CSE 272 Homework 2 Report: Recommendation Systems

Chris Liu

1. Introduction

In this assignment, I implemented a mini Amazon product recommendation system with a variety of approaches, including a baseline predictor, a predictor with random distribution assumption, four KNN-based methods, three matrix factorization techniques, and two collaborative filtering algorithms. As suggested by the assignment instruction, I utilized a Python recommendation package, Surprise (Hug, 2020), for the actual rating prediction algorithms. The best results are achieved by a baseline estimator, SVD, and SVD++. The code and results are included in https://github.com/chrisliu298/amazon-product-recommendation.

2. Method

2.1. Data preprocessing

The split `reviews_Video_Games_5` is used for this assignment. I employed Pandas (pandas development team, 2020) to read the raw data in `json` format. This approach maintains clean code and makes the data reading particularly efficient. Each row in the `json` file is considered as a row in the final data frame. Only four attributes of the raw dataset are used as features: `reviewerID`, `asin`, `unixReviewTime`, and `overall`. The dataset is split into three sets: training (64%), validation (16%), and testing (20%) using `train_test_split()` by scikit-learn (Buitinck et al., 2013). To save memory, all splits are saved locally to three `tsv` files.

2.2. Recommendation algorithms

In total of eleven models were used to predict user ratings given user id, item id, time stamp, and the ground-true rating as labels, including 1) a baseline estimator for given user and item, 2) a random rating predictor based on the normal distribution (assumed) of the data, 3) four variants collaborative filtering methods based on KNNs (basic, baseline, means, and z-scores), 4) three variants of matrix factorization methods (SVD, SVD++, and NMF), 5) Slope One, and 6) coclustering.

Due to the high-level encapsulation of the Surprise API, I do not have to interact with the underlying implementation of the used models except for the hyperparameter configuration. Given that most algorithms already had high scores during initial experiments on the validation set and the time constraint of this assignment, only default hyperparameters of each algorithm were used. For all models, they were trained and validated on the training and validation set for performance evaluation and tuning. The test set was used only once when reporting.

Seven evaluation metrics are used, including mean absolute error, root mean square error, precision, recall, F1 score, NDCG, and conversion rate.

2.3. Recommendation lists

The top-10 most relevant items for each user in both the validation and the test set are computed and saved to files. The function `top_n()` was adapted from the official tutorial.

3. Results

All validation and test metrics are reported in Table 1 and Table 2 in Appendix A. The models are consistent across both validation and training splits. Specifically, in terms of mean absolute error and root mean square error, SVD++ achieves the lowest errors. The regular SVD has the highest precision, and the baseline predictor gets the best recall, F1 score, and NDCG. The conversion rates are not reported in the tables as they are all 1's.

References

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.

Hug, N. Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174, 2020. doi: 10.21105/joss.02174. URL https://doi.org/10.21105/joss.02174.

pandas development team, T. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

A. Validation and test results

Model	MAE	RMSE	Precision	Recall	F1 score	NDCG
Normal-Predictor	1.1910	1.5547	0.8308	0.8377	0.8342	0.8132
Baseline	0.8383	1.0791	0.8753	**0.9236**	**0.8988**	**0.9775**
SVD	0.8236	1.0740	**0.8756**	0.9227	0.8986	0.9747
SVD++	**0.8112**	**1.0669**	0.8753	0.9219	0.8980	0.9667
NMF	0.9430	1.2318	0.8372	0.8539	0.8455	0.9257
KNN-basic	0.9210	1.2581	0.8654	0.8968	0.8808	0.8613
KNN-baseline	0.8651	1.1713	0.8671	0.8999	0.8832	0.9162
KNN-means	0.8498	1.1922	0.8509	0.8789	0.8647	0.9124
KNN-zscore	0.8495	1.2025	0.8483	0.8750	0.8614	0.9114
SlopeOne	0.8974	1.2474	0.8458	0.8683	0.8569	0.9010
Coclustering	0.8175	1.1529	0.8487	0.8736	0.8609	0.9290

Table 1. Validation metrics of the 11 implemented algorithms.

Model	MAE	RMSE	Precision	Recall	F1 score	NDCG
Normal-Predictor	1.1890	1.5518	0.8399	0.8448	0.8424	0.8200
Baseline	0.8376	1.0752	0.8763	**0.9309**	**0.9028**	**0.9941**
SVD	0.8248	1.0724	**0.8763**	0.9296	0.9022	0.9765
SVD++	**0.8125**	**1.0664**	0.8761	0.9279	0.9013	0.9719
NMF	0.9406	1.2288	0.8394	0.8548	0.8470	0.9401
KNN-basic	0.9125	1.2486	0.8688	0.9038	0.8860	0.8691
KNN-baseline	0.8602	1.1649	0.8700	0.9066	0.8879	0.9243
KNN-means	0.8481	1.1903	0.8534	0.8837	0.8683	0.9178
KNN-zscore	0.8467	1.1991	0.8519	0.8813	0.8663	0.9159
SlopeOne	0.8939	1.2433	0.8500	0.8736	0.8616	0.9066
Coclustering	0.8208	1.1571	0.8518	0.8764	0.8639	0.9325

Table 2. Test metrics of the 11 implemented algorithms.