# bayesian project

*Schwinn(Xuan) Chen*

*July 11, 2017*

## Load packages and data

```
library(dplyr)
library(statsr)
library(BAS)
```

```
## Warning: package 'BAS' was built under R version 3.5.1
```

```
library(ggplot2)
load("movies.Rdata")
```

---

# Search for the best model to predict Rotten Tomatos audience scores of movies

In this report, we will use Bayesian Statistics to search for the best model to predict Rotten Tomatos audience scores of movies. * * *

# Part 1: Data

As shown in the codebook, the data set is comprised of 651 randomly sampled movies produced and released before 2016. Because random samples are used in this analysis. The conclusions from this analysis can be generalized to US population. As no random assignment was conducted in the sampling, we can only draw observational (non-causal) conclusions from this report.

---

# Part 2: Data manipulation

Five new categorical variables are created:

feature_film: yes (movies that are feature films) and no

drame: yes (movies that are dramas) and no

mpaa_rating_R: yes (movies that are R rated) and no

oscar_season: yes (if movie is released in November, October, or December) and no

summer_season: yes (if movie is released in May, June, July, or August) and no

```
#remove all NA data. We assume there are no systematic reasons for NA inputs
movies<-na.omit(movies)
movies<-movies%>%
#add feature_film variable
mutate(feature_film=factor(ifelse(title_type=="Feature Film","yes","no")))%>%
#add drama variable
mutate(drama=factor(ifelse(genre=="Drama", "yes", "no")))%>%
#add mpaa_rating_R variable
mutate(mpaa_rating_R=factor(ifelse(mpaa_rating=="R", "yes", "no")))%>%
#add oscar_season variable
mutate(oscar_season=factor(ifelse(thtr_rel_month>=10,"yes","no")))%>%
#add summer_season variable
mutate(summer_season=factor(ifelse(thtr_rel_month==5|thtr_rel_month==6|thtr_rel_month==7|thtr_rel_month==8,"
yes","no")))
```

---

# Part 3: Exploratory data analysis

Conduct exploratory data analysis of the relationship between audience_score and the new variables constructed in the previous part

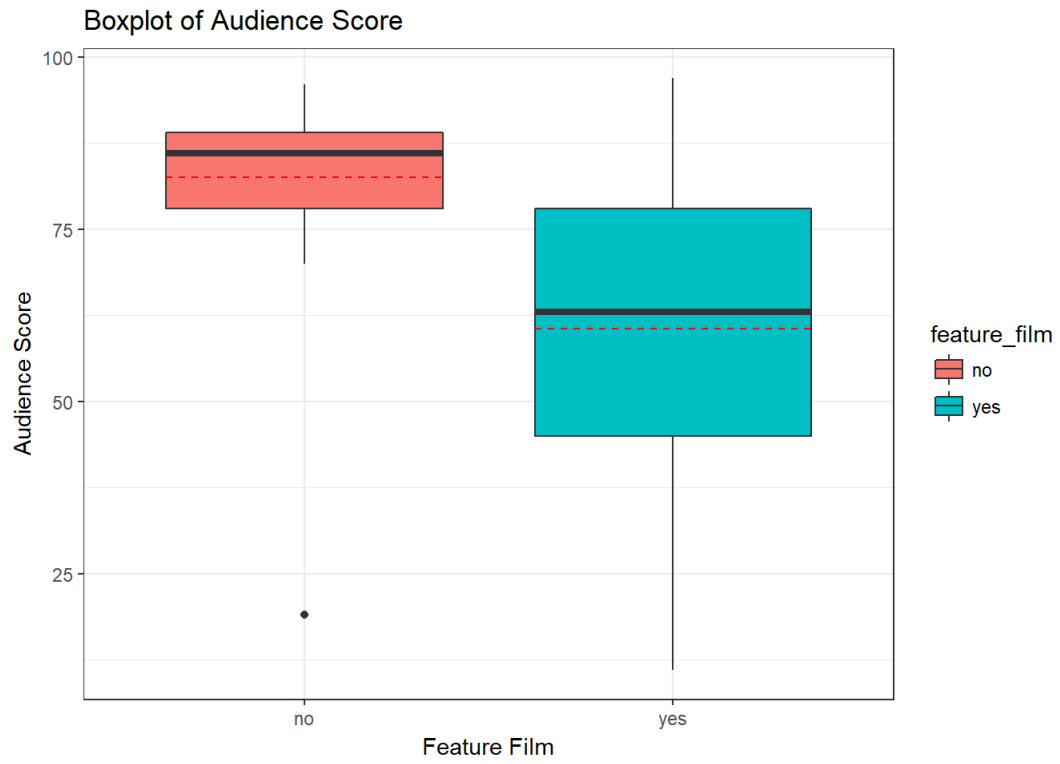
```
# create new data frame movies_new that include responsive variable audience_score and the five newly added explanatory variables feature_film,drama,mpaa_rating_R,oscar_season,summer_season
movies_new<-movies%>%
select(audience_score,feature_film,drama,mpaa_rating_R,oscar_season,summer_season)
names(movies_new)
```

```
## [1] "audience_score" "feature_film"   "drama"          "mpaa_rating_R"
## [5] "oscar_season"   "summer_season"
```

Now, we will analyze audience_score in terms of each newly added categorica variables.

1. andience_score and feature_film The following codes create side-by-side boxplot distribution of audience_score with respect to feature_film. Based on the statistic summary, non featured films have a much higher mean audience score of 82.5 than featured films (60.6), but the distribution of non featured films is highly left-skewed.

```
#side-by-side box plot of  distribution of audience_score and feature_film
ggplot(movies_new, aes(x=feature_film,y=audience_score, fill=feature_film))+theme_bw()+geom_boxplot(fatten=3)+stat_summary(fun.y=mean,geom='errorbar',aes(ymax=..y..,ymin=..y..),width=0.75,linetype='dashed', color='red')+labs(x='Feature Film', y='Audience Score', title='Boxplot of Audience Score')
```



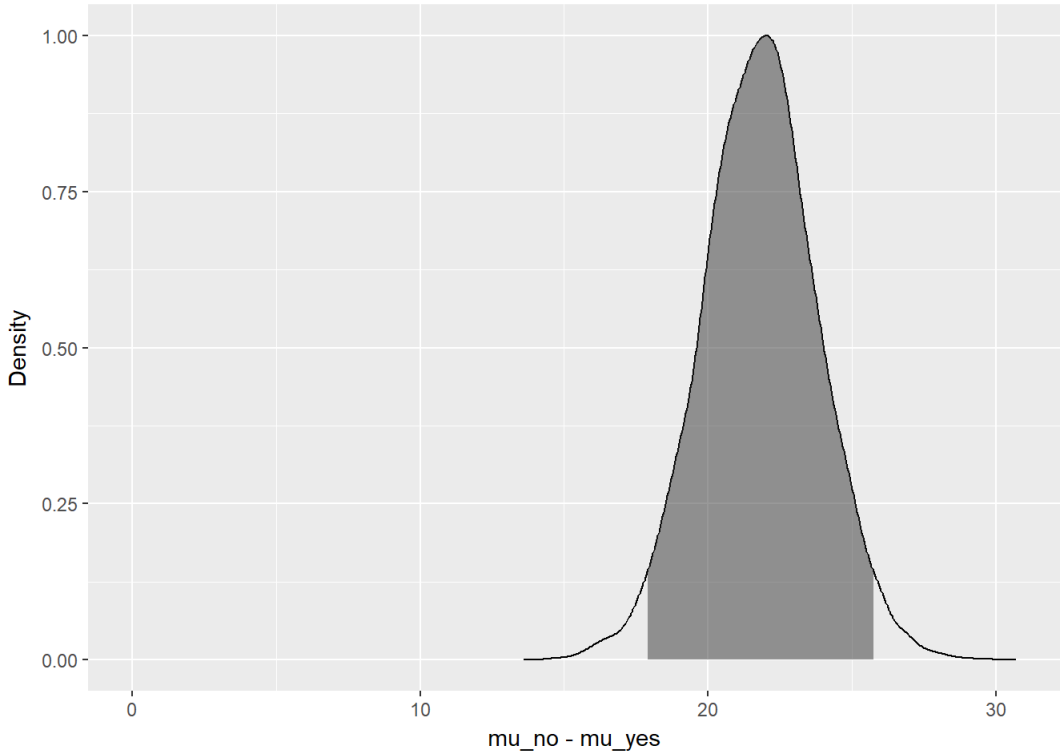

```
#statistical summary of the distribution
movies_new%>%
group_by(feature_film)%>%
summarise(mean=mean(audience_score),median=median(audience_score),sd=sd(audience_score))
```

```
## # A tibble: 2 x 4
##   feature_film  mean median    sd
##   <fct>        <dbl>  <dbl> <dbl>
## 1 no            82.5     86  11.9
## 2 yes           60.6     63  19.8
```

The Bayesian Factor [H2:H1] in the following inference is 1.212332e+13, a very large number, indicating that feature_film has a significant impact on the audience score.

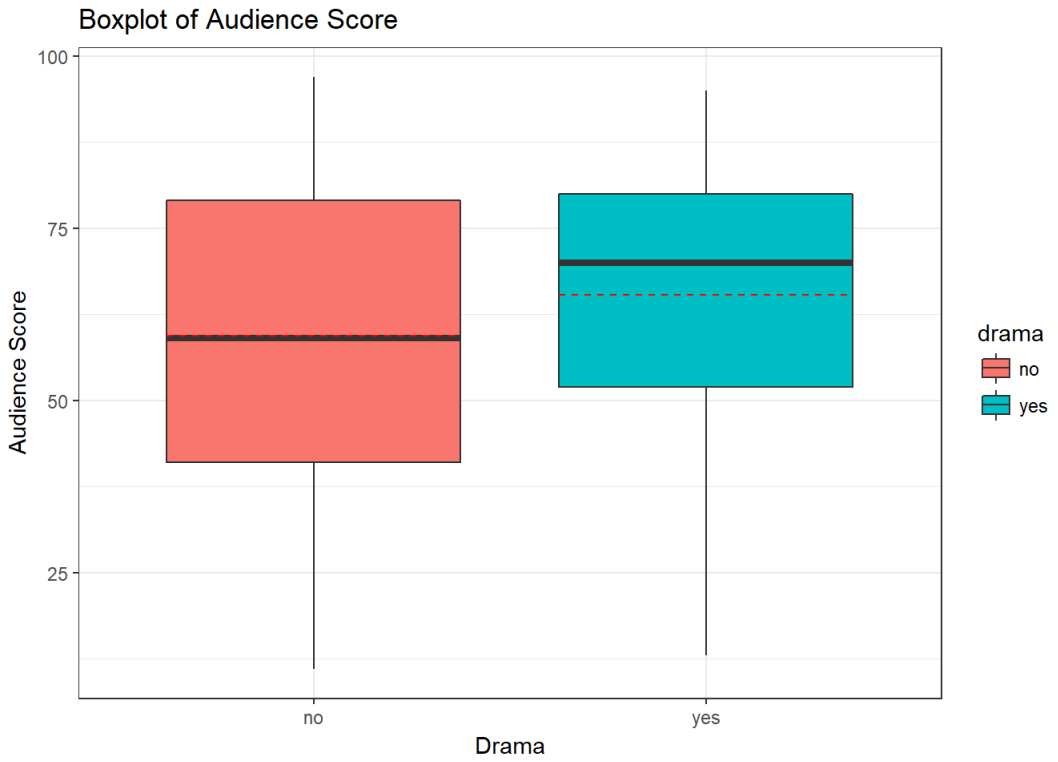
```
bayes_inference(y = audience_score, x = feature_film, data = movies_new, statistic = "mean", type = "ht", null = 0, alternative = "twosided")
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_no = 46, y_bar_no = 82.5435, s_no = 11.9177
## n_yes = 573, y_bar_yes = 60.5777, s_yes = 19.8187
## (Assuming intrinsic prior on parameters)
## Hypotheses:
## H1: mu_no  = mu_yes
## H2: mu_no != mu_yes
##
## Priors:
## P(H1) = 0.5
## P(H2) = 0.5
##
## Results:
## BF[H2:H1] = 1.212332e+13
## P(H1|data) = 0
## P(H2|data) = 1
```




2. andience_score and drama The following codes create side-by-side boxplot distribution of audience_score with respect to drama. Based on the statistic summary, drama films have a much higher mean audience score of 65.3 than non dramas (59.4), but the distribution of drams is highly left-skewed.

```
#side-by-side box plot of  distribution of audience_score and drama
ggplot(movies_new, aes(x=drama,y=audience_score, fill=drama))+theme_bw()+geom_boxplot(fatten=3)+stat_summary
(fun.y=mean,geom='errorbar',aes(ymax=..y..,ymin=..y..),width=0.75,linetype='dashed', color='red')+labs(x='Dr
ama', y='Audience Score', title='Boxplot of Audience Score')
```



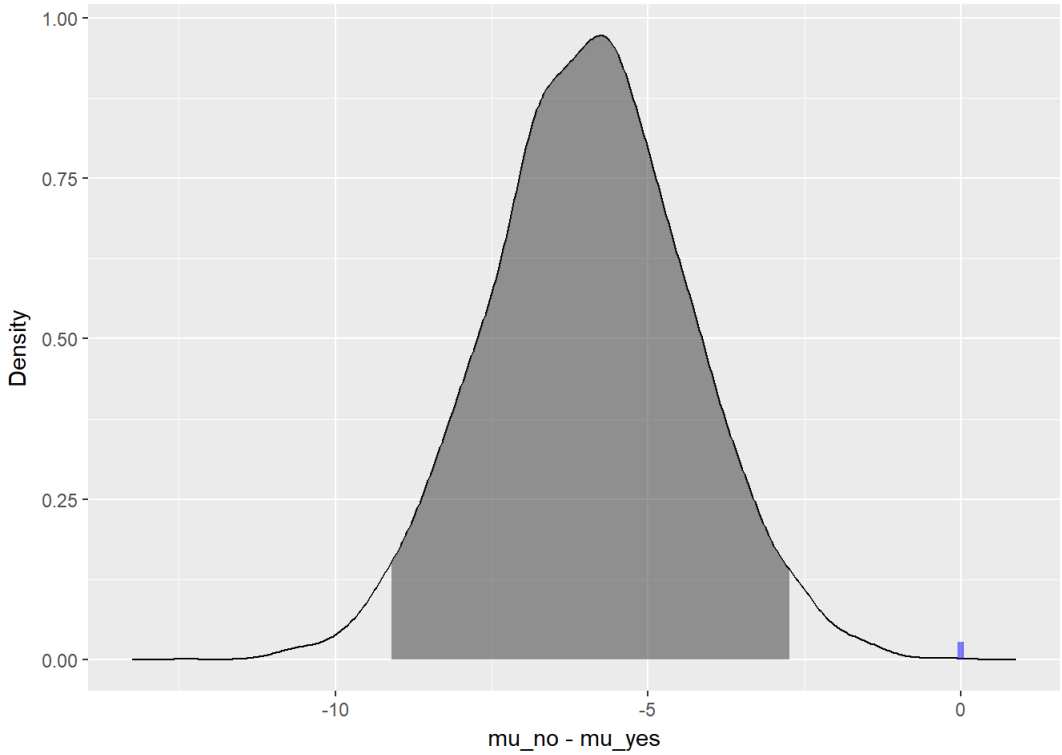

```
#statistical summary of the distribution
movies_new%>%
group_by(drama)%>%
summarise(mean=mean(audience_score),median=median(audience_score),sd=sd(audience_score))
```

```
## # A tibble: 2 x 4
##   drama  mean median    sd
##   <fct> <dbl>  <dbl> <dbl>
## 1 no     59.4     59  21.1
## 2 yes    65.3     70  18.6
```

The Bayesian Factor [H2:H1] in the following inference is 34.6357, a fairly large number, indicating that drama has a strong impact on the audience score.

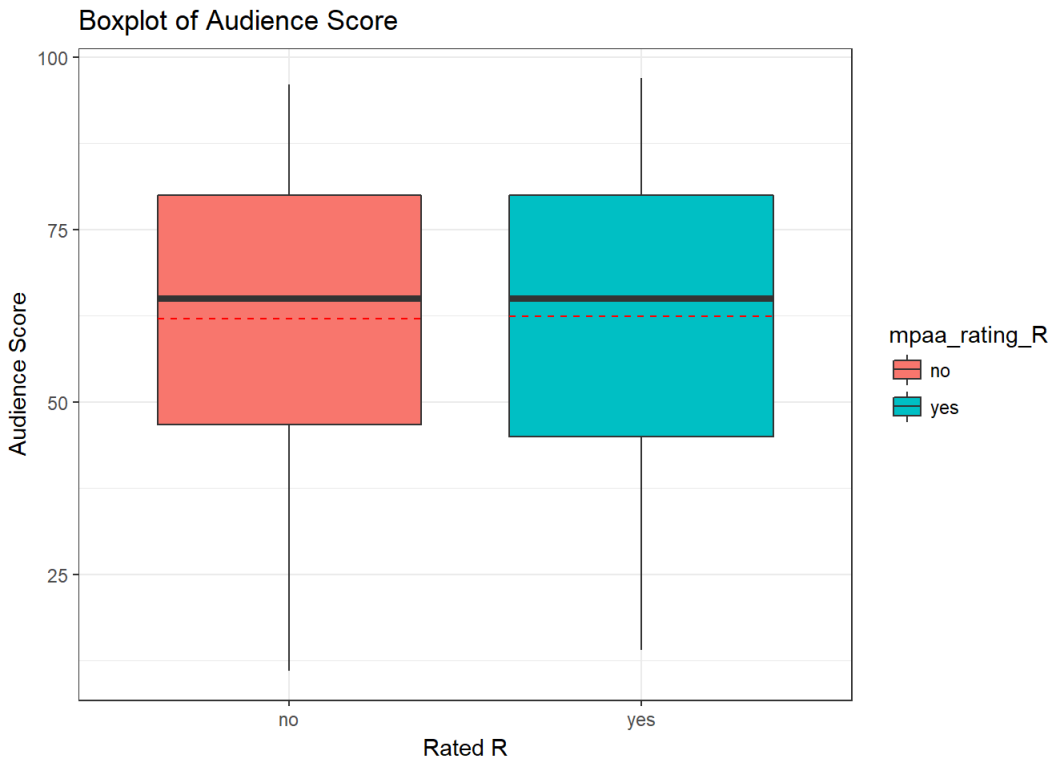
```
bayes_inference(y = audience_score, x = drama, data = movies_new, statistic = "mean", type = "ht", null = 0
, alternative = "twosided")
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_no = 321, y_bar_no = 59.352, s_no = 21.1448
## n_yes = 298, y_bar_yes = 65.2886, s_yes = 18.6305
## (Assuming intrinsic prior on parameters)
## Hypotheses:
## H1: mu_no  = mu_yes
## H2: mu_no != mu_yes
##
## Priors:
## P(H1) = 0.5
## P(H2) = 0.5
##
## Results:
## BF[H2:H1] = 34.6357
## P(H1|data) = 0.0281
## P(H2|data) = 0.9719
```




3. andience_score and mpaa_rating_R The following codes create side-by-side boxplot distribution of audience_score with respect to mpaa_rating_R. Based on the statistic summary, R-rated and non R-rated films have very close audience scores. Both distributions are slightly left-skewed.

```
#side-by-side box plot of  distribution of audience_score and mpaa_rating_R
ggplot(movies_new, aes(x=mpaa_rating_R,y=audience_score, fill=mpaa_rating_R))+theme_bw()+geom_boxplot(fatten
=3)+stat_summary(fun.y=mean,geom='errorbar',aes(ymax=..y..,ymin=..y..),width=0.75,linetype='dashed', color='
red')+labs(x='Rated R', y='Audience Score', title='Boxplot of Audience Score')
```



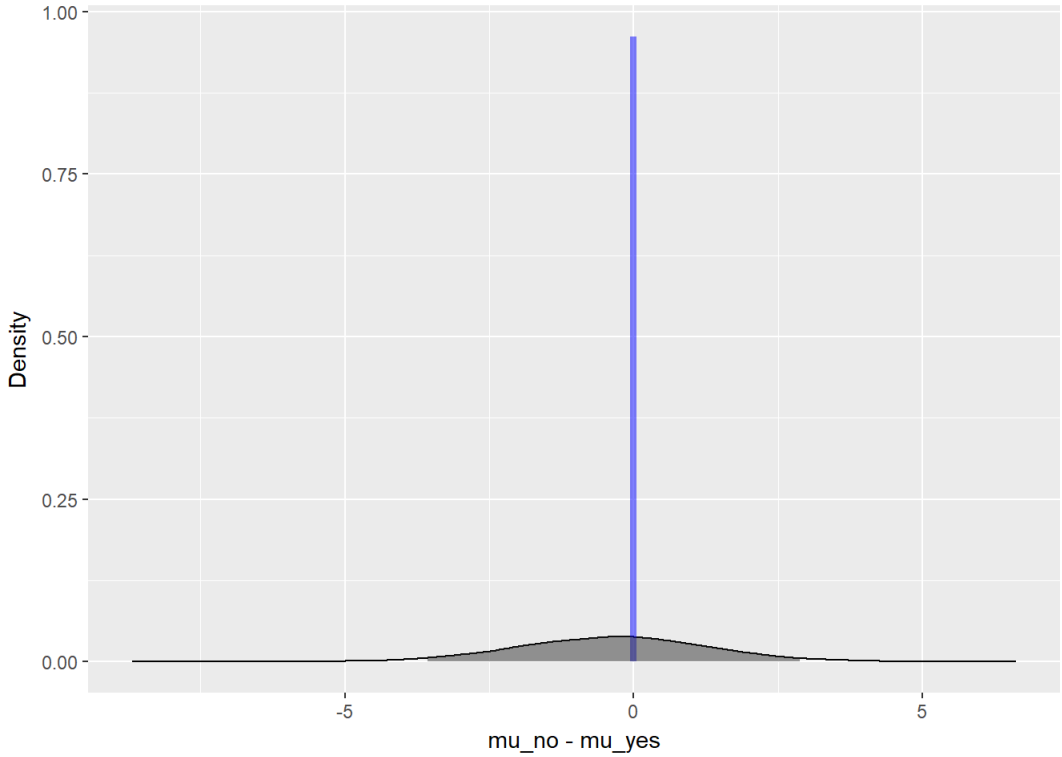

```
#statistical summary of the distribution
movies_new%>%
group_by(mpaa_rating_R)%>%
summarise(mean=mean(audience_score),median=median(audience_score),sd=sd(audience_score))
```

```
## # A tibble: 2 x 4
##   mpaa_rating_R  mean median    sd
##   <fct>         <dbl>  <dbl> <dbl>
## 1 no             62.0     65  20.3
## 2 yes            62.4     65  20.1
```

The Bayesian Factor [H1:H2] in the following inference is 24.83, indicating that mpaa_rating_R has almost no impact on the audience score.

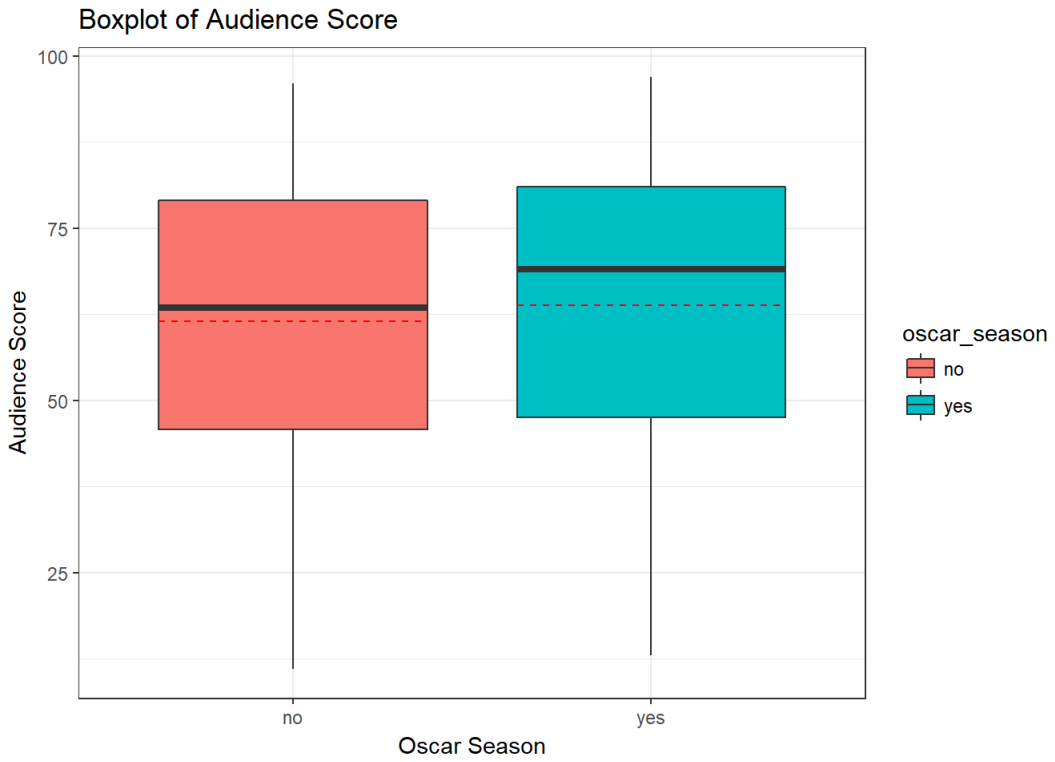
```
bayes_inference(y = audience_score, x = mpaa_rating_R, data = movies_new, statistic = "mean", type = "ht",
null = 0, alternative = "twosided")
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_no = 300, y_bar_no = 62.0367, s_no = 20.3187
## n_yes = 319, y_bar_yes = 62.373, s_yes = 20.0743
## (Assuming intrinsic prior on parameters)
## Hypotheses:
## H1: mu_no  = mu_yes
## H2: mu_no != mu_yes
##
## Priors:
## P(H1) = 0.5
## P(H2) = 0.5
##
## Results:
## BF[H1:H2] = 24.8392
## P(H1|data) = 0.9613
## P(H2|data) = 0.0387
```




4. andience_score and oscar_season The following codes create side-by-side boxplot distribution of audience_score with respect to oscar_season. Based on the statistic summary, movies released during oscar season and non oscar season have close audience scores. Distribution of movies released during oscar season is left-skewed.

```
#side-by-side box plot of  distribution of audience_score and oscar_season
ggplot(movies_new, aes(x=oscar_season,y=audience_score, fill=oscar_season))+theme_bw()+geom_boxplot(fatten=3
)+stat_summary(fun.y=mean,geom='errorbar',aes(ymax=..y..,ymin=..y..),width=0.75,linetype='dashed', color='re
d')+labs(x='Oscar Season', y='Audience Score', title='Boxplot of Audience Score')
```



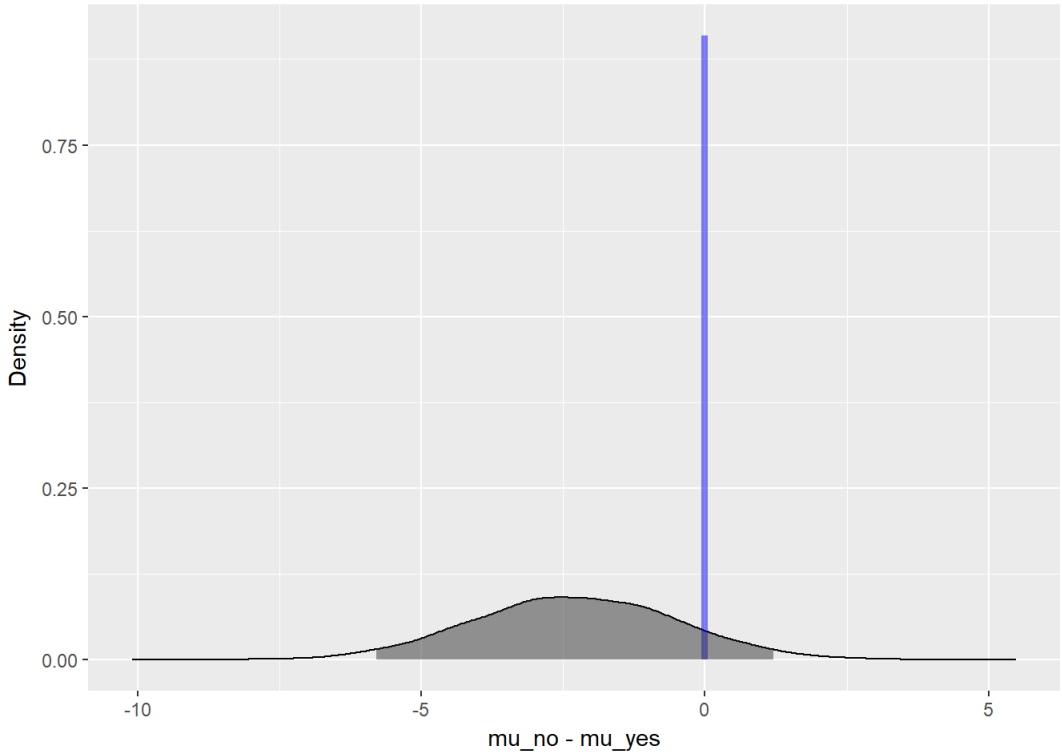

```
#statistical summary of the distribution
movies_new%>%
group_by(oscar_season)%>%
summarise(mean=mean(audience_score),median=median(audience_score),sd=sd(audience_score))
```

```
## # A tibble: 2 x 4
##   oscar_season  mean median    sd
##   <fct>        <dbl>  <dbl> <dbl>
## 1 no            61.5   63.5  20.1
## 2 yes           63.9   69    20.3
```

The Bayesian Factor [H2:H1] in the following inference is 10.019, indicating that oscar_season has no impact on the audience score.

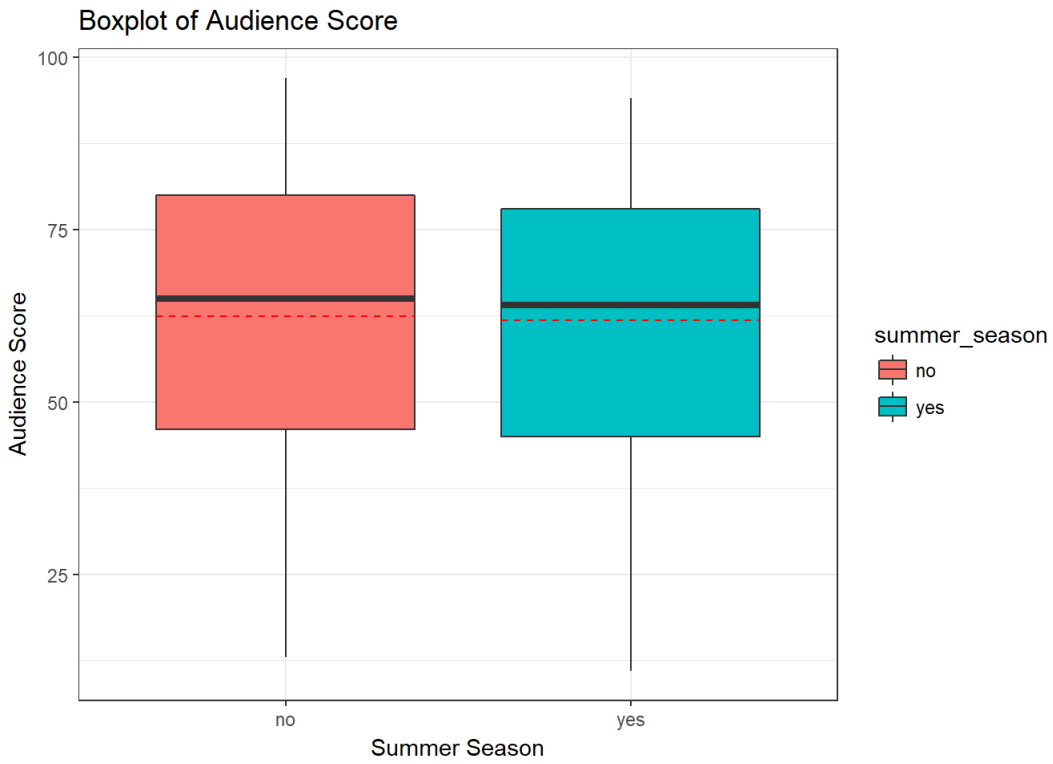
```
bayes_inference(y = audience_score, x = oscar_season, data = movies_new, statistic = "mean", type = "ht", nu
ll = 0, alternative = "twosided")
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_no = 440, y_bar_no = 61.5386, s_no = 20.107
## n_yes = 179, y_bar_yes = 63.8603, s_yes = 20.3118
## (Assuming intrinsic prior on parameters)
## Hypotheses:
## H1: mu_no  = mu_yes
## H2: mu_no != mu_yes
##
## Priors:
## P(H1) = 0.5
## P(H2) = 0.5
##
## Results:
## BF[H1:H2] = 10.019
## P(H1|data) = 0.9092
## P(H2|data) = 0.0908
```




5. andience_score and summer_season The following codes create side-by-side boxplot distribution of audience_score with respect to summer_season. Based on the statistic summary, movies released during summer season and non summer season have close audience scores. Both distributions are fairly normal.

```
#side-by-side box plot of  distribution of audience_score and summer_season
ggplot(movies_new, aes(x=summer_season,y=audience_score, fill=summer_season))+theme_bw()+geom_boxplot(fatten
=3)+stat_summary(fun.y=mean,geom='errorbar',aes(ymax=..y..,ymin=..y..),width=0.75,linetype='dashed', color='
red')+labs(x='Summer Season', y='Audience Score', title='Boxplot of Audience Score')
```



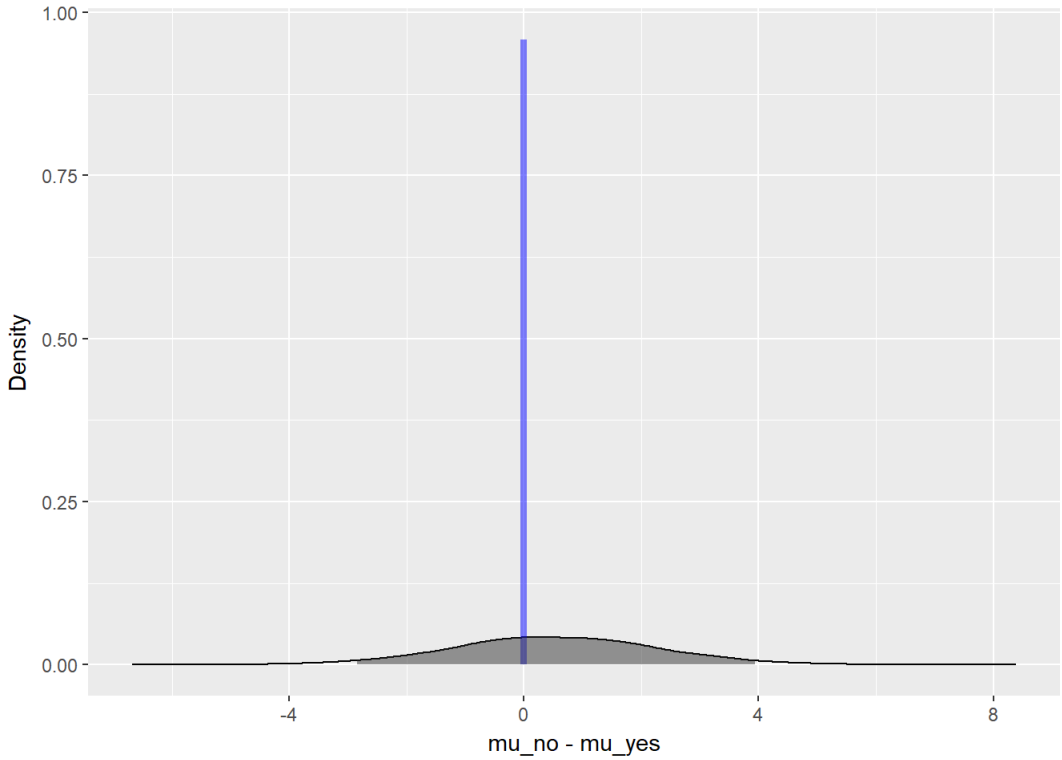

```
#statistical summary of the distribution
movies_new%>%
group_by(summer_season)%>%
summarise(mean=mean(audience_score),median=median(audience_score),sd=sd(audience_score))
```

```
## # A tibble: 2 x 4
##   summer_season  mean median    sd
##   <fct>         <dbl>  <dbl> <dbl>
## 1 no             62.4     65  20.3
## 2 yes            61.9     64  19.9
```

The Bayesian Factor [H1:H2] in the following inference is 22.7623, indicating that summer_season has a positive impact on the audience score.

```
bayes_inference(y = audience_score, x = summer_season, data = movies_new, statistic = "mean", type = "ht",
null = 0, alternative = "twosided")
```

```
## Response variable: numerical, Explanatory variable: categorical (2 levels)
## n_no = 418, y_bar_no = 62.3828, s_no = 20.3266
## n_yes = 201, y_bar_yes = 61.8507, s_yes = 19.9092
## (Assuming intrinsic prior on parameters)
## Hypotheses:
## H1: mu_no  = mu_yes
## H2: mu_no != mu_yes
##
## Priors:
## P(H1) = 0.5
## P(H2) = 0.5
##
## Results:
## BF[H1:H2] = 22.7623
## P(H1|data) = 0.9579
## P(H2|data) = 0.0421
```




# Part 4: Modeling

We will use Bayesian Information Criterion (BIC) to compare four modeling selection estimators, namely, Bayesian Model Averaging (BMA), Highest Probability Model (HPM), Median Probability Model (MPM), and Best Predictive Model (BPM) to choose the optimal model with the lowest Average Percent Error (APE).

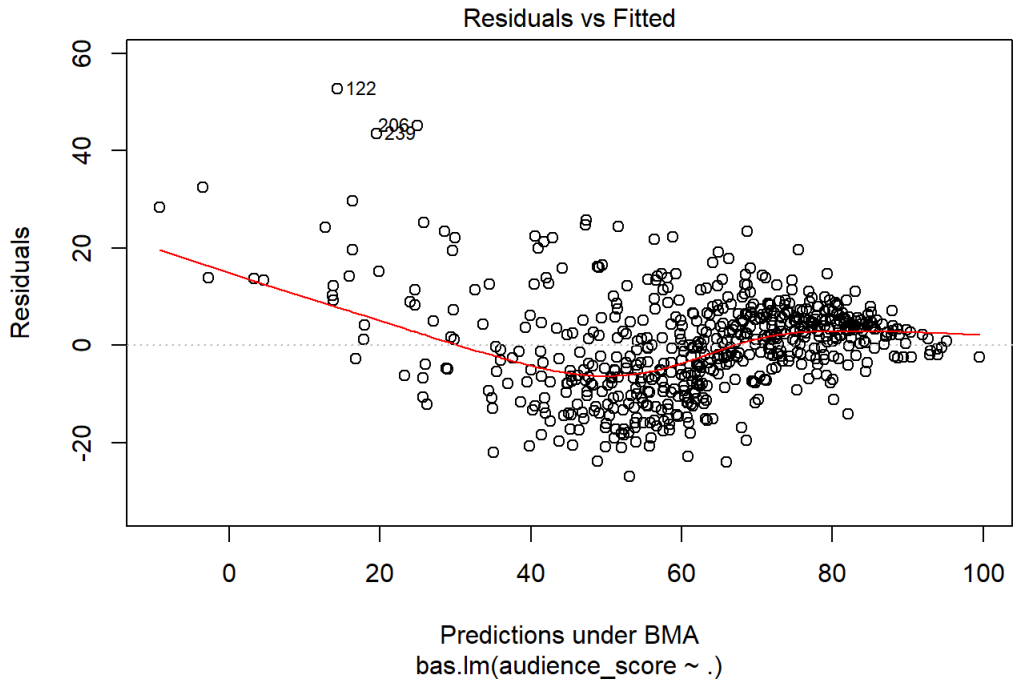
First, we need to construct the full model,which includes audience_score as the response variable and the following 16 explanatory variables.

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

```
#create the full model
movies_full<-movies%>%

select(audience_score,feature_film,drama,runtime,mpaa_rating_R,thtr_rel_year,oscar_season,summer_season,imdb
_rating,
       imdb_num_votes,critics_score,best_pic_nom,best_pic_win,best_actor_win,best_actress_win,best_dir_win,t
op200_box)
```

Before we narrow down our best model, we need to check the residual plot of observed and fitted values under BMA using the original model fit. We also need to find the variables that contribute to the best model.

```
#original model fit
bma_movies = bas.lm(audience_score ~., data = movies_full,
                     prior = "BIC",
                    method = "MCMC",
                    modelprior = uniform())

#residual plot of observed and fitted values
plot(bma_movies,which=1, ask=FALSE)
```




```
#mapping of the top models
image(bma_movies, rotate=F)
```


```
round(summary(bma_movies),3)
```

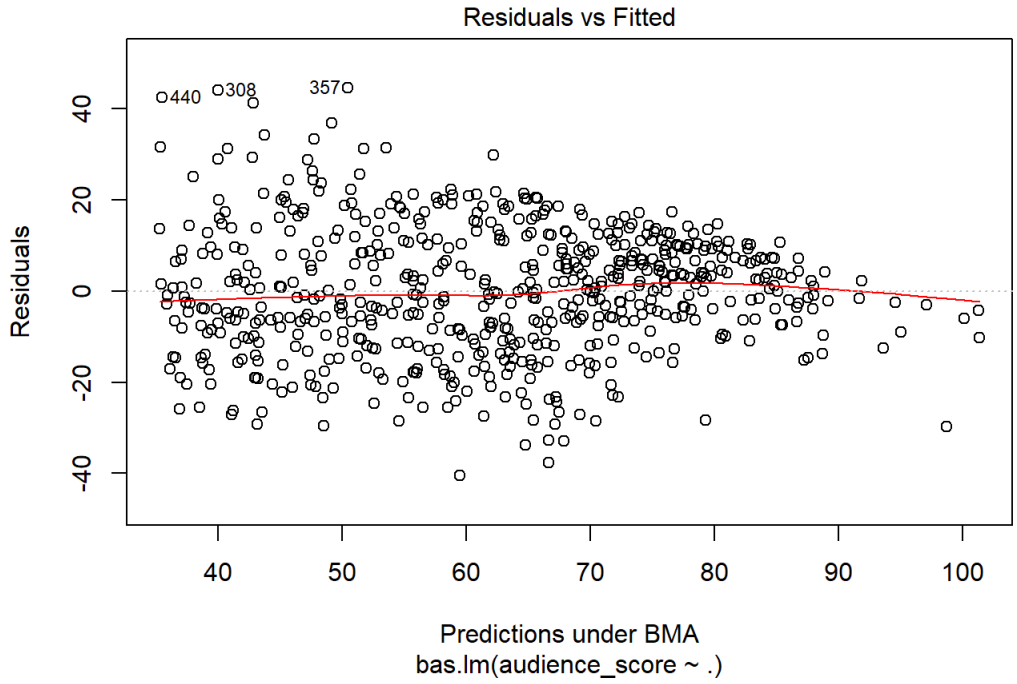
```
##                     P(B != 0 | Y)   model 1   model 2   model 3   model 4
## Intercept                   1.000     1.000     1.000     1.000     1.000
## feature_filmyes             0.059     0.000     0.000     0.000     0.000
## dramayes                    0.045     0.000     0.000     0.000     0.000
## runtime                     0.515     1.000     0.000     0.000     1.000
## mpaa_rating_Ryes            0.163     0.000     0.000     0.000     1.000
## thtr_rel_year               0.082     0.000     0.000     0.000     0.000
## oscar_seasonyes             0.064     0.000     0.000     0.000     0.000
## summer_seasonyes            0.079     0.000     0.000     0.000     0.000
## imdb_rating                 1.000     1.000     1.000     1.000     1.000
## imdb_num_votes              0.061     0.000     0.000     0.000     0.000
## critics_score               0.920     1.000     1.000     1.000     1.000
## best_pic_nomyes             0.129     0.000     0.000     0.000     0.000
## best_pic_winyes             0.041     0.000     0.000     0.000     0.000
## best_actor_winyes           0.114     0.000     0.000     0.000     0.000
## best_actress_winyes         0.146     0.000     0.000     1.000     0.000
## best_dir_winyes             0.067     0.000     0.000     0.000     0.000
## top200_boxyes               0.048     0.000     0.000     0.000     0.000
## BF                             NA     1.000     0.872     0.205     0.204
## PostProbs                      NA     0.156     0.137     0.033     0.032
## R2                             NA     0.748     0.746     0.747     0.750
## dim                            NA     4.000     3.000     4.000     5.000
## logmarg                        NA -3434.752 -3434.889 -3436.338 -3436.342
##                       model 5
## Intercept               1.000
## feature_filmyes         0.000
## dramayes                0.000
## runtime                 1.000
## mpaa_rating_Ryes        0.000
## thtr_rel_year           0.000
## oscar_seasonyes         0.000
## summer_seasonyes        0.000
## imdb_rating             1.000
## imdb_num_votes          0.000
## critics_score           1.000
## best_pic_nomyes         1.000
## best_pic_winyes         0.000
## best_actor_winyes       0.000
## best_actress_winyes     0.000
## best_dir_winyes         0.000
## top200_boxyes           0.000
## BF                      0.185
## PostProbs               0.029
## R2                      0.750
## dim                     5.000
## logmarg             -3436.438
```

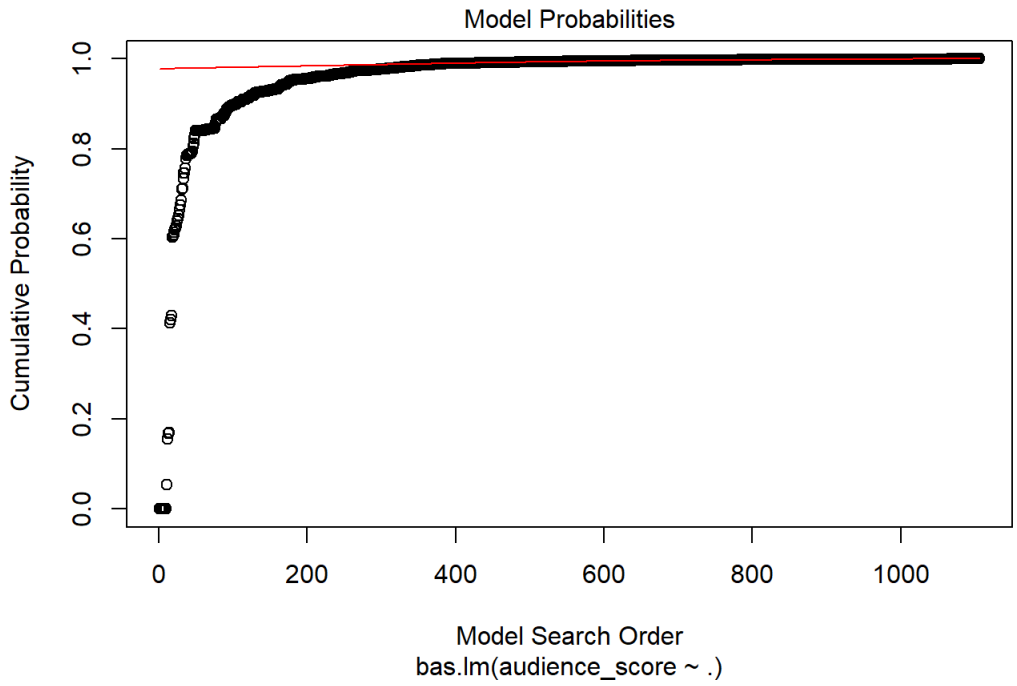
The “Residuals vs Fitted” plot doesn’t show a constant variance and residuals are not centered along the zero line, indicating residuals are not normally distributed. The best model consists of Intercept, Runtime, imdb_rating and critics_score. It is interesting that the model only includes the three numerical variables, while all categorical variables are excluded. In part 3, we have analyzed that feature_film should have a significant impact on audience_score but it is not included in our best model. The numerical variables might be highly collinear to our response variable audience_score. I removed the numerical variable imdb_rating from the full model and conduct Bayesian approach with the updated model, which yield a much better result.

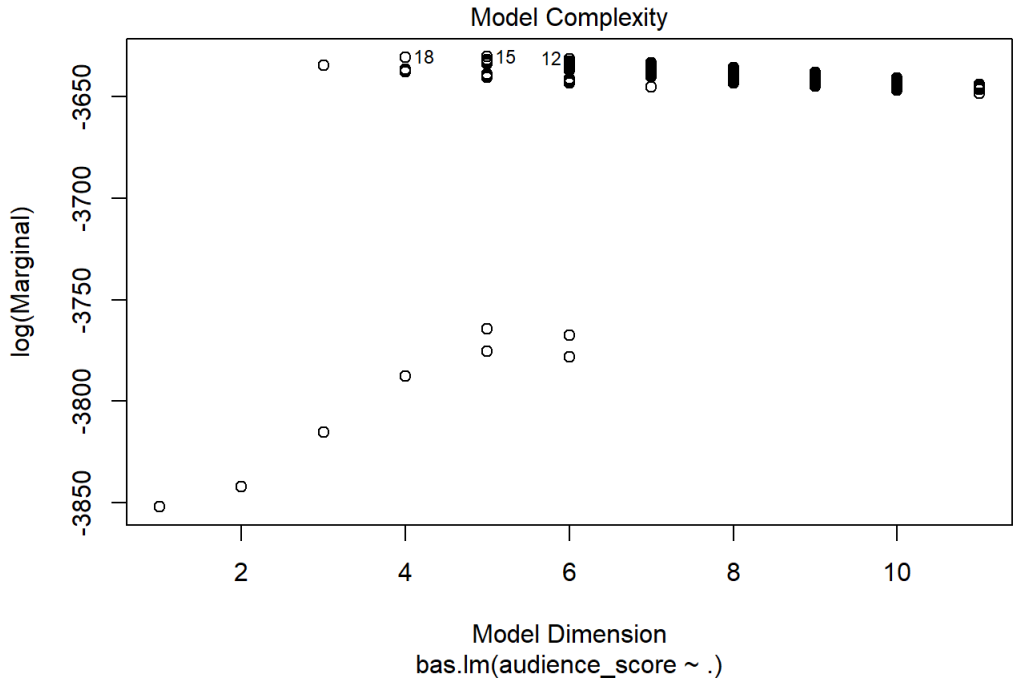
```
#update model, remove imdb_rating
movies_update<-movies%>%

select(audience_score,feature_film,drama,runtime,mpaa_rating_R,thtr_rel_year,oscar_season,summer_season,
       imdb_num_votes,critics_score,best_pic_nom,best_pic_win,best_actor_win,best_actress_win,best_dir_win,t
op200_box)
```

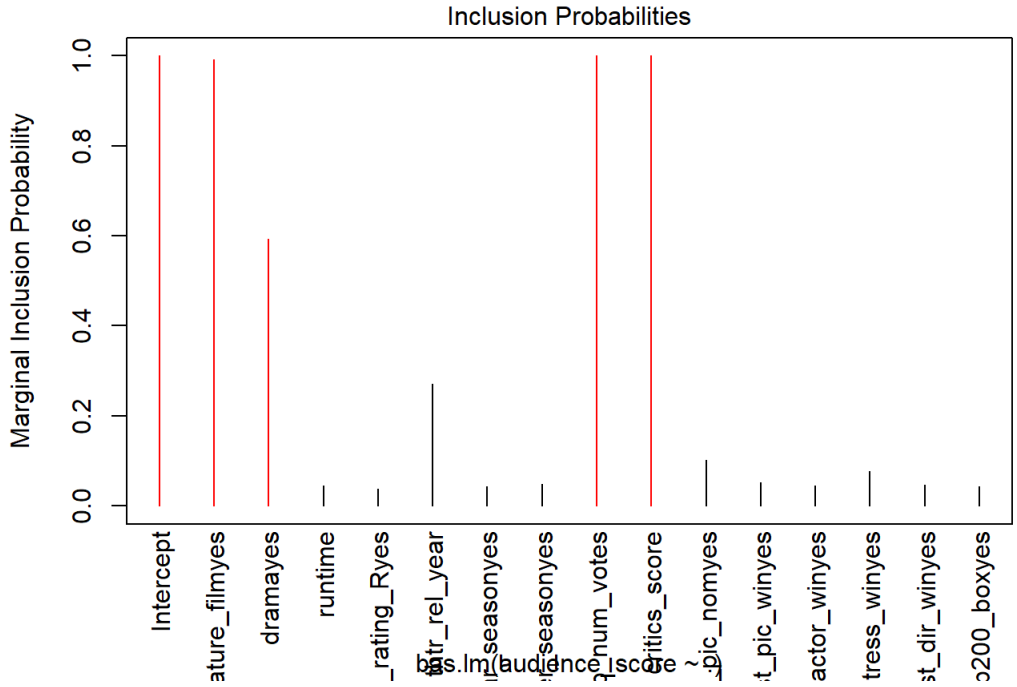
The “Residuals vs. Fitted” plot below shows much better distribution by removing imdb_rating. The best model consists of Incercept, feature_film, drama, imdb_num_votes, and critics_score. It is consistent with our analysis in part 3.

```
#original model fit
bma_movies_update = bas.lm(audience_score ~., data = movies_update, prior = "BIC", method = "MCMC", modelpri
or = uniform())

#residual plot of observed and fitted values
plot(bma_movies_update)
```

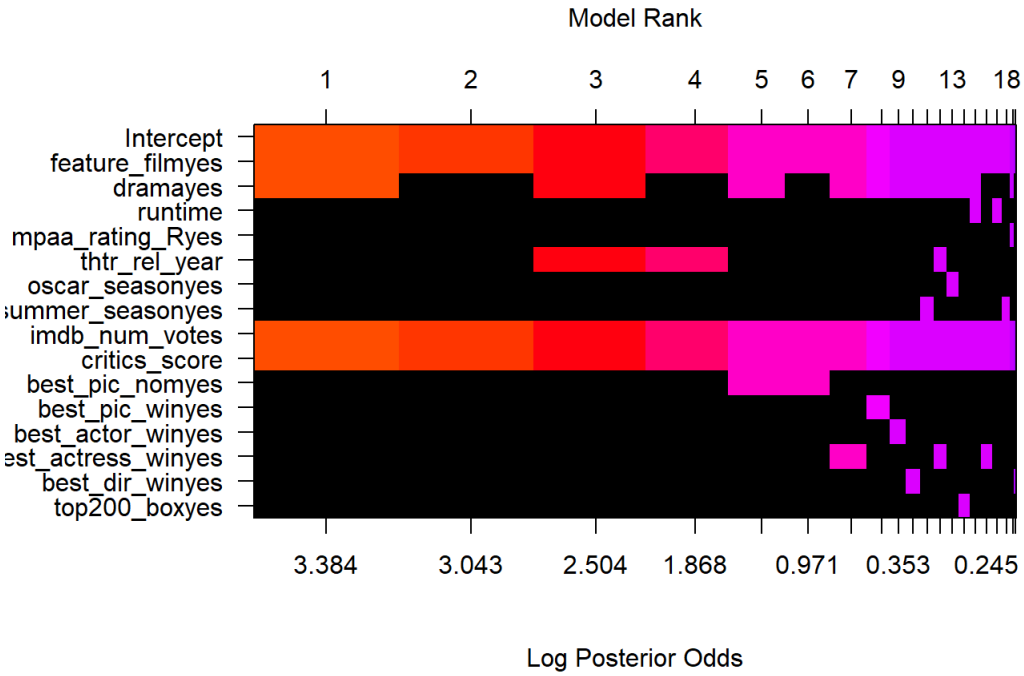

Residuals vs Fitted
440
308
357
Residuals
40
20
0
-20
-40
40
50
60
70
80
90
100
Predictions under BMA
bas.lm(audience_score ~ .)


Model Probabilities
Cumulative Probability
0.0
0.2
0.4
0.6
0.8
1.0
0
200
400
600
800
1000
Model Search Order
bas.lm(audience_score ~ .)




```
#mapping of the top models
image(bma_movies_update, rotate=F)
```



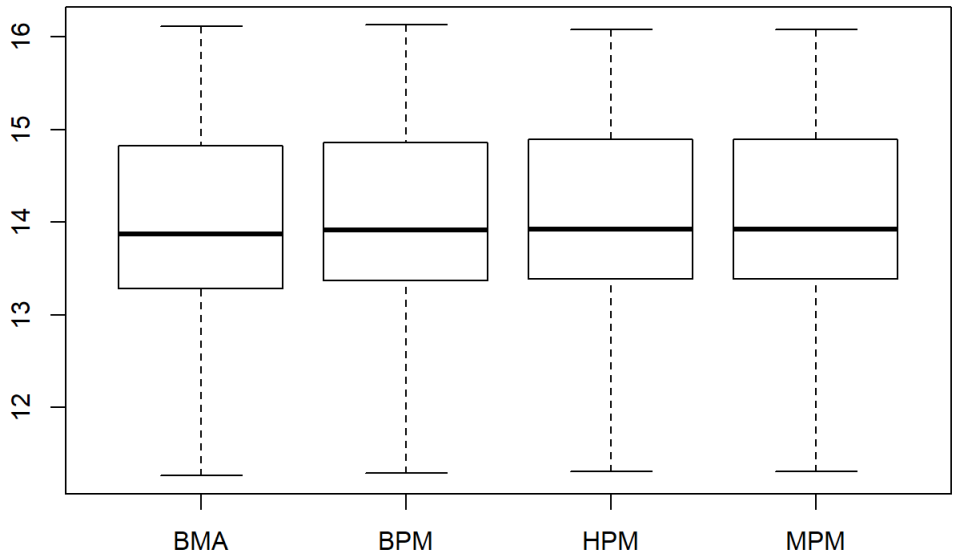

```
round(summary(bma_movies_update),3)
```

```
##                      P(B != 0 | Y)   model 1   model 2   model 3   model 4
## Intercept                    1.000     1.000     1.000     1.000     1.000
## feature_filmyes              0.991     1.000     1.000     1.000     1.000
## dramayes                     0.593     1.000     0.000     1.000     0.000
## runtime                      0.046     0.000     0.000     0.000     0.000
## mpaa_rating_Ryes             0.038     0.000     0.000     0.000     0.000
## thtr_rel_year                0.271     0.000     0.000     1.000     1.000
## oscar_seasonyes              0.043     0.000     0.000     0.000     0.000
## summer_seasonyes             0.049     0.000     0.000     0.000     0.000
## imdb_num_votes               1.000     1.000     1.000     1.000     1.000
## critics_score                1.000     1.000     1.000     1.000     1.000
## best_pic_nomyes              0.101     0.000     0.000     0.000     0.000
## best_pic_winyes              0.052     0.000     0.000     0.000     0.000
## best_actor_winyes            0.045     0.000     0.000     0.000     0.000
## best_actress_winyes          0.076     0.000     0.000     0.000     0.000
## best_dir_winyes              0.046     0.000     0.000     0.000     0.000
## top200_boxyes                0.043     0.000     0.000     0.000     0.000
## BF                              NA     1.000     0.718     0.421     0.227
## PostProbs                       NA     0.244     0.173     0.101     0.054
## R2                              NA     0.531     0.526     0.535     0.529
## dim                             NA     5.000     4.000     6.000     5.000
## logmarg                         NA -3630.353 -3630.685 -3631.218 -3631.837
##                        model 5
## Intercept                1.000
## feature_filmyes          1.000
## dramayes                 1.000
## runtime                  0.000
## mpaa_rating_Ryes         0.000
## thtr_rel_year            0.000
## oscar_seasonyes          0.000
## summer_seasonyes         0.000
## imdb_num_votes           1.000
## critics_score            1.000
## best_pic_nomyes          1.000
## best_pic_winyes          0.000
## best_actor_winyes        0.000
## best_actress_winyes      0.000
## best_dir_winyes          0.000
## top200_boxyes            0.000
## BF                       0.103
## PostProbs                0.024
## R2                       0.533
## dim                      6.000
## logmarg              -3632.624
```

Now, we can BIC to compare the four modeling selection estimators to find our final model.

```
# use BIC to compare four modeling selection estimators, BMA, HPM, MPM and BPM.
set.seed(898)
n = nrow(movies_update)
n_cv = 50
ape = matrix(NA, ncol=4, nrow=n_cv)
colnames(ape) = c("BMA", "BPM", "HPM", "MPM")

for (i in 1:n_cv) {
  train = sample(1:n, size=round(.90*n), replace=FALSE)
  movies_train = movies_update[train,]
  movies_test =movies_update[-train,]

  bma_movies_score = bas.lm(audience_score~., data=movies_train,
                            prior="BIC", modelprior=uniform(), initprobs="eplogp")
  yhat_bma = predict(bma_movies_score, movies_test, estimator="BMA")$fit
  yhat_hpm = predict(bma_movies_score, movies_test, estimator="HPM")$fit
  yhat_mpm = predict(bma_movies_score, movies_test, estimator="MPM")$fit
  yhat_bpm = predict(bma_movies_score, movies_test, estimator="BPM")$fit
  ape[i, "BMA"] = cv.summary.bas(yhat_bma, movies_test$audience_score)
  ape[i, "BPM"] = cv.summary.bas(yhat_bpm, movies_test$audience_score)
  ape[i, "HPM"] = cv.summary.bas(yhat_hpm, movies_test$audience_score)
  ape[i, "MPM"] = cv.summary.bas(yhat_mpm, movies_test$audience_score)
}
```

```
boxplot(ape)
```




```
apply(ape, 2, mean)
```

```
##      BMA      BPM      HPM      MPM
## 13.87186 13.92950 13.93584 13.93659
```

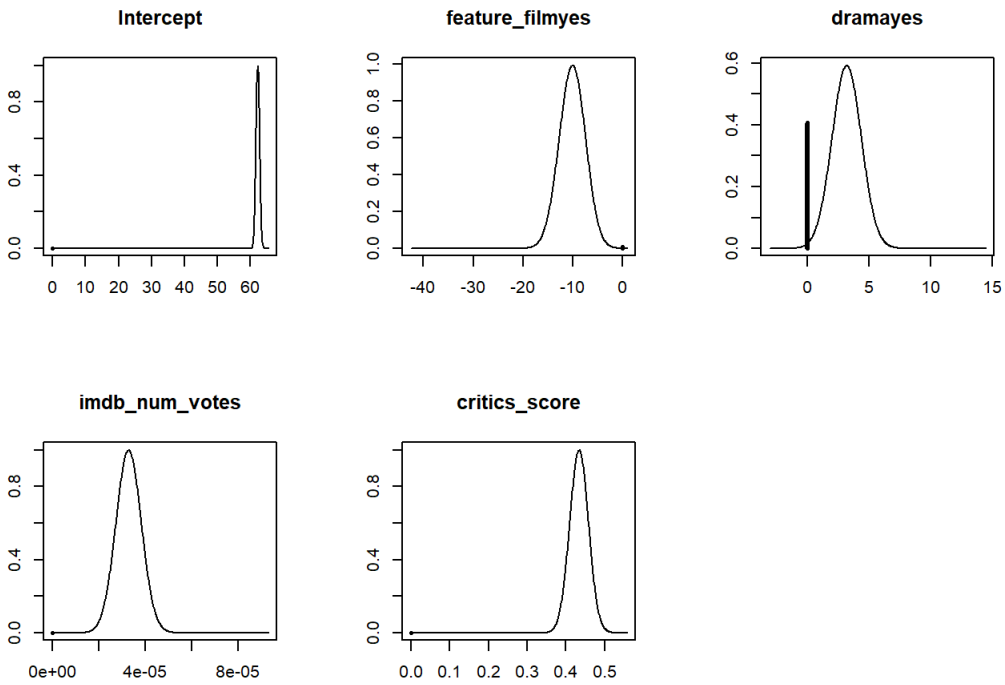
We find the sensitivity of the four choice of estimators by viewing the side-by-side boxplots of the average prediction errors as well as the mean of APE over the different test sets. It appears that BMA has the lowest APE of 13.87. We will use BMA as the estimator in our prediction.

From the map of model ranks, we find that the best model consists of Intercept, Runtime, imdb_rating and critics_score. The coefficient plots (below) also show strong evidence against the null values.

```
#coefficient of the best model, Intercept, Runtime, imdb_rating and critics_score
confint(coefficients(bma_movies_update))
```

```
##                              2.5%         97.5%          beta
## Intercept            6.116582e+01  6.333962e+01  6.221002e+01
## feature_filmyes     -1.546385e+01 -4.750541e+00 -9.959527e+00
## dramayes             0.000000e+00  4.826766e+00  1.885029e+00
## runtime              0.000000e+00  0.000000e+00  8.033733e-04
## mpaa_rating_Ryes     0.000000e+00  0.000000e+00  2.043719e-03
## thtr_rel_year       -1.663428e-01  0.000000e+00 -3.089909e-02
## oscar_seasonyes      0.000000e+00  0.000000e+00  2.306401e-02
## summer_seasonyes     0.000000e+00  0.000000e+00 -4.549137e-02
## imdb_num_votes       2.225532e-05  4.352482e-05  3.286377e-05
## critics_score        3.845861e-01  4.801535e-01  4.343835e-01
## best_pic_nomyes      0.000000e+00  5.615267e+00  4.858125e-01
## best_pic_winyes     -9.553278e-01  0.000000e+00 -2.327703e-01
## best_actor_winyes    0.000000e+00  0.000000e+00 -4.553033e-02
## best_actress_winyes -1.907650e+00  4.942012e-03 -1.625199e-01
## best_dir_winyes     -6.087836e-03  3.219826e-03 -6.495188e-02
## top200_boxyes        0.000000e+00  0.000000e+00 -8.350162e-02
## attr(,"Probability")
## [1] 0.95
## attr(,"class")
## [1] "confint.bas"
```

```
par(mfrow=c(2,3))
plot(coefficients(bma_movies_update), subset=c(1,2,3,9,10), ask=FALSE)
```




# Part 5: Prediction

We choose the 2016 animated movie Zootopia. Information about the movie is found at: https://www.imdb.com/title/tt2948356/ https://www.rottentomatoes.com/m/zootopia/

```
#dataframe of Zootopia
zootopia <- data.frame(feature_film="yes",drama="no",runtime=108,mpaa_rating_R="no",
                       thtr_rel_year=2016,oscar_season="no",summer_season="no",

                       imdb_num_votes=348759,critics_score=98,best_pic_nom="no",
                       best_pic_win="no",best_actor_win="no",best_actress_win="no",
                       best_dir_win="no",top200_box="no",audience_score=92)
```

Predict the audience score of Zootopia using our selected model and BMA estimator method.

```
pred_zootopia<- predict(bma_movies_update, newdata=zootopia, estimator="BMA", se.fit=TRUE, interval="predict
")


pred_zootopia$Ybma
```

```
##          [,1]
## [1,] 87.14633
```

The predicted score is 87 which is close to the real score of 92 on Rotten Tomatoes.

# Part 6: Conclusion and Discussion

In this report, we used BIC prior and BMA estimator to find the model which best predicts the audience score of a movie. The final model consists of variables “feature_film”, “drama”, “imdb_num_votes”, and “critics_score”. Our predicted audience score of Zootopia is 87, which is very close to the real score of 92.

Possible limitations of this model: 1) There are three outliers, 440, 308 and 357 in the residual plot. More analysis need to be performed to determine whether to keep or remove these these outliers in our model.

2. As mentioned in Part 4, we have suspected collinearity between numerical variables such as imdb_rating, critics_score, and our response variable audience_score, we removed imdb_rating based on more normal distribution of the residuals. However, we need to perform analysis such as checking the variance inflation factor (VIF) to remove highly correlated predictors from the model. This analysis is beyond the scope of this class.