Code

Data Analysis Project

Bayesian modeling and prediction using movies

References

1. https://rstudio-pubs-static.s3.amazonaws.com/342314_b1db7ca80c0c4d4eabde95310c0452b2.html (https://rstudio-pubs-static.s3.amazonaws.com/342314_b1db7ca80c0c4d4eabde95310c0452b2.html)

Setup

Load packages

Hide

```
library(ggplot2)
library(dplyr)
#install.packages('statsr')
library(statsr)
#package_version('statsr')
library(BAS)
library(caret)
library(grid)
library(gridExtra)
detach("package:gridExtra",character.only = TRUE, unload=TRUE)
library(lattice)
```

Hide

```
set.seed(123)
```

Load data

The data set is comprised of 651 randomly sampled movies produced and released before 2016. Some of the varibles provides extra information for analysis but are not useful for prediction, we will exclude them before building the model.

Hide

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

Part 1: Data

check data type

Hide

```
str(movies)
```

```
tibble [651 × 32] (S3: tbl_df/tbl/data.frame)
 $ title           : chr [1:651] "Filly Brown" "The Dish" "Waiting for Guffman" "The Age
of Innocence" ...
 $ title_type      : Factor w/ 3 levels "Documentary",..: 2 2 2 2 2 1 2 2 1 2 ...
 $ genre           : Factor w/ 11 levels "Action & Adventure",..: 6 6 4 6 7 5 6 6 5 6
...
 $ runtime         : num [1:651] 80 101 84 139 90 78 142 93 88 119 ...
 $ mpaa_rating     : Factor w/ 6 levels "G","NC-17","PG",..: 5 4 5 3 5 6 4 5 6 6 ...
 $ studio          : Factor w/ 211 levels "20th Century Fox",..: 91 202 167 34 13 163 14
7 118 88 84 ...
 $ thtr_rel_year   : num [1:651] 2013 2001 1996 1993 2004 ...
 $ thtr_rel_month  : num [1:651] 4 3 8 10 9 1 1 11 9 3 ...
 $ thtr_rel_day    : num [1:651] 19 14 21 1 10 15 1 8 7 2 ...
 $ dvd_rel_year    : num [1:651] 2013 2001 2001 2001 2005 ...
 $ dvd_rel_month   : num [1:651] 7 8 8 11 4 4 2 3 1 8 ...
 $ dvd_rel_day     : num [1:651] 30 28 21 6 19 20 18 2 21 14 ...
 $ imdb_rating     : num [1:651] 5.5 7.3 7.6 7.2 5.1 7.8 7.2 5.5 7.5 6.6 ...
 $ imdb_num_votes  : int [1:651] 899 12285 22381 35096 2386 333 5016 2272 880 12496 ...
 $ critics_rating  : Factor w/ 3 levels "Certified Fresh",..: 3 1 1 1 3 2 3 3 2 1 ...
 $ critics_score   : num [1:651] 45 96 91 80 33 91 57 17 90 83 ...
 $ audience_rating : Factor w/ 2 levels "Spilled","Upright": 2 2 2 2 1 2 2 1 2 2 ...
 $ audience_score  : num [1:651] 73 81 91 76 27 86 76 47 89 66 ...
 $ best_pic_nom    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ best_pic_win    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ best_actor_win  : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 2 1 1 ...
 $ best_actress_win: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ best_dir_win    : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 1 1 1 ...
 $ top200_box      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ director        : chr [1:651] "Michael D. Olmos" "Rob Sitch" "Christopher Guest" "Mar
tin Scorsese" ...
 $ actor1          : chr [1:651] "Gina Rodriguez" "Sam Neill" "Christopher Guest" "Danie
l Day-Lewis" ...
 $ actor2          : chr [1:651] "Jenni Rivera" "Kevin Harrington" "Catherine O'Hara" "M
ichelle Pfeiffer" ...
 $ actor3          : chr [1:651] "Lou Diamond Phillips" "Patrick Warburton" "Parker Pose
y" "Winona Ryder" ...
 $ actor4          : chr [1:651] "Emilio Rivera" "Tom Long" "Eugene Levy" "Richard E. Gr
ant" ...
 $ actor5          : chr [1:651] "Joseph Julian Soria" "Genevieve Mooy" "Bob Balaban" "A
lec McCowen" ...
 $ imdb_url        : chr [1:651] "http://www.imdb.com/title/tt1869425/" "http://www.imd
b.com/title/tt0205873/" "http://www.imdb.com/title/tt0118111/" "http://www.imdb.com/titl
e/tt0106226/" ...
 $ rt_url          : chr [1:651] "//www.rottentomatoes.com/m/filly_brown_2012/" "//www.r
ottentomatoes.com/m/dish/" "//www.rottentomatoes.com/m/waiting_for_guffman/" "//www.rott
entomatoes.com/m/age_of_innocence/" ...
```

check summary statistics

Hide

```
summary(movies)
```

```
   title                  title_type
Length:651         Documentary : 55
Class :character   Feature Film:591
Mode  :character   TV Movie    :  5





               genre        runtime       mpaa_rating
Drama            :305   Min.   : 39.0   G      : 19
Comedy           : 87   1st Qu.: 92.0   NC-17  :  2
Action & Adventure: 65   Median :103.0   PG     :118
Mystery & Suspense: 59   Mean   :105.8   PG-13  :133
Documentary      : 52   3rd Qu.:115.8   R      :329
Horror           : 23   Max.   :267.0   Unrated: 50
(Other)          : 60   NA's   :1
                             studio    thtr_rel_year
Paramount Pictures             : 37   Min.   :1970
Warner Bros. Pictures          : 30   1st Qu.:1990
Sony Pictures Home Entertainment: 27   Median :2000
Universal Pictures             : 23   Mean   :1998
Warner Home Video              : 19   3rd Qu.:2007
(Other)                        :507   Max.   :2014
NA's                           :  8
thtr_rel_month   thtr_rel_day    dvd_rel_year
Min.   : 1.00   Min.   : 1.00   Min.   :1991
1st Qu.: 4.00   1st Qu.: 7.00   1st Qu.:2001
Median : 7.00   Median :15.00   Median :2004
Mean   : 6.74   Mean   :14.42   Mean   :2004
3rd Qu.:10.00   3rd Qu.:21.00   3rd Qu.:2008
Max.   :12.00   Max.   :31.00   Max.   :2015
                                NA's   :8
dvd_rel_month     dvd_rel_day     imdb_rating
Min.   : 1.000   Min.   : 1.00   Min.   :1.900
1st Qu.: 3.000   1st Qu.: 7.00   1st Qu.:5.900
Median : 6.000   Median :15.00   Median :6.600
Mean   : 6.333   Mean   :15.01   Mean   :6.493
3rd Qu.: 9.000   3rd Qu.:23.00   3rd Qu.:7.300
Max.   :12.000   Max.   :31.00   Max.   :9.000
NA's   :8        NA's   :8
imdb_num_votes           critics_rating critics_score
Min.   :   180   Certified Fresh:135    Min.   :  1.00
1st Qu.:  4546   Fresh          :209    1st Qu.: 33.00
Median : 15116   Rotten         :307    Median : 61.00
Mean   : 57533                          Mean   : 57.69
3rd Qu.: 58300                          3rd Qu.: 83.00
Max.   :893008                          Max.   :100.00

audience_rating audience_score  best_pic_nom best_pic_win
Spilled:275     Min.   :11.00   no :629      no :644
Upright:376     1st Qu.:46.00   yes: 22      yes:  7
                Median :65.00
                Mean   :62.36
```

```
                  3rd Qu.:80.00
                  Max.   :97.00

 best_actor_win best_actress_win best_dir_win top200_box
 no :558        no :579          no :608      no :636
 yes: 93        yes: 72          yes: 43      yes: 15




   director           actor1             actor2
 Length:651         Length:651         Length:651
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character




    actor3             actor4             actor5
 Length:651         Length:651         Length:651
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character




   imdb_url            rt_url
 Length:651         Length:651
 Class :character   Class :character
 Mode  :character   Mode  :character
```

Reasoning for generabizability

We assume random sampling in this data set. However, due to the lack of the sampling method, we are inable to provide any information to the prior of the model.

Part 2: Data manipulation

###Create new variables

- Create new variable based on title_type: New variable should be called feature_film with levels yes (movies that are feature films) and no Create new variable based on genre: New variable should be called drama with levels yes (movies that are dramas) and no
- Create new variable based on mpaa_rating: New variable should be called mpaa_rating_R with levels yes (movies that are R rated) and no
- Create two new variables based on thtr_rel_month:

- New variable called oscar_season with levels yes (if movie is released in November, October, or December) and no
- New variable called summer_season with levels yes (if movie is released in May, June, July, or August) and no

Hide

```
movies <- mutate(movies, feature_film = as.factor(ifelse(movies$'title_type' == 'Feature
Film', 'yes', 'no')))
movies <- mutate(movies, drama = as.factor(ifelse(movies$'genre' == 'Drama', 'yes', 'no'
)))
movies <- mutate(movies, mpaa_rating_R = as.factor(ifelse(movies$mpaa_rating == 'R', 'ye
s', 'no')))
movies <- mutate(movies, oscar_season = as.factor(ifelse(movies$thtr_rel_month %in% c(10
:12), 'yes', 'no')))
movies <- mutate(movies, summer_season = as.factor(ifelse(movies$thtr_rel_month %in% c(5
:8), 'yes', 'no')))
```

Save only complete rows of our data

Hide

```
movies <- movies[complete.cases(movies),]
```

Part 3: Exploratory data analysis

Plots

Distribution of audience_score

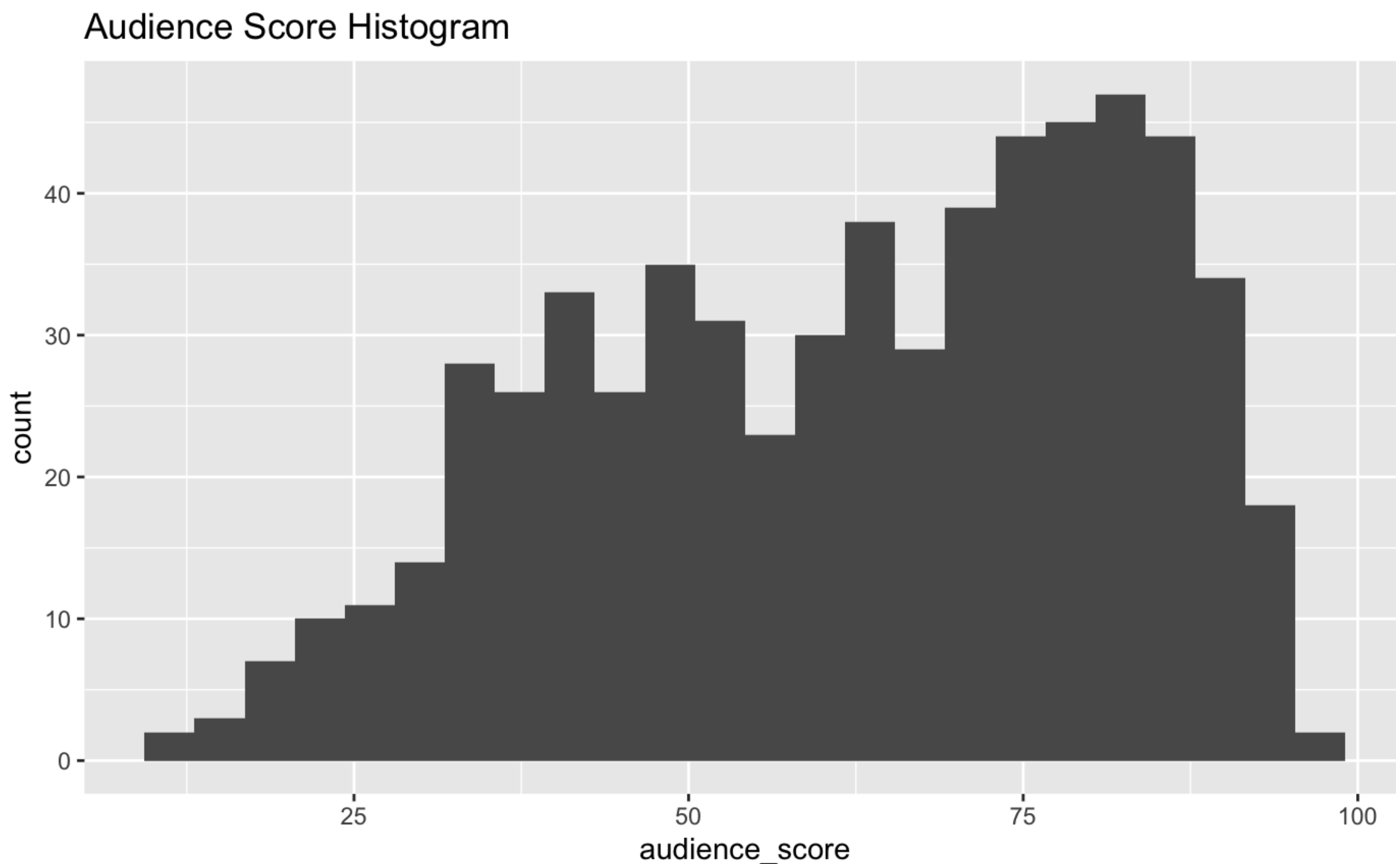
Hide

```
new_features <- select(movies, c('audience_score', 'feature_film', 'drama', 'mpaa_rating
_R', 'oscar_season', 'summer_season'))
summary(new_features)
```

```
 audience_score  feature_film drama     mpaa_rating_R
 Min.   :11.00   no : 46      no :321   no :300
 1st Qu.:46.00   yes:573      yes:298   yes:319
 Median :65.00
 Mean   :62.21
 3rd Qu.:80.00
 Max.   :97.00
 oscar_season summer_season
 no :440      no :418
 yes:179      yes:201
```

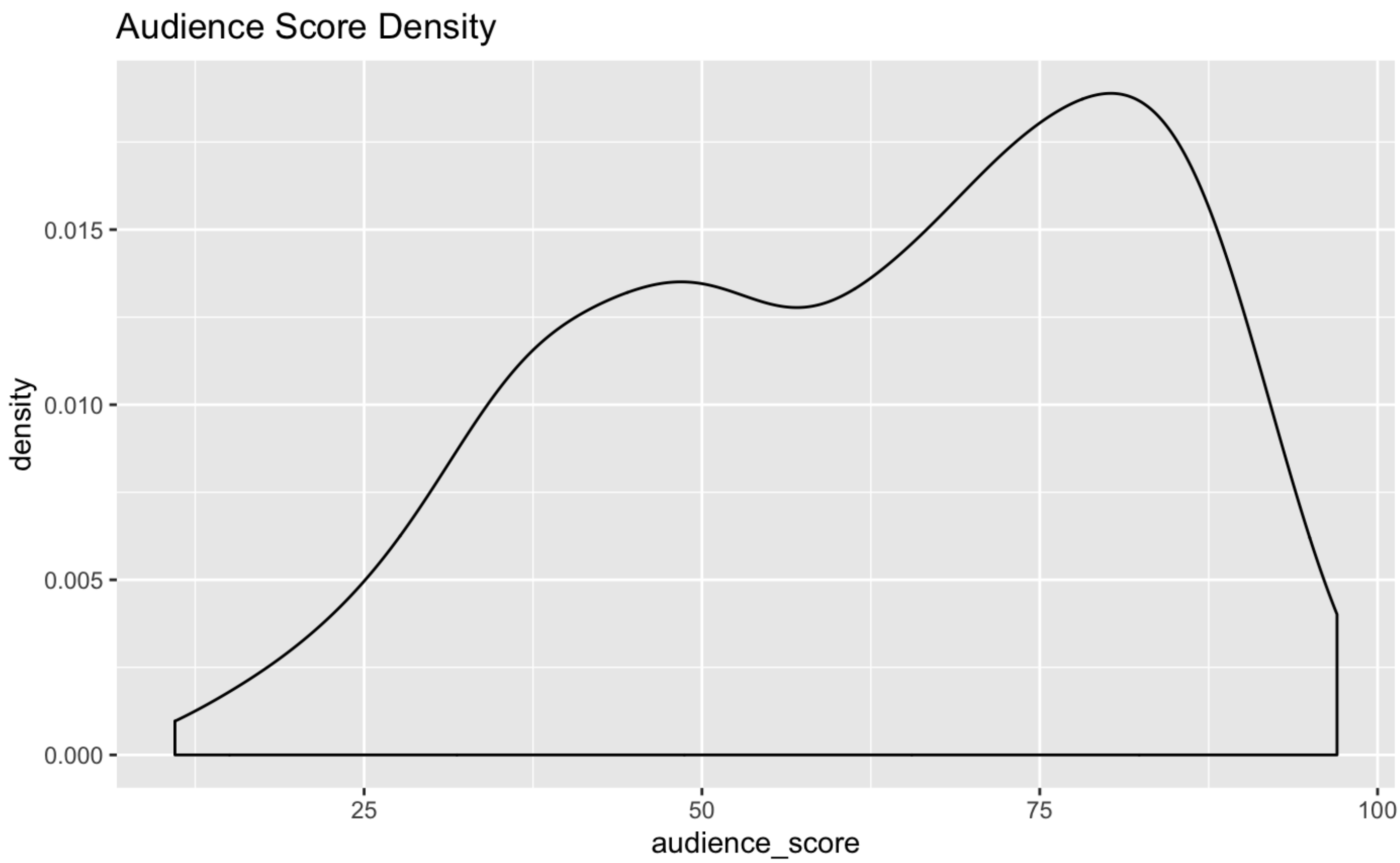
Hide

```
options(repr.plot.width = 5, repr.plot.height = 2)

audience_score_hist <- ggplot(data=movies, aes(x = audience_score)) +
  geom_histogram(bins=floor(sqrt(length(movies$audience_score)))) +
  ggtitle("Audience Score Histogram")
ggplot(data=movies, aes(x = audience_score)) +
  geom_histogram(bins=floor(sqrt(length(movies$audience_score)))) +
  ggtitle("Audience Score Histogram")
```



Hide

```
audience_score_density <- ggplot(movies, aes(x=audience_score)) +
  geom_density(alpha=.5) +
  ggtitle("Audience Score Density")
ggplot(movies, aes(x=audience_score)) +
  geom_density(alpha=.5) +
  ggtitle("Audience Score Density")
```



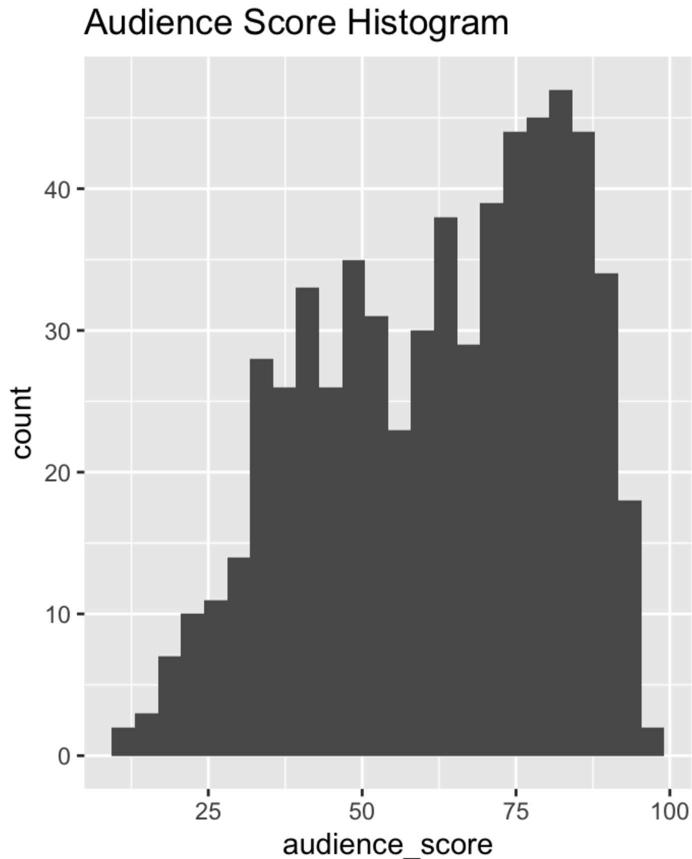
Hide

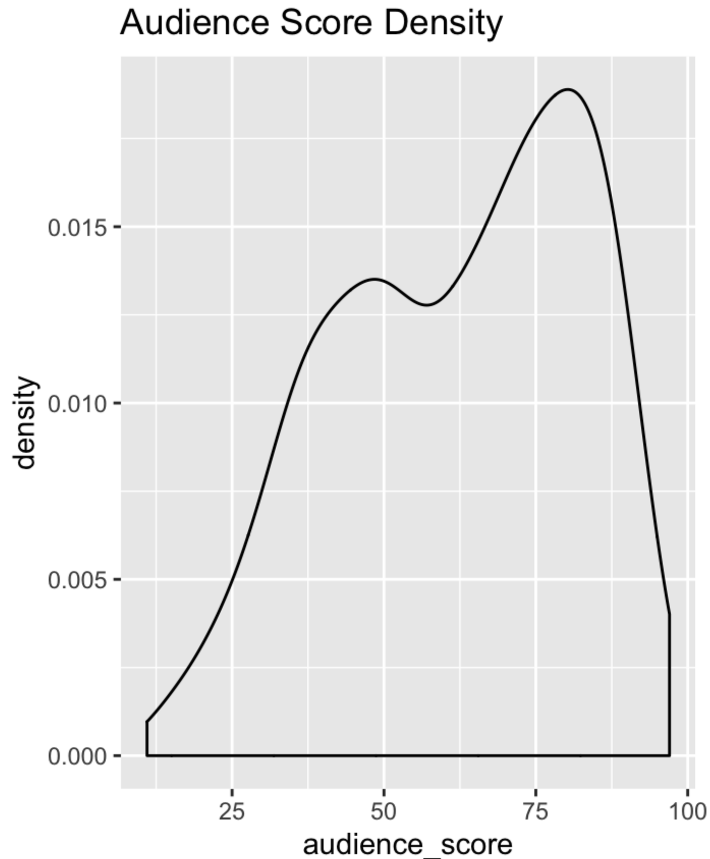
```
require(ggplot2)
require(gridExtra)
```

```
Loading required package: gridExtra

Attaching package: ‘gridExtra’

The following object is masked from ‘package:dplyr’:

    combine
```

Hide

```
grid.arrange(audience_score_hist, audience_score_density, nrow=1,  ncol=2)
```





Hide

```
grid.arrange
```

```
function (..., newpage = TRUE)
{
    if (newpage)
        grid.newpage()
    g <- arrangeGrob(...)
    grid.draw(g)
    invisible(g)
}
<bytecode: 0x7febf9708348>
<environment: namespace:gridExtra>
```

Conditional Histograms

Hide

```
film_hist <- ggplot(movies, aes(x=audience_score, fill=feature_film)) + geom_histogram(a
lpha=.5, position="dodge")
film_density <- ggplot(movies, aes(x=audience_score, fill=feature_film)) + geom_density
(alpha=.5)
```

Hide

```
drama_hist <- ggplot(movies, aes(x=audience_score, fill=drama)) + geom_histogram(alpha=.
5, position="dodge")
drama_density <- ggplot(movies, aes(x=audience_score, fill=drama)) + geom_density(alpha=
.5)
```

Hide

```
RR_hist <- ggplot(movies, aes(x=audience_score, fill=mpaa_rating_R)) + geom_histogram(al
pha=.5, position="dodge")
RR_density <- ggplot(movies, aes(x=audience_score, fill=mpaa_rating_R)) + geom_density(a
lpha=.5)
```

Hide

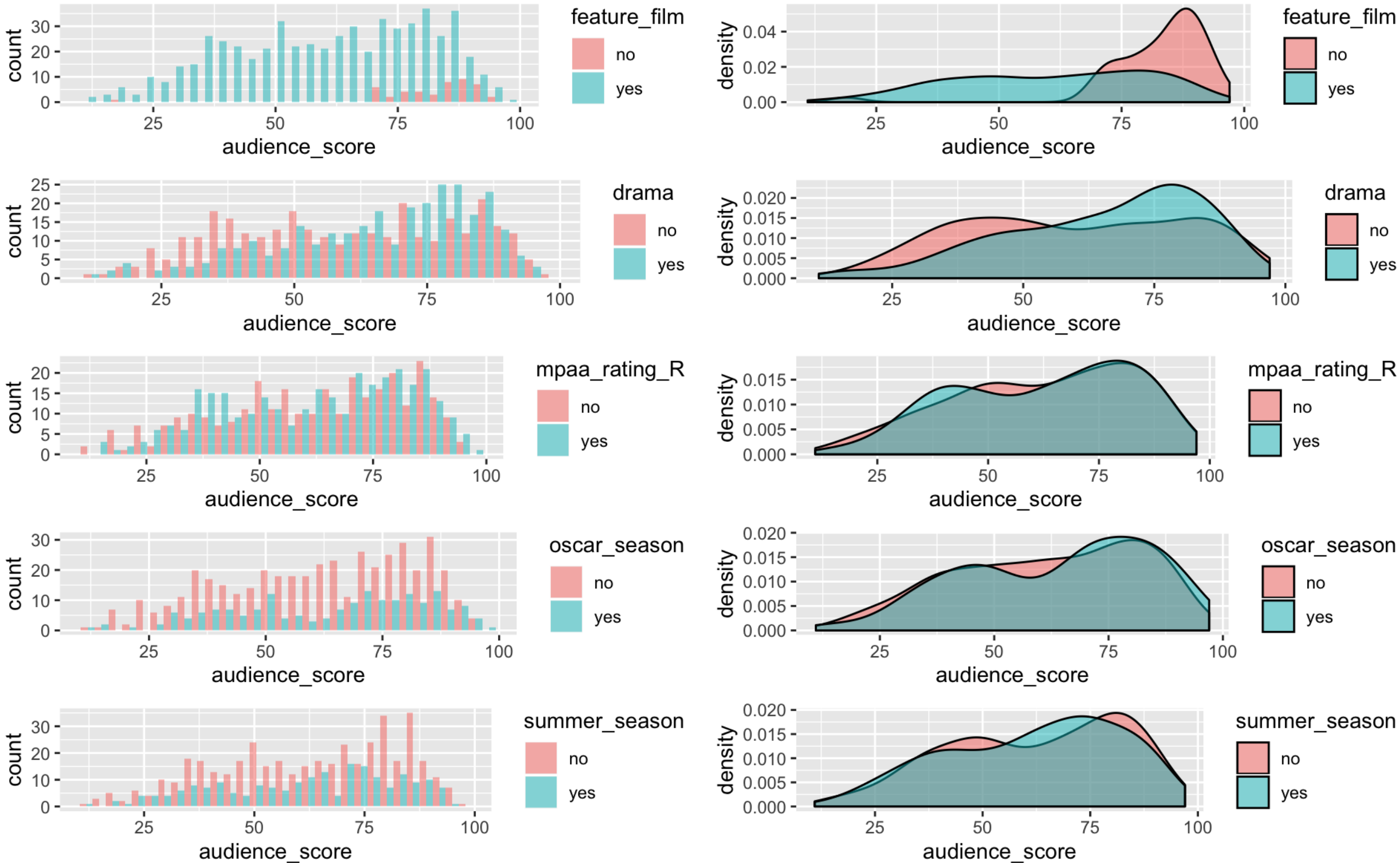
```
oscar_hist <- ggplot(movies, aes(x=audience_score, fill=oscar_season)) + geom_histogram
(alpha=.5, position="dodge")
oscar_density <- ggplot(movies, aes(x=audience_score, fill=oscar_season)) + geom_density
(alpha=.5)
```

Hide

```
summer_hist <- ggplot(movies, aes(x=audience_score, fill=summer_season)) + geom_histogra
m(alpha=.5, position="dodge")
summer_density <- ggplot(movies, aes(x=audience_score, fill=summer_season)) + geom_densi
ty(alpha=.5)
```

Hide

```
grid.arrange(film_hist, film_density, drama_hist, drama_density, RR_hist, RR_density, os
car_hist, oscar_density, summer_hist, summer_density, ncol=2)
```



Feature film & drama have a much more overlap of the densities. With more overlap, we can see that there is a relatiohip the variable and autdience_score(responce) since different values of the variables will affect autdience_score.

###Summary statistics

Quantiles

To examine which of the new parameters are most descriptive, we will look at their summary quantiles first.

Hide

```
movies %>% group_by(feature_film) %>% summarise(min=min(audience_score), q25=quantile(audience_score,0.25), median=median(audience_score), mean=mean(audience_score), q75=quantile(audience_score,0.75), max=max(audience_score))
```

feature_film <fctr>	min <dbl>	q25 <dbl>	median <dbl>	mean <dbl>	q75 <dbl>	max <dbl>
no	19	78	86	82.54348	89	96
yes	11	45	63	60.57766	78	97

2 rows

Hide

```
movies %>% group_by(drama) %>% summarise(min=min(audience_score), q25=quantile(audience_score,0.25), median=median(audience_score), mean=mean(audience_score), q75=quantile(audience_score,0.75), max=max(audience_score))
```

drama <fctr>	min <dbl>	q25 <dbl>	median <dbl>	mean <dbl>	q75 <dbl>	max <dbl>
no	11	41	59	59.35202	79	97
yes	13	52	70	65.28859	80	95

2 rows

Hide

```
movies %>% group_by(mpaa_rating_R) %>% summarise(min=min(audience_score), q25=quantile(a
udience_score,0.25), median=median(audience_score), mean=mean(audience_score), q75=quant
ile(audience_score,0.75), max=max(audience_score))
```

mpaa_rating_R <fctr>	min <dbl>	q25 <dbl>	median <dbl>	mean <dbl>	q75 <dbl>	max <dbl>
no	11	46.75	65	62.03667	80	96
yes	14	45.00	65	62.37304	80	97

2 rows

Hide

```
movies %>% group_by(oscar_season) %>% summarise(min=min(audience_score), q25=quantile(au
dience_score,0.25), median=median(audience_score), mean=mean(audience_score), q75=quanti
le(audience_score,0.75), max=max(audience_score))
```

oscar_season <fctr>	min <dbl>	q25 <dbl>	median <dbl>	mean <dbl>	q75 <dbl>	max <dbl>
no	11	45.75	63.5	61.53864	79	96
yes	13	47.50	69.0	63.86034	81	97

2 rows

Hide

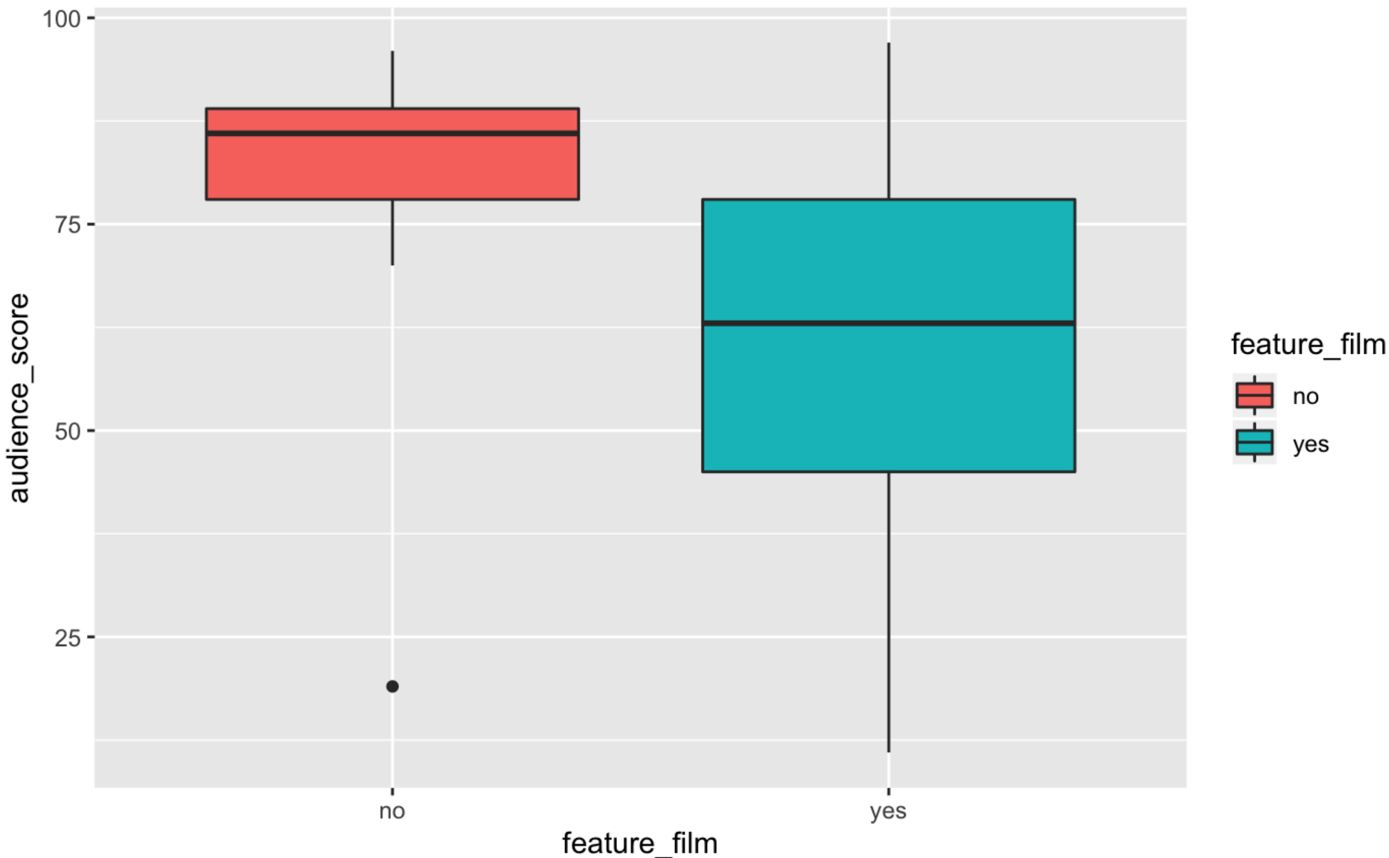
```
movies %>% group_by(summer_season) %>% summarise(min=min(audience_score), q25=quantile(a
udience_score,0.25), median=median(audience_score), mean=mean(audience_score), q75=quant
ile(audience_score,0.75), max=max(audience_score))
```

summer_season <fctr>	min <dbl>	q25 <dbl>	median <dbl>	mean <dbl>	q75 <dbl>	max <dbl>
no	13	46	65	62.38278	80	97
yes	11	45	64	61.85075	78	94

2 rows

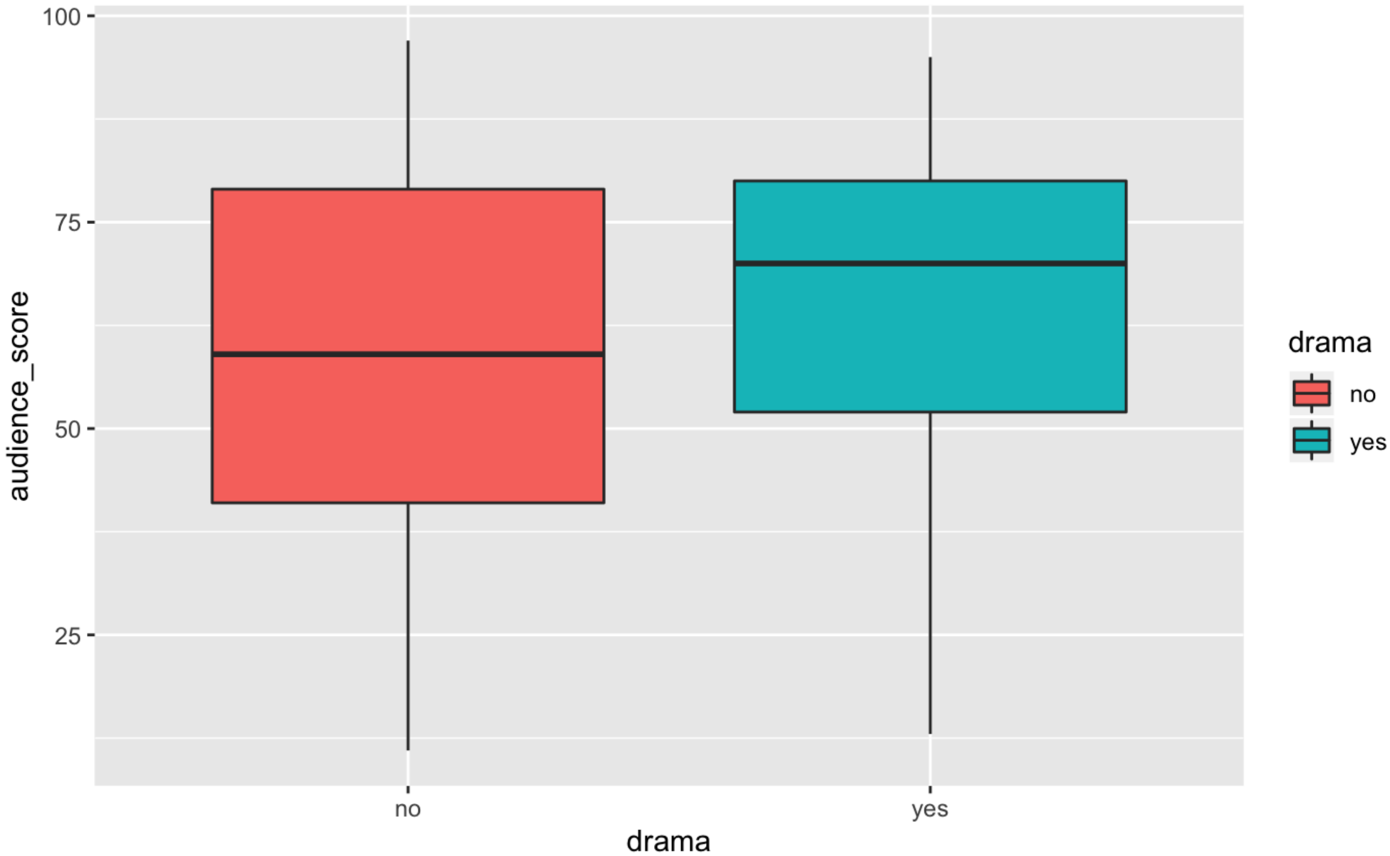
Hide

```
ggplot(movies, aes(x=feature_film, y=audience_score, fill=feature_film)) + geom_boxplot
()
```



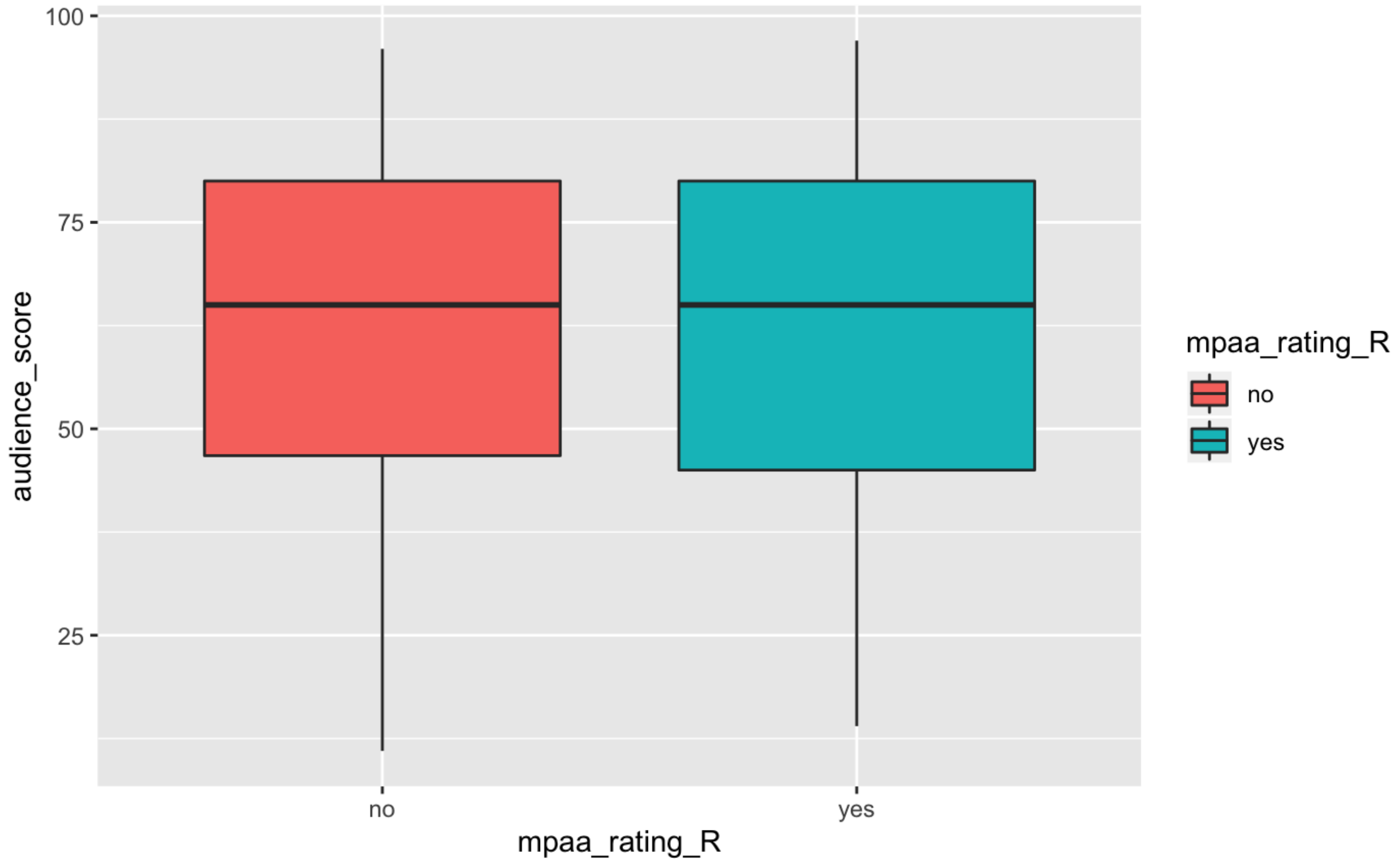
Hide

```
ggplot(movies, aes(x=drama, y=audience_score, fill=drama)) + geom_boxplot()
```



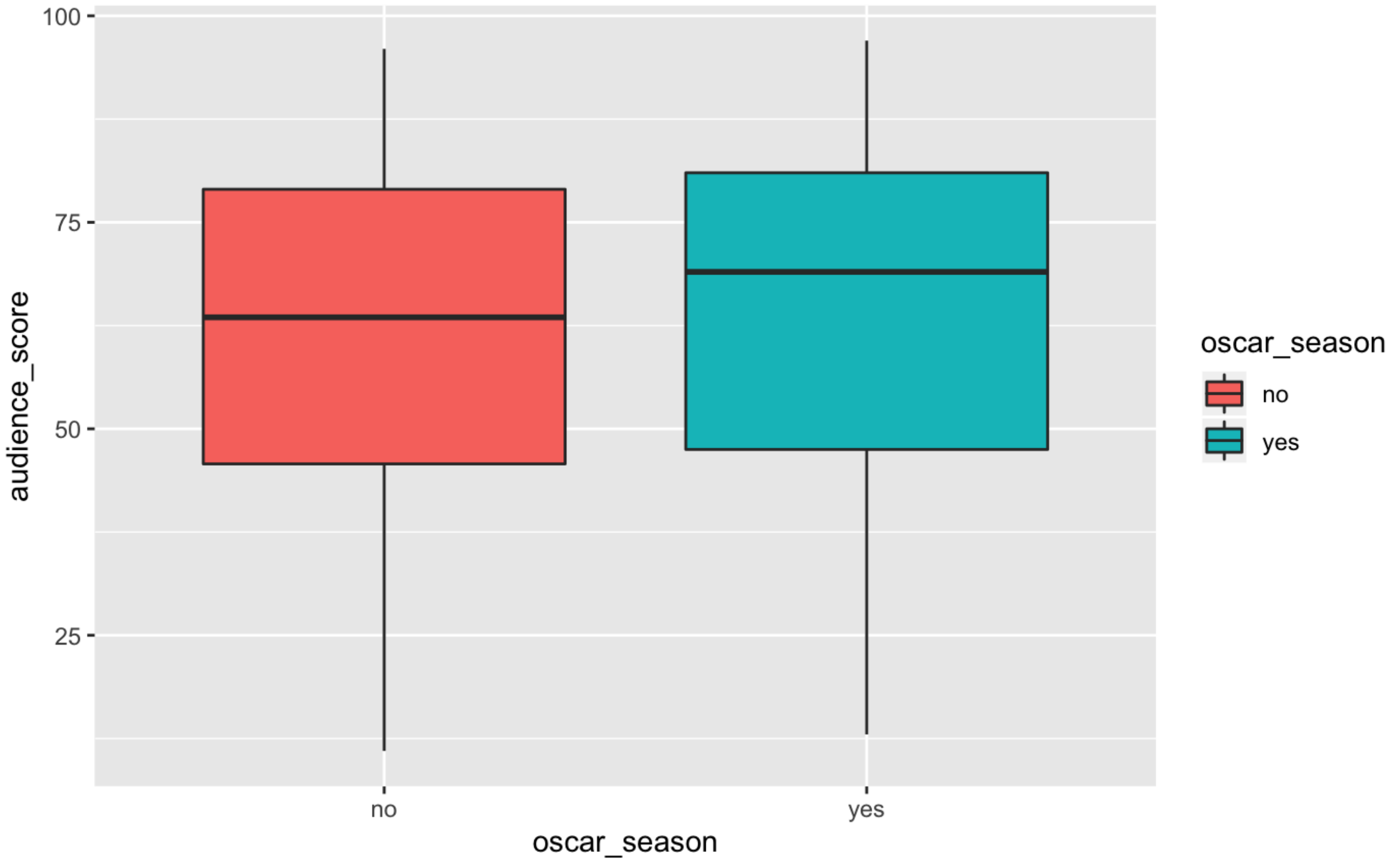
Hide

```
ggplot(movies, aes(x=mpaa_rating_R, y=audience_score, fill=mpaa_rating_R)) + geom_boxplo
t()
```



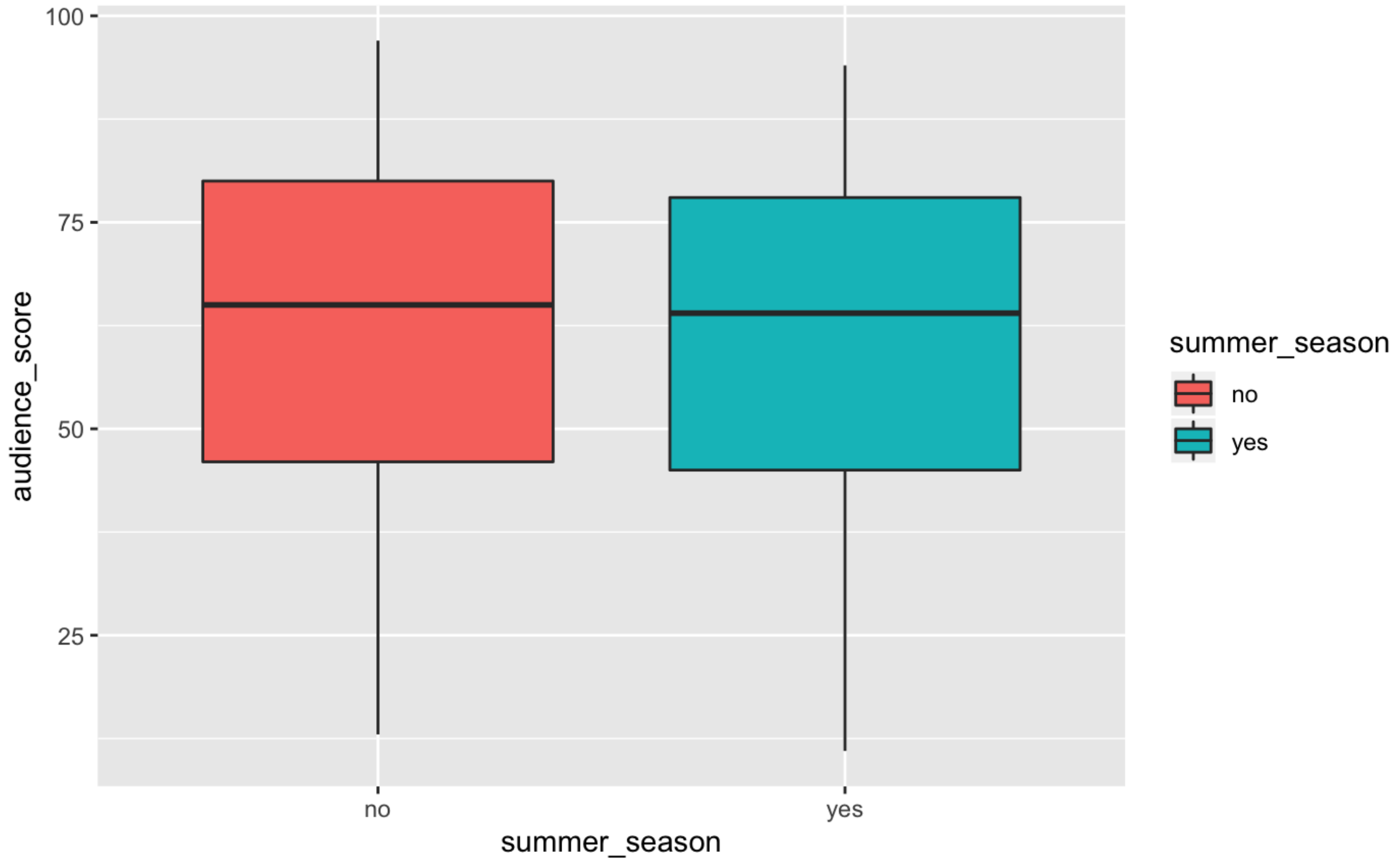
Hide

```
ggplot(movies, aes(x=oscar_season, y=audience_score, fill=oscar_season)) + geom_boxplot
()
```



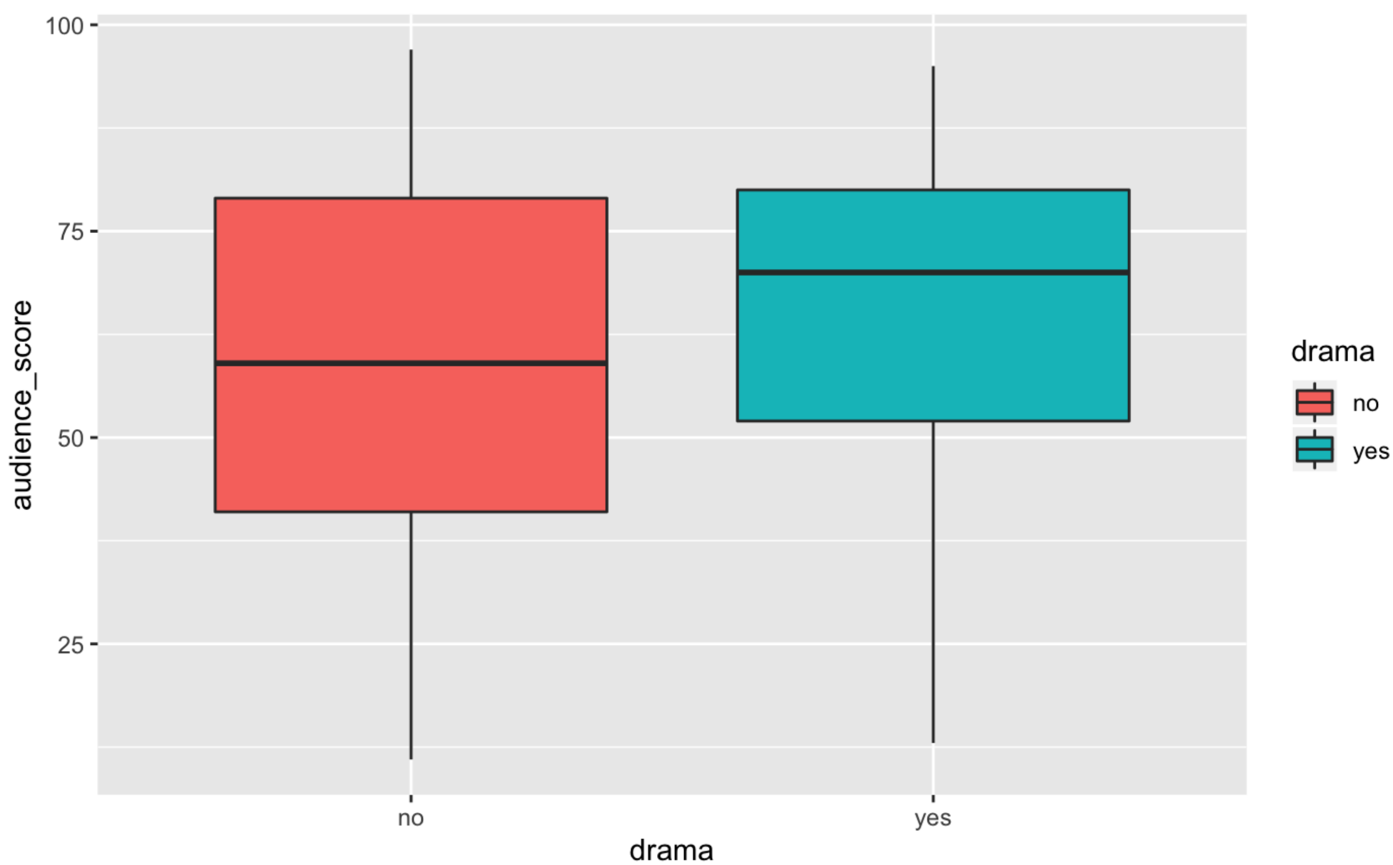
Hide

```
ggplot(movies, aes(x=summer_season, y=audience_score, fill=summer_season)) + geom_boxplo
t()
```



Hide

```
ggplot(movies, aes(x=drama, y=audience_score, fill=drama)) + geom_boxplot()
```



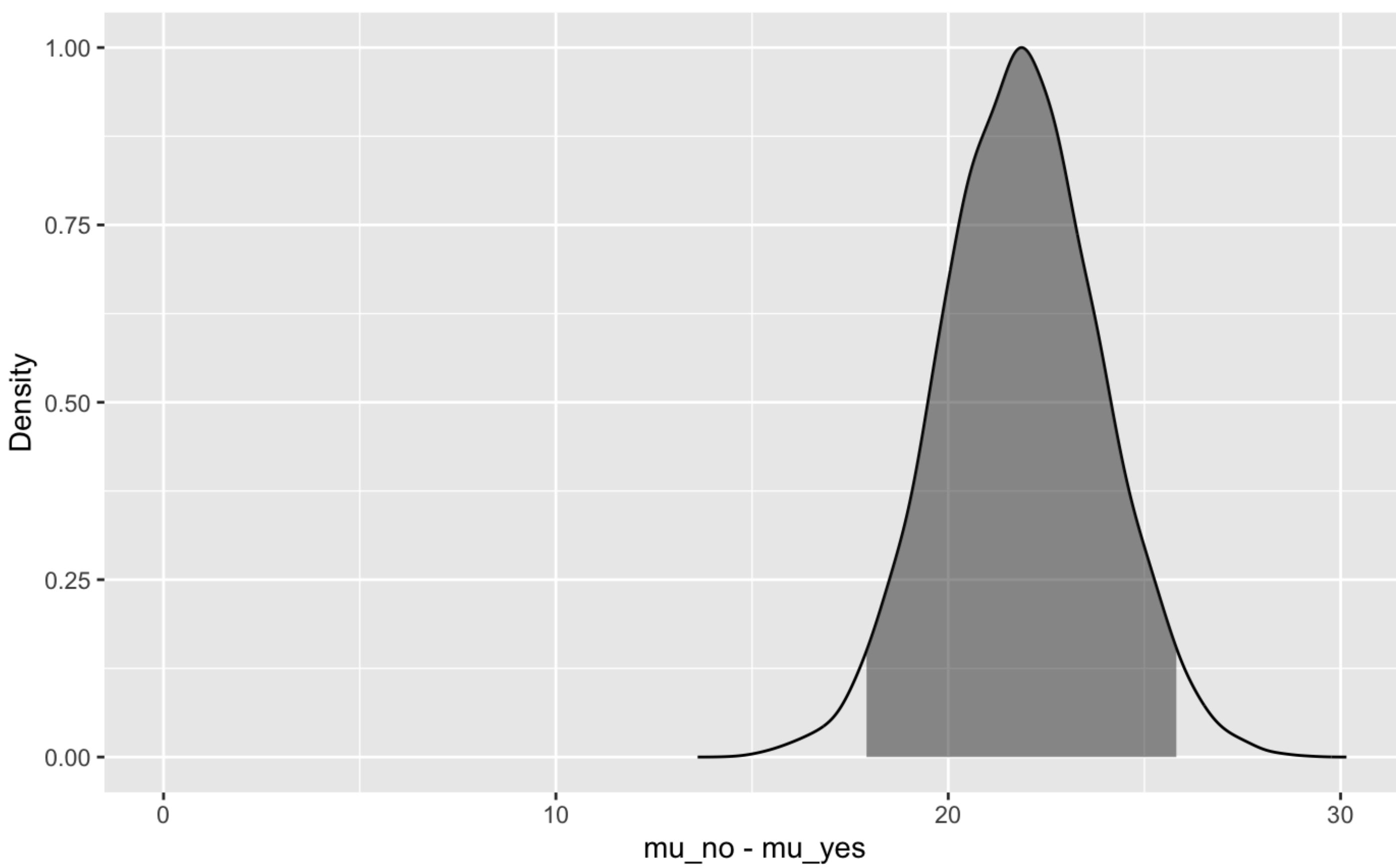
From the plots we can see that Feature_Film and drama have the most diffent distribution of IQR separately(big difference in adience score for different values) and are mostly related to the audience score.

####Baysien inference

Hide

```
bayes_inference(y=audience_score, x=feature_film, data=movies, statistic="mean", type="h
t", null=0, alternative="twosided")
```

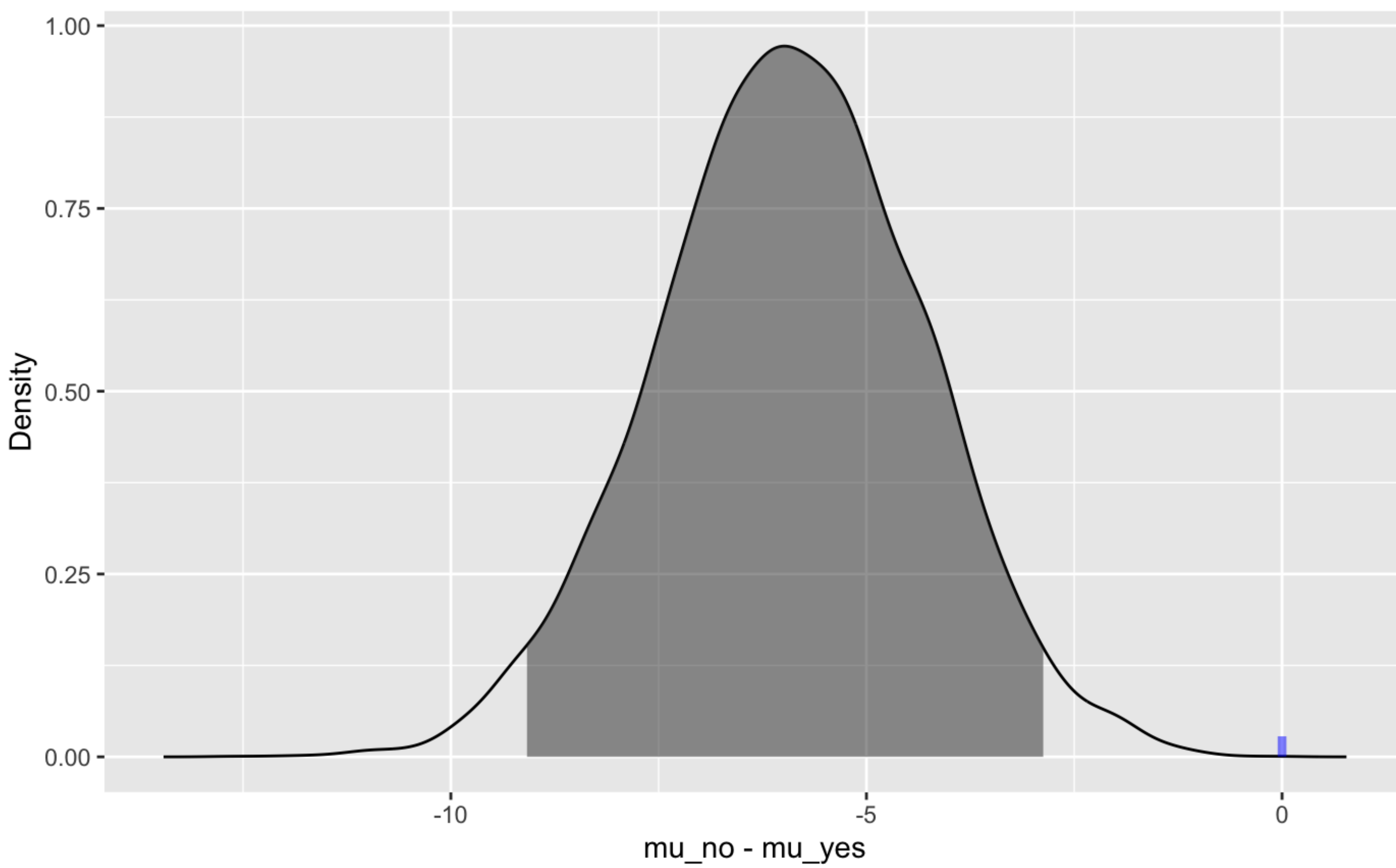
```
Response variable: numerical, Explanatory variable: categorical (2 levels)
n_no = 46, y_bar_no = 82.5435, s_no = 11.9177
n_yes = 573, y_bar_yes = 60.5777, s_yes = 19.8187
(Assuming intrinsic prior on parameters)
Hypotheses:
H1: mu_no  = mu_yes
H2: mu_no != mu_yes

Priors:
P(H1) = 0.5
P(H2) = 0.5

Results:
BF[H2:H1] = 1.212332e+13
P(H1|data) = 0
P(H2|data) = 1
```



Hide

```
bayes_inference(y=audience_score, x=drama, data=movies, statistic="mean", type="ht", nul
l=0, alternative="twosided")
```

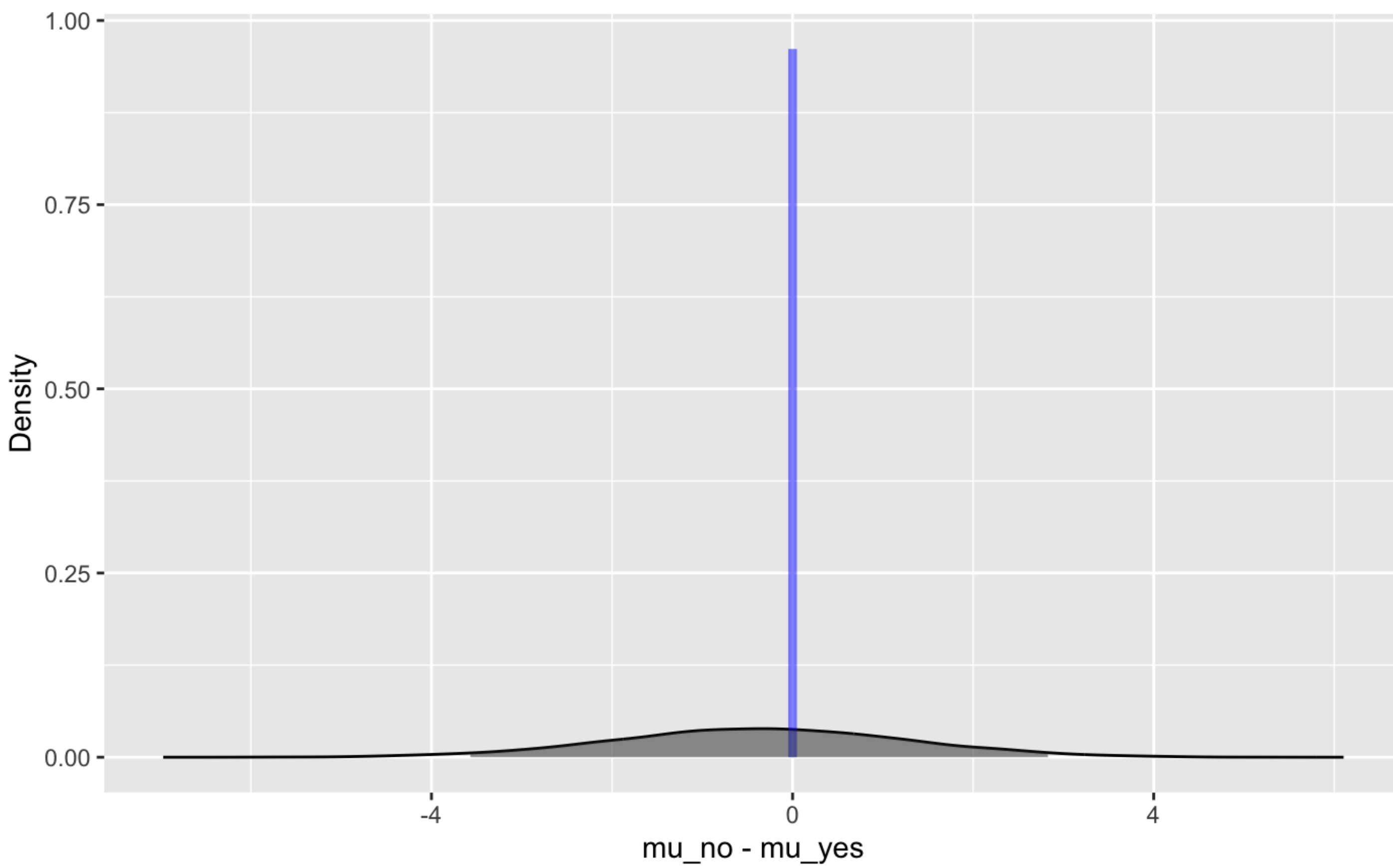
```
Response variable: numerical, Explanatory variable: categorical (2 levels)
n_no = 321, y_bar_no = 59.352, s_no = 21.1448
n_yes = 298, y_bar_yes = 65.2886, s_yes = 18.6305
(Assuming intrinsic prior on parameters)
Hypotheses:
H1: mu_no  = mu_yes
H2: mu_no != mu_yes

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
```



Hide

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

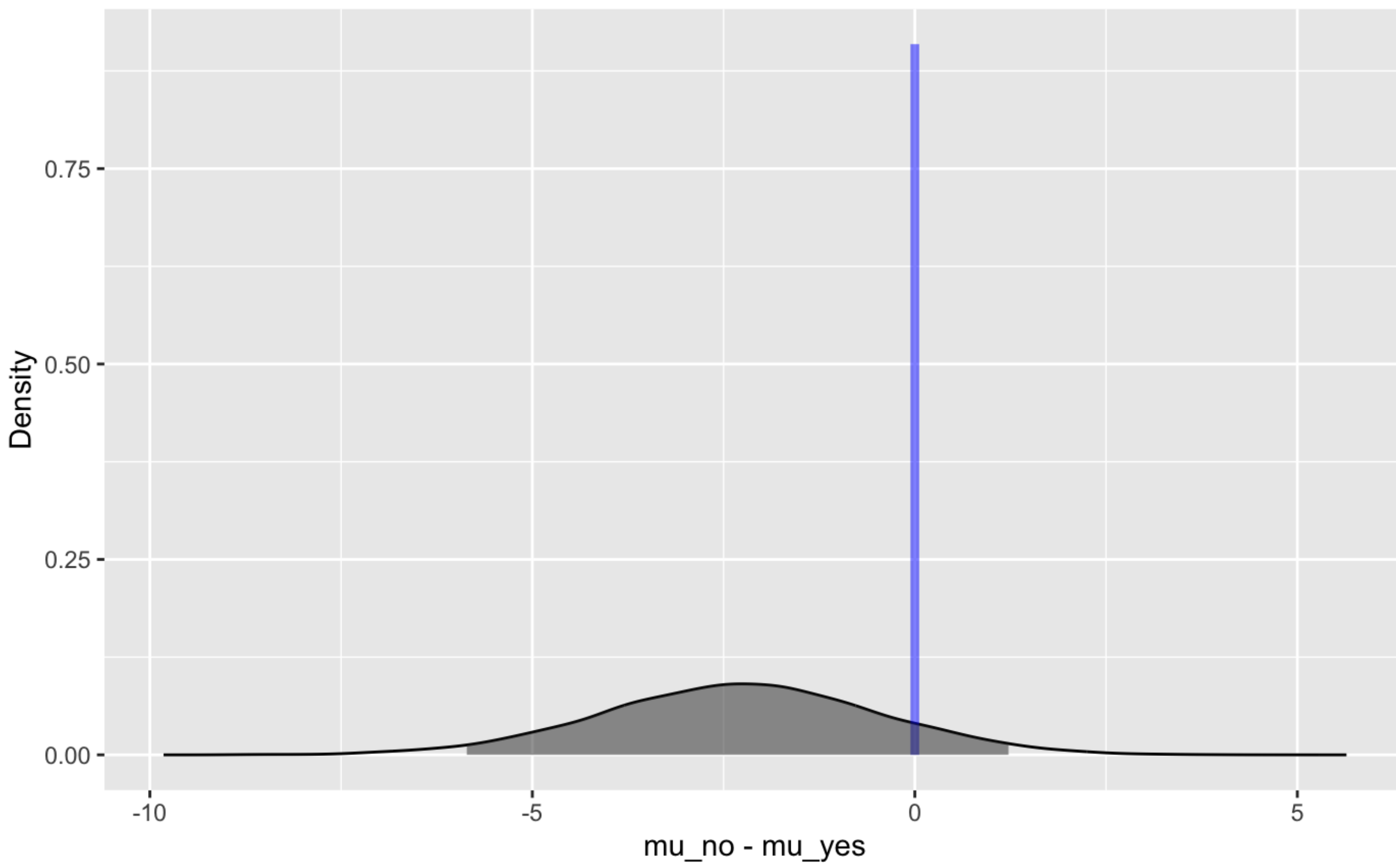
```
Response variable: numerical, Explanatory variable: categorical (2 levels)
n_no = 300, y_bar_no = 62.0367, s_no = 20.3187
n_yes = 319, y_bar_yes = 62.373, s_yes = 20.0743
(Assuming intrinsic prior on parameters)
Hypotheses:
H1: mu_no  = mu_yes
H2: mu_no != mu_yes

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
```



Hide

```
bayes_inference(y=audience_score, x=oscar_season, data=movies, statistic="mean", type="h
t", null=0, alternative="twosided")
```

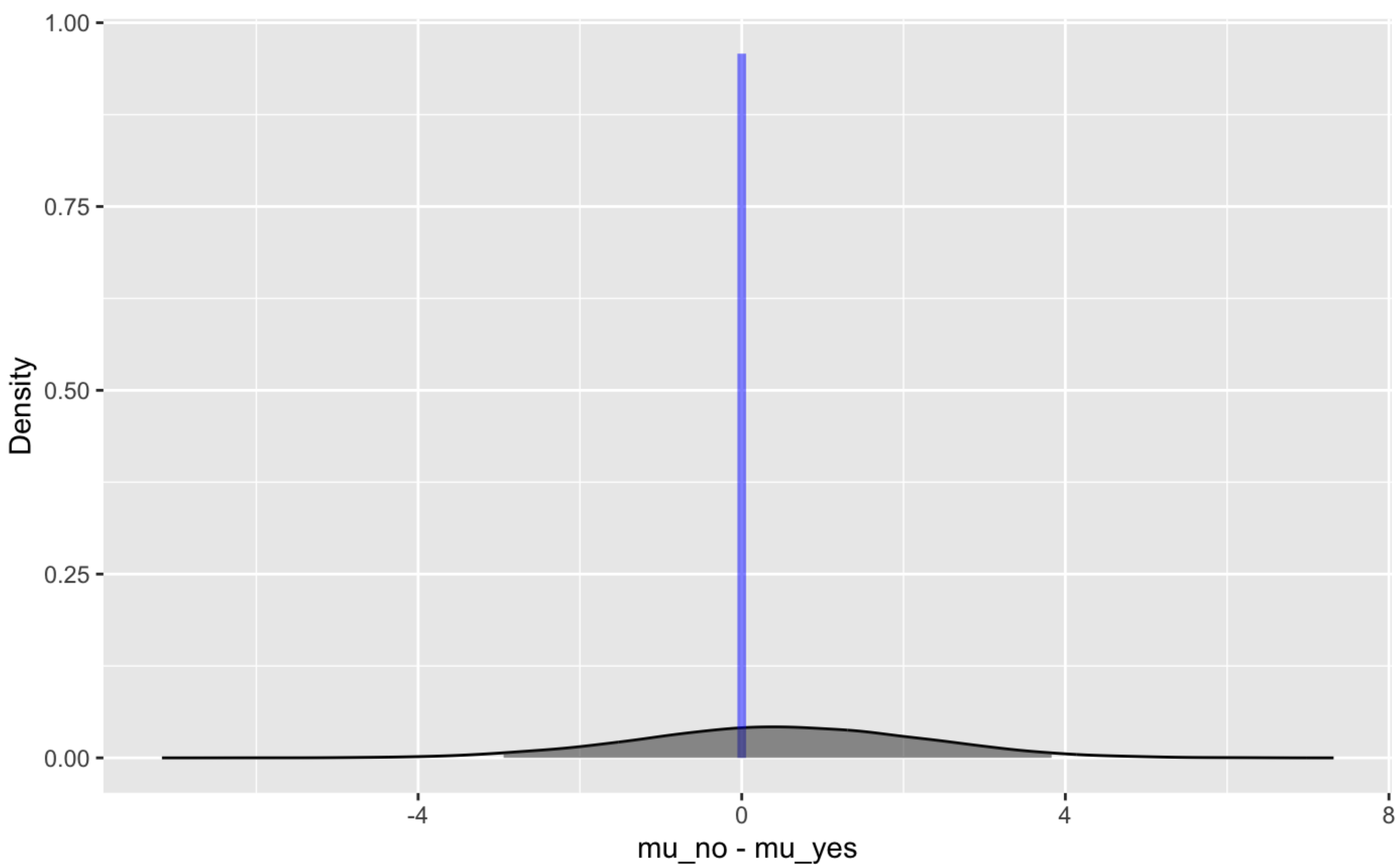
```
Response variable: numerical, Explanatory variable: categorical (2 levels)
n_no = 440, y_bar_no = 61.5386, s_no = 20.107
n_yes = 179, y_bar_yes = 63.8603, s_yes = 20.3118
(Assuming intrinsic prior on parameters)
Hypotheses:
H1: mu_no  = mu_yes
H2: mu_no != mu_yes

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
```



Hide

```
bayes_inference(y=audience_score, x=summer_season, data=movies, 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
```



Summary of Bayes Factor: BF(feature_film) = 1.212332e+13 BF(drama) = 34.6357 BF(mpaa_rating_R) = 24.8392 BF(summer_season) = 22.7623 BF(oscar_season) = 10.019

This shows that feature_film, feature_film mpaa_rating_R and summer_season have strong evidence in supporting relationship with audience_score. On the other hand, Oscar_season does not have strong eveidence in supporting relationship with audience_score.

Part 4: Modeling

Model parameters select

We only select variables that might have predictive power on our target(audience_score) and exclude some parameters such as actors, audience_rating since theycould influnce our the acuracy of our model.

Hide

```
mpar = c('feature_film', 'drama', 'runtime', 'mpaa_rating_R', 'thtr_rel_year', 'oscar_se
ason', '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', 'a
udience_score')
select_movies<- select(movies, mpar)
```

Train and Test set splitting

We split of 80% of the movies into trainning set and 20% of the movies into test set

Hide

```
set.seed(123)
train_ind <- createDataPartition(select_movies$audience_score, p = 0.8,list = FALSE)
train <- select_movies[train_ind, ]
test <- select_movies[-train_ind, ]
```

Bayesian Model Averaging

Fit the model

Hide

```
set.seed(123)
# We use the Bayesian linear regression, `bas.lm` function in the `BAS` package
bma_regressor <- bas.lm(audience_score ~ .,
                     data = train,
                     prior = "BIC",
                     modelprior = uniform(),
                     method = "MCMC",
                     MCMC.iterations = 10^7)
```

Marginal posterior inclusion probabilities for each variable

Hide

```
bma_regressor
```

```
Call:
bas.lm(formula = audience_score ~ ., data = train, prior = "BIC",
    modelprior = uniform(), method = "MCMC", MCMC.iterations = 10^7)


 Marginal Posterior Inclusion Probabilities:
         Intercept      feature_filmyes             dramayes
           1.00000              0.05710              0.05351
           runtime     mpaa_rating_Ryes        thtr_rel_year
           0.63025              0.05809              0.07313
    oscar_seasonyes     summer_seasonyes          imdb_rating
           0.04695              0.04478              1.00000
     imdb_num_votes        critics_score      best_pic_nomyes
           0.06680              0.96234              0.05824
    best_pic_winyes    best_actor_winyes  best_actress_winyes
           0.04793              0.05528              0.07370
    best_dir_winyes        top200_boxyes
           0.07950              0.04502
```

Top 5 most probably models

Hide

```
summary(bma_regressor)
```

```
                     P(B != 0 | Y)    model 1       model 2
Intercept                1.0000000     1.0000     1.0000000
feature_filmyes          0.0571030     0.0000     0.0000000
dramayes                 0.0535108     0.0000     0.0000000
runtime                  0.6302524     1.0000     0.0000000
mpaa_rating_Ryes         0.0580908     0.0000     0.0000000
thtr_rel_year            0.0731331     0.0000     0.0000000
oscar_seasonyes          0.0469476     0.0000     0.0000000
summer_seasonyes         0.0447787     0.0000     0.0000000
imdb_rating              0.9999976     1.0000     1.0000000
imdb_num_votes           0.0667997     0.0000     0.0000000
critics_score            0.9623405     1.0000     1.0000000
best_pic_nomyes          0.0582386     0.0000     0.0000000
best_pic_winyes          0.0479298     0.0000     0.0000000
best_actor_winyes        0.0552818     0.0000     0.0000000
best_actress_winyes      0.0736994     0.0000     0.0000000
best_dir_winyes          0.0794951     0.0000     0.0000000
top200_boxyes            0.0450216     0.0000     0.0000000
BF                              NA     1.0000     0.5561334
PostProbs                       NA     0.2835     0.1593000
R2                              NA     0.7574     0.7538000
dim                             NA     4.0000     3.0000000
logmarg                         NA -2684.8738 -2685.4605025
                           model 3       model 4       model 5
Intercept             1.000000e+00  1.000000e+00  1.000000e+00
feature_filmyes       0.000000e+00  0.000000e+00  0.000000e+00
dramayes              0.000000e+00  0.000000e+00  0.000000e+00
runtime               1.000000e+00  1.000000e+00  1.000000e+00
mpaa_rating_Ryes      0.000000e+00  0.000000e+00  0.000000e+00
thtr_rel_year         1.000000e+00  0.000000e+00  0.000000e+00
oscar_seasonyes       0.000000e+00  0.000000e+00  0.000000e+00
summer_seasonyes      0.000000e+00  0.000000e+00  0.000000e+00
imdb_rating           1.000000e+00  1.000000e+00  1.000000e+00
imdb_num_votes        0.000000e+00  1.000000e+00  0.000000e+00
critics_score         1.000000e+00  1.000000e+00  1.000000e+00
best_pic_nomyes       0.000000e+00  0.000000e+00  0.000000e+00
best_pic_winyes       0.000000e+00  0.000000e+00  0.000000e+00
best_actor_winyes     0.000000e+00  0.000000e+00  0.000000e+00
best_actress_winyes   0.000000e+00  0.000000e+00  0.000000e+00
best_dir_winyes       0.000000e+00  0.000000e+00  1.000000e+00
top200_boxyes         0.000000e+00  0.000000e+00  0.000000e+00
BF                    8.084384e-02  8.094043e-02  6.927468e-02
PostProbs             2.300000e-02  2.290000e-02  1.990000e-02
R2                    7.580000e-01  7.580000e-01  7.579000e-01
dim                   5.000000e+00  5.000000e+00  5.000000e+00
logmarg              -2.687389e+03 -2.687388e+03 -2.687543e+03
```

Model summary

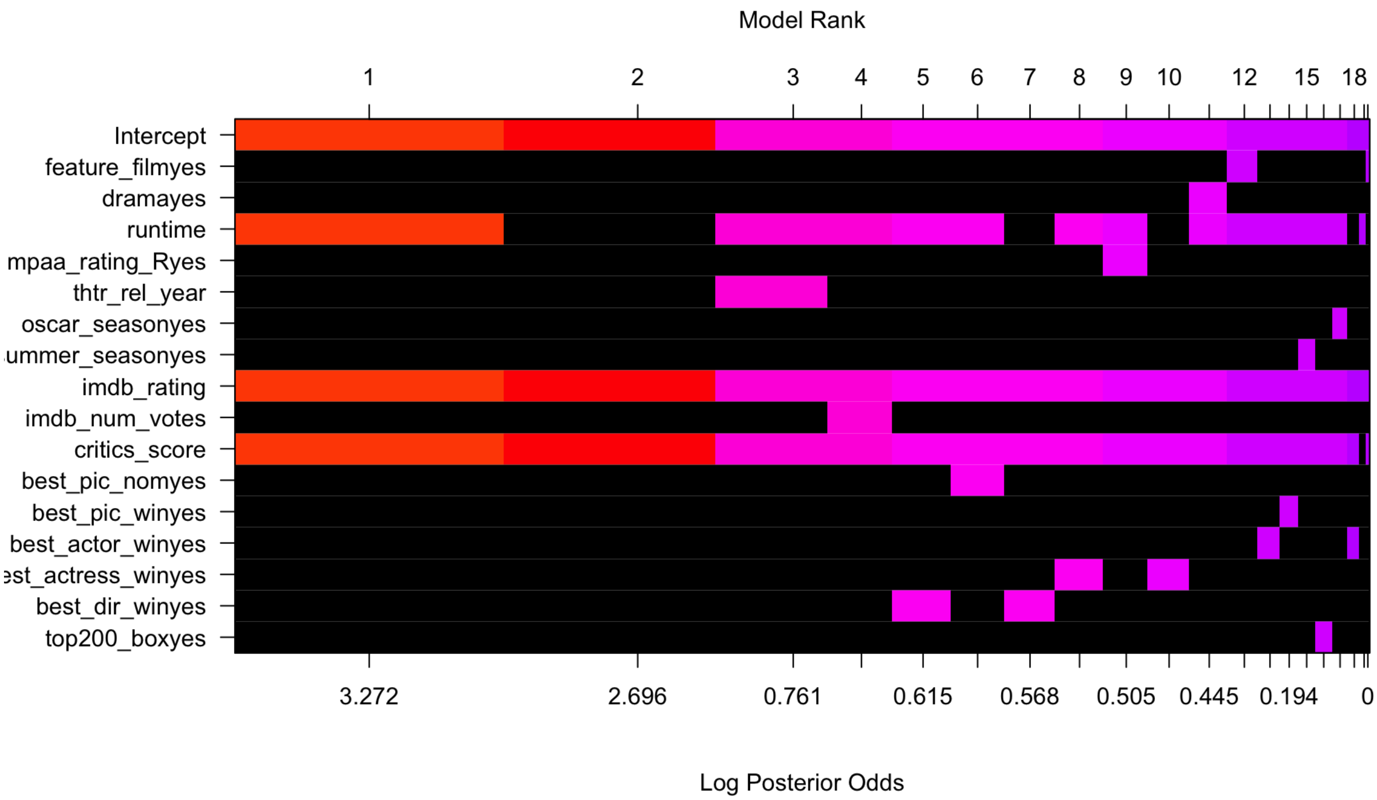
The 3 Most Highest Marginal Posterior Inclusion Probability Variables: Variables Marginal Posterior Inclusion Probability(>0.5) critics_score 0.9623405 imdb_rating 0.9999976 runtime 0.6302524 Posterior Probability : The model that includes run_time, imdb_rating & critics_score has the highest posterior probability 0.28. The seconde highest posterior probability model also includes run_time, imdb_rating, critics_score with a posterior probability

0.1593000. Additionally, the model contains best_pic_nomyes, mpaa_rating_R. ALthough the posterior probability seems quite small, but itt is much larger than the uniform prior probability assigned to it, since there are 2^{17} possible models. Now we have 3 potential predictors: runtime, imdb_rating, critics_score.

Visualization

Hide

```
image(bma_regressor, rotate=FALSE, top.models = 20)
```

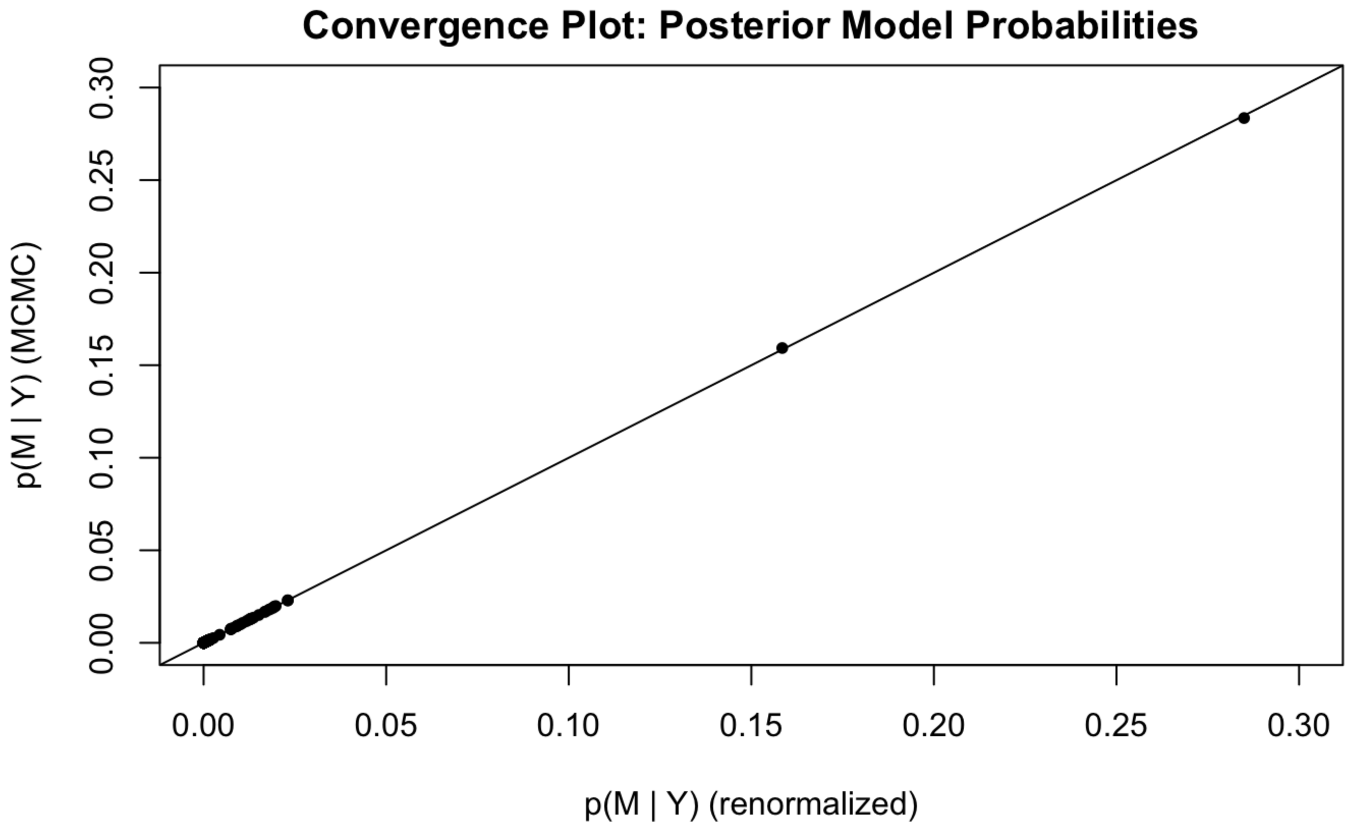


The plot shows just the top 20 models. The highest probability model is the leftmost column. Each row corresponds to one of the predictor variables. The color corresponds to the log posterior odds. We can see that runtime, imdb_rating, critics_score are often included in the models.

Marginal Posterior Inclusion Probabilities

Hide

```
diagnostics(bma_regressor, type="model",pch=20)
```



Model posterior for coefficients:

Hide

```
coefficients(bma_regressor, estimator = 'BMA')
```

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  5649 models
                     post mean   post SD     post p(B != 0)
Intercept             6.237e+01   4.442e-01   1.000e+00
feature_filmyes      -8.148e-02   5.654e-01   5.710e-02
dramayes              3.193e-02   2.561e-01   5.351e-02
runtime              -4.187e-02   3.759e-02   6.303e-01
mpaa_rating_Ryes     -4.094e-02   2.719e-01   5.809e-02
thtr_rel_year        -3.230e-03   1.622e-02   7.313e-02
oscar_seasonyes      -1.625e-02   2.368e-01   4.695e-02
summer_seasonyes      9.947e-03   2.098e-01   4.478e-02
imdb_rating           1.482e+01   7.652e-01   1.000e+00
imdb_num_votes        2.603e-07   1.494e-06   6.680e-02
critics_score         8.330e-02   2.896e-02   9.623e-01
best_pic_nomyes       1.270e-01   8.694e-01   5.824e-02
best_pic_winyes      -8.962e-02   1.122e+00   4.793e-02
best_actor_winyes    -4.738e-02   3.713e-01   5.528e-02
best_actress_winyes  -1.168e-01   5.846e-01   7.370e-02
best_dir_winyes      -1.637e-01   7.630e-01   7.950e-02
top200_boxyes        -3.446e-02   6.770e-01   4.502e-02
```

We can provide 95% credible intervals for these coefficients:

Hide

```
confint(coefficients(bma_regressor))
```

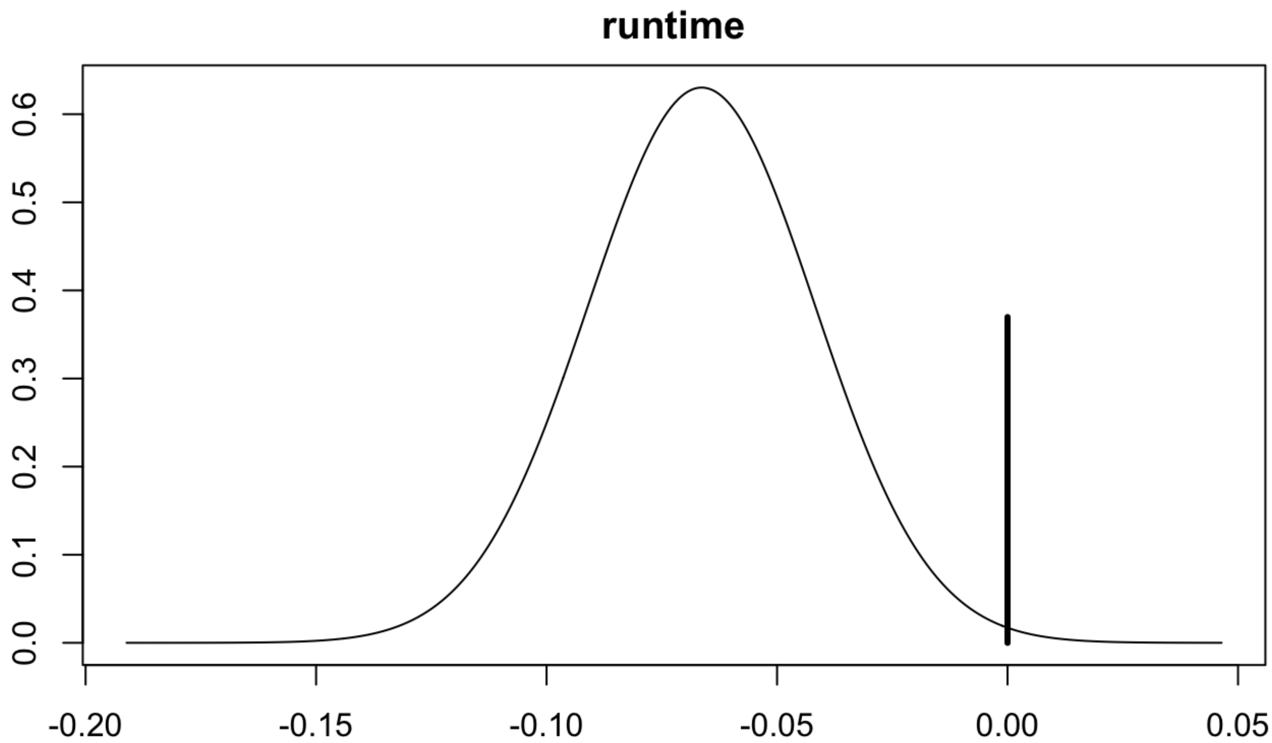
```
                           2.5%        97.5%          beta
Intercept           61.513188914 6.323998e+01  6.237298e+01
feature_filmyes     -0.437480410 5.581645e-01 -8.147621e-02
dramayes            -0.006726647 2.127862e-01  3.192507e-02
runtime             -0.101030522 0.000000e+00 -4.186946e-02
mpaa_rating_Ryes    -0.128649291 1.085443e-01 -4.093980e-02
thtr_rel_year       -0.040745178 6.785001e-04 -3.230353e-03
oscar_seasonyes      0.000000000 0.000000e+00 -1.624908e-02
summer_seasonyes     0.000000000 0.000000e+00  9.947073e-03
imdb_rating         13.361425137 1.647444e+01  1.481742e+01
imdb_num_votes       0.000000000 3.248859e-06  2.602539e-07
critics_score        0.024212779 1.469027e-01  8.330466e-02
best_pic_nomyes     -0.005877975 1.474743e+00  1.269659e-01
best_pic_winyes      0.000000000 0.000000e+00 -8.962379e-02
best_actor_winyes   -0.202008608 3.706423e-02 -4.737735e-02
best_actress_winyes -1.570204823 0.000000e+00 -1.168129e-01
best_dir_winyes     -2.149156282 1.182404e-02 -1.636917e-01
top200_boxyes        0.000000000 0.000000e+00 -3.446404e-02
attr(,"Probability")
[1] 0.95
attr(,"class")
[1] "confint.bas"
```

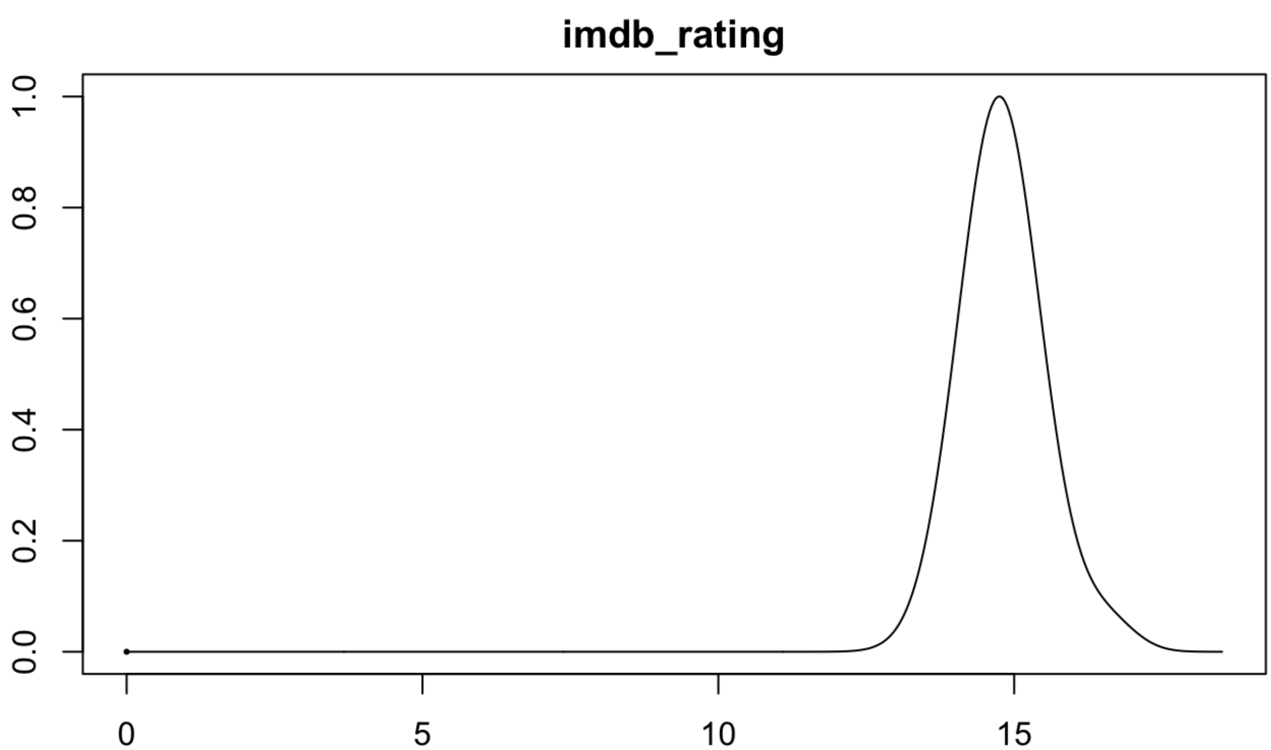
Based on this data, there is a 95% chance that coefficient of imdb_rating lies from 1.331548e+01 to 1.636430e+01.

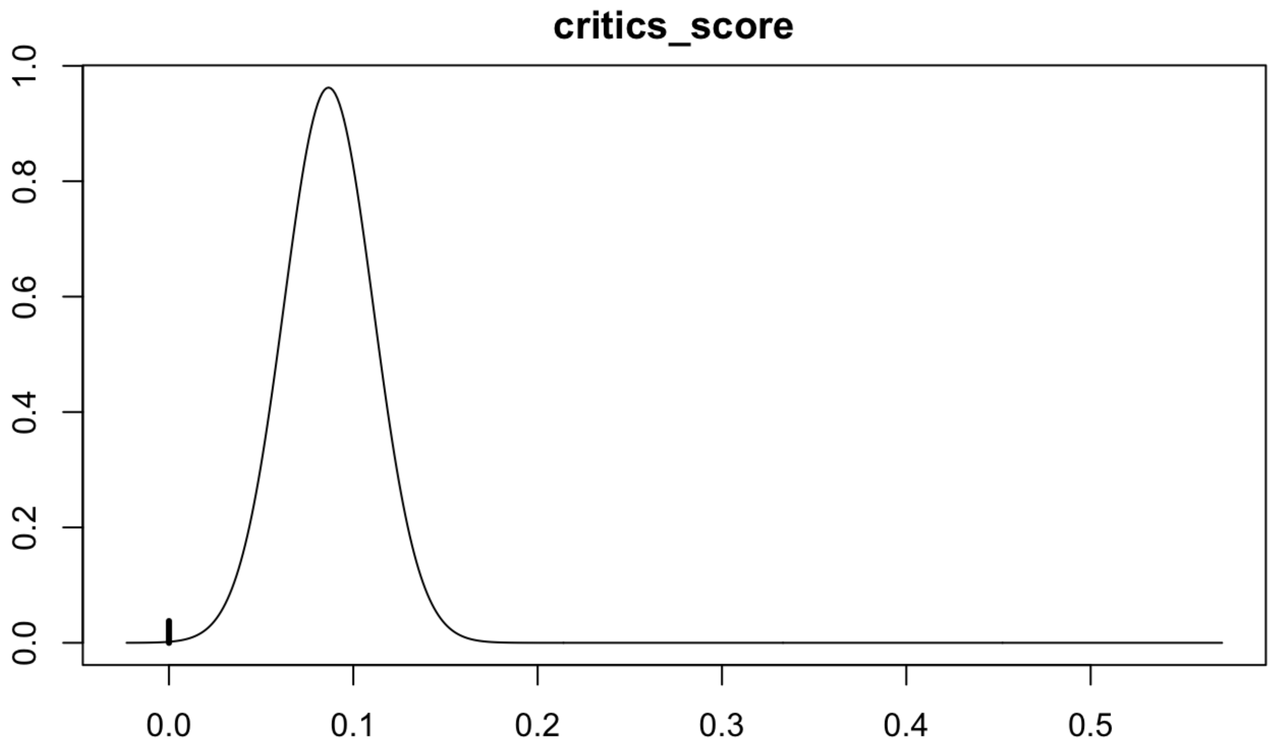
We can visualize the posterior distribution of the coefficients using the baysien model averaging approach. The posterior distribution of the coefficients of runtime, imdb_rating, critics_score are shown below.

Hide

```
plot(coefficients(bma_regressor), subset=c(4, 9, 11), ask=FALSE)
```


runtime
0.0
0.1
0.2
0.3
0.4
0.5
0.6
-0.20
-0.15
-0.10
-0.05
0.00
0.05


imdb_rating
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15



Before moving on to prediction, our top model has revealed runtime, imdb_rating & critics_score as most informative regression parameters.

These results show that also mpaa_rating_R & summer_season have some additional influence, while oscar season, suprisingly, has the lowest BF.

Part 5: Prediction

Predictions of the test set

Now we will use Bayesian predictive distribution for predictions and interpretation of predictions.

First we find the predictive values under the *Best Predictive Model* (`BPM`), the one which has predictions closest to BMA and corresponding posterior standard deviations.

Hide

```
pred <- predict(bma_regressor, newdata=test, estimator="BPM", se.fit=TRUE)
```

95% Credible Interval for predictions

Hide

```
ci_pred <- confint(pred, parm = "pred")

df = data.frame(movie_title=movies[-train_ind, ]$title, audience_score=test$audience_sco
re, prediction=pred$Ybma, lower=ci_pred[,1], upper=ci_pred[,2])

head(df, 20)
```

	movie_title <fctr>	audie
1	Filly Brown	
2	Leap of Faith	
3	Rhinestone	
4	The Wood	
5	Fallen	
6	Imagine: John Lennon	
7	The Color Purple	
8	Viva Knievel!	
9	The English Patient	
10	The Last Kiss	

1-10 of 20 rows | 1-3 of 5 columns Previous **1** 2 Next

Hide

```
ci_pred
```

```
             2.5%     97.5%      pred
 [1,]  28.6257270  67.51078 48.068253
 [2,]  36.1480171  74.99828 55.573147
 [3,]  -4.4807950  34.69235 15.105780
 [4,]  49.2180661  88.02430 68.621183
 [5,]  47.6721620  86.58877 67.130465
 [6,]  67.1483210 106.04093 86.594627
 [7,]  61.6869881 100.72960 81.208294
 [8,] -17.5768706  21.92766  2.175393
 [9,]  54.8061578  93.93448 74.370318
[10,]  41.3647419  80.18524 60.774992
[11,]  66.5086250 105.43372 85.971175
[12,]  43.0292535  81.91164 62.470447
[13,]  31.4243601  70.28983 50.857096
[14,]  43.9210054  82.72615 63.323578
[15,]   8.8029493  47.76015 28.281548
[16,]  71.0780107 110.02923 90.553620
[17,]  32.7555414  71.56879 52.162167
[18,]  63.3390349 102.19529 82.767164
[19,]  -0.3554794  38.72032 19.182423
[20,]  29.0493470  67.96673 48.508037
[21,]  47.7632860  86.59068 67.176982
[22,]  43.3964518  82.45988 62.928165
[23,]  34.1303115  72.98676 53.558536
[24,]   9.7545819  48.72149 29.238037
[25,]  45.6180022  84.47508 65.046543
[26,]  32.7356547  71.56092 52.148286
[27,]  20.0926386  58.99271 39.542674
[28,]  70.3991614 109.31312 89.856141
[29,]  62.0687693 100.92665 81.497707
[30,]  44.1797048  83.08478 63.632241
[31,]  48.3488422  87.16792 67.758379
[32,]  42.0146038  80.88687 61.450738
[33,]  65.6603082 104.54437 85.102341
[34,]  31.4865455  70.32300 50.904771
[35,]  62.5146356 101.36167 81.938154
[36,]  33.3476398  72.22410 52.785872
[37,]  40.7040236  79.51181 60.107917
[38,]  36.6919758  75.53069 56.111331
[39,]  37.9330325  76.81484 57.373938
[40,]   4.7652436  43.82038 24.292814
[41,]  63.9910539 102.93063 83.460844
[42,]  27.2150328  66.07157 46.643302
[43,]  27.7619824  66.62793 47.194958
[44,]  55.2118171  94.08517 74.648492
[45,]  30.3166906  69.21700 49.766845
[46,]  60.6323344  99.54704 80.089688
[47,]  34.5985474  73.41148 54.005013
[48,]  60.9705958  99.83390 80.402246
[49,]  48.1825453  87.00803 67.595286
[50,]  38.1077753  76.94381 57.525791
[51,]  44.4403138  83.34684 63.893578
[52,]  51.8408829  90.69777 71.269328
```

```
 [53,]  54.1186665  92.97005 73.544359
 [54,]  59.0365033  97.95472 78.495613
 [55,]  26.3607096  65.22805 45.794380
 [56,]  37.7783399  76.62714 57.202740
 [57,]  38.5275214  77.40860 57.968059
 [58,]  54.8828203  93.70016 74.291491
 [59,]  56.8806126  95.75935 76.319983
 [60,]  54.7084625  93.82100 74.264729
 [61,]  21.9562767  60.83888 41.397581
 [62,]  22.2116521  61.10517 41.658412
 [63,]  47.6229952  86.49516 67.059079
 [64,]  53.3084519  92.16289 72.735671
 [65,]  42.7593967  81.55638 62.157889
 [66,]  32.6161437  71.48577 52.050958
 [67,]  33.7265997  72.65003 53.188317
 [68,]  28.9645961  67.84394 48.404269
 [69,]  69.4402250 108.39162 88.915922
 [70,]  31.6439930  70.58062 51.112307
 [71,]  35.9377164  75.00167 55.469692
 [72,]  64.5587473 103.43767 83.998209
 [73,]  21.0639707  59.94610 40.505036
 [74,]  27.9677091  66.90769 47.437698
 [75,]  55.1065220  94.12915 74.617834
 [76,]  31.1966369  70.11819 50.657411
 [77,]  62.2281276 101.16716 81.697645
 [78,]  65.6785202 104.68777 85.183146
 [79,]  20.2380722  59.14570 39.691886
 [80,]  30.9968551  69.85215 50.424500
 [81,]  65.8946790 104.79630 85.345491
 [82,]  75.4681840 114.53724 95.002713
 [83,]  38.4146575  77.40826 57.911460
 [84,]  52.3896871  91.26798 71.828835
 [85,]  43.3896275  82.26527 62.827448
 [86,]  61.6614044 100.81662 81.239011
 [87,]  45.0731542  83.94388 64.508515
 [88,]  39.3211621  78.15746 58.739311
 [89,]  49.0997999  88.06707 68.583433
 [90,]  31.6061214  70.43476 51.020442
 [91,]  15.1768339  54.08597 34.631401
 [92,]  -5.9597287  33.19960 13.619935
 [93,]   6.3522667  45.32930 25.840782
 [94,]  44.3049393  83.13505 63.719992
 [95,]  49.0137629  87.82641 68.420087
 [96,]  44.1167658  83.16737 63.642070
 [97,]  24.7903248  64.35719 44.573756
 [98,]  21.2445576  60.17530 40.709931
 [99,]  58.4208381  97.25872 77.839779
[100,]  25.9408202  64.79563 45.368225
[101,]  57.3937707  96.25729 76.825533
[102,]  65.5521144 104.42933 84.990722
[103,]  -6.2499462  32.93957 13.344812
[104,]  25.9293406  64.77539 45.352365
[105,]  28.2334344  67.16444 47.698939
[106,]  50.1040620  89.03476 69.569410
```

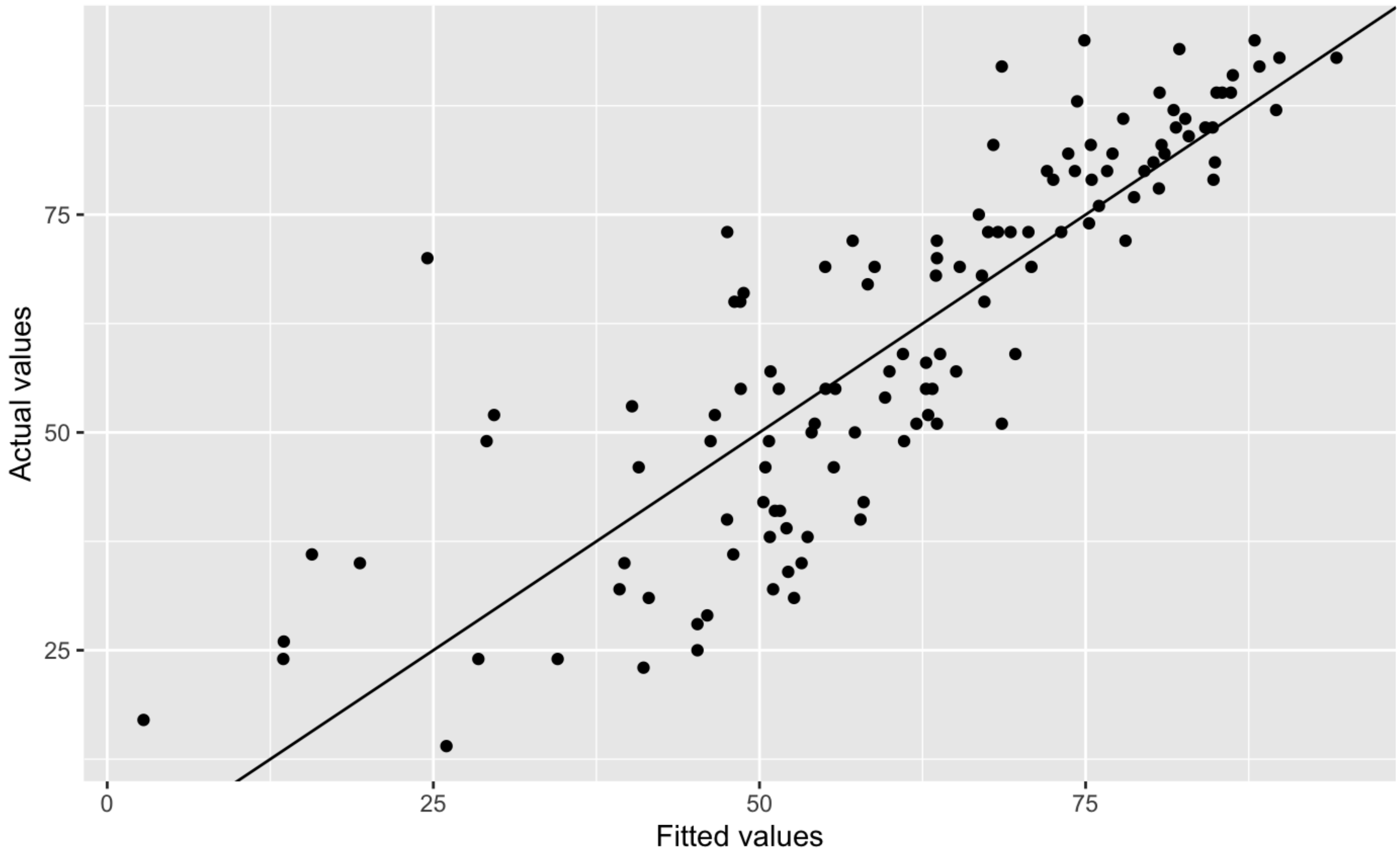
```
[107,]  43.7753354  82.58099 63.178165
[108,]  57.5539083  96.39691 76.975407
[109,]  30.3687767  69.26473 49.816751
[110,]  39.7546542  78.60322 59.178938
[111,]  58.5909785  97.53028 78.060629
[112,]  61.5953687 100.59993 81.097651
[113,]  50.7127701  89.56523 70.139000
[114,]  61.0166631  99.88267 80.449668
[115,]  35.5788812  74.48325 55.031066
[116,]  34.1040076  72.91771 53.510861
[117,]  66.0183271 104.94778 85.483053
[118,]  10.4474883  49.41148 29.929485
[119,]  29.2465302  68.11672 48.681623
[120,]  56.3063587  95.18997 75.748162
[121,]  54.2802216  93.11980 73.700011
[122,]  68.1196362 107.04094 87.580290
[123,]  50.2516571  89.11537 69.683514
attr(,"Probability")
[1] 0.95
attr(,"class")
[1] "confint.bas"
```

The data shows 20 results of our predictions and their 95% credible interval.

Diagnostics

Hide

```
df2 <- as.data.frame(cbind(pred$Ybma,test$audience_score))
colnames(df2) <- c('fit','actual')
ggplot(data = df2, aes(x = df2$fit, y= df2$actual)) +
  geom_point(alpha = 1) +
  geom_abline( slope = 1,intercept =0,)+
  labs(x = "Fitted values", y = "Actual values")
```



Most of our points fall in the diagnal line, meaning the predictions correspond to the actual values closely

We can print out the quantiles of residual errors.

Hide

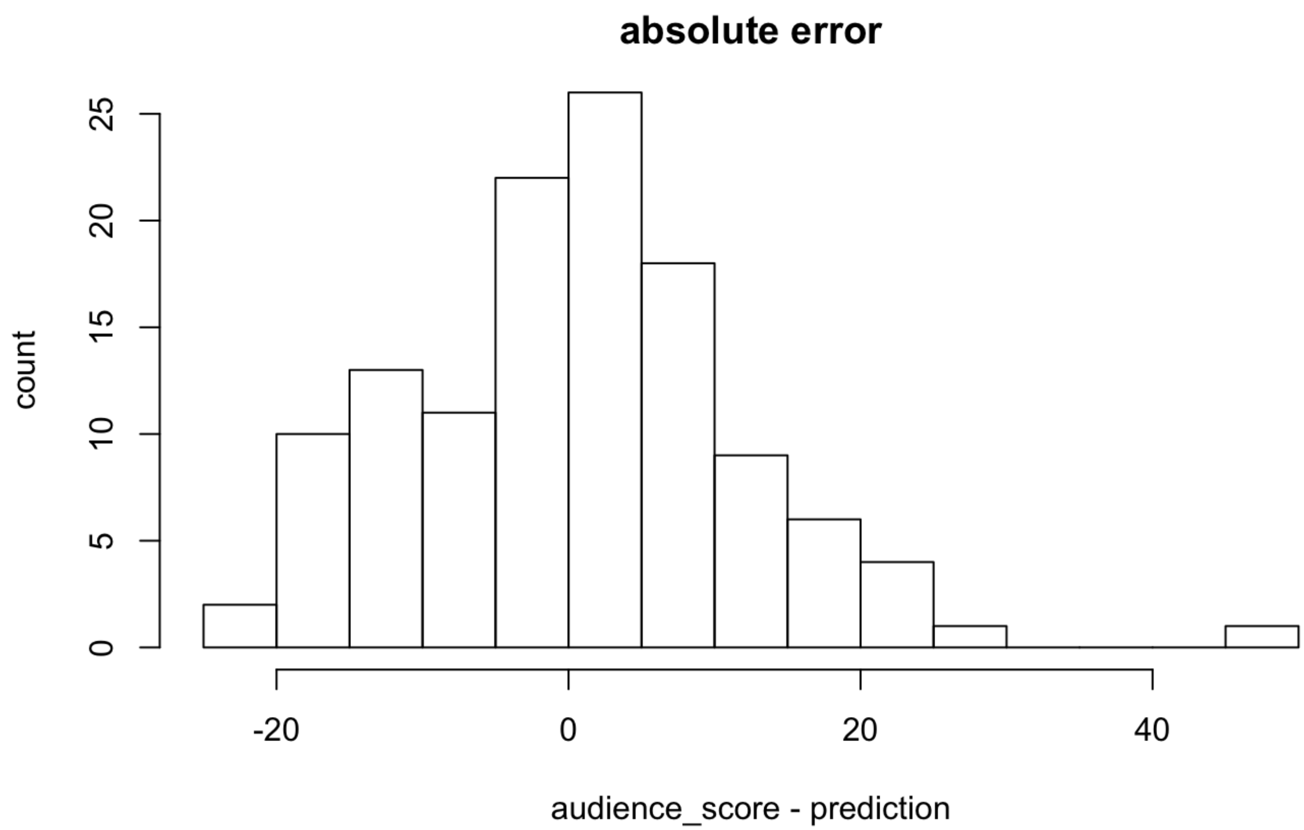
```
print(quantile(df$audience_score - pred$Ybma, probs = c(0,0.25, 0.5, 0.75, 1)))
```

```
        0%         25%         50%         75%        100% 
-21.6482012  -7.4126458   0.8258802   6.4597293  45.4552977 
```

Show the distribution of residual errors in histgrams.

Hide

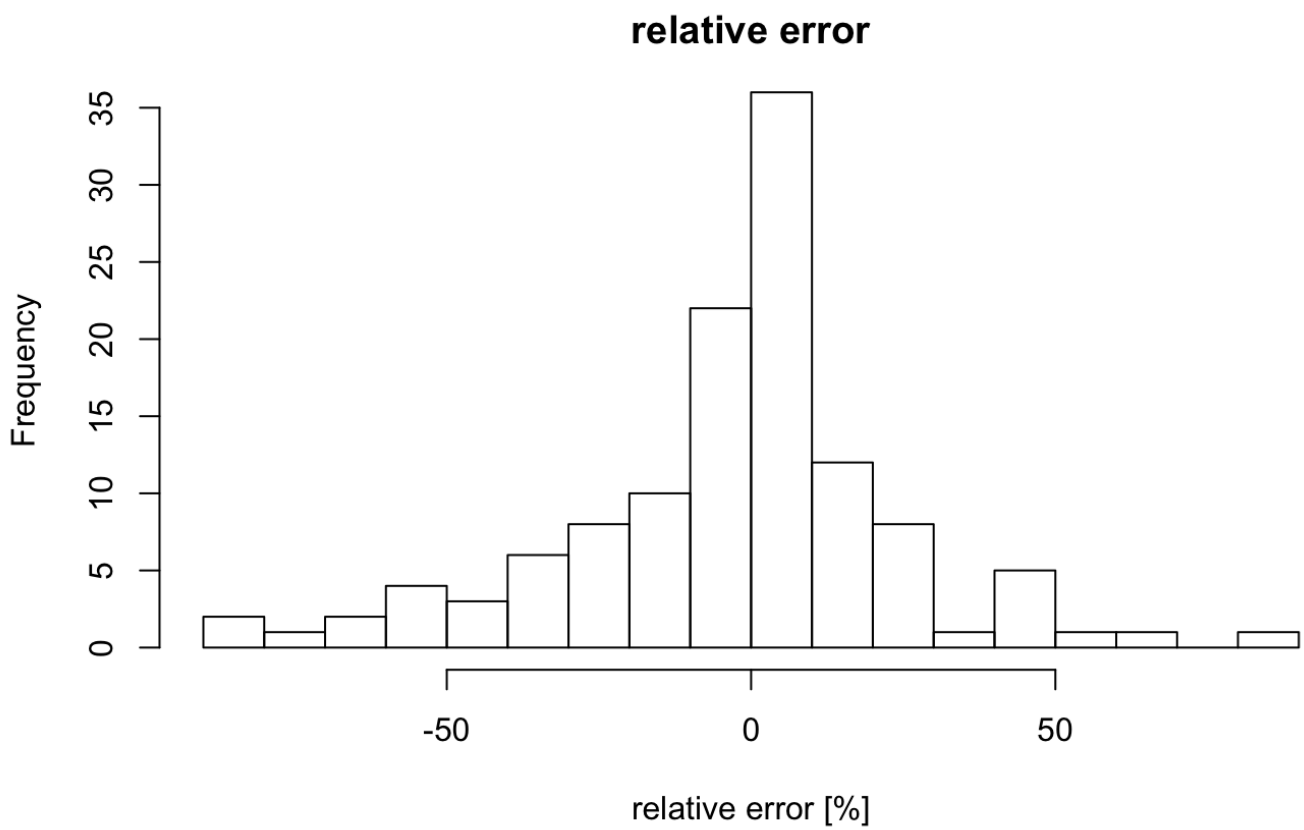
```
hist(df$audience_score - pred$Ybma, breaks=2*floor(sqrt(length(pred$Ybma))), main="absol
ute error", xlab="audience_score - prediction", ylab = "count")
```



Show the distribution of relative residual errors in histgrams.

Hide

```
hist(100 * (df$audience_score - pred$Ybma)/(df$audience_score), breaks=2*floor(sqrt(leng
th(pred$Ybma))), main="relative error", xlab="relative error [%]")
```



Predictions are within the 95% CI

Hide

```
# in_ci: A list contains TURE and FALSE
in_ci = (as.numeric(df$audience_score > df$lower) & as.numeric(df$audience_score < df$up
per))
# n: number of audience_score in credible interval
n = length(in_ci[in_ci])
n
```

```
[1] 114
```

Hide

```
# Percentage of predictions are within the 95% CI
within_interval = 100 *  n/ length(in_ci)

result <- data.frame(Total =nrow(movies[-train_ind, ]), Tests_in_interval=n, tests_withi
n_CI95=within_interval)

result
```

Total <int>	Tests_in_interval <int>	tests_within_CI95 <dbl>
123	114	92.68293

1 row

118 tests are within their 95% CI, the tests_within_prediction_CI rate is 92.7%

Part 6: Conclusion

Discussions

- We implement Bayesian Model Averaging approach with a BIC Prior and an MCMC method to predict audience_score using selected variables. We split 80% of our data set into training and the rest into test.
- The new variables performs well when we evaluate it conditionally. However, our final model does include them.

Limitations

- There are some information that we cannot incorparate into our models. For example, if the starring of the movie has actors with high popularity, the movies' ratings might tend to be higher.
- Although randomness is assumed among variables, colinearty may exist bwtween variables. Maybe some types of movies usually get higher imdb_rating_num because their movie types are popular.

Improvements

- The plots show some outliers, suggesting addtional parameters are required to or they might be simply outliers caused by measurement errors. Either case investigation into the cases is needed.
- To improve the accuracy of our model, we can expect to have more sample sizes to enumerate the entire regression space as much as possible.
- If we can gather information about the sampling methods then we can update our prior. For example, age distribution of our survey groups might be helpful.
- We simply just ommitted missing data during modeling which is bad for model accuracy. We do not know if they are missing at random. Mean substitution, regression imputation can come to resuce sometimes.