Feedback Clustering for Online Travel Agencies Searches: a Case Study

Sara Scaramuccia[1,2][0000−0003−4010−6507]
Simon Nanty[2] and Florent Masseglia[3]

[1] Université de la Côte d'Azur, France sara.scaramuccia@amadeus.com
[2] Amadeus S.A.S., Sophia Antipolis, France Simon.nanty@amadeus.com
[3] florent.masseglia@inria.fr

Abstract. Understanding choices performed by online customers is a growing need in the travel industry. In many practical situations, the only available information is the flight search query performed by the customer with no additional profile knowledge. In general, customer flight bookings are driven by prices, duration, number of connections, and so on. However, not all customers might assign the same importance to each of those criteria. Here comes the need of grouping together all flight searches performed by the same kind of customer, that is having the same booking criteria. Better recommendations can be proposed to customers with similar booking criteria. The effectiveness of some set of recommendations, for a single cluster, can be measured in terms of the number of bookings historically performed. This effectiveness measure plays the role of a feedback, that is an external knowledge which can be recombined to iteratively obtain a final segmentation. In this paper, we describe our Online Travel Agencies (OTA) flight search use case and highlight its specific features. We address the flight search segmentation problem motivated above by proposing a novel algorithm called Split-or-Merge (S/M). This algorithm is a variation of the Split-Merge-Evolve (SME) method. The SME method has already been introduced in the community as an iterative process updating a clustering given by the K-means algorithm by splitting and merging clusters subject to feedback independent evaluations. No previous application of the SME method to the real-word data is reported in literature to the best of our knowledge. Here, we provide experimental evaluations over real-world data to the SME and the S/M methods. The impact on our domain-specific metrics obtained under the SME and the S/M methods suggests that feedback clustering techniques can be very promising in the handling of the domain of OTA flight searches.

Keywords: feedback clustering · flight search recommendations · flight booking · active segmentation

1 Introduction

In the travel industry, there is a strong need in understanding customer needs for applications such as, pricing, revenue management, service development, and the one addressed in this paper, namely flight search recommendations. Most of times customers' interests are only expressed as a flight search request and this makes the case of flight recommendations so peculiar. As pointed out in [12], the challenge in this domain is the lack of a customer profile knowledge to rely on. Authors in [12] exploit Discrete Choice Modeling to better understand customers' behaviors. However, this implies to have some predefined customer classes.

We tackle a similar task by focusing on clustering techniques. Our approach aims at grouping together those flight searches which prize similar criteria when booking/choosing a flight, that is similar priorities assigned to recommendation features such as price, duration, number of connections, and so on. We do not want simply to find the cluster of flight searches corresponding to some given priorities in the booking criterion. Indeed, that would be similar to labeling a cluster as business or leisure in advance as in the case of Discrete Choice Modeling approaches. Moreover, the similarity among flight searches in the same cluster is not necessarily correlated to some standard clustering quality measure, such as Silhouette or Adjusted Rand indexes. Hence, we want the quality of the flight recommendations obtained for a single cluster to be treated as an external knowledge to be added to drive the clustering process. This motivates our approach. We choose an Active Learning strategy [14], here applied in particular to domain segmentation and called Feedback Clustering.

In Feedback Clustering, one starts with an initial clustering. Then, the clustering is evaluated, that is a feedback is collected. Finally, based on that feedback the clustering model is updated and a new clustering is produced. The iterative process stops when a threshold quality value is satisfied. In the example in Fig-

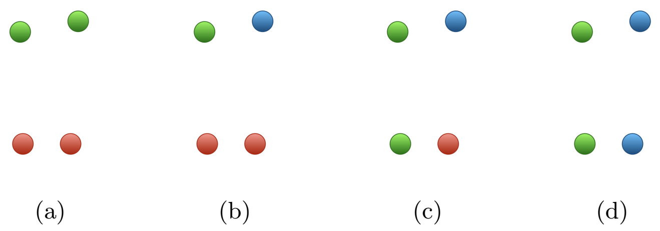

Fig. 1: Example: a segmentation where clusters (marked by colors) are progressively arranged according to the external feedback of grouping points with similar horizontal component.

ure 1, we show feedback clustering in action in a toy example. We have an initial clustering (1a) corresponding to the output of any clustering algorithm simply based on the geometrical arrangement of our data. We imagine that our feedback prizes clusters where points share similar horizontal values. We imagine that as something we could not know in advance and thus could not have been exploited in the clustering algorithm. Rather, we think of itas a very simplified possible result of some domain analysis. In the example, the feedback is in the form of rejecting the cluster where points are to different. Then in step (1b), the worst cluster is modified by splitting it into two. Afterwards in step (1c), the two singletons are merged to form a cluster whose points have very similar horizontal coordinates. Finally in step (1d), the final clustering is produced by merging the two left points. The so obtained clustering reflects better than the initial one the feedback request.

In our application, we have a specific feedback we call *customizability*. Customizability measures in each cluster how effective are the customer preferences estimated by an Amadeus optimization routine based on historical data. In some sense, we are measuring how “learnable” the class of customer is from a single cluster.

Since the customizability feedback is a numerical value independently associated to each cluster, we focus on the Feedback Clustering technique introduced in [17], here called Split-Merge-Evolve (SME) method, along with specifically tailored variations. The SME method is an example of Feedback Clustering where the clustering model is updated from one iteration to another by independently acting over the worst found clusters which can either be split or merged to others to form better clusters. Moreover, the SME method is presented as a modification of the k-means algorithm but the approach can be easily adapted to any clustering algorithm in the future.

Our main contributions are:

– a discussion on the results we obtained via the SME method under both a standard cluster quality evaluation, namely the Residual Square Sum (RSS) and the non-deterministic customizability feedback
– a similar discussion on a variation of the SME method here introduced and called the S/M method where split and merge operators might be applied independently, that is not necessarily in a row
– a comparison between the SME and S/M methods in terms of the obtained results and the future perspectives.

The paper is organized as follows. In Section 2, we review the related works. In Section 3, we formalize our problem to be used in Section 4 to present the details of the SME method and in Section 5 its variation: the S/M method. In Section 6, we present our results and highlight our contribution. In Section 7, we discuss the relevance of the work and future directions.

2 Related Works

In this section, we present an overview on the related works.

The problem of classifying customers in the travel industry by simply relying on flight searches and no prior available classes is addressed in [6]. Therein, the segmentation is achieved by exploiting consensus clustering techniques surveyed in [16]. Therein, consensus clustering is integrated with a multi-objective optimization process to evaluate the effectiveness of the recommendations over each segment independently. In consensus clustering, several, heterogeneous enough, segmentation models are taken as an initial population, possibly with a different number of clusters. Afterwards, the genetic algorithm NSGA-II [8] is adapted to find the segmentation which maximizes the consensus while minimazing the standard deviation in similarity.

As already explained in Section 1, our framework is a specific area of the more general framework of active learning, namely that of *feedback clustering*. To the best of our knowledge there is no attempt to achieve a similar task in the framework of feedback clustering. It is worth noticing that a feedback might come in several forms (categorical/numerical) and levels (instance, cluster, clustering). Therefore, there are several and very different ways to perform feedback clustering in the literature.

In [5], the authors consider *multiple clusterings* to be successively selected based on the user feedback with no particular restriction on the kind of feedback.

In the case of a feedback at *instance level*, the methods described in [3][9] exploit pair-wise similarity (numerical) while [4][11][10] exploit must/cannot link information (categorical), among domain points. The methods in [7][18] require a preferred component direction in the feature space to be specified and exploit spectral clustering.

In the case of a feedback at *cluster level*, we mention [13] for numerical (dis)similarity among clusters. Among the methods exploiting categorical cluster feedbacks, let us mention the work of [2][1][15][17] based on *cluster rejection*. One or more clusters are processed and the feedback consists in keeping the cluster, modifying it, or rejecting it. Those methods are the best suited to our purposes because they act at cluster level. In [2], the user can reject a cluster to be split into two clusters of equal cardinality or reject two clusters to be merged. A local version of a similar approach is proposed in [1]. In [15], the authors introduce a Bayesian elicitation process to let the model learn how to clusterize data based on previous feedbacks of that kind. In [17] authors introduce the SME method by applying split/merge operators to modify rejected clusters obtained by applying k-means. More precisely, the rejected cluster is, at a first step, split into two new clusters, at a second step, the two closest clusters are merged into a single one. The method combines advantages of bisecting k-means clustering (the split phase) to those of agglomerative hierarchical clustering (the merge phase). Under the Adjusted Rand Index (ARI), the SME method show its effectiveness compared to k-means and agglomerative hierarchical clustering over both synthetic and real datasets.

3 Problem Formulation

In this section, we formalize our problem. Let X be the set of flight searches whose elements are points in the Euclidean space $\mathbb{R}^n$, with the integer $n > 0$ representing the number of flight search features. A *clustering of* X into k clusters is a surjective function $c : X \longrightarrow \{0, \ldots, k-1\}$, where the *cluster* i is the subset of X defined by $X_i := c^{-1}(\{i\})$, for each $i \in \{0, \ldots, k-1\}$. For each cluster

X_i, we denote by $y_i \in \mathbb{R}$ the *(cluster) feedback*. For the whole clustering c, the *feedback* is denoted by the k-tuple $(y_0, \ldots, y_{k-1})$. An *evaluation of a clustering* c is a real number y depending on $(y_0, \ldots, y_{k-1})$, denoted by $y_c = y_c(y_0, \ldots, y_{k-1})$ The problem we address is that of optimizing the evaluation $y_c(y_0, \ldots, y_{k-1})$ by iteratively redefining the clustering c in terms of the clustering feedback $(y_0, \ldots, y_{k-1})$.

4 SME Clustering Framework

In this section, we recall the SME method. More details can be found in the original work [17]. We proceed by describing the method in terms of a generic feedback evaluation. Later on in section 4.1, we define the feedback evaluations of our interest.

In the SME (centroid-based) clustering framework, the dataset X is clusterized into clusters $X_0, \ldots, X_{k-1}$ by the k-means algorithm initialized by k random centroids $m_0, \ldots, m_{k-1}$ in $\mathbb{R}^n$. The current clustering is set to $X_0, \ldots, X_{k-1}$.

In the *evaluation phase*, the k-tuple $(y_0, \ldots, y_{k-1})$ of cluster feedbacks is retrieved. Moreover, the clustering is assigned its evaluation $y = y(y_0, \ldots, y_{k-1})$.

The iterative process starts with a *split action* over the current clustering. In the split action, the cluster i corresponding to the worst feedback value y_i is selected. Two new clusters X_k, X_{k+1}, and relative centroids m_k, m_{k+1} replace the old cluster c_i and centroid m_i in the current status. In addition to what authors presented in [17], a single k-means iteration is performed over the current set of centroids. This is done for technical reasons. In particular, we need to have each cluster described as the set of all points sharing the same closest centroid. Afterwards, in the *merge action*, the two closest pairs of centroids among $m_1 \ldots, m_{i-1}, m_{i+1}, \ldots, m_{k+1}$ are selected and their corresponding clusters discarded by the current status and replaced by their union: X_{k+2} with new centroid m_{k+2}. After the split and merge actions, the number of clusters is preserved.

In the *evolve phase*, the evaluation phase is applied to retrieve the clustering evaluation y_{new} relative to the current clustering. If y_{new} is better than y, then y is set to y_{new} thus becoming the new best evaluated clustering. In any case, the next iteration starts with a new split action over the current clustering. Finally, the *stopping criterion* is the following. The iterations are repeated till a certain value of the evaluation y is reached or a certain number of iterations is performed.

4.1 Evaluations.

In our study, we consider two possible evaluations of a clustering. Both of them are aggregating internal indexes, that is numerical values associated independently over each cluster. However, the first index we introduce is purely geometrical whereas the second one is domain-specific to flight recommendations and provides a fully external knowledge to be added in the form of feedback.

RSS Feedback. For each cluster X_i, the *Residual Square Sum* (RSS) is defined by

$$\mathrm{RSS}(i) := \frac{1}{|X_i|} \sum_{x \in X_i} \|x - m_i\|^2, \tag{1}$$

and its evaluation over the whole clustering is obtained by taking the average weighted by cluster sizes

$$\frac{1}{|X|} \sum_{i=0}^{k-1} |X_i| \cdot \mathrm{RSS}(i). \tag{2}$$

The so-obtained evaluation is a deterministic one.

Customizability feedback As already stated in the introduction, customizability measures how "learnable" the class of customer is from a single cluster. In order to explain that, we need to introduce some intermediate notions.

First of all, we need to point out that, for each flight search, a list of 200 flight recommendation is provided by the Amadeus search engine. The Amadeus search engine acts differently if set in terms of *price* or *value*. The value of a single recommended flight is obtained as a weighted combination of the flight price and other 24 not independent booking criteria such as duration, number of connections, time to wait from one flight to another, and so on. If all weights are set to 0 then the value selection boils down to recommend flights based on price (only). Once the weights for the value computation are set, the 200 flights are selected among several thousands according to the *value* associated to each single recommended flight. Such a list of recommendations is evaluated in terms of *popularity*. Popularity is a counter, weighted by flight ages, of the previous bookings of the same flight in history. In these terms, a better recommendation includes more flights which have been booked a lot in the past.

An instance of optimized weights $\bar{w}$ is found for a specific set (or a cluster) of flight searches by running an Amadeus multi-objective bayesian optimization routine. The procedure acts on the 100 flight searches in the cluster corresponding to the most booked flights in history. The routine provides the optimized weights $\bar{w}$. Our assumption is that, the more the flight searches are homogeneous in terms of preferences in the booking criteria (hidden knowledge to us), the more the weights found by the Amadeus routine give recommendations with higher popularity. Hence, the so-found weights are evaluated by picking up 100 other flight searches in the cluster randomly selected among the most booked ones. This step makes such evaluation a non-deterministic one. Moreover, since the absolute values of popularity might vary a lot from one set of flight searches to another, we take popularity based on price (weights set to 0) as a stable reference. The average popularity $pop_{\bar{w}}$ obtained over all 200 recommended flights is compared to the popularity of the recommendations obtained with weights set to 0 pop_0. Specifically, the effectiveness of the chosen weights is measured by taking the relative change between popularity based on price and popularity based on the value obtained by the chosen weights.

$$\text{Custom.}(i) := \frac{pop_{\bar{w}} - pop_0}{|pop_0|}, \tag{3}$$

For instance, a positive cluster feedback 0.50 means that recommendations obtained with the optimized weights are 50% more popular than recommendations obtained with all weights set to 0, that is recommendations based on price only. However, negative cluster feedbacks can also be obtained.

To evaluate the entire clustering $X_0, \ldots, X_{k-1}$, we take the following

$$\frac{1}{|X|} \sum_{i=0}^{k-1} |X_i| \cdot \text{Custom.}(i). \tag{4}$$

The so-obtained evaluation is not deterministic since values of $Custom.(i)$ might fluctuate. We quantify later on in Section 6 the relevance of the fluctuation.

5 S/M Clustering Framework: an SME variation

Analogously to the SME clustering framework, in the S/M clustering framework, the dataset X is clusterized into clusters $X_0, \ldots, X_{k-1}$ by the k-means algorithm initialized by k random centroids $m_0, \ldots, m_{k-1}$ in $\mathbb{R}^n$. The current clustering is set to $X_0, \ldots, X_{k-1}$.

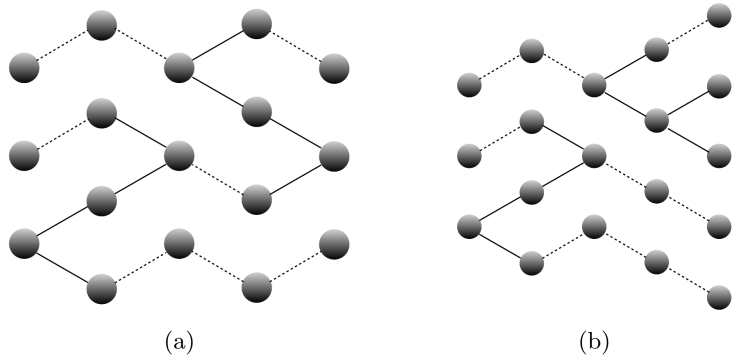

Fig. 2: The logical schemas of the SME (a) and the S/M (b) methods in comparison. Each grey ball represents a cluster. On the same vertical axis, we have a clustering. Solid lines connect clusters obtained by split/merge actions from one iteration to another. Dashed lines track corresponding clusters from one iterations to another.

The evaluation phase corresponds to that of the SME framework. Thus, it provides the k-tuple $(y_0, \ldots, y_{k-1})$ of cluster feedbacks with corresponding clustering evaluation $y = y(y_0, \ldots, y_{k-1})$.

Differences are to be found in the iterative process. This time it is decided whether to apply a *split action* or a *merge action* based on the size s_i of the worst-evaluated cluster i. If s_i is among the first half of size values in the current clustering, then a split action is performed. Otherwise, a merge action is performed. Both split and merge actions correspond to the same actions already described in Section 4 for the SME framework.

The evolve phase and the stopping criterion are analogous to the evolve phase of the SME framework. However, as shown in fig. 2, we underline that this time, the number of clusters k is not necessarily preserved by the S/M method. Indeed, split and merge actions can be applied a different number of times. Moreover, in the S/M method the evaluation is performed after each split/merge action.

6 Experimental Evaluation

In this section, we present and discuss the results we obtained in applying feedback clustering to our use case.

The current section is structured as follows. In Section 6.1, we describe our test settings. In Section 6.2, we test the impact of methods according to the kind of feedback they are acting on. In Section 6.3, we compare all tested methods to the specific task of increasing customizability feedback, whatever their feedback.

6.1 Test settings

In this section, we outline the settings considered for our tests. The methods to be tested are the SME method introduced in Section 4 and the S/M method introduced in Section 5. Each method, can be implemented with respect to the RSS or the customizability (Custom.) feedback introduced in Section 4.1. Hence, we have four methods to be tested:

- *SME(RSS)*: split-merge-evolve method first introduced in [17] (see Section 4) where the RSS index evaluates each cluster. Each SME iteration is performed over the worst-valued cluster (maximum under RSS)
- *SME(Custom.)*: split-merge-evolve method first introduced in [17] (see Section 4) where the customizability index evaluates each cluster. Each SME iteration is performed over the worst-valued cluster (minimum under customizability)
- *S/M(RSS)*: split-merge-evolve method is modified so that split and merge phases are separated (see Section 5). The worst-values cluster (maximum under RSS) is either split or merged to its closest cluster according to its size.
- *S/M(Custom.)*: split merge evolve method is modified so that split and merge phases are separated (see Section 5). The worst-values cluster (minimum under customizability) is either split or merged to its closest cluster according to its size.

Flight searches are represented as points in the 8-dimensional space. Indeed, along with origin and destination not included in the 8 dimensions, each flight search provides: distance between origin and destination, advance purchase, stay duration, number of passengers, number of children, geography (categorical value to distinguish among domestic, continental and intercontinental flight searches), departure day of the week, return day of the week (taking values from 0 to 6). Thus, we have heterogeneous features mixing numerical and categorical values. Moreover, our features might be dependent from one another. Every feature value is numerically treated as a real number value. For technical simplicity reasons, only round-trip searches are considered. Beyond the 8 features, each flight search comes associated with its own origin/destination data, that is the departure and arrival airport, respectively. Origin and destination are not directly used in our clustering phase.

In our tests, we considered 3 datasets varying according to the country of origin (see 1).

Name	Size	Expected Relative Change
FR	348 k	0.197
GB	4 M	0.201
AR/BR	1.5 M	0.206

Table 1: Datasets characteristics.

Moreover, datasets are chosen to be various in terms of size and expected relative change in their customizability evaluation. Indeed, as already mentioned, customizability evaluations are subject to value fluctuation. We quantify it over each dataset. For each number of clusters k in $[2, 3, 4, 5, 6, 7]$ we measure the expected relative change over 10 customizability evaluation calls over the same clustering of k clusters.

For each dataset, methods are called with a number of initial clusters k varying in $[2, 3, 4, 5, 6, 7]$. The SME(RSS) and the SME(Custom.) methods are tested over all datasets by applying 6 iterations. The S/M(RSS) and the S/M(Custom.) methods are tested over all datasets by applying 12 iterations to have the same number of elementary operators as for the SME methods. Independently from the method, both evaluations RSS and customizability are stored.

Our aim is to measure the impact of the tested methods on the initial clustering evaluation: RSS_0 for the RSS feedback and $Custom_0$ for customizability. The initial evaluation is compared to the best obtained during the method call: $\min RSS$ for the RSS feedback and $\max Custom$ for the customizability feedback. The comparison is numerically obtained by taking the relative change in between initial and best iteration.

6.2 Test 1: own feedback impact evaluation of SME

The first comparison we report is that in between the SME(RSS) method and the SME(Custom.). The impact of each method for a given number k of initial clusters is taken with respect to the feedback evaluation leading the process. Indeed, for SME(RSS), the impact (blue bars in Figure 3) is the relative change between initial and best RSS evaluations along the process. The average of all impacts is taken over runs with k varying in $\{2, 3, 4, 5, 6, 7\}$. Analogously, the impact for SME(Custom.) (orange bars in Figure 3) is the relative change between initial and best customizability evaluations.

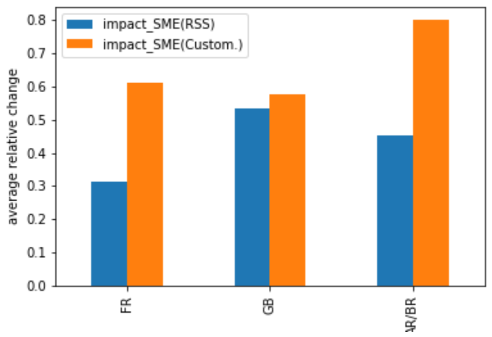

Fig. 3: Average relative change by dataset in between initial and best iteration under the SME method. *impact_SME(RSS)* and *impact_SME(Custom.)* reports relative change under the RSS evaluation and customizability evaluation, respectively.

We can use the SME(RSS) impact as a reference for SME(Custom.). For all datasets, the SME(Custom.) presents higher values. We believe that this is partially due to the fact that the RSS feedback is somehow already optimized by the k-means algorithm whereas the customizability feedback is a purely external feedback. The order of magnitude of the two impacts is the same. This, in lack of a ground-truth to measure the effectiveness of the SME(Custom.), suggests reliability on the impact evaluations. For AR/BR and FR datasets, values of SME(RSS) are doubled by those of SME(Custom.). For GB dataset, values are closer.

For the RSS evaluation (blue bars), our measures confirm the effectiveness of the SME method applied to the domain of online flight searches. For customizability (orange bars), we found all dataset impacts more relevant compared to the dataset expected relative change reported in Table 1. This confirms the adaptability of the SME method being not limited to standard clustering quality indexes (RSS), rather for improving a domain specific quality index such as customizability.

6.3 Test 2: SME and S/M for segmenting online flight searches

Similarly to what done in Section 6.2, we compare methods *SME(RSS)*, *SME(Custom.)*, *S/M(RSS)*, and *S/M(Custom.)*. This time, each method is evaluated with respect to customizability feedback. This means that methods led by customizability feedback are evaluated as in Test 1. Instead, methods led by RSS evaluation are evaluated by considering as their best clustering that one obtained under RSS. The corresponding customizability evaluation is stored as the reference one. Then, the relative change is computed with respect to initial and referenced customizability evaluations. Afterwards, the average among all possible initial clusters is computed.

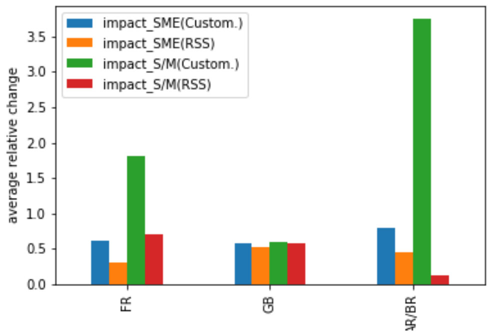

Fig. 4: Method impacts measured by average relative change of customizability. Results are shown by dataset.

Dataset	Number of Initial Clusters					
	2	3	4	5	6	7
FR	0.072	0.208	0.208	0.180	0.221	0.251
GB	0.565	0.360	0.379	0.466	0.447	0.357
AR/BR	0.183	0.141	0.264	0.257	0.342	0.272

Table 2: Average initial customizability evaluations by number of initial clusters

As expected, we found methods *SME(Custom.)* and *S/M(Custom.)* to be more effective than the others since they are led by the feedback we want to optimize. In particular, method *S/M(Custom.)* outperforms the others while *SME(Custom.)* has results more similar to the methods driven by RSS. Compared to dataset expected relative changes in Table 1, we register impacts higher than expected in all cases but for *S/M(RSS)* over dataset AR/BR.

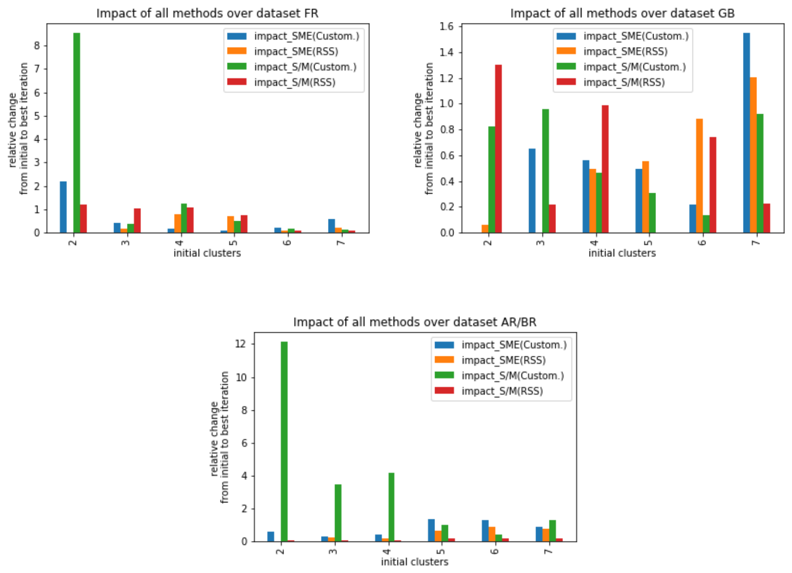

Fig. 5: Customizability relative changes by initial number of clusters k

In Figure 5, we show our results for each dataset as the number of initial clusters k varies. This detailed view allows us to notice a high variance in terms of both datasets and number of clusters k within the same dataset. Indeed, dataset GB presents very low relative changes compared to the others probably due to higher initial evaluations as shown in Table 2.

We see that most of peaks corresponds to S/M(Custom.) method. However, the GB dataset shows a more various behavior and its best relative change is obtained by SME(Custom.) over 7 clusters. Remarkably for $k = 6$, we have the two RSS-driven methods getting better results than the customizability-driven ones. The results in Figure 5 might be helpful to determine the number of optimal clusters k to subdivide a given dataset. However, we recall that for methods S/M, the setting of $k = 2$ does not mean that the best found clustering contains 2 clusters. For instance, for dataset GB with initial clusters $k = 2$, the best clustering contains 5 clusters although the process started with 2. It means that, in that case, the split phase was called more times than the merge one. The S/M(Custom.) bar for $k = 2$ is significantly higher that all bars at $k = 5$, in particular those for the SME method. This suggests that, in that particular case, we were able to find a better segmentation into 5 parts by starting with 2 clusters than those obtained by recombining 5 clusters. This is one of the strength points in the S/M method over the SME one. For dataset AR/BR, we observe remarkable peaks for $k = 2, 4$ and method S/M(Custom.). The corresponding best found clusterings contain 6 clusters each where a particular cluster has been well-evaluated. The fact the the bar for $k = 2$ is higher that $k = 4$ is due to the initial evaluation being different (see Table 2). This suggests a more local behavior of the S/M method compared to SME in finding particularly well-behaved clusters, whereas the SME method seems to reward the overall improvement.

7 Conclusions

In this paper, we have presented how feedback clustering techniques can be applied to a specific use case: recommendations for online flight searches.

In particular, we have shown how our use case can be formalized within the feedback clustering framework. This has been achieved by considering two cluster evaluations as our feedbacks independently taken for each cluster. The former is the RSS index, that is a well-known deterministic cluster index measuring how points within a cluster are spread apart from the cluster centroid in average. The latter has been introduced in this work and called customizability. Customizability is a non-deterministic value which is domain-specific to flight searches. Indeed, customizability measures how much a specific optimization process we exploit in our use case is efficient over some cluster. The optimization process is meant to assign a specific customer behavior (schematically: business, leisure traveller, family group, etc.) to some cluster by learning it from flight searches only. Customizability measures how satisfying is the customer behavior detected by the optimization process in terms of efficiency of the corresponding flight recommendations found.

We detected the Split-Merge-Evolve (SME) method introduced in [17] as a suitable one to be tested. Indeed, it is very flexible in terms of the core clustering algorithm in use (here it is the k-mean algorithm) and it acts on any numerical feedback being independent over each cluster (required by our use case). Based on that, we defined a clustering evaluation depending on each cluster feedback as the average cluster feedback evaluations weighted with cluster sizes.

Our first contribution has been to compare the effectiveness of a domain-specific feedback (customizability) over a standard cluster index (RSS). In order to do so, we first tested the SME method implemented with the RSS and the customizability feedbacks over 3 heterogeneous datasets of flight searches. We measured how much the SME method improves the global clustering feedback, depending on the chosen feedback. Results in Section 6.2 confirm effectiveness of the SME method for our domain for both feedback choices. For the RSS feedback, results confirmed the efficiency of the SME method already obtained in [17] for the case of synthetic data. For the customizability feedback, the impact is higher than for RSS. In our opinion, this is partially due to the customizability feedback being purely external whereas the RSS index improvement is already part of the k-mean algorithm's target. This provides a real-world use case where the feedback clustering framework has been successful. However, the two impacts had the same order of magnitude and this confirms the flexibility of the SME method under feedback evaluation changes. Moreover, this, in lack of a solid clustering ground-truth for a domain-specific feedback, strengthen the reliability our results.

Secondly, in Section 6.3, we evaluated SME method under the RSS and the customizability index in improving customizability, specifically. As expected, we found in average better results with the customizability feedback. However, for particular choices in the number of initial clusters per dataset (5 out of 18), our results presented better improvement under the RSS instead of the customizability feedback. In general, our results provide an instance of effectiveness of a domain-specific feedback over a standard cluster index.

As a second contribution, we introduced a variation of the SME method called S/M and tested it. S/M is theoretically as flexible as SME in terms of cluster feedback to drive it. The S/M method differs from SME only in the way clusterings are altered from one iteration to another. Specifically, the worst cluster under the feedback evaluation is either split or merged to the closest one according to its size in terms of number of points. In Section 6.3, we compared the customizability improvement obtained under the S/M method to the SME method. Our results showed that S/M behave like SME in presenting better results when driven by customizability feedback rather than by RSS. As for the comparison in between S/M and SME, the former had average better results over all datasets with a remarkable gap for 2 datasets out of 3. We suggest that this behavior is probably due to S/M acting more locally than SME. Indeed, on our domain, the combination of split and merge operators in S/M seems more suitable to isolate clusters with bad feedbacks or to favoring clusters with very good scores. Other domains might prefer the SME method where this polarized effect is tamed.

Ongoing work directions include the followings. First, we are setting further tests to compare possible clustering evaluations based on cluster customizability other than taking the average weighted on size. This would help us in rewarding single cluster good peaks. Secondly, we are designing new split and merge combinations so that to better handle the fluctuation of customizability values for each cluster. For instance, we could consider multiple runs of split and merge operators at each iteration. Lastly, we are working on comparing the impact on flight recommendations of SME and S/M to other frameworks such as ensemble clustering [6]. This would provide further tests for the effectiveness of customizability index, specifically and as a domain-specific feedback.

References

1. Awasthi, P., Balcan, M.F., Voevodski, K.: Local algorithms for interactive clustering. Journal of Machine Learning Research **18**, 1–35 (2013)
2. Balcan, M.F., Blum, A.: Clustering with Interactive Feedback. In: Freund Y., Györfi L., Turán G., Z.T. (ed.) International Conference on Algorithmic Learning Theory, Lecture Notes in Computer Science, vol. 5254, chap. ALT 2008, pp. 316–328. Springer, Berlin, Heidelberg (2008)

3. Balcan, M.F., Blum, A., Vempala, S.: A discriminative framework for clustering via similarity functions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. p. 671. STOC (2008)
4. Basu, S., Banerjee, A., Mooney, R.J.: Active Semi-Supervised for Pairwise Constrained Clustering. In: Proceedings of the SIAM International Conference on Data Mining. pp. 333–344. SDM (2004)
5. Caruana, R., Elhawary, M., Nguyen, N., Smith, C.: Meta Clustering. In: Sixth International Conference on Data Mining. pp. 107–118. ICDM'06, IEEE (dec 2006)
6. Chatterjee, S., Pasquier, N., Nanty, S., Zuluaga, M.A.: Multi-objective Consensus Clustering Framework for Flight Search Recommendation. ArXiv preprint (feb 2020), arXiv:2002.10241
7. Dasgupta, S., Ng, V.: Which clustering do you want? Inducing your ideal clustering with minimal feedback. Journal of Artificial Intelligence Research **39**, 581–632 (2010)
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)
9. Eriksson, B., Dasarathy, G., Singh, A., Nowak, R.: Active Clustering: Robust and Efficient Hierarchical Clustering using Adaptively Selected Similarities. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. PMLR, vol. 15, pp. 260–268 (2011)
10. Jain, A.K., Mallapragada, P.K., Law, M.: Bayesian Feedback in Data Clustering. In: 18th International Conference on Pattern Recognition. ICPR'06, vol. 3, pp. 374–378. IEEE (2006)
11. Klein, D., Kamvar, S.D., Manning, C.D.: From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering. In: Proceedings of the Nineteenth International Conference on Machine Learning. p. 706. Morgan Kaufmann Publishers (2002)
12. Mottini, A., Lhéritier, A., Acuna-Agost, R., Zuluaga, M.A.: Understanding customer choices to improve recommendations in the air travel industry. In: CEUR Workshop Proceedings. vol. 2222, pp. 28–32 (2018)
13. Quan, W., Zhou, Q., Nan, H., Chen, Y., Wang, P.: A user-satisfaction-based clustering method. In: Proceedings of 2018 International Conference on Mathematics and Artificial Intelligence. pp. 56–6. ICMAI, ACM Press, New York, New York, USA (2018)
14. Settles, B.: Active Learning Literature Survey. In: CS Technical Reports. University of Wisconsin-Madison Department of Computer Sciences (2009)
15. Srivastava, A., Zou, J., Adams, R.P., Sutton, C.: Clustering with a Reject Option: Interactive Clustering as Bayesian Prior Elicitation (2016), arXiv:1606.05896
16. Vega-Pons, S., Ruiz-Shulcloper, J.: A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence **25**(3), 337–372 (2011)
17. Wang, M., Huang, V., Bosneag, A.M.C.: A novel split-merge-evolve κ clustering algorithm. Proceedings - IEEE 4th International Conference on Big Data Computing Service and Applications, BigDataService 2018 pp. 229–236 (2018)
18. Wang, X., Davidson, I.: Active spectral clustering. In: Proceedings - IEEE International Conference on Data Mining. pp. 561–568. ICDM, Sydney, NSW, Australia (2010)