Context Based Recommender

Prabhdeep Singh

Nathan White

**Problem**

While driving, quite a few people will listen to music. The songs that they like might depend on external factors along with the song itself. These external factors (dimensions) can be traffic conditions, location, time of day, etc... For example, on a rainy night I might like to listen to a classical song, but on sunny day I like an upbeat, party song. Since external information, the context, has an impact on how much a user likes a song, we want predict how much a user will like a song given that the user is in a certain context. How much they like a song, will be based on a rating system from 0 to 5. Since we have access to so much contextual data, and preferences can change based on this data, this is both a relevant and important problem.

**Method**

Given a user, song/item, and context, the goal is to predict the rating. The predicted rating will be equal to some baseline rating plus a context modifier. The thought behind this is that for a particular song, the user has some baseline preference toward it. For example, if a user absolutely loves a song under most circumstance then they would have a larger baseline. In this project, there were two different baseline conducted.

The first baseline was the average rating the user had given the song across all contexts. If the user had not rated the song, then baseline was the average rating the user had given to all songs across all contexts. The second baseline was found by ignoring the context and reducing the dataset to have at most 1 rating for a user, item pair. Matrix factorization was then performed on this reduced dataset. The baseline would be the rating predicted by the Matrix factorization.

The context modifier is then calculated as follows. Find the average across each context dimension, where for each context dimension given, for all users who rated the item, find the weighted average of the change in rating for that user. All context dimensions had equal weights except the no context given dimension which had ½ of the weight. This means that we assumed that the time of day has the same impact as weather, traffic conditions, etc… The change in rating was found by computing the average rating that user has given that song minus the actual rating that user gave the item in that context dimension. The weight for each user was the user-user similarity in that context.

There are then 4 predicted rating generated from this method, average baseline, model baseline, average baseline + context modifier and model baseline + context modifier.

**Implementation**

The project was implemented in Python and used our own code except a few external libraries. The custom CSR module created for previous programming assignments in the course was utilized and extended for the project. The following external libraries were used

math <https://docs.python.org/2/library/math.html>

numpy <http://www.numpy.org/>

os.path <https://docs.python.org/2/library/os.path.html>

random <https://docs.python.org/2/library/random.html>

collections <https://docs.python.org/2/library/collections.html>

**Dataset**

The data set is a public data and can be found <https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Music_InCarMusic.zip>

|  |  |
| --- | --- |
| Number of Users | 42 |
| Number of Items | 139 |
| Number of Ratings | 4012 |
| Total number of Contexts | 26 |
| Number of Dimensions | 8 |

Originally the dataset consisted of some irrelevant information such as song name/artists that were all removed. The only data we kept was the rating information as well as a key. Since the data originally had string values like “sunny”, the key was used to translate them into integer values. The user id numbers, and song id number numbers were also change to start from 1 and increment rather than starting in the 1000s (users) or sporadic numbering (items).

**Protocol**

The testing data set was created by randomly selecting a single rating for each user in the original data set. The training dataset then consisted of all of the remaining songs. For the model based prediction, the training data set was further reduced by ignoring context and selecting only unique user, item ratings. If a user had rated an item in multiple contexts then one rating was select at random.

The evaluation metrics that were selected was root mean squared error (RMSE) and mean absolute error (MAE). Lower values for both metrics means better results. Based on our predicted rating and the actual rating in the testing set, we are able to compute the error for each test item. MAE is computed by taking the mean of the absolute errors. RMSE is calculated by taking the square root of the mean of the squared errors. RMSE should be less impacted by outliers in the data set.

**Results**

In the early phases of the project we came to the realization that even though our dataset fit the mold of what we needed, a context-aware music in car dataset, the size of the dataset was less than ideal. Unfortunately, we were unable to find any relevant alternative datasets. Nevertheless, we were still confident that we would be able to draw meaningful conclusions from the project using the dataset at hand.

The following four tables display the error from the previously mentioned rating predictions:

**Average Baseline + Context Modifier**

|  |  |
| --- | --- |
| RMSE | 1.49453830577 |
| MAE | 1.20615022423 |

**Model Baseline + Context Modifier**

|  |  |
| --- | --- |
| RMSE | 2.85978928039 |
| MAE | 2.47963364955 |

**Average Baseline**

|  |  |
| --- | --- |
| RMSE | 1.47847074008 |
| MAE | 1.18298514286 |

**Model Baseline**

|  |  |
| --- | --- |
| RMSE | 2.85149591843 |
| MAE | 2.4643562751 |

**Conclusion**

As we can see from the results it seems like using the average rating as the baseline performed better than using matrix factorization. However due to sparse data set this comparison might not be completely accurate. When we compare just using the baseline to using baseline plus context modifier, we find that just using baseline seems to perform slightly better than the using the context modifier. This implies that including any context information doesn’t seem to help for this data set, and might be a result of the sparse data set that we have. The method might be improved by applying appropriate weights to each context dimension. For example, location might have greater impact than time of day. This might be done through manual experimentation with the weights, or applying other machine learning techniques to learn these weights. Overall, this project has taught us how consider different variables/data and include them to make a recommendation.

**Contributions**

Prabhdeep Singh – Contributed by researching the problem and theory on how to solve/implement the problem. Implementation of the model recommender from project 3 as well as the algorithm for the context based recommender.

Nathan White – Contributed by researching the problem. extending the CSR module with project applicable methods, and implementing methods to parse and manipulate the dataset.