Bayesian modeling and prediction for movies

Setup

Load packages

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

Load data

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

Part 1: Data

The data set includes 651 randomly sampled movies produced and released before 2016.

Since the sample is randomly collected we can generalize prediction to new movies. However, there is no evidence that random assignment was used so we can not refer to causuality, for example, one variable causes changes in the other variable(s).

Part 2: Data manipulation

Create feature_film

```
movies<-movies %>%
mutate(feature_film = ifelse(title_type=="Feature Film", "Yes", "No"))
```

Create drama

```
movies<-movies %>%
mutate(drama = ifelse(genre=="Drama", "Yes", "No"))
```

Create mpaa_rating_R

```
movies<-movies %>%
mutate(mpaa_rating_R = ifelse(mpaa_rating=="R", "Yes", "No"))
```

Create oscar_season

```
movies<-movies %>%
mutate(oscar_season= ifelse(thtr_rel_month==11|thtr_rel_month==12|thtr_rel_month==10,
"Yes", "No"))
```

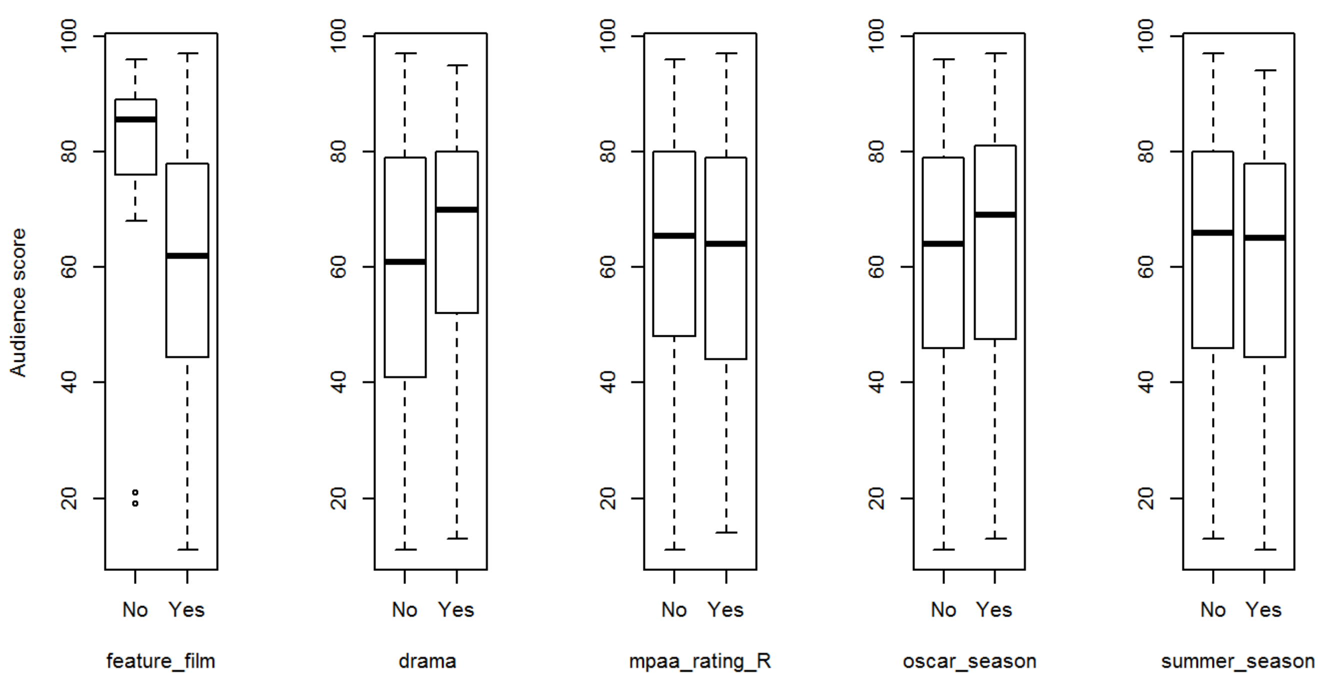
Create summer_season

```
movies<-movies %>%
mutate(summer_season = ifelse(thtr_rel_month==5|thtr_rel_month==6|thtr_rel_month==7|tht
r_rel_month==8, "Yes", "No"))
```

Part 3: Exploratory data analysis

First of all, we use plot box to have a big picture of the realationship between the `audience_score` and new variables constructed.

```
par(mfrow = c(1,5))
boxplot(movies$audience_score~movies$feature_film, ylab="Audience score", xlab="feature
_film")
boxplot(movies$audience_score~movies$drama , xlab="drama ")
boxplot(movies$audience_score~movies$mpaa_rating_R, xlab="mpaa_rating_R")
boxplot(movies$audience_score~movies$oscar_season,xlab="oscar_season")
boxplot(movies$audience_score~movies$summer_season, xlab="summer_season")
```



The first boxplot on the left shows that all else being equal, feature films receive lower meadian score compared to not feature ones. The median scores of feature films is 62 while the median score of not feature one is 85.5

```
movies %>%
group_by(feature_film) %>%
summarise(median=median(audience_score))
```

```
## # A tibble: 2 x 2
##   feature_film median
##          <chr>  <dbl>
## 1           No   85.5
## 2          Yes   62.0
```

The second box plot on the left illustates the drama receive a higher median score compared to not drama ones with all else being equal. The scores are 70 and 61, respectively.

```
movies %>%
group_by(drama) %>%
summarise(median=median(audience_score))
```

```
## # A tibble: 2 x 2
##   drama median
##   <chr>  <dbl>
## 1    No     61
## 2   Yes     70
```

The third box plot shows mppa_rating_R gets lower median audience score but the difference seems to be insignificant. The effect is the same for the last box plot on the right which shows that summer season films get the lower meadian score compared to other season and likewise the difference is not significant.

```
movies %>%
group_by(mpaa_rating_R) %>%
summarise(median=median(audience_score))
```

```
## # A tibble: 2 x 2
##   mpaa_rating_R median
##           <chr>  <dbl>
## 1            No   65.5
## 2           Yes   64.0
```

```
movies %>%
group_by(summer_season) %>%
summarise(median=median(audience_score))
```

```
## # A tibble: 2 x 2
##   summer_season median
##           <chr>  <dbl>
## 1            No     66
## 2           Yes     65
```

And lastly, the oscar season films get highers median scored compared to not oscar ones. The meadian scores of oscar season films is 69 while the out of oscar season movies median score is 64.

```
movies %>%
group_by(oscar_season) %>%
summarise(median=median(audience_score))
```

```
## # A tibble: 2 x 2
##   oscar_season median
##          <chr>  <dbl>
## 1           No     64
## 2          Yes     69
```

Part 4: Modeling

Full model

```
movies_score_full = bas.lm(audience_score ~ feature_film + drama + runtime + mpaa_ratin
g_R + thtr_rel_year + oscar_season + summer_season + imdb_rating + imdb_num_votes + cri
tics_score + best_pic_nom + best_pic_win + best_actor_win + best_actress_win + best_dir
_win + top200_box, prior = "BIC", modelprior = uniform(), data = na.omit(movies))
movies_score_full
```

```
##
## Call:
## bas.lm(formula = audience_score ~ feature_film + drama + runtime +     mpaa_rating_R
 + thtr_rel_year + oscar_season + summer_season +     imdb_rating + imdb_num_votes + cr
itics_score + best_pic_nom +     best_pic_win + best_actor_win + best_actress_win + bes
t_dir_win +     top200_box, data = na.omit(movies), prior = "BIC", modelprior = uniform
())
##
##
##  Marginal Posterior Inclusion Probabilities:
##           Intercept      feature_filmYes            dramaYes
##             1.00000              0.05876             0.04509
##             runtime     mpaa_rating_RYes       thtr_rel_year
##             0.51400              0.16498             0.08089
##      oscar_seasonYes     summer_seasonYes         imdb_rating
##             0.06526              0.07935             1.00000
##       imdb_num_votes        critics_score      best_pic_nomyes
##             0.06242              0.92016             0.13201
##      best_pic_winyes    best_actor_winyes  best_actress_winyes
##             0.04077              0.11565             0.14770
##      best_dir_winyes        top200_boxyes
##             0.06701              0.04876
```

```
summary(movies_score_full)
```

```
##      Intercept feature_filmYes dramaYes runtime mpaa_rating_RYes
## [1,]         1               0        0       1                0
## [2,]         1               0        0       0                0
## [3,]         1               0        0       0                0
## [4,]         1               0        0       1                1
## [5,]         1               0        0       1                0
##      thtr_rel_year oscar_seasonYes summer_seasonYes imdb_rating
## [1,]             0               0                0           1
## [2,]             0               0                0           1
## [3,]             0               0                0           1
## [4,]             0               0                0           1
## [5,]             0               0                0           1
##      imdb_num_votes critics_score best_pic_nomyes best_pic_winyes
## [1,]              0             1               0               0
## [2,]              0             1               0               0
## [3,]              0             1               0               0
## [4,]              0             1               0               0
## [5,]              0             1               1               0
##      best_actor_winyes best_actress_winyes best_dir_winyes top200_boxyes
## [1,]                 0                   0               0             0
## [2,]                 0                   0               0             0
## [3,]                 0                   1               0             0
## [4,]                 0                   0               0             0
## [5,]                 0                   0               0             0
##             BF PostProbs     R2 dim   logmarg
## [1,] 1.0000000    0.1558 0.7483   4 -3434.752
## [2,] 0.8715404    0.1358 0.7455   3 -3434.889
## [3,] 0.2048238    0.0319 0.7470   4 -3436.338
## [4,] 0.2039916    0.0318 0.7496   5 -3436.342
## [5,] 0.1851908    0.0289 0.7495   5 -3436.438
```

From the summary table,we can see that the most likely model with posterior probability 0.1558 includes an intercept, runtime, idmb_rating and critic_score.

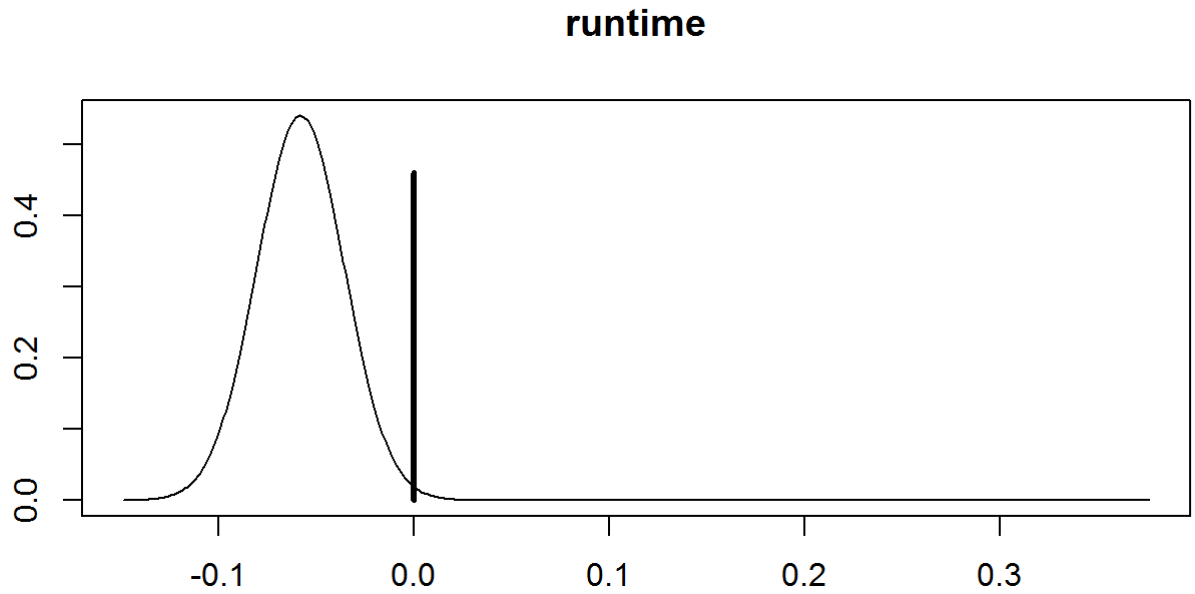
In the next step, the suggested model is executed:

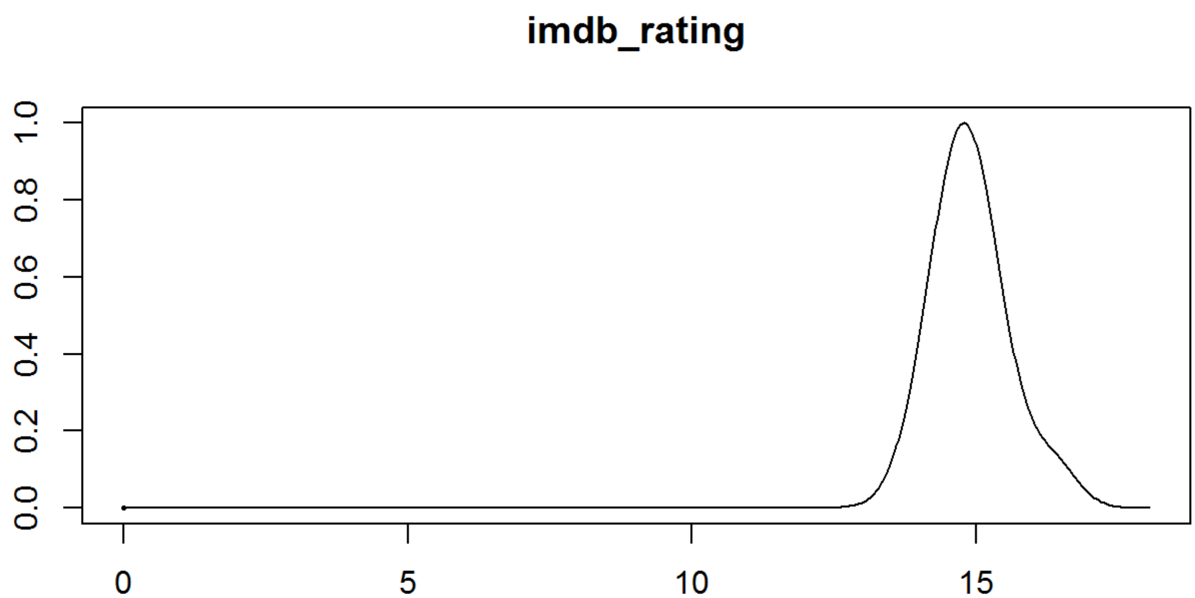
Reduced model

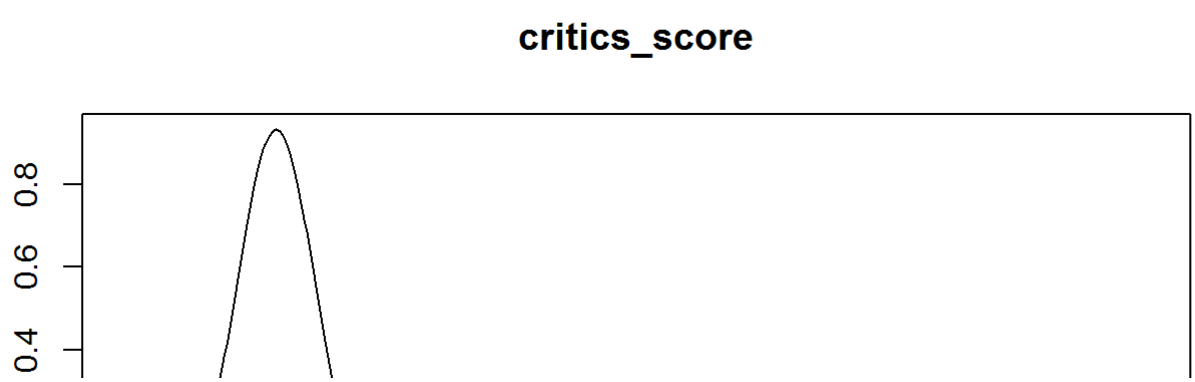
```
movies_score_reduced = bas.lm(audience_score ~ runtime + imdb_rating + critics_score, p
rior = "BIC", modelprior = uniform(), data = na.omit(movies))

coef_movies_score_reduced = coefficients(movies_score_reduced)
confint(coef_movies_score_reduced)
```

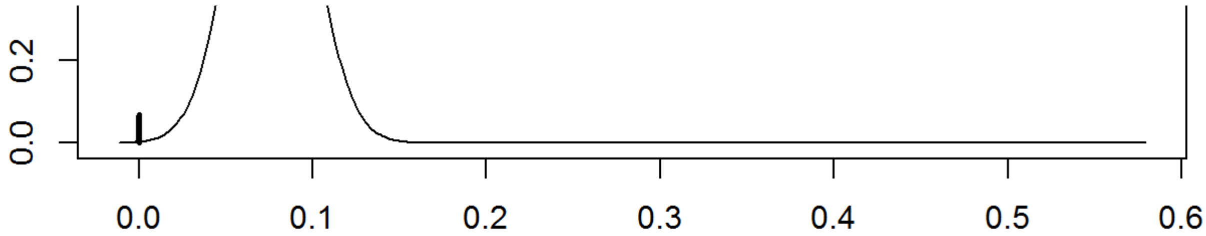
```
##                    2.5  %    97.5  %        beta
## Intercept     61.37098501 62.9899877 62.21001616
## runtime       -0.08894145  0.0000000 -0.03138428
## imdb_rating   13.51235809 16.4793047 14.89079350
## critics_score  0.00000000  0.1129825  0.07128748
## attr(,"Probability")
## [1] 0.95
## attr(,"class")
## [1] "confint.bas"
```

```
plot(coef_movies_score_reduced, subset = c(2, 3, 4), ask=FALSE)
```


runtime
0.0
0.2
0.4
-0.1
0.0
0.1
0.2
0.3


imdb_rating
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15

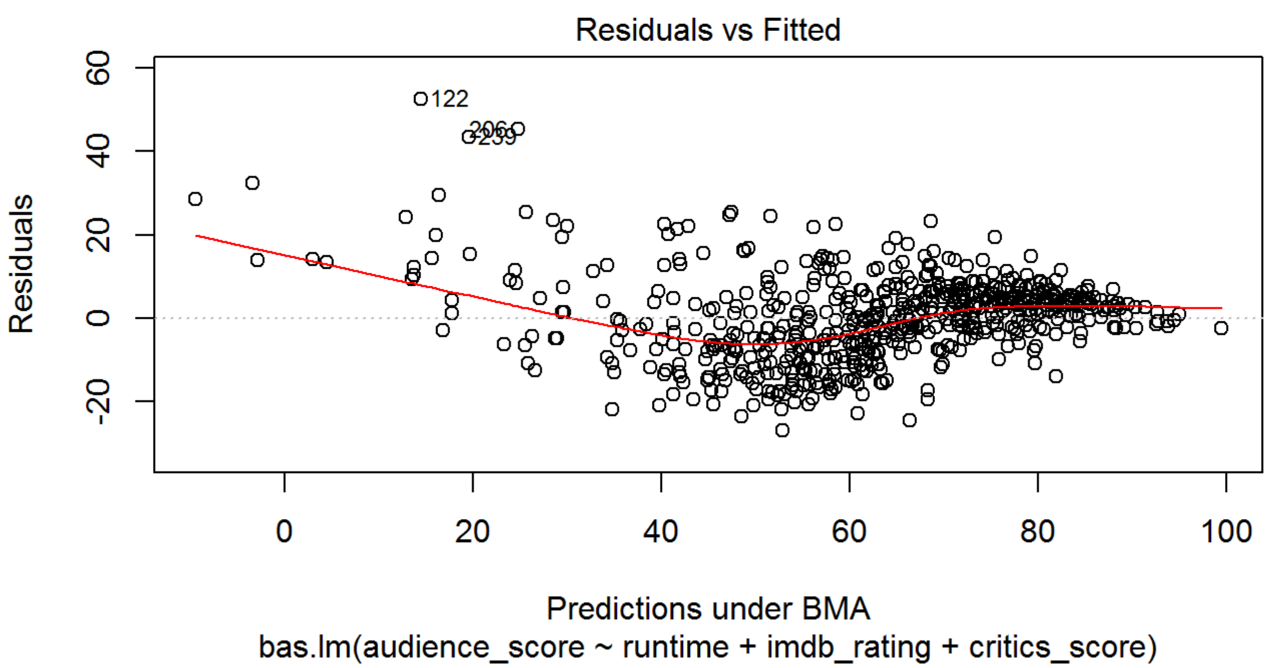

critics_score
0.4
0.6
0.8



From 3 posterior distribution plots above we see that the probability the coefficient runtime =0 given data is about 0.45, probability the coefficient of critic_score =0 given data is about 0.1 and probability of the coefficient of imdb_rating = 0 given data is 0.

Diagnostic test

```
plot(movies_score_reduced, which = 1)
```



From the fitted values vs observed values plot we can see that there might be some residuals that are 122, 206 and 239. We need to carefully deal with these values. In fact, we can check wheather these values stay outside the range of 3 standard deviations from mean or not.

Part 5: Prediction

References: https://www.rottentomatoes.com/m/rogue_one_a_star_wars_story (https://www.rottentomatoes.com/m/rogue_one_a_star_wars_story) http://www.imdb.com/title/tt3748528/?ref_=nv_sr_1 (http://www.imdb.com/title/tt3748528/?ref_=nv_sr_1) Movie: ROGUE ONE: A STAR WARS STORY audience_score = 89, runtime = 133, imdb_rating = 8.2, critics_score = 75

```
movie2016 <- data.frame(audience_score = 89, runtime = 133, imdb_rating = 8.2, critics_
score = 75)
predict(movies_score_reduced, movie2016, estimator = "BMA", se.fit = FALSE)
```

```
## $fit
##          [,1]
## [1,] 88.16072
##
## $Ybma
##          [,1]
## [1,] 88.16072
##
## $Ypred
##          [,1]
## [1,] 87.63197
## [2,] 88.59666
## [3,] 88.84519
## [4,] 89.90819
## [5,] 70.99020
## [6,] 72.65144
## [7,] 67.87944
## [8,] 62.21002
##
## $postprobs
## [1]  4.981641e-01  4.341702e-01  4.149929e-02  2.616645e-02  1.405138e-92
## [6]  6.957269e-93 5.486013e-178 2.963655e-182
##
## $se.fit
## NULL
##
## $se.pred
## NULL
##
## $se.bma.fit
## NULL
##
## $se.bma.pred
## NULL
##
## $df
## [1] 615 616 616 617 617 616 617 618
##
## $best
## [1] 3 2 7 6 8 5 4 1
##
## $bestmodel
## $bestmodel[[1]]
## [1] 0 1 2 3
##
## $bestmodel[[2]]
## [1] 0 2 3
##
## $bestmodel[[3]]
## [1] 0 1 2
##
## $bestmodel[[4]]
## [1] 0 2
```

```
##
## $bestmodel[[5]]
## [1] 0 3
##
## $bestmodel[[6]]
## [1] 0 1 3
##
## $bestmodel[[7]]
## [1] 0 1
##
## $bestmodel[[8]]
## [1] 0
##
##
## $prediction
## [1] FALSE
##
## $estimator
## [1] "BMA"
##
## attr(,"class")
## [1] "pred.bas"
```

The model predicts audience score is 88.16 while the real audience score is 89

Part 6: Conclusion

To sum up, the model with intercept, runtime, imdb_rating and critic score is the most likely one with probability of 15.8 % given uniform prior and the data.

However, we have not completedly dealed with outliers in this study. Specifically, the constant variability of residuals seems to be violated. For future research these outliers should be dealt with more carefully.