Global Baseline Estimate

Jawaid Hakim

2022-09-17

Contents

1 Global Baseline Estimate

Most recommender systems use personalized algorithms like "content management" and "item-item collaborative filtering." Sometimes non-personalized recommenders are also useful or necessary. One of the best non-personalized recommender system algorithms is the "Global Baseline Estimate," which is demonstrated in the attached Excel spreadsheet, MovieRatings.xlsx. We will also walk through this spreadsheet in our 14-Sep-2022 meetup. Your job here is to use the survey data you collected in your previous assignment, and write the R code that lets you make an item (e.g. movie) recommendation using the Global Baseline Estimate algorithm.

There needs to be at least one respondent with at least two unrated items. You may modify your survey data if needed. Your task is to use the global baseline estimate algorithm to determine ratings for unseen items, then recommend an item to at least one of your survey respondents.

2 Data

Assumes the input file *MovieRatings.xlsx* has been downloaded from github and installed in the current working directory.

Let's look at the data sheets. There are 2 input data sheets: *MovieRatings* contains the sample data provided with the assignment and *MyMovieRatings* contains the actual survey data.

```
readxl::excel_sheets("MovieRatings.xlsx")
```

```
## [1] "MovieRatings"             "Problem Statement"
## [3] "MeanCenteredMovieRatings" "Global Baseline"
## [5] "MyMovieRatings"           "My Global Baseline"
```

Load *MovieRatings* sheet to reproduce model *Global Baseline* provided with the assignment. Otherwise, to reproduce *My Global Baseline*, load *MyMovieRatings*.

Here we load *MyMovieRatings*.

```
ds <- readxl::read_excel("MovieRatings.xlsx", sheet = "MyMovieRatings")
str(ds)
```

```
## tibble [5 x 15] (S3: tbl_df/tbl/data.frame)
##  $ Critic                          : chr [1:5] "Claudia" "Alex" "Igor" "Howard" ...
##  $ TOPGUN:MAVERICK                 : num [1:5] 1 5 5 4 5
##  $ ELVIS                           : num [1:5] 2 3 4 4 4
##  $ EVERYTHING_EVERYWHRE_ALL_AT_ONCE: num [1:5] 5 4 3 5 5
##  $ THE_POWER_OF_THE_DOG            : num [1:5] 5 5 3 5 5
##  $ DON'T_LOOK_TP                   : num [1:5] NA 3 4 4 4
##  $ THE_LOST_DAUGHTER               : num [1:5] 5 5 3 5 4
##  $ HOUSE_OF_GUCCI                  : num [1:5] 4 5 NA 4 4
##  $ THE_INVENTOR_OUT_FOR_BLOOD      : num [1:5] 3 5 5 0 5
##  $ CODA                            : num [1:5] 2 3 3 4 NA
##  $ BOHEMIAN_RHAPSODY               : num [1:5] 4 4 3 2 2
##  $ CRAZY_RICH_ASIANS               : num [1:5] 2 4 5 3 4
##  $ OCEANS_8                        : num [1:5] 3 5 5 4 5
##  $ A_STAR_IS_BORN                  : num [1:5] 3 4 4 NA 4
##  $ A_QUIET_PLACE                   : num [1:5] 1 3 3 4 5
```

Let's remove the *Critic* column from the data frame. Doing so has the advantage that all remaining columns are numeric and it's easier to perform computations on a data frame with only numeric data.

```
num_ds <- ds %>%
    select(-Critic)
num_ds
```

```
## # A tibble: 5 x 14
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1          1     2       5       5      NA       5       4       3     2       4
## 2          5     3       4       5       3       5       5       5     3       4
## 3          5     4       3       3       4       3      NA       5     3       3
## 4          4     4       5       5       4       5       4       0     4       2
## 5          5     4       5       5       4       4       4       5    NA       2
## # ... with 4 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, and abbreviated variable names
## #   1: `TOPGUN:MAVERICK`, 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

```
num_ds <- num_ds %>%
            mutate_if(is.numeric, function(x) ifelse(x == 0, NA, x))
num_ds
```

```
## # A tibble: 5 x 14
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1          1     2       5       5      NA       5       4       3     2       4
## 2          5     3       4       5       3       5       5       5     3       4
## 3          5     4       3       3       4       3      NA       5     3       3
## 4          4     4       5       5       4       5       4      NA     4       2
## 5          5     4       5       5       4       4       4       5    NA       2
## # ... with 4 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, and abbreviated variable names
## #   1: 'TOPGUN:MAVERICK', 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: 'DON'T_LOOK_TP', 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

3 Contruct Global Baseline model

Global Baseline sheets in input XLSX contain the model that is necessary for making predictions. Model *Global Baseline* is for *MovieRatings* data, *My Global Baseline* is for *MyMovieRatings*.

Let's build the *My Global Baseline* model one piece at a time (as Johnny Cash would say!).

3.1 Compute mean movie rating

Now let's compute the mean movie rating. Note, *mean_movie* matches the model value.

```
all_rat <- unlist(num_ds, use.names = FALSE)    # extract all movie ratings
all_rat <- all_rat[!is.na(all_rat)]             # remove NA

sum_rating <- sum(all_rat, na.rm = TRUE)        # sum all ratings
sum_rating
```

```
## [1] 252
```

```
count_rating <- length(all_rat)                 # count number of ratings
count_rating
```

```
## [1] 65
```

```
mean_movie <- round(sum_rating / count_rating, 2) # compute mean movie rating
mean_movie
```

```
## [1] 3.88
```

3.2 Compute movie averages

Compute (movie) averages of all numeric columns. Note, move averages match the model.

```
movie_average <- num_ds %>%
                    summarise_if(is.numeric, mean, na.rm = TRUE)
movie_average
```

```
## # A tibble: 1 x 14
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1          4   3.4     4.4     4.6    3.75     4.4    4.25     4.5     3       3
## # ... with 4 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, and abbreviated variable names
## #   1: `TOPGUN:MAVERICK`, 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

Add 2 rows at the end of data frame: 1st row contains movie averages, 2nd row contains the *movie average - mean movie*. Note, values of both rows matches the model.

```
num_ds <- add_row(num_ds, movie_average)
num_ds <- add_row(num_ds, round(movie_average - mean_movie, 2))
tail(num_ds)
```

```
## # A tibble: 6 x 14
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1       5     3       4       5       3       5       5       5     3       4
## 2       5     4       3       3       4       3      NA       5     3       3
## 3       4     4       5       5       4       5       4      NA     4       2
## 4       5     4       5       5       4       4       4       5    NA       2
## 5       4     3.4     4.4     4.6     3.75    4.4     4.25    4.5   3       3
## 6       0.12 -0.48    0.52    0.72   -0.13    0.52    0.37    0.62 -0.88   -0.88
## # ... with 4 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, and abbreviated variable names
## #   1: `TOPGUN:MAVERICK`, 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

3.3 User Average

Compute and add *user average* as column to data frame.

```
num_ds <- num_ds %>%
            mutate('user avg' = rowMeans(., na.rm = TRUE))

num_ds <- num_ds %>%
            mutate('user avg' = round(num_ds$`user avg`, 2))

head(num_ds, n = 10)
```

```
## # A tibble: 7 x 15
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1       1     2       5       5      NA       5       4       3     2       4
## 2       5     3       4       5       3       5       5       5     3       4
## 3       5     4       3       3       4       3      NA       5     3       3
## 4       4     4       5       5       4       5       4      NA     4       2
## 5       5     4       5       5       4       4       4       5    NA       2
## 6       4     3.4     4.4     4.6     3.75    4.4     4.25    4.5   3       3
## 7       0.12 -0.48    0.52    0.72   -0.13    0.52    0.37    0.62 -0.88   -0.88
## # ... with 5 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, `user avg` <dbl>, and
## #   abbreviated variable names 1: `TOPGUN:MAVERICK`,
## #   2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE, 3: THE_POWER_OF_THE_DOG,
## #   4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER, 6: HOUSE_OF_GUCCI,
## #   7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

Compute and add *user average - mean movie* as column to data frame.

```
num_ds <- num_ds %>%
            mutate("user avg - mean movie" = round(num_ds$`user avg` - mean_movie, 2))
head(num_ds)
```

```
## # A tibble: 6 x 16
##   TOPGUN:M~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA BOHEM~8
##        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
## 1          1   2       5       5      NA       5       4       3       2       4
## 2          5   3       4       5       3       5       5       5       3       4
## 3          5   4       3       3       4       3      NA       5       3       3
## 4          4   4       5       5       4       5       4      NA       4       2
## 5          5   4       5       5       4       4       4       5      NA       2
## 6          4   3.4     4.4     4.6     3.75    4.4     4.25    4.5     3       3
## # ... with 6 more variables: CRAZY_RICH_ASIANS <dbl>, OCEANS_8 <dbl>,
## #   A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>, `user avg` <dbl>,
## #   `user avg - mean movie` <dbl>, and abbreviated variable names
## #   1: `TOPGUN:MAVERICK`, 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD, 8: BOHEMIAN_RHAPSODY
```

Extract *critics* and *movies* from original data source.

```
critics <- unlist(ds[, 'Critic'], use.names = FALSE)
critics
```

```
## [1] "Claudia" "Alex"    "Igor"    "Howard"  "Monish"
```

```
movies <- colnames(ds)
movies <- movies[-1]    # remove Critics from movie names
movies
```

```
##  [1] "TOPGUN:MAVERICK"                   "ELVIS"
##  [3] "EVERYTHING_EVERYWHRE_ALL_AT_ONCE" "THE_POWER_OF_THE_DOG"
```

```
##  [5] "DON'T_LOOK_TP"                    "THE_LOST_DAUGHTER"
##  [7] "HOUSE_OF_GUCCI"                   "THE_INVENTOR_OUT_FOR_BLOOD"
##  [9] "CODA"                             "BOHEMIAN_RHAPSODY"
## [11] "CRAZY_RICH_ASIANS"                "OCEANS_8"
## [13] "A_STAR_IS_BORN"                   "A_QUIET_PLACE"
```

Add *movie avg* and *movie avg - mean movie* as rows to critics.

```
critics_expanded <- append(critics, 'movie avg')

critics_expanded <- append(critics_expanded, 'movie avg - mean movie')

critics_expanded
```

```
## [1] "Claudia"                "Alex"                   "Igor"
## [4] "Howard"                 "Monish"                 "movie avg"
## [7] "movie avg - mean movie"
```

Add expanded *critics* columns back into data source. This data frame should be identical in all important aspects to the *My Global Baseline* sheet in input XSLX.

```
global_baseline <- add_column(num_ds, 'Critic' = critics_expanded, .before = 0)
global_baseline
```

```
## # A tibble: 7 x 17
##   Critic     TOPGU~1 ELVIS EVERY~2 THE_P~3 DON'T~4 THE_L~5 HOUSE~6 THE_I~7  CODA
##   <chr>        <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>
## 1 Claudia       1     2       5       5      NA       5       4       3     2
## 2 Alex          5     3       4       5       3       5       5       5     3
## 3 Igor          5     4       3       3       4       3      NA       5     3
## 4 Howard        4     4       5       5       4       5       4      NA     4
## 5 Monish        5     4       5       5       4       4       4       5    NA
## 6 movie avg     4     3.4     4.4     4.6     3.75    4.4     4.25    4.5   3
## 7 movie avg~    0.12 -0.48    0.52    0.72   -0.13    0.52    0.37    0.62 -0.88
## # ... with 7 more variables: BOHEMIAN_RHAPSODY <dbl>, CRAZY_RICH_ASIANS <dbl>,
## #   OCEANS_8 <dbl>, A_STAR_IS_BORN <dbl>, A_QUIET_PLACE <dbl>,
## #   `user avg` <dbl>, `user avg - mean movie` <dbl>, and abbreviated variable
## #   names 1: `TOPGUN:MAVERICK`, 2: EVERYTHING_EVERYWHRE_ALL_AT_ONCE,
## #   3: THE_POWER_OF_THE_DOG, 4: `DON'T_LOOK_TP`, 5: THE_LOST_DAUGHTER,
## #   6: HOUSE_OF_GUCCI, 7: THE_INVENTOR_OUT_FOR_BLOOD
```

4 Predictions

Now that we have our Global Baseline model, let's define a function that predicts the movie rating for any pair of movie and critic.

```
me.predictRating <- function(movie, critic) {
            if (! movie %in% colnames(global_baseline)) {
                print(paste("Error - invalid movie:", movie))
                NA
```

```
            }
            else if (! critic %in% critics) {
                print(paste("Error -  critic:", critic))
                NA
            }
            else {
                mean_movie +
                global_baseline[global_baseline$Critic == 'movie avg - mean movie', movie][[1]] +
                global_baseline[global_baseline$Critic == critic, 'user avg - mean movie'][[1]]
            }
}
```

4.1 Test predictions

Sanity check a few predictions. Note, movie prediction for *Param: PitchPerfect2* matches the prediction in the *Global Baseline* sheet in input XSLX.

```
test_data <- c('HOUSE_OF_GUCCI', 'Igor', 'HOUSE_OF_GUCCI', 'Howard', 'CODA', 'Monish', 'OCEANS_8', 'Clau
#test_data <- c('PitchPerfect2', 'Param', 'CaptainAmerica', 'Burton', 'JungleBook', 'Vuthy', 'Frozen',

for (i in seq(1, length(test_data), 2)) {
                        if (! test_data[i] %in% colnames(global_baseline)) {
                            print(paste("*Error - invalid movie:", test_data[i]))
                        }
                        else if (! test_data[i + 1] %in% critics) {
                            print(paste("*Error - invalid critic:", test_data[i + 1]))
                        }
                        else {
                            print(
                                paste(
                                "Rating for movie", test_data[i], "by critic", test_data[i + 1], "- Pred
                                me.predictRating(test_data[i], test_data[i + 1]),
                                ", Actual:",
                                global_baseline[global_baseline$Critic == test_data[i + 1], test_data[i]
                        }
}
```

```
## [1] "Rating for movie HOUSE_OF_GUCCI by critic Igor - Predicted: 4.22 , Actual: NA"
## [1] "Rating for movie HOUSE_OF_GUCCI by critic Howard - Predicted: 4.37 , Actual: 4"
## [1] "Rating for movie CODA by critic Monish - Predicted: 3.43 , Actual: NA"
## [1] "Rating for movie OCEANS_8 by critic Claudia - Predicted: 3.6 , Actual: 3"
```

4.2 Predict rating for all unrated movies

Make rating predictions for all unrated movies.

```
c <- as.character(c()) # critics
m <- as.character(c()) # movies
r <- as.numeric(c())   # predictions

for (critic in critics) {
```

```
    for (movie in movies) {
        if (is.na(global_baseline[global_baseline$Critic == critic, movie][[1]])) {
            m <- append(m, movie)
            c <- append(c, critic)
            r <- append(r, me.predictRating(movie, critic))
        }
    }
}
predictions <- data.frame('Critic' = c, 'Movie' = m, 'Predicted Rating' = r)
predictions <- predictions %>%
    group_by(Critic)
```

4.3 Output movie predictions

Display all predictions. Note, OUSE_OF_GUCCI rating prediction for Igor, 4.22, matches the prediction in model.

```
print(predictions, n = nrow(predictions))
```

```
## # A tibble: 5 x 3
## # Groups:   Critic [4]
##   Critic  Movie                       Predicted.Rating
##   <chr>   <chr>                                  <dbl>
## 1 Claudia DON'T_LOOK_TP                           2.95
## 2 Igor    HOUSE_OF_GUCCI                          4.22
## 3 Howard  THE_INVENTOR_OUT_FOR_BLOOD              4.62
## 4 Howard  A_STAR_IS_BORN                          3.87
## 5 Monish  CODA                                    3.43
```