On the discriminative power of Hyper-parameters in Cross-Validation and how to choose them

Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Claudio Pomo
Polytechnic University of Bari
Bari, Italy
firstname.lastname@poliba.it

Azzurra Ragone*
Independent Researcher
Milan, Italy
azzurra.ragone@gmail.com

ABSTRACT

Hyper-parameters tuning is a crucial task to make a model perform at its best. However, despite the well-established methodologies, some aspects of the tuning remain unexplored. As an example, it may affect not just accuracy but also novelty as well as it may depend on the adopted dataset. Moreover, sometimes it could be sufficient to concentrate on a single parameter only (or a few of them) instead of their overall set. In this paper we report on our investigation on hyper-parameters tuning by performing an extensive 10-Folds Cross-Validation on MovieLens and Amazon Movies for three well-known baselines: User-kNN, Item-kNN, BPR-MF. We adopted a grid search strategy considering approximately 15 values for each parameter, and we then evaluated each combination of parameters in terms of accuracy and novelty. We investigated the discriminative power of *nDCG, Precision, Recall, MRR, EFD, EPC*, and, finally, we analyzed the role of parameters on model evaluation for Cross-Validation.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Recommender systems, Parameter tuning, Discriminative power

ACM Reference Format:
Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Claudio Pomo and Azzurra Ragone. 2019. On the discriminative power of Hyper-parameters in Cross-Validation and how to choose them. In *Thirteenth ACM Conference on Recommender Systems (RecSys '19), September 16–20, 2019, Copenhagen, Denmark*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3298689.3347010

1 INTRODUCTION

Recommenders Systems now play an important role in the lives of users. These systems avoid massive data overwhelm users and help them in finding a path to relevant information [35]. To enhance the expressiveness of the models or to improve the learning phase, these models can be equipped with a special class of parameters, named hyperparameters. Since many recommender systems come with one or more hyperparameters, the goodness of the system depends on how these parameters are selected. Although several strategies are available [9, 14, 23, 29, 39], the choice of the metric to evaluate them is not manifest. Are there particular measures that are well-suited for hyper-parameters tuning? Are the changes in the metric's values significant or they are fundamentally originated by chance?

Up to the Netflix prize [4], the research community widely considered the recommendation problem as a rating prediction task [42, 46]. Consequently, the optimization goal was the minimization of the prediction error [20, 38]. However, in real recommender systems applications, only a small subset of relevant items are provided to users [20]. Indeed, several studies acknowledged that rating prediction optimization was not able to produce the optimal top-N recommendation lists [32]. Recommendation problem was hence re-defined as a top-N recommendation task [16], in which the optimization goal shifted to items ranking.

From this new perspective, many Information Retrieval metrics came to play to evaluate recommender systems. After decades, accuracy is still broadly considered as the key element in evaluation. Nonetheless, new dimensions as novelty and diversity of recommendation [12, 21, 45] became progressively important. In compliance with the purpose of the system, accuracy, novelty, and diversity metrics are used both to evaluate the recommender and tune the hyperparameters. Although recommender systems are evaluated using an online or offline setting, hyperparameters are usually tuned in an offline setting [38]. In this setting, to evaluate the competing models (or hyperparameters candidate values), past users interactions are split adopting distinct strategies like Hold-Out [15, 27, 43] and k-Folds Cross-Validation (CV) [17, 24, 26]. In the former, the training set is split into two further sets: training, and validation set. In the latter, data is divided into k sets, retaining in rotation one of them as the validation set and the remaining ones as the training set.

The choice of hyperparameters values to test has also been deeply investigated. Among all the most adopted techniques are Random [5, 6, 33], Bayesian optimization [10, 40], and Grid Search [7, 19, 22]. Nevertheless, even though a recommender system's hyperparameter tuning is wisely designed to achieve more robust results, some aspects need further investigation. For instance, the behaviour of different metrics when varying the folds is still an almost unexplored field. As an example, if the metric is not able to capture significant differences when different values of parameters are set, that metric is not the ideal one to tune hyperparameters. A recent study [44] depicted a new interesting methodology to establish

*Authors are listed in alphabetical order. Corresponding author: V.W. Anelli vitowalter.anelli@poliba.it

RecSys '19, September 16–20, 2019, Copenhagen, Denmark

ACM ISBN 978-1-4503-6243-6/19/09…$15.00
https://doi.org/10.1145/3298689.3347010

whether a metric is discriminative, robust and the authors also performed a metric-to-metric comparison. Even though the authors designed it to measure the robustness of metrics to changes in the cut-off (previously explored in Information Retrieval in [28]), the overall procedure, along with the Discriminative Power and Robustness definition inspired us to propose a new procedure to evaluate metrics and study the metric-hyperparameter combined behaviour.

Our experiments on two well-known datasets show that *Precision* and *normalised Discounted Cumulative Gain* are the most discriminative accuracy metrics to use in model selection. We additionally show that also considering the standard deviation variations over the folds these metrics still behave better than *Recall* and *Mean Reciprocal Rank*. We show that *Expected Free Discovery* and *Expected Popularity Complement* metrics are discriminative and are significantly influenced by changes in hyperparameters values. Finally, if more than one hyperparameters have to be set, it is possible to investigate how changes in the specific parameters affect the considered metric.

Main contributions of the paper are: i) a study on the discriminative power of accuracy and novelty metrics for the k-Folds Cross-Validation hyperparameter tuning for three well-known collaborative models; ii) a procedure for models with more than one parameter to check if variations in one of the parameters were particularly relevant for accuracy of recommendation iii) a study on the impact of "Number of latent factors", "Number of iterations", and "learning rate" on accuracy of recommendation for BPR-MF. The remainder of this paper is structured as follows: Section 2 introduces the setting of our experiments describing the adopted methodologies; then we focus on the discriminative power of metrics and their variations across folds; in Section 5 we study separately the hyper-parameters of BPR-MF. Conclusions close the paper.

2 EXPERIMENTAL SETTINGS

Discriminative Power (DP) is a metric proposed by [44] to measure the discriminative power of an evaluation metric over a set of competing algorithms. The procedure was originally presented by [36] in 2006. Given a metric, a dataset, and a set of recommender systems, the authors perform a statistical test considering all the possible system pairs. The obtained *p-values* can be sorted by decreasing value and plotted. The resulting curve is the *p-values* curve of the considered metric. Analogously, the corresponding *p-values* curve can be obtained for each considered metric. Since lower values of *p-values* denote statistical differences between system pairs, the metric with the lower area under the curve can be considered as the most discriminative. DP consists of the summation of all the *p-values* for a given metric, and it can be considered as an approximation of the area under the curve for that metric. Interestingly, in [44], the authors extend the idea of competing algorithms to a set composed of instances of the same algorithm considering different cut-off values. However, this idea can be further extended to consider a set of instances of the same algorithm considering different hyper-parameter values. This idea is the starting point of our work.

Dataset	Users	Items	Ratings	Sparsity
MovieLens-1M	6040	3706	1000209	95.53%
Amazon Movies	16141	111537	858314	99.95%

Table 1: Datasets statistics

Datasets. To conduct our study we exploited two different datasets in the Movies domain: Amazon Movies[1] and MovieLens-1M. Both datasets contain explicit ratings on a 1-5 scale. For Amazon Movies dataset we removed users with less 20 interactions and items with less than 25 votes. Then we sampled the resulting dataset to generate a subset that preserves the original distribution of data [30, 31]. Experiments were conducted on a dedicated server equipped with an *Intel Xenon* with *32 cores*, and *256GB RAM memory*. The sampling step was necessary to perform experiments in a reasonable time. The characteristics of datasets are reported in Table 1.

Over the years, several splitting methodologies have been proposed [2, 3, 38]. We decided to split our data in training and test set using a temporal Hold-Out splitting [38]. For each user, the first 80% of the interactions are considered as the training set, whereas the remaining 20% is used as the test set. The training set is further divided using a 10-Folds Cross-Validation.

Evaluation protocol. Offline evaluation in recommender systems is a well-studied field. To evaluate the approaches we decided to use "All Unrated Items" protocol [41], in which the set of candidate items for user u is composed of all items i not rated in u's training set. Many metrics make use of binary relevance. Since we use datasets with explicit ratings, a relevance threshold τ [11] should be set to establish whether the items in each user's test set are relevant or not. We set τ to 4 for both datasets: only items with a rate above τ are considered as relevant during evaluation.

Algorithms. To study the hyper-parameters influence on the discriminative power of metrics we decided to consider two distinct families of algorithms: Neighborhood models, and Matrix Factorization models. For the former, we considered both the *User-based* and *Item-based* scheme [1, 37]. For the latter, *BPR-MF* [34] was considered. In BPR-MF the classic MF model is optimized adopting the Bayesian Personalized Ranking Criterion, a well-known pairwise *"learning to rank"* algorithm.

Metrics. We decided to study the discriminative power of some widely used metrics along two dimensions: Accuracy and Novelty. In order to evaluate the accuracy of the algorithms, we measured *normalised Discount Cumulative Gain@N* (*nDCG@N*) [25], *Precision* (*Prec@N*), *Recall* (*Rec@N*), and *Mean Reciprocal Rank* (*MRR@N*). To evaluate Novelty, we decided to measure *Expected Free Discovery* (*EFD@N*) [45], and *Expected Popularity Complement* (*EPC@N*) [12]. These metrics were computed per user to perform the *Student's t* statistical test. The metrics values and the overall mean was computed using the RankSys framework [13].

Grid Search. To study the DP of the metrics for the different algorithms, we conducted a grid search exploration. This procedure is very common for hyperparameters tuning. However, based on how much exhaustive this search is, the operation can be time and space consuming. For this reason, we needed to determine the boundaries of this grid. The number of hyper-parameters we decided to explore was 1 for Neighborhood models (the *number of Neighbors*), and 3 for Matrix Factorization (*latent factors, iterations, learning rate*). For Matrix Factorization we assumed user and item regularization to be dependent on learning rate, with a scale factor of respectively 1/20 and 1/200. We computed the values of the grid exploiting an exponential function with base 2 [8, 18]. To

[1] https://snap.stanford.edu/data/web-Movies.html

determine the evolution of latent factors, we used as an exponent for our function values within the range *[3.321,10.821]* with a step of 0.5. The same procedure and the same step have been used to define all the hyper-parameters values. The difference basically consists of the considered range, chosen coherently with literature. For the number of iterations, we considered a range of *[0,7]* as the exponent. Finally, for the learning rate the exponent is in the range *[-2.3219,-16.3219]*. This choice led to learning values in the range *[0.00001220726897, 0,2000038948]* considering 5 orders of magnitude. Summing up, for Matrix Factorization we had a grid of dimensions $15 \times 15 \times 14$. This grid generates 3150 different configurations of hyperparameters for each fold. Since the exploration on Amazon would have required several months we extracted a sub grid ($5 \times 3 \times 3$) with the best value for each hyper-parameter (considering *nDCG@10*) ± 2 neighbors in the grid. The overall hyper-parameters values are depicted in Table 2.

Latent factors	Iterations	Learning rate	Nearest neighbors
10	1	0.200 003 89	10
14	2	0.100 001 95	14
20	3	0.050 000 97	20
28	4	0.025 000 49	28
40	6	0.012 500 24	40
57	8	0.006 250 12	57
80	11	0.003 125 06	80
113	16	0.001 562 53	113
160	23	0.000 781 27	160
226	32	0.000 390 63	226
320	45	0.000 195 32	320
452	64	0.000 097 66	452
640	91	0.000 048 83	640
905	128	0.000 024 41	905
1809		0.000 012 21	1809

Table 2: Hyper-parameters grid

3 DISCRIMINATIVE POWER OF METRICS ON HYPERPARAMETERS

The discriminative power (DP) of each metric using hyper parameters depicted in Table 2 can be studied exploiting the same strategy proposed in [44]. We compute the *p-value* across all folds in our k-fold cross-validation setting. For each algorithm, we generate all possible combinations of pairs of hyperparameters and we randomly take 25 combinations. Thus, for these pairs, we compute the *p-values* for each fold. The *p-values* of the paired statistical tests are sorted by decreasing value and the corresponding values are averaged over the folds. For memory-based algorithms, the pairs correspond to pairs of systems with a different number of nearest neighbors. Also for BPR-MF, these pairs are pairs of systems.

MovieLens	EFD@N	EPC@N	MRR@N	nDCG@N	Prec@N	Rec@N
Item-kNN	1.884	1.840	1.766	**1.710**	1.855	2.385
User-kNN	1.688	1.576	2.196	**1.665**	1.869	1.993
BPR-MF	0.875	0.776	0.803	0.594	**0.585**	1.314

Amazon	EFD@N	EPC@N	MRR@N	nDCG@N	Prec@N	Rec@N
Item-kNN	0.188	0.188	0.188	0.213	**0.161**	0.281
User-kNN	4.738	5.717	6.646	6.310	**5.474**	6.281
BPR-MF	3.771	3.841	4.518	3.989	**3.289**	4.434

Table 3: Metrics Discriminative Power.

However, each system is defined by a triple: *Latent Factors, Iterations, Learning Rate*. The sorted and averaged values can be plotted and they correspond to the averaged *p-values* curve. For each Accuracy and Novelty metrics, these plots are depicted in Figure 1. For each metric, the corresponding curve gives us an intuition on how that metric is discriminative with respect to the evolution of hyperparameters. Table 3 summarises the DP values of the metrics respectively on Movielens and Amazon. We used **bold** to highlight the best-performing metric and underline to highlight the worst one. On Movielens the trends of accuracy metrics seem to be clear: *nDCG@N* always performs better than the others while *MRR@N* and *Rec@N* seem to be the worst metrics. It is interesting to notice that Novelty metrics achieve good DP values. This could be a signal that changes in parameters values lead to significant differences in terms of Novelty. On Amazon, the Best Accuracy metric is definitely *Prec@N*, while also in this case, for User-kNN and BPR-MF the worst metric is *MRR@N*. In [44], it is suggested that this kind of behaviour can be due to the complexity of the metric. We agree with the authors and we reckon that this could be also due to some characteristics of the dataset, like the average number of rating per user (lower than Movielens's), the sparsity of the dataset (higher than Movielens's), and the low average number of ratings per item.

4 METRICS CONFIDENCE

In the previous section, we compared the Discriminative Power of different metrics and we found that *nDCG@N* and *Prec@N* are two good choices to select the best hyper-parameters value for Neighborhood-based models and BPR-MF. In details, we consider the best hyper-parameters value as the value which ensures the best performance with respect to the most discriminative metric. However, since we computed the averaged *p-values* curves across 10 Folds, and hence the averaged DP, it is still possible that these metrics could be much less discriminative on some folds. If this is true, the choice of an elected metric to conduct hyper-parameters learning could be argued. For this reason, now we study the variations across different folds of the metrics *p-values*. Given the sorted *p-values* for each fold, we computed the standard deviation for each ordered pair across folds. These values can be exploited to define two additional *p-values* curves, which represent reasonable boundaries of *p-values* for each metric. Moreover, it is possible to compute the corresponding DP values, for each metric ± the standard deviation. This could give us an intuition of the metric's DP robustness across folds. Table 4 shows the results for respectively Amazon and Movielens dataset. On Amazon, we may notice the good performance of *Prec@N* with Item-kNN model. Although Amazon is a very sparse dataset with a large number of items, *Prec@N* is able to capture significant differences between similar models with a different number of neighbours. Moreover, if we observe the DP value considering the standard deviation, this extreme value is still very close to the DP of the worst metric. This behaviour is present also on Movielens, for both Item-kNN and BPR-MF. We considered the worst scenario in which we added the standard deviation value to each pair in comparison.

MovieLens	Best metric	Worst metric	Best avg	Best + Std Dev	Worst avg
Item-kNN	nDCG@N	Recall@N	1.710	2.940	2.385
User-kNN	nDCG@N	MRR@N	1.665	2.958	1.704
BPR-MF	Prec@N	Recall@N	0.585	1.126	1.314

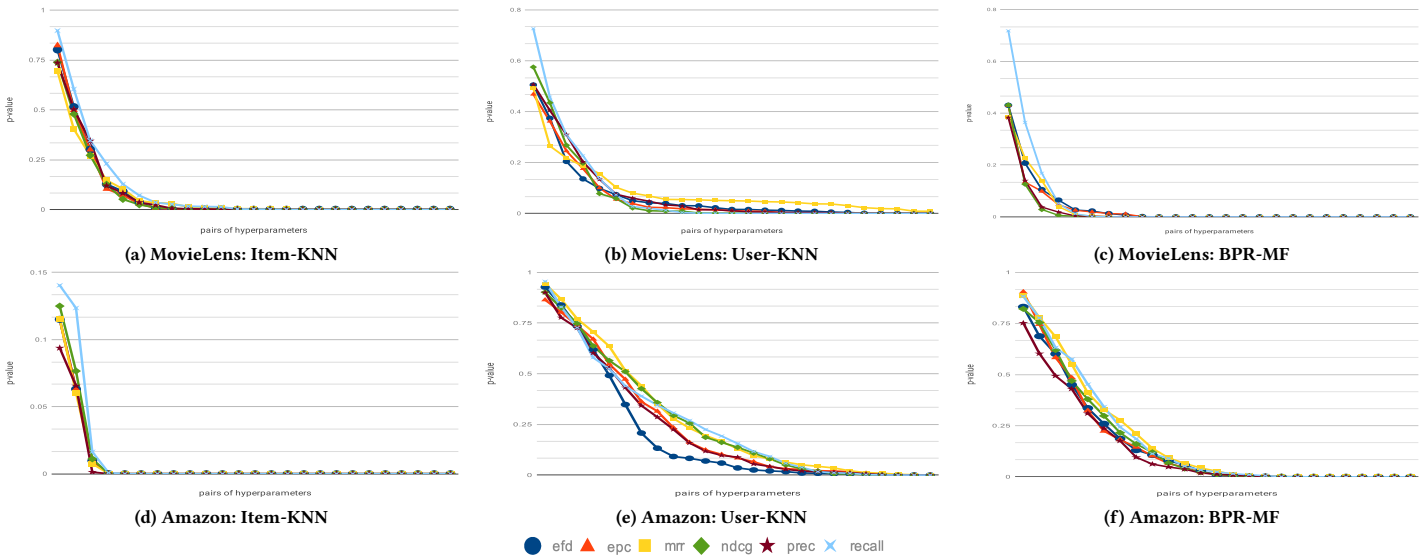
Amazon	Best metric	Worst metric	Best avg	Best + Std Dev	Worst avg
Item-kNN	Prec@N	Recall@N	0.161	0.287	0.281
User-kNN	Prec@N	MRR@N	5.474	7.410	6.646
BPR-MF	Prec@N	MRR@N	3.289	4.689	4.518

Table 4: Metrics Discriminative Power deviation.

However, it seems clear that the metrics chosen with the previous procedure show good performance across the different folds.

5 DOMINANT HYPERPARAMETER

In the previous section, we mainly focused on the Discriminative Power of the metrics to find out the best metric for hyper-parameter tuning taking into account the recommendation model and the considered dataset. In this section, we fix the metric to study the specific hyper-parameters. Differently from nearest neighbors models, in BPR-MF we decided to explore three different hyper-parameters:



(a) MovieLens: Item-KNN (b) MovieLens: User-KNN (c) MovieLens: BPR-MF

(d) Amazon: Item-KNN (e) Amazon: User-KNN (f) Amazon: BPR-MF

Figure 1: Discriminative Power of Accuracy and Novelty metrics

	MovieLens															Amazon				
Latent factors	10	14	20	28	40	56	80	113	160	226	320	452	640	905	1809	113	160	226	320	452
DP	0.703	0.664	0.602	0.512	0.592	0.707	0.575	0.535	0.635	0.639	0.572	0.499	0.303	0.565	**0.290**	2.775	2.681	2.348	**2.127**	2.542
Iterations	1	2	3	4	6	8	11	16	23	32	45	64	91	128		64	92	128		
DP	1.633	0.796	**0.707**	0.939	0.978	0.781	1.037	1.338	1.045	1.210	1.222	1.196	1.008	0.939		4.106	3.299	**2.357**		
Learning rate	0.20000	0.10000	0.05000	0.02500	0.01250	0.00625	0.00312	0.00156	0.00078	0.00039	0.00019	0.00009	0.00004	0.00002	0.00001	0.20000	0.10000	0.05000		
DP	0.765	**0.702**	1.918	2.667	2.131	2.429	2.811	1.552	1.044	0.803	1.083	1.225	0.950	2.335	3.548	3.941	**3.423**	4.803		

Table 5: DP w.r.t. hyper-parameters evolution

number of latent factors, number of iterations, learning rate. Usually, during the tuning phase, these dimensions are handled in the same way. Indeed, irrespective of the adopted search strategy, all the hyper-parameters are equally important and should be explored. However, it is straightforward that one or more hyper-parameters changes could influence more the accuracy of the provided recommendation list. This led us to pose two additional research questions:

- Is there one or more hyper-parameters that affect more the accuracy of recommendations?
- Could be established a procedure to check if different values for a specific hyper-parameter can lead to significant differences?

To answer these research questions, we analysed the three hyper-parameters of BPR-MF separately. In details, for a certain parameter, we want to define a procedure to check if different values of that hyper-parameter lead to systems which show significant differences in accuracy of recommendation. Let us suppose we fix the metric and the number of latent factors: we still have two other parameters that can vary. We computed all the possible combinations of the remaining hyper-parameters. From the set of combinations, we randomly chose 25 pairs of combinations. We recall that a pair of combinations corresponds to a pair of systems that share the number of latent factors, and differ in the number of iterations and learning rate. Now we compute the *p-values* of all these pairs and we order them by decreasing value. These values correspond to a *p-values* curve which is peculiar for the considered metric and number of latent factors. Consequently, for the curve, the corresponding Discriminative Power can be computed. This procedure can be repeated to analyse the discriminative power of various values of latent factors parameter. Thus, the whole procedure can be repeated to analyse the remaining hyper-parameters. The results of this study is depicted in Tables 5. We used **bold** to highlight the best DP value for each hyper-parameter analysis. As suggested in [44], the results between the two tables are not comparable. However, for both datasets, the *number of latent factors* seems to be the dimension on which variations in hyper-parameter value lead to significant differences in recommendation accuracy. Moreover, for Movielens dataset, the DP value is much lower than the best values for *"Number of iterations"* and *"Learning rate"* hyper-parameter analysis. This suggests that *"Number of latent factors"* is dominant with respect to the other hyper-parameters. *"Learning rate"* dimension shows a different behaviour on the two datasets: on Movielens it shows big variations in terms of DP values, while on Amazon it shows oscillating performance. Finally, DP values confirm that conducting the study on the sub-grid was a reasonable choice.

6 CONCLUSIONS AND FUTURE WORK

We explored the behavior of accuracy and novelty metrics in response to changes in hyper-parameters values. We found that nDCG@N and Precision@N represent a good choice for hyper-parameters tuning for NN-based models and BPR-MF. Novelty metrics also showed to be sensitive to changes in hyper-parameters values. We proposed a general procedure for models with more than one parameter to check if variations in one of the parameters were particularly relevant for accuracy of recommendation. We explored the BPR-MF hyper-parameters and we found that the number of latent factors is dominant w.r.t. the learning rate and the number of iterations. Following this research direction, we want to explore other well-known algorithms and datasets to check if our findings can be further generalized.

REFERENCES

[1] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta. 2019. The importance of being dissimilar in Recommendation. to appear.

[2] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta. 2019. Local Popularity and Time in top-N Recommendation. In *Advances in Information Retrieval - 41st European Conf. on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proc., Part I*. 861–868.

[3] Alejandro Bellogín and Pablo Sánchez. 2017. Revisiting Neighbourhood-Based Recommenders For Temporal Scenarios. In *Proc. of the 1st Workshop on Temporal Reasoning in Recommender Systems co-located with 11th Int. Conf. on Recommender Systems (RecSys 2017), Como, Italy, August 27-31, 2017*. 40–44.

[4] James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proc. of KDD cup and workshop*, Vol. 2007. New York, NY, USA., 35.

[5] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems 24: 25th Annual Conf. on Neural Information Processing Systems 2011. Proc. of a meeting held 12-14 December 2011, Granada, Spain*. 2546–2554.

[6] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* 13 (2012), 281–305.

[7] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. 2015. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery* 8, 1 (2015), 014008.

[8] James Bergstra, Dan Yamins, and David D Cox. 2013. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proc. of the 12th Python in science conf.* Citeseer, 13–20.

[9] Alex Beutel, Ed Huai-hsin Chi, Zhiyuan Cheng, Hubert Pham, and John R. Anderson. 2017. Beyond Globally Optimal: Focused Learning for Improved Recommendations. In *Proc. of the 26th Int. Conf. on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*. 203–212.

[10] Eric Brochu, Vlad M. Cora, and Nando de Freitas. 2010. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *CoRR* abs/1012.2599 (2010).

[11] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Model. User-Adapt. Interact.* 24, 1-2 (2014), 67–119.

[12] Pablo Castells, Neil J. Hurley, and Saul Vargas. 2015. Novelty and Diversity in Recommender Systems. See [35], 881–918.

[13] Pablo Castells, Neil J. Hurley, and Saul Vargas. 2015. *Novelty and Diversity in Recommender Systems*. Springer US, Boston, MA, 881–918.

[14] Simon Chan, Philip C. Treleaven, and Licia Capra. 2013. Continuous hyperparameter optimization for large-scale recommender systems. In *Proc. of the 2013 IEEE Int. Conf. on Big Data, 6-9 October 2013, Santa Clara, CA, USA*. 350–358.

[15] Paolo Cremonesi, Franca Garzotto, Sara Negro, Alessandro Vittorio Papadopoulos, and Roberto Turrin. 2011. Looking for "Good" Recommendations: A Comparative Evaluation of Recommender Systems. In *Human-Computer Interaction - INTERACT 2011 - 13th IFIP TC 13 Int. Conf., Lisbon, Portugal, September 5-9, 2011, Proc., Part III*. 152–168.

[16] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of the 2010 ACM Conf. on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*. 39–46.

[17] Paolo Cremonesi, Roberto Turrin, Eugenio Lentini, and Matteo Matteucci. 2008. An evaluation methodology for collaborative recommender systems. In *2008 Int. Conf. on Automated Solutions for Cross Media Content and Multi-Channel Distribution*. IEEE, 224–231.

[18] Bruno Feres de Souza, André Carlos Ponce de Leon Ferreira de Carvalho, Rodrigo Calvo, and Renato Porfirio Ishii. 2006. Multiclass SVM Model Selection Using Particle Swarm Optimization. In *6th Int. Conf. on Hybrid Intelligent Systems (HIS 2006), 13-15 December 2006, Auckland, New Zealand*. 31.

[19] Frauke Friedrichs and Christian Igel. 2005. Evolutionary tuning of multiple SVM parameters. *Neurocomputing* 64 (2005), 107–117.

[20] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1 (2004), 5–53.

[21] Neil Hurley and Mi Zhang. 2011. Novelty and Diversity in Top-N Recommendation – Analysis and Evaluation. *ACM Trans. Internet Technol.* 10 (March 2011), 14:1–14:30. Issue 4.

[22] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2014. An Efficient Approach for Assessing Hyperparameter Importance. In *Proc. of the 31th Int. Conf. on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014 (JMLR Workshop and Conf. Proc.)*, Vol. 32. JMLR.org, 754–762.

[23] Frank Hutter, Jörg Lücke, and Lars Schmidt-Thieme. 2015. Beyond Manual Tuning of Hyperparameters. *KI* 29, 4 (2015), 329–337.

[24] Michael Jahrer, Andreas Töscher, and Robert A. Legenstein. 2010. Combining predictions for accurate recommender systems. In *Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, Bharat Rao, Balaji Krishnapuram, Andrew Tomkins, and Qiang Yang (Eds.). ACM, 693–702.

[25] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.

[26] Ron Kohavi. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proc. of the Fourteenth Int. Joint Conf. on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*. Morgan Kaufmann, 1137–1145.

[27] Nathan Nan Liu and Qiang Yang. 2008. EigenRank: a ranking-oriented approach to collaborative filtering. In *Proc. of the 31st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong (Eds.). ACM, 83–90.

[28] Xiaolu Lu, Alistair Moffat, and J. Shane Culpepper. 2016. The effect of pooling and evaluation depth on IR metrics. *Inf. Retr. Journal* 19, 4 (2016), 416–445.

[29] Xin Luo, MengChu Zhou, Shuai Li, Zhu-Hong You, Yunni Xia, and Qingsheng Zhu. 2016. A Nonnegative Latent Factor Model for Large-Scale Sparse Matrices in Recommender Systems via Alternating Direction Method. *IEEE Trans. Neural Netw. Learning Syst.* 27, 3 (2016), 579–592.

[30] Gurmeet Singh Manku, Sridhar Rajagopalan, and Bruce G. Lindsay. 1999. Random Sampling Techniques for Space Efficient Online Computation of Order Statistics of Large Datasets. In *SIGMOD 1999, Proc. ACM SIGMOD Int. Conf. on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA*. 251–262.

[31] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *22nd Int. World Wide Web Conf., WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. 897–908.

[32] Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI EA '06)*. 1097–1101.

[33] Nicolas Pinto, David Doukhan, James J. DiCarlo, and David D. Cox. 2009. A High-Throughput Screening Approach to Discovering Good Forms of Biologically Inspired Visual Representation. *PLoS Computational Biology* 5, 11 (2009).

[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proc. of the Twenty-Fifth Conf. on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*. 452–461.

[35] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2015. *Recommender Systems Handbook*. Springer.

[36] Tetsuya Sakai. 2006. Evaluating evaluation metrics based on the bootstrap. In *SIGIR 2006: Proc. of the 29th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*. 525–532.

[37] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2000. Analysis of recommendation algorithms for e-commerce. In *EC*. 158–167.

[38] Guy Shani and Asela Gunawardana. 2011. Evaluating Recommendation Systems. In *Recommender Systems Handbook*. 257–297.

[39] Michael R. Smith, Logan Mitchell, Christophe G. Giraud-Carrier, and Tony R. Martinez. 2014. Recommending Learning Algorithms and Their Associated Hyperparameters. In *Proc. of the Int. Workshop on Meta-learning and Algorithm Selection co-located with 21st European Conf. on Artificial Intelligence, MetaSel@ECAI 2014, Prague, Czech Republic, August 19, 2014*. 39–40.

[40] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25: 26th Annual Conf. on Neural Information Processing Systems 2012. December 3-6, 2012, Lake Tahoe, Nevada, United States*. 2960–2968.

[41] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Seventh ACM Conf. on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, Qiang Yang, Irwin King, Qing Li, Pearl Pu, and George Karypis (Eds.). ACM, 213–220.

[42] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. 2008. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *Proc. of the 2008 ACM Conf. on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008*. 267–274.

[43] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. 2009. Scalable Collaborative Filtering Approaches for Large Recommender Systems. *Journal of Machine Learning Research* 10 (2009), 623–656.

[44] Daniel Valcarce, Alejandro Bellogín, Javier Parapar, and Pablo Castells. 2018. On the Robustness and Discriminative Power of Information Retrieval Metrics for top-N Recommendation. In *Proc. of the 12th ACM Conf. on Recommender Systems (RecSys '18)*. ACM, New York, NY, USA, 260–268.

[45] Saul Vargas and Pablo Castells. 2011. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proc. of the 2011 ACM Conf. on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*. 109–116.

[46] Yunhong Zhou, Dennis M. Wilkinson, Robert Schreiber, and Rong Pan. 2008. Large-Scale Parallel Collaborative Filtering for the Netflix Prize. In *Algorithmic Aspects in Information and Management, 4th Int. Conf., AAIM 2008, Shanghai, China, June 23-25, 2008. Proc.* 337–348.