An Evaluation of Recommendation Algorithms for Tourist Attractions

Anderson Feitosa Júnior
Informatics Coordination
Federal Institute of Alagoas
Maceió, Alagoas, Brasil
amfj1@ifal.edu.br

Flávio Medeiros
Informatics Coordination
Federal Institute of Alagoas
Maceió, Alagoas, Brasil
flavio.medeiros@ifal.edu.br

Ivo Calado
Informatics Coordination
Federal Institute of Alagoas
Maceió, Alagoas, Brasil
ivo.calado@ifal.edu.br

***Abstract*—Tourism is an important activity for the economy of several places in the world. In this context, the use of information systems might bring some benefits for tourists, for example, by helping people to select tourist attractions, such as restaurants, beaches, and museums. In this paper, we perform two evaluations of 22 recommendation algorithms provided by existing recommendation system libraries, aiming to identify efficient algorithms in terms of prediction accuracy. In the first evaluation, we compare the algorithms by measuring different metrics, such as *RMSE*, *MAE*, *precision*, *recall*, and *F1 Score*. In the second evaluation, we compare the recommendation algorithms based on the answers of 172 people that participated in our survey by evaluating different kinds of tourist attractions. The results of our study show that some recommendation algorithms remain on the top list with regards to efficiency on both studies, such as the *SVD++*, *Baseline Only*, and *KNN Z Score with Pearson Baseline Similarity*. Others are efficient in the first evaluation, or for some metrics, but not in the second study, for example, or the other way around. The results of our study are useful for people that are creating solutions in the tourism domain.**

***Index Terms*—Recommendation systems, Tourist attractions**

I. Introduction

Tourism is an important activity for the economy of several places in the world [1]. In this context, the use of information systems might bring some benefits for tourists, which nowadays can find information about hotels, tourist attractions, and restaurants in many different places (e.g. *Four Square*, *Yelp*, and *Google*). Thus, the task of selecting what is relevant is difficult and time-consuming. Recommendation systems can help tourists in this task by recommending tourist attractions, restaurants, and hotels based on their profile and interests.

Considering the relevance of the topic, several approaches for system recommendation have been proposed, and many of those recommendation algorithms are available in well-known libraries, including *Surprise*, *Apache Mahout*, and *LibRec*. These libraries provide different algorithms to make recommendations, including neighborhood methods [2–4], matrix factorization-based techniques [5–7], and other types, such as SlopeOne [8], and CoClustering [9]. However, a little effort has been put into comparing the efficiency of these algorithms in practice, for example, by considering the perspective and opinions of human beings.

In this paper, we perform two evaluations of 22 collaborative filtering recommendation algorithms. The implementation of such algorithms were obtained from mature libraries, e.g., *Surprise*, *Apache Mahout*, and *LibRec*. After selecting the algorithms, we performed two complementary evaluations. In the first one, we compared the algorithms based on a set of metrics, such as *RMSE*, *MAE*, *precision*, *recall*, and *F1 Score*, to measure the efficiency of the algorithms in terms of prediction accuracy. Afterwards, in the second evaluation, we compared the same 22 algorithms based on the answers of 172 people that participated in our experiment by evaluating tourist attractions of different kinds. At the end, we triangulate the results of both studies with the goal of identifying the most efficient algorithms, that is, the ones that make predictions with fewer mistakes.

The results of our study show that some recommendation algorithms remain on the top list with regards to efficiency on both studies, such as the *SVD++*, *Baseline Only*, and *KNN Z Score with Pearson Baseline Similarity* algorithms. Other algorithms are efficient in the first evaluation, or for some metrics, but not at the second evaluation, for example, or the other way around. For instance, the *SVD* algorithm is the second more efficient for metrics *RMSE* and *MAE*, while in the second evaluation, it takes the 16 position in the ranking of more efficient algorithms. On the other hand, the *KNN Z Score with Cosine Similarity* is the most efficient in the evaluation with people, but it takes position 14 when considering metric *RMSE*. By triangulating the results of both studies, we improve evidence regarding the most efficient algorithms for recommendation systems in the tourism domain.

The remainder of this article is organized as follows. In Section II, we present some background information that is necessary to understand our study. In Section III, we discuss the setup of our study, and Sections IV and V present the results of the evaluations based on metrics and users perspective respectively. In Section VI, we compare the results of both evaluations. In Section VII, we discuss the related work, and Section VIII presents the concluding remarks and conclusions.

II. Background

In this section, we present a brief overview of recommendation systems and machine learning. Recommendation systems have emerged in response to the huge amount of information available nowadays [10], which makes it difficult

DOI reference number: 10.18293/SEKE2020-070

and time-consuming for choosing between a wide variety of products and services available, for example, in websites around the world. By definition, recommendation systems are software tools and techniques that automatically provide recommendations for users based on their past experience, that is, recommending the items that are most likely to interest a particular user [11]. Recommendation systems use the available user data, information about other users, and information about the environment with the goal of making predictions.

Currently, recommendation systems are a common application area in machine learning, which is a collection of algorithms and techniques that software developers can use to create computational systems that automatically learn by analyzing the available data to make predictions and inferences [12]. The algorithms for recommendation systems in machine learning are typically classified into three categories [13]:

1) **Content-Based Filtering**: algorithms in which the user receives recommendations based on similar items that the user has shown interest in the past [14];
2) **Collaborative Filtering**: the algorithm makes recommendations for users based on items that people with a similar profile has shown interest in the past. This category is subdivided into two subcategories [15]:
    a) **User-based Collaborative Filtering**: a memory-based algorithm that uses information about users and items to generate recommendations [16]. It calculates a certain similarity score among users, and based on this score, it selects the most similar users and recommends items that these similar users have previously shown interest;
    b) **Item-based Collaborative Filtering**: it is a model-based algorithm that provides item recommendations based on a model of user ratings [16]. In this algorithm, the similarities between pairs of items are calculated, and the similarity is used to recommend items.
3) **Hybrid Filtering**: algorithms that combine collaborative and content-based methods [17].

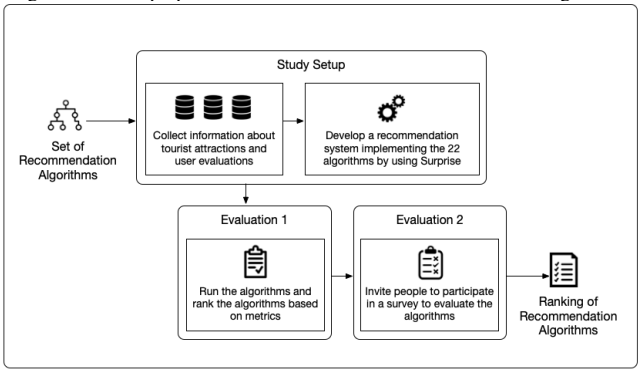
Recommendation systems are currently applied in many different domains, such as e-commerce [18], and video streaming [19], due to information overload. In the tourism field, there is also a huge amount of information available. For example, tourist information on the internet spreads across a wide range of different websites. In addition to the official websites of destinations, attractions or services, there are a wide range of blogs, and wikis, offering additional information about tourist attractions [20].

Developers can use a number of programming libraries in different languages to implement recommendation systems, such as *Surprise*, *Apache Mahout*, and *LibRec*. These libraries provide several algorithms to make predictions, including neighborhood methods [2–4], matrix factorization-based [5–7], and other types, such as SlopeOne [8], and CoClustering [9]. However, little effort has been put into comparing the efficiency of these algorithms in practice, considering the perspective of humans. In this study, we few this gap by performing two evaluations with the goal of ranking collaborative filtering algorithms: (1) by comparing the efficiency of the collaborative filtering algorithms by using a set of five metrics; and (2) by comparing the algorithms based on the answers of 172 people that participated in our survey.

III. Study Setup

To rank the collaborative filtering recommendation algorithms based on their efficiency, we followed the process described in Figure 1. In *Step 1*, we selected a set of platforms with tourist attractions information and selected four options to use in our study: *Trip Advisor*, *Google Places*, *Yelp*, and *Foursquare*. We extracted the attractions information manually and by using Application Programming Interface (API). Overall, we collected information about 92 tourist attractions, and more than 76 thousand ratings.

Fig. 1. The Steps performed to Rank the Recommendation Algorithms.



After collecting information about tourist attractions and ratings, in *Step 2*, we implemented a tool by using the *Surprise* library. This library provides a large set of out-of-box implementations for all recommendation algorithms arbitrarily selected to our study. Moreover, the *Python* programming language was selected due to its growing community working with machine learning and recommendation systems. In Table I, we list the 22 algorithms considered in our study.

In *Step 3*, we performed the first evaluation of the recommendation algorithms based on a set of five metrics, and ranking the algorithms based on the *RMSE*, *MAE*, *F1 Score*, *Precision*, and *Recall*. We made a system on the top of Jupyter [21], which allowed us to create an interactive visualization to compare the results of the recommendation algorithms. We present the details of this step in Section IV.

Last, in *Step 4*, we integrated the recommendation system that we developed in *Python* to a *RESTful API* [22] developed in *Flask* [23]. For the user interface, we implemented a responsive website by using *ReactJS*, which connects to the *RESTful API* through *HTTP* calls. After developing this infrastructure, we sent emails asking users to participate in our survey. We present the details of this step in Section V.

Algorithm	RMSE	MAE	Precision	Recall	F1
Baseline Only	0.89	0.70	0.86	0.92	0.89
SVD	0.90	0.71	0.86	0.87	0.87
SVD++	0.91	0.71	0.86	0.89	0.88
kNN Baseline with pearson baseline similarity	0.98	0.75	0.86	0.88	0.87
kNN Means with pearson baseline similarity	0.99	0.76	0.85	0.91	0.88
kNN Z Score with pearson baseline similarity	0.99	0.76	0.85	0.91	0.88
kNN Baseline with MSD	1.02	0.77	0.88	0.81	0.84
kNN Baseline with cosine similarity	1.02	0.77	0.88	0.81	0.84
kNN Means with with cosine similarity	1.03	0.78	0.87	0.82	0.85
kNN Basic with pearson baseline similarity	1.03	0.79	0.84	0.95	0.89
kNN Means with MSD	1.03	0.78	0.87	0.82	0.85
kNN Z Score with MSD	1.03	0.78	0.87	0.82	0.85
SlopeOne	1.03	0.79	0.87	0.82	0.84
kNN Z Score with cosine similarity	1.03	0.78	0.87	0.83	0.85
kNN Baseline with pearson correlation similarity	1.04	0.79	0.87	0.82	0.85
kNN Z Score with pearson correlation similarity	1.05	0.80	0.86	0.84	0.85
kNN Means with pearson correlation similarity	1.05	0.80	0.87	0.84	0.85
CoClustering	1.05	0.80	0.87	0.81	0.84
kNN Basic with cosine similarity	1.07	0.80	0.86	0.86	0.86
kNN Basic with MSD	1.07	0.80	0.86	0.86	0.86
kNN Basic with pearson correlation similarity	1.10	0.82	0.85	0.89	0.87
NMF	1.13	0.90	0.90	0.65	0.75

TABLE I
RANKING OF RECOMMENDATION ALGORITHMS BASED ON METRICS.

IV. METRIC-BASED EVALUATION

To evaluate the 22 recommendation algorithms, we consider a set of five metrics, as we explain next.

A. Mean Absolute Error (MAE)

The Mean Absolute Error calculates the error of estimated evaluations of users. For example, if a user rated an item with a five note and the system predicted a three note, the MAE is two. The formula used to calculate MAE is presented in Figure 2.

$$\text{MAE} = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} |r_{ui} - \hat{r}_{ui}|$$

Fig. 2. The MAE formula used in our study.

B. Root Mean Square Error (RMSE)

The Root Mean Square Error is a quadratic scoring rule that also calculates the error of estimated evaluations of users. It is the square root of the average of squared differences between the prediction given by the system and the user's evaluation. The RMSE assigns higher weights to larger errors, since errors are squared before the average. The formula used to calculate RMSE is presented in Figure 3.

$$\text{RMSE} = \sqrt{\frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} (r_{ui} - \hat{r}_{ui})^2}$$

Fig. 3. The RMSE formula used in our study.

C. Precision

The precision is used to know how many items the user really liked considering all recommended items. $Precision = tp/tp + fp$, where tp is the number of recommended items the user liked (true positives) and $tp + fp$ is the total number of recommended items (true positives, and false positives).

D. Recall

The recall is used to calculate the proportion of recommended items that the user likes. $Recall = tp/tp + fn$, where tp is the number of recommended items the user likes (true positives) and $tp + fn$ is the total number of items that user likes (true positives, and false negatives).

E. F1 Score

The F1 Score uses Precision and Recall in its calculation, and can be interpreted as a weighted average of both. $F1Score = 2 * [(precision * recall)/(precision + recall)]$.

F. k-Fold Cross Validation

Together with RMSE, MAE, Precision, and Recall, we used the k-fold cross validation [24] method. In the k-fold cross-validation, the data is divided into k subsets. Thus, the metrics equations is repeated k times, in which the time one of the k subsets is used as a validation set, and the other $k-1$ subsets are used as the training set. The error estimate is calculated based on all k attempts to get the full effectiveness of the model. By exchanging the training and test sets, we increase the effectiveness of this method. In our study, we used $k = 5$. In Figure 4, we illustrate this process in detail.

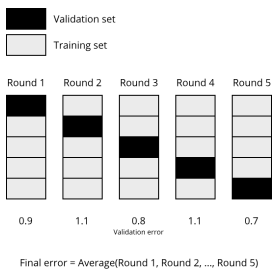


Fig. 4. Cross validation process used in our study with $k = 5$.

In Table I, we list not only the 22 algorithms considered in our study, but also the metric values according to the use of statistics. Notice that the algorithms are ranked based on the value of RMSE. In this context, the most efficient algorithms are *Baseline Only*, *SVD*, and *SVD++*. In the next section, we present the second evaluation based on answers of real users, which we performed to triangulate the results and increase evidence in the results.

V. Evaluation Based on Real-User Evaluations

To evaluate the 22 recommendation algorithms by taking into account the perception of real users, we integrated our recommendation system into a tourism application. The main goal of this evaluation is to compare the evaluations of real users with the evaluations provided by the recommendation system (that is, the evaluation based on metrics) for each algorithm considered in our study.

To perform the evaluation, we integrated the recommendation system into the *Hupi* application, which provides two main functionalities: (1) a centralized database with tourist attractions information; and (2) recommendations of tourist attractions for tourists based on their profile. In Figure 5, we present the main screen of the evaluation of the recommendation algorithms based on the *Hupi* application. In this screen, the users could see the tourist attractions and evaluate the attractions based on their perceptions by clicking on the number of stars. The user could choose to not evaluate any tourist attraction, and the system automatically generates a new attraction for evaluation, as the user might not know some attractions listed in the page.

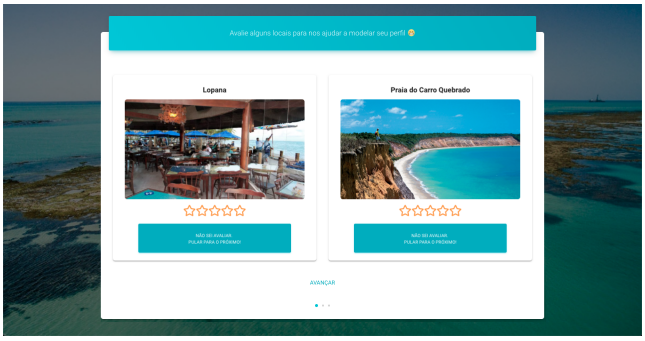


Fig. 5. The screen of the *Hupi* application used during our evaluation with real users.

The evaluation consists of two steps. In the first step, the system selects a number of tourist attractions and presents to the users, which evaluate each attraction based on their perceptions. This first step is used in our system to model the profile of the user. Based on the profile of the user, the systems runs the recommendation system to measure the possible rates for a set of tourist attractions. In the second step, the system selects a number of attractions, ask the users to evaluate again, and compare the rates of users with the rates generated by the recommendation system.

To select participants for our evaluation, we sent emails asking people to participate in the experiment. Overall, 172 users participated in the evaluation. In Table II, we show the results by ranking the algorithms considered in our study. It is important to notice here that the results of algorithms has changed when compared with the results of the first evaluation (see Table I). Error = $\int_1^n |system\ rating(i) - user\ rating(i)|$, where n is the number of evaluations, and i the current evaluation number.

Algorithm	Error
KNN Z Score with cosine similarity	18.08
KNN Baseline with Pearson correlation similarity	19.04
SVD++	20.04
KNN Means with with cosine similarity	20.08
KNN Means with MSD	20.62
Baseline Only	21.89
KNN Z Score with Pearson baseline similarity	22.38
KNN Means with Pearson correlation similarity	22.64
CoClustering	23.13
KNN Z Score with MSD	26.94
KNN Basic with Pearson baseline similarity	27.22
KNN Basic with Pearson correlation similarity	28.02
KNN Baseline with Pearson baseline similarity	28.85
KNN Baseline with cosine similarity	29.56
KNN Means with Pearson baseline similarity	30.27
SVD	30.76
SlopeOne	33.52
KNN Basic with cosine similarity	35.60
KNN Baseline with MSD	35.76
KNN Basic with MSD	41.49
NMF	42.30
KNN Z Score with Pearson correlation similarity	45.36

TABLE II
Ranking of recommendation algorithms based on the answers of real users.

VI. Discussion

We triangulate the results of both evaluations to rank the algorithms. As the evaluations and metrics ranked the algorithms differently, we compared the rankings to suggest the most efficient recommendation algorithms taking into account the results of both evaluations. Thus, we summed the ranking positions of each algorithm in both evaluations, for each metric, to identify the top algorithms based on their efficient. The top algorithms considering the results of both evaluations are *SVD++*, *Baseline Only*, and *KNN Z Score with Pearson baseline similarity*.

We summarise the results in Table III, in which we show the ranking of the algorithms in both evaluations. To rank the algorithms taking the results of the two studies, we calculated the Total Error (TE), by summing the rankings of both studies, that is $TE = RMSE + MAE + Recall + Precision + F1Score + People$, as we were interested in selecting the most efficient algorithms. So, the final ranking represents the algorithms with better ranking considering the two evaluations that we performed in this study, that is, the most efficient algorithms are the ones with the lowest total error values.

VII. Related Work

Many studies that compares recommendation algorithms focus on evaluating the quality of the recommendation system forecasts by using metrics. In [25–27], a recommendation system for tourism is evaluated by using cross-validation and several metrics, such them the Root Mean Squared Error (RMSE), which is also used in our study.

Algorithm	RMSE	MAE	People	Precision	Recall	F1	Total Error
SVD++	3	3	3	12	5	5	9
Baseline Only	1	1	6	17	2	2	8
kNN Z Score with pearson baseline similarity	6	6	7	19	3	3	19
kNN Means with with cosine similarity	9	9	4	4	17	16	22
kNN Z Score with cosine similarity	14	11	1	9	13	13	26
kNN Means with MSD	11	10	5	10	16	17	26
kNN Baseline with pearson baseline similarity	4	4	13	15	7	6	21
kNN Baseline with pearson correlation similarity	15	14	2	8	15	15	31
SVD	2	2	16	14	8	7	20
kNN Means with pearson baseline similarity	5	5	15	20	4	4	25
kNN Basic with pearson baseline similarity	10	15	11	22	1	1	36
kNN Z Score with MSD	12	12	10	6	14	14	34
kNN Baseline with cosine similarity	8	8	14	3	20	21	30
kNN Means with pearson correlation similarity	17	16	8	11	12	12	41
kNN Baseline with MSD	7	7	19	2	21	20	33
CoClustering	18	17	9	5	19	19	44
kNN Z Score with pearson correlation similarity	16	18	13	13	11	11	47
SlopeOne	13	13	17	7	18	18	43
kNN Basic with pearson correlation similarity	21	21	12	21	6	8	54
kNN Basic with cosine similarity	19	20	18	16	10	10	57
kNN Basic with MSD	20	19	20	18	9	9	59
NMF	22	22	21	1	22	22	65

TABLE III
RANKING OF RECOMMENDATION ALGORITHMS BASED ON METRICS AND ANSWERS OF REAL USERS.

Arsan et al. [28] proposes a study by using the *Mahout* framework with the goal of searching the best collaborative filtering algorithms. The study uses the metrics RMSE, and Mean Absolute Error (MAE) metric. In their study, they used movie rating dataset to compare the algorithms. In another study, Seminar and Wilson [29] used the open source *MovieLens* database to evaluate *Mahout* algorithms with the MAE metric.

Said and Bellogín [30] compared the results of some algorithms of the *Mahout*, *LensKit* and *MyMediaLite* frameworks in relation to the RMSE metric and time. In the tourism domain yet, Nilashi et al. [31] proposed a recommendation system for tourist places and evaluated it by using several metrics, such as RMSE, MAE, F1-measure, and Precision.

Noguera et al. [32] developed a mobile hybrid recommendation system by using user-based collaborative filtering combined with item-based filtering. The main contributions of this study is related to identifying the limitations of precision metrics. Liu et al. [33] proposed a new model of user similarity with the goal of improving the accuracy of collaborative filtering. It uses the metrics precision and recall to evaluate its results by claiming that the metrics RMSE and MAE, although widely used, do not guarantee user satisfaction. In our study, we complement these studies by comparing the results of RMSE with the results of a study with real users, which rated a number of tourist places.

Our study complements the existing studies by considering a big data set, by comparing 22 algorithms, using a set of five metrics, as well as by considering an evaluation with real users to get more evidence with the results.

VIII. CONCLUSION REMARKS

In this study, we performed a comparison of 22 recommendation algorithms in the tourism domain. We performed two evaluations: (1) based on metrics; and (2) based on the answers of real users, to rank the algorithms according to their efficiency. By triangulating the results of both studies, we improve evidence regarding the most efficient algorithms for recommendation systems.

The results of the evaluations ranked the algorithms differently. Some recommendation algorithms remain on the top list with regards to efficiency, such as the *SVD++*, *Baseline Only*, and *KNN Z Score with Pearson Baseline Similarity* algorithms, while others are efficient in the first evaluation, but not at the second one, or the other way around.

As future work, we intend to evaluate the recommendation algorithms in different domains, such as movie reviews, and music.

REFERENCES

1. Bunghez, C. L. The Importance of Tourism to a Destination's Economy. *Journal of Eastern Europe Research in Business and Economics,* 1–9 (2016).
2. Kramer, O. in *Dimensionality Reduction with Unsupervised Nearest Neighbors* 13–23 (Springer Berlin Heidelberg, 2013).
3. Koren, Y. *Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model* in *Proceedings of the International Conference on Knowledge Discovery and Data Mining* (ACM, 2008), 426–434.
4. Koren, Y. Factor in the Neighbors: Scalable and Accurate Collaborative Filtering. *ACM Trans. Knowl. Discov. Data* **4,** 1:1–1:24 (Jan. 2010).
5. Lee, D. D. & Seung, H. S. *Algorithms for Non-negative Matrix Factorization* in *Proceedings of the International Conference on Neural Information Processing Systems* (MIT Press, 2000), 535–541.

6. Luo, X., Zhou, M., Xia, Y. & Zhu, Q. An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems. *IEEE Transactions on Industrial Informatics* **10,** 1273–1284 (2014).
7. Koren, Y., Bell, R. & Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **42,** 30–37 (Aug. 2009).
8. Lemire, D. & Maclachlan, A. Slope One Predictors for Online Rating-Based Collaborative Filtering. *CoRR* (2007).
9. George, T. & Merugu, S. *A Scalable Collaborative Filtering Framework Based on Co-Clustering* in *Proceedings of the IEEE International Conference on Data Mining* (IEEE Computer Society, 2005), 625–628.
10. Melinat, P., Kreuzkam, T. & Stamer, D. *Information Overload: A Systematic Literature Review* in *Perspectives in Business Informatics Research* (eds Johansson, B., Andersson, B. & Holmberg, N.) (Springer int. Pub., Cham, 2014), 72–86.
11. Ricci, F., Rokach, L. & Shapira, B. in *Recommender Systems Handbook* (eds Ricci, F., Rokach, L. & Shapira, B.) 1–34 (Springer US, 2015).
12. Swamynathan, M. *Mastering Machine Learning with Python in Six Steps. A Practical Implementation Guide to Predictive Data Analytics Using Python* 1st ed. (Apress, 2017).
13. Akhil, P. V. & Joseph, S. A Survey of Recommender System Types and Its Classification. *International Journal of Advanced Research in Computer Science* **8,** 486–491 (2017).
14. Deshpande, M. & Karypis, G. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* **22,** 143–177. ISSN: 1046-8188 (2004).
15. Ekstrand, M. D., Riedl, J. T. & Konstan, J. A. Collaborative Filtering Recommender Systems. *Found. Trends Hum.-Comput. Interact.* **4,** 81–173 (2011).
16. Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. *Item-based Collaborative Filtering Recommendation Algorithms* in *Proceedings of the International Conference on World Wide Web* (ACM, 2001), 285–295.
17. Mourão, F., Rocha, L., Konstan, J. A. & Meira Jr., W. *Exploiting Non-content Preference Attributes Through Hybrid Recommendation Method* in *Proceedings of the ACM Conference on Recommender Systems* (ACM, 2013), 177–184.
18. Linden, G., Smith, B. & York, J. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing* **7,** 76–80 (2003).
19. Gomez-Uribe, C. A. & Hunt, N. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.* **6,** 13:1–13:19 (Dec. 2015).
20. Inversini, A. & Buhalis, D. *Information Convergence in the Long Tail: The Case of Tourism Destination Information* in *Information and Communication Technologies in Tourism 2009* (eds Höpken, W., Gretzel, U. & Law, R.) (Springer Vienna, Vienna, 2009), 381–392.
21. Kluyver, T. *et al. Jupyter Notebooks-a publishing format for reproducible computational workflows.* in *ELPUB* (2016), 87–90.
22. Schreier, S. *Modeling RESTful Applications* in *Proceedings of the Second International Workshop on RESTful Design* (ACM, 2011), 15–21.
23. Grinberg, M. *Flask Web Development: Developing Web Apps with Python* 1st (O'Reilly Media, Inc., 2014).
24. Rodriguez, J. D., Perez, A. & Lozano, J. A. Sensitivity Analysis of k-Fold Cross Validation in Prediction Error Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32,** 569–575 (2010).
25. Lim, K. H., Chan, J., Leckie, C. & Karunasekera, S. *Personalized Tour Recommendation Based on User Interests and Points of Interest Visit Durations* in *Proceedings of the International Conference on Artificial Intelligence* (AAAI Press, 2015), 1778–1784.
26. Lim, K. H. *Recommending Tours and Places-of-Interest Based on User Interests from Geo-tagged Photos* in *Proceedings of the ACM SIGMOD on PhD Symposium* (ACM, 2015), 33–38.
27. Lim, K. H., Chan, J., Leckie, C. & Karunasekera, S. Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency. *Knowl. Inf. Syst.* **54,** 375–406 (2018).
28. Arsan, T., Koksal, E. & Bozkus, Z. Comparison of Collaborative Filtering Algorithms with Various Similarity Measures for Movie Recommendation. *International Journal of Computer Science, Engineering and Applications* **6,** 1–20 (2016).
29. Seminario, C. E. & Wilson, D. C. *Case Study Evaluation of Mahout as a Recommender Platform* in *Workshop on Recommendation Utility Evaluation: Beyond RMSE* (2012).
30. Said, A. & Bellogın, A. *Comparative Recommender System Evaluation: Benchmarking Recommendation Frameworks* in *Proceedings of the 8th ACM Conference on Recommender Systems* (ACM, Foster City, Silicon Valley, California, USA, 2014), 129–136. ISBN: 978-1-4503-2668-1.
31. Nilashi, M., bin Ibrahim, O., Ithnin, N. & Sarmin, N. H. A multi-criteria collaborative filtering recommender system for the tourism domain using Expectation Maximization (EM) and PCA-ANFIS. *Electronic Commerce Research and Applications* **14,** 542–562 (2015).
32. Noguera, J. M., Barranco, M. J., Segura, R. J. & MartıNez, L. A Mobile 3D-GIS Hybrid Recommender System for Tourism. *Inf. Sci.* **215,** 37–52 (Dec. 2012).
33. Liu, H., Hu, Z., Mian, A., Tian, H. & Zhu, X. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems* **56,** 156–166 (Jan. 2014).