Preference Mining in the Travel Domain

Alireza Javadian Sabet*[0000-0001-9459-2411], Sankari Gopalakrishnan‡[0000-0002-7738-6947],
Matteo Rossi†[0000-0002-9193-9560], Fabio A. Schreiber*[0000-0002-5237-0455] and Letizia Tanca*[0000-0003-2607-3171]
* ‡ Dipartimento di Elettronica, Informazione e Bioingegneria
† Dipartimento di Meccanica
Politecnico di Milano, Piazza Leonardo da Vinci 32, I-20133 Milano, Italy
* † Email: firstname.lastname@polimi.it ‡ firstname.lastname@mail.polimi.it

***Abstract*—Personalization of user experience through recommendations involves understanding their preferences and the context they are living in. In this work, we present a method to rank travel offers returned in response to a travel request made by a user. To give a sensible answer, we learn users' preferences over time and use them to understand travelers' needs. Our solution is based on a data-mining-based recommender system. We first design a database of historical traveler data and populate it with data generated according to rules mimicking the features of actual user profiles. These rules are then used as ground truth to validate the accuracy of the proposed learning algorithm. After performing data pre-processing, a knowledge base is set up by mining association rules from the database, which will then be used along with the travel request to assign a score to each of the potential travel offers, thus ranking them. To test the proposed methodology, we generate synthesized data according to some distributions. The results of the experiments approve the effectiveness of the proposed ranking mechanisms. Finally, we demonstrate the presentation of the ranked offers to the user via some mock-ups of the intended application.**

***Keywords*—Context-awareness, Preferences, Personalization, Travel, Data Mining, Recommender System**

I. Introduction

With the abundance of data that is available from various sources, personalization of applications and services has become the need of the hour [1]. Recommender Systems, as a means of personalization, offer a suggestive mechanism to keep users engaged [2] while retrieving information from them based on their activities. The information used to personalize the user experience can be static profile data, as well as the evolving preferences of the user. Moreover, the user's environment and conditions play a role in filtering the information presented to them. The level of user engagement and personalization increase over time, as the system improves its knowledge of the user and its ability to correctly predict their choices [3].

This research is performed within the RIDE2RAIL (R2R) [4] project, in the frame of Innovation Programme 4 (IP4) of the Shift2Rail (S2R) initiative [5]. S2R's goal is to build a collaborative ecosystems through its Interoperability Framework [6] that offers a variety of modules such as data conversion approach [7], automated mapping [8] [9], and ontology management [10]. The ecosystem facilitates the interoperability among all the IP4 services (e.g., Booking, Journey Planning, etc.) and travel service providers (TSPs) and allows them to interact with each other, share data, and build up more complex services together. Among the services, the so-called Travel Companion (TC) enables travelers to fulfill their mobility requests and drivers to offer their rides to other travelers.

Objectives: The goal of this work is to provide a personalized set of travel offers to users of the TC, based on their context and preferences. The recommender core of the TC proposed in [11] takes as input user data, knowledge models, and service-related information and accordingly presents a list of travel offers ranked according to the user's contextual preferences. To enable the TC to capture users' preferences, Javadian et al. [12] designed a model of preferences using the context dimension tree methodology [13].

This work aims to design and implement the TC's recommender core using contextual preferences to rank travel offers. The solution is to implement a data mining-based recommender system. We start by designing a database of historical traveler data, including travel offers. The database is then populated using data generation rules that enforce some desired features in the data by fixing possible values and applying constraints. After pre-processing the data, the TC's knowledge base is enriched by adding association rules mined from the database. The rules are designed to address new users with no past preferences or past travel history. The knowledge base will then be used—along with the travel request—to compare and assign a score to each travel offer. The computed score is used to rank the offers when they are presented to the traveler. The knowledge base of association rules works to mine the user preferences and understand the traveler's needs.

The paper is structured as follows: Sect. II overviews some relevant related works; Sect. III discusses the steps to build a data mining based ranking module; Sect. IV presents the results of the validation, and Sect. V concludes.

II. Related Work

This section provides the state-of-the-art in the design of recommender systems for travelers.

Recommender Systems in the travel industry help to cope with the personalized mobility demand [14].

Venkatesh and Jabez [3] implement a system using collaborative, hybrid recommendation approach [15]. The work discusses the Tourist-Area-Season-Topic (TAST) model and extends it by adding the relationship between tourist groups. It divides the tourists into groups by following K-means clustering algorithm [16] and provides a targeted recommendation.

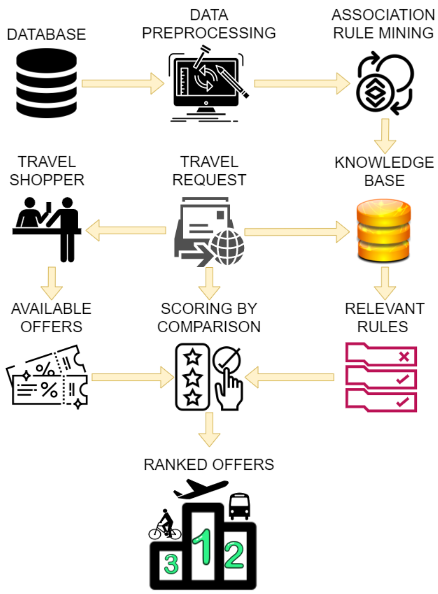


Fig. 1: The high-level architecture of the proposed ranking mechanism.

Another application of the hybrid group recommender system proposed in [17] where they provide destination suggestions based on user preferences and past ratings as feedback. It uses the freely available data on travel, WikiVoyage [18] as its main information source supported by popularity rating from Tripadvisor. Group recommendation [19] combines profiles from a family or a group of friends to leverage all their information specifications. The work in [20] uses a recommendation algorithm to combine multiple travel destinations into one trip while considering budget and time constraints.

III. Methodology

This work presents a system to rank the travel offers such that they are ranked personally for each traveler based on their contextual preferences. A new traveler's recommendations will be based on other travelers' historical data with similar profiles and preferences. The system also aims to update the preferences over time. Hence, the solution is to build a recommender system using *data mining* techniques that provide a knowledge base that will be used along with the travel request to assign a score to each of the candidate travel offers. The scores are then used to rank the offers and provide them to the traveler. Fig. 1 presents the high-level procedure of the proposed ranking mechanism. Sect. III-A details the database design; Sect. III-B details the rule mining to build the knowledge base; Sect. III-C presents the ranking mechanism.

A. Historical Database Design

The database include information about the traveler's profile, past preferences, travel requests, potential travel offers and characteristics, and the final choice made with each request.
Traveler's profile encompasses the name, age, mobile number, gender, education, occupation, marital status, card details (membership/debit/credit), and citizenship.
Preference consists of health issues, accompanying person and luggage information, and service and meal preferences.
Travel request contains information about the travel purpose, source, destination, role (i.e., passenger or driver), and frequency of the trip if recurring.
Travel offer information consists of an identifier to map it to the request, source, destination, number of segments, duration, price, and discount. It also includes information about pets being allowed in the journey.
Offer characteristics includes the special facilities it includes, service type, meal availability, and luggage facility.

Moreover, each request is also linked to the final choice of offer that the traveler made.

After designing the Entity Relationship (ER) Diagram, we converted it to Relational Schema. Figure 2 shows the Relational schema of the designed database. It represents data as a set of related tables. The entities and relationships are both written as tables. The conversion from ER Schema to Relational Schema helps in the next step of implementing the database. The tables, attributes, primary keys, and foreign keys are all understood from the relational schema. It also provides an idea of the database's size, which plays a major role in deciding the technology for the database.

TRAVELLER(UserID,MobileNumber,Name,Age,Education,Gender,Occupation, MaritalStatus,Citizenship)
CARD(CardNumber,CardType,Username,ExpiryDate,UserID)
REQUEST(RequestID,Purpose,Role,Leg,Recurring,Source,Destination,Interface)
USER_TRAVEL_REQUEST(UserID,RequestID,Timestamp,OfferID,CardNumber)
SERVICE_PREFERENCES(UserID,ServiceID,Timestamp)
MEAL_PREFERENCES(UserID,MealID,TimeStamp)
LUGGAGE_PREFERENCES(UserID,LuggageID,Timestamp)
HEALTH_ISSUES(HealthID,IssueType,Severity,Category,Aid)
USER_HEALTH(UserID,HealthID,StartDate,EndDate)
ACCOMPANYING(AccompanyID,Category,Type,Number)
USER_PARTNER(UserID,AccompanyID,Timestamp)
SERVICE(ServiceID,Type,Condition,Class)
MEAL(MealID,Mealtype,Excluded)
LUGGAGE(LuggageID,LuggageType,Number)
TRAVEL_OFFER(OfferID,TagID,Source,Destination,Duration, Number_segments, Pets allowed,Price,Discount)
SPECIAL_FACILITIES(FacilityID,Type,Detail)
OFFER_DETAIL(OfferID,Legnum,ServiceID,MealID,LuggageID,FacilityID)

Fig. 2: Relational Schema of the Database design.

B. Knowledge Base

Recommender Systems employ rich knowledge bases of users' preferences and behavioral characteristics. A knowledge base can be a logical model, or it is built from rules mined from historical data. In this work, we follow the latter. The rules, in turn, can be mined from the user's history or, as in the current work, from other users with similar profiles/preferences. The dependency on the existence of historical data is known as the "cold start problem" [21]. Our system overcomes this by mining association rules from the travelers' dataset. The recommendation has to be handled in multiple situations in the system, and the knowledge base should be accordingly diverse [22]. Also, the representation and the data format of the rules play a vital role in the system's efficiency.

The association rules should address both travelers whose preferences are not known/specified, and travelers without a past purchasing history to understand their travel choices. The information available from the dataset is the static profile data, traveler preferences, health issues, travel choices, and the facilities of corresponding travel offers.

The required formats of such rules are as follows.

1) **RULE TYPE 1:** Profile $\implies$ Preferences
Given a traveler's profile like age, gender, and occupation, we predict their preferred service, meal, etc.
2) **RULE TYPE 2:** Preferences $\implies$ Offers
Given a traveler's preferences in the request, we predict their preferred offers (based on its facilities).

Both rules together cover the significant part of personalizing the travel experience for a traveler. They suggest the possible preferences and possible travel choices that would be essential for a new user. Once the rules' type was decided, the next step is to implement the association rule mining on the dataset. The process is composed of the following steps.

- Table joins along with preprocessing to serve as the input for the algorithm.
- Generate rules using the algorithm.
- Visualizing and validating the results.

1) Table Joins: To unify the users' profile and preferences into one table, *Traveler, Travel Request, Request, User Health, User Partner, Service Preferences, Luggage Preferences,* and *Meal Preferences* tables were joined to create one table called **UserProfile**. The timestamp was used to retrieve the preference closest to the request (for Rule type 1). In order to facilitate frequent itemset mining, preference options and age were converted to categorical data. For example, age < 30 as young adult or age > 60 as elderly.

In order to get the users' preferences and travel choices together, *Traveler, Travel Request, Request, User Health, User Partner, Service Preferences, Luggage Preferences, Meal Preferences, Travel Offer,* and *Offer Detail* tables were joined to create one table called **UserOffer**. Different segments' facilities are concatenated together (for Rule type 2).

The created tables, representing the transactions, are then fed into the apriori algorithm.

2) Generate rules using the algorithm: Implementing the same algorithm using different programming languages could offer insight into the computational time, the storage, simplicity of the syntax, and also the results. The final result mainly differs in the rules' template, the data type in use, quality metrics, and their visualization. For our experiments, the two languages in the discussion are Python and R programming. Both are well known for their simplicity of the language and specificity to data analysis and mining. Moreover, they have respective modules for implementing association rule mining via apriori algorithm [23]. For brevity, here, we discuss the implementation and results based on R.

`arules` [24] is a package to represent, manipulate, and analyze transaction data and patterns (frequent itemsets and association rules). It differs from the association rule mining in Python [25] with its ability to apply rule templates.

The *UserProfile* and *UserOffer* tables were used as input. *UserProfile* to mine preferences with profile and *UserOffer* to mine offer choices with preferences. The tables can be read as a csv file and then converted into *'transactions'* class.

We set a minimum support of 10% and confidence of 50%. `arules` also allows setting the rule templates according to our required format using the 'appearance' parameter. Both the antecedents and consequents can be restricted to follow a pattern and the rest will be filtered out. The pattern can be a direct match filter or it could be specified by a regular expression. By default, `arules` allows only one item in the consequent. This could be changed by *minlen* parameter, but for our system, having one item as the consequent makes it easier in the scoring process. It also has a subset feature that can be used to filter out redundant rules as longer rules can imply the shorter rules with fewer items. However, in our case, we keep all the rules as every match of rule consequent will help to rank the offers. The mined rules can now be retrieved to see the top few rules or be visualized by plotting them. The antecedents and consequents of the rules and their quality metrics can also be retrieved separately as a list or item matrix.

The rule template was applied successfully, and the rules followed the target format. However, on close observation, the rules did not suggest much information. It provided information only on specific traveler preferences and features of the offer. For instance, (purpose=work $\implies$ luggageid=bag), (mstatus=single $\implies$ accid=pet), (age=elderly $\implies$ accid=person), (pref_serviceid=car $\implies$ offer_accid=yes).

This led to the realization that only some of the features were always filled with data, and most of them were missed out as the data population code had included NULL values to make it close to reality.

To generate a proper knowledge base filled with helpful information, it is necessary that the data is complete and provides enough to mine rules that can provide suggestions on all features of the data items. Hence, the *UserOffer* and *UserProfile* tables were further pre-processed to include more traveler profiles and their travel choices and make the dataset suitable for association rule mining. The plot's choice is according to the application. In our scenario, the 'graph' and 'paracoord' bring out the rule characteristics in the best way.

Observing the final rules helps us validate the process of creating the knowledge base, which implies that in the presence of actual historical traveler data, the predictions will be precise in guiding their recommendations and providing a personalized experience. This set of rules has to be updated periodically to keep it updated with the incoming data, and the period should be determined based on the system demand.

C. Ranking

Recalling our goal for the system, to rank the travel offers such that it is ranked personally for the traveler. To achieve this, we use the rules mined in the previous step to rank the travel offers. The ranking module contains the following steps.

1) Compare preferences in travel requests and in the rule antecedents to get the relevant rules.

2) Use the rule consequents (facilities) to rank the available travel list by how precise the facilities match.

1) Preference Vector Comparison: From the mined rules' knowledge base, we need to pick out the relevant rules for the travel request. This gives the rule consequents required for the next step. Comparing the preferences mentioned in travel requests and preferences present in the rule antecedents compares two preference items lists. The cases are as follows.

Presence of complete vector: The request preferences are entirely present in the rule. It could be a single item or multiple items, and their order does not matter. All the preference items' presence implies that the rule is completely relevant and should be picked for the ranking process.

Partial Presence: The request is partially present in the rule, i.e., the rule could include additional preference items, or the request could have additional preference items. Still, the rule is partially relevant and can be picked for the ranking process based on how well they match. An indicator of relevance could be the match level between the preference vectors.

Since the order does not matter, we need a comparison based on the items. The solution is **Set Comparison** to retrieve the rules that are relevant to the request. The match level is also valuable for picking only the top few relevant rules for scoring the offers later. In the code, the set comparison works by intersection and union to get match level. Some examples of the retrieved relevant rules for a travel request are as follows.

For a preference vector like: ("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag"), the following rule item is a complete match: (pref_healthid=visual, pref_mealid=vegeterian) $\Longrightarrow$ (offer_serviceid=airplane), and the following rule item is a partial match with match level 1: (pref_accid=pet, pref_mealid=vegeterian) $\Longrightarrow$ (offer_facility=walking).

Conditions for set comparison: To implement the various comparison scenarios, we need conditions based on which the relevant rules can be retrieved. These compare the rule items from the knowledge base and preference vector from the request and calculate how well they match.
Perfect match: Matchlevel=len(PreferenceVector)=len(RuleItem)
Complete presence: Matchlevel=len(PreferenceVector)
Partial: Matchlevel=len(RuleItem)
Matchlevel $>$ len(PreferenceVector)/2 AND (len(RuleItem) - Matchlevel) $<$ len(PreferenceVector)/2

2) Scoring List of Offers: The aim is to rank the travel offers that have been received for the current request. The top relevant rules that were picked in the earlier step will provide a set of rule consequents. These are basically suggestions for some facilities that should be present in the travel offers. Moreover, the travel offer should match with the request as well. The traveler while providing the preferences in the request, can add an additional priority number. This priority is useful in overriding the suggestion provided by the rules and thus, preferences are more personalized than guided by suggestions made from popularity or past history. Hence the scoring is based on a comparison between *rule consequents*, *preference given by user* and *facilities in the offer*.

Scoring strategy: Each rule consequent adds a point to the matching offer multiplied by confidence (only when the lift $>= 1$). This is repeated with all the rule consequents to get a final *rule_score* by adding all *item_score*. The preference vector adds a point to the matching offer multiplied by the priority. Finally, the overall score is the sum of *rule_score* and *preference_score*. Table I provides the code snippet showing the implementation of the scoring of potential travel offers in R. Each of the offers was compared with the top relevant rules and scored with the corresponding rules' quality metrics.

TABLE I: Code snippet of comparing rule consequents and request preference to available offers and scoring them.

```
for (offer in avail_offers){
  score=0
  for (row in 1:nrow(rulesdf)){
    relevance=length(intersect(offer,
    rulesdf[row, "Rule_Consequent"]))
    lift=rulesdf[row, "Lift"]
    conf=rulesdf[row, "Confidence"]
    if(lift>1){
      formula=relevance * conf
      score=score+formula}}
  prefscore=length(intersect(substring(offer,
  regexpr("=", offer) + 1),
  substring(req_pref, regexpr("=", req_pref) + 1)))
  pref_priority=sample(1:3, 1)
  print(offer)
  print(score+prefscore * pref_priority)}
```

IV. Validation

In order to test the proposed system, we need an existing dataset. At first, we attempted to find a suitable publicly available dataset. Although the datasets available had some information about travelers in Europe and public transport facilities, we encountered some system compatibility issues.

We then decided to synthesize our dataset, and the resulting database can be further extended as the system backend once the travelers are acquainted with the system. Also, knowing the distribution of the dataset allows us to validate the results.

A. Data Population

The strategy is to split the process into two steps to populate the tables according to the database design (see Sect. III-A).

1) Populate Independent Tables: Independent tables include Traveler, Card, Health Issues, Accompanying, Request, Service, Meal, Luggage, Travel Offer and Special Facilities. To populate them, we need some rules to avoid invalid data (e.g., a male traveler with the pregnancy as a health condition). Also, the data should be directed/trained in following a direction and some predefined characteristics that could be used to validate the results. For brevity, we do not provide details about the rules and possible values for the features. But, it worth mentioning that we synthesized the dataset such that all the possible values for each feature are uniformly distributed.

2) Populate Relationship Tables: Relationships tables include User health, User partner, Membership, Service preferences, Meal preferences, Luggage preferences, Travel request, and Offer detail. The possible values are as follows.

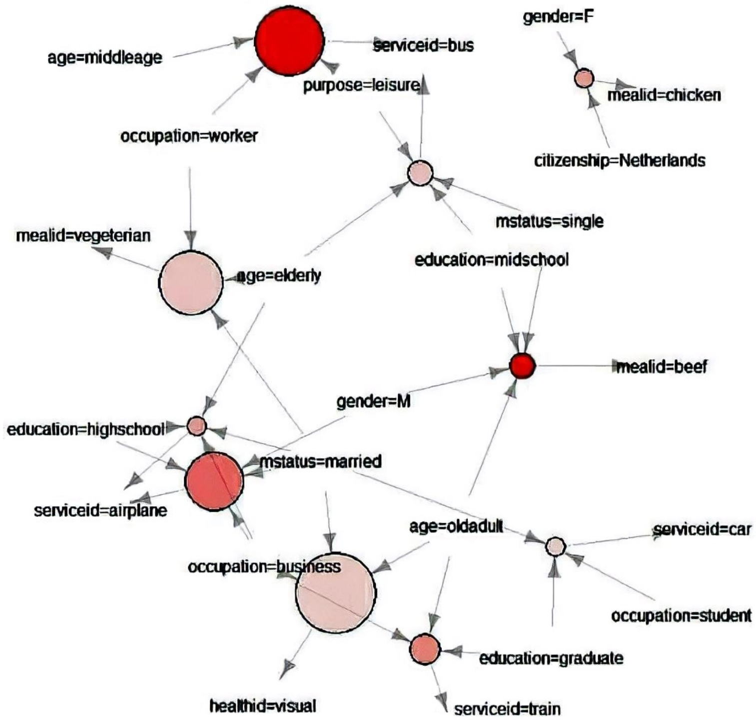


Fig. 3: Graph representation of the top 10 association rules mined from User Profile. Size of the nodes represent support and the color corresponds to lift.

Travel Request: Different types of request should be generated for each traveler, differing either by purpose or source/destination values.

User Health: Each health issue can be added in a single severity level with a single aid.

User Partner: The aid type can be added only with health issues; else, the partner should be a family, friend, or pet.

Service, Meal, Luggage Preference: To cover all types of preferences for each traveler and avoid repetitions.

Offer Detail: For each offer, and for each of their segments, service type, meal availability, luggage facility, and special facility details should be added. They are grouped into one table as they are tied together with the offer.

B. Association Rules Results

The populated dataset was a collection of data based on the combination of $1,000$ traveler profiles, $2,000$ travel requests and $3,000$ travel offers. Consequently, UserProfile data generated 4000+ rules and UserOffer generated 600+ rules.

Figure 3 presents the graph visualization of the top rules obtained from the experiment. The nodes' size represent support and the color corresponds to the lift. For example, (age=middleage, occupation=worker, purpose=leisure $\Longrightarrow$ serviceid=bus), (age=elderly, occupation=worker, mstatus= married $\Longrightarrow$ mealid=vegeterian), and (age=oldadult, occupation =business, education=graduate $\Longrightarrow$ serviceid=bus).

Figure 4 presents the parallel coordinate visualization of the top rules. The width of the arrows represents support and the intensity of the color represent confidence. For example, (purpose=leisure, occupation=worker, age=middleage $\Longrightarrow$ serviceid=bus) and (mstatus=married, occupation=business, purpose=leisure $\Longrightarrow$ serviceid=airplane).

Figures 3 and 4 are essentially showing practically sensible profiles and preferences. A middle-aged worker traveling for leisure prefers a bus; an elderly married worker prefers a vegetarian meal; a businessman prefers an airplane. The graph plot allows visualizing the rules with good readability, while the paracord plot gives more insight into the support and confidence quality metrics.

The same procedure has been applied to the rules obtained from the UserOffer table, which, for brevity, are omitted.

C. Ranking Results

Sect. III-C1 explained the conditions to choose the top relevant rules from the knowledge base that can be later used for scoring. To demonstrate our example, we ran these conditions on the knowledge base. Table II shows the results. These rule consequents give insights into the facilities that the potential offers should include to be best suited for the user. The overall calculated score is then used to rank the travel offers that can be presented to the user.

A sample run with a travel request and a set of potential travel offers were ranked by the system and Figure 5 shows the results. It includes intermediate results of the relevant rules picked from the knowledge base and the final list of ranked offers. The ranking in this example is mainly influenced by the rules mined and stored in the knowledge base.

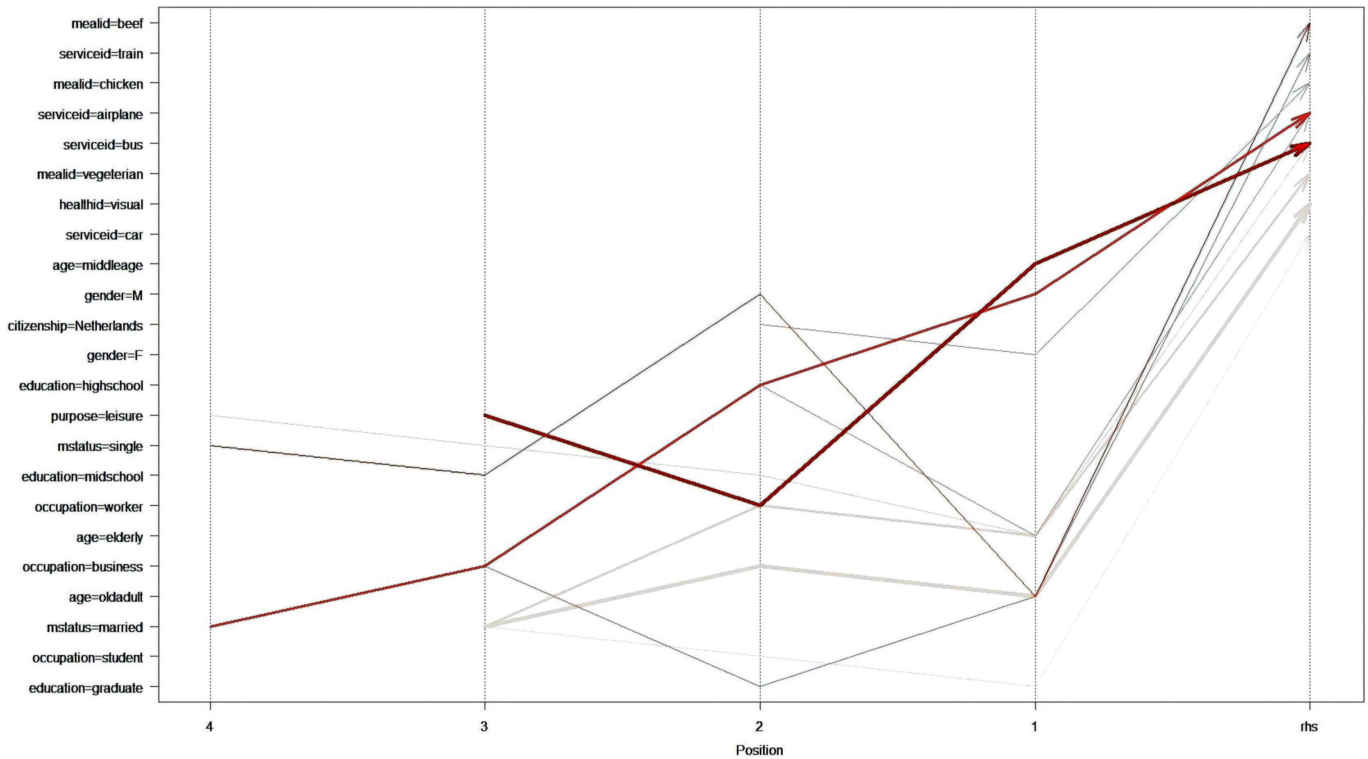
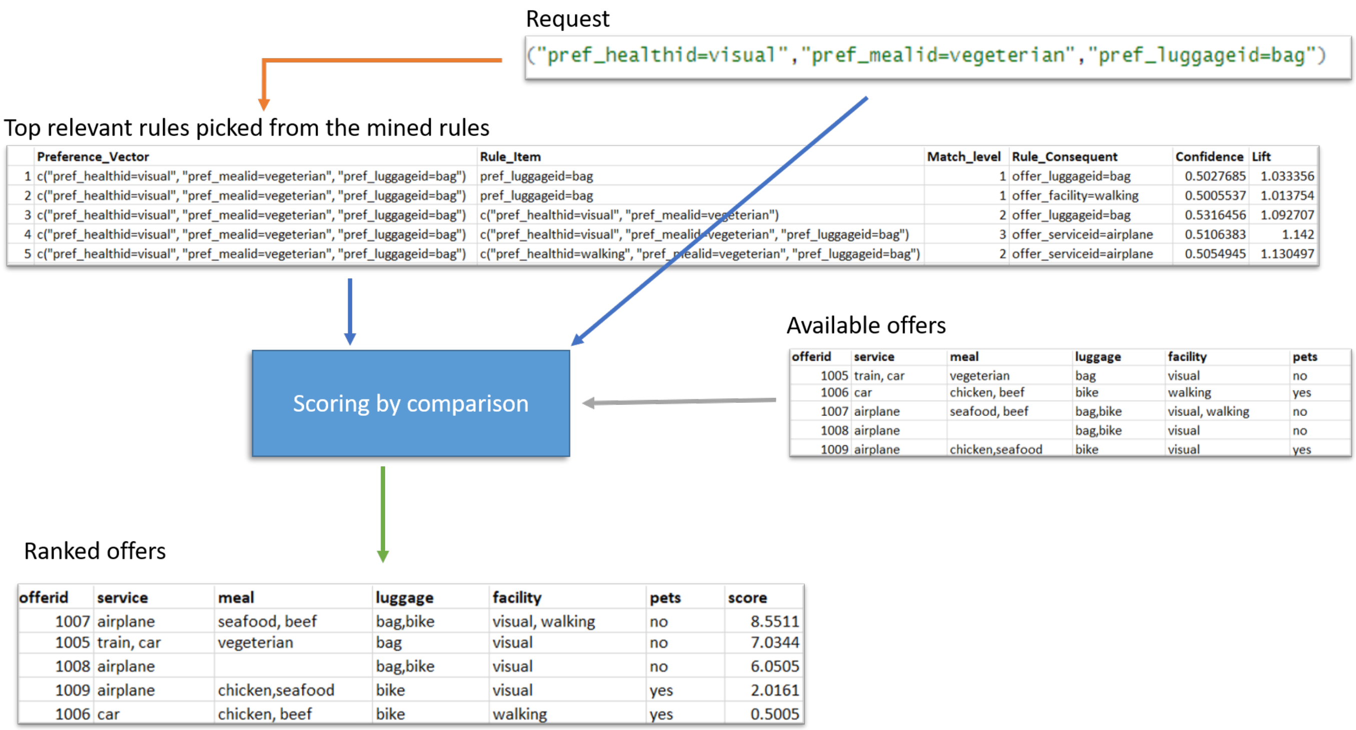


Fig. 4: Paracord representation of the top 10 association rules mined from User Profile. The width of the arrows represents support and the intensity of the color represent confidence.



	Preference_Vector	Rule_Item	Match_level	Rule_Consequent	Confidence	Lift
1	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	pref_luggageid=bag	1	offer_luggageid=bag	0.5027685	1.033356
2	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	pref_luggageid=bag	1	offer_facility=walking	0.5005537	1.013754
3	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	c("pref_healthid=visual", "pref_mealid=vegeterian")	2	offer_luggageid=bag	0.5316456	1.092707
4	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	3	offer_serviceid=airplane	0.5106383	1.142
5	c("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag")	c("pref_healthid=walking", "pref_mealid=vegeterian", "pref_luggageid=bag")	2	offer_serviceid=airplane	0.5054945	1.130497

offerid	service	meal	luggage	facility	pets
1005	train, car	vegeterian	bag	visual	no
1006	car	chicken, beef	bike	walking	yes
1007	airplane	seafood, beef	bag,bike	visual, walking	no
1008	airplane		bag,bike	visual	no
1009	airplane	chicken,seafood	bike	visual	yes

offerid	service	meal	luggage	facility	pets	score
1007	airplane	seafood, beef	bag,bike	visual, walking	no	8.5511
1005	train, car	vegeterian	bag	visual	no	7.0344
1008	airplane		bag,bike	visual	no	6.0505
1009	airplane	chicken,seafood	bike	visual	yes	2.0161
1006	car	chicken, beef	bike	walking	yes	0.5005

Fig. 5: Sample run of the ranking module that returns the ranked offers.

TABLE II: Sample results of picking relevant rules with match level.

rule	match level
"pref_luggageid=bag" $\implies$ offer_luggageid=bag	1
"pref_luggageid=bag" $\implies$ offer_facility=walking	1
("pref_healthid=visual", "pref_mealid=vegeterian") $\implies$ offer_luggageid=bag	2
("pref_healthid=visual", "pref_mealid=vegeterian", "pref_luggageid=bag") $\implies$ offer_serviceid=airplane	3
("pref_healthid=walking", "pref_mealid=vegeterian", "pref_luggageid=bag") $\implies$ offer_serviceid=airplane	2

D. Prototype

In this section, we present our solution via a prototype of the TC. We have built a prototype to envision the system to understand the user's input and how the results are presented in a personalized manner to suit their requirement.

Considering the following scenario: A trip from Milan to Bratislava on October 10th, 2020 for leisure. The user has visual disabilities, prefers an airplane, vegetarian meal and has luggage requirements of a bag. Figure 6 shows the list of travel offers for the travel request made. They are ranked according to the preferences made by the user and recommendations from the system knowledge base. The icons on the left depict the mode of transport, and the icons on the left show the special facilities available in the offer. It also includes the allowed luggage and meal availability. The *paw* icon symbolizes pets being allowed in the journey and the green leaf symbolizes the Eco-friendliness of the offer.

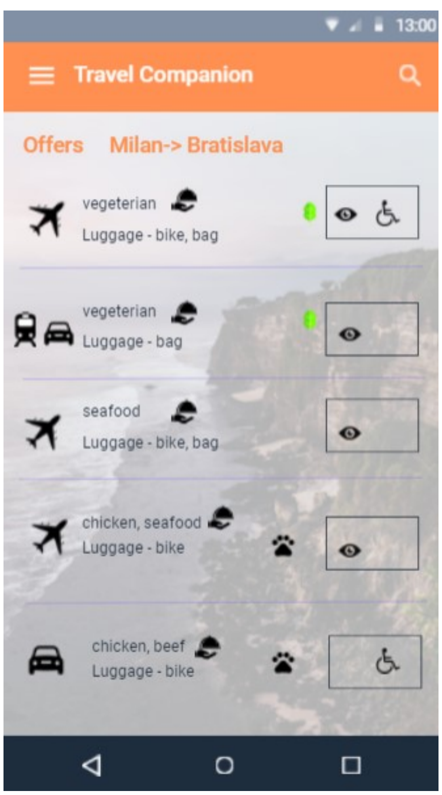


Fig. 6: The prototype of the Travel Companion: Travel Offers.

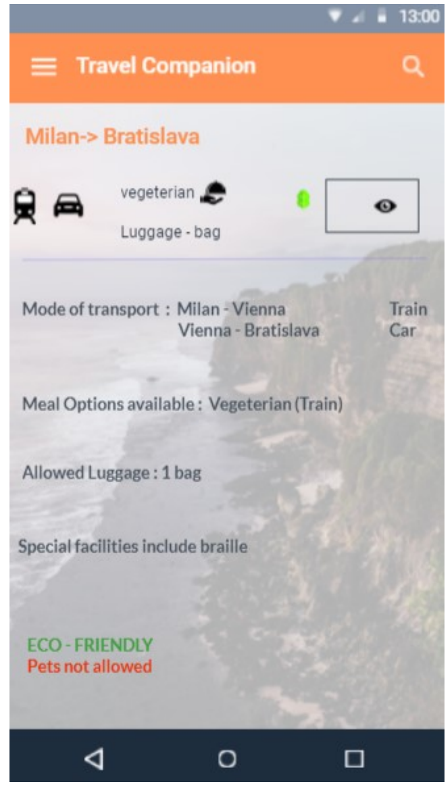


Fig. 7: The prototype of the Travel Companion: Detailed Offer Information.

Figure 7 shows the detailed offer information that can be seen when the user clicks on a particular offer. The sample is shown for the second offer on the offers page. It includes information regarding stopovers and the icons transformed into words for clarity, helping them make their travel choice.

V. Conclusion

In this work, we presented a solution to build a data-mining-based recommender system that provides a ranked list of travel offers, filtered and personalized by the user's context and preferences. To test the proposed solution, we synthesized a dataset close to reality. Finally, we demonstrated the user experience through a prototype. It is observed that the solution for the system presented is indeed suitable and can be further explored to enhance its feature set. The rule-based knowledge model's demonstration with the custom dataset shows that the algorithm would continue to learn the user's preferences and provide personalized suggestions on replacing them with actual data. The framework could be followed with enriched modules as an addition.

Acknowledgment

Work supported by Shift2Rail and the EU H2020 research and innovation programme under grant agreement No: 881825 (RIDE2RAIL)

REFERENCES

[1] S. Canale, A. Di Giorgio, F. Lisi, M. Panfili, L. R. Celsi, V. Suraci, and F. D. Priscoli, "A future internet oriented user centric extended intelligent transportation system," in *2016 24th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2016, pp. 1133–1139.

[2] N. Bendakir and A. Esma, "Using association rules for course recommendation," 2006.

[3] K. Venkatesh and J. Jabez, "Recommendation and ranking of travel package to tourist," *Contemporary Engineering Sciences*, vol. 8, pp. 401–407, 01 2015.

[4] "RIDE2RAIL." [Online]. Available: https://ride2rail.eu

[5] "Shift2Rail." [Online]. Available: http://www.shift2rail.org/

[6] M. Sadeghi, P. BuchnÃ■Ä■ek, A. Carenini, O. Corcho, S. Gogos, M. Rossi, and R. Santoro, "SPRINT: Semantics for PerfoRmant and scalable INteroperability of multimodal Transport," in *TRA*, 2020, pp. 1–10.

[7] A. Carenini, U. DellâĂŹArciprete, S. Gogos, M. M. P. Kallehbasti, M. Rossi, and R. Santoro, "ST4RT âĂŞ semantic transformations for rail transportation," in *TRA 2018*, 2018, pp. 1–10.

[8] M. Hosseini, S. Kalwar, M. G. Rossi, and M. Sadeghi, "Automated mapping for semantic-based conversion of transportation data formats," in *1st International Workshop On Semantics For Transport*, vol. 2447, 2019, pp. 1–6. [Online]. Available: http://ceur-ws.org/Vol-2447/paper7.pdf

[9] S. Kalwar, M. Sadeghi, A. Javadian Sabet, A. Nemirovskiy, and M. G. Rossi, "Smart: Towards automated mapping between data specifications," in *The 33rd International Conference on Software Engineering and Knowledge Engineering, SEKE 2021, KSIR Virtual Conference Center, Pittsburgh, USA, July 1-10, 2021*. KSI Research Inc. and Knowledge Systems Institute Graduate School, 2021, pp. 429–436. [Online]. Available: https://doi.org/10.18293/SEKE2021-161

[10] A. Alobaid, D. Garijo, M. Poveda-Villalón, I. Santana-Perez, A. Fernández-Izquierdo, and O. Corcho, "Automating ontology engineering support activities with ontoology," *Journal of Web Semantics*, vol. 57, p. 100472, 2019.

[11] A. Javadian Sabet, M. Rossi, F. A. Schreiber, and L. Tanca, "Towards learning travelers' preferences in a context-aware fashion," in *Ambient Intelligence – Software and Applications*. Cham: Springer International Publishing, 2021, pp. 203–212. [Online]. Available: https://doi.org/10.1007/978-3-030-58356-9_20

[12] A. J. Sabet, M. Rossi, F. A. Schreiber, and L. Tanca, "Context awareness in the travel companion of the shift2rail initiative," in *Proceedings of the 28th Italian Symposium on Advanced Database Systems*, ser. CEUR Workshop Proceedings, vol. 2646. CEUR-WS.org, 2020, pp. 202–209. [Online]. Available: http://ceur-ws.org/Vol-2646/15-paper.pdf

[13] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca, "Context information for knowledge reshaping," *International Journal of Web Engineering and Technology*, vol. 5, no. 1, pp. 88–103, 2009.

[14] G. Yoon, J. Y. Chow, A. Dmitriyeva, and D. Fay, "Effect of routing constraints on learning efficiency of destination recommender systems in mobility-on-demand services," *IEEE trans Intell Transp Syst*, 2020.

[15] R. Burke, *Hybrid Web Recommender Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 377–408. [Online]. Available: https://doi.org/10.1007/978-3-540-72079-9_12

[16] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[17] T. De Pessemier, J. Dhondt, K. Vanhecke, and L. Martens, "Travelwithfriends: a hybrid group recommender system for travel destinations," in *Work. touRS15, in conj. with the 9th ACM recsys*, 2015, pp. 51–60.

[18] "Wikivoyage." [Online]. Available: https://dumps.wikimedia.org/enwikivoyage/

[19] A. Felfernig, L. Boratto, M. Stettinger, and M. TkalÄ■iÄ■, *Group Recommender Systems - An Introduction*, 05 2018.

[20] D. Herzog and W. WÃűrndl, "A travel recommender system for combining multiple travel regions to a composite trip," *CEUR Workshop Proceedings*, vol. 1245, pp. 42–47, 01 2014.

[21] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. of the 25th ann. int. ACM SIGIR conf. on Research and dev. in IR*, 2002, pp. 253–260.

[22] D. Ai, H. Pan, X. Li, Y. Gao, and D. He, "Association rule mining algorithms on high-dimensional datasets," *Artificial Life and Robotics*, vol. 23, no. 3, pp. 420–427, 2018.

[23] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. VLDB*, vol. 1215, 1994, pp. 487–499.

[24] M. Hahsler, B. Gr, and K. Hornik, "Introduction to arules âĂŤ mining association rules and frequent item sets," 2006.

[25] S. Raschka, "Mlxtend: Providing machine learning and data science utilities and extensions to pythonâĂŹs scientific computing stack," *The Journal of Open Source Software*, vol. 3, no. 24, Apr. 2018. [Online]. Available: http://joss.theoj.org/papers/10.21105/joss.00638