MovieLens Report

Dylon Hussain

1/5/2022

Introduction

Machine learning is a ubiquitous tool used in industry today that enables people to make informed decisions from massive amounts of data that would not be feasible without computers. This report describes a supervised machine learning algorithm that predicts user ratings for movies, that was trained with the MovieLens dataset. The following packages in this report.

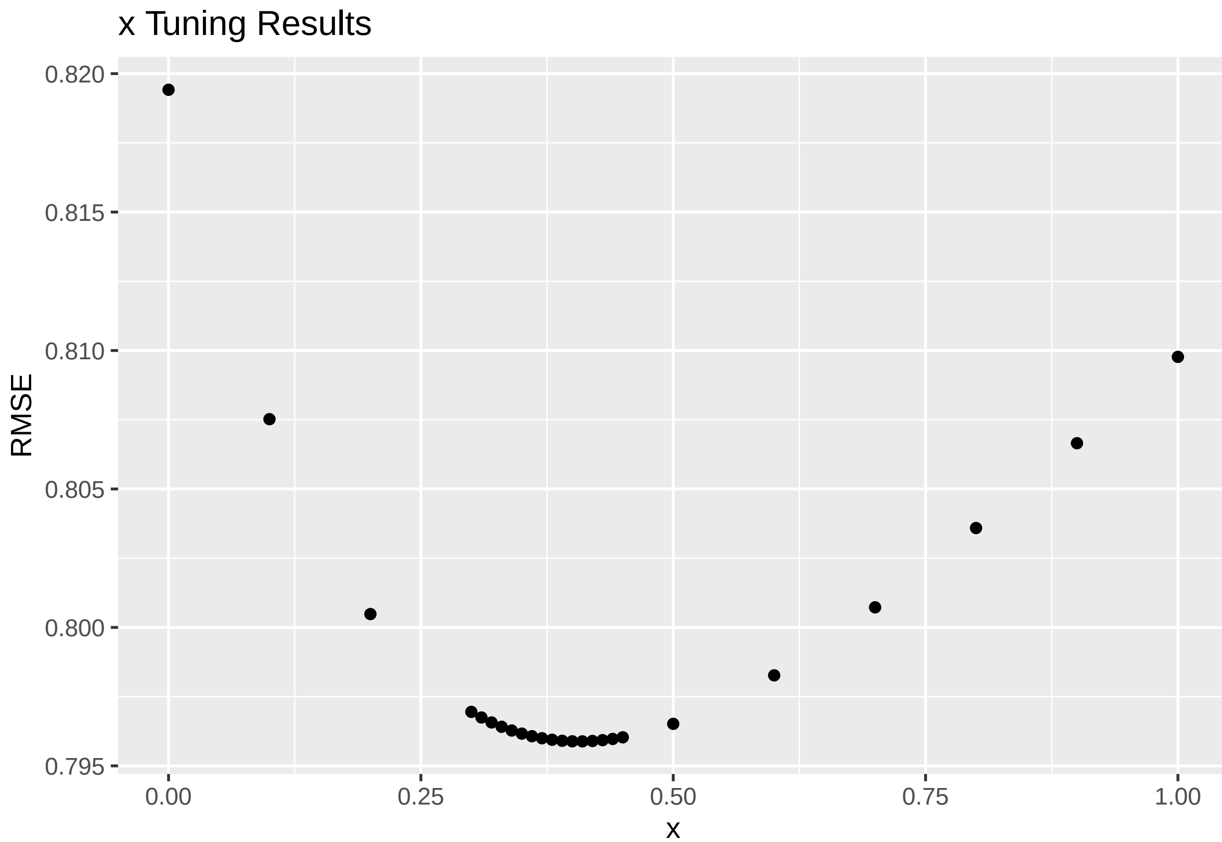
```
library(ggplot2)
library(tidyverse)
```

Methods

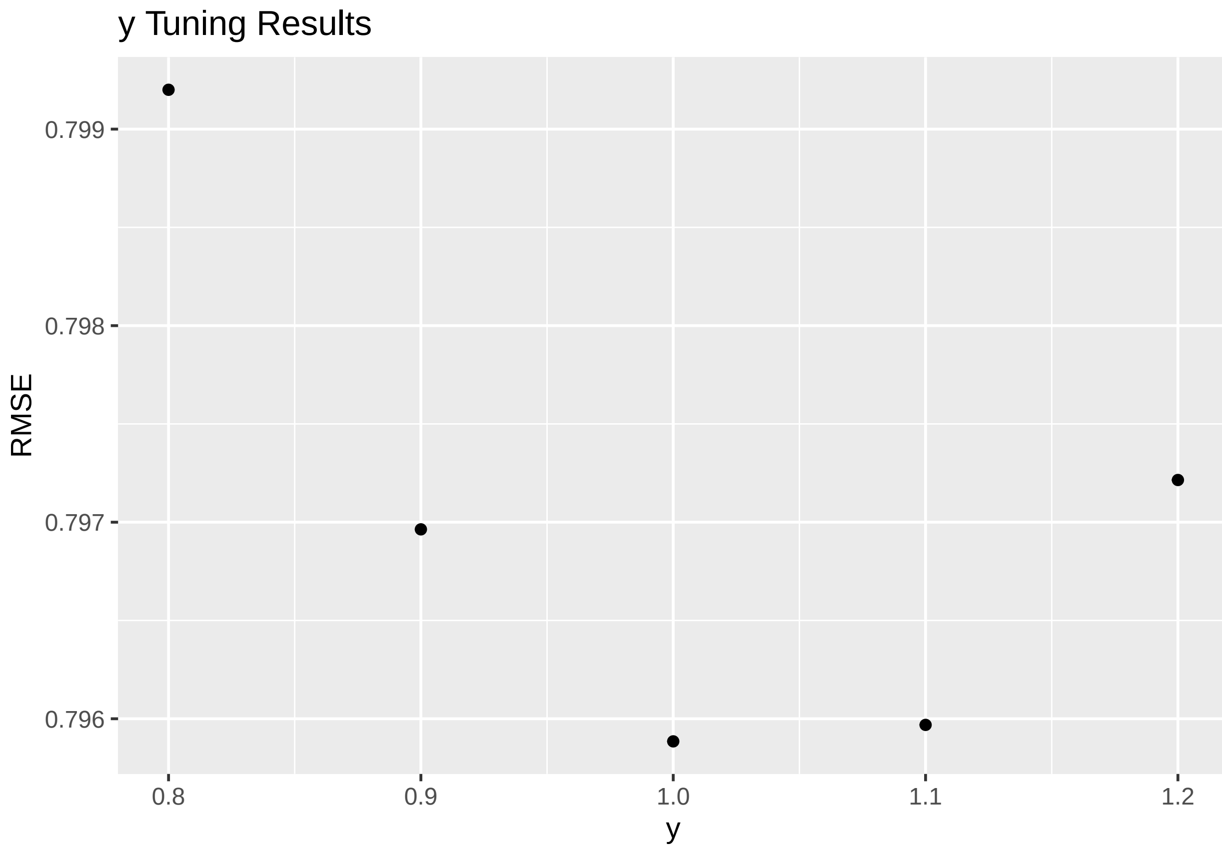
The prediction algorithm uses matrix factorization (described in Introduction to Data Science R. Irizarry) where it is assumed that $Y_{m,u,t} = \mu + b_m + b_u + bg_{u,m} + b_t + \epsilon_{u,m,t}$ where $Y_{m,u}$ is the rating user u will give movie m, μ is the average movie rating, b_m is the effect of movie m, b_u is the effect of user u, bg_{um} is the genere effect for user u and the genres of movie m, and ϵ is the independent sampling error. To estimate μ the mean of all ratings in the training set was taken, $\hat{\mu} = \bar{Y}$. b_m was similarly calculated to be the mean of ratings that movie received minus μ, $\hat{b}_m = \overline{Y_m - \mu}$. $\hat{b}_u$ was similarly calculated with the following formula $\hat{b}_u = \overline{Y_m - \mu - \hat{b_m}}$, this was done to minimize movie biases on the estimate of the user effect. $bg_{u,m}$ was more complicated to estimate: first the estimate of the effect a particular genre would have on a particular users rating was calculated by $\hat{b}_{u,g(m)} = \overline{Y_g - \mu - \hat{b_m} - \hat{b_u}}$ then it was assumed that $bg_{u,m} = f_{x,y}(b_{u,g_1}, b_{u,g_2}, ..., b_{u,g_N}, N) = y\frac{\sum b_{u,g_i(m)}}{N^x}$ where y and x are tuning parameters and N is the number of genres assigned to the movie. This function was chosen because it includes both the sum ($x = 0$) and the mean ($x = 1$) of b_{u,g_i} and is versatile in the effect of N on $bg_{u,m}$. To tune the parameters of f 10 fold cross validation was used on the training set to estimate the root mean squared error (RMSE) for a set of tuning parameters and the parameters that minimized the root mean squared error were chosen. It was then assumed that $b_t \approx \overline{Y_g - \mu - \hat{b_m} - \hat{b_u} - \hat{b}_{u,g}}$ and a loess algorithm was applied to $b(t)$. To estimate the root mean square error that would be expected for future predictions, a validation set (0.1 of the original dataset) was removed and the remainder was used to train the algorithm.

Results

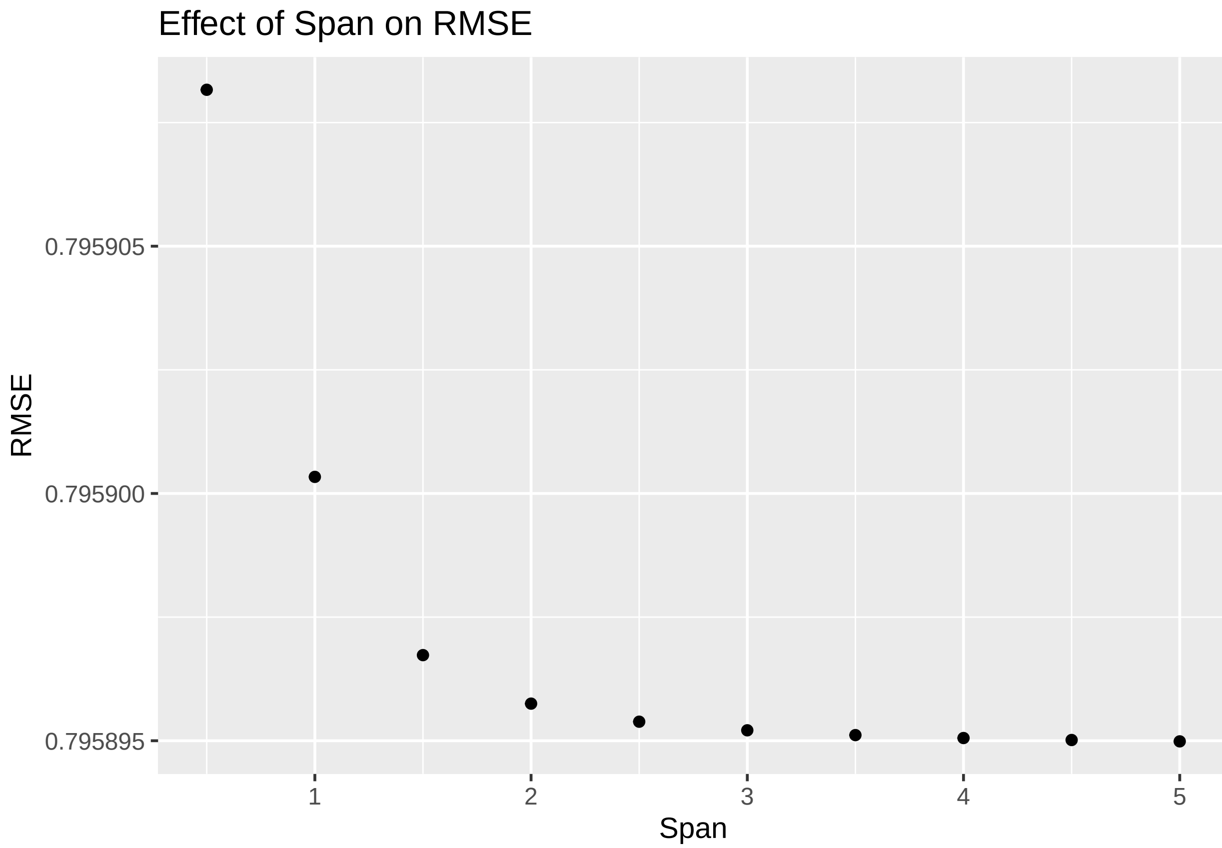
As shown in the graph below, the optimal value for x (for y = 1) was found to be .41. This is consistent with intuition because it was less than 1 (implying that a movie with multiple genres that a user dislikes would would be rated lower than a movie with just one genre the user dislikes) and greater than 0 (this would likely cause a run-away effect on movies with many genres).



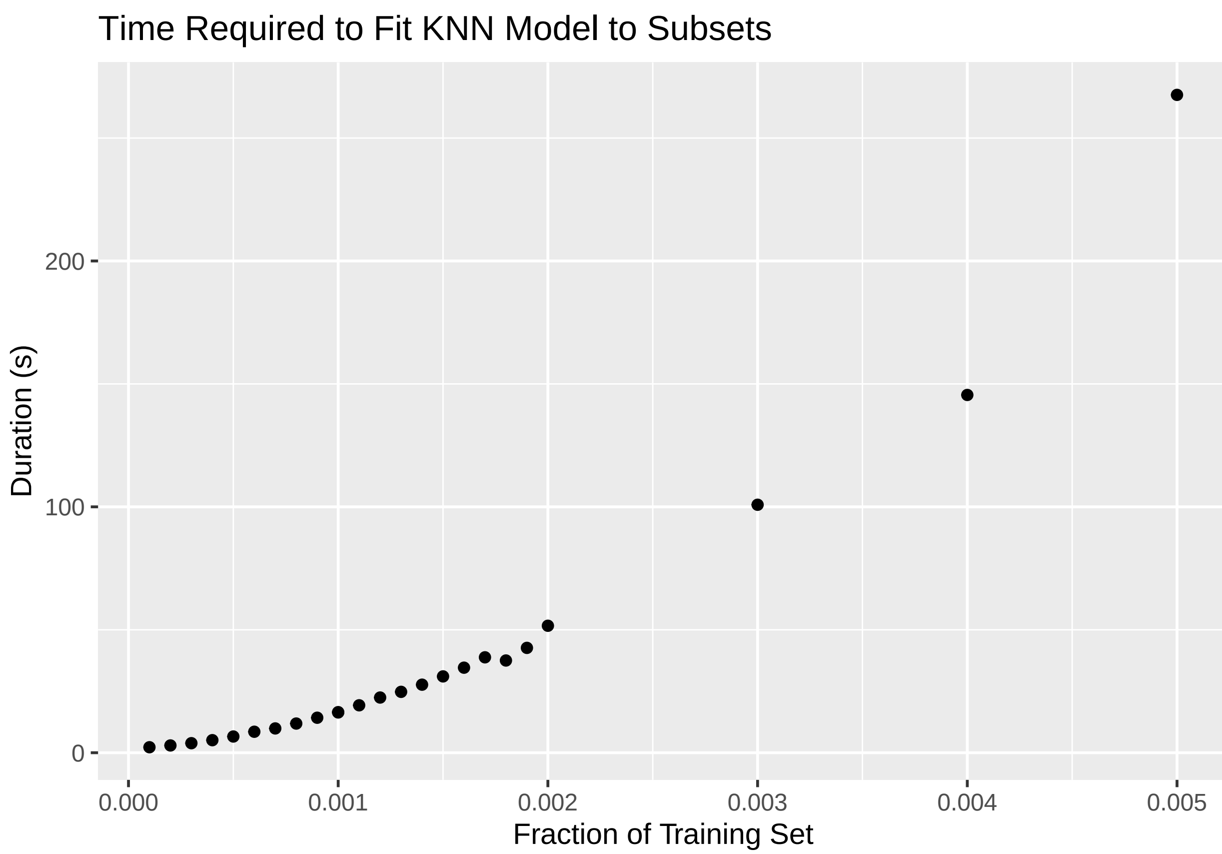
The program was structured to allow for easy minimum searching on the x-y plane but, auspiciously, the minimum y happened to be 1 as shown below. This is not surprising because it indicates that the genre effect has the same weight as the other effects.



When the loess smoothing algorithm was applied to time in an attempt to improve the prediction, the estimated root mean squared error asymptotically approaches a value greater than that which was achieved without smoothing, shown in the graph below. For this reason, it was assumed that $b_t = 0$.



A KNN model was initially implemented to explore potential means to improve the root mean squared error. As shown below, this algorithm scales exponentially with the number of observations making it prohibitively expensive to run on the entire training set. Also the root mean squared error estimation actually increased after applying the KNN algorithm. For these reasons no KNN model was used in the the final code.



The final model using $\hat{Y}_{m,u} = \mu + \hat{b}_m + \hat{b}_u + \hat{bg}_{u,m}$ was corrected so that any $\hat{Y}_{m,u} = 0.5 \forall \hat{Y}_{m,u} < 0.5$ and $\hat{Y}_{m,u} = 5 \forall \hat{Y}_{m,u} > 5$ to keep the predicted rating in the bounds of possible ratings. When this algorithm was applied to the validation set, a root mean squared error of 0.85852 was achieved.

Conclusion

A satisfactory prediction algorithm was produced using matrix decomposition with the model $Y_{m,u} = \mu + b_m + b_u + bg_{u,m} + \epsilon_{u,i}$. Future work on this algorithm would likely benefit from an implementation of a KNN algorithm that uses the entire training set. Due to computational and temporal limitations, only a small subset of the training set was used to train the KNN algorithm. Also, exploration of other smoothing algorithms might improve the root mean squared error. The these changes would increase the computational cost of running the algorithm so the benefits, in relation to the cost, must be assessed before implementing these changes. In conclusion, a satisfactory prediction algorithm was achieved using only the user, movie, genre effects.