Movie Recommendation System

HardvardX Professional Certificate in Data Science - Capstone Project 1

Joao Rodrigues

21 May 2019

1. Executive summary

This report documents the analytical approach, modelling results and proposed next steps for the HarvardX Data Science Capstone Project, to build a recommendation system based upon the Movielens 10M data. In terms of approach, limited data pre-processing was required, exploratory data analysis dimensioned the dataset and tested for select correlations to movie ratings, and various predictive models were then constructed and tuned. In the final model recommendation model, a regularized movie and user effects model yielded an RMSE of 0.86511, below the target RMSE of 0.87750 for this assignment. The computational effort of applying alternative prediction approaches, (e.g. knn, random forests), ruled these out as feasible approaches, although these were tested on a much smaller subset of the Movielens dataset.

2. Objectives of this analysis

This project sets out to demonstrate an appropriate approach to data exploration, and application of select machine learning algorithms to predict movie ratings from the Movielens data set. Specifically, this project demonstrates:

- A few basic checks in the data pre-processing phase;
- Exploratory data analysis to dimension the dataset and test for correlations to ratings;
- The construction and testing of models to target a RMSE lower than 0.87750.
  - Prediction models were evaluated using the residual mean squared error (RMSE) measure, defined as:
    RMSE $= \sqrt{\frac{1}{N}\sum_{u,i}\left(\hat{y}_{u,i} - y_{u,i}\right)^2}$.
  - The Netflix grand prize winners achieved a final RMSE of approximately 0.857, and the grading rubric for this capstone targets an RMSE <= 0.87750 for full points. This informs the target range to evaluate the recommendation algorithms that follow.

3. Methodology

3.1. Preprocessing

- Tests for complete cases in the `edx` dataset revealed no partial data entries, i.e. no null data in any of the data entries. This is tested with the following code chunk:

```
sum(complete.cases(edx)) == nrow(edx)
```

- I hypothesise that the number of ratings that each movie received may be a predictor of the final rating (more popular movies attract more viewers and hence ratings). As such an additional predictor, `n_rating`, is added to the edx dataset for use in subsequent model tests.

```
edx <- edx %>% group_by(movieId) %>% mutate(n_ratings = n()) %>% ungroup()
```

- Genres are also likely to be a useful predictor. In the `edx` dataset, the `genres` column contains a single string entry for each movie, which is the concatenation of all applicable genres. However, there are 797 distinct combinations of genres, and would be viewed by training algorithms as discreet genres. Applying the `sep` string function to the `genres` data, the number of unique genres are reduced to 20, and a matrix of unique movie genres is appended to the `edx` training set.

- In the development and testing stages of the appropriate training algorithms, the `edx` dataset was further partitioned into training set that only represented 1-10% of the full dataset. This allowed for more rapid training on a smaller `edx_sample` and `validation_sample`.

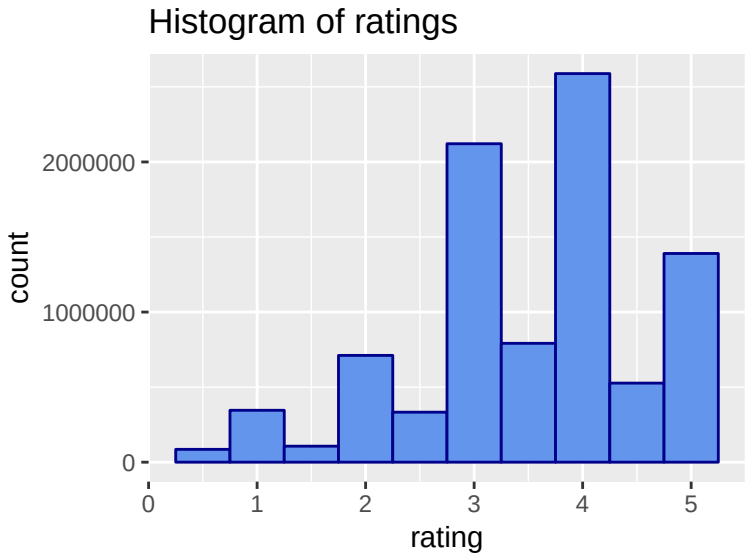
3.2. Exploratory data analysis

Exploratory analysis of the `edx` dataset was undertaken to provide insights into the distributions of key parameters in the training data.

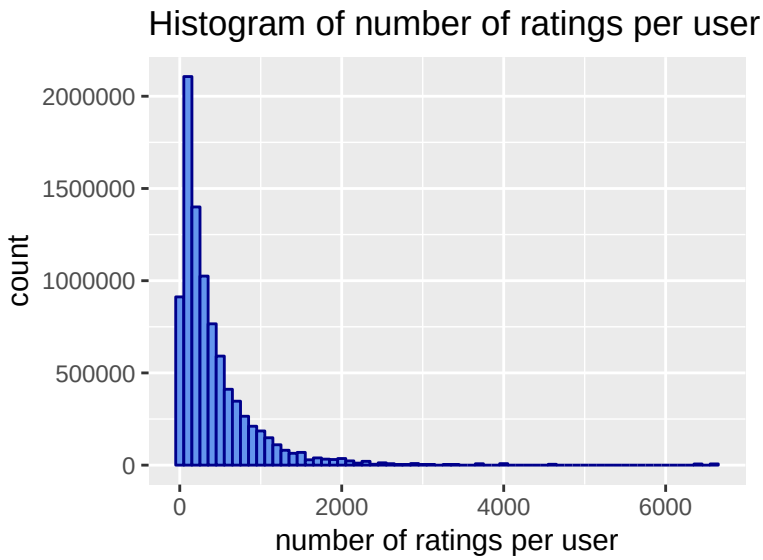
- A quick summary of the full `edx` dataset informs distributions across select key parameters

```
##      userId         movieId          rating        n_ratings
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :    1
##  1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.: 1637
##  Median :35753   Median : 1834   Median :4.000   Median : 4228
##  Mean   :35873   Mean   : 4120   Mean   :3.512   Mean   : 6788
##  3rd Qu.:53610   3rd Qu.: 3625   3rd Qu.:4.000   3rd Qu.: 9821
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :31322
```

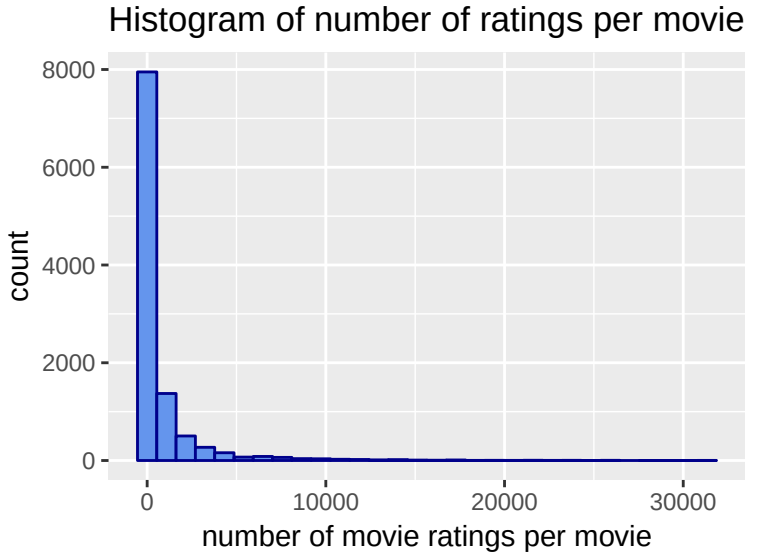
- Of particular interest is the histogram of ratings across the training set. Here one observes a far higher number of whole number ratings vs. ratings with half scores (potentially an insight for predictors at the training stage of the model).

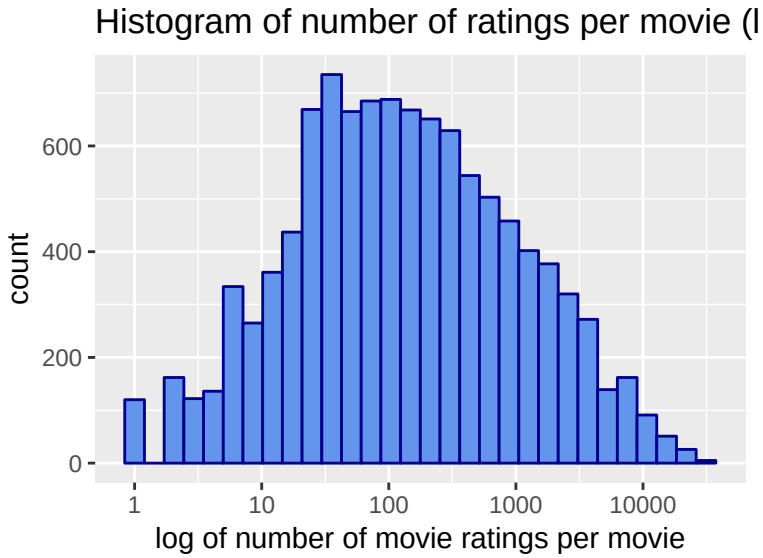


- The number of distinct users and movies are 69878 and 10677 respectively. With only 9000056 lines in the `edx` training set, this indicates the sparseness of the training matrix (i.e. only 1.2% - 9000056/(69878 * 10677) of the matrix of movie-user ratings).
- `edx` is further tested for small samples of either `movieId` or `userId` by filtering out movies and users with less than 100 ratings - approximately 75.1% of the movies and users have 100 or more entries.
- In terms of user activity levels to rate movies, the median number of ratings per user is 62, with a mean of 128.8 and standard deviation of 195.1
- Similarly, the movies data shows a median of 121 ratings per movie, a mean of 842.9 and standard deviation of 2238.7



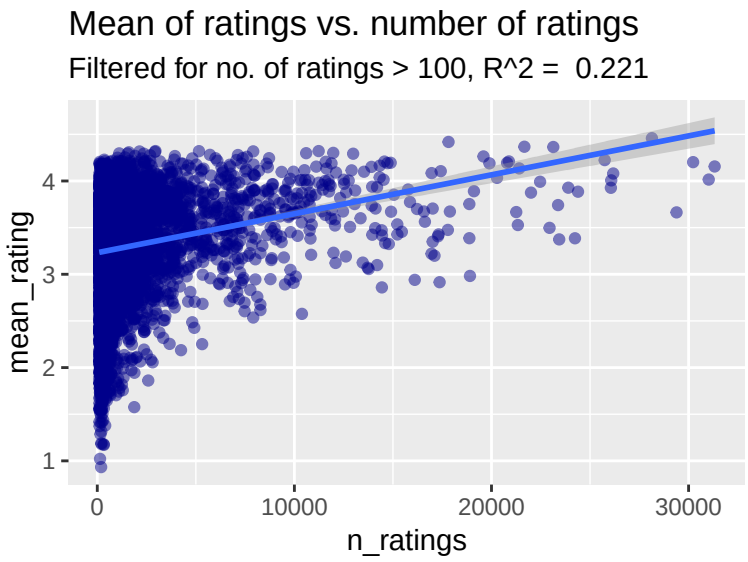
- Sorting for the top movies by number of ratings, we see blockbusters with >30,000 ratings as illustrated in the below histograms (linear and logarithm scales) and supporting summary table of the top 10 movies by number of ratings.






```
## # A tibble: 10 x 4
## # Groups:   movieId [10]
##    movieId title                                     n_rating mean_rating
##      <dbl> <chr>                                        <int>       <dbl>
##  1     296 Pulp Fiction (1994)                          31322        4.16
##  2     356 Forrest Gump (1994)                          31022        4.02
##  3     593 Silence of the Lambs, The (1991)             30229        4.20
##  4     480 Jurassic Park (1993)                         29389        3.66
##  5     318 Shawshank Redemption, The (1994)             28140        4.46
##  6     110 Braveheart (1995)                            26177        4.08
##  7     457 Fugitive, The (1993)                         26075        4.01
##  8     589 Terminator 2: Judgment Day (1991)            26058        3.93
##  9     260 Star Wars: Episode IV - A New Hope (a.k.a.~   25744        4.22
## 10     150 Apollo 13 (1995)                             24390        3.89
```

- This also aligns with a hypothesis to test for key correlations, e.g. the number of ratings a movie receives versus the rating itself (more people will watch and rate popular movies, and the top 10 movies by number of ratings have mean ratings higher than the global mean for the `edx` set, i.e. 3.512). This relationship between the number of ratings and the mean rating per movie shows a moderate correlation of 0.221.



3.3. Developing a recommendation system

Modelling was initially tested on a subset of the `edx` training set, in the interests of speed of processing and confirmation of coding and overall approach. In the final modelling, the training was conducted on the entire `edx` training set and tested against the `validation` test set.

The chosen approach follows the HardvardX course approach. Whilst knn and random forest training approaches where attempted, the size of the dataset made these approaches computationally infeasible on a standard business laptop (running a windows based i7 @ 1.8GHz with 8Gb RAM)

The results section that follows summarises 4 models tested, i.e. a simple mean of ratings, adding a movie bias effect, adding a movie and user bias effect, and finally regularizing the movie and user bias effects with optimal lambdas for each parameter.

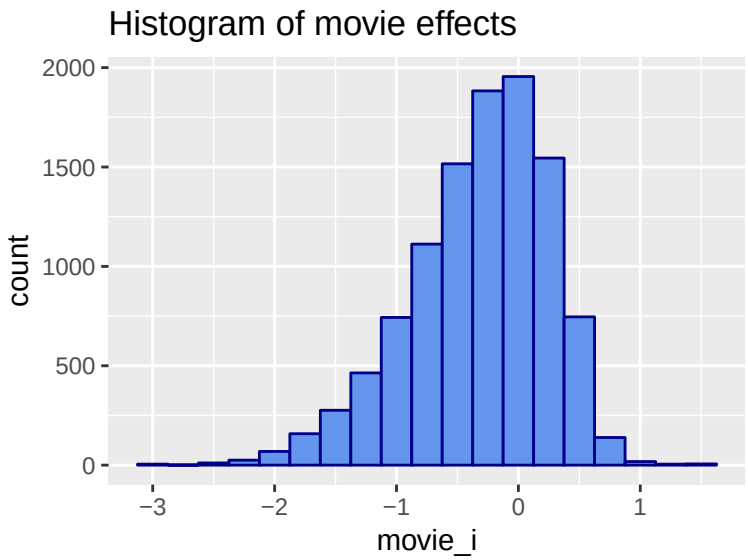
4. Results

4.1. Test model 1 - simple mean

This first model calculates the mean of the entire `edx` training set (3.5124266) and uses this as an estimate for the test set. The resultant RMSE is 1.0605554

4.2. Test model 2 - adding movie bias

It can be expected that certain movies will, on average, score higher or lower than the mean of all movies given its overall popularity with viewers. Model 2 adds a movie bias to the basic mean estimate (test 1), i.e. predicted rating = mu + b_movie where b_movie_i is the mean of the differences between a specific movie's ratings and the average of the full dataset.

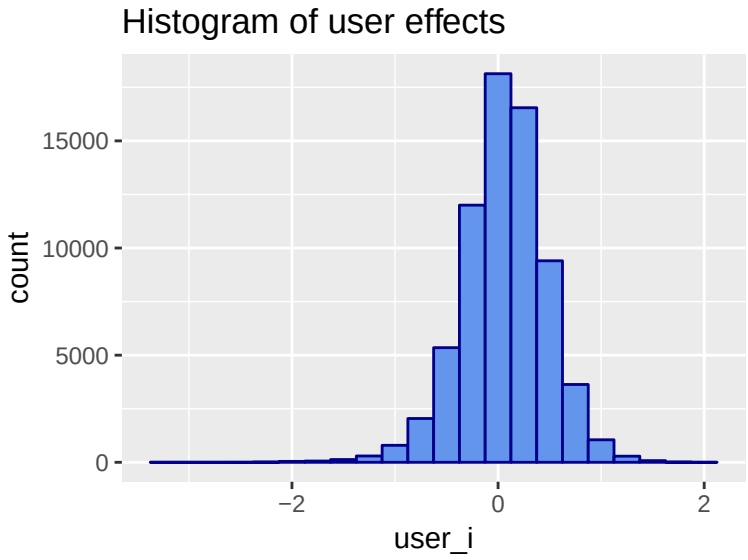


Testing for the spread of movie biases on the `edx` dataset one can see a longer tail to the downside, i.e. certain movies score very poorly versus the mean.

Applying the movie bias to the recommendation algorithm for model 2 gives an RMSE of 0.943831, a big improvement on the simple mean estimate in Test 1.

4.3. Test model 3 - adding user bias in addition to the movie bias

Similarly for model 3, the biases for individual users are added to the prediction model. Some users can be expected to rate movies more harshly than others, and this spread is also seen in the below figure.



Test model 3 - adding a user bias to the mean rating estimate and movie bias - gives an RMSE of 0.8655584, a further improvement as we add an additional predictor.

4.4. Test model 4 - adding both movie and user bias, and regularizing both with differing lambdas

In the final model selected, the movie and user effects are regularized with different lambdas. After a few runs to narrow the range of the lambdas, the resultant minimal RMSE is found at lambda_movie = 4.75 and lambda_user = 5, as seen per the table below:

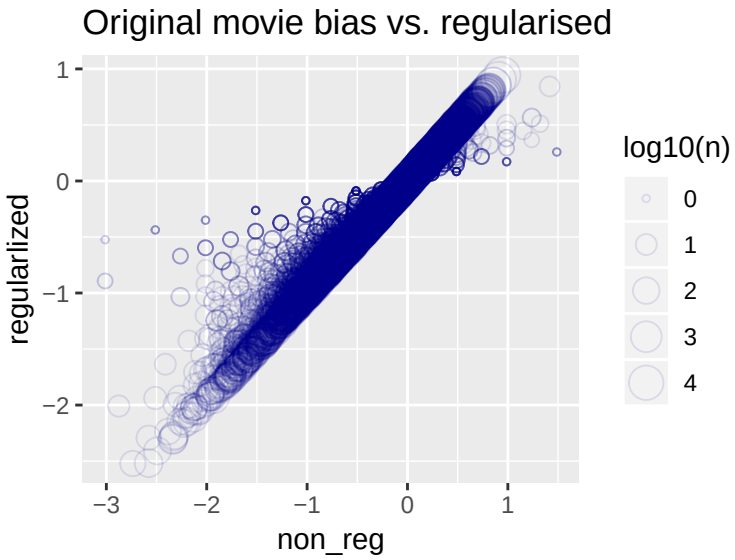
```
##          4.5    4.75       5    5.25     5.5    5.75       6
## 3.5  0.86512 0.86512 0.86512 0.86512 0.86512 0.86513 0.86513
## 3.75 0.86512 0.86512 0.86512 0.86512 0.86512 0.86512 0.86513
## 4    0.86512 0.86512 0.86512 0.86512 0.86512 0.86512 0.86513
## 4.25 0.86512 0.86511 0.86511 0.86512 0.86512 0.86512 0.86513
## 4.5  0.86512 0.86511 0.86511 0.86512 0.86512 0.86512 0.86513
## 4.75 0.86512 0.86511 0.86511 0.86512 0.86512 0.86512 0.86513
## 5    0.86512 0.86511 0.86511 0.86512 0.86512 0.86512 0.86513
## 5.25 0.86512 0.86511 0.86511 0.86512 0.86512 0.86512 0.86513
## 5.5  0.86512 0.86512 0.86512 0.86512 0.86512 0.86512 0.86513
## 5.75 0.86512 0.86512 0.86512 0.86512 0.86512 0.86512 0.86513
## 6    0.86512 0.86512 0.86512 0.86512 0.86512 0.86512 0.86513
```

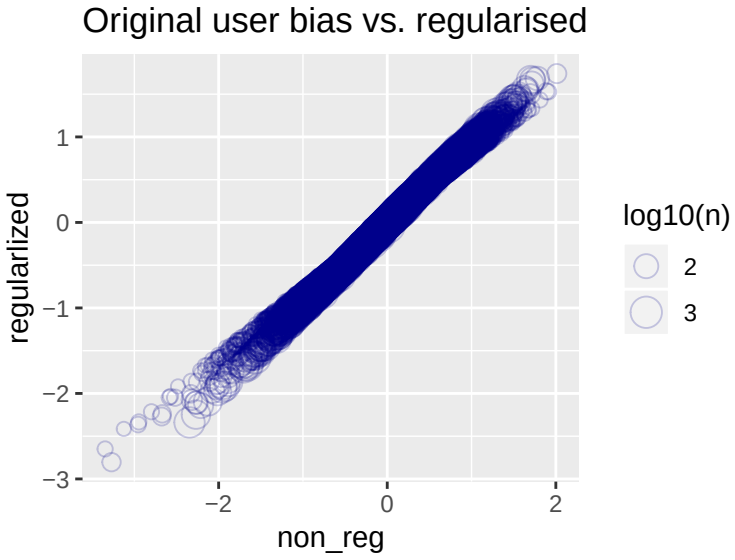
The optimal lambdas are generated from the following code, running a nest for loop to test discrete lambdas for both.

```
lambdas_movie <- seq(3.5, 6, 0.25)
lambdas_user <- seq(4.5, 6, 0.25)
rmses <- matrix(nrow = length(lambdas_movie), ncol = length(lambdas_user))

for (l_m in 1:length(lambdas_movie)){
  for (l_u in 1:length(lambdas_user)){

  mu <- mean(train_set_sample$rating)

  b_movie_i <- train_set_sample %>%
    group_by(movieId) %>%
    summarize(b_movie_i = sum(rating - mu)/(n()+lambdas_movie[l_m]))

  b_user_i <- train_set_sample %>%
    left_join(b_movie_i, by="movieId") %>%
    group_by(userId) %>%
    summarize(b_user_i = sum(rating - b_movie_i - mu)/(n()+lambdas_user[l_u]))

  pred_ratings <-
    test_set_sample %>%
    left_join(b_movie_i, by = "movieId") %>%
    left_join(b_user_i, by = "userId") %>%
    mutate(pred = mu + b_movie_i + b_user_i) %>%
    pull(pred)

  rmses[l_m,l_u] <- RMSE(pred_ratings, test_set_sample$rating)
  }
}
colnames(rmses) <- lambdas_user
rownames(rmses) <- lambdas_movie

lambdas_movie_optimal <-
  lambdas_movie[ifelse(which.min(rmses) %% length(lambdas_movie) == 0,
```

```
                        length(lambdas_movie),
                        which.min(rmses) %% length(lambdas_movie))]
lambdas_user_optimal <- lambdas_user[ceiling(which.min(rmses)/length(lambdas_movie))]
```

Applying these optimal lamdas to regularize the this model yields a RMSE of 0.865114.
We see the impact of regularization on the movie effect and on the user effect with the below 2 graphs that plot the original unregularized estimates, and then the regularized that dampens the effect for movies/users with a lower number of ratings (thus not overbiasing the model towards these smaller samples)





4.5. Summary of test models

In summary, the 4 test models yield the following RMSEs with progressively better RMSEs

```
##                          method      RMSE
## 1                   Global mean 1.0605554
## 2                 Add Movie bias 0.9438310
## 3       Add Movie and User Bias 0.8655584
```

```
## 4 Regularized Movie and User Bias 0.8651140
```

5. Conclusions

The final model selected (Test model 4) achieves an RMSE of 0.865114 (lower than the targeted 0.87750 per the grading rubric). Beyond this project and with more time, I would like to further explore some of the following refinements to the current prediction approach:

- Test alternative ML methodologies on a smaller dataset, e.g. KNN, random forests, UBCF
- Test combinations of approaches into an ensemble prediction
- Test the impact of adding additional predictors, e.g. time effects, genres. (I did test the number of ratings on top of my final model with no noticeable improvement in RMSE and thus dropped that parameter from the final approach)

The subsequent "choose your own" Capstone will afford an opportunity to test a broader range of machine learning approaches on a smaller dataset.