Alternating Least Squares for Movie Recommendation in Spark

Toby Staines
toby.staines@city.ac.uk

Adrian Ellis
adrian.ellis@city.ac.uk

Domain

Recommendation systems are a key part of many modern websites and applications. The ability to accurately select items which a customer is likely to be interested in is highly beneficial; if executed well it can improve both sales and customer satisfaction.

The task of a recommendation system is to match users to products which they might be interested in, either by identifying similar users and looking at their history, or by identifying products which are similar to those the user has already reviewed and/or purchased.

One of the key difficulties with recommendation data is that it tends to be both big and sparse, making it a good target for this project. A website may have hundreds of thousands of both users and products, making a potentially vast matrix comparing the two. However, the majority of users will review, purchase, or even view only a tiny proportion of the available products.

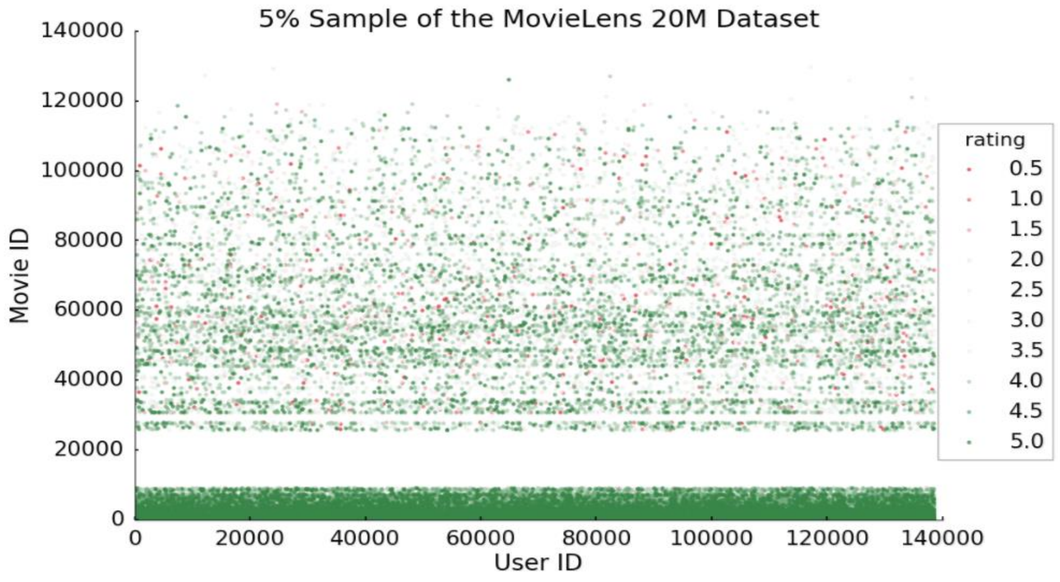
Here we illustrate a Collaborative Filtering approach to building a recommendation system in Apache Spark, using an Alternating Least Squares model

Data

The MoveLens 20M Dataset is a benchmark dataset consisting of 20 million film reviews, in the format of a user ID, a movie ID, and a rating from 0 – 5 in increments of 0.5. It consists of:

- **27,278** distinct movies
- **138,493** distinct users
- **20,000,263** reviews

This is a large amount of data, but still results in a matrix that is **99.47%** sparse
The graph bellow shows some interesting patterns in the data: There is a group of films with low IDs which have been very heavily reviewed, and several other bands indicating commonly reviewed films. A group of IDs between 1000 and 2500 appear to be unused.



Approach

To investigate the impact of changing the rating system on the accuracy of the model, we use the ML Pipeline function QuantileDiscritizer to create a new ratings column, on a scale of 0-2, in increments of 1.

A 10% sample of the data is taken as a test set. Then, with the remaining data, we repeatedly take a training sample and train one model for each rating system, with the training sample size increasing logarithmically from 0.01% to 100% of the non-test data.

Models are trained using the Alternating Least Squares (ALS) algorithm, with a root mean squared error loss function. ALS is a popular and successful choice for large scale collaborative filtering problems due to its efficiency working with sparse data, in this case reducing the effective dimensionality of the data from millions to no more than 20. RMSE works well as a loss function as the resulting loss is in the same scale as the value being predicted, so is easily interpretable.

For each model, a grid search over the following parameter settings is conducted:

Sample size: [0.0001, 0.001, 0.01, 0.1, 1]

A larger sample size should lead to more accurate predictions, but will be more costly to train

Rank: [1, 5, 10, 15, 20]

The rank parameter represents the number of latent variables to be calculated. The true number of strong latent variables in the data is unknown. Increasing the rank will increase computation time, but should also increase prediction accuracy, up to the point where the true number of latent variables is met or exceeded.

Regularisation: [0.1, 0.3, 0.5, 0.7, 0.9]

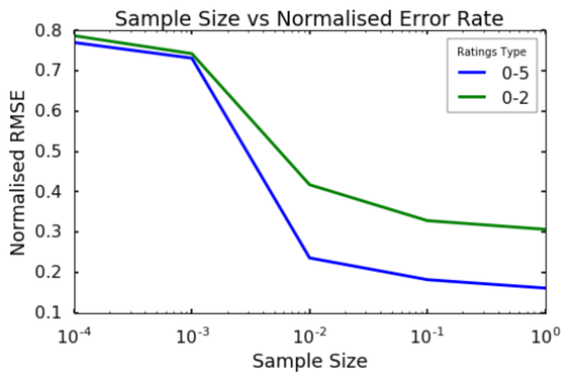
A regularisation parameter combats overfitting by penalising large parameters. However, too large of a regularisation may inhibit the algorithm from finding the optimal model. We anticipate an optimal regularisation parameter somewhere in the range tested.

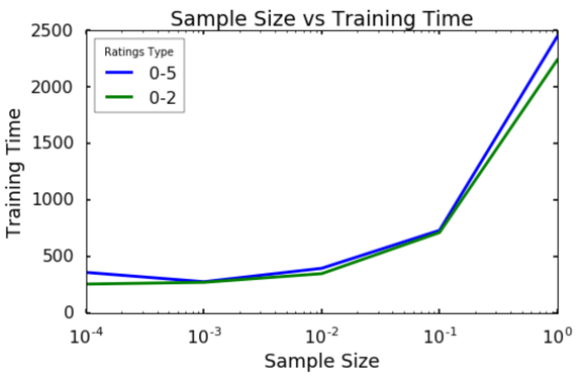
Results & Analysis

Sample Size and Rating System:

Results demonstrate that increasing the training sample size significantly reduces prediction error.
Training time increases with increasing sample size, but not in a linear fashion, and training on the full available dataset, which produced the best results, is not prohibitively slow.
The two ratings systems show no difference in training time, but the more granular five star system does produce a significantly more accurate morel at larger training set sizes (>1%)

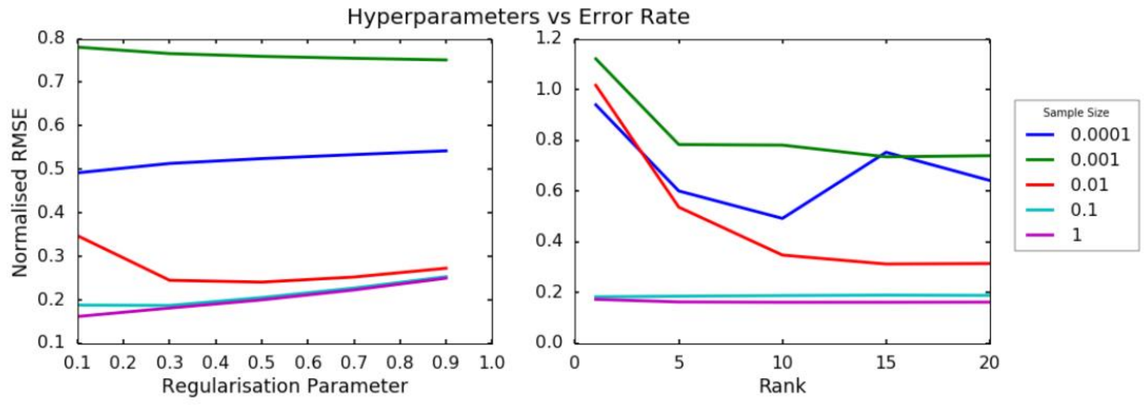
	Rating Type	Sample Size	Normalised Test RMSE	Train Time	Test Time
0	0-5	0.0001	0.770381	357.177402	33.280585
1	0-5	0.0010	0.731557	274.069499	28.747549
2	0-5	0.0100	0.235891	393.637302	32.936442
3	0-5	0.1000	0.181961	729.472953	32.849063
4	0-5	1.0000	0.161080	2447.796975	32.277814
5	0-2	0.0001	0.787223	253.666170	32.246799
6	0-2	0.0010	0.742793	269.423333	34.883843
7	0-2	0.0100	0.417516	346.211740	31.366062
8	0-2	0.1000	0.328370	710.671868	37.669535
9	0-2	1.0000	0.306886	2239.924758	30.024852





Rank and Regularisation:

As expected, increasing the rank resulted in an improvement in prediction accuracy, up to a point, although with the largest training set sizes it had very little impact.
Varying the regularisation parameter had surprisingly little impact. With a small sample size, a larger regularisation was preferable, but with larger sample sizes the opposite was true.
Further investigation in this area, with a wider range of regularisation may provide answers.

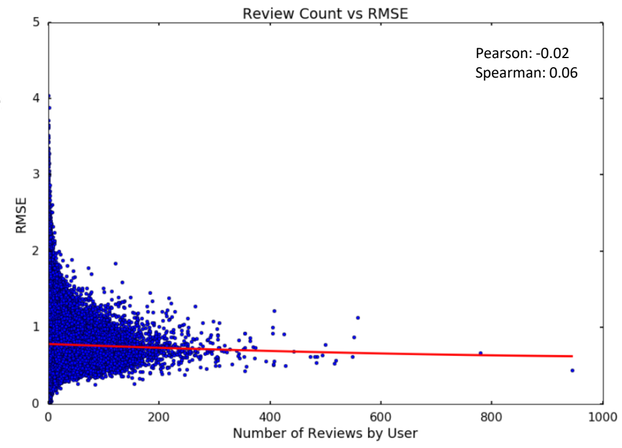


The Best Model:

The bets model developed showed a normalised RMSE on the test set of 0.16, a reasonably strong result.

	Rating	Prediction
Mean	3.53	3.42
Std Dev	1.05	0.63

Interestingly, summary statistics show that although the mean prediction is close to the mean rating, the standard deviation of the predictions is much lower – the model seems to be hedging its bets, and sticking closer to the mean that real reviewers do. This could indicate a weakness in RMSE as an evaluation measure, and further work focussing on top recommendations for each user may be beneficial.



Statistically, we find no correlation between number of reviews made by a user and accuracy of predictions made for them, although increased review count does clearly lead to a lower variance in mean error.

Conclusion

We have shown that Alternating Least Squares provides an effective solution to the problem of Collaborative Filtering in a large, sparse data environment. The more data available for model training and the higher the granularity of the recommendation scale, the better the accuracy achieved.
While the benefits of varying the rank hyperparameter were relatively clear, further work is required to fully explain the impact of the regularisation parameter. Additionally, further analysis focusing on the top recommendations for each user may uncover more about the true value of the model.