MovieLens project

Angus Jenner

05/03/2022

Overview:

This project was to use machine learning to build a movie recommendation system. The recommendation system will be evaluated on the root mean square error (RMSE) of its ratings predictions. The movielens dataset - which has 10,000,054 ratings from 69,878 unique users - was split into development and validation sets, where the validation set represented 10% of the total data. Ratings predictions were generated from the development set and applied to the validation set to determine a final RMSE.

The movielens data set includes fields for user id, movie id, movie title, movie genres, the time of review (as a timestamp),and movie rating. Movie year was not recorded in its own field, but was able to be extracted via data cleaning.

To develop a machine learning algorithm, the development set was split into training and test sets - where the test set represented 10% of the data in the development set. Algorithms were able to be evaluated using this data and once an algorithm had been selected, the final stage was to evaluate this (using the entire development set) against the final validation set. The final RMSE for the chosen algorithm is the primary deliverable of this report.

Methods:

Data cleaning:

The first step of the process was to examine the data and determine what data cleaning could, and should, be completed. Movie year was not reported separately, but was included within parenthesis within the movie title (a character field). To extract this data and include it within its own numeric column, "right" and "left" functions were built. These functions take data and output a chosen number of characters counting from either the left or right of the data. An example of the procedure followed is included below.

```
as.numeric(left(right("Toy Story (1995)", 5), 4))
```

```
## [1] 1995
```

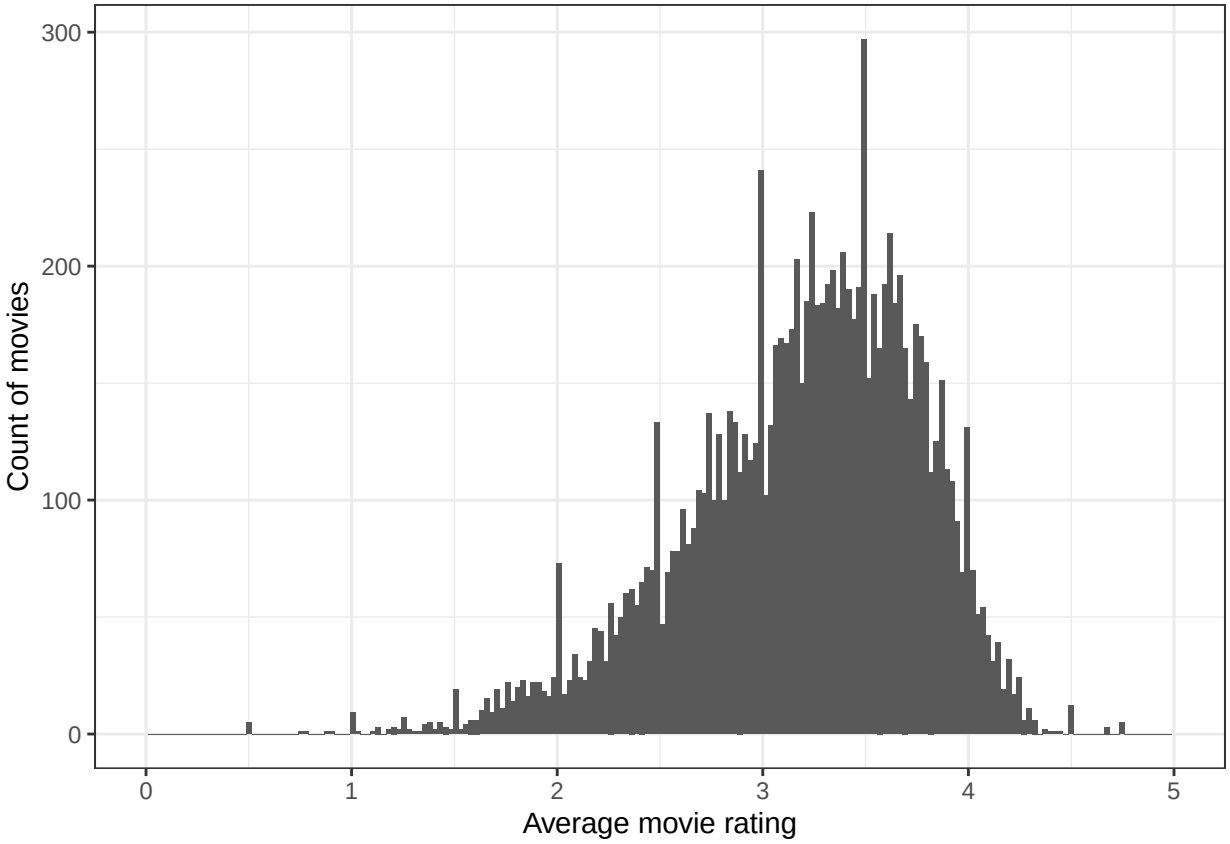
Genre data was reported for each movie within a single column. Since movies have multiple genres, the data reported all relevant genres within a single field, with genres split by "|". This format makes it difficult to identify movies by single genres, e.g. all dramas, and as a result would hamper attempts to use this data to improve predictions. As a result a list of all unique genres was extracted and for each genre a column was set up which displayed TRUE if the movie was of that genre, or FALSE if it was not. This allowed one to build algorithms that, for example, could consider users who have a preference for/or against certain genres. After determining the unique genres reported - using the XXX function - the grepl function was used to build columns that displayed TRUE or FALSE for each genre.

```
Comedy = grepl("Comedy", genres, fixed = TRUE)
```

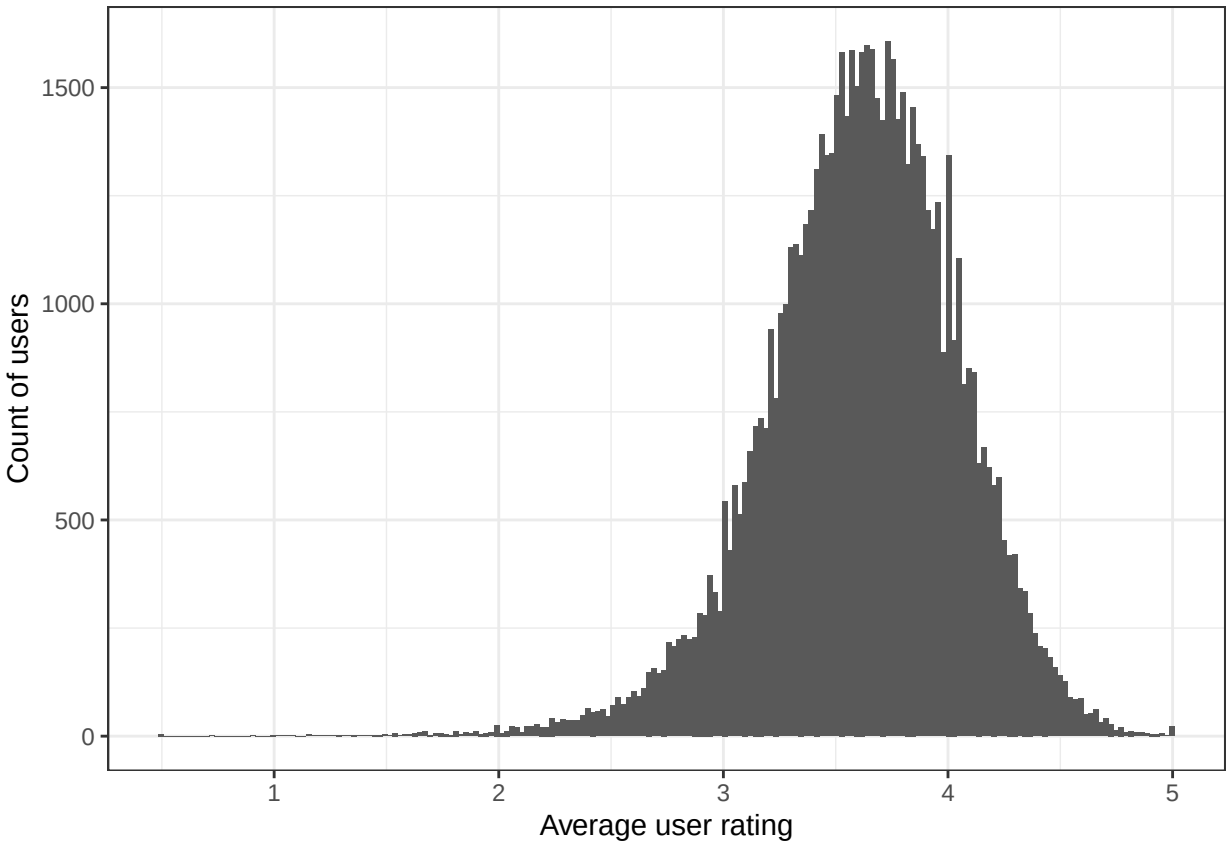
Data exploration, visualisation and relevant insights:

To explore whether movie specific effects would be needed, each movie's average rating was plotted as a histogram. This plot shows a right skewed distribution - mean = 3.51, standard deviation = 1.06 . As expected, there was wide variation in average movie ratings and therefore it was important to include movie specific effects within the model. If there were no variation in average movie ratings, these effects would not be needed.

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

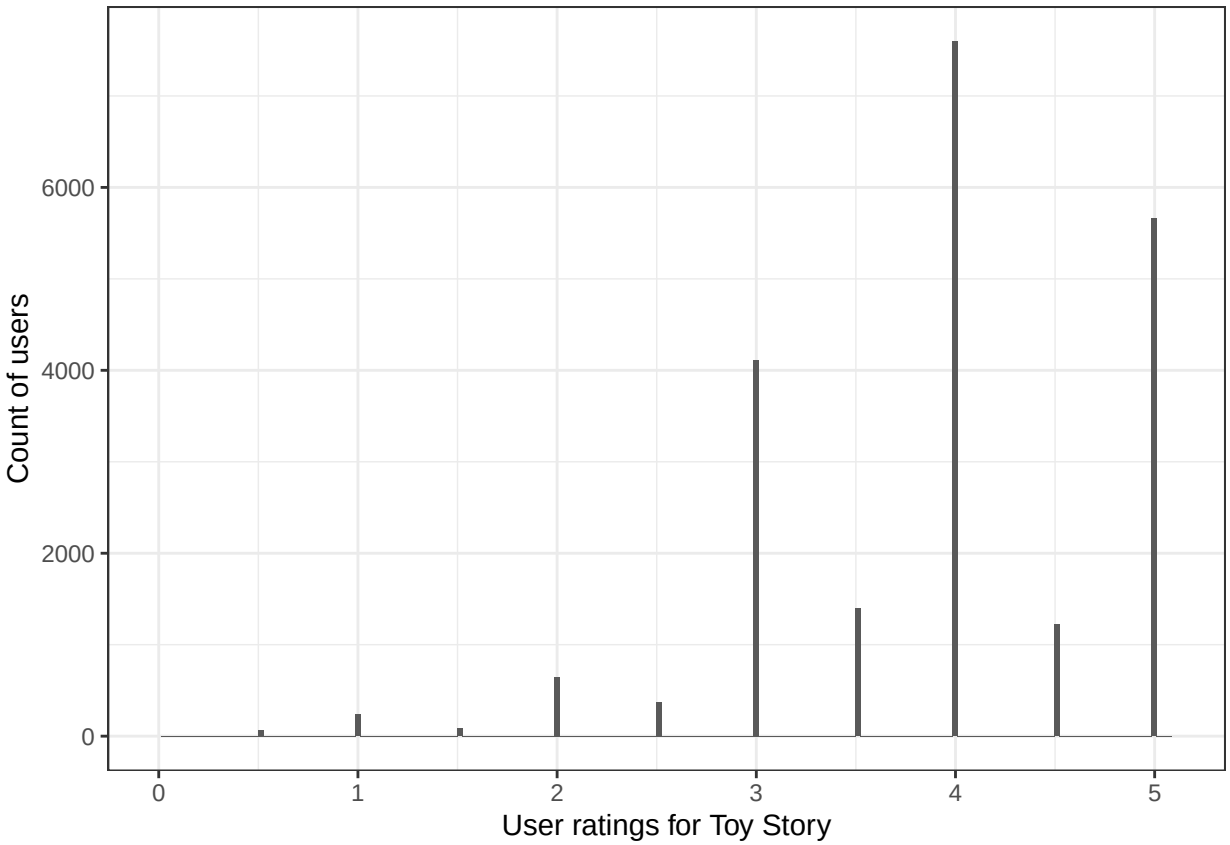


As would be expected there was also variation in user effects - i.e. there were users that were particularly generous, or stingy with their ratings across all rated films. This can be seen within the below histogram.



The effect persisted even when controlled for the films that the users were rating - see below for the variation in ratings for the film Toy Story. As a result there was a benefit to exploring user effects.

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



To develop a more complex algorithm, consideration also needed to be given to variation within users. The primary area of interest was in genre. It was anticipated that some individuals would, for example, have a preference for "Action" movies where others would have preferences against "Action" movies. To explore this, genres that corresponded to a large numbers of films - Action, Drama, Sci-Fi, Crime and Adventure - were examined. For each, the mean of the absolute difference between the average score for genre = TRUE movies, and genre = FALSE movies was computed across users within the training set. In addition, the standard deviation for the difference in average genre = TRUE movies, and genre = FALSE movies was computed across users. Also the proportion of all movies that had this genre = TRUE was computed.

Each of these measures gave an indication of the importance of these genres from an algorithmic point of view. We would ideally like to explore a genre where there is wide deviation between ratings (given by the first two measures) and where this genre is popular (so there is maximum differentiation). To do this we computed a Weight term to help identify the best genres to explore - this Weight was the the product of the difference terms and the proportion of TRUE movies. A table of results is included below - Action and Drama achieved the highest weights.

```
##       Genre Average_abs_diff SD_of_diff Prop_TRUE     Weight
## 1    Action        0.3804499  0.4953724 0.2844909 0.05361642
## 2     Drama        0.3728496  0.4044841 0.4344392 0.06551853
## 3 Adventure        0.3401354  0.4713235 0.2120961 0.03400193
## 4     Crime        0.3949927  0.5224210 0.1474976 0.03043650
## 5    Sci-Fi        0.4269420  0.5672780 0.1490493 0.03609896
```

From this data it was anticipated that the machine learning algorithm could be improved by considering genre dependent user effects, as opposed to just user effects - i.e. include users' preferences over different genres.

Modeling approach:

It was deemed useful to start with a more simple algorithm and build additional layers of complexity onto this algorithm. This approach allowed one to examine the improvement that an additional layer of complexity had on the prediction accuracy - reported in terms of RMSE. To start, therefore, the overall mean rating (across all movies and users) was considered as the baseline algorithm. On top of this, as alluded to in the data exploration section, additional layers were built on top of this. Movie effects, user effects and genre dependent user effects.

Work was also completed with the recommenderlab package (Michael Hahsler (2016)). When applied to a small subset of the data this package improved predictions markedly. This approach, however, had to be abandoned since the hardware R was running on did not have enough memory to deal with the size of the development dataset - even when extra work was completed to run from a sparse matrix.

Results:

Algorithm 1: Using the mean rating to predict ratings

This was the baseline algorithm, which enabled a RMSE floor against which to compare other algorithms. The mean rating across all ratings in the training set was taken and the RMSE computed against the test set.

```
##        Algorithm             RMSE
## 1 Rating average 1.06005370222409
```

Algorithm 2: Computing movie effects

As noted within the data exploration section, there is variation in movies' average ratings - implying differences in quality. As a result one can predict a user's rating of a movie by using the movie's average rating.

```
##        Algorithm              RMSE
## 1 Rating average  1.06005370222409
## 2  Movie effects 0.942961498004501
```

Algorithm 3: Computing movie effects and user effects

As well as variation within movies, there was also variation between users - different users rating the same films differently. By including a term which represented how the user's rating differs from the average rating of that film (i.e. algorithm 2) additional accuracy was obtained.

```
##                Algorithm              RMSE
## 1         Rating average  1.06005370222409
## 2          Movie effects 0.942961498004501
## 3 Movie and user effects  0.86468429490229
```

Would regularisation improve estimates?

Following the first three algorithms, it was noted that movies with few ratings, and users with few reviews were likely to excessively weighted within our algorithms. For example. if a movie were to have a single five star review, this would be predicted for that movie within the test set. In the same way a user who reviewed

a single film five stars would be expected to be a very generous rater. This is not an inference that we wish to make given the small amount of data relied on.

As a result, the best and worst movies were examined to understand if this small n problem was occurring.

```
##       Best_movies
##  [1,] "Hellhounds on My Trail (1999)"
##  [2,] "Satan's Tango (SÃ¡tÃ¡ntangÃ³) (1994)"
##  [3,] "Shadows of Forgotten Ancestors (1964)"
##  [4,] "Fighting Elegy (Kenka erejii) (1966)"
##  [5,] "Sun Alley (Sonnenallee) (1999)"
##  [6,] "Blue Light, The (Das Blaue Licht) (1932)"
##  [7,] "Who's Singin' Over There? (a.k.a. Who Sings Over There) (Ko to tamo peva) (1980)"
##  [8,] "Life of Oharu, The (Saikaku ichidai onna) (1952)"
##  [9,] "Human Condition II, The (Ningen no joken II) (1959)"
## [10,] "Human Condition III, The (Ningen no joken III) (1961)"
##       Number_of_reviews
##  [1,] "1"
##  [2,] "1"
##  [3,] "1"
##  [4,] "1"
##  [5,] "1"
##  [6,] "1"
##  [7,] "4"
##  [8,] "2"
##  [9,] "4"
## [10,] "4"
```

```
##       Worst_movies                                    Number_of_reviews
##  [1,] "Besotted (2001)"                               "1"
##  [2,] "Hi-Line, The (1999)"                           "1"
##  [3,] "Accused (Anklaget) (2005)"                     "1"
##  [4,] "Confessions of a Superhero (2007)"             "1"
##  [5,] "War of the Worlds 2: The Next Wave (2008)"     "2"
##  [6,] "SuperBabies: Baby Geniuses 2 (2004)"           "47"
##  [7,] "Disaster Movie (2008)"                         "30"
##  [8,] "From Justin to Kelly (2003)"                   "183"
##  [9,] "Hip Hop Witch, Da (2000)"                      "11"
## [10,] "Criminals (1996)"                              "1"
```

As can be seen there are many cases where there are very few reviews (often only one) in these groups - regularisation is needed. Regularisation attempts to counteract this issue by increasing the value of the denominator when averaging takes place - rather than dividing by n, one divides by n + lambda. For small values of n this makes a big difference, but for large values of n this makes far less of a difference. As a result, small n items are downweighted relative to high n items. To demonstrate a regularisation factor of 3 was used - results below:

```
## # A tibble: 10,677 x 4
##    movieId   b_i   n_i title
##      <dbl> <dbl> <int> <chr>
##  1     318 0.944 25188 Shawshank Redemption, The (1994)
##  2     858 0.904 15975 Godfather, The (1972)
##  3      50 0.854 19457 Usual Suspects, The (1995)
##  4     527 0.852 20877 Schindler's List (1993)
```

```
##  5     904 0.812  7115 Rear Window (1954)
##  6     912 0.807 10141 Casablanca (1942)
##  7     922 0.803  2633 Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)
##  8    3435 0.794  1950 Double Indemnity (1944)
##  9    2019 0.793  4648 Seven Samurai (Shichinin no samurai) (1954)
## 10    1178 0.792  1410 Paths of Glory (1957)
## # ... with 10,667 more rows
```

```
## # A tibble: 10,677 x 4
##    movieId   b_i   n_i title
##      <dbl> <dbl> <int> <chr>
##  1    8859 -2.60    47 SuperBabies: Baby Geniuses 2 (2004)
##  2    6483 -2.60   183 From Justin to Kelly (2003)
##  3   61348 -2.50    30 Disaster Movie (2008)
##  4    6371 -2.41   124 PokÃ©mon Heroes (2003)
##  5    1826 -2.34   186 Barney's Great Adventure (1998)
##  6    4775 -2.32   311 Glitter (2001)
##  7    6587 -2.29   281 Gigli (2003)
##  8    3574 -2.29    61 Carnosaur 3: Primal Species (1996)
##  9    5672 -2.27   188 Pokemon 4 Ever (a.k.a. PokÃ©mon 4: The Movie) (2002)
## 10    5739 -2.18    73 Faces of Death 6 (1996)
## # ... with 10,667 more rows
```

These results are much more plausible. As a result regularisation will be used.

Algorithm 4: Regularised movie effects (user effects excluded)

One issue with regularisation is choosing of the optimal lambda which should be chosen to maximise the accuracy of the algorithm. To do this, cross validation was used. For the movie-only regularised effects, the value of lambda that maximised accuracy was 1.5.

```
##                    Algorithm              RMSE
## 1             Rating average  1.06005370222409
## 2              Movie effects 0.942961498004501
## 3    Movie and user effects  0.86468429490229
## 4 Regularised movie effects 0.942936965845895
```

Algorithm 5: Regularised movie and user effects

Regularisation was done now for both user and movie effects. Again cross-validation was used to choose the optimal lambda to be used for both the user and movie effects. In this algorithm both effects used the same value of lambda - technically different values of lambda could be used but for memory purposes the cross validation was limited to one variable. The value of lambda that maximimsed accuracy was 5. Relevant RMSE reported below - an improvement on the non-regularised version (Algorithm 3).

```
##                            Algorithm              RMSE
## 1                       Rating average  1.06005370222409
## 2                        Movie effects 0.942961498004501
## 3              Movie and user effects  0.86468429490229
## 4           Regularised movie effects 0.942936965845895
## 5 Regularised movie and user effects 0.864136179290374
```

Algorithm 6: Regularised movie and genre-dependent user effects

As noted in the data exploration section, one was expected to gain accuracy by splitting users' average ratings by whether the film was a certain genre or not. The two genres that scored highest on the Weight measure were Action and Drama. As a result, Action and Drama were considered separately:

Some work was required to prepare an genre-dependent user effects matrix to refer to - once this was developed, the merge (rather than the left-join) function could be used to match by both user and genre at the same time.

Regularised movie and Action-dependent user effects

```
##                                          Algorithm              RMSE
## 1                                   Rating average  1.06005370222409
## 2                                    Movie effects 0.942961498004501
## 3                              Movie and user effects  0.86468429490229
## 4                            Regularised movie effects 0.942936965845895
## 5                  Regularised movie and user effects 0.864136179290374
## 6 Regularised movie + action-dependent user effects 0.859954798987423
```

Regularised movie and Drama-dependent user effects

```
##                                          Algorithm              RMSE
## 1                                   Rating average  1.06005370222409
## 2                                    Movie effects 0.942961498004501
## 3                              Movie and user effects  0.86468429490229
## 4                            Regularised movie effects 0.942936965845895
## 5                  Regularised movie and user effects 0.864136179290374
## 6 Regularised movie + action-dependent user effects 0.859954798987423
## 7  Regularised movie + drama-dependent user effects 0.860953190063743
```

The action-dependent user effects improved accuracy marginally more.

Algorithm 7: Regularised movie and two sets of genre-dependent user effects

It was clear from Algorithm 6A and 6B that individual genre-dependent user effects improved accuracy. It stands to reason that including genre-dependent effects for both of these genres within the same algorithm should improve accuracy. In theory one should be able to do this for all genres, but limitations of laptop memory meant that a maximum of two genres could be considered.

Regularised movie, Action-dependent user effects and Drama-dependent user effects

```
##                                                    Algorithm
## 1                                             Rating average
## 2                                              Movie effects
## 3                                        Movie and user effects
## 4                                      Regularised movie effects
## 5                            Regularised movie and user effects
## 6                 Regularised movie + action-dependent user effects
## 7                  Regularised movie + drama-dependent user effects
## 8 Regularised movie + both action- and drama-dependent user effects
```

```
##                RMSE
## 1  1.06005370222409
## 2 0.942961498004501
## 3  0.86468429490229
## 4 0.942936965845895
## 5 0.864136179290374
## 6 0.859954798987423
## 7 0.860953190063743
## 8 0.857975507871183
```

The performance of the algorithm was improved with the addition of both genre-dependent user effects. It is anticipated that with each added genre the improvement in accuracy would be smaller.

Final algorithm to be evaluated:

Algorithm 7 will be used on the entire development set to make predictions - and compute a final RMSE - for the validation set. This algorithm was chosen as it shows evidence of improving the accuracy of predictions while also being manageable on the hardware being used, where memory limits have served to be an issue.

The RMSE recorded for the validation set was 0.858301.

```
##                                                          Final_algorithm
## 1 Regularised movie + both action- and drama-dependent user effects
##               RMSE
## 1 0.85830143733664
```

Conclusion:

Recommendation algorithms are a useful demonstration of machine learning in practice. This report explored a movie rating dataset and explored the extent to which prediction accuracy could be improved, the results of which could be used for recommendation purposes. The report explored movie and user effects and then incorporated additional genre data (extracted through data cleaning) to define genre-specific effects. The rationale behind this is that there are individuals who have strong genre preferences and as a result additional prediction accuracy could be obtained by utilising these. Algorithms were also improved via the use of regularisation.

This report was, however, limited by the memory on available hardware. The recommendation lab package was explored, but the dataset was too large for computation on this hardware. This approach therefore had to be abandoned. The memory issues also impacted the ability to cross-validate multiple regularisation factors at a single time (in the final algorithm the movie and action-dependent user-effects have a pooled lambda), and stopped additional genre-specific user effects being added to the model - the final algorithm is limited to two of these.

To improve the final algorithm, future work should take place on hardware with more memory to remove the limitations faced within this project.

References:

Michael Hahsler (2016). recommenderlab: A Framework for Developing and Testing Recommendation Algorithms, R package. https://CRAN.R-project.org/package=recommenderlab