Basic MovieLens Recommendation Report

HarvardX Data Science Capstone Project

Codrin Kruijne

2019-05-31

This report has been prepared for the HarvardX Data Science program Capstone course final project submission and consists of three files of original work that can be found on GitHub (with all commits): this report in PDF, this report in R markdown and an R markdown script file with all the code (also knitted to PDF with results for convenience). Results are read in from data objects stored locally (also saved in the GitHub repository), that were the output of the script file. Next to the course requirements my learning goal was to get familiar with parallel processing and to explore using AutoML with H2O in R.

Introduction

The dataset consists of 10M movie ratings by users, including movie title, year and genre. The goal of the assignment was to devise a way to predict movie ratings. I generated predictions derived from user and movie characteristics, using linear regression and applying AutoML. Model performances was assessed using RMSE. Derived models, requiring little computation, get to under 0.95 deviation from the actual ratings, the best linear model achieves below 0.85 and the ensemble models from autoML do not improve upon that. Derived models may be improved by calculating genre and period statistics in future to include as features in further refined models.

Analysis

Recommendation systems

Exploring infromation on recommender systems from the course and on Wikipedia, algorithms can roughly be based on collaborative filtering (recommendations based on users with similar behaviour) or content-based filtering (recommendation based on movie characteristics.)

Raw data

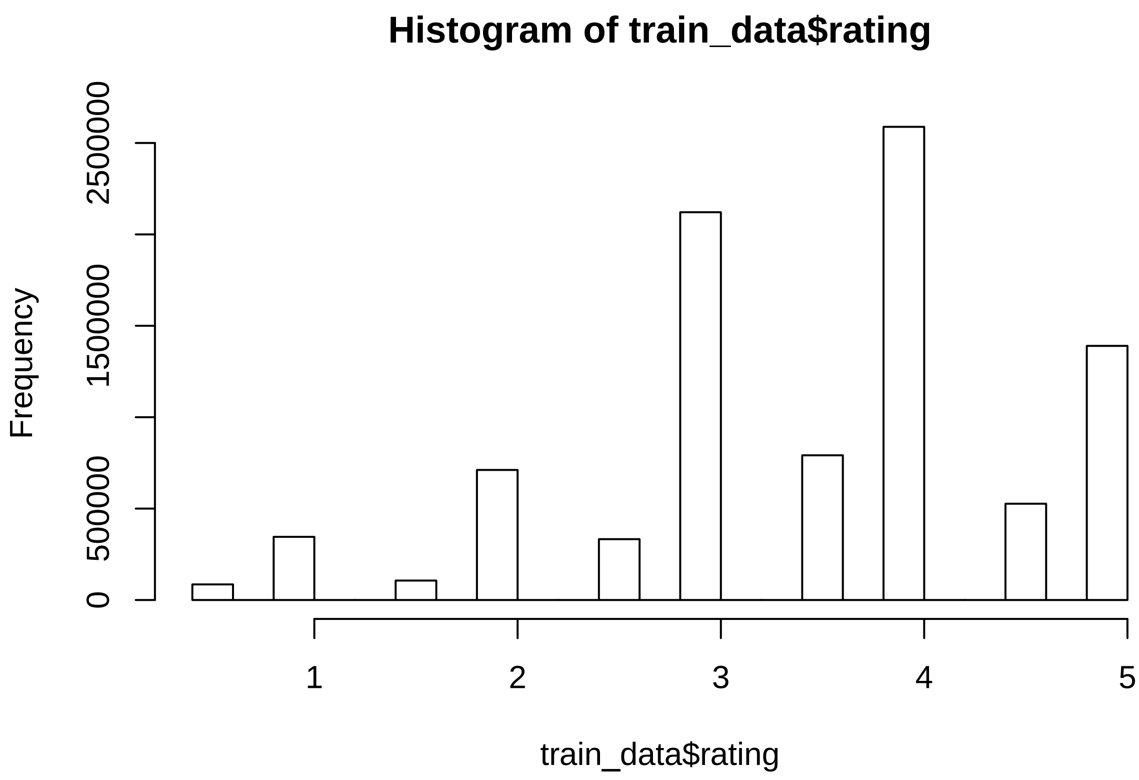
Training and validation datasets were provided, totalling about 10M records, which may be a challenge to compute complex models on a desktop. The data is a combination of user (identity), movie (title, year and genre) and rating (rating on a scale of 1 to 5 and date) characteristics.

Data preparation

Data on all entities (user, movie and rating) were in one table. I separated the data into user, movie and rating characteristics tables. Those tables were used to calcluate some additional information like average movie rating, average user rating, etc. Movie genre information was extracted from a compound string, so that it could be using as boolean variables, as well as the movie release year.

Data exploration

```r
rating_hist <- readRDS("rating_hist.rds")
plot(rating_hist)
```



The data is not normally distributed, so there is a bias from the rating system used.

Modeling

As there is a lot of data an computing model can be quite resource intensive, we first looked at some derived approaches like predicting with average overall rating and average movie rating. This got our results (RMSE was required as evaluating measure) down to below 0.95.

```r
dr <- readRDS("derived_results.rds") %>% arrange(RMSE)
knitr::kable(dr, caption = "Derived results")
```

Table 1: Derived results

method	RMSE
Movie mean	0.9383091
Rating effect	0.9383092
User effect	0.9446028
Average training rating	1.0612018

method	RMSE
Movie effect	1.0612018
User-movie effect	1.0612018
Fixed number 3.5	1.0612699

Then, I continued with Linear Modelling using Caret. Based on the theory that recommendation systems be based on rater and movie characteristics, we generated a number of models with original data and information calculated therefrom, which already improved results to below 0.85.

```r
lmr <- readRDS("lm_results.rds")  %>% arrange(rmse)
knitr::kable(lmr, caption = "Linear models results")
```

Table 2: Linear models results

model	rmse
rating ~ movie_median + movie_mean + user_median + user_mean	0.8452112
rating ~ movie_mean + user_mean	0.8452248
rating ~ movie_median + movie_mean + movie_sd + user_median + user_mean + user_sd	0.8467370
rating ~ movie_median + user_median	0.8843992
rating ~ movie_median + movie_mean	0.9383091
rating ~ movie_median + movie_mean + movie_sd	0.9386363
rating ~ user_median + user_mean	0.9395215
rating ~ user_median + user_mean + user_sd	0.9413628
rating ~ movie_median	0.9578211
rating ~ user_median	0.9679871

Finally, I used autoML from H2O to train and combine the best performing models automatically, which does not yield improved results. Even running for 10 hourse did not get under 0.9.

```r
amlr <- readRDS("autoML_results.rds")  %>% arrange(rmse)
knitr::kable(amlr, caption = "AutoML results in 1 hour")
```

Table 3: AutoML results in 1 hour

model_id	rmse
StackedEnsemble_AllModels_AutoML_20190531_192253	0.9636793
StackedEnsemble_BestOfFamily_AutoML_20190531_192253	0.9638613
GBM_5_AutoML_20190531_192253	0.9763070
DRF_1_AutoML_20190531_192253	0.9889825
XRT_1_AutoML_20190531_192253	0.9930478
GBM_4_AutoML_20190531_192253	1.0087012
GBM_3_AutoML_20190531_192253	1.0196827
GBM_2_AutoML_20190531_192253	1.0240631
GBM_1_AutoML_20190531_192253	1.0277414
DeepLearning_1_AutoML_20190531_192253	1.0468895
GBM_grid_1_AutoML_20190531_192253_model_1	1.0569857
GLM_grid_1_AutoML_20190531_192253_model_1	1.0596654

Conclusion

Basic derived models already provide predictions to less than one star accuracy. Machine Learning with linear regression improves results when both movie and rater data are used. H2O autoML, which includes various approaches, does not improve results with 10 hours of calculation. Maybe calculating genre popularity and some statistics regarding years of movie and rating, we may improve simple derived models.

References

1. Winning the Netflix Prize: A Summary
2. Wikipedia article "Recommender system"