Bayesian modeling and prediction for movies

Jeff Spoelstra

2016-09-10

Setup

Load packages

```
library(ggplot2)
library(dplyr)
library(statsr)
library(BAS)
library(grid)
library(gridExtra)

# Set random number seed to get consistent results.
set.seed(3514)
```

Load data

```
load("movies.Rdata")
```

Part 1: Data

The raw data provided for this analysis consists of audience and critic review scores for 651 movies released during the years prior to 2016. The data comes from the Rotten Tomatoes (http://www.rottentomatoes.com) and IMDb (http://www.imdb.com) web sites. In addition to review scores, the data contains several other variables for descriptive information regarding each movie such as genre, running time, MPAA rating, production studio, Oscar nominations, and more.

The raw data is not a complete list of all movies released prior to 2016. Instead, it is a random sample taken from the full data set. No information is available regarding the specific sampling method used. It is assumed for the sake of this analysis that the conclusions drawn are generalizable to the population of all movies and that there is no bias introduced by the sampling method.

A possible non-independent bias may arise with regard to movie sequels (for example, all of the *Star Wars* episodes) whereby the popularity of a sequel movie may be influenced by that of the prior releases. A 2011 study by metacritic (http://www.metacritic.com/feature/movie-sequels-remakes-and-adaptations)

showed that critics and audiences tended to rate sequels and other derivative works higher than the original work. On the other hand, a 2016 chart produced by reddit (https://www.reddit.com/r/dataisbeautiful/comments/4shlda/movie_sequels_19902016_compared_to_their/) of ratings for original movies versus their sequls (considering 1990-2016 films only) appears to indicate just the opposite. In light of

1. these conflicting results,
2. not knowing how sequels were treated in the data sampling method, and
3. not having a definitive means of identifying sequels and originals in the raw data set so as to try to identify a bias,

it is assumed that no bias exists for the purpose of this analysis.

Overall, it is assumed that the individual movie observations are independent and identically distributed. The size of the general population of movies produced is difficult to quantify exactly, however the IMDB web site lists over 105,000 feature film titles produced in the U.S. alone. A sample of 651 movies is certainly well below the 10% of the population threshold needed for independence.

As the raw data is a sample taken from existing data, this is an observational study and no causal relationships can be inferred or assumed from the conclusions drawn.

Part 2: Data manipulation

Data Codebook

The following is a copy of the codebook provided with the raw data.

1. `title` : Title of movie
2. `title_type` : Type of movie (Documentary, Feature Film, TV Movie)
3. `genre` : Genre of movie (Action & Adventure, Comedy, Documentary, Drama, Horror, Mystery & Suspense, Other)
4. `runtime` : Runtime of movie (in minutes)
5. `mpaa_rating` : MPAA rating of the movie (G, PG, PG-13, R, Unrated)
6. `studio` : Studio that produced the movie
7. `thtr_rel_year` : Year the movie is released in theaters
8. `thtr_rel_month` : Month the movie is released in theaters
9. `thtr_rel_day` : Day of the month the movie is released in theaters
10. `dvd_rel_year` : Year the movie is released on DVD
11. `dvd_rel_month` : Month the movie is released on DVD
12. `dvd_rel_day` : Day of the month the movie is released on DVD
13. `imdb_rating` : Rating on IMDB
14. `imdb_num_votes` : Number of votes on IMDB
15. `critics_rating` : Categorical variable for critics rating on Rotten Tomatoes (Certified Fresh, Fresh, Rotten)
16. `critics_score` : Critics score on Rotten Tomatoes
17. `audience_rating` : Categorical variable for audience rating on Rotten Tomatoes (Spilled, Upright)

18. `audience_score` : Audience score on Rotten Tomatoes
19. `best_pic_nom` : Whether or not the movie was nominated for a best picture Oscar (no, yes)
20. `best_pic_win` : Whether or not the movie won a best picture Oscar (no, yes)
21. `best_actor_win` : Whether or not one of the main actors in the movie ever won an Oscar (no, yes) - note that this is not necessarily whether the actor won an Oscar for their role in the given movie
22. `best_actress_win` : Whether or not one of the main actresses in the movie ever won an Oscar (no, yes) - not that this is not necessarily whether the actresses won an Oscar for their role in the given movie
23. `best_dir_win` : Whether or not the director of the movie ever won an Oscar (no, yes) - not that this is not necessarily whether the director won an Oscar for the given movie
24. `top200_box` : Whether or not the movie is in the Top 200 Box Office list on BoxOfficeMojo (no, yes)
25. `director` : Director of the movie
26. `actor1` : First main actor/actress in the abridged cast of the movie
27. `actor2` : Second main actor/actress in the abridged cast of the movie
28. `actor3` : Third main actor/actress in the abridged cast of the movie
29. `actor4` : Fourth main actor/actress in the abridged cast of the movie
30. `actor5` : Fifth main actor/actress in the abridged cast of the movie
31. `imdb_url` : Link to IMDb page for the movie
32. `rt_url` : Link to Rotten Tomatoes page for the movie

Data Processing

In addition to the variables in the raw data set, the following variables were synthesized for this analysis.

33. `feature_film` : "yes" if `title_type` is "Feature Film", "no" otherwise
34. `drama` : "yes" if `genre` is "Drama", "no" otherwise
35. `mpaa_rating_R` : "yes" if `mpaa_rating` is "R", "no" otherwise
36. `oscar_season` : "yes" if movie is released in November, October, or December (based on `thtr_rel_month`), "no" otherwise
37. `summer_season` : "yes" if movie is released in May, June, July, or August (based on `thtr_rel_month`), "no" otherwise

The following code creates the new variables in the data set.

```
# Create new variables in the data set.
movies <- movies %>%
          mutate(feature_film=as.factor(ifelse(title_type == 'Feature Film', 'yes', '
no'))) %>%
          mutate(drama=as.factor(ifelse(genre == 'Drama', 'yes', 'no'))) %>%
          mutate(mpaa_rating_R=as.factor(ifelse(mpaa_rating == 'R', 'yes', 'no'))) %>
%
          mutate(oscar_season=as.factor(ifelse(thtr_rel_month %in% c(10:12), 'yes', '
no'))) %>%
          mutate(summer_season=as.factor(ifelse(thtr_rel_month %in% c(5:8), 'yes', 'n
o')))
```

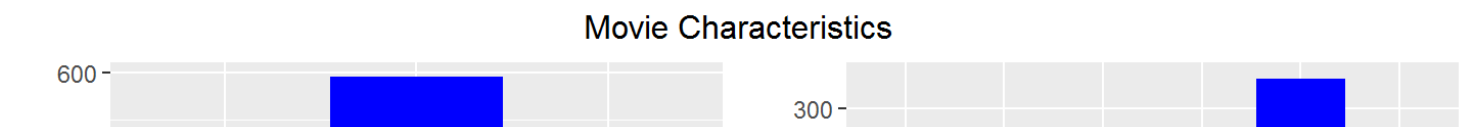
The entry for the movie "The End of America" was found to have a `N/A` value in the `runtime` variable. This observation was removed from the data set.

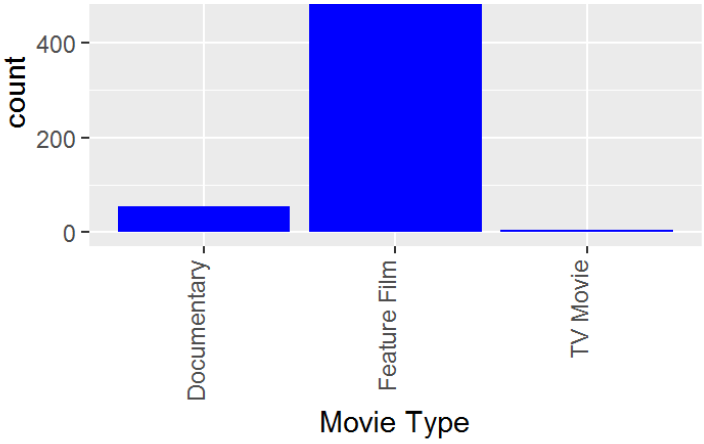
```
# Remove N/A values in key variable in the data set.
movies <- filter(movies, !is.na(runtime))
```

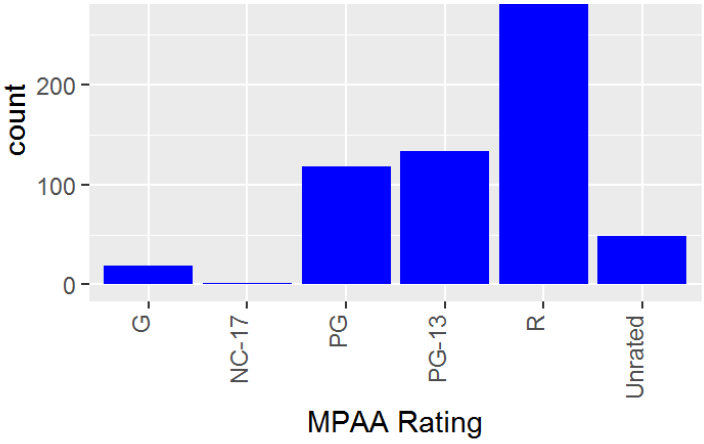
Part 3: Exploratory data analysis

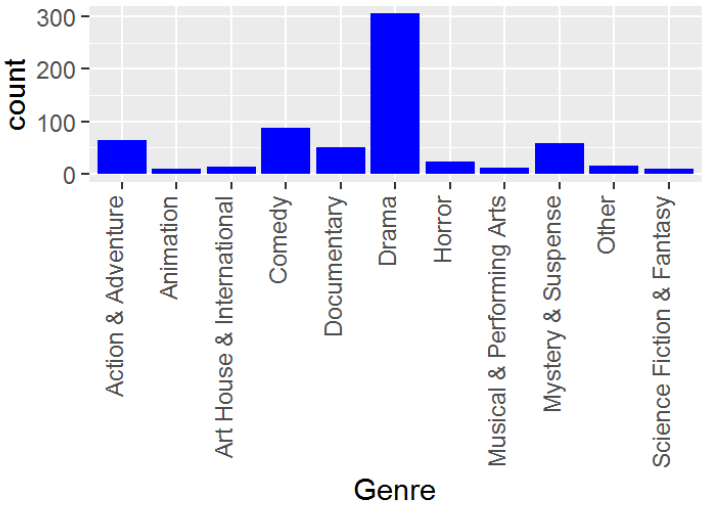
After pre-processing the data, there were 650 movies in the final data set for analysis. The following charts show a breakdown of various movie characteristics.

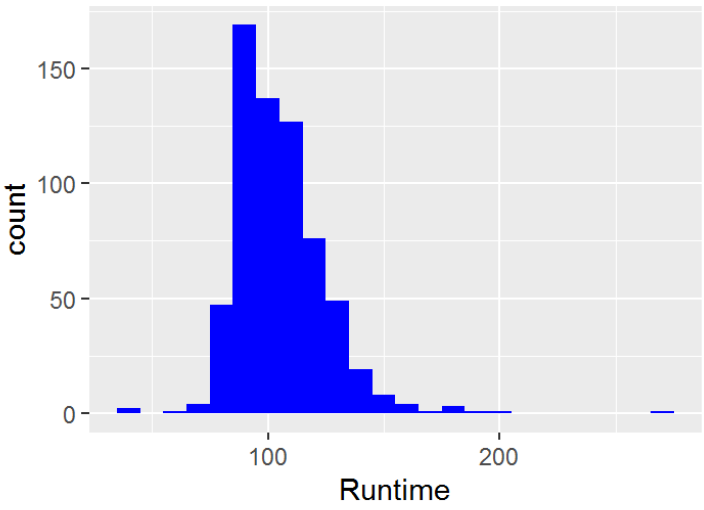
```
# Create histograms of some of the key movie characteristic data.
p1 <- ggplot(data=movies, aes(x=genre)) +
      geom_bar(fill="blue") +
      xlab("Genre") +
      theme(axis.text.x=element_text(angle=90, hjust=1, vjust=0))
p2 <- ggplot(data=movies, aes(x=title_type)) +
      geom_bar(fill="blue") +
      xlab("Movie Type") +
      theme(axis.text.x=element_text(angle=90, hjust=1, vjust=0))
p3 <- ggplot(data=movies, aes(x=mpaa_rating)) +
      geom_bar(fill="blue") +
      xlab("MPAA Rating") +
      theme(axis.text.x=element_text(angle=90, hjust=1, vjust=0))
p4 <- ggplot(data=movies, aes(x=runtime)) +
      geom_histogram(binwidth=10, fill="blue") +
      xlab("Runtime")
p5 <- ggplot(data=movies, aes(x=thtr_rel_month)) +
      geom_histogram(binwidth=1, fill="blue") +
      scale_x_continuous(breaks=c(1:12)) +
      scale_y_continuous(limits=c(0, 100)) +
      geom_vline(xintercept=c(5, 8, 10, 12), colour='green', linetype='longdash') +
      geom_text(label='Summer', x=6.5, y=95, hjust='center', size=3) +
      geom_text(label='Oscar', x=11, y=95, hjust='center', size=3) +
      geom_text(label='Season', x=6.5, y=89, hjust='center', size=3) +
      geom_text(label='Season', x=11, y=89, hjust='center', size=3) +
      xlab("Theater Release Month")
p6 <- ggplot(data=movies, aes(x=audience_score)) +
      geom_histogram(binwidth=5, fill="blue") +
      geom_vline(xintercept=mean(movies$audience_score), colour='green', linetype='lo
ngdash') +
      geom_text(label='Mean', x=55, y=60, hjust='center', size=3) +
      xlab("Audience Score")
p7 <- ggplot(data=movies, aes(x=imdb_rating)) +
      geom_histogram(binwidth=1, fill="blue") +
      geom_vline(xintercept=mean(movies$imdb_rating), colour='green', linetype='longd
ash') +
      geom_text(label='Mean', x=6, y=225, hjust='center', size=3) +
      xlab("IMDb Rating")
p8 <- ggplot(data=movies, aes(x=critics_score)) +
      geom_histogram(binwidth=5, fill="blue") +
      geom_vline(xintercept=mean(movies$critics_score), colour='green', linetype='lon
gdash') +
      geom_text(label='Mean', x=51, y=50, hjust='center', size=3) +
      xlab("Critics Score")
grid.arrange(p2, p3, p1, p4, p5, p6, p7, p8, nrow=4,
             top="Movie Characteristics")
```

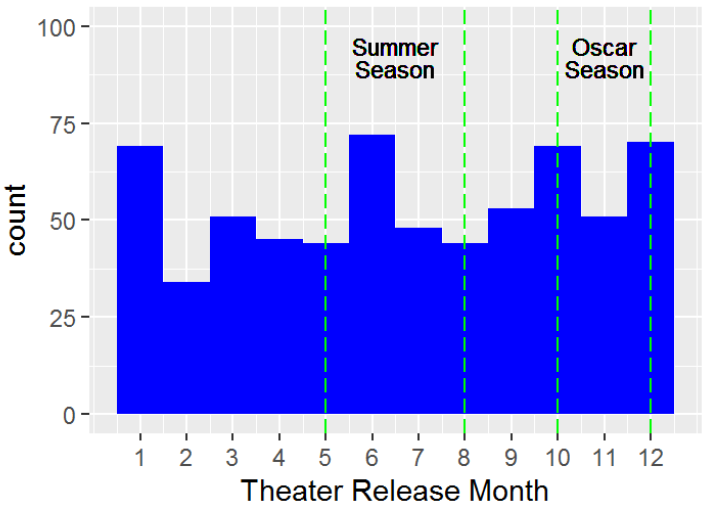


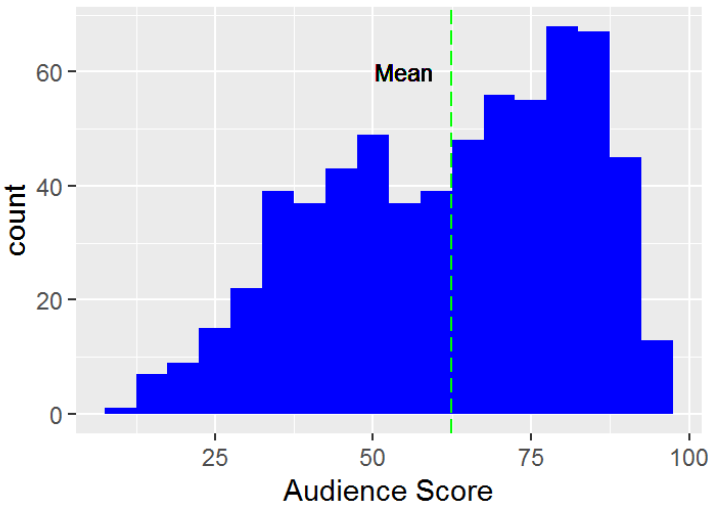

count
400
200
0
Documentary
Feature Film
TV Movie
Movie Type

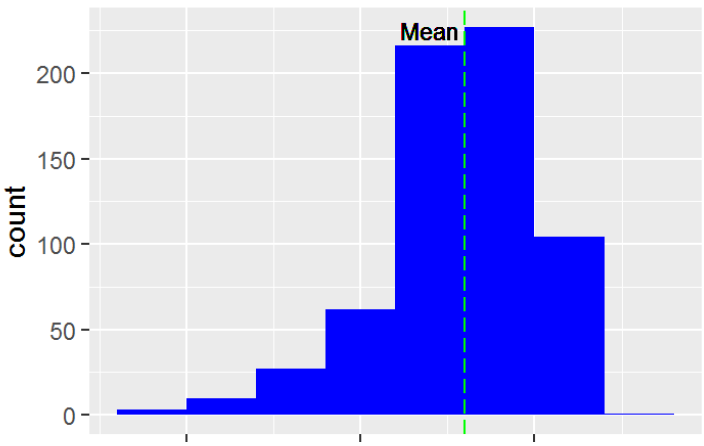

count
200
100
0
G
NC-17
PG
PG-13
R
Unrated
MPAA Rating

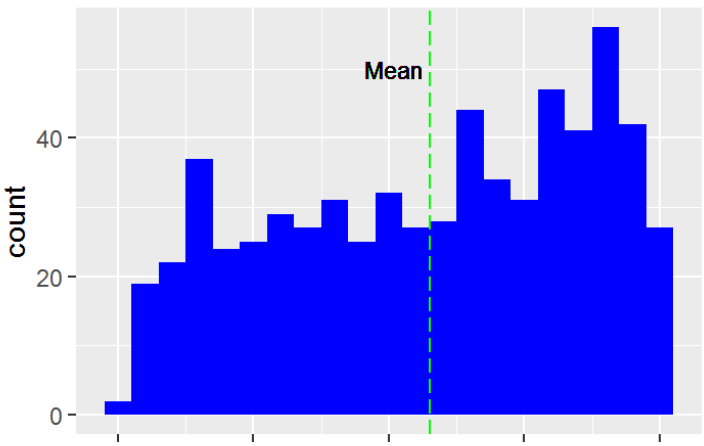

count
300
200
100
0
Action & Adventure
Animation
Art House & International
Comedy
Documentary
Drama
Horror
Musical & Performing Arts
Mystery & Suspense
Other
Science Fiction & Fantasy
Genre


count
150
100
50
0
100
200
Runtime


count
100
75
50
25
0
Summer Season
Oscar Season
1
2
3
4
5
6
7
8
9
10
11
12
Theater Release Month


count
60
40
20
0
Mean
25
50
75
100
Audience Score


count
200
150
100
50
0
Mean


count
40
20
0
Mean

The most common movie in the data set is a rated R, feature film, drama with a runtime of around 90 minutes. Approximately 32% of the movies were released during the summer season, and 29% of the movies were released during the Oscar season.

The distribution of audience scores shows an interesting bi-modality. As this is the value to be predicted by the model, one would like to see a normal distribution for this.

Part 4: Modeling

The modeling goal was to predict the popularity of a movie - as quantified by the `audience_score` variable - using a linear regression model and bayesian model averaging.

The initial set of predictors chosen for the model were:

- `runtime`
- `thtr_rel_year`
- `imdb_rating`
- `imdb_num_votes`
- `critics_score`
- `best_pic_nom`
- `best_pic_win`
- `best_actor_win`
- `best_actress_win`
- `best_dir_win`
- `top200_box`
- `feature_film`
- `drama`
- `mpaa_rating_R`
- `oscar_season`
- `summer_season`

As the first step, the data set was reduced to include only the response variable and the initial set of predictor variables.

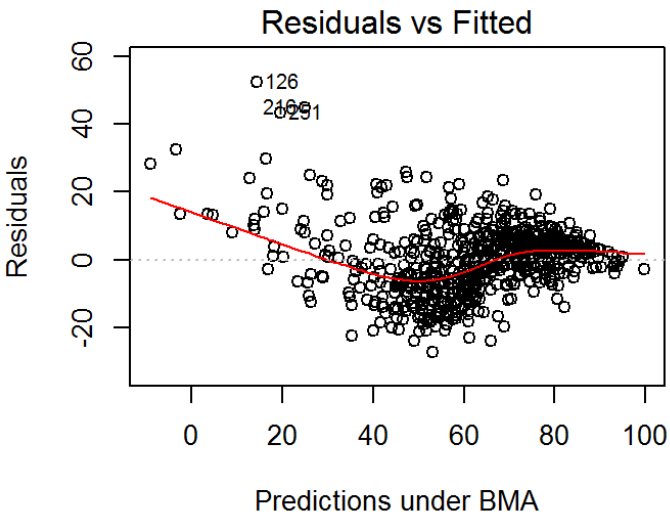
```
# Shrink the data set variables to only the response variables and the anticipated
# predictors.
movies <- select(movies, runtime, thtr_rel_year, imdb_rating, imdb_num_votes,
                 critics_score, audience_score, best_pic_nom, best_pic_win,
                 best_actor_win, best_actress_win, best_dir_win, top200_box,
                 feature_film, drama, mpaa_rating_R, oscar_season, summer_season)
```

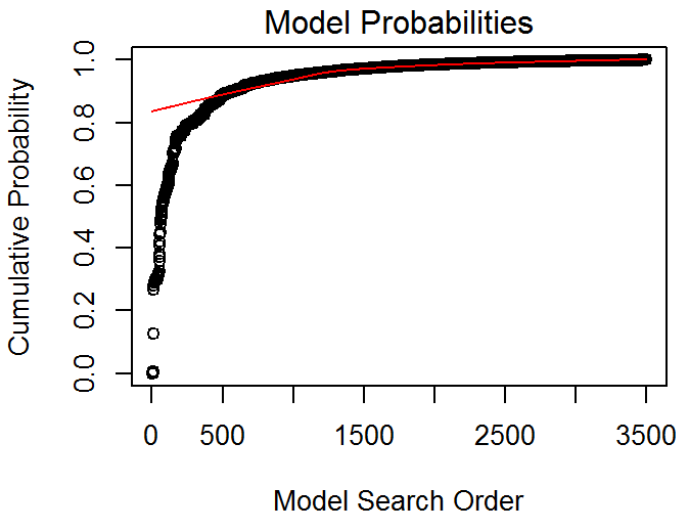
A bayesian linear regression model was created using all of the initial predictor variables. Because of the large number of predictors (leading to a very large number of possible model combinations), the Markov Chain Monte Carlo (MCMC) method was used for sampling models during the fitting process and the prior

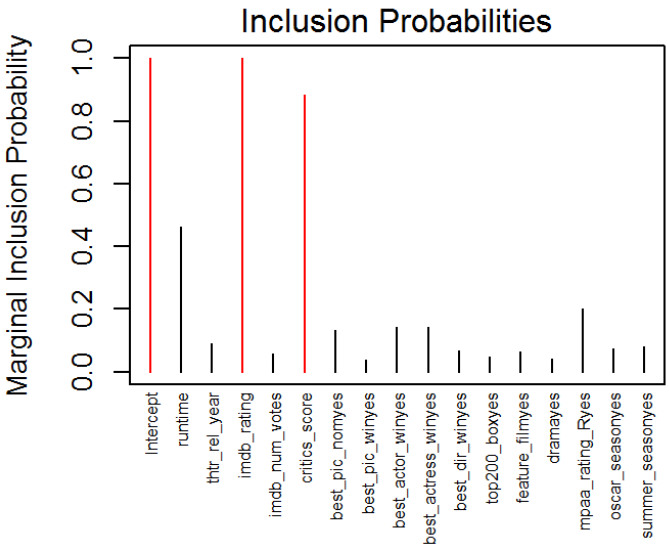
probabilities for the regression coefficients were assigned using the Zellner-Siow Cauchy distribution. A uniform distribution was used for the prior probabilities for all models.

```
# Create a basian linear regression model to derive audience_score from the
# initial set of predictors.
basLM1 <- bas.lm(audience_score ~ ., data=movies, method='MCMC',
                 prior='ZS-null', modelprior=uniform())
```

```
par(mfrow=c(2,2))
plot(basLM1, which=c(1, 2), ask=FALSE)
plot(basLM1, which=4, ask=FALSE, cex.lab=0.5)
```







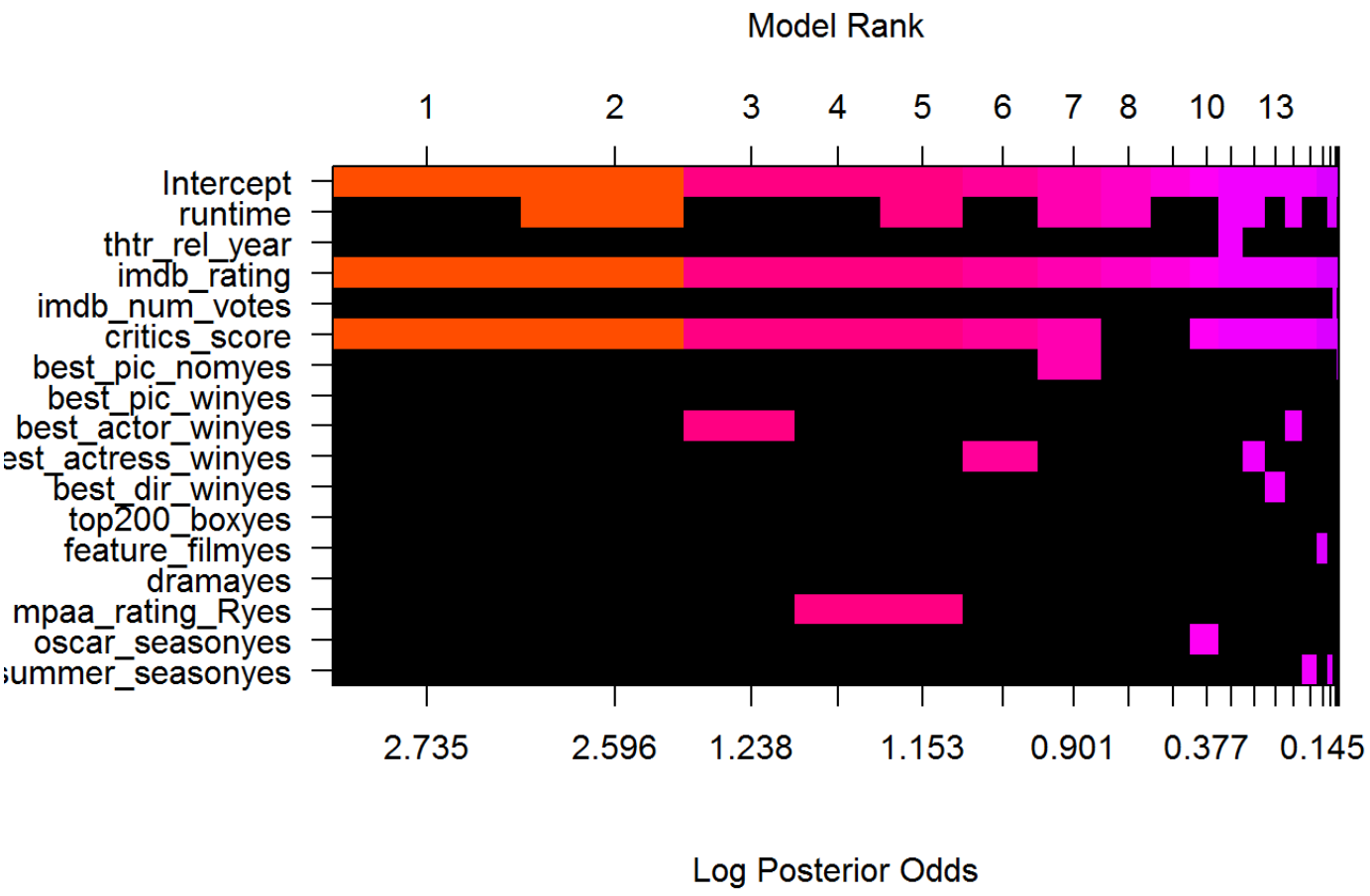
Shown above are diagnostic charts of the regression fitting process.

The top-left chart does not show a random scattering of the residuals versus the fitted values as one would like to see. There is a wave pattern indicating that the model tended to predict too low for ratings under a value of 30 and over a value of about 75. In between 30 and 75, the model tended to predict high. Above a rating of 75, the residuals appear to scatter randomly. This would indicate that perhaps some of the other predictor candidates should be further evaluated for inclusion in the model, or perhaps there is something else affecting the ratings that is not accounted for at all by the initial set of chosen predictors.

The top-right chart shows that the model posterior probability density leveled off at 1 after approx. 3,000 (3,500 to be exact) model combinations had been sampled. The number of models was capped at this point rather than proceed with the evaluation of all 2^16 possible model combinations.

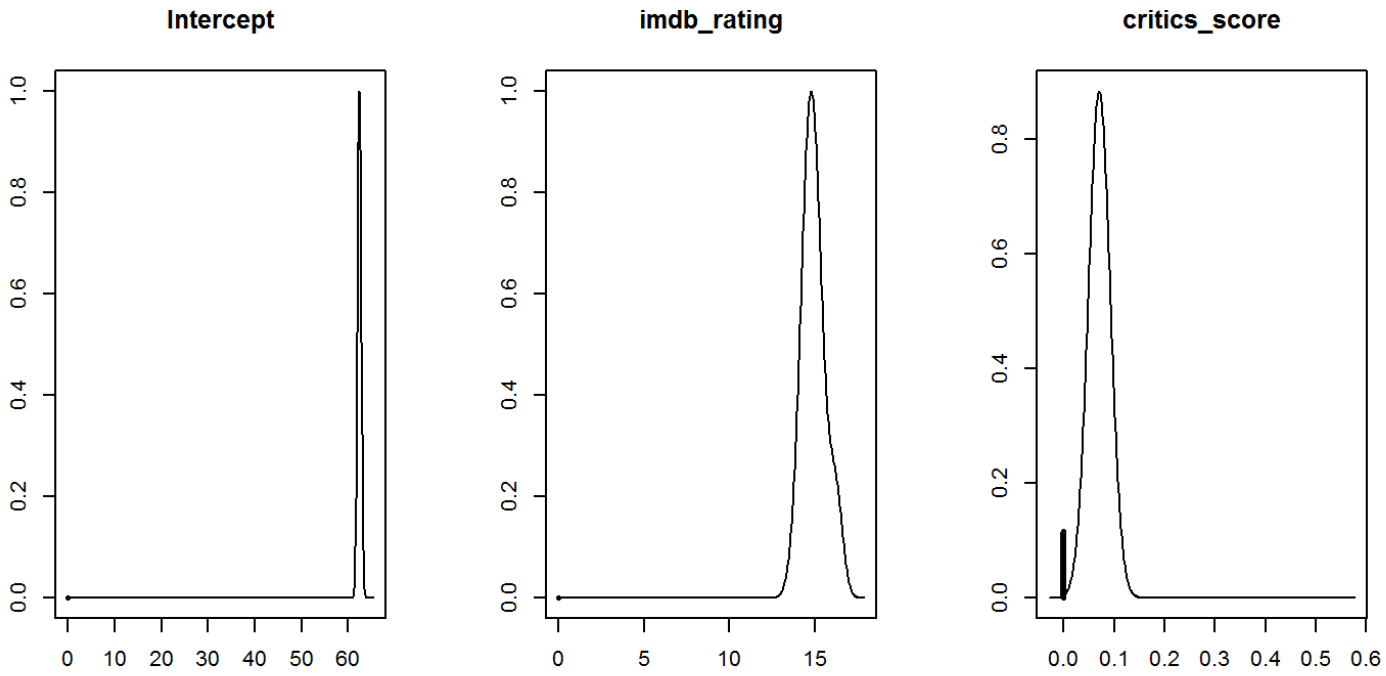
The bottom-left chart shows the inclusion probabilities for each of the predictors. How each predictor was used is shown graphically below for the top 20 models tested.

```
image(basLM1, rotate=FALSE)
```



The model with the highest posterior probability (0.1404) contains only two of the predictors: `imbd_rating` and `critics_score`. The posterior distributions of the regression coefficients are shown below. The vertical line at 0 on the X axis indicates the posterior probability of the coefficient being zero (i.e., that the particular predictor would not be included in any model).

```
par(mfrow=c(1,3))
plot(coefficients(basLM1), subset=c(1, 4, 6), ask=FALSE)
```



It is noteworthy that the distribution for `imdb_rating` does not visually look entirely normal as there appears to be a small non-symmetric bump on the right side. Possible reasons for this were not explored; although it could be related to the wave pattern shown on the residuals plot above or the non-normal distribution of audience score values.

Also noteworthy is that the regression intercept is right at the median of the range of `audience_score` with only positive coefficients for the other regression coefficients - meaning that a predicted audience score cannot be below 65. This almost certainly contributes to the wave pattern in the residuals plot above.

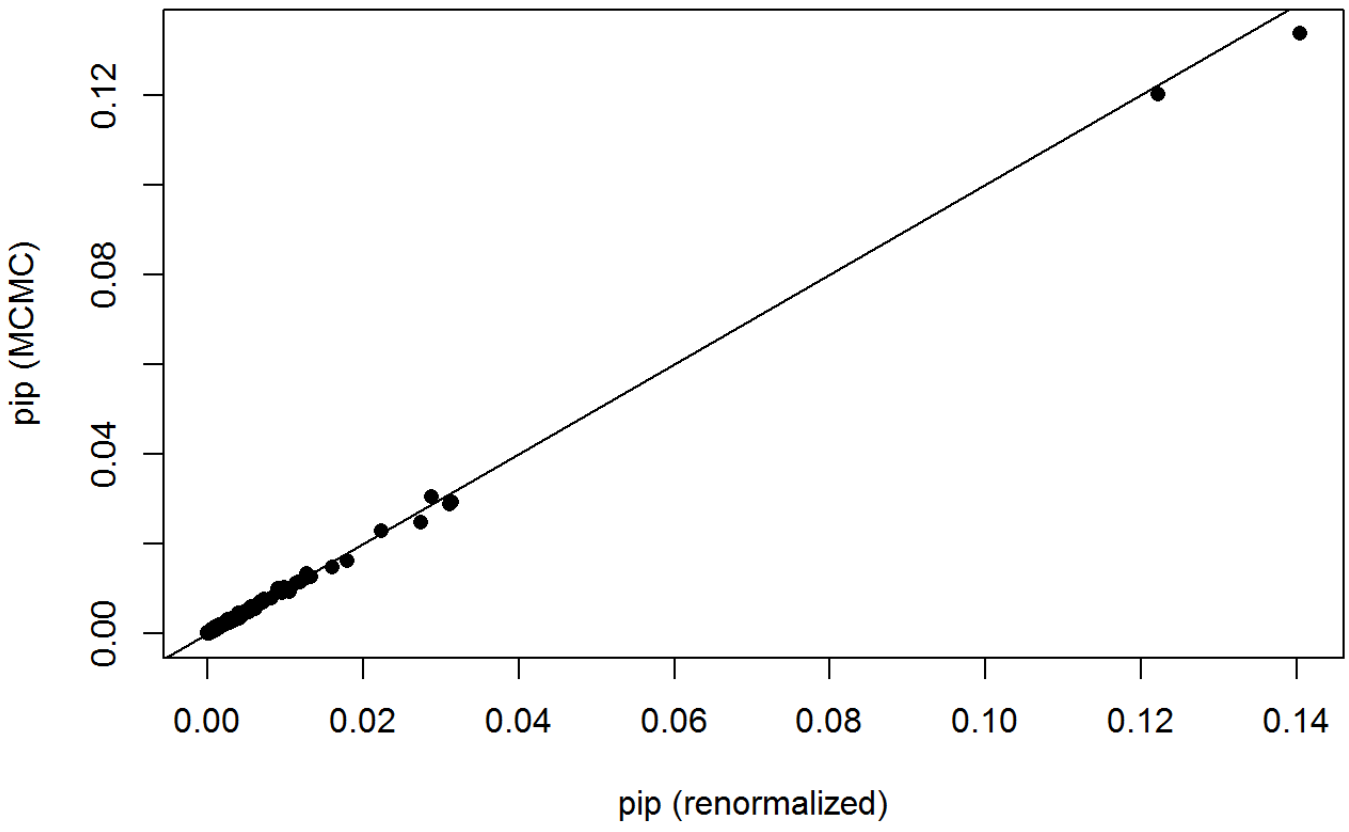
Credible intervals were determined by using the model to predict the same audience scores as were used to fit the model originally. Below are shown the intervals for the fitted and predicted values.

```
# Use the model to predict the audience scores in the analysis data set and
# determine credible intervals for both the fitted/predicted values.
BMA_basLM1 = predict(basLM1, estimator="BMA", se.fit=TRUE)
BMA_confint_fit = confint(BMA_basLM1, parm="mean")
BMA_confint_pred = confint(BMA_basLM1, parm="pred")
head(cbind(BMA_confint_fit, BMA_confint_pred), 10)
```

```
##          2.5  %  97.5  %     mean   2.5  %   97.5  %     pred
##  [1,] 45.32129 49.10278 47.20271 27.70732  66.87554 47.20271
##  [2,] 74.95756 78.86044 77.15016 57.98057  97.30029 77.15016
##  [3,] 79.42336 83.69130 81.54769 62.14936 101.33407 81.54769
##  [4,] 70.48988 75.62114 73.25034 52.79367  93.01857 73.25034
##  [5,] 38.62553 41.84064 40.24054 20.05164  59.47792 40.24054
##  [6,] 82.65753 87.49256 84.94767 65.17872 105.09630 84.94767
##  [7,] 69.56713 74.67067 72.24660 52.31558  92.37070 72.24660
##  [8,] 42.34002 47.51150 44.83727 26.15274  65.13206 44.83727
##  [9,] 78.06836 82.11057 80.13008 60.63771  99.66641 80.13008
## [10,] 62.91609 67.30672 65.35915 45.45142  85.49192 65.35915
```

The chart below shows that the posterior model probabilities follow a normal distribution. Had this not been the case then more MCMC iterations may have been necessary to get a normal distribution.

```
diagnostics(basLM1, type="model",  pch=16)
```



Part 5: Prediction

The predictive capability of bayesian model averaging was tested using data for a movie that was not included in the analysis data set. The movie chosen was *Deadpool* which was released early in 2016. An early release date was chosen to give time for the movie to have been in the theaters for a while and for the ratings to mature. The information for this movie was obtained from the IMDb and Rotten Tomatoes web sites in order to be consistent with the analysis data (*Deadpool* on IMDb (http://www.imdb.com/title/tt1431045/?ref_=fn_al_tt_1) and *Deadpool* on Rotten Tomatoes (https://www.rottentomatoes.com/m/deadpool)).

The results are shown below.

```
# Create a data frame with the predictor variable data for the movie rating to
# be predicted.
dfDeadpool <- data.frame(runtime=108,
                         thtr_rel_year=2016,
                         imdb_rating=8.1,
                         imdb_num_votes=500049,
                         critics_score=84,
                         audience_score=0,
                         best_pic_nom=factor("no", levels=c("no", "yes")),
                         best_pic_win=factor("no", levels=c("no", "yes")),
                         best_actor_win=factor("no", levels=c("no", "yes")),
                         best_actress_win=factor("no", levels=c("no", "yes")),
                         best_dir_win=factor("no", levels=c("no", "yes")),
                         top200_box=factor("no", levels=c("no", "yes")),
                         feature_film=factor("yes", levels=c("no", "yes")),
                         drama=factor("no", levels=c("no", "yes")),
                         mpaa_rating_R=factor("yes", levels=c("no", "yes")),
                         oscar_season=factor("no", levels=c("no", "yes")),
                         summer_season=factor("no", levels=c("no", "yes")))

# Make a prediction of audience_score using bayesian model averaging.
BMA_basLM1_DP <- predict(basLM1, newdata=dfDeadpool, estimator="BMA", se.fit=TRUE)

# Calculate 95% margin of error for the prediction interval.
BMA_basLM1_predME <- qt(0.95, df=BMA_basLM1_DP$se.bma.pred[1]) *
                     mean(BMA_basLM1_DP$se.bma.pred)

# Show prediction results.
df <- data.frame(t="Deadpool",
                 p=sprintf("%2.1f", BMA_basLM1_DP$Ybma),
                 i=sprintf("%2.1f - %2.1f", BMA_basLM1_DP$Ybma - BMA_basLM1_predME,
                           BMA_basLM1_DP$Ybma + BMA_basLM1_predME),
                 r=84)
colnames(df) <- c("Movie Title", "Predicted Rating", "95% Prediction Interval",
                  "Actual Rating")
print(df)
```

```
##   Movie Title Predicted Rating 95% Prediction Interval Actual Rating
## 1    Deadpool             87.9            69.6 - 106.2            84
```

The true audience score for the movie (taken from the IMDb web site) is 84. The model prediction is 88 with a 95% prediction interval of 69.6 to 106.2 (calculated using a Student t distribution with degrees of freedom and standard error of 10.11).

Part 6: Conclusion

In this analysis, a parsimonious, linear regression model using bayesian model averaging was created that proved to have some capability for predicting movie popularity based on certain movie characteristics.

But, there is much room for further analysis in at least the following areas.

1. Why is does the `imdb_rating` regression coefficient distribution not show a normal curve? Is it related to the non-normal distribution of audience score values?
2. What is the nature of the wave pattern in the residuals plot?
3. Would it be better to create separate models for each movie type or genre? Would doing so eliminate the non-normal distribution of audience score values in the analysis data?
4. How should sequels be handled?

Citations

1. Much of the introductory and exploratory analysis work was replicated from an earlier analysis (by the author) exploring a typical frequentist approach to linear regression modeling to make similiar predictions of movie popularity. Some of the conclusions drawn by that analysis were applied to this one as well.