Skyline-based Recommendation Considering User Preferences

Shuhei Kishida[1], Seiji Ueda[1], Atsushi Keyaki[1], and Jun Miyazaki[1]

Tokyo Institute of Technology
kishida@lsc.cs.titech.ac.jp,ueda@lsc.cs.titech.ac.jp,
keyaki@lsc.cs.titech.ac.jp,miyazaki@cs.titech.ac.jp

Abstract. In this paper, we propose a skyline-based recommendation and ranking function. We suppose that some recommender systems, such as hotel recommender systems, are based not only on user preferences but also cost performance. For these kinds of applications, We first extract items with good cost performance and then identify items that users prefer, which reduce the computational cost of the online process. Based on the results of our preliminary experiments, we propose user feedback-based scoring and density-aware scoring methods where items that are highly similar to a user's latent requirements are recommended and attribute values in a dense area are quantized into a single value. The result of the experiments suggest that the density-aware scoring provides equal to or greater accuracy than the basic scoring.

Keywords: skyline operator, recommender system, user feedback, density

1 Introduction

A vast amount of data has been generated and is now stored on the Web. It is expected that much more data will be generated in the future. In this situation, a recommender system is required because finding information a user needs from such big data is laborious. Hence, many studies on recommender systems have been conducted. In particular, content filtering [8] and collaborative filtering [10] [12] has emerged as one of the most beneficial techniques. As the number of items becomes large, the cost of calculating the similarity between items becomes very expensive.

We suppose that some kinds of applications of recommender systems are based not only on user preference but also cost performance. For example, suppose a user would like to be recommended some useful hotels for his/her business trip. Some users may want to give priority to the price of the hotels (group A in Figure 1), while others may prioritize the distance from the hotels to the nearest station (group B in Figure 1). Others may prefer hotels with a balance between price and distance (group C in Figure 1). This implies that no user prefers to stay in hotels that are both expensive and far from the nearest station (group D in Figure 1). As a result, preferable hotels, namely, the items of groups A, B, and C, tend to be located in the lower left of the figure. We regard these items

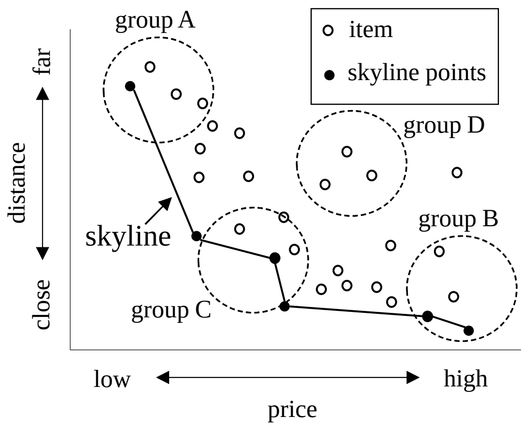

Fig. 1. Skyline operator and skyline points

as hotels with good cost performance. Considering the current user's behavior, we first extract only items with good cost performance, and then identify items that the user prefers. An advantage of this approach is that the computational cost of the online process can be reduced.

The hotels at the lowest and furthest left in Figure 1 (indicated by black dots) are hotels with the best cost performance. The *skyline operator* is an algorithm that efficiently extracts these hotels [1]. These items are called *skyline points*, and formally defined as hotels that are not dominated by any other hotel in all dimensions. The line connecting the skyline points is called the *skyline*. Features of an item such as price and distance are called *attributes*.

The skyline operator helps to efficiently extract the best cost performance items. Thus, a skyline-based recommender system is expected to be useful in some specific scenarios. Problems on achieving a useful skyline-based recommender system are as follows:

– The original skyline operator itself does not support a ranking function of extracted items, although a ranking function is a highly desirable feature of a recommender system.
– Some researches expanded the skyline operator and embedded the scoring function [9][7][14][11][2][6]. However, some of them suppose a situation where user's attribute weights are given and the others does not take account of a user's preference in scoring function. Moreover, to our knowledge, no research conducted a user study to evaluate effectiveness of the scoring function.

Our preliminary experiments for confirming the potential of a skyline-based recommender system revealed that the priority that users give to their preferences does not always agree with the actual importance of the preferences in decision making. To resolve this issue, we propose the user feedback-based scoring and density-aware scoring methods to the skyline-based recommender system.

2 Related Work

Researches on a ranking/scoring function of skyline operation tend to suppose *fuzzy query* [3][4][5]. Fuzzy query is a query where a user expresses his/her latent requirement in not concrete value but fuzzy expression, e.g., "inexpensive" for price and "close" for distance. Thus, a user's preference in this study is related to fuzzy query, although our approach suppose that a user just selects attributes and not implicitly express largeness of an attribute value.

An initial research [9] of embedding a scoring function to the skyline operator requires that the scoring function is predefined. Latter research [11] exploits context information to choose an appropriate scoring function. However, an effective scoring function for skyline-based recommender system is not obvious. Some researches [7][14] proposed a skyline operation with a scoring function. These researches rank items according to the number of dominating items. Other research [2] ranks items based on occurrence probability of an event. These ranking function is too naive to satisfy a user's requirements because these reflect only distribution of items and does not reflect a user's preference. Then, flexible fuzzy skyline [6] argues the necessity of a ranking function according to a user's preference, however, a concrete ranking function is not proposed.

3 Basic Scoring function for Skyline Points

In this study we focus on ranking of only skyline points, although items close to skyline are expected to be useful. This is because the first priority of this research is to confirm the potential of effectiveness of a skyline-based recommender system. Items, i.e., hotel data in this paper, are normalized in advance. $base(h, a)$, a base score of a hotel h in terms of an attribute a, is calculated as follows:

$$base(h, a) = \frac{original(h, a) - worst_a}{best_a - worst_a} \tag{1}$$

where $original(h, a)$ is a original value of h in terms of a, $worst_a$ is the worst value of a of all items, and $best_a$ is the best value of a in all items. In this study we do not take account of categorical data because categorical data is used as a filter rather than used in scoring items.

We introduce weighted linear sum into scoring skyline points because a base score of each attribute and importance of the attribute in terms of a user's preference in this study can be regarded as each score and a weight of the score in information retrieval, respectively. The basic score $S_{basic}(h)$ of hotel h is calculated as follows:

$$S_{basic}(h) = \sum_{a \in A} weight(a) \cdot base(h, a) \tag{2}$$

where A is a set of attributes used for the skyline operator, a is an attribute in A, $weight(a)$ is a weight of a, and $base(h, a)$ is a base score of h in terms of a.

4 Improvement of the Skyline-based Recommendation

4.1 Preliminary Experiment

We conducted preliminary experiments which confirmed 1) weighted linear sum scoring function is effective, and 2) a skyline-based recommendation does not

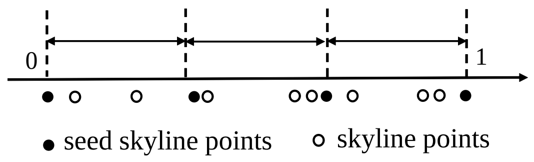

Fig. 2. Seed hotels for user feedback-based scoring

decrease accuracy by extracting only hotels with good performance. We should note that accuracy improves when the attribute weights are set equally with the exception of price. Specifically, the attribute weight of price should be set large for accurate recommendation. One very interesting result is that half of the users gave a different order of importance to the attributes than they did at the beginning. It is not always effective for a user to set the attribute weights by him/herself. It also indicated that stated user preferences do not always agree with actual importance in decision making.

4.2 User Feedback-based Scoring

The preliminary experiments suggest that users cannot always assign proper weights to attributes. Thus, we propose a method that leverages user feedback to interpret a user's latent requirements. Items that are highly similar to a user's latent requirements are recommended. This approach is a kind of content filtering because we calculate similarities between items.

Figure 2 draws arranged skyline points according to the base score of a certain attribute. First, we split the space into k subspaces (k is 3 in this case). We then extract $k+1$ skyline points located at the nearest to each border of the subspaces. We define these skyline points as seed skyline points (indicated by black dots in Figure 2). Note that seed skyline points are extracted for every attribute used by the skyline operator.

Next, we ask a user to choose the most preferable skyline points among the seed skyline points. We assume that the chosen seed skyline points represent the user's latent requirement. Hence, similar skyline points to the chosen seed skyline points are reasonable to be recommended. Therefore, a feedback score $S_{Feedback}(h)$ of a hotel h is calculated by similarities between the chosen seed skyline points and other skyline points as follows:

$$S_{Feedback}(h) = \sum_{c \in C} sim(h, c) \tag{3}$$

where C is a set of chosen seed skyline points, c is an chosen seed skyline point in C, $sim(h, c)$ is a similarity between h and c.

We use cosine similarity as a similarity measure in this paper; however, any similarity measure can be applied. There are also some options for calculating similarity because each skyline point has multiple attributes. We vary the number of attributes to be used when calculating a similarity. Concretely, we use three variations in the experiment: only the first attribute, three attributes used by the skyline operator, and all eight available attributes for each item, as will be

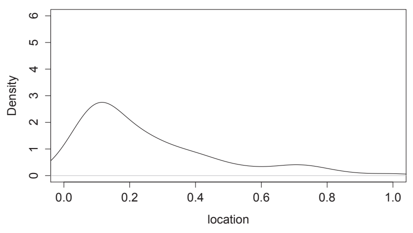

Fig. 3. Location kernel density distribution

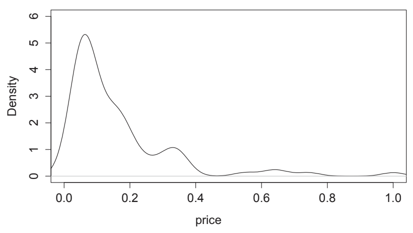

Fig. 4. Price kernel density distribution

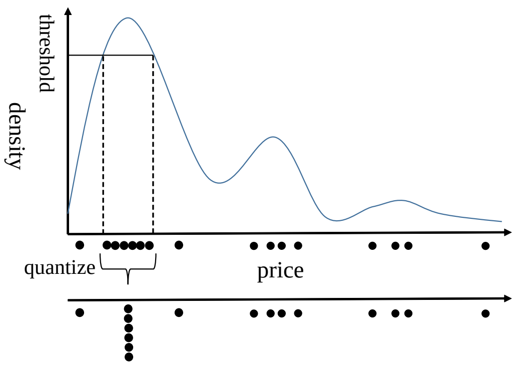

Fig. 5. Quantization of base scores in dense areas

introduced in Section 5.1. Thus, a similarity measure is calculated as follows:

$$sim(h,c) = \frac{\sum_{a' \in A'} v_{h,a'} v_{c,a'}}{\sqrt{\sum_{a' \in A'} {v_{h,a'}}^2} \sqrt{\sum_{a' \in A'} {v_{c,a'}}^2}} \tag{4}$$

where A' is a set of attributes to be chosen for calculating similarity, a' is an attribute in A', $v_{h,a'}$ is a base score of h in terms of a', and $v_{c,a'}$ is a base score of c in terms of a'. In this study, we set k to 3 because we assume that a three-way split is enough to capture a user's latent requirements[1]. We also decided to use only one item for feedback to avoid a user's labor.

4.3 Density-aware Scoring

Judging from the interview results of the preliminary experiments, it seems that dense areas occur in some attributes. When evaluating skyline points in these areas, it is expected that users will give more importance to other attributes because there is little difference that can be used to determine candidates along an attribute with many skyline points with a similar score. In other words, there is a gap between the scoring method of base score and decision making because even trivial differences of the base scores among the skyline points are

[1] As k increases, the system may be able to express the user's latent requirements more precisely. However, a user's labor increases when the number of seed skyline points increases.

Table 1. Attribute weights

	First attribute	Second attribute	Third attribute
p1	0.33	0.33	0.33
p2	0.50	0.50	0.0
p3	0.90	0.05	0.05

considered. To address this issue, we quantize the base scores of skyline points in dense areas.

We investigated the distributions of the skyline points along each attribute of the skyline operators for many combinations of attributes using kernel density estimation [13]. As a result, the density curves of the distributions have a largely similar shape, that is, a gentle curve. Figure 3 illustrates the kernel density distribution for location. In contrast, price is an exception to this trend. Its shape is sharp, as shown in Figure 4.

It seems that the features of the distributions are appropriately captured using kernel density estimation, as Figures 3 and 4 show. Therefore, we use the kernel density distribution to identify dense areas. An example of quantization for price is depicted in Figure 5. A dense area is defined as a range where the density exceeds a threshold value. The base scores of skyline points located within the dense area are quantized to a single value, which is the average value of these base scores. Consequently, a density score $S_{Density}(h)$ of a hotel h is calculated as follows:

$$S_{Density}(h) = \sum_{a \in A} weight(a) \cdot density(h, a) \tag{5}$$

$$density(h, a) = \begin{cases} base(h, a) & (base(h, a) < \tau) \\ quant(h, a) & (otherwise) \end{cases} \tag{6}$$

where A is a set of attributes, a is an attribute in A, $weight(a)$ is a weight of a, τ is a threshold value, and $quant(h, a)$ is a quantized value of h in terms of a.

The difference between the density-aware scoring and the basic scoring is that the density-aware scoring quantizes the base scores in the dense areas. Accordingly, the subsequent processes for density-aware scores are the same as those for the basic ones. Note that the best threshold value will be examined in the next section.

5 Experimental Evaluation

5.1 Data Set

In the experiments, we used the Rakuten Travel Data Set, which contains Japanese hotel data. We used 692 hotels located in Tokyo. Each hotel has eights attributes, that is, price, distance, and six user review ratings (service, location, facility, room, bath/spa, and food). The users review ratings are between 1 to 5; a higher value indicates better rating.

5.2 Experimental Setting

The number of the participants in these experiments was 30, and they are graduate students. We operated two experiments: one did *not* include the price attribute for the skyline operator and the other included the price attribute. This is

Table 2. Results without price

	$nDCG_{30}$
p1-Basic	0.990
p1-Linear	0.990
p1-Density-3	0.988
p2-Density-3	0.962
p2-Basic	0.960
p2-Linear	0.956
Feedback-8	0.945
p3-Density-2	0.940
p3-Basic	0.932
p3-Linear	0.926
Feedback-3	0.912
Feedback-2	0.892
Sort	0.851

Table 3. Results including price

	$nDCG_{30}$
p3-Density-5	0.918
Feedback-8	0.916
Feedback-3	0.909
p1-Density-5	0.901
p2-Density-5	0.895
p2-Basic	0.888
p2-Linear	0.878
p3-Basic	0.877
p3-Linear	0.876
p1-Basic	0.866
p1-Linear	0.863
Feedback-1	0.848
Sort	0.368

because the tendency between them was quite different, as the results of the preliminary experiments showed. Evaluation measures are nDCG at 30. The users choose three attributes for the skyline operator and the weight combinations of attribute are shown in Table 1.

We examined five methods. Two of them were proposed to further improvement of accuracy in Section 4, i.e., a user feedback-based scoring (*Feedback*) and a density-aware scoring (*Density*). The other methods are *Basic*, *Linear*, and *Sort* where hotels are sorted by the base score of the first axis as a naive method.

5.3 Experimental Results

Experiment without Price Table 2 shows the result of the experiment without price. The number of attributes used in the similarity calculation for *Feedback* and the threshold value for *Density* are added to the end of the method labels. Moreover, we only include the most accurate version of *Density* per weight combination. For example, p1-Density-1 is omitted because it is less accurate than p1-Density-2. It is effective to count every attribute in the scoring function when the attributes for the skyline operator do not include price.

In contrast, *Feedback* is less effective than all other methods except *Sort.* The accuracy of *Density* is slightly inferior to that of *Linear* and *Basic.* As a result, *Feedback* and *Density* do not improve accuracy in the experiment without price.

Experiment with Price *Density* is the most accurate method in the experiment that includes price, as illustrated in Table 3. We assume that this is achieved because the attribute of price has a wider range of dense areas and Density works well in this situation. Further, *Feedback* improved accuracy compared with *Basic* and *Linear* when eight or three attributes were used.

Discussion According to the above results, *Density* attains accurate and stable recommendations. In this method, weights for attribute do not have to be tuned and can be just assigned an equal value when there is no attribute that overwhelms the other attributes in decision making. In contrast, it is effective to assign a high weight to an attribute that overwhelms the others.

6 Conclusion

In this paper, we discussed a skyline-based recommendation with a ranking function taking account of a user's preference and conducted a user study. We proposed the *Feedback* and *Density* methods for improving existing ones. Experimental evaluations showed that *Density* recommends items accurately with stability. With *Density*, automatically tuning the threshold value is a part of our future work. Applying more advanced score fusion methods instead of weighted linear sum is also our future work.

7 Acknowledgements

This work was partly supported by JSPS KAKENHI Grant Numbers 15H02701, 15K20990, 16H02908, 26280115, and 25240014. The Rakuten Travel data set was provided according to the contract between NII and Rakuten, Inc.

References

1. Stephan Börzsönyi, Donald Kossmann, and Konrad Stocker. The Skyline Operator. In *Proc. of the 17th ICDE*, pages 421–430, 2001.
2. Patrick Bosc, Allel Hadjali, and Olivier Pivert. On Possibilistic Skyline Queries. In *Proc. of 9th FQAS*, pages 412–423, 2011.
3. Patrick Bosc and Olivier Pivert. Fuzzy Queries and Relational Databases. In *Proc. of SAC*, pages 170–174, 1994.
4. Patrick Bosc and Olivier Pivert. SQLf: a Relational Database Language for Fuzzy Querying. *IEEE Transactions on Fuzzy Systems*, 3:1–17, 1995.
5. Patrick Bosc, Olivier Pivert, and Amine Mokhtari. Top-k Queries with Contextual Fuzzy Preferences. In *Proc. of the 20th DEXA*, pages 847–854, 2009.
6. Allel Hadjali, Olivier Pivert, and Henri Prade. On Different Types of Fuzzy Skylines. In *Proc. of 19th ISMIS*, pages 581–591, 2011.
7. Hsin-Hsien Lee and Wei-Guang Teng. Incorporating Multi-Criteria Ratings in Recommendation Systems. In *Proc. of the IEEE IRI*, pages 273–278, 2007.
8. Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. *Content-based Recommender Systems: State of the Art and Trends*, pages 73–105. Springer US, 2011.
9. Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. An Optimal and Progressive Algorithm for Skyline Queries. In *Proc. of SIGMOD*, pages 467–478, 2003.
10. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: an Open Architecture for Collaborative Filtering of Netnews. In *Proc. of CSCW*, pages 175–184, 1994.
11. Dimitris Sacharidis, Anastasios Arvanitis, and Timos Sellis. Probabilistic Contextual Skylines. In *Proc. of the 26th ICDE*, pages 273–284, 2010.
12. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based Collaborative Filtering Recommendation Algorithms. In *Proc. of the 10th WWW*, pages 285–295, 2001.
13. Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis.* CRC press, 1986.
14. Man Lung Yiu and Nikos Mamoulis. Efficient Processing of Top-k Dominating Queries on Multi-Dimensional Data. In *Proc. of the 33rd VLDB*, pages 483–494, 2007.