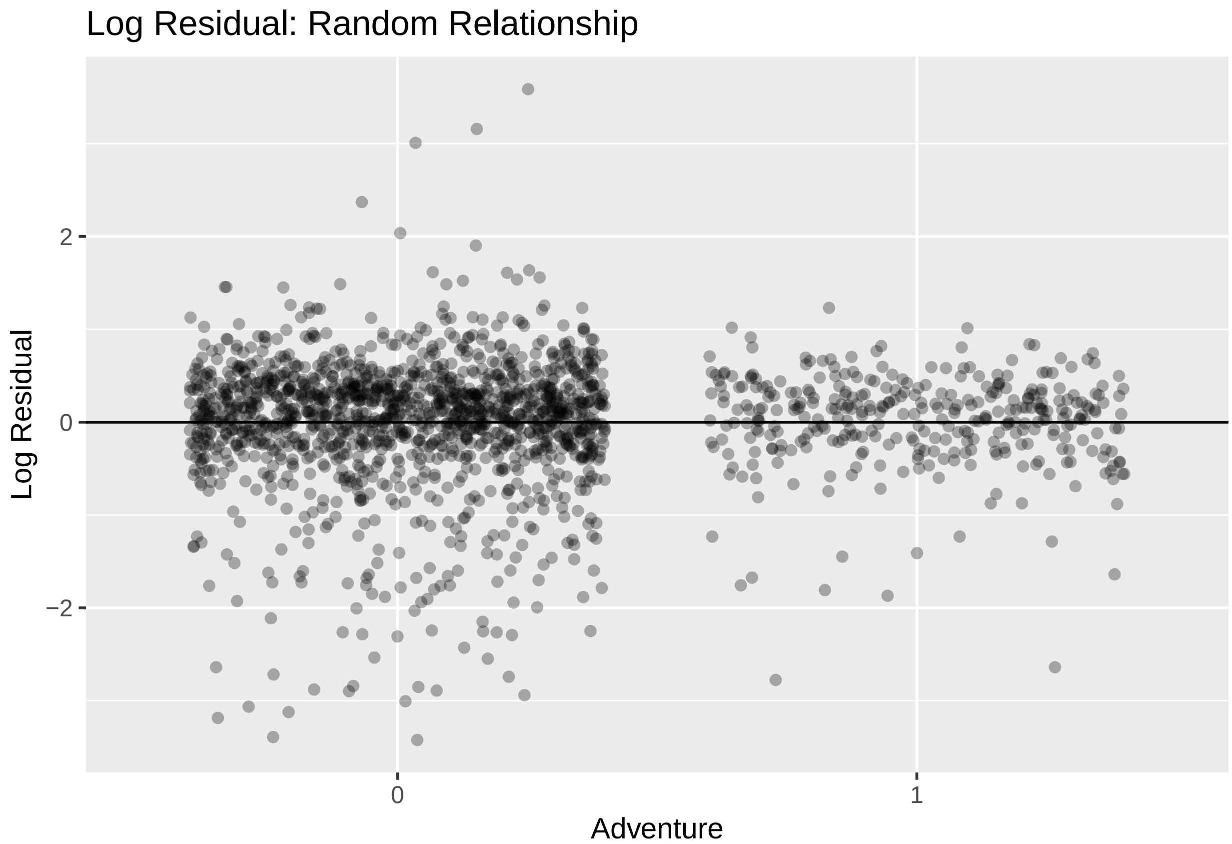

```
gr_cat('content_rating', 'Log Residual: Non-Random Relationship', 'Content Rating')
```



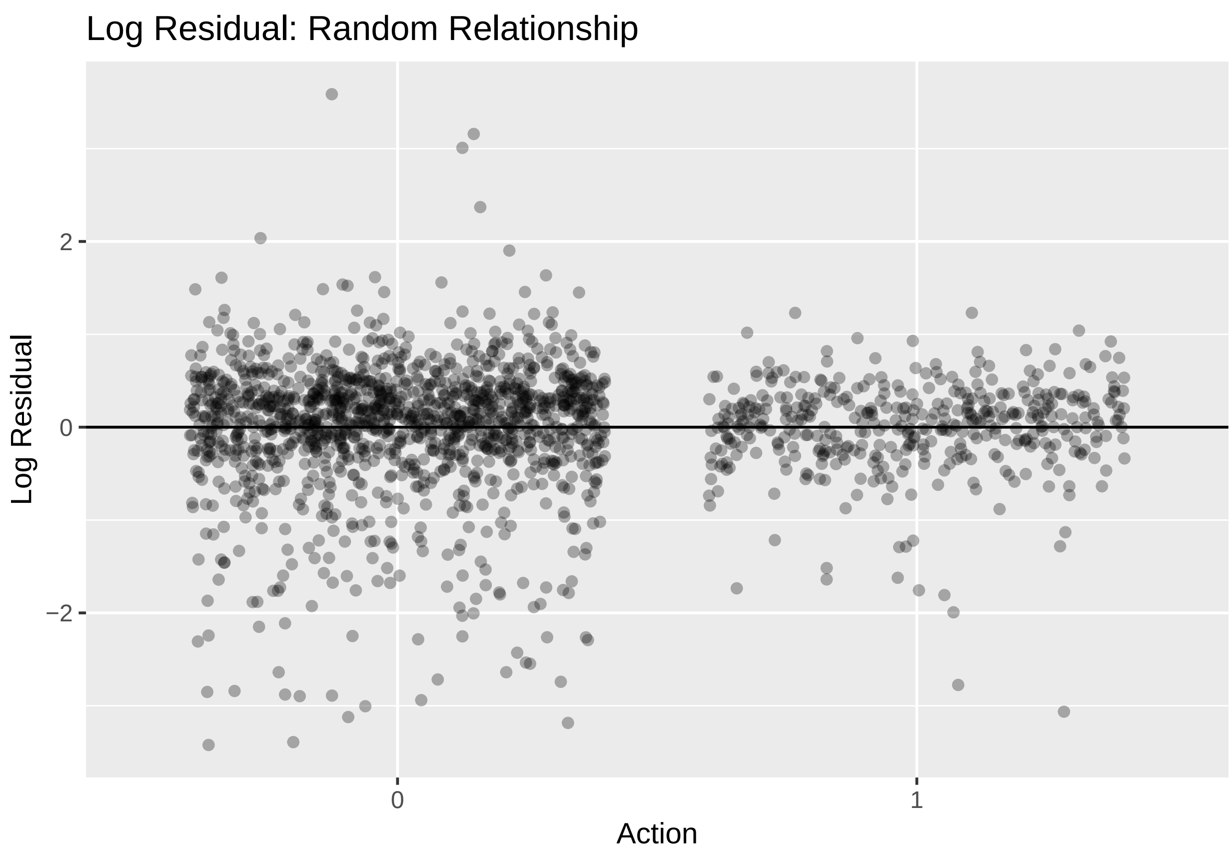

```
gr_cat('year', 'Log Residual: Non-Random Relationship', 'Year', flip = T)
```




```
gr_cat('Adventure', 'Log Residual: Random Relationship', 'Adventure')
```



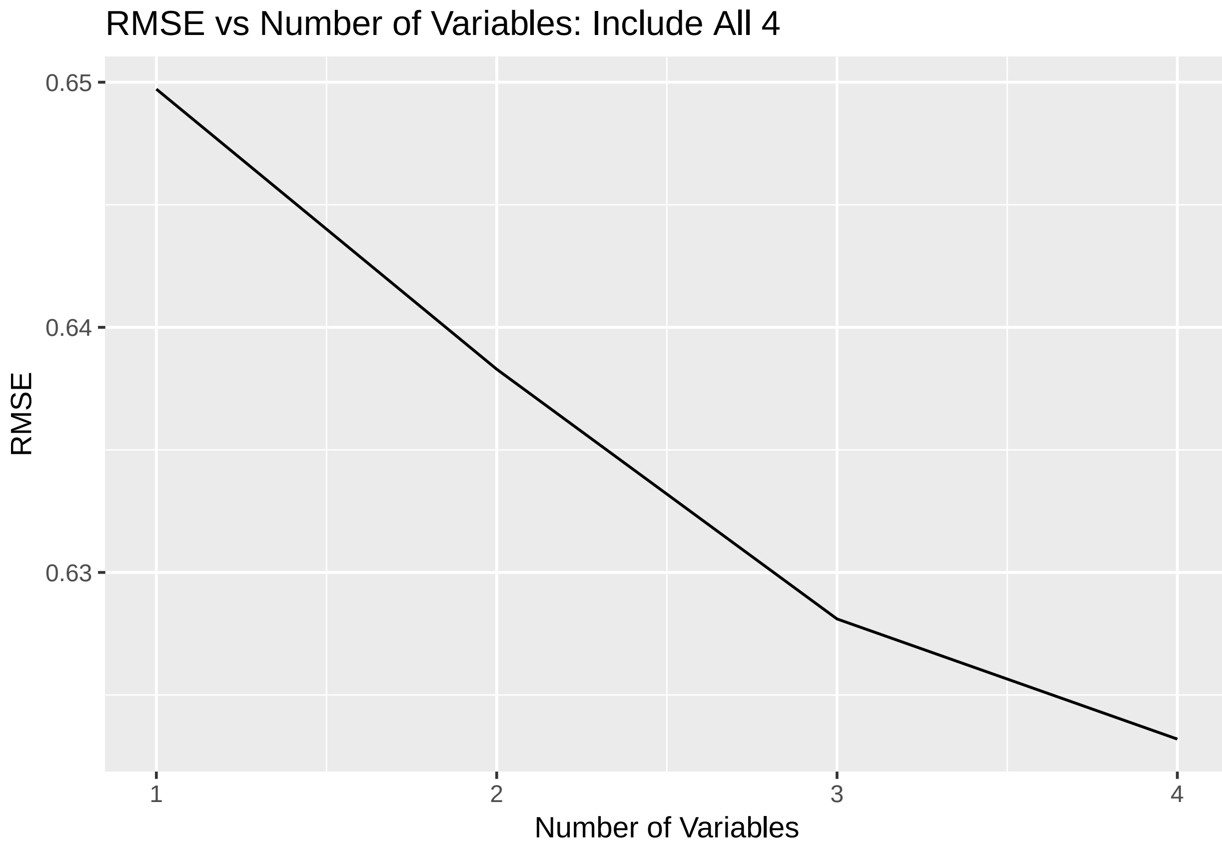

```
gr_cat('Action', 'Log Residual: Random Relationship', 'Action')
```




```r
# stepwise
# ALL potentially relevant variables
rmse_lst <- step_wise_loop(df = train %>% select(real_budget_log, imdb_score_log, year,
                                                  content_rating))
```

```
## real_budget_log
##       0.6497192
## [1] 1
## imdb_score_log
##      0.6382915
## [1] 2
##      year
## 0.6281035
## [1] 3
## content_rating
##       0.623201
```

```r
# graph RMSE vs number of variables
fit_rmse <- tibble(nvar = 1:length(rmse_lst),
                   rmse = rmse_lst)
ggplot(fit_rmse, aes(x = nvar, y = rmse)) + geom_line() +
  scale_x_continuous(breaks = seq(1, length(rmse_lst), by = 1)) +
  labs(x = 'Number of Variables', y = 'RMSE',
       title = 'RMSE vs Number of Variables: Include All 4')
```




```r
# after var 4, decreases too small or increase

mod_simple2 <- lm(real_gross_log ~ real_budget_log + imdb_score_log + year + content_rating,
                  data = train)

summary(mod_simple2)
```

```
##
## Call:
## lm(formula = real_gross_log ~ real_budget_log + imdb_score_log +
##     year + content_rating, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4983 -0.2230  0.0973  0.3455  3.4758
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.444778   0.391868   1.135  0.25653
## real_budget_log        0.821725   0.026058  31.534  < 2e-16 ***
## imdb_score_log         1.626517   0.199024   8.172 5.89e-16 ***
## year1981              -0.238907   0.376557  -0.634  0.52587
## year1982               0.297881   0.346750   0.859  0.39043
## year1983               0.235272   0.393319   0.598  0.54981
## year1984               0.504909   0.336175   1.502  0.13331
## year1985               0.335770   0.346892   0.968  0.33322
```

```
## year1986             0.111891   0.331663   0.337  0.73589
## year1987             0.212045   0.322869   0.657  0.51143
## year1988             0.226528   0.316488   0.716  0.47424
## year1989             0.285472   0.311721   0.916  0.35991
## year1990             0.178997   0.312725   0.572  0.56714
## year1991             0.014859   0.316338   0.047  0.96254
## year1992             0.045736   0.321585   0.142  0.88692
## year1993            -0.003696   0.312675  -0.012  0.99057
## year1994            -0.184559   0.306846  -0.601  0.54761
## year1995            -0.042381   0.298579  -0.142  0.88714
## year1996            -0.208917   0.291490  -0.717  0.47365
## year1997            -0.131576   0.291460  -0.451  0.65173
## year1998            -0.246108   0.290619  -0.847  0.39721
## year1999            -0.232583   0.287331  -0.809  0.41837
## year2000            -0.141456   0.288697  -0.490  0.62421
## year2001            -0.245089   0.286678  -0.855  0.39271
## year2002            -0.222859   0.286430  -0.778  0.43665
## year2003            -0.176337   0.287847  -0.613  0.54022
## year2004            -0.193313   0.287370  -0.673  0.50123
## year2005            -0.210828   0.286412  -0.736  0.46177
## year2006            -0.308781   0.286785  -1.077  0.28177
## year2007            -0.267002   0.288303  -0.926  0.35452
## year2008            -0.323941   0.286439  -1.131  0.25825
## year2009            -0.325844   0.285759  -1.140  0.25434
## year2010            -0.338852   0.286066  -1.185  0.23637
## year2011            -0.217375   0.288402  -0.754  0.45112
## year2012            -0.110811   0.287004  -0.386  0.69948
## year2013            -0.037216   0.286569  -0.130  0.89669
## year2014            -0.103257   0.288515  -0.358  0.72047
## year2015            -0.216276   0.290823  -0.744  0.45718
## year2016            -0.069711   0.307146  -0.227  0.82048
## content_ratingNC-17 -0.197326   0.272730  -0.724  0.46946
## content_ratingPG    -0.123677   0.119665  -1.034  0.30150
## content_ratingPG-13 -0.175106   0.115920  -1.511  0.13109
## content_ratingR     -0.346131   0.115941  -2.985  0.00287 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6214 on 1676 degrees of freedom
##   (132 observations deleted due to missingness)
## Multiple R-squared:  0.4685, Adjusted R-squared:  0.4552
## F-statistic: 35.18 on 42 and 1676 DF,  p-value: < 2.2e-16
```

```r
rmse(mod_simple2, data = valid)
```

```
## [1] 0.623201
```

```r
# when consider factors as one variable, they are significant
anova(mod_simple2)
```

```
## Analysis of Variance Table
##
## Response: real_gross_log
##                   Df Sum Sq Mean Sq   F value    Pr(>F)
## real_budget_log    1 491.67  491.67 1273.3487 < 2.2e-16 ***
```

```
## imdb_score_log     1  28.18   28.18   72.9937 < 2.2e-16 ***
## year              36  36.51    1.01    2.6269 6.334e-07 ***
## content_rating     4  14.13    3.53    9.1519 2.589e-07 ***
## Residuals       1676 647.14    0.39
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# number of observations
nobs(mod_simple2)
```

```
## [1] 1719
```

```r
# compare with anova to more complex models
anova(mod_all, mod_simple2)
```

```
## Analysis of Variance Table
##
## Model 1: real_gross_log ~ Adventure + Action + Family + Mystery + Documentary +
##     Drama + History + Romance + real_budget_log + imdb_score_log +
##     content_rating + year
## Model 2: real_gross_log ~ real_budget_log + imdb_score_log + year + content_rating
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1   1668 614.85
## 2   1676 647.14 -8   -32.291 10.95 3.744e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# to compare to mod_simple, need some number of observations: filter out missing content rating and the
mod_simple <- lm(real_gross_log ~ real_budget_log, data = train %>% filter(!is.na(content_rating)))
anova(mod_simple, mod_simple2)
```

```
## Analysis of Variance Table
##
## Model 1: real_gross_log ~ real_budget_log
## Model 2: real_gross_log ~ real_budget_log + imdb_score_log + year + content_rating
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   1717 725.98
## 2   1676 647.14 41    78.834 4.9797 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
stargazer(mod_simple2, type = 'latex', out = 'C:/Users/Gimli/Documents/DS5110/project/output/final_mod.
          no.space = T, nobs = T, dep.var.caption = '', align = T, dep.var.labels = '',
          title = 'Final Model Summary \\\\ Response Variable: Log Real Gross Revenue',
          omit = c('1981', '1982', '1983', '1984', '1985', '1986', '1987', '1988', '1989', '1990', '199
          '1994', '1995', '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003', '2004', '2005
          '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016'),
          covariate.labels = c('Log Real Budget', 'Log IMDB Score', 'NC17', 'PG', 'PG13', 'R'),
          notes = 'Year dummies omitted for presentation purposes', notes.align = 'l', column.sep.width
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Wed, Apr 10, 2019 - 6:11:11 PM
## % Requires LaTeX packages: dcolumn
## \begin{table}[!htbp] \centering
##   \caption{Final Model Summary \\ Response Variable: Log Real Gross Revenue}
##   \label{}
```

```
# stargazer anova doesn't let you do nice variable labels: covariate.labels is for column headers in th
train_labels <- train %>%
  rename(`Log Real Budget ($)` = real_budget_log,
         `Content Rating` = content_rating,
         `Log IMDB Score` = imdb_score_log,
         Year = year)
mod_simple2_labels <- lm(real_gross_log ~ `Log Real Budget ($)` + `Log IMDB Score` + `Content Rating` +
               data = train_labels)
stargazer(anova(mod_simple2_labels), type = 'latex', out = 'C:/Users/Gimli/Documents/DS5110/project/outp
          no.space = T, nobs = T, dep.var.caption = '', align = T, dep.var.labels = '',
          title = 'Final Model Model Anova \\\\ Response Variable: Log Real Gross Revenue', summary = F)
```

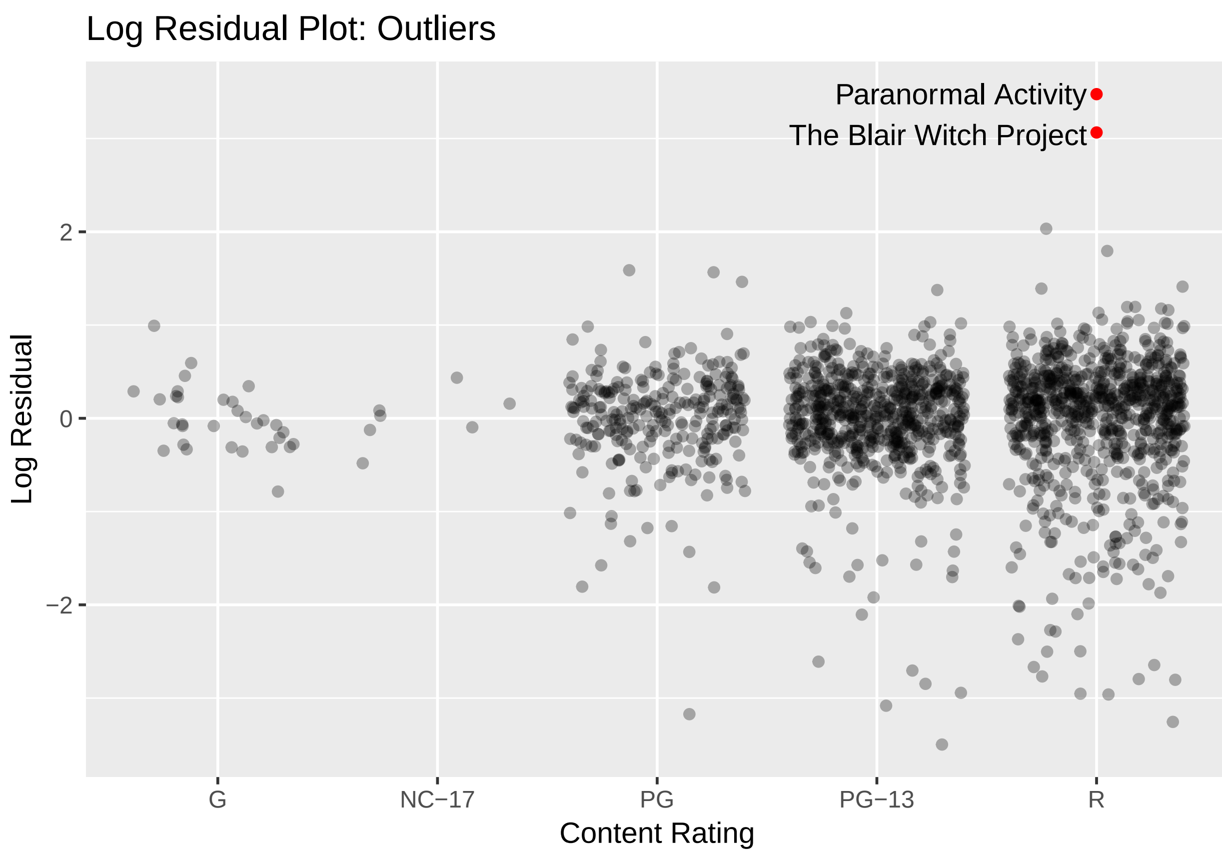
```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Wed, Apr 10, 2019 - 6:11:11 PM
## % Requires LaTeX packages: dcolumn
## \begin{table}[!htbp] \centering
##   \caption{Final Model Model Anova \\ Response Variable: Log Real Gross Revenue}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.}{.}{-3} D{.
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## \multicolumn{1}{c}{} & \multicolumn{1}{c}{Df} & \multicolumn{1}{c}{Sum Sq} & \multicolumn{1}{c}{Mean
```

```
## \hline \\[-1.8ex]
## \multicolumn{1}{c}{`Log Real Budget (\$)`} & 1 & 491.668 & 491.668 & 1,273.349 & 0 \\
## \multicolumn{1}{c}{`Log IMDB Score`} & 1 & 28.184 & 28.184 & 72.994 & 0 \\
## \multicolumn{1}{c}{`Content Rating`} & 4 & 14.321 & 3.580 & 9.272 & 0.00000 \\
## \multicolumn{1}{c}{Year} & 36 & 36.329 & 1.009 & 2.613 & 0.00000 \\
## \multicolumn{1}{c}{Residuals} & 1,676 & 647.141 & 0.386 &  &  \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

```
gr_resid(mod_simple2)
```

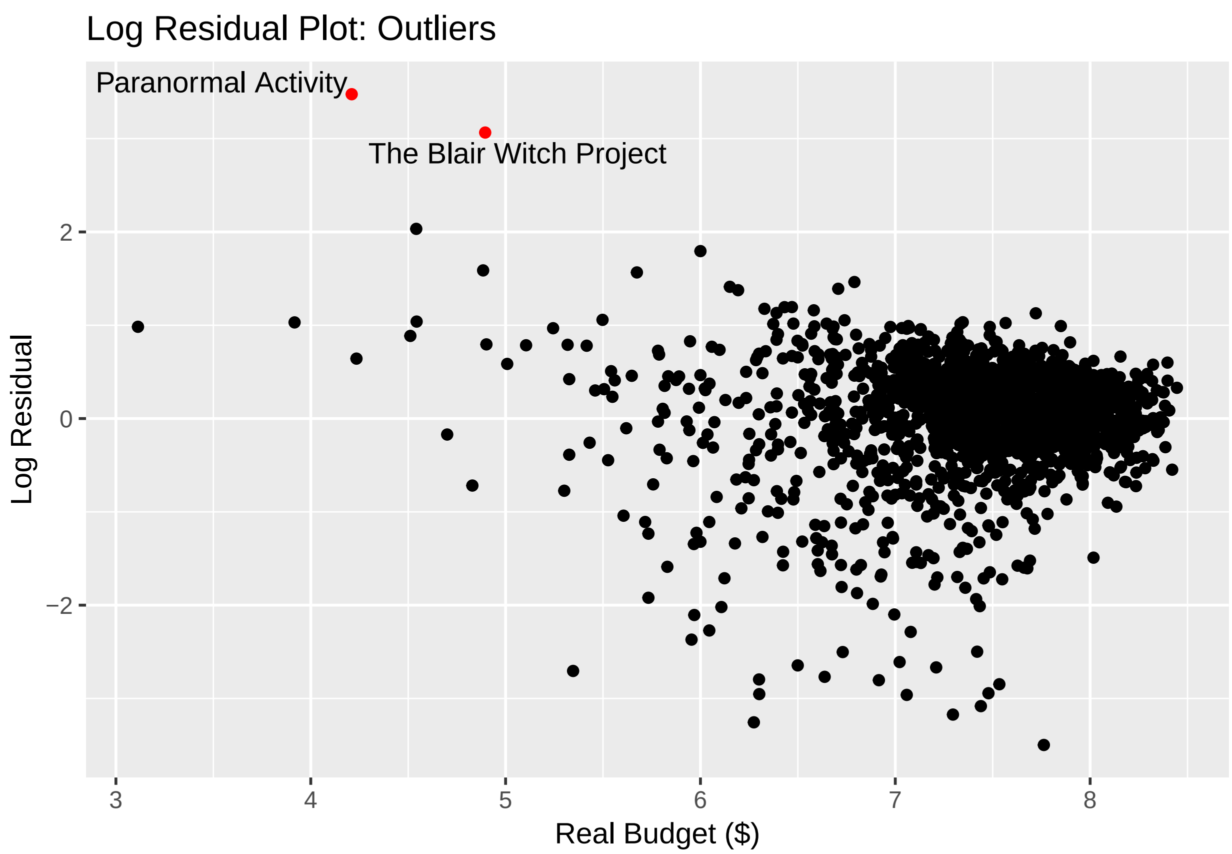
```
label <- train %>%
  add_residuals(mod_simple2, 'lresid') %>%
  filter(lresid > 2.5)

train_resid <- train %>%
    add_residuals(mod_simple2, 'lresid')

lapply(c('content_rating'), function(var) {
  train_resid %>%
    filter(!is.na(!!rlang::sym(var))) %>%
    filter(lresid < 2.5) %>%
    ggplot(aes_string(var, 'lresid')) +
    geom_jitter(alpha = .3) +
    geom_point(data = train_resid %>% filter(lresid > 2.5), color = 'red') +
    geom_text_repel(aes(label = movie_title), data = label, nudge_x = -.6, segment.color = NA) +
    labs(x = label(train_resid[var]), y = 'Log Residual', title = 'Log Residual Plot: Outliers')
})
```

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

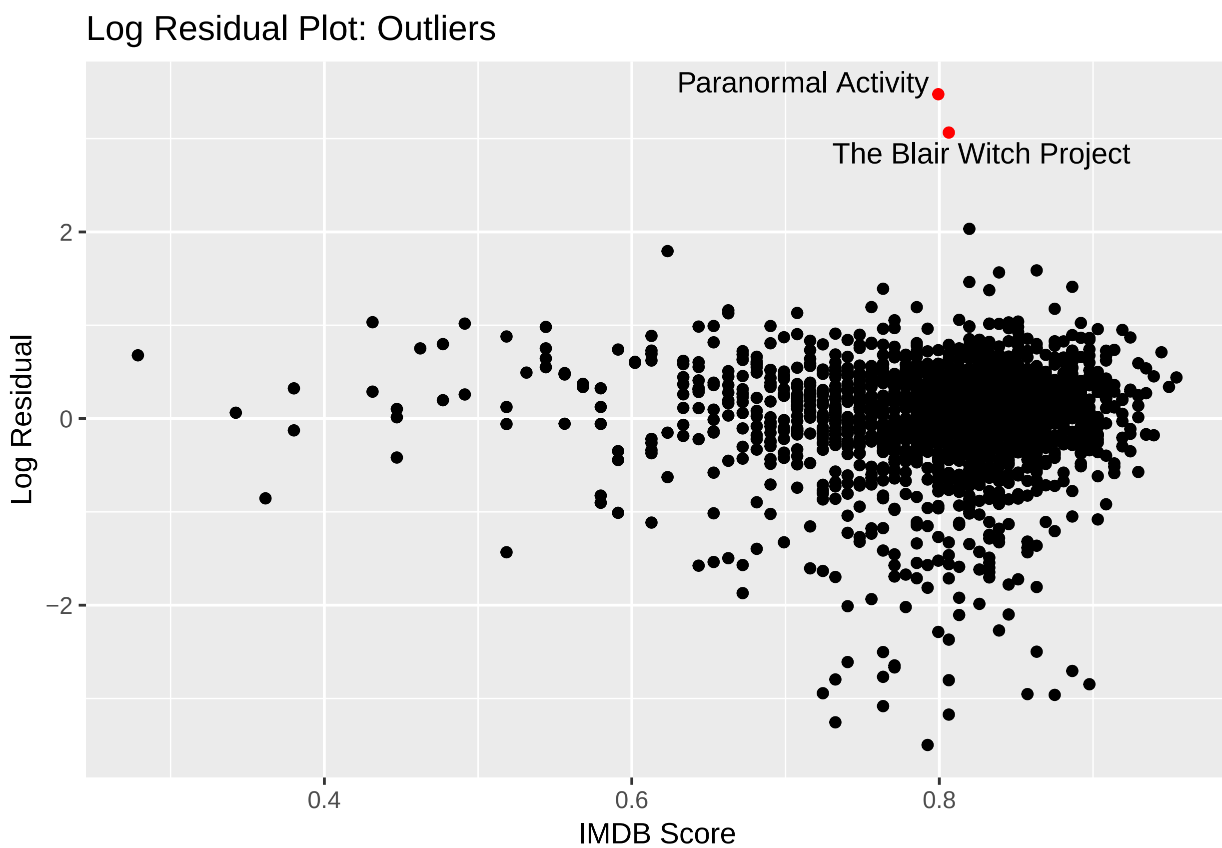



```
lapply(c('real_budget', 'imdb_score'), function(var) {
  train_resid %>%
    filter(lresid < 2.5) %>%
    ggplot(aes_string(str_c(var, '_log'), y = 'lresid')) +
    geom_point() +
    geom_point(data = train_resid %>% filter(lresid > 2.5), color = 'red') +
    geom_text_repel(aes(label = movie_title), data = label, segment.color = NA) +
    labs(x = label(train_resid[var]), y = 'Log Residual', title = 'Log Residual Plot: Outliers')
})
```

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


Log Residual Plot: Outliers
Paranormal Activity
The Blair Witch Project
Log Residual
Real Budget ($)
2
0
−2
3
4
5
6
7
8

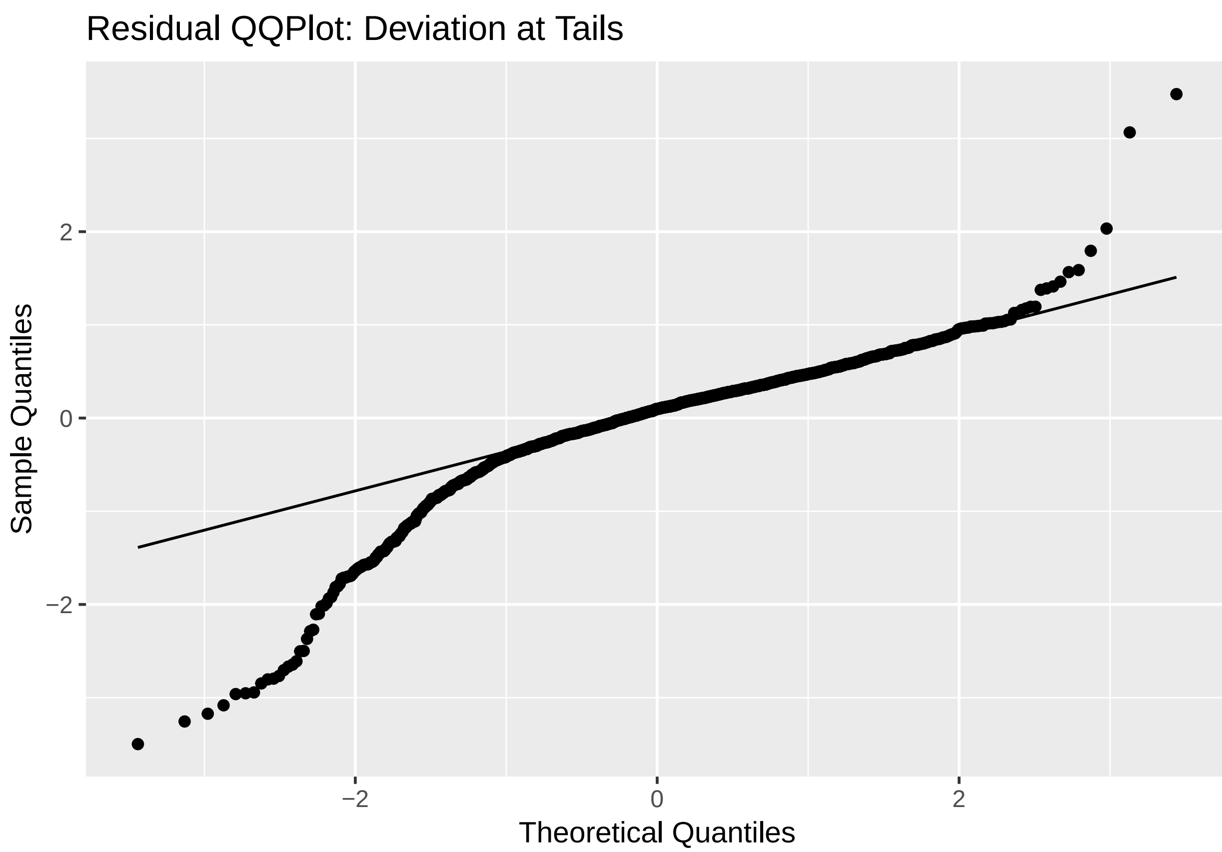
```
##
## [[2]]
```




```
train %>%
  add_residuals(mod_simple2,'lresid') %>%
  ggplot(aes(sample = lresid)) +
    geom_qq() +
    geom_qq_line() +
    labs(title = 'Residual QQPlot: Deviation at Tails',
       x = 'Theoretical Quantiles', y = 'Sample Quantiles')
```

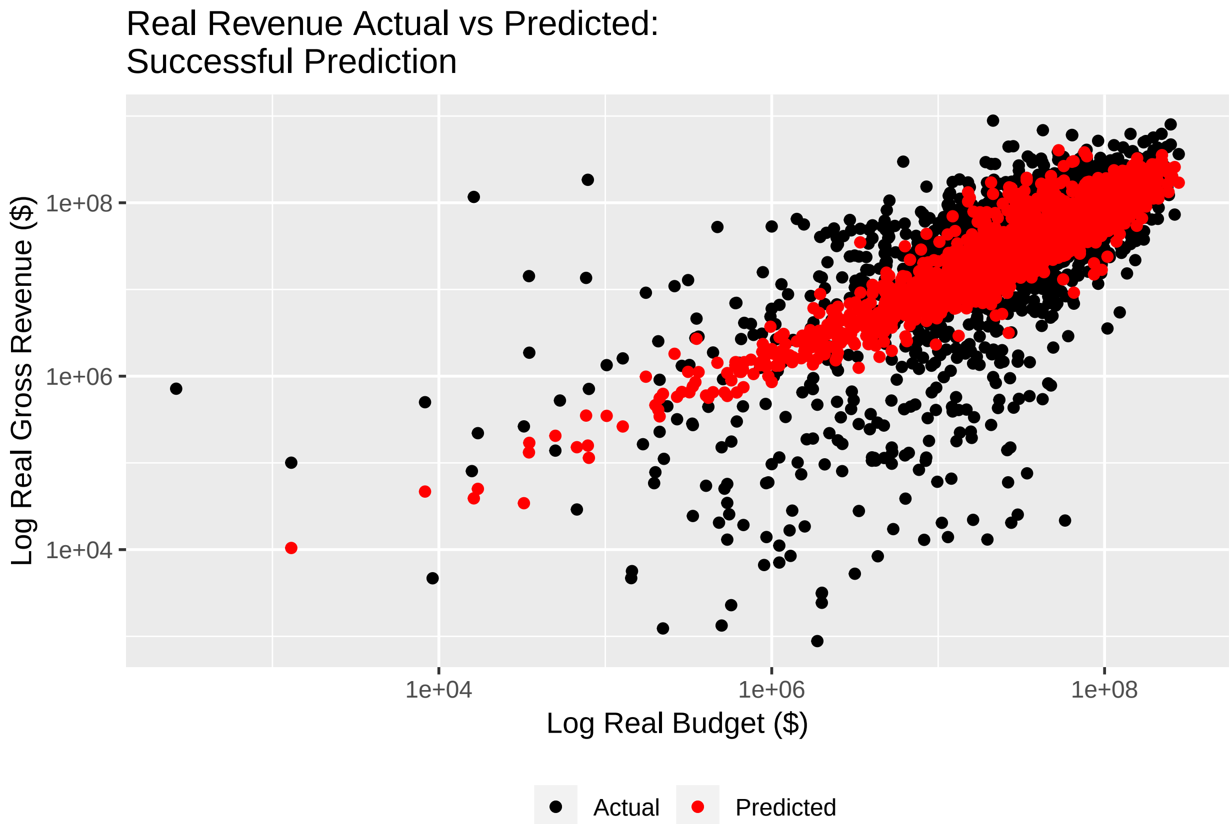
```
## Warning: Removed 132 rows containing non-finite values (stat_qq).
```

```
## Warning: Removed 132 rows containing non-finite values (stat_qq_line).
```

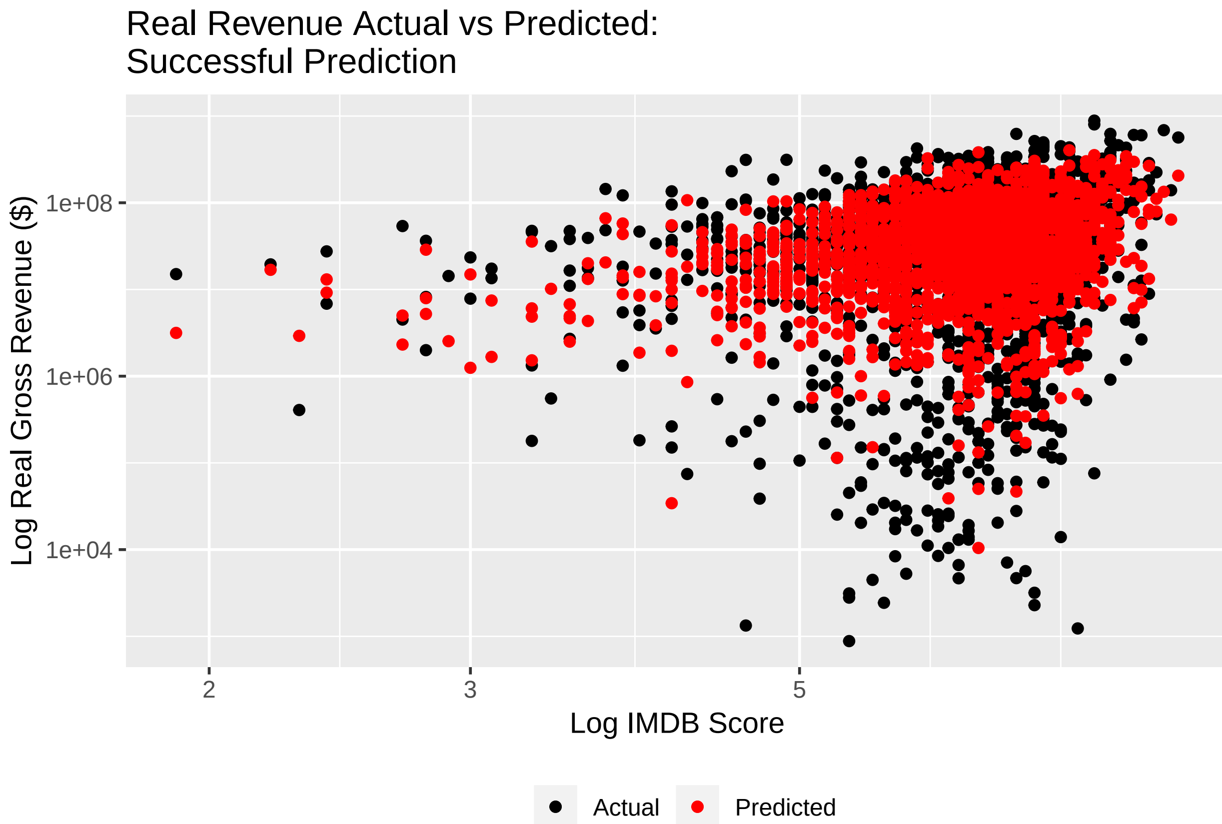


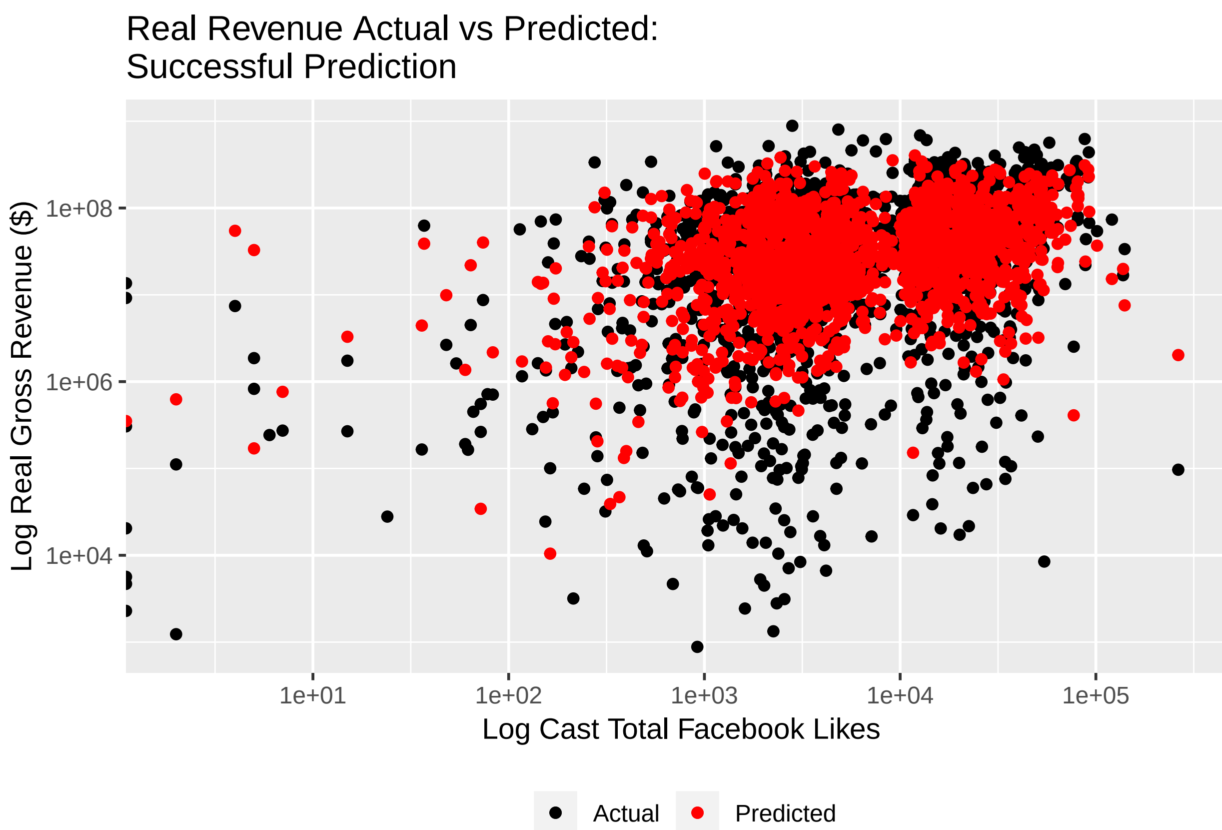

```
train_pred_simple <- train %>%
  add_predictions(mod_simple2, 'lpred') %>%
  mutate(pred = 10^lpred)

# point because many factors. won't just be a single coefficient determining
  # SEE BOOK where did prediction based on a bunch of variables
lapply(c('real_budget', 'director_facebook_likes', 'cast_total_facebook_likes', 'imdb_score'), function
    train_pred_simple %>%
    ggplot(aes_string(x = var)) +
    geom_point(aes(y = real_gross, color = 'Actual')) +
    geom_point(aes(y = pred, color = 'Predicted')) +
    scale_y_log10() + scale_x_log10()
})
```

```
# predictions against other genres
lapply(c('content_rating', 'year', 'total_oscars_actor', 'total_oscars_director', all_genre_vars), func
  train_pred_simple %>%
    ggplot(aes_string(x = var)) +
    geom_boxplot(aes(y = real_gross)) +
    # include mean
    stat_summary(aes(y = real_gross), fun.y = mean, geom = 'point', size = 4) +
    geom_point(data = train_pred %>% group_by(!!rlang::sym(var)) %>% summarise(mean = mean(pred)),
               aes(y = mean), color = 'red', size = 2) +
    scale_y_log10()
})
```

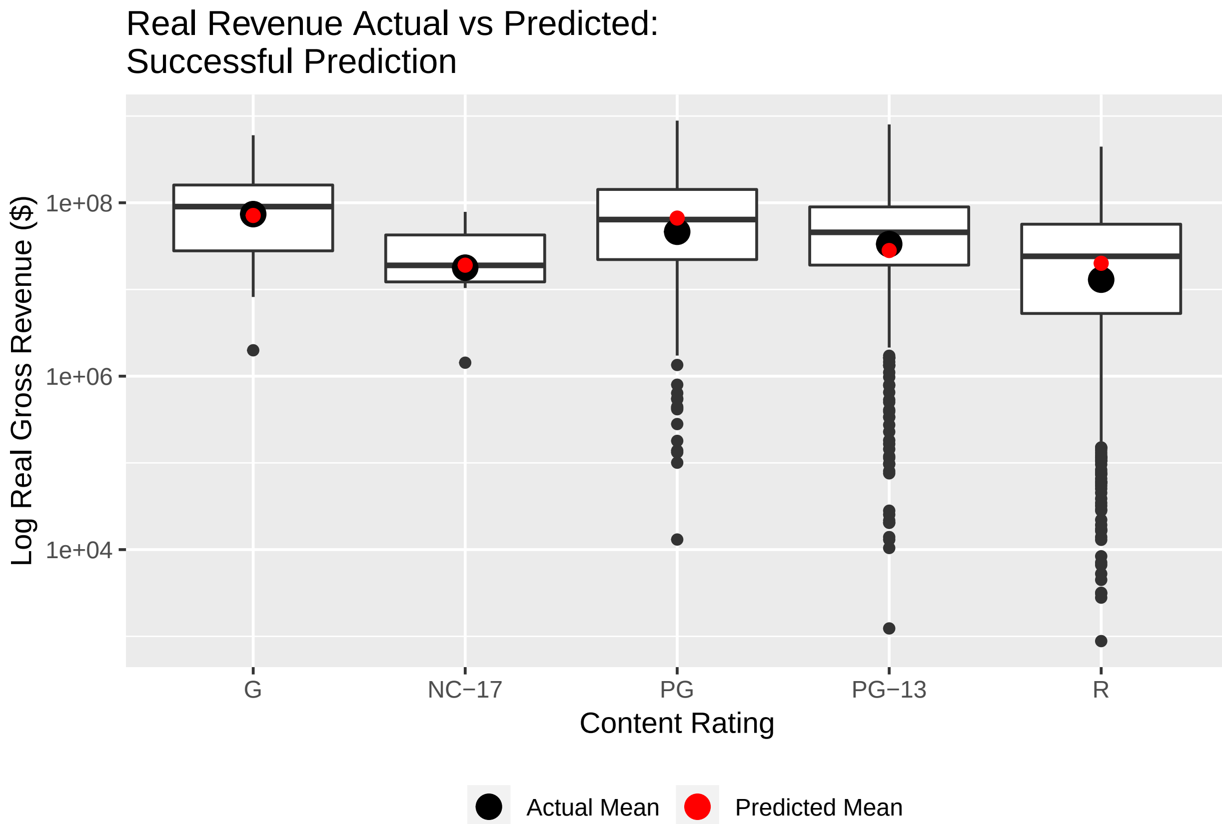


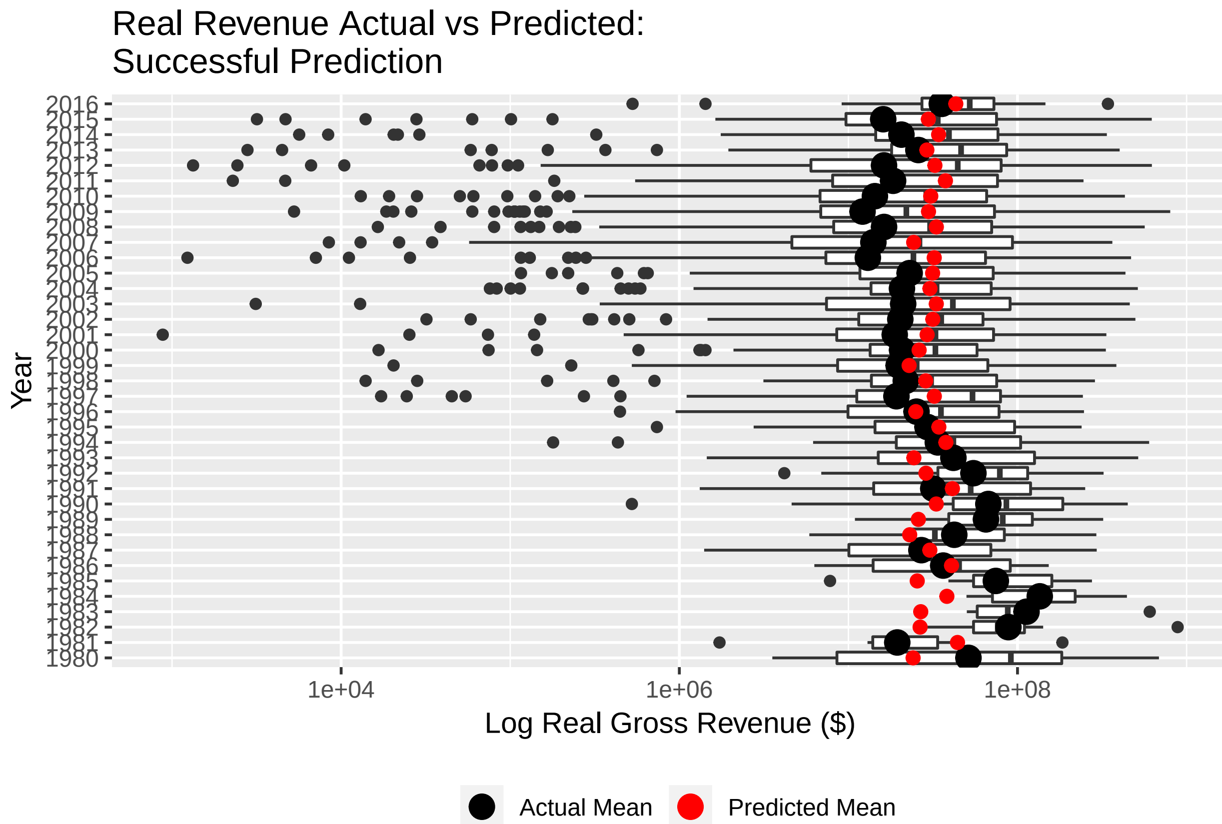
Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
1e+04
1e+06
1e+08
Log Real Budget ($)
Actual
Predicted

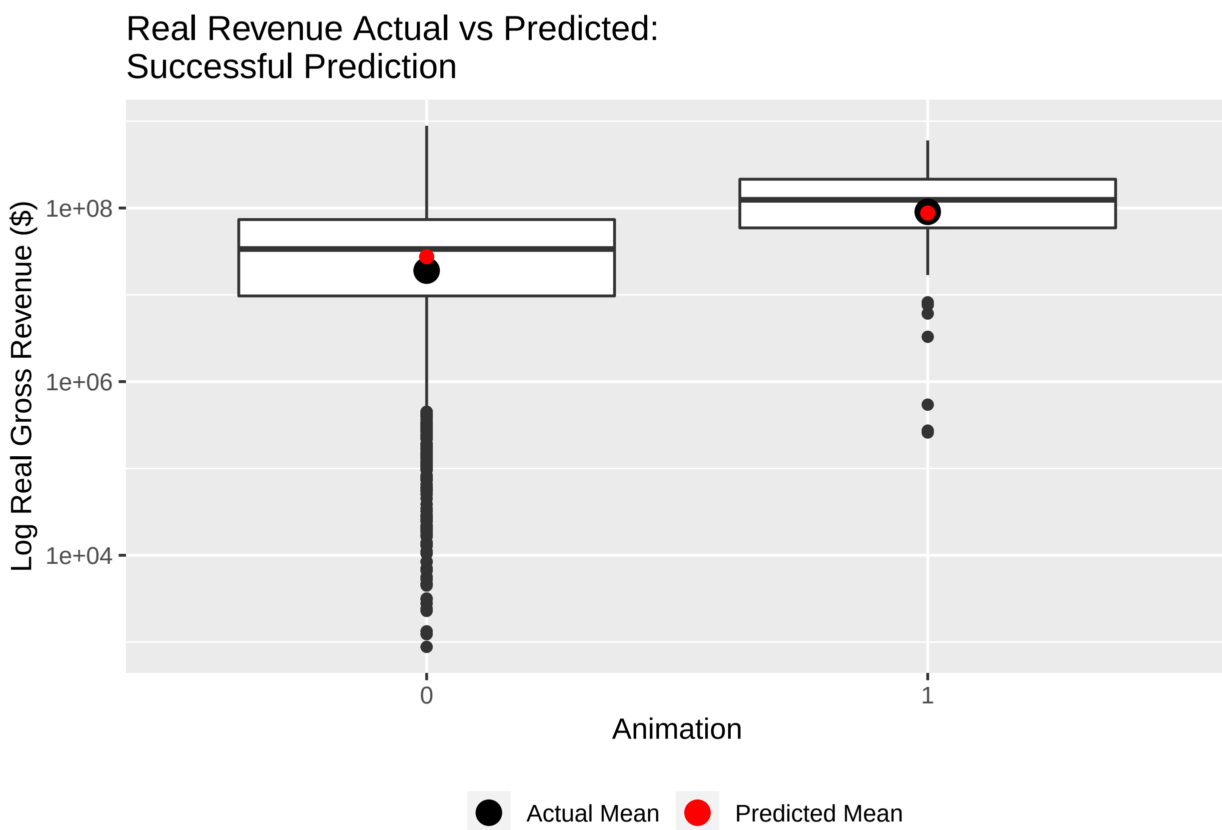

Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
2
3
5
Log IMDB Score
Actual
Predicted

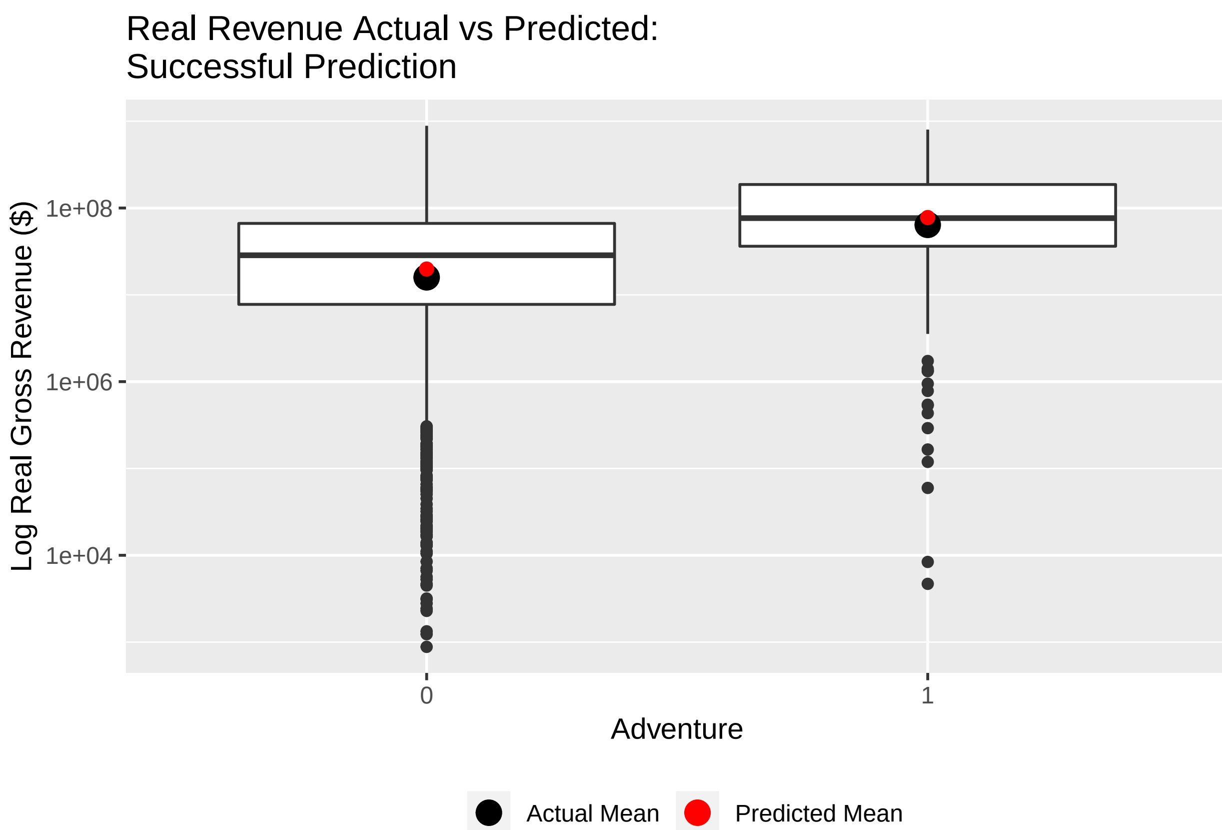


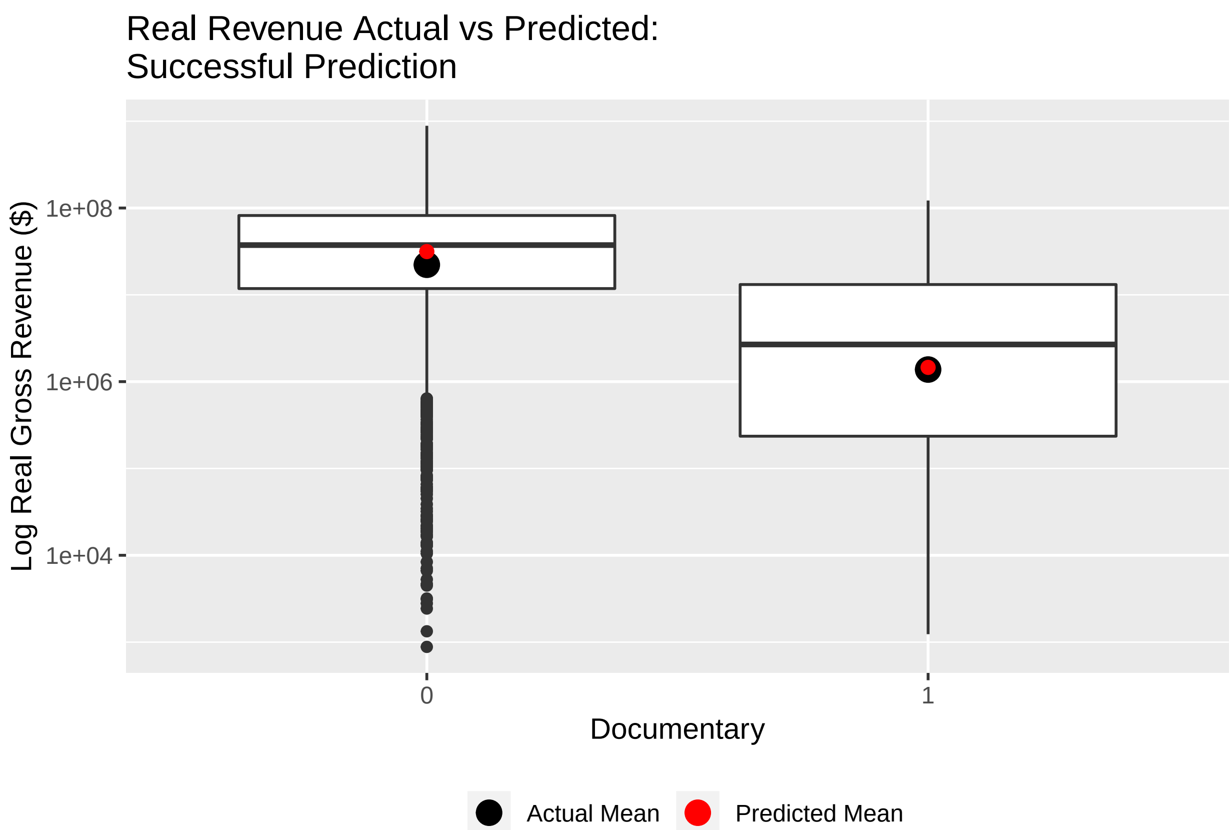
Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
1e+01
1e+02
1e+03
1e+04
1e+05
Log Cast Total Facebook Likes
Actual
Predicted


Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
G
NC−17
PG
PG−13
R
Content Rating
Actual Mean
Predicted Mean


Real Revenue Actual vs Predicted:
Successful Prediction
Year
2016
2015
2014
2013
2012
2011
2010
2009
2008
2007
2006
2005
2004
2003
2002
2001
2000
1999
1998
1997
1996
1995
1994
1993
1992
1991
1990
1989
1988
1987
1986
1985
1984
1983
1982
1981
1980
1e+04
1e+06
1e+08
Log Real Gross Revenue ($)
Actual Mean
Predicted Mean

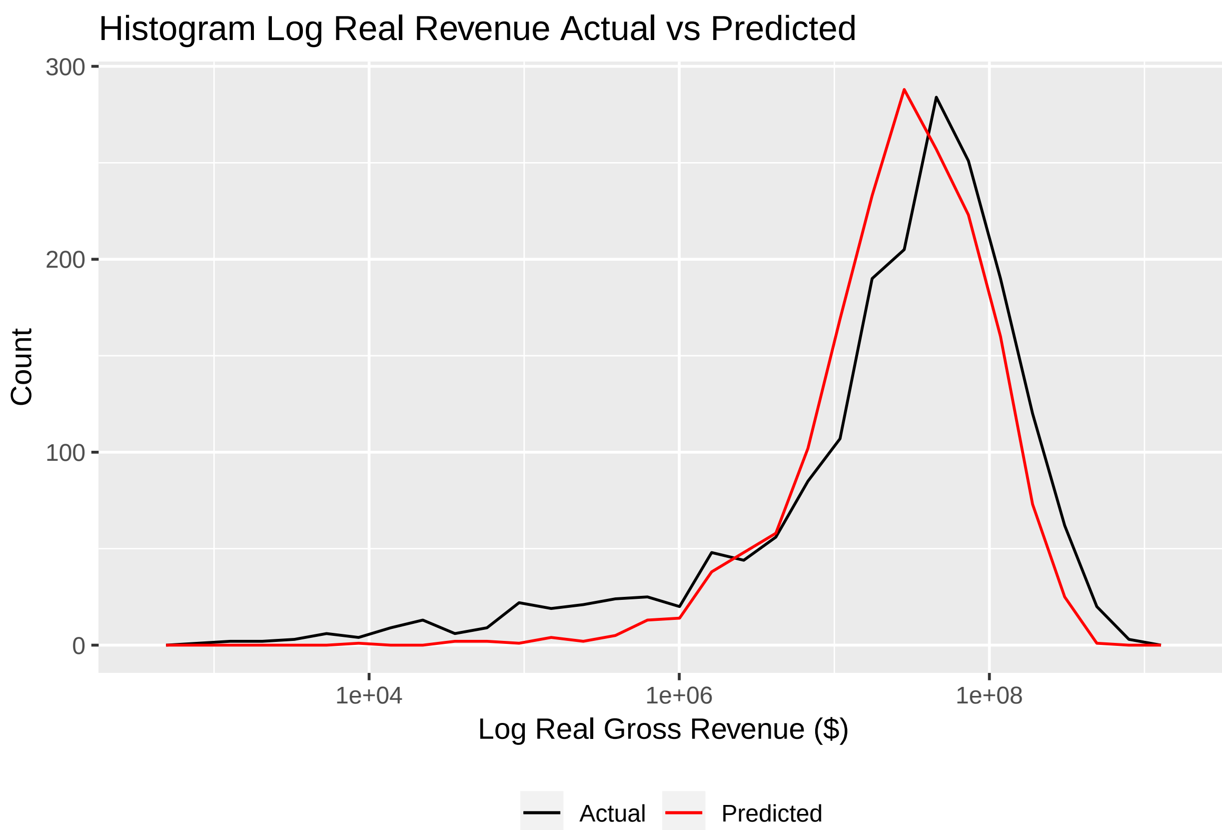

Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
0
1
Animation
Actual Mean
Predicted Mean


Real Revenue Actual vs Predicted:
Successful Prediction
Log Real Gross Revenue ($)
1e+08
1e+06
1e+04
0
1
Adventure
Actual Mean
Predicted Mean




```
train %>%
  add_predictions(mod_simple2, 'lpred') %>%
  mutate(pred = 10^lpred) %>%
  ggplot() +
  geom_freqpoly(aes(x = real_gross, color = 'Actual')) +
  geom_freqpoly(aes(x = pred, color = 'Predicted')) +
  scale_x_log10() +
  theme(legend.position="bottom") +
  labs(x = 'Log Real Gross Revenue ($)', y = 'Count', title = 'Histogram Log Real Revenue Actual vs Pre
  scale_color_manual(name = '', values = c(Actual = 'black', Predicted = 'red'))
```

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

## Warning: Removed 132 rows containing non-finite values (stat_bin).
```



Compare Models

- Somewhat of a toss-up between model without genre and model with some genre variables. Predictions similar, random relationships with resiudals.
- Histogram of actual vs predicted revenue is, if anything, more similar for the no genre model.
- Genres are captured by other variables: for example, genre and budget vary together significantly, as well as genre and content rating, so those effects are captured. If I was a movie executive, picking genre is still tied up in this because that likely effects the budget and content rating, which are significant and strong predictors.
  - See graphs at end of EDA: compare genre and budget and content rating. For budget, ranking of genre almost identical to ranking based off of revenue
- Very similar RMSE (.621 vs .623) and R-squared (.48 vs .46)
- Anova shows that the more complex genre model is significantly different from the simpler model without genre.
- However, given that the predictions etc. are so similar, it is better to go with the simpler model. Worry about over-fitting with the genre model. Especially because some of the genres don't have that many observations.
- Main take away is that budget has the strongest, most significant relationship with revenue. The model with just budget had a RMSE that was only slightly higher (.65 vs .62) and an R-squared that was only slightly lower (.45 vs .46) than the models with more variables. Also, the other variables that we add to the model had fairly close to random relationships with the residuals from the model with just budget.
- However, predicted vs actual histogram of real gross shows that it is not perfect. Model with the three extra variables does a better job predicting.

Final Statistics on Test Set

```r
rmse(mod_simple2, data = test)
```

```
## [1] 0.606151
```

```r
dev.off()
```

```
## pdf
##   3
```