Real Estate Loan Knowledge-Based Recommender System

Abdelkader Adla
Department of Computer Science
University of Oran 1
Oran, Algeria
{Adla.abdelkader.adla@univ-oran1.dz}

Journal of Digital Information Management

ABSTRACT: *In decision making, the decision-makers frequently employ and perform routine tasks. These processes normally are time-intensive, complex, and in most cases occur regularly. To address this challenge decision makers reuse the already successful decisions. During difficult times, such actions may lead to save time, energy and man-hours, and also result in effective decision making. Memory building depends on how we successfully store earlier knowledge. We through this work introduce a recommender system which is names as BLKBRS which utilized the earlier successful models. In this work we use a case of bank loan and experimented using a semi-structured multiple attribute recommendation environment, and equate the RL-KBRS with a conventional case based reasoning system. RL-KBRS will compensate for lack of experience of young bank consultants, which permits the spread of knowledge distribution to other banks.*

Subject Categories and Descriptors
[H.3] Information Storage and Retrieval; [I.2] Artificial Intelligence

General Terms: Memory-based Approach, Information Search, and retrieval, Recommending systems, Case-Based Reasoning

Keywords: Recommender Systems, Case-Based Reasoning, Information Retrieval, Memory-Based Approach, Bank Loan

Received: 18 September 2019, Revised 13 December 2019, Accepted 28 December 2019

Review Metrics: Review Scale- 0/6, Review Score- 4.98, Inter-reviewer consistency: 72.5%

DOI: 10.6025/jdim/2020/18/2/65-77

1. Introduction

Recommending processes involve users performing periodic and repetitive activities. This prompts them to store their knowledge and their experience invoked in prior sessions in a shared repository. This shared repository together with the appropriate means for managing its content is in fact a utilization of memory.

Previous research in recommending systems has proposed the use of memory to accumulate, organize, preserve, link and share diverse knowledge coming from past experiences [1] [2] [3] [4] [5]. The development of a memory is seen as the means by which knowledge from the past can be used to influence present activities. The memory and its associated mechanisms to capture the experiential knowledge of the users can be of significant value to the organization in general, and the users in particular.

Many advantages are devolved to memory. The latter, that stores the knowledge of users and experience, retains the rules and procedures, is useful for users engaged in similar recommending activities. It clearly assists them, reduces the time required to come to a choice, particularly, in a critical situation, simplifies the process, and codifies strategies [6] [7] [8]. In addition, the memory can be used to increase user effectiveness by supporting the

management of information [9]. Other cases where such a memory will be of utility are multi-stage problem solving where tasks cannot be satisfactorily completed in a single session, creating a need to carryover intermediate findings to subsequent sessions.

Memory concepts in recommender systems have been addressed by many earlier papers such as [10], [11] memory to support repetitive recommender processes is not adequately described in the previous papers. Recommender systems that support the reuse of choices and recommendations are partially addressed. Even research has specified such systems, a comprehensive approach is not yet arrived.

We apply our approach to real estate loan management; a financial domain where products usually require a long-term significant financial commitment, compared to those of conventional recommender systems, as their utility is not attained immediately. This depends on several external factors (like market returns, governmental regularizations, currency, etc.). Furthermore, in this domain, expert knowledge is necessary to judge which product is a good choice. As a way to deal with these needs, we propose a recommender system called RL-KBRS for a bank to access and retrieve knowledge in a similar activity. The integration of a memory in the form of a case base within a recommendation system is likely to provide additional information processing support. The memory indexes and retrieves cases to propose whole or part of them as solution(s) to a new problem [12]. In this way, the recommending system increases knowledge reuse functionalities through cases and their components, and allows reducing the time required to come to a decision, compensating for lack of experience of young bank consultants, and disseminating and distributing available experience to different bank sites.

The remaining part of the paper is organized as follows. First, in section 2 we give a literature review and background on recommender systems, the major datamining techniques applied in developing such systems, and in section 3 their application domains. Next, we describe our knowledgebased approach to recommender system and its integration within group decision support systems. In section 5 we apply our approach to a bank loan assessment, a case study to illustrate the feasibility and applicability of our idea. A comparative study is done in section 6. Finally, concluding remarks and future work are given in section 7.

2. Literature Review and Background

2.1 Recommender Systems

Recommender Systems [13] are information filtering and decision supporting tools for interacting with large and complex information spaces. They help respond to the information overload, being able to provide the user with advice about a decision to make or an action to take, when there may be a great many options to consider. Such systems provide a personalized view of such spaces, prioritizing items likely to be of interest to the user in a specific context. These systems learn about user preferences over time and automatically suggest items that fit the learned user model. To be able to provide recommendations, information should be acquired from the users, either explicitly (by obtaining user ratings of the items) or implicitly (by tracking user behavior). Recommender systems may also utilize information contained in the user profile [14].

Recommender systems apply several data mining techniques such as collaborative and content-based filtering, hybrid techniques, knowledge-based methods depending on the characteristics of the domain, the quality of available data and the business goals.

2.1.1 Collaborative Filtering

Collaborative Filtering (CF) has been described as “the process of filtering or evaluating items using the opinions of other people” [5-15]. Traditional CF technique works using the analysis of the historical rating data, finds a user’s neighbors (i.e., users having a rating history similar to that of the current user), predicts the ratings of the item for the target user and generates the recommendations [16]. If a user has rated items in common with other users, it is assumed that they have similar preferences; the other users will then be called the user’s “neighbors”. Users will get recommendations for items they have not previously rated but have been positively rated by their neighbors [17].

Collaborative-Filtering systems focus on the relationship between users and items. One of the main features of the CF technique is calculating the similarity between users or items. The items recommended to a user are those preferred by similar users. Similarity of items is determined by the similarity of the ratings of those items by the users who have rated the same items.

A similar technique called Item-based Collaborative filtering was also proposed [18]. It looks at similarities between items. Instead of identifying similarities between users, this method attempts to find the most similar items to those the candidate user has already rated. The similarity between two items is computed by finding the users who have rated these items and then applying a similarity computation method [18]. Similarity computation methods rely on user ratings on items, and not on item characteristics, as is the case in Content-Based filtering (discussed below). Once the similarity of items is calculated and similar items have been specified, the process uses prediction techniques to recommend to the user the most appropriate items [18].

The biggest advantage of Collaborative Filtering is that it does not depend on any system representations of the items to be recommended and can function well with

complex items such as music and movies [9]. However, the most well-known problem of this method is the "New user" problem, which states that a new user has to rate a certain amount of items before the system may effectively apply the algorithm. This is true if we consider that neighborhood formation, which finds user similarities, is based on previous user ratings. Therefore, a new user with no previous ratings is left with no recommendations. Moreover, the "New item" problem is also very common where the recommender cannot recommend a new item until it has been rated by a number of users. Finally, scalability issues have been reported, as well as performance problems for users with large information set [18].

2.1.2 Content-Based Filtering

Content-Based Filtering (CBF) performs a matching between features of alternatives and the user's interests on the values of those features. Content-based filtering [19] compares the content of already consumed items with new items that can potentially be recommended to the user, i.e., to find items that are similar to those already rated positively by the user. Content-Based systems focus on properties of items. Similarity of items is determined by measuring the similarity in their properties. Items are similar in terms of their content (characteristics, features and attributes of the items are used), and therefore the algorithm differs from Item-based CF.

As Content Based recommenders tend to use items represented by text, the content is normally represented by keywords using simple retrieval models, the most representative example being the Vector Space Model, where cosine similarity with weights computed using either TF or TFIDF are the most popular techniques.

Some of the most important Content-based issues are [20,21]: items suffer from over-specialization, since the algorithm focuses only on items already rated by a user as well as other items similar to those, excluding in this manner other, different types of items. The "New user" problem also applies here, where a new user has to rate a certain amount of items before the system can apply the algorithm. This happens because in order for the system to learn any user preferences, the user must rate a number of items. Moreover, Content-based filtering is limited to use only features that are explicitly associated with the items to be recommended.

2.1.3 Hybrid Filtering

The shortcomings of both collaborative filtering and contentbased techniques can be overcome by combining them by adding content-based capabilities to a collaborative based approach (and vice versa), or by unifying the approaches into one model [22] to achieve a higher performance while limiting the potential drawbacks that each system may have separately. For example, when combining content-based filtering with collaborative recommendation, content-based recommendation helps to recommend unrated items.

The combination can exist either within a system, i.e. the system uses a combination of CF and CBF methods, or by using two separate systems, i.e. a CF recommender and a CBF recommender accordingly [20]. In the first case, an example would be to construct user profiles via CBF techniques and then directly compare these profiles to evaluate the similarity of users to provide collaborate recommendations [20]. In the latter case, one can either combine the output of the two systems in one common recommendation list, or choose which of the two recommendation sets is going to be displayed according to appropriate metrics.

CF and CBF systems are easy to set up since only basic information about item names, descriptions, and graphical representations is needed. Hybrid Filtering (HF) methods can be more precise than conventional ones; however, the efficient implementation of such solutions can be very difficult for complex problems. Both approaches, if not trained with lot of examples (item ratings or pattern of user preferences), deliver poor recommendations. This limitation mostly motivated a fourth approach, knowledge-based, that tries to better use preexisting knowledge specific of the application domain (e.g. travels vs. computers) to build a more accurate model requiring less training instances.

2.1.4 Knowledge-Based Filtering

The Knowledge-Based Filtering (KBF) approach is considered complementary to the other approaches [23]. This KBF recommender provides recommendations based on the domain knowledge of what items features will match a user's interest. Knowledge can be expressed as a detailed user model, a model of the selection process or a description of the items that will be suggested. KBF recommender systems require a more detailed specification of the recommendation knowledge (represented in terms of attributes, constraints, and/or similarity metrics) and also of the corresponding items (semantic properties have to be specified).

The most important advantage of KBF method is that the recommendations relies only on the domain-knowledge and constraints of the user preferences and does not depend on (exclusively) user's rates, hence avoiding the mentioned difficulty in bootstrapping the system. However, the usually complex and error prone process required for extracting the required knowledge and building the needed models (knowledge representation), is seen as a limitation of this approach.

KBF recommender systems are mainly based on Case-Based Reasoning (CBR) to provide recommendations to users. CBR systems are distinguished from other forms of CBF systems by using fairly well-structured descriptions of those items [24]. The main difference with CF and CBF is that in KBF recommender systems, the user interactively specifies a single example of interest, and the nearest neighbors of this example are retrieved as possible items of interest for the user. KBF recommender

systems are easy to be explained. Furthermore, a significant amount of domain knowledge is used in the design of the similarity function, because only a single example is available. This single example can be more appropriately viewed as a user requirement rather than a historical rating, because it is specified interactively. In KBF systems, there is less emphasis on using historical ratings.

2.2 Knowledge-Based Recommender Systems

In knowledge-based recommender systems, the memory which contains a collection of data items is organized in a knowledge base taking originally the form of a database. Over the years, the knowledge becomes more complex and its nature has evolved. Several knowledge base recommender systems tend to apply a case-based approach where the knowledge base is structured as a case base.

2.2.1 Role of Memory in Recommendation

Recommender systems focus on activities or tasks which involve a sequence of operations and users [25]. Recommender systems researchers have employed shared repositories such as knowledge bases to store user information and knowledge from prior sessions. Recommendation literature has frequently used the term "memory" for this shared repository and its tools.

Recommender systems can greatly benefit from memory. Indeed memory has a significant role in intelligent recommending support. Such a memory can support a work within and across recommending sessions. It can also provide uniform and consistent knowledge acquired from prior sessions. Recommender systems based on memory support the capture, storage, and recall of memory [26]. It is assumed that intelligent recommending support should provide some memory aids for the users and include "learning" from the users' experience.

Memory integrated within a recommender system should be designed as means for providing easy access and retrieval of relevant information to users. Case-Based Reasoning (CBR) systems are the most prevalent ones that implement memory. The case-based reasoning formalism was proposed as a way of reflecting human knowledge by storing data about significant experiences as "cases" and manipulating reasoning by analogy.

2.2.2 Case-Based Reasoning

Case-Based Reasoning (CBR) is one of the most successful machine learning methodologies that exploit a knowledge rich representation of the application domain. The main hypothesis behind CBR is simply that similar problems have similar solutions. Basically, CBR is a problem solving methodology that addresses a new problem presented to the system by first searching and retrieving the past, already solved most similar case(s) in the case base, and then reusing an adapted version of the retrieved solution to solve the new problem. CBR systems are designed to keep the experiences or cases of the user in a case-base, which requires maintenance to keep system's response accurate and efficient.

Most case-based reasoning systems represent problems and solutions as cases. In CBR terminology, the central notion is a case which is a contextualized piece of knowledge representing a previous experience. A case usually denotes a problem situation which has been captured and learned in a such way that it can be reused to solve future problems. It is referred to as a past case, previous case, stored case, or retained case. Correspondingly, a new case or unsolved case is the description of a new problem to be solved.

CBR recommender system is very similar to generic CBR problem solving system including the classical four steps of CBR methodology (retrieve, reuse, adapt and retain) [27]. In the simplest recommendation process, the user is supposed to be looking for some items and therefore is asked by the system to provide some item requirements, those that he/she considers as the most important. In reply, the system initiates a search in the case base to retrieves similar problems from the case base that should be suggested and recommended to the user i.e. those that satisfy these requirements and terminates the process by retaining the new case.

Sometime the solution retrieved can be straightforwardly reused in the new problem, but in the majority of the situations the retrieved solution is not directly applicable and must be adapted to the specific requirements of the new problem. The adaptation phase is split into two sub-steps: revise and review. In the revise step the system modifies the solution to fit the specific requirements of the new problem whereas in the review step, the constructed solution is evaluated by applying it to the new problem understanding where it fails and making the necessary corrections. After this adaptation the system creates a new case and could retain it in the case base (learning).

3. Domain Applications of Recommender Systems

Recommender Systems have been used primarily by online e-commerce sites to suggest products to their customers and improve the look to buy ratio. Amazon.com is a very popular example of this type of systems. Nowadays, recommender systems are widely used to provide recommendations for a set of items or products that might prove interesting to a collection of users. They are applied in a very broad scale of domains where a significant amount of choices exists in the system and users are interested in just a small portion of items [28].

Recommender Systems (RS) are receiving significant attention and a number of research projects have focused on recommender systems. A simple similarity based col laborative-filtering method is applied for personalized ranking of properties; however, their data were collected by questionnaires [29]. A study about the application of recommender systems is presented. In [30] the authors

propose a CBR-based application for recommending insurance policies. Case-based reasoning is is employed for portfolio recommendation. They point out that CBR is better than CF for financial domains. The recommendation approach presented in [32] follows a knowledge-based (rule based) paradigm. Several solutions are proposed for recommending various financial products using constraint based reasoning, which is a type of knowledge-based methods [33, 34].

4. A CBR-Based Recommender System

In a Case-Based Reasoning Recommender System the effectiveness of the recommendation is based on: the ability to match user preferences with item description; the tools used to explain the match and to enforce the validity of the suggestion; the range of available functionalities and the graphical interface that support the user in browsing the information content, either the cases or the items to recommend. Using memory, the recommending process should involve the representation and utilization of knowledge organized in cases, and adopts a case-based reasoning approach accessing and searching in this memory.

A case is the most basic element representing a past experienced situation. The main goal of the case representation is to organize the knowledge needed in a relevant way to identify the main characteristics describing a problem and its associated solution and to ensure the retrieval of the most appropriated cases. Each case is in general, a couple of (problem, solution), storing both the problem description and the solution applied in that situation. It is necessary to decide which attributes should compose a case and what representation is better suited to represent the particular knowledge involved in the recommending process. A case may be defined in various ways (like item description, user preference, search criteria and outcome of case). The problem component of the case is the user's query, it is typically represented by a set of item features, those specified by the user, and the solution component of the case is the item itself.

The number of cases in the case base is constantly increasing due continuous memorization of new transactions. In order to reduce the research time and to increase the effectiveness of the retrieval process, the latter is carried out in two steps: The first one consists in organizing the case base in such a way just a subset of relevant case is selected whilst the second one is dedicated to the similarity measurement and the ranking of source cases included in the subset.

4.1 Case Indexing

Several approaches can be applied to organize cases for efficient retrieval. A flat case base is the most simple and common organization. Here, all the cases are represented at the same level by a set of attributes-value pairs i.e. an attribute-value vector. The major drawback of the flat case base remains the exhaustive search through the whole memory. Moreover, this drawback increases sharply when the number of both features and cases increase which requires a huge computational effort.

Case indexing and storage are an important aspect in designing efficient CBR systems in that, it should reflect the conceptual view of what is represented in the case and take into account the indices or features that characterize the case and help in distinguishing one case from another. Case indexing refers to creating additional data structures and partitioning the memory to speed up the search process and retrieval of relevant subset focusing on the most relevant dimensions. Hence, each case in the memory is assigned a specific label which identifies under what conditions the case may be useful, and also suggests the applicability of the case to a new one when retrieved. The case descriptors or attributes can be either of nominal type or numeric one. They are used to measure case similarity.

Instead of an exhaustive search, we use a binary decision tree structure to organize the case-base. The case library is split into groups of cases in such a way that each group contains cases that are similar to each other according to a given similarity measure. In such a hierarchical structuring approach, index cases are structured into categories to reduce the number of available cases for similarity measurement. Using this manageable structure, a balance is found between storing methods that preserve the semantic richness of cases and their indices, and methods that supports efficient search, and simplify the access and retrieval of relevant cases. Indexing trees also enable richer case representations of past situations than a simple database, since a case includes information about the context of the event as well as details.

4.2 Case Retrieval

The retrieval process is driven by a similarity metric that computes the similarity of the problem description, i.e., the current user requirements and the items in the case base. More precisely, it is to calculate the degree of similarity between the source cases and the target case. CBR Recommender system guarantees that it retrieves the cases that are maximally similar to the target problem. This operation is carried out over two stages: 1) Measuring local similarities between the characteristics of a source case and those of the target case using a similarity operator; 2) Measuring global similarity from the local similarities.

The local similarities are used to compute similarities between values of single attributes. They are calculated for each attribute i by comparing the value of the target case x_i with the corresponding source one y_i. However, as problem features are described by different types of values (nominal or numeric), the local similarity is calculated regarding these types.

4.2.1 Local Similarity Measurement for Numerical Features

For numerical values, the local similarity is often calculated with a distance measurement. Various distances have been proposed to measure the variation between two values. The most used ones are the Euclidian and the Manhattan distances which are in fact particular kinds of the Minkowski measurement. Eq. (1):

For two cases X and Y:

$$d(X, Y) = (\Sigma_{i=1}^{L} w_i | x_i - y_i|^p)^{1/p} \quad (1)$$

Where x_i and y_i, represent respectively the *ith* feature of X and Y, and w_i the associated weight to this feature.

As the global distance cannot be calculated due to the different features values types, we get a local distance for each feature. For numerical features, the calculation of the local distance is based on the following equation Eq. (2): (derived from Eq. (1)):

$$d(x_i, y_i) = |x_i - y_i| \quad (2)$$

However, this way of measuring can distort the results when the features have different definition domain sizes. Thus, to normalize the distance calculation, we introduce the Int_i function to explicitly express the definition domain. This function expresses the difference between the maximum and the minimum values for the feature i.

$$d(x_i, y_i) = \frac{|x_i - y_i|}{int_i} \quad (3)$$

Finally, the local similarity for a numerical characteristic can be calculated from the distance given by equation (3). According to equation (4, two closest problems are the most similar.

$$sim(x_i, y_i) = 1 - \frac{|x_i - y_i|}{int_i} \quad (4)$$

4.2.2 Local Similarity Measurement for Nominal Features

For nominal features, the classical local measurement considers two values Eq. (5):

$$sim(x_i, y_i) = \begin{cases} 1 \; if \; x_i = y_i \\ 0 \; if \; x_i \neq y_i \end{cases} \quad (5)$$

So, if two features (x_i, y_i) have the same nominal value, then they are identical (similarity = 1), otherwise they are different (similarity = 0).

4.2.3 Global Similarity

The global similarity criteria allow ranking all the source cases from the most similar to the less one. The former are used to express the different importance between features. The user can assign the weight values, or rank the attributes upon their importance. Attribute having the rank of 1 is the most important, and two attributes can have the same rank. For each attribute, the corresponding weight is calculated by Eq. (6):

$$w_i = 1 - \frac{rank_i - 1}{Max(rank_i)} \quad (6)$$

Where:

- Wi is the attribute weight
- Ri is the importance rank of the attribute in the CBR system
- $Max(Ri)$ is the maximum value of importance between the attributes

With the weight w_i, the user can customize the global similarity weighting one feature more importantly than the others. This choice is crucial to obtain relevant similarity measurement.

The global similarity is calculated from all the local similarities in order to establish similarities and more precisely the degree of similarity between the target problem and source ones by taking weighted sums of local similarities of n attributes. The target problem (X) is compared with a source problem (Y) in the case base by means of the global similarity measurement (Eq. 7):

$$sim(X, Y) = \frac{\Sigma_i^n w_i sim(x_i, y_i)}{\Sigma_i^n w_i} \quad (7)$$

A set of retrieved relevant cases (items) is then recommended to the user. If the user is not satisfied with these suggestions he/she can modify the requirements in the query and a new recommendation cycle is started.

5. Application to Bank Loan Assessment

The model that we have presented is being tested using a specific case in a selected bank for loan evaluation. A loan is ending money from one entity (individual or organization) to another one with specified conditions. Under a loan product, we mean a debt with a promissory note specifying the amount of money borrowed the interest rate and the dates of payment. In this domain, the recommendation problem is finding the right product of the loan company for the borrower, which both satisfies his financial needs and will be likely to be paid back by the borrower.

To make a loan to a customer for housing, the bank must take into account some lineaments allowing its real price assessment so that if the customer would be unable to repay the loan, the bank can sell the real state consisting of the land, its natural resources and the buildings on it with its price. The latter varies from one area of real estate to another. It depends on number of lineaments such as façades of the plot of real estate and whether is built or

not. This makes the banking operation complex and various phenomena may occur. Therefore, the bank must consider all these parameters to compare the amount of the loan requested by the customer and the real price of the real estate and give the right choice: accepting or refusing the loan demand.

The bank receives lot of loan requests from customers. To assist bank financial consultant in loan analysis and endow her/him with tools aiding her/him to make quickly right decisions without comparing and analyzing each time all the characteristics of the real estate and preferences of the customer, we propose to develop and use the case-base recommender system, called RL-KBRS.

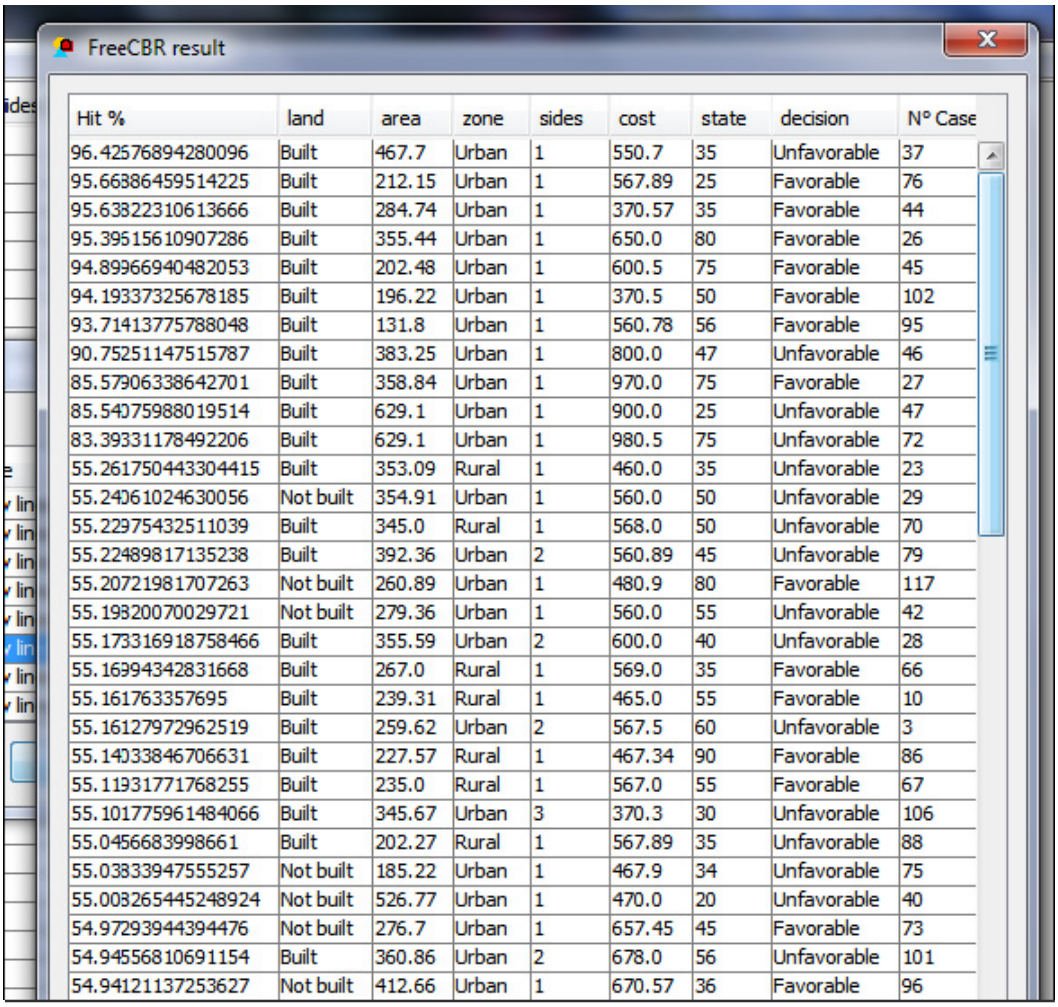
GMSS supports the bank financial consultant to make his analysis based on expert knowledge and past experiences of experts. The bank organizational memory is a case base where each case consists of an already analyzed loan demand with its features and its associated decision.

In order to implement our system, we have collected the data from the "Cadastre "office (land register). The latter manages the land and buildings of the sites built or not, urban or rural through the national territory. The collected data serve to analyze and manage the bank loans demands in order to finance the granting of a real estate by customers. Based on these cadastral data, we performed a simulation for a bank wanting to manage bank loans. We considered a sample of 122 land properties from the Directorate of Cadastre of Oran department, all types combined (rural or urban, built or not built, one or two facades, or even three, etc.).

To evaluate our system and assess the system performance, we have carried out a number of experiments. Our system will be also compared with a conventional CBR called *FreeCBR* (a free open source Java implementation of a Case Based Reasoning tool) authored by Lars Johanson [35].

Loan Requested (Millions of DA)	**Real Estate**				
	Type of the Plot	**Area (m^2)**	**Zone**	**Number of de façades**	**Conditions (%)**
5	Built	350.50	Urban	1	70

Table 1. Example of a bank loan request



Hit %	land	area	zone	sides	cost	state	decision	N° Case
96.42576894280096	Built	467.7	Urban	1	550.7	35	Unfavorable	37
95.66886459514225	Built	212.15	Urban	1	567.89	25	Favorable	76
95.63822310613666	Built	284.74	Urban	1	370.57	35	Favorable	44
95.39515610907286	Built	355.44	Urban	1	650.0	80	Favorable	26
94.89966940482053	Built	202.48	Urban	1	600.5	75	Favorable	45
94.19337325678185	Built	196.22	Urban	1	370.5	50	Favorable	102
93.71413775788048	Built	131.8	Urban	1	560.78	56	Favorable	95
90.75251147515787	Built	383.25	Urban	1	800.0	47	Unfavorable	46
85.57906338642701	Built	358.84	Urban	1	970.0	75	Favorable	27
85.54075988019514	Built	629.1	Urban	1	900.0	25	Unfavorable	47
83.39331178492206	Built	629.1	Urban	1	980.5	75	Unfavorable	72
55.261750443304415	Built	353.09	Rural	1	460.0	35	Unfavorable	23
55.24061024630056	Not built	354.91	Urban	1	560.0	50	Unfavorable	29
55.22975432511039	Built	345.0	Rural	1	568.0	50	Unfavorable	70
55.22489817135238	Built	392.36	Urban	2	560.89	45	Unfavorable	79
55.20721981707263	Not built	260.89	Urban	1	480.9	80	Favorable	117
55.19820070029721	Not built	279.36	Urban	1	560.0	55	Unfavorable	42
55.173316918758466	Built	355.59	Urban	2	600.0	40	Unfavorable	28
55.16994342831668	Built	267.0	Rural	1	569.0	35	Favorable	66
55.161763357695	Built	239.31	Rural	1	465.0	55	Favorable	10
55.16127972962519	Built	259.62	Urban	2	567.5	60	Unfavorable	3
55.14033846706631	Built	227.57	Rural	1	467.34	90	Favorable	86
55.11931771768255	Built	235.0	Rural	1	567.0	55	Favorable	67
55.101775961484066	Built	345.67	Urban	3	370.3	30	Unfavorable	106
55.0456683998661	Built	202.27	Rural	1	567.89	35	Unfavorable	88
55.03833947555257	Not built	185.22	Urban	1	467.9	34	Unfavorable	75
55.008265445248924	Not built	526.77	Urban	1	470.0	20	Unfavorable	40
54.97293944394476	Not built	276.7	Urban	1	657.45	45	Favorable	73
54.94556810691154	Built	360.86	Urban	2	678.0	56	Unfavorable	101
54.94121137253627	Not built	412.66	Urban	1	670.57	36	Favorable	96

Figure 2. Cases retrieved with FreeCBR [38]

6. Comparative Study

The Figures 1 and 2 show the results of relevant cases retrieving with both systems. It should be noted that FreeCBR retrieves all cases, from the case base, with different similarities. On the other hand, our system only retrieve cases with a similarity greater than or equal to 0.5 (> = 0.5) to the new case. We harmonized the case retrieving process for both systems by considering only cases with a similarity of 0.5 (> = 0.5).

Specifically, as shown in Tables 2.a and 2.b, we calculated some values such as *A* the number of relevant cases retrieved, *B* the number of non-relevant cases retrieved, *C* the number of relevant cases non-retrieved by the system, and *D* the number of non-relevant cases nonretrieved:

To evaluate the effectiveness of the search strategies, we use statistical measures such *Precision, Recall, Accuracy and F-Measure*.

There is an inverse relationship between precision and recall. Increasing precision often involves reducing recall. The inverse relationship between precision and recall compels the system to come to a compromise between them. To this end, F-Measure (F-score) is calculated by Eq. 12:

$$Precision = \frac{A}{A+B}$$

$$Recall = \frac{A}{A+C}$$

$$Accuracy = \frac{A+D}{A+B+C+D}$$

$$F-Measure = \frac{2 \times Precision \times Recall}{(Precision + Recall)}$$

Figure 2. Cases retrieved with CBR-DSS [38]

	CBR-DSS	FreeCBR
A	9	14
B	19	51
C	8	3
D	86	54
A+B	28	65
C+D	94	57
A+C	17	17
A+D	95	68
B+D	105	105
A+B+C+D	122	122

Table 2(a). Comparing Case retrieved

		Relevant	Irrelevant	Total
Retrieved	**CBR-DSS**	A : 9	B : 19	A+B : 28
	FreeCBR	A : 14	B : 51	A+B : 65
Notretrieved	**CBR-DSS**	C : 8	D : 86	C+D : 94
	FreeCBR	C : 3	D : 54	C+D : 57
Total	**CBR-DSS**	A+C :17	B+D : 105	A+B+C+D :122
	FreeCBR	A+C :17	B+D : 105	A+B+C+D :122

Table 2(b). Comparing Case retrieved

All these measures are calculated for both systems (Table 3) in order to evaluate their performance.

We ran five queries on both systems. The results of retrieving relevant and irrelevant cases with both systems are summarized in (Table 4). The analysis of these results leads to a comparisons between the two systems.

The FreeCBR system retrieves more relevant cases (the A number) than RL-KBRS. But on the other hand, RL-KBRS retrieves much less irrelevant cases (the number D) than FreeCBR. That said, RL-KBRS retrieves a few less relevant cases compared to FreeCBR, but it eliminates many more unnecessary irrelevant cases.

The two systems are also compared against the four calculated statistical measures (Table 5).

• ***Precision and recall:*** To see the difference between the two systems, the values of these two parameters are shown graphically. Then, as depicted in (Figure 3), we can see that RL-KBRS has more precision than FreeCBR. In other words, RL-KBRS is more accurate in recovering relevant cases than FreeCBR. On the other hand, RL-KBRS has missed more relevant cases in the collection of relevant cases, in general, which is measured by recall, figure (Figure 4).

• ***F-measure:*** The inverse relationship between precision and recall, mentioned by F-measure forces to a compromise: some tasks particularly require good precision whereas others need good recall. Out of the five tests, RL-KBRS has, in three times, an F-measure higher than that of FreeCBR, i.e. RL-KBRS makes a good compromise between precision and recall than FreeCBR. (Figure 5).

• ***Accuracy:*** As depicted in (Figure 6), out of the five tests, RL-KBRS has a higher accuracy than FreeCBR. This finding means that RL-KBRS has the highest proportion of correctly classified cases either relevant or not.

	Precision	Recall	Accuracy	F-mesure
RL-KBRS	0,321	0,529	0,779	0,4
FreeCBR	0,215	0,824	0,557	0,341

Table 3. GMSS and FreeCBR comparison based on statistical values

Loan Requests	RL-KBRS			FreeCBR		
	A+B	A	D	A+B	A	D
Req. 1	28	9	86	65	14	51
Req. 2	7	3	101	63	17	59
Req. 3	7	3	108	55	10	67
Req. 4	24	7	93	59	10	61
Req. 5	7	5	109	19	10	102

Table 4. Performance Thresholds

		RL-KBRS	FreeCBR
Precision	Req. 1	0.321	0.215
	Req. 2	0.429	0.270
	Req. 3	0.429	0.182
	Req. 4	0.292	0.169
	Req. 5	0.714	0.526
Recall	Req. 1	0.529	0.824
	Req. 2	0.176	1
	Req. 3	0.300	1
	Req. 4	0.583	0.833
	Req. 5	0.455	0.909
Accuracy	Req. 1	0.779	0.557
	Req. 2	0.852	0.623
	Req. 3	0.910	0.631
	Req. 4	0.820	0.582
	Req. 5	0.934	0.918
F-measure	Req. 1	0.400	0.341
	Req. 2	0.250	0.425
	Req. 3	0.353	0.308
	Req. 4	0.389	0.282
	Req. 5	0.556	0.667

Table 5. Performances Comparison based on statistical measures

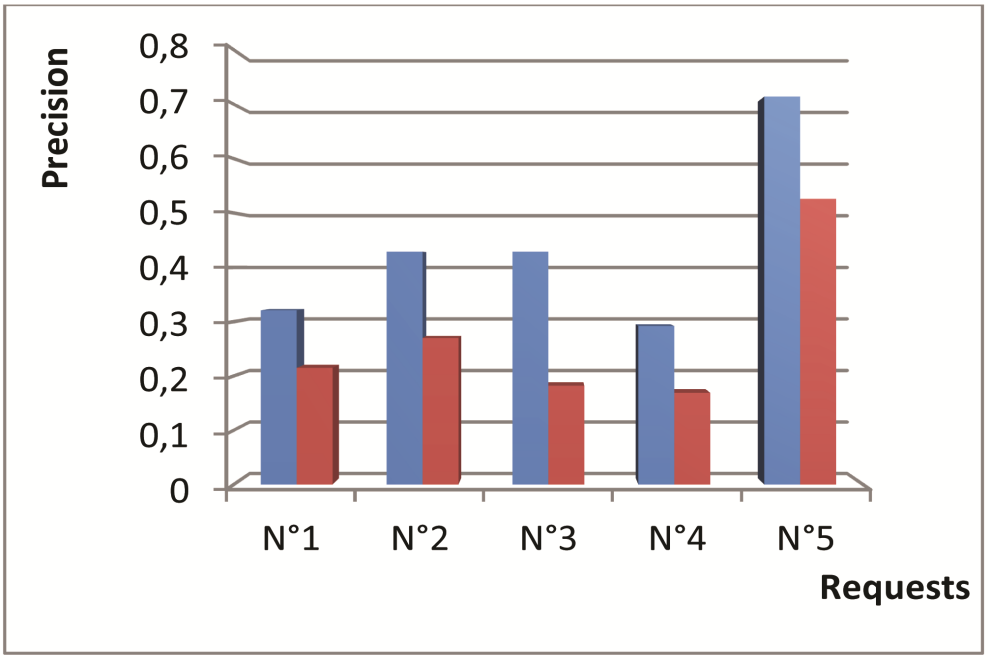


Figure 3. Precision

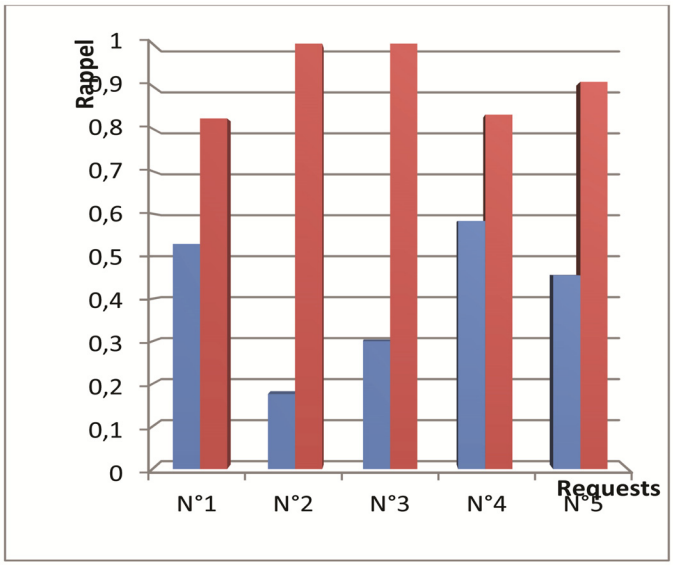


Figure 4. Recall

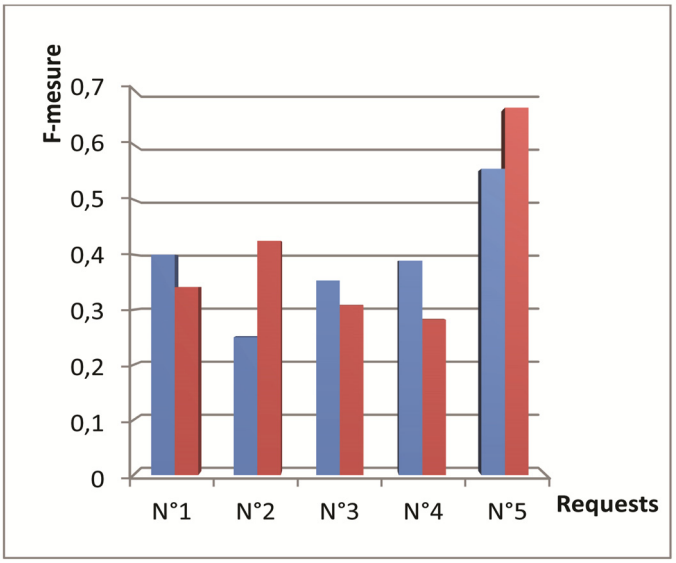


Figure 5. F-Measure

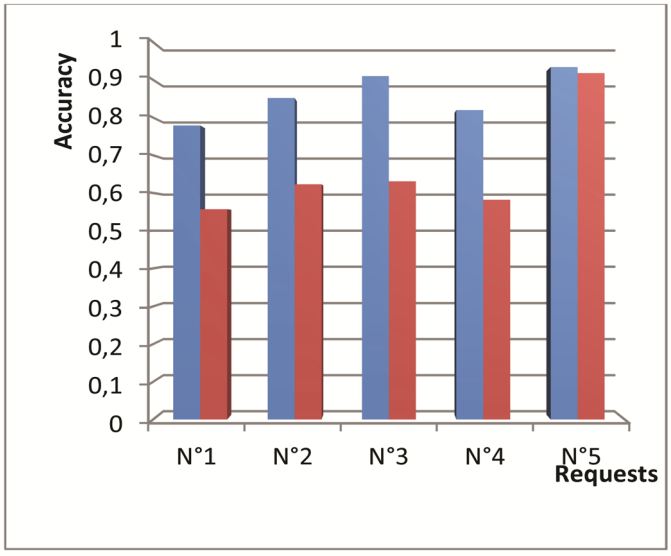


Figure 6. Accuracy [38]

In current bank activity, we must come to an appropriate decision in a short time. It's a precision-critical task where retrieving just one relevant and more similar case to the target problem is sufficient. That means a full recall of all relevant cases is not required. In other words, we need a quick search of the relevant cases to make promptly the decision. So, we need a precision in the relevant cases and a higher recall, i.e. a higher F-measure. Regarding all the previous results, the proposed RL-KBRS system outperforms FreeCBR, the conventional CBR. Nevertheless, more experiments and comparisons with other systems are needed to establish the performance of our system.

7. Conclusion

Knowledge-based Systems have been recognized as means for supporting the collection, storage and distribution of the knowledge and experience of the organization. These systems are proposed to improve the quality of recommending processes by using the power of recent computer technologies.

We described in this paper a case-based reasoning approach. The use of this reasoning mode enables decision makers to reduce the time required to come to a decision, particularly, in a critical situation. We applied our approach on bank loan analysis to provide a bank consultant with access to past experience in the form of cases using a reasoning mechanism to retrieve the relevant cases.

It appears evidence that this form of system is useful for intelligent decision support. Overall the experiment showed in this study the recommender System using CBR to search and recall previous decisions which are in some way "similar" to the current decision situation improve the efficiency of performance of the bank consultant.

Finally, in order to evaluate the proposed system, a comparative performance was undertaken with FreeCBR, a conventional CBR, using statistical parameters. We found that the RL-KBRS system shows better values in relation to some parameters and outperforms the conventional FreeCBR particularly in in relation to precision and accuracy measures. Besides, a new index method is used in order to facilitate the retrieval of relevant cases using the decision tree index and thus reducing the search time.

In future work, we plan to improve RL-KBRS in the case retrieving step to allow calculating and retrieving the similar cases and choosing the most appropriate and relevant source case(s) by coupling the traditional similarity measure with the fuzzy set theory. Also we intend to integrate RLKBRS in the GDSS framework [36,37].

References

[1] Gallupe, B. (2001). Knowledge management systems: surveying the landscape, *International Journal of agement Reviews*, 3 (1) 61-77.

[2] Minhyung, K., Byoungsoo, K. (2017). Motivation, opportunity, and ability in knowledge transfer: a social network approach, *Knowledge Management Research & Practice*, 15 (2) 214-224.

[3] Connelly, C. E., Ford, D. P., Turel, O., Gallupe, B., Zweig, D. (2014). I'm busy (and competitive)!, Antecedents of knowledge sharing under pressure, *Knowledge Management Research & Practice*, 12 (1) 74-85.

[4] Ramos, I., Lavina, L. (2014). Organizational Memory: A Preliminary Model Based on Insights from Neuroscience, *In*: C. Machado and J. P. Davim (Eds), *Transfer and Management of Knowledge*, p. 167-205.

[5] Akoumianakis, D. (2009). Practice oriented toolkits for virtual communities of practice, *Journal of Enterprise Information Management*, 22 (3), p. 317.

[6] Ackerman, M. S., Halverson, C. A. (2000). Reexamining organizational memory, Communications of the ACM, 43 (1) 58-64.

[7] Jackson, P. (2012). Transactive directories of organizational memory: Towards a working data model, *Journal: Information & Management*, 49 (2), p. 118.

[8] Weiser, M., Morrison, J. (1998). Project memory: Information management for project teams, *Journal of Management Information Systems*, 14 (4) 149-166.

[9] Stein, E. W., Zwass, V. (1995). Actualizing organizational memory with information systems, *Information Systems Research*, 6 (2) 85-117.

[10] Hasemana, W. D., Nazaretha, L. D., Paulb, S. (2005). Implementation of a group decision support system utilizing collective memory, *Information & Management*, vol. 42, p. 591-605.

[11] Burstein, F., Jennex, M. E., Olfman, L. (2003). Organizational Memory, Handbook on Knowledge Management, Edited by C. W. Holsapple, Springer, Berlin - Heidelberg, p. 207-234.

[12] Stein, E. W. (1995). Organizational memory: review of concepts and recommendations for management, *International Journal of Information Management*, 15 (1) 17–32.

[13] Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to recommender systems handbook, Springer.

[14] Ortega, B. F., Hernando, A., Gutiérrez, A. (2013). Recommender systems survey, *Knowledge.-Based Systems*. vol. 46, p. 109–132.

[15] Balabanovic, M., Shoham, Y. (1997). Content-based, collaborative recommendation. Communications of the ACM, 40 (3) 66–72.

[16] Aktas, M. S., Pierce, M., Fox, G. C., Leake, D. (2004). A web based conversational case-based recommender system for ontology aided metadata discovery, *In*: Proceedings of the 5th IEEE/ACM international workshop on grid computing, IEEE Computer Society, p. 69–75.

[17] Asanov, D. (2011). Algorithms and Methods in Recommender Systems, Berlin Institute of Technology, Berlin, Germany.

[18] Sarwar, B., Karypis, G., Konstan, J., Riedl, J. (2001). Itembased collaborative filtering recommendation algorithms, *In*: Proceedings of the 10th international conference on World Wide Web, p. 285–295.

[19] Pazzani, M., Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites, *Machine Learning*, vol. 27, p. 313–331.

[20] Adomavicius, G., Sankaranarayanan, R., Sen, S., Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach, ACM Trans. Inf. Syst. (TOIS), vol. 3, p.103–14.

[21] Burke, R. (2002). Hybrid recommender systems: survey and experiments, User Model User Adapt Interaction, 12 (4) 331–370.

[22] Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G. (2014). Mobile recommender systems in tourism, *Journal of Network and Computer Applications*, vol. 39, p. 319–333.

[23] Burke, R., Felfernig, A., Göker, M. H. (2011). Recommender Systems: An Overview Ai Magazine, vol. 32, p.13-18.

[24] Manouselis, N., Drachsler, H., Verbert, K., Duval, E. (2012). Recommender Systems for Learning. Springer, Berlin.

[25] Hibbard, J. (1997). Knowledge management—knowing what we know, Information Week.

[26] Ackerman, M. S., Halverson, C. (2004). Organizational Memory as Objects, Processes, and Trajectories: An Examination of Organizational Memory in Use, Computer Supported Cooperative Work (CSCW), 13 (2), p. 155.

[27] Wess, S., Althoff, K. D., Derwand, G. (1993). Using kd-trees to improve the retrieval step in case based reasoning, In Wess S, Althoff K-D, Richter MM (eds.) Topics in case-based reasoning, EWCBR-93: First European Workshop, Kaiserslautern, Germany, p. 1–5, 1993.

[28] Lu, J., Wu, D., Mao, M., Wang, W., Zhang, G. (2015). Recommender system application developments: A survey, *Decision Support Systems*, vol. 74, p. 12–32.

[29] Wang, L., Hu, X., Wei, J., Cui, X. (2013). A Collaborative Filtering Based Personalized TOP-K Recommender System for Housing, *In*: Proceedings of the 2012 International Conference of Modern Computer Science and Applications SE - 74, ed., Z. Du, vol. 191 of Advances in Intelligent Systems and Computing, p. 461–466, Springer Berlin Heidelberg.

[30] Rahman, S. A., Norman, A. A., Soon, K. J. (2006). MyINS: A CBR e-Commerce Application for Insurance Policies, *Electronic Commerce Research*, 5 (1) 373– 380.

[31] Musto, C., Semeraro, G., Lops, P., Gemmis, M., Lekkas, G. (2015). Personalized finance advisory through casebased recommender systems and diversification strategies, *Decision Support Systems*, vol.77, p. 100– 111.

[32] Magerkurth, C., Sperner, K., Meyer, S., Strohbach, M. (2011). Towards context-aware retail environments: An infrastructure perspective, *In*: MobileHCI, Stockholm, Sweden, p. 1–4.

[33] Felfernig, A., Boratto, L., Stettinger, M. (2018). Group recommender systems: an introduction, Springer.

[34] Felfernig, A., Polat-Erdeniz, S., Uran, C., Reiterer, S. M. Atas. (2019). An overview of recommender systems in the internet of things, *Journal of Intelligent Information Systems*, vol. 52, p. 285–309.

[35] Johanson, L. http://lotsen.ivf.se/CBR/Shaping/index.jsp, accessed 2018

[36] Adla, A., Soubie, J. L., Zaraté, P. (2007). A cooperative Intelligent Decision Support System for Boilers Combustion Management based on a Distributed Architecture, *Journal of Decision Systems* (*JDS*), 16 (2) 241-263.

[37] Nachet, B., Adla, A. (2014). An agent-based distributed collaborative decision support system, *Intelligent Decision Technologies* (*IJDSST*), 8 (1) 15-34.

[38] Adla, Abdelkader., Benmessaoud, Noureddine. (2019). A CBR Decision Support System for Loan Evaluation, *Journal of Digital Information Management*, 17 (2) 75.