Universitat Politècnica de Catalunya

Supervised and Experiential Learning

Practical Work 3

A CBR prototype for a synthetic task: planning / design / configuration in a concrete domain

---

Authors:

Joao Agostinho Valerio

Eirik Armann Grytøyr

Clara Rivadulla Duró

John Kelly Villota Pismag

Spring Semester

2022-23

Contents

1 Introduction

This study focuses on creating and implementing a Case-Based Reasoning system designed for a specific synthetic task. The primary objective is to use this system for effective planning based on the chosen domain's carefully conceived design and configuration.

The present challenge revolves around a synthetic problem, which implies that ready-made solutions are unavailable. Instead, solutions must be constructed by adapting and reusing past experiences stored in the Case Base.

This study commences with an in-depth analysis of both user and technical requirements for the engine under investigation. Subsequently, the functional architecture of the Case-Based Reasoning (CBR) system is presented. The proposed CBR solution is then thoroughly described, encompassing the case structure, methods employed in each phase, testing and evaluation procedures, and a comprehensive discussion of the results obtained. In conclusion, this study culminates with a concise summary of the key insights derived from the entire process.

The implemented CBR engine is based on the 4R's Scheme: *Retrieve* (obtain similar cases to a new case), *Reuse* (suggest a solution), *Revise* (evaluate if it's a success or a failure) and *Retain* (add the new case to the case base). These 4R's can be also be seen as these 4 steps to follow: *Retrieval*, *Adaptation*, *Evaluating* and *Learning*.

The Case Base implemented uses *Flat memory*, which allowed us to easily manage cases and memory. Although this structure might be slow for retrieval, it should always be able to find the best case.

In this project, our developed system aims to facilitate travel planning procedures based on a carefully chosen dataset. The selected dataset for our Case Base comprises nine distinct features, three of which are numerical and the remaining six are categorical. The numeric features include price, number of persons, and duration, while the categorical ones consist of holiday type, region, transportation, season, accommodation, and hotel. Notably, we have considered 1024 instances to accurately represent a diverse range of user preferences within our analysis.

2 Domain Description

The CBR system developed in this study focuses on the travel domain. As is well known travelling is one of the most common ideas explored when people think about summer or winter vacations or other holidays [1]

In 2017, data presented by Google from the searches performed with their engine showed a significant increase in people's interest in traveling[2]. Right now, after the COVID-19 pandemic,

the interest on tourism has increased [3] [4], but also the uncertainty about an interesting place with multiple attraction, a place where the hospitality industry has good offer or, even, the traveling conditions.

This CBR system is intended to help amateur travellers, experts, and travel agencies find the best possible experience based on criteria given by the future traveller related to budget, hotel, region and other factors to ensure a successful vacation or weekend trip.

3 Requirement Analysis

Chapter 3 provides a detailed overview of the user requirements and technical requirements of the system. The user requirements encompass the essential functionalities and features that the system should fulfill according to the user's needs. On the other hand, the technical requirements delve into both the functional and non-functional aspects, outlining the specific criteria and constraints that the system must adhere to. This chapter aims to establish a comprehensive understanding of the expectations and specifications driving the development of the system.

3.1 User requirements

The user requirements of the system revolve around providing a range of essential functionalities. The primary objective of the developed model is to suggest a suitable trip to the user based on their specified preferences. For a suggestion to be considered, not all the field requirements must be met, but a potential trip based on the midway of the requirements should be generated. At this stage, the user should have the option to either accept or reject the suggestion and eventually make a new modified request. If the user rejects the suggestion, then it should readjust the features in order to obtain a different one and the case is not added to the system. Conversely, if the suggestion is accepted, the new case should be added to the system.

Additionally, as a secondary goal, the system should allow the user to contribute by adding new cases. This can be achieved by enabling the user to select their preferred values for the different features, thereby updating the dataset. This feature empowers the user to actively participate in expanding the system's knowledge base. However, in this case, it is crucial that all field requirements are met for the input to be considered valid. Therefore, each individual field must satisfy the necessary conditions or criteria in order to be deemed acceptable.

3.2 Technical requirements

3.2.1 Functional requirements

From a technical standpoint, we begin by discussing the functional technical requirements, which encompass the essential technical characteristics that the system must possess to ensure

its proper functionality. These requirements serve as the foundation for the system's effective operation and performance.

Table 1: Functional Requirements.

Req. ID	Requirement	Description
01	System	The CBR system must create a trip recommendation base on a set of parameters given by the user.
02	System	The CBR system must complement the parameters not given by the user in order to generate a trip recommendation.
03	System	The CBR system must use the feedback provide by the user after a suggestion is made, as well as evaluate diverisity, to add the suggested trip into the case base.
04	System	The CBR system should take each case data from an expert and add this new case, validated externaly and internatly, to the case base.
05	System	The CBR system must be able to read and write the case base. In this case base all the cases must be sucessfuly stored.
06	System	The CBR system must allow users to check the quality of the possible results any time is needed.
07	System	The CBR system must ensure all data store or collected is anonymized.
08	System	The CBR system must delete irrelevant cases, to keep the CB updated.

3.2.2 Non Functional requirements

In addition to the explained functional requirements, a set of non-functional requirements has been implemented to facilitate intuitive user interaction with the recommendation system. These non-functional requirements focus on enhancing aspects such as user experience, usability, and system performance, ultimately ensuring a seamless and user-friendly experience.

Table 2: Non Functional Requirements.

NF Req. ID	Requirement	Description
01	Scalability	The CBR system must stablish the bases for a web app which allow more than one user at a time.
02	Usability	The system should allow 9 out of 10 users to use the basic "Find me a trip" functions with out any instruction.
03	Performance	El CBR system must delivery accurate answers in minimum time.
04	Security	The CBR system should properly save all the data provided by the general user, expert user, and the project team.
05	GDPR	The CBR system must ensure the right usage of the user data and the implementation of every law or directive establish in any influence area or regions where the CBR system is deployed.

3.2.3 System requirements

In the same way as the system fulfill the requirements previously presented, the CBR need a proper environment to work. Next table present what need to be ready on the Computational device to run the Travel recommendation system.

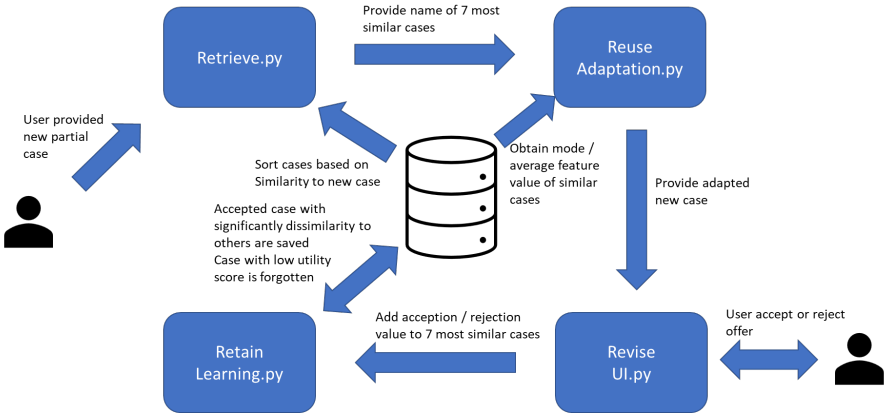

Figure 1: Model's diagram.

Table 3: System Requirements.

Sys. Req. ID	Requirement	Description
01	Operative System	The system will be able to run on Linux, Wndows and MacOS operative system. From the beginning of the project the support for mobile devices were not intended.
02	Interpreter	The CBR system will run under Python version 3.x or more recent.
03	Framework	The CBR system will use next packages and libraries: • customtkinter • ipython • numpy • pandas • scikit_learn • xlrd

4 Functional Architecture

The system consists of 4 main parts. The Retrieval, reuse, revise and retain as in the Case based cycle as shown in figure 1. The retrieval, Adaptation, and Learning, are handled by individual files, with required functions. The UI file is calling the adequate functions based on the user input, and the state in the CBR cycle. The input to the system is the user, which adds a request providing some information about the desired trip. Then the Retrieval gathers all the cases from the case base, deciding the most similar ones to the new provided case. The 7 most similar cases are given in form of indexes to the reuse function. This is calling the 7 most similar cases from the case base, combining them to one new case. This is given back to the user from the UI. The user can then give feedback, in the form of accepting or rejecting the new case. This information is sent to the Learning function, to decide whether the new case should be saved or not to the case base. If it is saved, the learning function will call all the cases from the case base and decide if one of them should be deleted. Then the cycle continues, as a new

user can request a trip.

5 Proposed CBR engine

5.1 Case Structure and Case Library

The CBR system developed in this study focuses on the travel domain and utilizes a dataset comprising 9 features: holiday-type, price, num-persons, region, transportation, duration, season, accommodation, and hotel. Among these, three features—price, num-persons, and duration—are numerical, while the remaining six are categorical.

Regarding the numerical characteristics, they exhibit the following properties.

Table 4: Numerical Features.

Feature	**Range**
price	[200, 10000]
num-persons	[1, 15]
duration	[1, 56]

On the contrary, the categorical characteristics possess the following attributes.

Table 5: Categorical Features.

Feature	**Num. of Possible Values**
holiday-type	8
region	44
transportation	4
season	12
accommodation	5
hotel	273

A total of 1024 instances have been considered to ensure a comprehensive representation of diverse user preferences in our analysis.

Lastly, each case has a field telling, how many times the case is used to make an accepted offer, and how many times it is used resulting in a rejected offer. those fields are not shown to the user but are utilized by the learning function in the system. The indexing of the case is obtained from the ordering in the CSV file, and is used internally as a reference during program execution, but is not saved.

At the application level, we have implemented the case base using a Pandas DataFrame structure from the Pandas Library. This choice offers us a multitude of benefits, including increased flexibility and efficient system responsiveness. The decision to adopt a flat structure and utilize

a Pandas DataFrame object was driven by the dataset's low complexity and the limited range of possible feature values. By leveraging the DataFrame's capabilities, we can easily manipulate the data, perform complex operations, and derive meaningful insights.

An example of a case showing the overall structure is shown in table 6.

Table 6: Case Base.

Holiday	City
Price	4300.0
Persons	4
Region	Alps
Transp.	Car
Days	4
Month	April
Accommodation	TwoStars
Hotel	Hotel White House, Egypt
Acceptances	56
Rejections	30

The case shown in Table 6 is used for both adding a new case and Finding a trip for a user.

5.2 Retrieval

To retrieve the most similar cases, the pairwise distance is computed between the new case and the cases of the case base. The distance function is reinformed for each type of feature. The distance between the quantitative features is measured with the Canberra distance as shown in Equation 1

$$\frac{|p_i - q_i|}{|p_i| + |q_i|} \tag{1}$$

The reason for using Canberra is that it's a relative distance measure. In the domain, the relative difference of the number of persons, price, or duration is more important than the absolute difference calculated in for instance L2-norm. Some of the qualitative attributes are transformed during computation, based on domain knowledge. In the selection of the number of stars for the accommodation, normalized label encoding is used in the distance measure, with L2-norm. For the comparison between months, this is computed in a cyclical manner, so the absolute distance between the adjacent months is normalized and calculated from L2-norm. The rest of the qualitative attributes are considered with distance 0 if they are equal, and 1 if they are not. the total distance from the vector of distances is summarized and returned. To know how each feature is calculated, the feature type is specified in a separate JSON file in the Data folder, so this can be modified by the user.

5.3 Adaptation

As part of the reuse process, after the retrieval of the most similar cases is done, the creation of the future suggested trip is performed through an adaptation process. In this project, the method used is a mix between Weighted Adaptation and Rule base adaptation.

In an early stage of the development a pure weighted adaptation algorithm was implemented, offering reliable results for most of the cases. The suggestion was made complementing the missing parameters (the ones not given by the user) with the average in case of continuous features (KNN regressor) and with the most common one if the feature was categorical (KNN classifier). The default value for k, is set to 7, as it is balancing stability and similarity.

In this point, some inconsistencies were found. After realizing a few suggestions were not matching values like Region with the proposed Hotel, for example, a rule was introduced. Having in mind the region has influence over the type of trip (Holiday) and the Hotel selection the rule or, even, the price; the rule created takes as the first option the region selected by the user to filter the results but, in case there are no results matching this criterion, the most common region will be selected.

The analysis of the results and the team discussion about other options show this mix implementation as the best alternative to use.

5.4 Evaluation

The evaluation is based on the user input. When the new adapted case is shown to the user, the user will evaluate the case. If the person likes it, the case is accepted, if not the case is rejected. An accepted case will get an increase in its acceptance count, while a rejected case, will receive an increase in the rejection count.

5.5 Learning

The learning function is called when a new case is accepted. This calls the retrieval function to get a distance matrix between the adapted case and the other cases in the case base. If the new difference to the most similar case is higher than a given threshold, 0.3 by default, the new case is considered as providing some diversity and is added to the case base. When a new case is added to the case base, to keep the case base updated, and restrict the size in the long run, a utility score is calculated for each case following equation2:

$$UM(C) = \frac{(\#UaS/\#S) - (\#UaF/\#F) + 1}{2} \tag{2}$$

Where C is a retrieved case, UaS is the number of times that the case was used and there was a Success, when the case was among retrieved cases S is the total amount of Successes when the case was among retrieved cases UaF is the number of times that the case was Used and there

was a Failure, when the case was among retrieved cases F is the total amount of Failures when the case was among retrieved cases

If the case with the lowest utility score in the case basis is below a given threshold, 0.5 by default, it's deleted.

6 Testing and evaluation

In Chapter 5, the testing and evaluation phase of the developed system is carried out. This chapter focuses on analyzing both the quantitative and qualitative results of the system. By examining these results from multiple perspectives, we gain valuable insights into the system's performance and effectiveness.

6.1 Quantitative Results

In order to validate the model developed for the travel planning domain, a series of tests were conducted within the system. To begin, the data was divided into two parts: 80% for training and 20% for testing. Subsequently, a specific number of random features were systematically removed to observe the evolution of distances in both the most similar and weight-adapted cases. To support the analysis, the test set yielded a set of distances for each case, from which the maximum, minimum, average, and most frequent distance values were extracted and recorded in the table below. It is important to note that the dataset consists of 9 features, and thus the number of removed features ranges from 0 (inclusive) to 8 (inclusive).

Table 7: Distance values.

Features Removed	Type of Distance	Max	Min	Average	Most Frequent
0	Weight Adapted	0.33	0.00	0.03	0.00
0	Most Similar	0.75	0.33	0.59	0.58
1	Weight Adapted	0.47	0.00	0.22	0.33
1	Most Similar	0.75	0.33	0.61	0.67
2	Weight Adapted	0.58	0.00	0.37	0.33
2	Most Similar	0.75	0.47	0.63	0.67
3	Weight Adapted	0.67	0.00	0.47	0.47
3	Most Similar	0.75	0.33	0.64	0.67
4	Weight Adapted	0.75	0.33	0.53	0.58
4	Most Similar	0.82	0.47	0.66	0.67
5	Weight Adapted	0.82	0.33	0.61	0.58
5	Most Similar	0.88	0.33	0.68	0.67
6	Weight Adapted	0.82	0.47	0.67	0.75
6	Most Similar	0.94	0.33	0.72	0.75
7	Weight Adapted	0.88	0.47	0.73	0.75
7	Most Similar	0.94	0.47	0.75	0.82
8	Weight Adapted	0.94	0.47	0.79	0.82
8	Most Similar	0.94	0.33	0.79	0.82

Upon analyzing the values presented in the previous table, a consistent pattern emerges regarding the distance values. As the number of removed features increases or the considered features decrease for the new case, the distance between the original case and the suggested one also increases. This implies that with fewer features considered, the model possesses less information to provide accurate suggestions, resulting in a decrease in the quality of the suggestions.

However, it is worth noting that this pattern is less pronounced in the minimum distance column compared to the other columns. This suggests that while the model generally exhibits a decline in the quality of suggestions, it is still capable of producing some specific instances that are closer to the original case.

When comparing the two types of distances, namely the most similar case distance (directly obtained from the train set) and the weight-adapted distance (adjusted based on weights), it becomes evident that the discrepancy between the two approaches becomes more significant as the number of randomly removed features decreases. In this scenario, the weighted distance approach proves to be more suitable. Thus, this validates the latter as a promising method for the task at hand.

In summary, it is apparent that the model is well-suited for the chosen domain, as it consistently produces satisfactory results until three features are removed, inclusively.

Furthermore, to enhance the comprehensive analysis, we have decided to explore the behaviour of the model when each individual feature is removed while maintaining the same data-splitting configuration. Moreover, we have retained the distance metrics mentioned in the previous analysis.

Table 8: Distance values.

Feature Removed	**Type of Distance**	**Max**	**Min**	**Average**	**Most Frequent**
None	Weight Adapted	0.33	0.00	0.03	0.00
None	Most Similar	0.75	0.33	0.59	0.58
holiday-type	Weight Adapted	0.47	0.00	0.21	0.33
holiday-type	Most Similar	0.75	0.33	0.61	0.67
price	Weight Adapted	0.47	0.00	0.33	0.33
price	Most Similar	0.75	0.33	0.59	0.58
num-persons	Weight Adapted	0.47	0.00	0.21	0.33
num-persons	Most Similar	0.75	0.33	0.61	0.67
region	Weight Adapted	0.33	0.00	0.28	0.33
region	Most Similar	0.75	0.33	0.61	0.67
transportation	Weight Adapted	0.47	0.00	0.05	0.00
transportation	Most Similar	0.75	0.33	0.60	0.58
duration	Weight Adapted	0.47	0.00	0.20	0.33
duration	Most Similar	0.75	0.33	0.61	0.67
season	Weight Adapted	0.47	0.00	0.30	0.33
season	Most Similar	0.75	0.33	0.61	0.67
accommodation	Weight Adapted	0.47	0.00	0.22	0.33
accommodation	Most Similar	0.75	0.33	0.61	0.67
hotel	Weight Adapted	0.47	0.00	0.31	0.33
hotel	Most Similar	0.75	0.33	0.60	0.58

The values presented in the previous table offer a more focused analysis of the dataset by examining the influence of each individual feature.

As observed previously, when no features are removed, the model achieves the best results with lower distances, indicating that it performs optimally when provided with the complete set of information. Additionally, consistent with expectations, it is evident that the weight-adapted distance technique outperforms the most similar distance technique regardless of the feature being removed.

When comparing the results obtained by removing different types of features, it becomes apparent that the importance of the features is relatively evenly distributed. However, the 'transportation' feature stands out as the least influential in terms of information. Even when this feature is removed, the average distance remains exceptionally low (0.05), differing only by 0.02 from the best result achieved when no features are removed. On the contrary, it can be concluded that the three most significant features are the price, with an average distance of 0.33,

followed by the hotel with an average distance of 0.31, and the season with an average distance of 0.30.

Lastly, the technical requirements of the system were examined, including the time response and memory size. It is important to note that the presented values for time response and memory size are only applicable when no features are removed. Through computational tests, it has been determined that these factors are independent of both the removal of features and the type of distance metric used. Therefore, providing additional combinations would be redundant.

Table 9: Time and Memory values.

Feature Removed	**Time Response [ms]**	**Memory Size [MB]**
None	10.25	120.48

Taking into account the anticipated maximum response time and memory constraints of the system, the current model's performance falls within the acceptable range. In quantitative terms, the execution time is remarkably low, indicating a system that can deliver prompt recommendations—a highly valuable feature that enhances user experience. Additionally, considering the specific context and requirements of our application, memory usage is deemed reasonable. Consequently, we can conclude that the model demonstrates efficiency in terms of both time and memory utilization.

6.2 Qualitative Results

In order to illustrate the functionality of the system and its alignment with user preferences, a specific example is provided to showcase the qualitative results.

Let's consider the following user preferences:

Table 10: Case Base.

Holiday	**Price**	**Persons**	**Region**	**Transp.**	**Days**	**Month**	**Accommodation**	**Hotel**
City	4300.0	4	Alps	Car	4	April	TwoStars	Hotel White House, Egypt

The system offers the following adapted solution:

Table 11: Case Base.

Holiday	**Price**	**Persons**	**Region**	**Transp.**	**Days**	**Month**	**Accommodation**	**Hotel**
City	4300.0	4	Alps	Car	4	April	TwoStars	Hotel Chiemgauer Hof, Upper Bavaria

Upon comparing the user preferences table with the system's suggestions, it is evident that the

system successfully provides the user with a suggestion that closely aligns with their specifications.

All the specified variables were compatible except for the hotel choice, as it is not feasible to have the Alps region in Egypt. Nevertheless, the system adeptly identifies this discrepancy and offers a solution that is remarkably close by suggesting a slight modification to the hotel choice. This demonstrates the system's ability to accommodate the user's preferences effectively.

7 Code

7.1 Organization

The code developed is organized into 7 main classes:

- ui.py: interface shown to the user (this is the file that must be run when executing the code).
- main.py: contains the code to test the model with distinct inputs.
- preprocessing.py: the preprocessing of the data.
- Retrieve.py: obtain the closest case.
- Adaptation.py: adaptation of the closest case.
- Learning.py: learning process of the model.
- testing.py: model's tests performed.

The description of each function can be accessed through the .py files, as well as the detailed description of each step of the code.

7.2 Modules

The necessary modules for the code development and the respective versions are the following: customtkinter - version 5.1.3, ipython - version 8.14.0, numpy - version 1.24.2, pandas - version 1.5.3, scikit_learn - version 1.2.2, and xlrd - version 2.0.1. The Python version used is 3.8.8 through PyCharm CE software.

7.3 Execution

To execute the code through the terminal the following steps should be taken:

1. pip install customtkinter

2. pip install ipython
3. pip install numpy
4. pip install pandas
5. pip install scikit_learn
6. pip install xlrd
7. python3 /PATH_WHERE_THE_FILE_ui.py_IS_INSERTED Ex: /Users/joaovalerio/Documents/"MAI UPC"/"2 Semester"/SEL/W3/source/ui.py

From the execution of the code, the user's interface is displayed.

8 Conclusions

All the goals set for the system have been successfully achieved, and the following concluding paragraphs will highlight the key insights gained from this study.

First and foremost, it was determined that having a database that comprises pertinent and representative features of the targeted domain is of utmost importance. In the context of the domain under investigation, it was found that an extensive number of instances is not necessarily required to generate accurate and valuable suggestions.

The functionality of the system is shared between Retrieval, Adaptation, evaluation, and Learning functions, which form the Case-based Reasoning (CBR) cycle. The system utilizes a case base with 1024 instances in the travel domain, comprising both numerical and categorical features. The retrieval process involves gathering similar cases from the case base based on user input, while the reuse function combines the most similar cases into a new case for the user. The user can provide feedback on the new case, which is then used by the learning function to determine whether to save or delete the case from the case base. The system employs various distance measures, such as Canberra distance for quantitative features and normalized label encoding for qualitative attributes. Cases in the case base are evaluated and updated based on user acceptance or rejection, and the utility score is calculated to manage the case base's size and diversity.

In terms of the system's achieved results, it can be concluded that it is well-suited in terms of memory and time complexity, offering a favourable level of usability to the user. The implementation of weighted adapted solutions has proven to be successful, as the obtained results closely approximate the original preferences and effectively fulfil the user's requirements.

Furthermore, when considering the importance of the features, it is evident that they hold similar levels of significance, with each feature playing a substantial role. Therefore, removing any

feature, aside from transportation, significantly impacts the system's performance. Moreover, as more features are removed, the model faces increasing difficulty in making suitable decisions.

In summary, the travel planning system has demonstrated its capability to deliver satisfactory results, providing the user with valuable suggestions that align with their preferences.

9 Future Work

Some of the following specifications can be considered for future improvement of the present CBR model:

1. Allow the user to select a priority list of features.
2. Allow the user to insert a value-feature pair as mandatory.
3. Allow the user to reject completely a value-feature pair option.
4. Allow the user to select hyper-parameters regarding the model (such as the threshold to accept or forget cases).
5. Use feature importance measure to perform feature weighting.
6. Search for a real-world dataset domain.
7. Implement audio feature.
8. Provide a Website to display the system.
9. Test the model with real users and consider their feedback.

References

[1] L. Gillespie, "Survey: 80% of summer vacationers are changing their plans due to inflation," 2023.

[2] G. C. insights, "Travel planning statistic," 2017.

[3] A. Graft, "Travel and tourism statistics: The ultimate collection," 2022.

[4] M. Turner, "Stats: 62% of americans planning to spend more on travel in 2023," 2023.