Núñez-Valdez, E.R., Quintana, D., González Crespo, R., Isasi, P., Herrera-Viedma, E. (2018). A recommender system based on implicit feedback for selective dissemination of ebooks. *Information Sciences*, 467, pp. 87-98.

DOI: https://doi.org/10.1016/j.ins.2018.07.068

A recommender system based on implicit feedback for selective dissemination of eBooks

Edward Rolando Núñez-Valdez[a,*], David Quintana[b], Ruben González Crespo[c], Pedro Isasi[b], Enrique Herrera-Viedma[d,e,*]

[a] *University of Oviedo, Department of Computer Science, Sciences Building, Oviedo, Asturias, Spain*
[b] *Universidad Carlos III de Madrid, Department of Computer Science, Avda. Universidad 30, Leganés, Madrid, Spain*
[c] *Universidad Internacional de La Rioja(UNIR), Engineering School, Gran Vía Rey Juan Carlos I núm. 41, 28002 Logroño (LA RIOJA), Spain*
[d] *University of Granada, Department of Computer Science and Artificial Intelligence, Granada, Spain*
[e] *King Abdulaziz University, Department of Electrical and Computer Engineering, Faculty of Engineering, Jeddah, Saudi Arabia*

Abstract

In this study, we describe a recommendation system for electronic books. The approach is based on implicit feedback derived from user's interaction with electronic content. User's behavior is tracked through several indicators that are subsequently used to feed the recommendation engine. This component then provides an explicit rating for the material interacted with. The role of this engine could be modeled as a regression task where content is rated according to the mentioned indicators. In this context, we benchmark twelve popular machine learning algorithms to perform this final function and evaluate the quality of the output provided by the system.

Keywords: Recommender systems, explicitation system, implicit feedback, classification algorithms

*Corresponding author

Email addresses: nunezedward@uniovi.es (Edward Rolando Núñez-Valdez), david.quintana@uc3m.es (David Quintana), ruben.gonzalez@unir.net (Ruben González Crespo), isasi@ia.uc3m.es (Pedro Isasi), viedma@decsai.ugr.es (Enrique Herrera-Viedma)

1. Introduction

Nowadays, the problem of information overload on the Internet remains unresolved. The amount of data that is available on the Internet continues to grow exponentially, and this situation makes it more difficult for users to discover or find easily and quickly relevant and interesting items [50].

Recommender systems are intelligent systems that, through the use of information retrieval and classification techniques, try to solve the problem of information overload on the Internet. Using different mechanisms, these systems can filter a lot of information available on the Internet and facilitates users to discover more valuable and interesting information for them [33, 44].

These systems are broadly studied and represent a mature research field. The main social networks existing today, such as Facebook, LinkedIn, Twitter, YouTube or other types of e-commerce websites like Amazon, HBO or Netflix, use recommender system technologies on their websites and are continuously improving them through the personalization of search results [46].

Even though the most common solutions rely on explicit ratings [9], there is an alternative approach to be explored, which is based on implicit ratings derived from user behaviour. In this case, the users' interactions with the electronic content would result in the automatic generation of a rating that could be subsequently used by the rest of users.

In a previously published paper [30], a set of indicators was defined to capture user interaction with e-books. The main objective of this cited research was to carry out a comparative analysis of these indicators and try to find the correlations between different feedback techniques on recommender systems. After obtaining a preliminary approximation of the correlation between these feedback mechanisms, we proposed in [32] an initial architecture for the construction of a content recommendation platform based on users' behaviour. In this case, we focused on the definition of a mathematical model that allowed us to develop an algorithm to transform implicit into explicit feedback in an e-book platform. This previous research suggests that, at least in this context, a recommender system based on implicit feedback might be feasible.

In this paper, we describe an example of such architecture and focus our attention on how the explicitation system gets implicit data supporting the recommendation system. These components of the system translate the mentioned indicators into ratings and use them for making content recommendations. If we consider that, at the core, this engine solves a regression problem, the range of potentially relevant algorithms is quite wide. For this reason, we benchmark several alternatives based on a sample of real data. The contribution of this paper is related to the structure of the system and the benchmarking of popular algorithms that represent a range of broad families (tree-based, function-based, rule-based) to identify the best and the worst alternatives for the recommendation engine in this kind of scheme.

The rest of this paper is structured as follows: Section 2 presents the background of recommender systems; Section 3 describes the suggested approach, including the description of the architecture, the indicators, etc.; Section 4 presents the experimental analysis; and finally, Section 5 includes the main conclusions and future work to be carried out.

2. Background

Several techniques and tools are currently used to analyse, classify or filter the large amount of information available on the Internet with the aim of analysing users' behaviour or tastes. Among these tools are machine learning, Big Data, Natural Language Processing (NLP) or recommender systems. In many cases, these techniques allow us to analyse the users' behaviour with the objective of predicting their future behaviour or discovering their tastes. For example, Baldominos et al. [4] try to predict gamers' behaviour in commercial video games using the Variable-Order Markov Model (VOM) and Big Data. Another example is proposed by San-Miguel [42], in which he uses regression techniques and Big Data in a predictive model to uncover important information related to adverse reaction to drugs in elderly patients. A current technique is text analytics, which is a subcategory of NLP. This allows to measure users' negative or positive perceptions about a product, brand or company [28].

Recommender systems help users to discover quickly and easily the information that they need in a specific context through information filtering. These systems are very important because they help minimize the time users spend searching for content that, in many cases, is not easily found on the Internet. With the implementation of recommender systems, users can find different types of information such as movies, series, books, songs, websites, electronic products, games, toys and any kind of information that may interest them [16].

According to Wang [47], a recommender system is defined as *"A system that has as its main task choosing certain objects that meet the requirements of users, where each of these objects are stored in a computer system and characterized by a set of attributes."*

Using custom filtering information, recommender systems can predict whether a user is interested in a specific content (prediction problem) or select a set of N contents that may be of interest to some users (top-N recommendation problem) [40]. These systems are excellent tools to improve Internet companies' marketing strategies because, in addition to helping users find products that interest them, it helps these companies to increase their sales and minimize advertising costs. In general, these systems help to minimize users' search time and to increase online businesses' profits.

As shown in [16, 33], recommender systems aim to solve the problem of information overload on the Internet using different mechanisms and algorithms for information filtering. However, when these systems do not have enough information about the contents or users' profiles, it is very difficult to carry out an adequate classification and filtering of the information to enable the system to make good recommendations.

The lack of sufficient information related to the users' profiles leads to the following system's issues:

(1) **Sparsity Problem** which occurs when it is very difficult to identify similar users due to lack of sufficient information [34]. Basically, this problem

appears when the number of ratings needed for prediction is greater than the number of ratings obtained from the users [27]. A very interesting thing about this issue is the claim made by Yu et al.[49]: items suffer from sparsity problems more severely than users, since items are usually observed with fewer features to support a feature-based or content-based algorithm. (2) **Cold Start Problem** which occurs when nobody has rated any item, either explicitly or implicitly, from a set of data [12]; (3) **Popularity Bias Problem** which states that different items cannot be recommended to someone with a unique taste; and (4) **New Item Problem**, which appears when systems do not consider an item because it has not been rated previously by anybody.

Traditionally, according to the algorithm or information filtering paradigm that is used, recommender systems can be classified into several types [1]:

- **Collaborative filtering** calculates the similitude between users and creates a so-called 'close neighbor'. This allows the identification of users with similar preference and recommends other similar user-preferred content.
- **Content-based** aims to recommend similar contents to a user on the basis of previous contents that the user liked in the past. These contents have been previously rated by a user and a 'keywords-based' search is performed to know whether an item is similar to another.
- **Hybrid approach** is a combination between collaborative filtering and content-based approaches. Hybrid systems exploit characteristics from content-based and collaborative systems due to their complementary nature. They seek to overcome the limitations from both systems to obtain better recommendations. Some examples of hybrid approaches are presented in [37] where the authors propose a hybrid fuzzy linguistic recommender system to help the Technology Transfer Office staff in the dissemination of research resources interesting to users, and in [24] where the authors propose a hybrid recommender system combining an associative classification algorithm and clustering technique to recommend touristic places to users.

In addition to the classification of the recommender systems cited above, other authors, as Adomavicius et al. [1], have proposed a variety of recommendation techniques such as: Knowledge-based recommendations, Demographic recommendations and Utility-based recommendations.

Currently, there are a lot of e-commerce websites, social networks, and others types of websites that are using recommender system to offer interesting content to their users such as Amazon.com [23], Facebook, LinkedIn, Twitter, HBO and Netflix [15], among others. In addition to these real cases of recommender systems applications, other scientific proposals have been presented in recent years, such as the recommender system presented by Christidis et al. [8] that suggests related items to the user browsing the offers in an electronic marketplace environment. In addition, Lee et al. [22] propose a mobile web news recommender system. Martinez-Cruz et al. [25] present another interesting study and propose a model to characterize user profiles using ontologies and fuzzy linguistic modeling to generate better recommendations, thus improving users' experiences. Tejeda-Lorente et al. [45] propose a recommender system based on items' quality to help users access relevant research resources. Nilashi et al. [29] propose a recommender system based on multi-criteria collaborative filtering in the tourism domain that uses prediction, dimensionality reduction and clustering methods to enhance its predictive accuracy. Park et al. [35] propose RecTime, a real-time recommender system for online broadcasting. The system simultaneously considers the users' preferences and time factors recommend other shows currently airing on other channels.

On the other hand, social networks are a source of information that can help improve users' experience through the use of recommender systems. In [13] a first approach is presented for the development of a platform that allows the analysis of users' comments on social networks with the objective of making recommendations that improve users' satisfaction with the network.

One of the most common issues when implementing a recommendation system is to choose the best recommendation algorithm to solve a specific problem. For this, one of the most interesting points presented by Cunha et al. in [11] is the experimental study on the metalearning approaches that allow the identification of the most important concepts for automatic selection of recommendation algorithms in different frameworks.

Another interesting research is presented in [21], where the authors provide a general overview on the diversification in recommender systems. This research covers three important areas in this field: the definition and evaluation of diversity; the development of diversification algorithms; and the impact of diversification on the quality of recommendation results.

Finally, Bobadilla et al. [5] propose a reliability quality prediction measure (RPI) and a reliability quality recommendation measure (RRI) with the objective of improving the reliability values associated with the predictions made by the recommender systems, and thus to improve users' experiences and satisfaction.

Through feedback information techniques, the recommender systems need to collect information about users' profiles. This process is the basis for these systems to be able to provide valid and interesting information to users [36]. Commonly, these feedback techniques are categorized in explicit and implicit feedback techniques. When these two feedback techniques are mixed, another paradigm for recommender systems is provided [18].

- **Explicit feedback:** It is the mechanism that allows a user to unequivocally express her interest in an object or set of objects. Typically, users assign a score to these objects through a survey process, such as the 5-star rating system or like/dislike rating system, to indicate their interest in an object [18]. As discussed in [14], recommender systems usually collect users' preferences using some of the rating systems cited above. For example, social networks such as Facebook, Twitter, Instagram, LinkedIn or YouTube use the like/dislike rating system as a mechanism for users to be able to rate contents explicitly. On the other hand, online stores such as Amazon, AliExpress and others use the star ratings system, allowing users to indicate which products are of interest to them. Recently, the streaming service platform Netflix has changed its feedback mechanism from a 5-star rating system to a like/dislike rating system.

- **Implicit feedback:** This process consists of getting the score of the objects or products automatically, through capturing, analysing and processing the information retrieved from users' behaviour in an application. For example, when a user reads news or accesses an online article, the time she takes for reading, comments on the content or whether the user has shared it in social networks, are automatically processed by the system to infer whether the article or news is of interest to her. The use of this feedback technique helps improve the users' experience and satisfaction when searching for content on the web, since it does not require explicit ratings to receive recommendations [30, 32].

Nowadays, there are a lot of study cases and widespread implementation of recommender systems based on explicit technical feedback. However, this can be a problem or limitation for users since they usually do not like to rate content because that represents a cognitive cost to them [12, 9]. In this way, implicit feedback technique is a feasible alternative which improves the information recovery process, because an additional effort is not required from the users of the system [19].

3. A new recommender system for e-books

The success or failure of implementing a recommendation system depends on the feedback mechanism that is used to retrieve users' information. Currently, the main problem facing these systems is that, in many cases, explicit feedback is used as the basis for their operation. But this can be an inconvenience for the users of the system since, generally, they do not like to rate the content. On the other hand, it is also a problem for the recommendation systems because if the users do not rate the contents, it is not possible to recommend interesting content.

To improve the feedback mechanisms and thus the recommendation systems, an implicit approach architecture based on the analysis and transformation of users' behaviour in an e-book platform into explicit feedback, is proposed and developed. This means that the system does not require direct user intervention in the feedback process.

One very interesting thing about this approach is that once the explicit feedback is generated, it is possible to use any recommendation engine based on this type of technique.

The mathematical model and the modules that make up the architecture were defined and developed. This allowed to analysis of users' behaviour in an e-book platform and validation of the model through a series of tests.

3.1. Architecture description

As Figure 1 shows, the recommender system platform based on implicit feedback is defined by a Three-Tier Architecture:

- **Presentation Tier:** this tier is composed of the different client applications through which the user can interact with the platform (e.g., a mobile application or a website).

- **Application Tier:** this tier is composed of the feedback system that is responsible for collecting users' behaviour through different client applications. It is also composed of the explicit system that is responsible for analysing and converting the collected implicit information into explicit values. It also contains the recommendation engine that offers interesting content to the users based on the processed data and their profiles.

- **Data tier:** this tier is composed of the storage systems which save and recover the implicit and explicit information of the platform, and the configuration files that contain meta-information about actions to be stored during the feedback process.

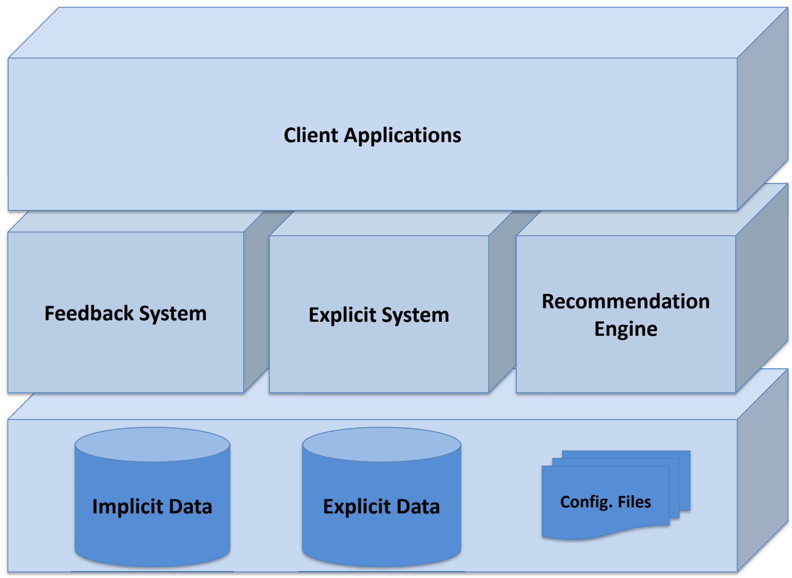

Figure 1: Three-Tier Architecture

3.2. Implicit Recommendation System

To obtain a recommender system based on implicit feedback, we have built the EBook Content Recommendation Platform (ECRP). On this platform, the recommender system offers electronic books that may is of interest to users based on analysis of their behaviour and reading habits.

Two of the most important components of this platform are the explicitation system that allows transformation of users' behaviour (implicit feedback) into ratings (explicit feedback), and the recommendation system that allows to make recommendations to users based on these ratings. We call the union of these systems the **Implicit Recommendation System**.

In order to evaluate the different users' behaviour according to their reading habits and interaction with the platform, a **User Interactions Converter Algorithm (UICA)** is developed [32]. This algorithm evaluates and converts users' behaviour (implicit actions) into explicit ratings. These ratings are generated within a previously established range that indicates users' interest, for

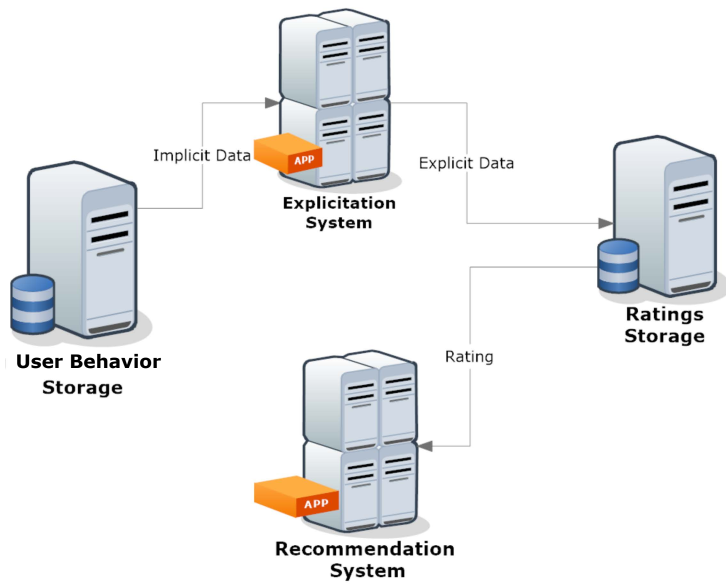

Figure 2: Implicit Recommendation System Architecture

example, a range from 1 to 5.

Table 1 shows a set of commonly performed user actions within an electronic books platform that has been evaluated with the EBook Content Recommendation Platform (ECRP) prototype.

As shown in Figure 2, the Implicit Recommendation System Architecture consists of a set of system applications and a set of data storage components. The implementation of this architecture requires an Explicitation System that extracts the implicit data (interactions between users and contents) from the Users Behaviour Storage and transforms it into ratings (Ratings storage) using the User Interactions Converter Algorithm (UICA). Finally, the recommendation system uses such ratings to make recommendations.

As seen in [31, 32], users rate content using explicit rating systems on the web or in mobile applications, such as the “5-star” or “Like/Dislike” systems to tell the application what content they do or do not like, but, as we said before, users do not usually like to rate content. In this case, the objective of **User**

Table 1: Some common actions that define the behaviour of the users in an electronic book platform

Id	Name	Type.	Indicator	Scope
A_1	Explicit rating of content	Explicit	None	Individual
A_2	Content reading time	Implicit	Positive	Social
A_3	Highlighting content	Implicit	Positive	Social
A_4	Adding a note to content	Implicit	Positive	Social
A_5	Commenting content	Implicit	Positive	Social
A_6	Suggesting content to a contact or friend	Implicit	Positive	Individual
A_7	Adding content to the collection	Implicit	Positive	Individual
A_8	Adding content to the list of favourites	Implicit	Positive	Individual
A_9	Rejecting a suggestion for content	Implicit	Negative	Individual

Interactions Converter Algorithm (UICA) is to evaluate a set of selected actions to convert them into an explicit value (rating).

In [31, 32], Núñez-Valdez et al. present a mathematical conversion model that was the basis for development and implementation of UICA. This model and algorithm allow calculation, transformation, and determination of the value of each action performed independently by a user on content. Finally, from these calculations, the estimated values that indicate a user's interest in a specific content are obtained.

Table 1 shows the actions that are measured and evaluated with the implementation of the **User Interactions Converter Algorithm (UICA)**. As we can see, the actions are defined by different attributes: (1) **identifier:** indicates the action ID, (2) **name:** represents the action name, (3) **type:** indicates if the action is explicit or implicit, (4) **Indicator:** indicates whether the action made by the user is negative or positive and (5) **scope:** indicates if the action is individual or social. The attribute scope is "social" if the value of the action is calculated by considering the way users of other platforms have interacted with the same content. Otherwise, the attribute scope is "individual".

3.2.1. Describing User Interaction Converter Algorithm

For UICA to obtain a rating that indicates a user's interest in a particular content a mathematical model and corresponding algorithm were developed to evaluate and transform each action performed by the user into a numerical value defined within a range of previously established values. This default range is defined with the objective of simulating the explicit rating of a content, that is to say, if the "5-star" system was used the range could be (1 to 5), and if the "Like/Dislike" system was used the range could be (1,2). The lower the value, the worse rating the user would give to the content. A zero (0) value means that the user has not rated the content yet [32].

The final rating of a specific piece of content for a particular user is determined by measuring and weighting each action separately. The weight assigns a level of importance to each action when calculating the user's final rating. The calculation shows that if the content is rated explicitly (A_1), the rating will be equal to the value given by the user. Otherwise, the rating will be equal to the implied actions calculation (A_2 ... A_k).

As shown in [32], the mathematical formula used to calculate the final rating of the user on a specific piece of content based on his behaviour is:

$$V(i,j) = \begin{cases} A_1 \, if \, A_1 > 0 \\ \\ S \, if \, A_1 \leq 0 \end{cases} \tag{1}$$

Where:

$V(i,j)$: is the rating of the $j-th$ content for the $i-th$ user.

i: is the $i-th$ user that performed an action around the $j-th$ content.

j: is the $j-th$ content around which the $i-th$ user performed an action.

A_1: is the explicit rating of the $j-th$ content assigned by the $i-th$ user.

S: is the value obtained by calculating the implicit actions, which is obtained through the following equation:

$$S = \frac{\sum_{k=2}^{n}(P_k + Pr)A_k + A_k}{N+1} \tag{2}$$

Where:

P_k: is the weight assigned to action A_k. Subject to:

- $0 \leq P_k \leq 1$
- $\sum_{k=2}^{n} P_k = 1$

k: is the sub-index that identifies the action. This variable starts at 2 because this calculation only considers implicit actions. $(P_k + Pr)A_k$: is the percentage of weight added to the value of the action. N: is the amount of actions with the $j-th$ content performed by the $i-th$ user. This value is obtained through the equation:

$$N = \sum_{k=2}^{n} f(A_k) \tag{3}$$

Where:

$f(A_k)$: is the function that shows that the $i-th$ user performed the A_k action in the $j-th$ content. The value of this function is determined through:

$$f(A_k) = \begin{cases} 1, \ if \ A_k > 0. \\ 0, \ if \ A_k \leq 0 \end{cases} \tag{4}$$

Where:

Pr: is the remaining weight of the $A_2 ... A_n$ actions NOT performed by the $i-th$ user around $j-th$ content which is redistributed between the P_k weights of the performed actions. The P_r value is calculated as follows:

$$Pr = \frac{\sum_{k=2}^{n} Q(A_k)}{N} \tag{5}$$

Where:

$Q(A_k)$: is the function that returns the value of the A_k action's weight that the

$i-th$ user didn't perform around the $j-th$ content. The value of this function is determined through:

$$Q(A_k) = \begin{cases} P,\ If\ A_k \leq 0. \\ \\ 0,\ If\ A_k > 0 \end{cases} \tag{6}$$

N: is calculated as per (3).

3.2.2. Actions description

This section describes the actions that have been analyzed and evaluated with the proposed algorithm. The mathematical formalization that allows transformation of these actions into explicit ratings are shown in [32]. For this reason, we will focus only on the definition of these actions and how users carry them out when interacting with an e-book.

- A_1 **- Explicit rating of a content:** When a user explicitly rates content, the other actions he performed on it are discarded, because the user is showing his interest in that content explicitly. This indicates that one of the main points is knowing if the user has explicitly rated the content. Thus, when measuring the user's implicit interactions, it must be known if that content has been previously rated and if that rating was explicit or implicit.

  If the content has a previous rating automatically calculated by the system implicitly and the user rates the content again but in an explicit way, then this new value replaces the previous one. This action is known and evaluated as an explicit and individual action. Its indicator is None because the user can rate the content positively or negatively.

- A_2 **- Content reading time:** As can be seen in [30, 32] the longer a user has spent reading a piece of content, the higher the probability that the user is interested in it. Thus, to establish a proper relationship between the time spent on the reading of the content and the real time spent reading the whole content, it is necessary to compare this time with the time that the other users of the platform spent on reading the same content.

To determine the reading value, we need to know how much time the user spent reading each chapter of the book. Measuring the reading by chapters is a better option than measuring by pages, since the amount of these can change depending on the device that is being used. That is because the electronic books automatically adapt their contents to the screen size of the device. This action is known and evaluated as an implicit, positive and social action.

- A_3**- Highlighting content:** When reading content, users usually highlight fragments of the text with different colours, giving them different levels of importance. This action is commonly performed by the user when she wants to highlight words, phrases or even paragraphs from the content that he finds interesting. This action is known and evaluated as an implicit, positive and social action.

- A_4 **- Adding notes to content** While reading content, the user might want to add his own comments and impressions about it through the notes. This action is usually performed by the users when they read a fragment of the text and want to write down their own thoughts about the content.This action is known and evaluated as an implicit, positive and social action.

- A_5 **- Commenting content:** According to the results shown in [30], when a user comments on content, it is because he finds it interesting. Because of this, it is necessary to know if the user has written a comment about the content that is being evaluated.

  To calculate the value of the comments written by a user, we take into account the maximum number of comments written by him, within the total amount of comments written by all the users on each of the contents of the platform. This action is known and evaluated as an implicit, positive and social action.

This action was considered positive because most of the comments made by the users were positive. Nonetheless, we consider that as a future work it is necessary to develop a model of artificial intelligence based on Natural Language Processing (NLP) that classifies the comments as positive, negative or neutral, automatically.

- A_6 **- Suggesting content to other contacts o friends:**

  As Núñez-Valdez et al. [30] claim, when a user recommends content, it is because he finds it interesting. In this platform, it is necessary to know the number of recommendations of the content performed by the user in comparison with the recommendations to other contacts or friends performed by all the users of the platform. This action is known and evaluated as an implicit, positive and individual action.

- A_7 **- Adding content to the collection:** When a user checks content and adds it to his collection, it might be a sign of interest in that content. The value for adding content to the collection is calculated through an equation that gives the value of the superior limit of the normalization if the content was added to a collection and zero(0) value if it was not added. This action is known and evaluated as an implicit, positive and individual action.

- A_8 **- Adding content to the list of favourites:** Normally, when a user adds content to his favourites list, it might be a sign of interest in that content. The value for adding a content to the favourites list is calculated through an equation that gives the value of the superior limit of the normalization if the content was added a favourites list and zero(0) value if it was not added. This action is known and evaluated as an implicit, positive and individual action.

- A_9 **- Rejecting a content recommendation:** When a contact recommends content to a user and this user rejects it, it is most likely that he is not interested in it, because he would normally add it to the collection. The value for rejecting a recommendation is calculated through an equation that gives the value of the inferior limit of the normalization if the content was rejected by a user and the zero(0) value if it was not rejected. This action is known and evaluated as an implicit, negative and individual action.

3.3. Data processing

In this section we focus our attention on the explicitation system that processes the implicit indicators to obtain ratings using machine-learning algorithms. In this instance, they are used as an alternative to the previously described converter algorithm. The problem of getting the appropriate score could be modelled as a regression task. Hence, the implicit indicators would be the independent variables and the output would be the category.

Among potential alternatives, we intend to base this component of the system on supervised machine-learning algorithms. Given the nature of the problem, there is a wide range of relevant techniques. For this reason, we consider twelve algorithms that represent different approaches. The list includes CART; decision tables; IBk; K*; LWL; M5P; M5Rules; multilayer perceptrons; radial basis neural networks; reduced error pruning; random forests and support vector regressions.

- CART [7]: Classification and Regression Trees.
- Decision Tables [20]: decision table majority classifier.
- IBK [2]: Implementation of the K-nearest neighbor classifier algorithm.
- K* [10]: instance-based algorithm that determines similarity using an entropy-based distance function.
- LWL [3]: local instance-based weighted learning algorithm. It builds a classifier from the weighted instances.

- M5P [38]: numerical classifier that combines decision trees with linear regressions in order to predict continuous variables.
- M5Rules [17]: this algorithm generates decision lists for regression problems using divide-and-conquer. It builds regression trees using M5, subsequently turning the best leaves into rules.
- Multilayer Perceptron: artificial neural network that simulates the biological process of learning through weight adjustment using backpropagation algorithm. [41].
- RBNN [26]: Radial Basis Neural Networks are another type of artificial neural network. It uses radial basis functions to approximate different regions of the input space depending on their characteristics.
- REPTree [39]: Reduced Error Pruning builds a regression tree based on information variance. The tree is pruned using reduced-error pruning.
- Random Forests [6]: ensemble of classification trees that assigns patterns to categories according to a voting mechanism.
- SVR [43]: support vector regression trained using sequential minimal optimization.

4. Experimental Analysis

4.1. Experimental Setup

The implicit recommendation system was tested on a set of 28 users that interacted with 11 electronic books. The age of the users ranged between 16 and 35 years old and had no previous experience with the reading material they were assigned. The users interacted as they saw fit using the described platform and, as a result, the feedback system captured the implicit indicators. Finally, the users provided explicit feedback on the perceived quality of the content. At this point, we obtained the sample required to use the set of supervised machine learning algorithms that lie at the heart of the recommendation engine. Since not every user interacted will all available content, the final sample has 154 elements. This set consists of 22, 9, 21, 59 and 43 evaluations rated from 1 to 5, respectively.

The comparison of techniques was made using a powerful, well-known and widely used Java package called WEKA (Waikato Environment for Knowledge Analysis) [48] and a 10-fold cross-validation. After some initial tests, we used the parameters reported in table 2

Table 2: Parameters used in the experimental analysis.

CART	
Min. terminal obs.	3
Pruning	Min. cost-complexity pruning
Decision Tables	
Search	BestFirst
Cross validation	Leave one out
IBk	
Neighbors	9
Method	Linear Search
K*	
G. blending param.	45
LWL	
Weighting Kernel	Linear
Classifier	Decision Stump
M5P	
Min. Inst/Leaf	5
M5Rules	
Min. Inst/Leaf	4
Multi Layer Perceptron	
Num. layers	3
Neur. hidden layer	3
Transfer function	Sigmoid

Learning rate	0.1
Momentum	0.2
Max. epochs	1000
RBFN	
Clusters	5
Min. std. dev.	0.1
Ridge	1.0E-8
REPTree	
Min. weight leaf	3
Num. var. prop.	0.001
Num. folds pruning	3
Random Forests	
Trees	25
K Value	$log_2(8) + 1$
Max. depth	Unlimited
SVR	
Complex. param.	1
Epsilon	1.0E-12
Tolerance	0.001
Kernel	Polynomial, exp=1

Given the stochastic nature of some algorithms, the experimental work was repeated 30 times using different seeds for the random number generator. We report the details in the next section.

4.2. Experimental Results

The comparison of algorithms for the recommendation engine is made in terms of mean absolute error. We summarize the results provided by the algorithms in two tables. The first one includes the main descriptive statistics computed across the 30 experiments and the 10 folds used in the cross validation. The second one shows the statistical significance of the observed differences.

As we can see in table 3, the algorithm with the highest accuracy is K*, closely followed by the random forest and the nearest neighbor classifier. Conversely, the alternatives based on M5, and especially M5Rules, together with decision tables, offered relatively poor performance. Among the stochastic alternatives, two function-based algorithms, support vector regression and multilayer perceptrons, provide the best results.

Table 3: Descriptive statistics. Mean Absolute Error over 30 experiments and 10 folds.

	Mean	Median	Var.	Max	Min
CART	0.8296	0.8043	0.0419	1.5569	0.4051
DTable	0.8435	0.8320	0.0432	1.5862	0.2884
IBK	0.7603	0.7485	0.0199	1.2600	0.4007
K*	0.7485	0.7360	0.0257	1.3045	0.3211
LWL	0.7755	0.7723	0.0297	1.5434	0.3596
M5P	0.8959	0.8903	0.0296	1.6406	0.3386
M5Rules	0.8372	0.8313	0.0407	1.6948	0.4052
MLP	0.7992	0.7947	0.0218	1.2733	0.3800
RBFN	0.8274	0.8131	0.0247	1.3400	0.3863
REPTree	0.7973	0.7908	0.0400	1.5997	0.3604
RForest	0.7595	0.7534	0.0291	1.4039	0.3813
SVReg	0.7948	0.7915	0.0232	1.4697	0.4143

If we consider reliability, K* also was the most reliable one, as it provided one of the lowest maximum errors, together with the third smallest variance. The implementation of this instance-based algorithm was beaten in terms of the latter indicator by a related one, IBk, and two algorithms that also provided a competitive average performance, support vector regression and multilayer perceptrons.

Regarding the formal statistical testing of the observed differences, given the lack of normality of the distributions shown by the Kolmogorov-Smirnov test, we use the Wilcoxon test. The statistical significance of the mentioned differences

Table 4: Statistical significance of the reported differences in mean absolute errors.

	CART	DTable	IBK	K*	LWL	M5P	M5Rules	MLP	RBFN	REPTree	RForest
DTable	=										
IBK	−−	−−									
K*	−−	−−	=								
LWL	−−	−−	=	+							
M5P	++	++	++	++	++						
M5Rules	=	=	++	++	++	−−					
MLP	=	−−	++	++	+	−−	−−				
RBFN	=	=	++	++	++	−−	=	+			
REPTree	−	−−	+	++	=	−−	−−	=	−−		
RForest	−−	−−	=	=	=	−−	−−	−−	−−	−−	
SVR	=	−−	++	++	=	−−	−−	=	−−	=	++

is reported in table 4. In this setting we use + to represent situations where the metric for the algorithm in the row is greater than the metric for the equivalent in the column at 5%. If the difference is significant at the 1% conventional level, we use ++. Symbols − and −− have the same interpretation in the opposite direction. Here we can see how the null hypothesis of equality between the median mean prediction errors for K* and the rest of the algorithms can be discarded at 1% for all but random forests and IBk.

5. Conclusions and Future work

In this study we described a recommender system for electronic books based on implicit feedback. The system tracks user interaction with electronic content to provides a rating that could be made available to the rest of the users.

The element of the system that provides the actual rating is the recommendation engine. This component turns the values of eight interaction indicators into a rating that ranges from 1 to 5. The problem handled by the engine can be conceived as a regression task that, based on historic information, models the relation among indicators and ratings.

As we mentioned in the introduction, the variables were identified based on the analysis of user behavior and linear correlations. In this case, we used the whole set to fit non-linear models that capture the connection between the independent variables and the explicit rating provided by the users. Even though the set might be extended in the future to improve the accuracy, the current one seems to be a good starting point.

The number of algorithms available to perform this function is wide, hence the need to benchmark them. The alternatives considered in this paper were CART; decision tables; IBk; K*; LWL; M5P; M5Rules; multilayer perceptrons; radial basis neural networks; reduced error pruning; random forests and support vector regressions. This selection considers different families of algorithms including decision trees, function-based approaches or lazy strategies.

The algorithms were assessed in terms of mean absolute errors. Out of the alternatives tested, K*, random forests, and IBk offered the best results. K* did beat the other two but, given that differences were not significant at conventional levels, we cannot confirm its superiority. If, in addition to performance, we consider consistency across folds and experiments, K* and IBk were among the most reliable ones, as they offered both some of the lowest variances and some of the best worst results.

At this point, the idea of assigning ratings to eBooks according to implicit indicators is promising. Having said that, there are a number of ways to extend this work. Future research avenues could include testing new indicators and algorithms while extending sample size would be beneficial. Also, we consider necessary to develop a model of artificial intelligence based on NLP that classifies the comments and other actions as positive, negative, neutral or other values, automatically, and thus improve the recommender system and the result obtained in this research.

6. Acknowledgements

The authors would like to acknowledge the financial support of the Spanish Ministry of Industry, Tourism and Trade under grant TSI-020110-2009-137 and FEDER financial support from the projects TIN2013-40658-P and TIN2016-75850-R.

References

[1] Adomavicius, G. and Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749.

[2] Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-Based Learning Algorithms. *Machine Learning*, 6(1):37–66.

[3] Atkeson, C. G., Moore, A. W., and Schaal, S. (1997). Locally Weighted Learning. *Artificial Intelligence*, 11:11–73.

[4] Baldominos Gómez, A., Albacete, E., Merrero, I., and Saez, Y. (2016). Real-Time Prediction of Gamers Behavior Using Variable Order Markov and Big Data Technology: A Case of Study. *International Journal of Interactive Multimedia and Artificial Intelligence*, 3(6):44.

[5] Bobadilla, J., Gutiérrez, A., Ortega, F., and Zhu, B. (2018). Reliability quality measures for recommender systems. *Information Sciences*, 442-443:145–157.

[6] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[7] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*, volume 19.

[8] Christidis, K. and Mentzas, G. (2013). A topic-based recommender system for electronic marketplace platforms. *Expert Systems with Applications*, 40(11):4370–4379.

[9] Claypool, M., Brown, D., Le, P., and Waseda, M. (2001). Inferring User Interest. *IEEE Internet Computing*, 5(6):32–39.

[10] Cleary, J. G. and Trigg, L. E. (1995). K*: An instance-based learner using an entropic distance measure. In Prieditis, A. and Russell, S., editors, *Machine Learning Proceedings 1995*, pages 108 – 114. Morgan Kaufmann, San Francisco (CA).

[11] Cunha, T., Soares, C., and de Carvalho, A. C. (2018). Metalearning and Recommender Systems: A literature review and empirical study on the algorithm selection problem for Collaborative Filtering. *Information Sciences*, 423:128–144.

[12] Elahi, M., Ricci, F., and Rubens, N. (2016). A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, 20:29 – 50.

[13] García, C. G., Meana-Llorián, D., García-Díaz, V., and Núñez-Valdez, E. R. (2017). Social Recommender System. In *Proceedings of the 4th Multidisciplinary International Social Networks Conference on ZZZ - MISNC '17*, pages 1–7, New York, New York, USA. ACM Press.

[14] Gena, C., Brogi, R., Cena, F., and Vernero, F. (2011). The Impact of Rating Scales on User s Rating Behavior. *User Modeling Adaption and Personalization 19th International Conference UMAP 2011*, 6787:123–134.

[15] Gomez-Uribe, C. A. and Hunt, N. (2015). The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manage. Inf. Syst.*, 6(4):13:1–13:19.

[16] González Crespo, R., Sanjuán Martínez, O., Cueva Lovelle, J. M., Pelayo García-Bustelo, B. C., Gayo, J. E. L., and de Pablo, P. O. n. (2010). Recommendation System based on user interaction data applied to intelligent electronic books. *Computers in Human Behavior*, In Press,:–.

[17] Holmes, G., Hall, M., and Prank, E. (1999). Generating rule sets from model trees. In Foo, N., editor, *Advanced Topics in Artificial Intelligence*, pages 1–12, Berlin, Heidelberg. Springer Berlin Heidelberg.

[18] Jawaheer, G., Szomszor, M., and Kostkova, P. (2010). Comparison of implicit and explicit feedback from an online music recommendation service. In *Proceedings of the 1st International Workshop on Information Heterogeneity*

and Fusion in Recommender Systems, HetRec '10, pages 47–51, New York, NY, USA. ACM.

[19] Kelly, D. and Belkin, N. J. (2001). Reading time, scrolling and interaction: Exploring implicit sources of user preference for relevance feedback. *SIGIR 01 Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 408–409.

[20] Kohavi, R. (1995). The power of decision tables. In *Proceedings of the 8th European Conference on Machine Learning*, ECML'95, pages 174–189, Berlin, Heidelberg. Springer-Verlag.

[21] Kunaver, M. and Požrl, T. (2017). Diversity in recommender systems A survey. *Knowledge-Based Systems*, 123:154–162.

[22] Lee, H. and Park, S. J. (2007). MONERS: A news recommender for the mobile web. *Expert Systems with Applications*, 32(1):143–150.

[23] Linden, G., Smith, B., and York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80.

[24] Lucas, J. P., Luz, N., Moreno, M. N., Anacleto, R., Almeida Figueiredo, A., and Martins, C. (2013). A hybrid recommendation approach for a tourism system. *Expert Systems with Applications*, 40(9).

[25] Martinez-Cruz, C., Porcel, C., Bernab-Moreno, J., and Herrera-Viedma, E. (2015). A model to represent users trust in recommender systems using ontologies and fuzzy linguistic modeling. *Information Sciences*, 311:102 – 118.

[26] Moody, J. and Darken, C. J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294.

[27] Moreno, A. and Redondo, T. (2016). Text Analytics: the convergence of Big Data and Artificial Intelligence. *International Journal of Interactive Multimedia and Artificial Intelligence*, 3(6):57.

[28] Moreno, M. N., Segrera, S., López, V. F., Muńoz, M. D., and Sánchez, a. L. (2016). Web mining based framework for solving usual problems in recommender systems. A case study for movies' recommendation. *Neurocomputing*, 176.

[29] Nilashi, M., Bagherifard, K., Rahmani, M., and Rafe, V. (2017). A recommender system for tourism industry using cluster ensemble and prediction machine learning techniques. *Computers & Industrial Engineering*, 109:357–368.

[30] Núñez Valdez, E. R., Cueva Lovelle, J. M., Sanjuán Martínez, O., García-Díaz, V., ordoñez de Pablos, P., and Montenegro Marín, C. E. (2012). Implicit Feedback Techniques on Recommender Systems applied to Electronic Books. *Computers in Human Behavior*.

[31] Núñez Valdez, E. R., Cueva Lovelle, J. M., Sanjuán Martínez, O., Montenegro Marín, C. E., and Infante Hernandez, G. (2011). Social voting techniques: A comparison of the methods used for explicit feedback in recommendation systems. *International Journal of Interactive Multimedia and Artificial Intelligence*, I:61–66.

[32] Núñez Valdez, E. R., Lovelle, J. M. C., Hernández, G. I., Fuente, A. J., and Labra-Gayo, J. (2015). Creating recommendations on electronic books: A collaborative learning implicit approach. *Computers in Human Behavior*, 51:1320 – 1330.

[33] O'Donovan, J. and Smyth, B. (2005). Trust in recommender systems. *Proceedings of the 10th international conference on Intelligent user interfaces IUI 05*, 05pages(June):167.

[34] Papagelis, M., Plexousakis, D., and Kutsuras, T. (2005). Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Proceedings of the Third international conference on Trust Management*, iTrust'05, pages 224–239, Berlin, Heidelberg. Springer-Verlag.

[35] Park, Y., Oh, J., and Yu, H. (2017). RecTime: Real-Time recommender system for online broadcasting. *Information Sciences*, 409:1–16.

[36] Pommeranz, A., Broekens, J., Wiggers, P., Brinkman, W.-P., and Jonker, C. M. (2012). Designing interfaces for explicit preference elicitation: a user-centered investigation of preference representation and elicitation process. *User Modeling and UserAdapted Interaction*, 22(22):357–397.

[37] Porcel, C., Tejeda-Lorente, A., Martnez, M., and Herrera-Viedma, E. (2012). A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. *Information Sciences*, 184(1):1 – 19.

[38] Quinlan, J. (1992). Learning with continuous classes. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 343–348.

[39] Quinlan, J. (1999). Simplifying decision trees. *International Journal of Human-Computer Studies*, 51(2):497 – 510.

[40] Resnick, P. and Varian, H. R. (1997). Recommender systems. *Commun. ACM*, 40(3):56–58.

[41] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

[42] San-Miguel Carrasco, R. (2016). Detection of Adverse Reaction to Drugs in Elderly Patients through Predictive Modeling. *International Journal of Interactive Multimedia and Artificial Intelligence*, 3(6):52.

[43] Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222.

[44] Taghipour, N. and Kardan, A. (2008). A hybrid web recommender system based on Q-learning. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 1164–1168, New York, NY, USA. ACM.

[45] Tejeda-Lorente, A., Porcel, C., Peis, E., Sanz, R., and Herrera-Viedma, E. (2014). A quality based recommender system to disseminate information in a university digital library. *Information Sciences*, 261:52 – 69.

[46] Verbert, K., Manouselis, N., Ochoa, X., Wolpers, M., Drachsler, H., Bosnic, I., Member, S., and Duval, E. (2012). Context-aware Recommender Systems for Learning: a Survey and Future Challenges. *IEEE Transactions on Learning Technologies*, 6(1):2007.

[47] Wang, P. (1998). Why recommendation is special? In *Workshop on Recommender Systems, part of the 15th National Conference on Artificial Intelligence*, volume 15, pages 111–113, Madison, Wisconsin, EUA). AAAI-98.

[48] Witten, I. H., Frank, E., Trigg, L., Hall, M., Holmes, G., and Cunningham, S. J. (1999). Weka : Practical Machine Learning Tools and Techniques with Java Implementations. *Seminar*, 99:192–196.

[49] Yu, L., Huang, J., Zhou, G., Liu, C., and Zhang, Z.-K. (2017). TIIREC: A tensor approach for tag-driven item recommendation with sparse user generated content. *Information Sciences*, 411:122–135.

[50] Zhang, Z.-K., Zhou, T., and Zhang, Y.-C. (2011). Tag-Aware Recommender Systems: A State-of-the-Art Survey. *Journal of Computer Science and Technology*, 26(5):767–777.