

Global Baseline Estimate

Jawaid Hakim

2022-09-17

Contents

1	Global Baseline Estimate	1
2	Data	1
3	Construct Global Baseline model	2
3.1	Compute mean movie rating	3
3.2	Compute movie averages	3
3.3	User Average	4
4	Predictions	5
4.1	Test predictions	6
4.2	Predict rating for all unrated movies	6
4.3	Output movie predictions	7

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 XLSX has been installed in the current working directory.

Let’s find out names of sheets in the XLSX file. The sheet we want to load is *MovieRatings*.

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

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

Let's load *MovieRatings* sheet from XLSX file.

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

```
## tibble [16 x 7] (S3: tbl_df/tbl/data.frame)
## $ Critic      : chr [1:16] "Burton" "Charley" "Dan" "Dieudonne" ...
## $ CaptainAmerica: num [1:16] NA 4 NA 5 4 4 4 NA 4 4 ...
## $ Deadpool     : num [1:16] NA 5 5 4 NA NA 4 NA 4 3 ...
## $ Frozen       : num [1:16] NA 4 NA NA 2 3 4 NA 1 5 ...
## $ JungleBook   : num [1:16] 4 3 NA NA NA 3 2 NA NA 5 ...
## $ PitchPerfect2 : num [1:16] NA 2 NA NA 2 4 2 NA NA 2 ...
## $ StarWarsForce : num [1:16] 4 3 5 5 5 NA 4 4 5 3 ...
```

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: 16 x 6
##   CaptainAmerica Deadpool Frozen JungleBook PitchPerfect2 StarWarsForce
##   <dbl>      <dbl>   <dbl>      <dbl>      <dbl>      <dbl>
## 1         NA         NA     NA         4         NA         4
## 2         4         5     4         3         2         3
## 3         NA         5     NA        NA        NA         5
## 4         5         4     NA        NA        NA         5
## 5         4         NA     2        NA         2         5
## 6         4         NA     3         3         4        NA
## 7         4         4     4         2         2         4
## 8         NA         NA     NA        NA        NA         4
## 9         4         4     1        NA        NA         5
## 10        4         3     5         5         2         3
## 11        5         5     5         5        NA         4
## 12        NA         NA     4         5        NA         3
## 13        5         5     5         4         4         5
## 14        4         NA     NA        NA        NA         4
## 15        4         5     3         3         3        NA
## 16        NA         NA     5         5        NA        NA
```

3 Contruct Global Baseline model

The Global Baseline sheet in input XLSX shows the model that is necessary for make predictions. Let's build this 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. Sum of all ratings is 240 and count of valid ratings (excluding NA) is 61. Mean of all movie ratings is 3.93.

```
all_rat <- unlist(num_ds, use.names = FALSE) # extract all movie ratings

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

```
## [1] 240
```

```
count_rating <- length(na.exclude(all_rat))  # count number of ratings, excluding NA
count_rating
```

```
## [1] 61
```

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

```
## [1] 3.93
```

3.2 Compute movie averages

Compute (movie) averages of all numeric columns.

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

```
## # A tibble: 1 x 6
##   CaptainAmerica Deadpool Frozen JungleBook PitchPerfect2 StarWarsForce
##   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1      4.27      4.44      3.73      3.9      2.71      4.15
```

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

```
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 6
##   CaptainAmerica Deadpool Frozen JungleBook PitchPerfect2 StarWarsForce
##   <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1      5      5      5      4      4      5
## 2      4     NA     NA     NA     NA      4
## 3      4      5      3      3      3     NA
## 4     NA     NA      5      5     NA     NA
## 5      4.27      4.44      3.73      3.9      2.71      4.15
## 6      0.34      0.51     -0.2     -0.03     -1.22      0.22
```

3.3 User Average

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

```
num_ds <- num_ds %>%
  mutate(UserAvg = rowMeans(., na.rm = TRUE)) %>%
  mutate(UserAvg = round(UserAvg, 1))
head(num_ds)
```

```
## # A tibble: 6 x 7
##   CaptainAmerica Deadpool Frozen JungleBook PitchPerfect2 StarWarsForce UserAvg
##   <dbl>      <dbl>   <dbl>      <dbl>      <dbl>      <dbl>   <dbl>
## 1          NA         NA     NA          4          NA         4       4
## 2           4         5     4          3          2         3     3.5
## 3          NA         5     NA          NA         NA         5       5
## 4           5         4     NA          NA         NA         5     4.7
## 5           4         NA     2          NA         2         5     3.2
## 6           4         NA     3          3          4         NA     3.5
```

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

```
num_ds <- num_ds %>%
  mutate("UserAvgMinusMeanMovie" = round(UserAvg - mean_movie, 2))
head(num_ds)
```

```
## # A tibble: 6 x 8
##   CaptainAmerica Deadpool Frozen JungleBook PitchPerfe-1 StarW-2 UserAvg UserA-3
##   <dbl>      <dbl>   <dbl>      <dbl>      <dbl>      <dbl>   <dbl>   <dbl>
## 1          NA         NA     NA          4          NA         4       4       0.07
## 2           4         5     4          3          2         3     3.5    -0.43
## 3          NA         5     NA          NA         NA         5       5       1.07
## 4           5         4     NA          NA         NA         5     4.7       0.77
## 5           4         NA     2          NA         2         5     3.2    -0.73
## 6           4         NA     3          3          4         NA     3.5    -0.43
## # ... with abbreviated variable names 1: PitchPerfect2, 2: StarWarsForce,
## #   3: UserAvgMinusMeanMovie
```

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

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

```
## [1] "Burton"    "Charley"   "Dan"       "Dieudonne" "Matt"      "Mauricio"
## [7] "Max"       "Nathan"    "Param"     "Parshu"    "Prashanth" "Shipra"
## [13] "Sreejaya" "Steve"     "Vuthy"     "Xingjia"
```

```
movies <- colnames(ds)
movies <- movies[-1]
movies
```

```
## [1] "CaptainAmerica" "Deadpool"      "Frozen"        "JungleBook"
## [5] "PitchPerfect2"  "StarWarsForce"
```

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] "Burton"          "Charley"          "Dan"
## [4] "Dieudonne"       "Matt"             "Mauricio"
## [7] "Max"             "Nathan"           "Param"
## [10] "Parshu"          "Prashanth"        "Shipra"
## [13] "Sreejaya"        "Steve"             "Vuthy"
## [16] "Xingjia"         "movie avg"        "movie avg - mean movie"
```

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

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

```
## # A tibble: 18 x 9
##   Critic      Capta~1 Deadp~2 Frozen Jungl~3 Pitch~4 StarW~5 UserAvg UserA~6
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 Burton      NA      NA      NA      4      NA      4      4      0.07
## 2 Charley      4      5      4      3      2      3     3.5    -0.43
## 3 Dan         NA      5      NA      NA      NA      5      5      1.07
## 4 Dieudonne    5      4      NA      NA      NA      5     4.7     0.77
## 5 Matt         4      NA      2      NA      2      5     3.2    -0.73
## 6 Mauricio     4      NA      3      3      4     NA     3.5    -0.43
## 7 Max          4      4      4      2      2      4     3.3    -0.63
## 8 Nathan      NA      NA      NA      NA      NA      4      4      0.07
## 9 Param        4      4      1      NA      NA      5     3.5    -0.43
## 10 Parshu       4      3      5      5      2      3     3.7    -0.23
## 11 Prashanth    5      5      5      5      NA      4     4.8     0.87
## 12 Shipra      NA      NA      4      5      NA      3      4      0.07
## 13 Sreejaya     5      5      5      4      4      5     4.7     0.77
## 14 Steve        4      NA      NA      NA      NA      4      4      0.07
## 15 Vuthy        4      5      3      3      3     NA     3.6    -0.33
## 16 Xingjia      NA      NA      5      5      NA     NA      5      1.07
## 17 movie avg    4.27    4.44    3.73    3.9     2.71    4.15    3.9    -0.03
## 18 movie avg - m~ 0.34    0.51   -0.2    -0.03   -1.22    0.22   -0.1    -4.03
## # ... with abbreviated variable names 1: CaptainAmerica, 2: Deadpool,
## # 3: JungleBook, 4: PitchPerfect2, 5: StarWarsForce, 6: UserAvgMinusMeanMovie
```

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, 'UserAvgMinusMeanMovie'][[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('PitchPerfect2', 'Param', 'CaptainAmerica', 'Burton', 'JungleBook', 'Vuthy', 'Frozen', 'Steve')

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], "- Predicted: ",
        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 PitchPerfect2 by critic Param - Predicted: 2.28 , Actual: NA"
## [1] "Rating for movie CaptainAmerica by critic Burton - Predicted: 4.34 , Actual: NA"
## [1] "Rating for movie JungleBook by critic Vuthy - Predicted: 3.57 , Actual: 3"
## [1] "Rating for movie Frozen by critic Steve - Predicted: 3.8 , Actual: NA"

```

4.2 Predict rating for all unrated movies

Let's 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.

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

```

## # A tibble: 35 x 3
## # Groups:   Critic [12]
##   Critic      Movie      Predicted.Rating
##   <chr>    <chr>          <dbl>
## 1 Burton  CaptainAmerica    4.34
## 2 Burton  Deadpool          4.51
## 3 Burton  Frozen            3.8
## 4 Burton  PitchPerfect2     2.78
## 5 Dan     CaptainAmerica    5.34
## 6 Dan     Frozen            4.8
## 7 Dan     JungleBook        4.97
## 8 Dan     PitchPerfect2     3.78
## 9 Dieudonne Frozen            4.5
## 10 Dieudonne JungleBook    4.67
## 11 Dieudonne PitchPerfect2    3.48
## 12 Matt    Deadpool          3.71
## 13 Matt    JungleBook        3.17
## 14 Mauricio Deadpool          4.01
## 15 Mauricio StarWarsForce    3.72
## 16 Nathan  CaptainAmerica    4.34
## 17 Nathan  Deadpool          4.51
## 18 Nathan  Frozen            3.8
## 19 Nathan  JungleBook        3.97
## 20 Nathan  PitchPerfect2     2.78
## 21 Param   JungleBook        3.47
## 22 Param   PitchPerfect2     2.28
## 23 Prashanth PitchPerfect2     3.58
## 24 Shipra  CaptainAmerica    4.34
## 25 Shipra  Deadpool          4.51
## 26 Shipra  PitchPerfect2     2.78
## 27 Steve   Deadpool          4.51

```

## 28	Steve	Frozen	3.8
## 29	Steve	JungleBook	3.97
## 30	Steve	PitchPerfect2	2.78
## 31	Vuthy	StarWarsForce	3.82
## 32	Xingjia	CaptainAmerica	5.34
## 33	Xingjia	Deadpool	5.51
## 34	Xingjia	PitchPerfect2	3.78
## 35	Xingjia	StarWarsForce	5.22