MovieLens

Jason Kang

6/1/2019

A. Intro

We will be creating a movie recommendation system using a sample provided by MovieLens (https://grouplens.org/datasets/movielens/10m/ (https://grouplens.org/datasets/movielens/10m/)). Note, this does not contain all movie ratings data but is an accurate reflection of the population consisting of 10,000 movies, 72,000 users and 10 million ratings. We will also be evaluating the RMSE score to understand if the prediction closely reflects the observations.

To begin, we run the script provided by the course which creates test and validation sets of the MovieLens data. Validation set consists of 10% of MovieLens data.

B. Exploring the data

Next, exploratory analysis is conducted to further assess the data

```
sapply(edx, function(x) sum(length(which(is.na(x))))) #There are no NAs found in the data
```

```
##    userId   movieId    rating timestamp     title    genres
##         0         0         0         0         0         0
```

```
n_distinct(edx$movieId) #10,677 films have been rated
```

```
## [1] 10677
```

```
n_distinct(edx$userId)  #69,878 users provided ratings
```

```
## [1] 69878
```

```
n_distinct(edx$genres)  #797 genres
```

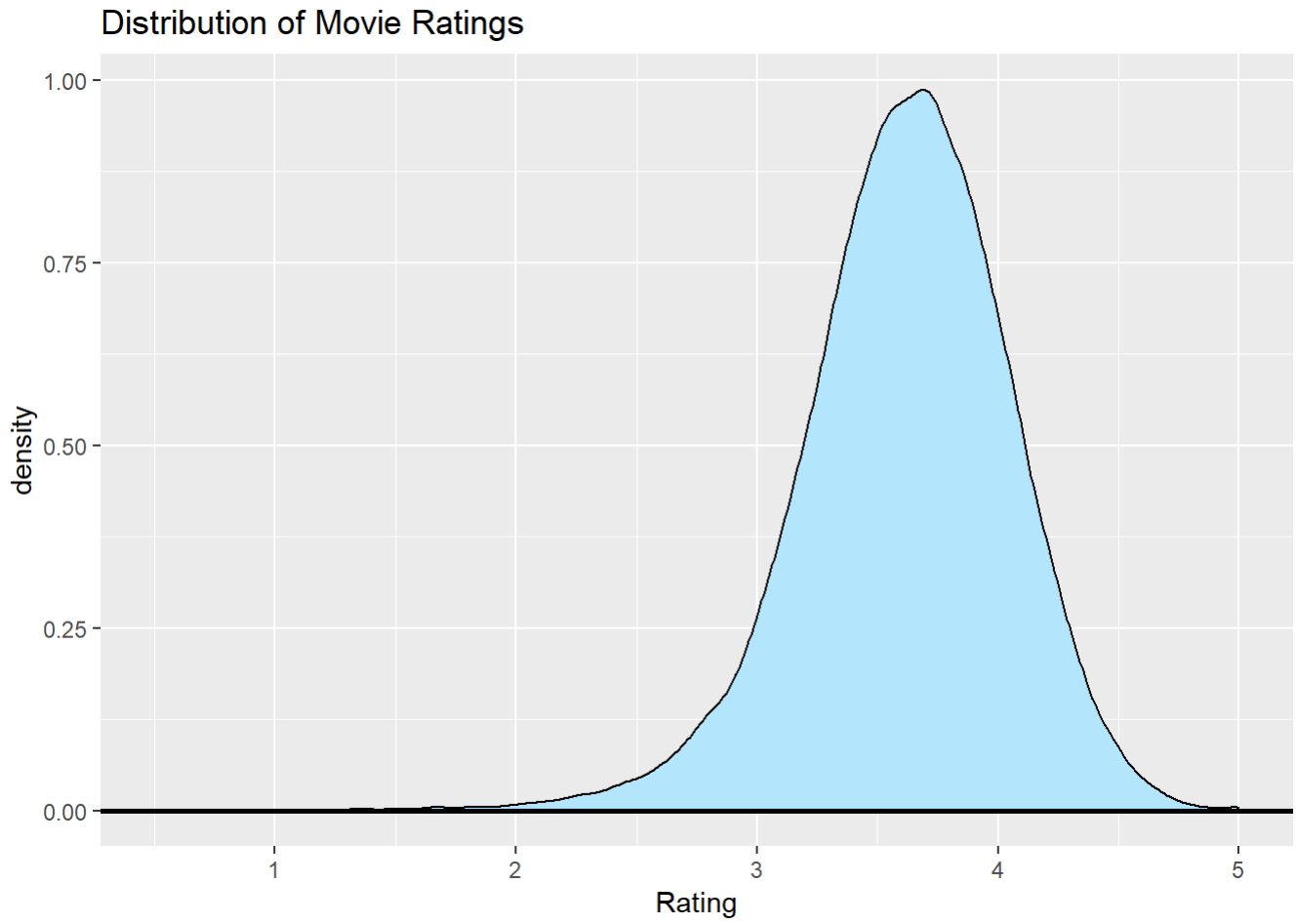
```
## [1] 797
```

```
avg <- mean(edx$rating) #3.5
```

C. Visualizations

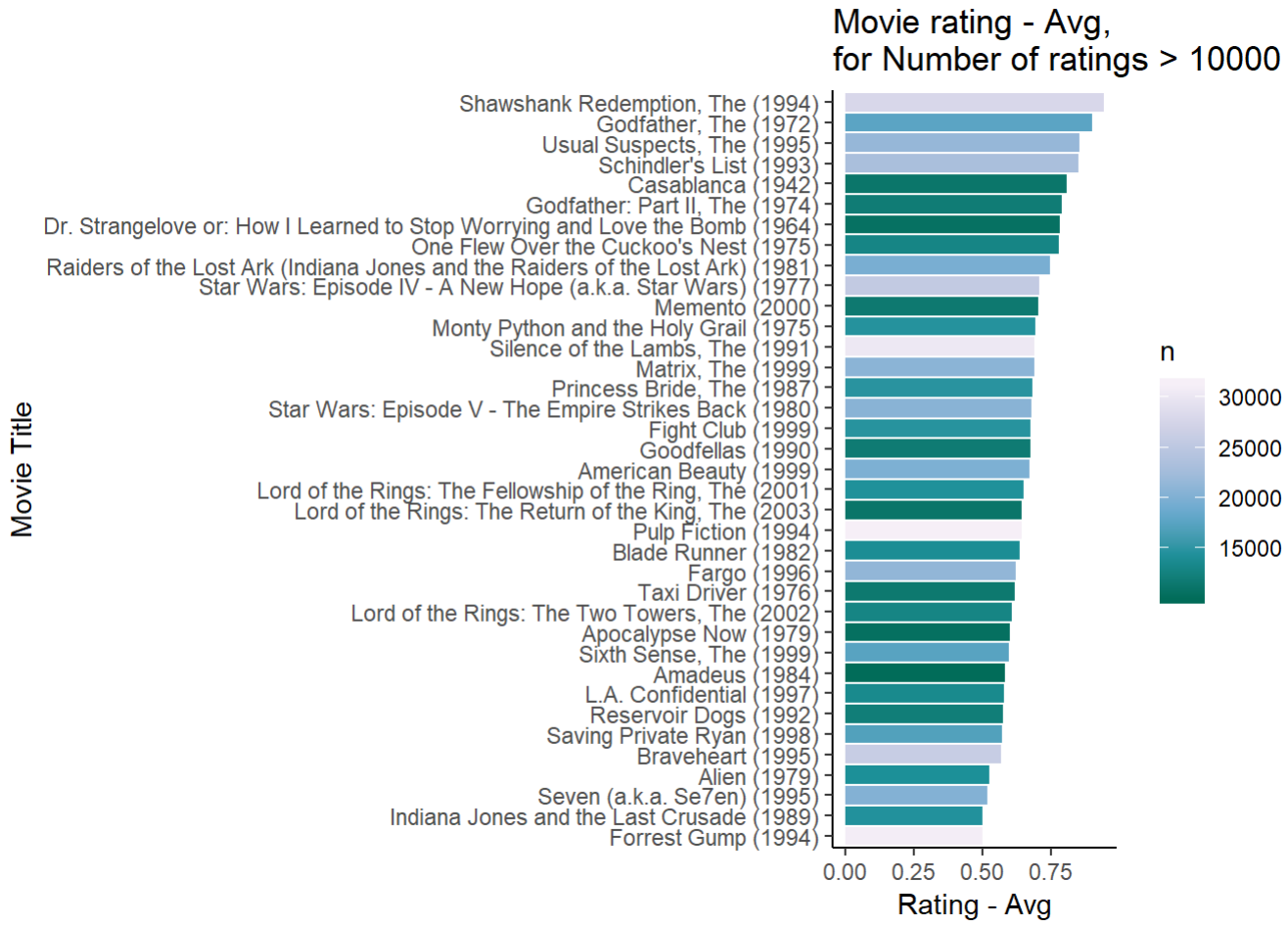
Distributions appear normal at user level with average rating roughly around 3.5

```
edx %>% group_by(userId) %>% summarize(mean_rating = mean(rating)) %>%
  ggplot(aes(mean_rating)) + geom_density(fill="#B3E5FC") +
  geom_hline(yintercept = 0, size = 1) +xlab("Rating") +
  labs(title="Distribution of Movie Ratings")
```



Top movies with large rating volume and above average rating appear to evenly distributed with a variety of genres indicating low user bias in sample

```
edx %>% group_by(title) %>%
  summarize(b_i = mean(rating - avg), n = n()) %>% filter(b_i > 0.5, n > 10000) %>%
  ggplot(aes(reorder(title, b_i), b_i, fill = n)) +
  geom_bar(stat = "identity") + coord_flip() + scale_fill_distiller(palette = "PuBuGn") +
  ggtitle("") + xlab("Movie Title") + ylab("Rating - Avg") +
  ggtitle("Movie rating - Avg,\nfor Number of ratings > 10000") +
  theme_classic()
```



D. Modeling Approach

Here we compute the variability against the mean at the movie level ‘me’ and user level ‘ue’

```
avg<-mean(edx$rating)
me<- edx %>% group_by(movieId) %>% summarize(b_i=mean(rating-avg))
ue<- edx %>% left_join(me, by="movieId") %>% group_by(userId) %>%
  summarize(b_u=mean(rating-b_i-avg))
```

This model foundationally follows the naive approach and predicts on the validation dataset by capturing only the movie effect. In other words, it is calculating the average rating for each movie and the bias for each

```
prediction1 <- validation %>% left_join(me, by = "movieId") %>%
  mutate(pred1 = avg + b_i) %>% .$pred1
```

Additionally this 2nd model predicts on the validation set using both movie and user effect

```
prediction2 <- validation %>% left_join(me, by="movieId") %>%
  left_join(ue, by="userId") %>% mutate(pred2= avg + b_i + b_u) %>%
  .$pred2
```

E. Results

Here’s the root mean square error function which takes the absolute value of the squared distance betwen actual and predicted.

```
rmse <- function(actual, predicted){
  sqrt(mean((actual-predicted)^2))}
```

Now we compute the RMSE for both models

```
rmse(validation$rating, prediction1) #RMSE for model accounting for movie effect only = 0.94
```

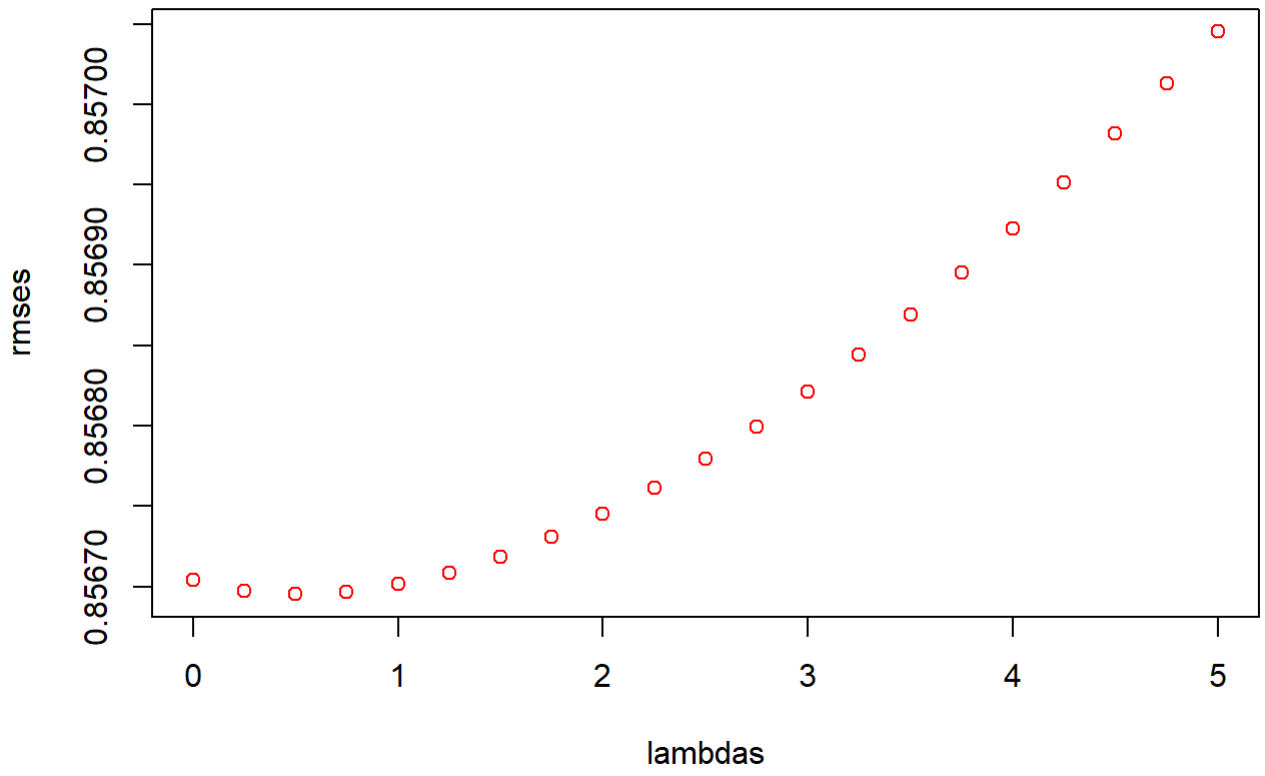
```
## [1] 0.9439087
```

```
rmse(validation$rating, prediction2) #RMSE for model accounting for movie and user effect = 0.86
5
```

```
## [1] 0.8653488
```

Here's with regulatization to improve the precision of the model by constraining total variability of effect sizes. After running several iterations, optimal lambda is set to be .5.

```
lambdas <- seq(0, 5, 0.25)
rmses <- sapply(lambdas,function(l){

  avg<-mean(edx$rating)

  #calculate movie effect and penalize low number of ratings
  b_i <- edx %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - avg)/(n()+l))

  #calculate user and movie effect and penalize low number of ratings
  b_u <- edx %>%
    left_join(b_i, by="movieId") %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - avg)/(n()+l))

  #predict ratings in the training set to derive optimal lambda
  prediction3 <-
    edx %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = avg + b_i + b_u) %>%
    .$pred

  return(rmse(prediction3, edx$rating))
})
plot(lambdas, rmses,
     col = "red")
```



F. Conclusion

It appears in order to predict with accuracy, approaching the model that captures both user and movie effects exceed our target RMSE of less than or equal to 0.8775. Adding regularization did not significantly impact RMSE specifically for the Movie+User effect as the model currently sufficed with a RMSE of 0.865. Next step would be to further improve the model by accounting for bias and variability seen in category which can reduce error as users rating behavior may also differ across categories.