Resource Optimisation in Cloud Computing: Comparative Study of Algorithms Applied to Recommendations in a Big Data Analysis Architecture

Submitted: 18th April 2021; accepted: 9th February 2022

Aristide Ndayikengurukiye, Abderrahmane Ez-Zahout,
Akou Aboubakr, Youssef Charkaoui, Omary Fouzia

DOI: 10.14313/JAMRIS/4-2021/28

Abstract:
Recommender systems (RS) have emerged as a means of providing relevant content to users, whether in social networking, health, education, or elections. Furthermore, with the rapid development of cloud computing, Big Data, and the Internet of Things (IoT), the component of all this is that elections are controlled by open and accountable, neutral, and autonomous election management bodies. The use of technology in voting procedures can make them faster, more efficient, and less susceptible to security breaches. Technology can ensure the security of every vote, better and faster automatic counting and tallying, and much greater accuracy. The election data were combined by different websites and applications. In addition, it was interpreted using many recommendation algorithms such as Machine Learning Algorithms, Vector Representation Algorithms, Latent Factor Model Algorithms, and Neighbourhood Methods and shared with the election management bodies to provide appropriate recommendations. In this paper, we conduct a comparative study of the algorithms applied in the recommendations of Big Data architectures. The results show us that the K-NN model works best with an accuracy of 96%. In addition, we provided the best recommendation system is the hybrid recommendation combined by content-based filtering and collaborative filtering uses similarities between users and items.

Keywords: *Cloud computing, Big Data, IoT, Recommender system and KNN algorithm*

1. Introduction

Nowadays, many countries are trying to put in place laws to make it easier for the public to express their opinion on a given issue and especially to make a choice of their leaders through voting. The conventional procedure is manual or paper voting which sometimes does not facilitate counting and is not convenient for voters which sometimes decreases the voting rate. With the evolution of information and communication technologies, electronic voting has become a necessity to overcome these problems and facilitate voting for any geographical area to participate.

This technological evolution has generated an explosion of a large mass of digital data, which makes it imperative to find solutions to enable adequate storage and analysis, thus giving rise to the concept of Big Data. Indeed, Big Data requires enormous storage and processing capacities that conventional methods cannot offer. To promote the deployment of Big Data comes the concept of Cloud Computing which allows access and use of computer resources such as storage and network on demand. In other words, it provides users with computing services that allow them to save and manipulate data, install models and access multiple resources on the web, according to their needs. One of the most advantageous features of the cloud is its elasticity, as it allows computing capacity to be adapted to demand and only the necessary resources to be allocated [1]: The most popular cloud-based services include Amazon web services, SAP (System Applications and Products), Oracle, Cloudera, Databricks, etc.

With this rapid increase in the volume of digital data on websites and applications worldwide has led to the problem of data overload which makes it difficult to extract relevant information. As a response to this problem, optimisation algorithms and recommender systems have been proposed and these various services mentioned above serve as a platform for the development and evaluation of recommender systems.

When search engines are not sufficient to process this large amount of online information, recommender systems try to guess what information or goods and services we really need. Any recommender system needs to know a user profile that, for example, contains the user's preferences. This profile can be acquired implicitly by analyzing the user's past behavior, or explicitly by asking the user about their preferences. In this case, recommender systems that take into account a community of users are often called collaborative approaches [2], while those based on the description of articles are called content-based approaches [3]. The hybrid approach [4] is a combination of two different methodologies or systems that aims to create a new model and is strongly rooted in information retrieval and information filtering. User preferences and article descriptions are somehow compatible so that it is possible to compare them and find an article that matches the user's preferences.

One of the central issues in recommender systems is how to model and then compare user preferences and item descriptions in a flexible and natural way. Obviously, user preferences, as well as item descriptions, can be imprecise, incomplete, subjective, and of different importance. The first approach is based on the use of the fuzzy set theory adopted by Yager [5]. Another approach, which uses fuzzy clustering as a tool for a recommendation, was presented in [6].

Throughout political elections, voters have to make a decision about who to vote for. This decision has become particularly difficult in recent times when much more attention is paid to personalities, images, and campaign events than to party manifestos and policy issues. In addition, various applications have been created to help voters easily pick their favorite candidates. The general idea of these systems is to put x on the symbols of the two candidates. Finally, these choices are compared to select the candidates that are accessible.

In this paper, we present an overview of the different algorithms used in recommender systems and the different tools used in Big Data analysis. We also make a comparative study of some of these algorithms in recommender systems applied to a vote.

The rest of this article is organized as follows. Section 2 presents the different tools used in Big Data analysis and a brief overview of recommender systems. Section 3 presents some of the work done on this topic. Section 4 presents the methodology used in our work. Section 5 presents the results obtained in this article. We end with a conclusion.

2. Literature Review

2.1. Big Data Framework

HDFS. Hadoop Distributed File System (HDFS) is an architecture that consists of NameNode which is a server that regulates file access by clients and manages the file system namespace. HDFS allows user data to be stored in files through a file namespace. In HDFS the opening, closing, and renaming of files and directories are operations performed by the NameNode and also determines the mapping of blocks to Datanodes [4].

HDFS has the ability to scale to applications with datasets ranging in size from a few gigabytes to a few terabytes. It can also scale to hundreds of nodes in a single cluster and can provide high bandwidth for aggregated data [7].

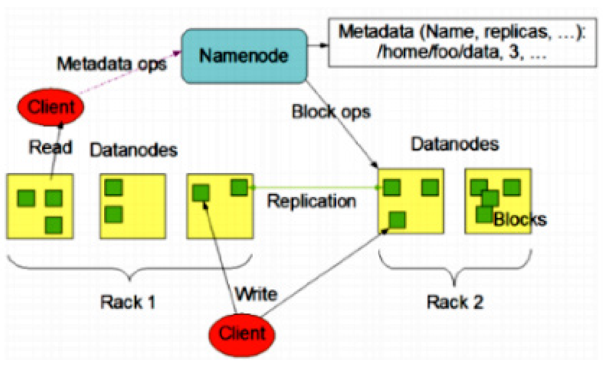


Fig. 1. Architecture of HDFS

Spark. In advanced data analysis in Big Data, Spark is the most widely used in-memory distributed processing framework. The reasons for its success include its Application Programming Interface (API) and a set of features that allow querying large amounts of data using Structured Query Language (SQL) to distill complex Machine Learning (ML) models using the most popular algorithms. Spark also reduces the execution time of an application by reducing the number of read/write operations on the disk [5].

Despite these advantages, it can mask some of the problems caused by a distributed code execution mechanism, making testing inefficient. Creating Dockers containers is the most efficient way to create a test environment that resembles the cluster environment. The advantage of this approach is that applications can be tested on different versions of the Framework by simply changing some parameters in the configuration file [6].

Another advantage is that it is designed to be run on different types of clusters. Cluster managers such as Yet Another Resource Negotiator(YARN), the Hadoop Platform Resource Manager, Mesos, or Kubernetes are supported by a spark [4]. The figure shows the architecture of HDFS [3].

MapReduce. The huge amount of data provides companies with many opportunities. One of the biggest problems is to be able to process it efficiently on traditional systems and therefore to turn to new solutions specifically designed for this purpose.

MapReduce makes data processing easier. It breaks down massive volumes of data into smaller chunks. These pieces of data are then processed in parallel on hadoop servers. Afterward, it sends the consolidated result to the application [8].

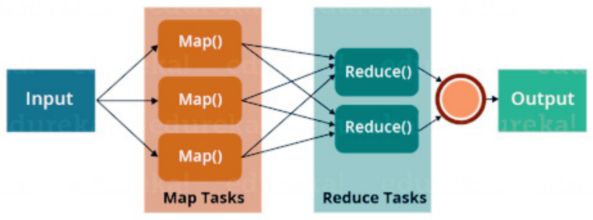


Fig. 2. Execution of MapReduce Job

2.2. Recommender Systems

With the evolution of technology and especially the advent of the web, the amount of data to be exploited and processed has become very large, making it difficult to search and find. One of the computer techniques that has been developed to extract relevant information is recommendation systems. They allow the user to be guided through the data in order to find the most relevant data.

In RS, the usefulness of the item is usually represented by a score that indicates how a user liked a particular item. The problem that arises is to solve the estimation of scores for items that have not been evaluated by the user. Whenever it is possible to estimate scores for items that have not yet been evaluated, users are recommended items with a higher estimated score [9].

RS are systems that produce personalised recommendations with the aim of guiding the user in a personalised way towards interesting or useful elements when many options are available. They have three phases: the first is the data collection phase, which consists of collecting relevant information about a user; the second is the learning phase, which uses the collected data to filter and exploit the user's features; and the final phase is the recommendation phase, which predicts or recommends the type of product that the user may prefer.

Collaborative filtering. This type of recommendation system aims to automate the recommendations that users sharing the same interests can make to each other. They allow a user to find the information that interests him/her based on the same tastes and preferences as other users. In this type of recommendation two methods can be used, namely user-based recommendation and item-based recommendation.

For example, the user-based recommendation is a technique used to predict which items a user might like based on the ratings given to that item by other users who have similar tastes to the target user. In this type of recommendation, the weight can be deduced as the correlation between users and , which is calculated by the following formula:

$$\boldsymbol{Sim(a,b)} = \frac{\sum_p (\boldsymbol{r_{ap}} - \boldsymbol{\bar{r}_a})(\boldsymbol{r_{ab}} - \boldsymbol{\bar{r}_b})}{\sqrt{\sum(\boldsymbol{r_{ap}} - \boldsymbol{\bar{r}_a})^2}\sqrt{\sum(\boldsymbol{r_{bp}} - \boldsymbol{\bar{r}_b})^2}} \quad (1)$$

In some cases, a user may have similarities to some users and few similarities to others. This means that the ratings given to a particular item by the most similar users should be given more weight than those given by less similar users, and so on. In this case, each user's rating is multiplied by a similarity factor, which is given by the formula:

$$\boldsymbol{r_{up}} = \boldsymbol{\bar{r}_b} + \frac{\sum_{i\in users} \boldsymbol{sim(u,i)} * \boldsymbol{\bar{r}_{ip}}}{\sum_{i\in users} |\ \boldsymbol{sim(u,i)}\ |} \quad (2)$$

Content-based filtering. To generate recommendations, these types of RS rely on user profiles and item descriptions while suggesting to the user items that best match their preferences based on similarity. In fact, profiles are built for users as well as for products. Thus, these descriptions (profiles, users) are represented as vectors of weighted terms. The formula below represents the calculation of similarities to predict user interests [10][11].

$$\boldsymbol{Sim(a,b)} = \boldsymbol{cos\theta} = \frac{\vec{a}.\vec{b}}{\|\vec{a}\|.\|\vec{b}\|} = \sum_{i=1}^{n} \frac{a_i b_i}{\sqrt{a_i^2 . b_i^2}} \quad (3)$$

Some authors in [12] have used probabilistic methods in content-based recommendations. The formula below represents the Bayesian classifier model checking the membership of an example a to a class taking into account the attributes.

$$\boldsymbol{P(C_i)} \prod_k \boldsymbol{P(A_k V_{kj} C_i)} \quad (4)$$

The other method that is used in this type of recommendation is the Term Frequency-Inverse Document Frequency (TF-IDF), which makes it possible to determine in what proportions certain words in a website, for example, or in a document, are evaluated in relation to the rest of the text. For example, the frequency criterion for increasing the relevance of a site, without the keyword density playing a major role. The TF is defined as follows[13]:

$$TF = \frac{Number of occurrences of term}{Total number of terms} \quad (5)$$

The other measure will focus on measuring the word as a function of document usage rather than measuring it as a function of frequency. The formula below defines the IDF[13]:

$$\boldsymbol{IDF} = \boldsymbol{log} \frac{Total number of documents}{Number of documents containing term} \quad (6)$$

The following formula calculates the TF weight:

$$W = IDF * TF \quad (7)$$

Hybrid recommender systems. Hybrid systems combine two or more techniques to achieve better performance, such as content-based filtering and collaborative filtering. The limitations of one technique can be overcome by another technique

Today, recommender systems have been studied in several areas such as smart cities[14], education[15], e-commerce[16], e-learning [17].

2. Related Works

In the past, many researchers have devoted much of their work to the development of electoral systems or IoT-based technologies. Previous work was mainly focused on a novel IoT recommender system that uses publicly available application data as a source for personalization. In [8], the authors proposed a system that infers the user's physical objects from the exploration of applications installed on their mobile devices such as smartphones and tablets and builds a digital inventory of each user's physical objects. These inventories can then be used to create personalized recommendations. In [18], the authors used recommendation systems for IoT users. In their work, they proposed a recommender system that works on the basis of the graph created between users, objects, and services while recommending an IoT device that is suitable for the customers based on their needs and interests. To recommend the best option to the users, they also took into account the characteristics of users and services. To cope with the difficult task of selecting services in cloud computing, the authors in [19]. proposed a recommendation system for these services to help different cloud users better find what they are looking for. To generate reliable recommendations, they adopted the formal fuzzy concept analysis using the lattice representation. This method allowed them to transform the repository of different cloud services into a set of small clusters, where relationships between high-quality services and high-quality services are highlighted.

3. Our Methods

The process of designing the recommendation system with the progression of Internet of Things (IoT), this case of cloud computing is evolving rapidly using clusters. He gives elasticity to users of associated information technology (IT) services to save and manage data, install models and easily provide recommendations as and when the best needs. On the other hand, we have proposed a hybrid recommendation approach to the context to get the candidates are selected.

Our approach combines two different methods: content filtering and collaborative filtering. Each method was adapted to a stage-specific to the voter. First, the approach contained attempts to create a profile used to predict ratings on unseen items. Secondly, the collaborative approach uses the products using user notes (explicit or implicit) from the given history. It works in a database of user preferences for the items (Items) that are similar. Finally, we propose a more efficient recommendation system that is based on the hybrid recommendation to put to eliminate the inconvenience in our born gift. Fig. 3. shows the methodology of our work.

Furthermore, we have implemented fourteen algorithms, but the best respectively of them are: K-NN, K-NN with GridSearchCV, SVM, and Decision Tree (DT) Classifier of precisions: 96%, 96%, 92% and 89%, such that the procedures of each algorithm is written:

Algorithm K-NN. The K-Nearest Neighbors (K-NN) algorithm is a machine learning algorithm that belongs to the class of simple and easy to implement supervised learning algorithms that can be used to solve classification problems and regression.

Algorithm: K-NN

```
Input: the training set D, test objet x, category label set C.
Output: the category c(x) of test object x, c(x) belongs to C.
  1.    Begin
  2.    For each y belongs to D calculate the
           distance D(y,x) between y and z.
        end for
  3.    Select the subset N from the data set D, the N
contains k training samples which are the k narest neighbors of
the test sample x.
  4.    Calculate the category of x:
        $c_k = \arg max_{c \in C} \sum_{y \in N} \mathbf{1}(c = class(y))$
  5.    End
```

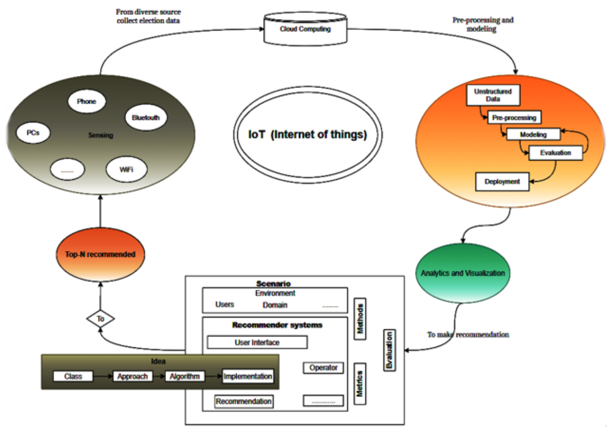


Fig. 3. Proposed analysis architecture

Algorithm SVM. SVM (Support Vector Machine) is supervised learning model with associated learning algorithms that analyze data for classification and regression analysis.

Algorithm: SVM

```
Input: Determine the various training and test data
Output: Determine the calculate accuracy
Select the optimal value of cost and gamma for SVM
While (stopping condition is not met) do
1.      Implement SVM train step for each data point.
2.      Implement SVM classify for testing data point.
End while

Return accuracy
```

Algorithm Decision Tree Classifier. The decision tree algorithm belongs to the family of supervised machine learning algorithms

Algorithm: Decision Tree

```
Input: S, where S= set of classified instances
Output: Decision Tree
Require: $S \neq \emptyset$, $num\_attibutes > 0$
1.      Procedure Build Tree
2.      Repeat
3.        maxGain ← 0
4.          splitA ← null
5.            e ← Entropy(Attributes)
6.          For all Attributes a in S do
7.               gain ← InformationGain(a,e)
8.               If gain > maxGain then
9.                   maxGain ← gain
10.                  splitA ← a
11.              End if
12.         End for
13.           Partition(S,splitA)
14.           Until all partitions processed
15.     End procedure
```

We evaluated and compared the performance of several recommendation algorithms. We conducted an experimental study on Databricks and Jupyter notebook, a publicly available election dataset, to analyse and evaluate several recommendation algorithms, namely: Term frequency-inverse document frequency [20], Rocchio [21], latent semantic analysis [22], K-nearest neighbours [23], Naïve Bayes [24], decision tree classifier[25], support vector machine [26], Linear regression(LR) [27], K-means [28], Slope One [29], Non-matrix factorisation [30], Singular decomposition vector [10], Co-clustering [31], Pearson correlation [32] and Artificial neural network5(ANN) [33]. The evaluation of the algorithms is based on precision, recall, f1 score, accuracy, area under the curve, RMSE, MSE, and MAE.

Definitions:

1. True positive (tp) = number of instances correctly predicted as a vote.
2. False positives (fp) = number of instances incorrectly predicted as votes.
3. True negative (tn) = number of instances correctly predicted as non-voting.
4. False negative (fn) = number of instances incorrectly predicted as non-voting.

a) **Precission:** this is the measure of quality. It is the measure of how well the predicted positive observations match the total sum of the predicted positive observations. It answers the following question: Of all the instances labeled as votes, how many times did this turn out to be true? It can be calculated as shown in equation 1:

$$\text{Precission} = \frac{tp}{tp + fp} \tag{8}$$

b) **Recall:** this is the measure of completeness. It determines the ratio of adequately predicted positive observations to the set of observations in the defined class – yes. Otherwise, it answers the following question: Of all the voting predictions that are true, how many have we labeled? Equation 2 shows how to calculate this:

$$\text{Recall} = \frac{tp}{tp + fn} \tag{9}$$

c) **F-score:** is the weighted average of both precision and recall values. Hence, this score regards both false positives and false negatives. It can be calculated as given in equation 3:

$$\text{F} - \text{score} = 2 * \frac{\text{precission} * \text{recall}}{\text{precission} + \text{recall}} \tag{10}$$

d) **Accuracy:** is a ratio of fittingly predicted instances to the total instances. The accuracy of an algorithm is its capability to differentiate the voting and no-voting cases accurately. The accuracy of a system can be measured as given in equation 4:

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn} \tag{11}$$

e) **RMSE:** Root Mean Square Error is a measure of the deviation of predictions from the actual value (calculated as the square root of an average square) as given in equation 5:

$$\text{RMSE} = \sqrt{\frac{\sum_{c\in C}\sum_{i\in Itest_C}(reco(c,i) - r_{c,i})^2}{\sum_{c\in C^{Itest_C}}}} \tag{12}$$

f) **MSE:** Mean Squared Error is the average squared between the estimated values and the actual value as given in equation 6:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}(Y - \hat{Y})^2 \tag{13}$$

g) **MAE:** Mean Absolute Error is the average over the verification sample of the absolute values of the differences between forecast and the corresponding their direction, as given in equation 7:

$$\text{MAE} = \frac{\sum_{c\in C}\sum_{i\in Itest_c} \mid reco(c,i) - r_{c,i} \mid}{\sum_{c\in C} Itest_c} \tag{14}$$

h) **AUC:** specifies the degree to which a model is proficient at differentiating between the given classes. Higher the value of AUC, superior the model is at foreseeing 0s as 0s and 1s as 1s i.e., the model can accurately distinguish between voting and no-voting cases.

5. Results and Discussion

During our work we used two programming languages: Python and Apache Spark in Jupyter, Colab, and Cloud Databricks in an Acer Predator computer with the following performance: operating system (window 10 Home), Processor Intel R core i7, speed 2.60 GHz, part of display & graphics (Graphics controller manufacturer NVIDIA R, Model GForce R GTX 1660Ti, Memory Capacity UP to 6 GB) and storage 1 TB, Total hard drive capacity 1 TB, 16 Go RAM.

In this paper, we consider the hybrid approach the strongest recommender system for recommending a candidate to an elector.

In this project, we use data from this link (https://www.kaggle.com/unanimad/us-election-2020?select=governors_county_candidate.csv:shows all files used on format CSV: comma separated values) as test data. What distinguishes our system is the fact that we take into consideration the possibility that the candidate intentionally did not vote. Therefore, we have to find the highest count of voters to candidates using similar methods. More precisely, we will solve our problem by steps: import our data, pre-processing our data, also modeling, evaluations (like recall, precision, f1-measure, RMSE, MSE, MAE, AUC) and compare our results of algorithms for making the strongest recommendation.

From Table 1 and Table 2, it can be observed that K-Nearest Neighbors and SGDC with GridSearchCV have performed really well with an accuracy of 96% on 7-fold cross-validation. SVM follows next with 92% accuracy. Further, Rocchio's algorithm has a minimum accuracy of 66% as compared with other algorithms. Figures 1 to 12, depicts and compares the performance of the used algorithms on the basis of precision, recall, f1-score, RMSE, MSE, MAE, and AUC (Area under the ROC Curve evaluated) for the election dataset.

Based on the results, we can clearly observe that the K-NN and SGDC GridSearchCV outperformed all the other classifiers. In the future, we will use those classifiers in our smart Recommender system, where personalized recommendations would be generated to each user based on his condition voting or not voting.

Tab. 1. Comparison between the algorithms by the metrics: precision, recall, F1-score, and accuracy

Algorithm	Precision	Recall	Fi-score	Accuracy
Rocchio's	0.77	0.56	0.66	0.66
KNN	0.96	0.96	0.96	0.96
SGDC GridSearch CV	0.96	0.96	0.96	0.96
DTC	0.88	0.89	0.88	0.89
SVM	0.85	0.92	0.89	0.92

Tab. 2. This table shows the comparison between the algorithms by the metrics: MSE, RMSE, MAE, and AUC

Algorithm	MSE	RMSE	MAE	AUC
Rocchio's	22.893	4.784	2.17	0.96
KNN	1871268587.5	43258.16	565.19	1.0
SGDC GridSearch CV	195255897.58	12460.17	429.92	1.0
Naïve Bayes	231533.567	481.17	16.638	0.99
DTC	231514.49	481.15	15.831	0.89
SVM	0.85	0.92	0.89	0.92
LR	1.0	1.0	1.0	1.0
ANN	0.991	0.9995	0.01	0.5

Tab. 3. This table shows the comparison between the algorithms by the metrics: MSE, RMSE, MAE, and MAE

Algorithm	MSE	RMSE	MAE
Slope One	0.59	0.768	0.760
NMF	40110374.3	2002.608221	2002.56
SVD	4010369.24	2002.608225	2002.56
SVDpp	231533.567	481.17	16.638

According to Table 3, the slope one algorithm has the lowest errors in MSE, RMSE and MAE compared to all other algorithms followed by the SVDpp algorithm. It can be seen that the SVD algorithm is the one that admits colossal values in terms of errors, which induces slope one represents the best algorithm that better explains our variable which moreover presents an MSE of 0.59<1, RMSE of 0.768<1, and MAE= 0.760<1.

The rest of the curve is made up of the precision and recall values for the threshold values which are between 0 and 1. Our goal is to make the curve as close as (1,1), which means a good precision and a good recall which is in the first curve of Fig. 4. in class 11 of area 87%.

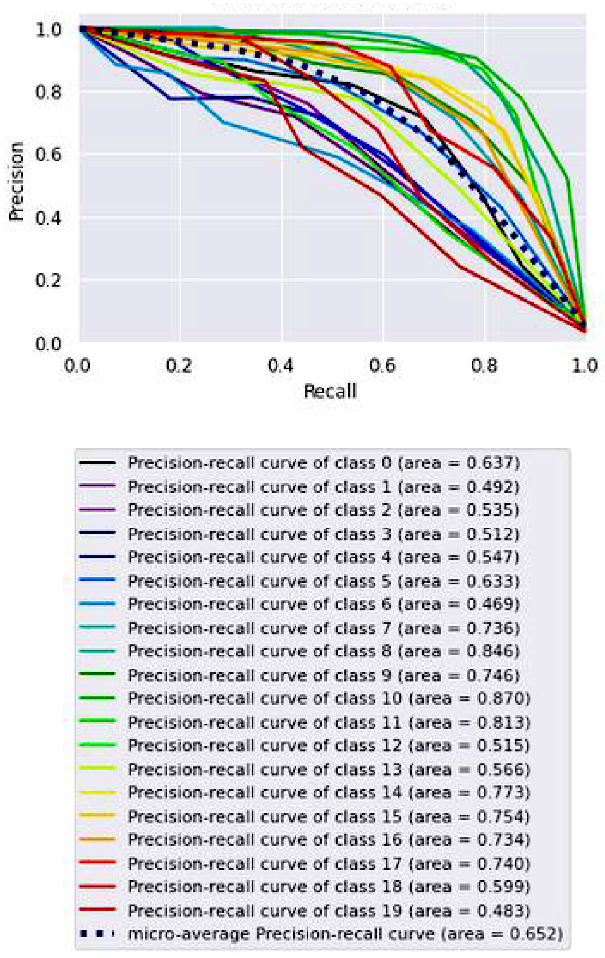


Fig. 4. ROC (Prediction Vs Test) of Rocchio's

In the Second curve Fig. 5., when 0.5 < AUC <1, there is a good chance that the classifier is able to distinguish positive class values from negative class values. This is because the classifier is able to detect more numbers of true positives and true negatives than false negatives and false positives. In our case, we have a multi-class binary classification problem using the technique One vs All. For this, the best class of ROC is Class 8 of the 96% area.

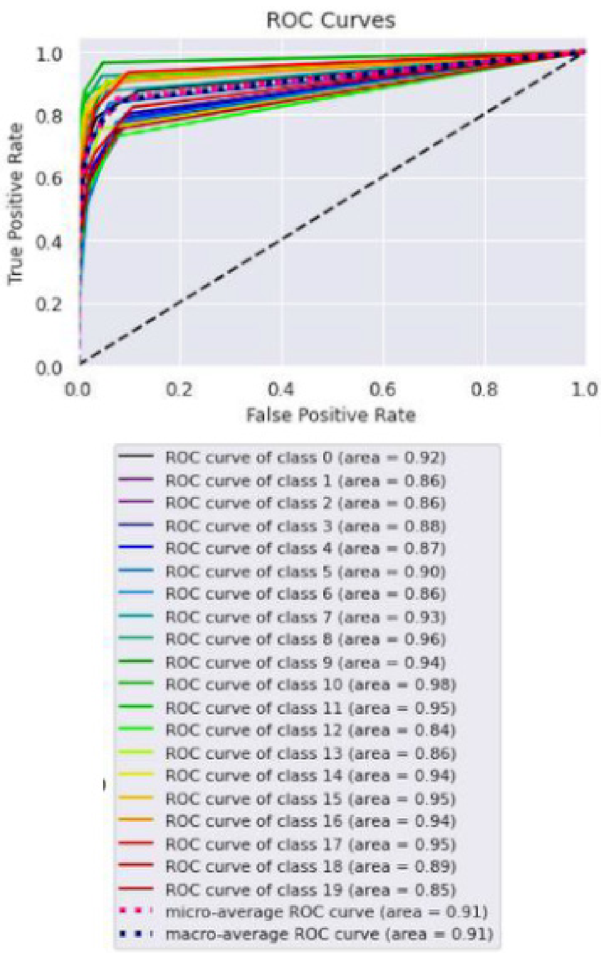


Fig. 5. ROC (Probability Vs Test) of Rocchio's

In Fig. 6., we have AUC = 1, then the classifier is able to distinguish perfectly all the positive and negative class points correctly. If, however, the AUC had been 0, then the classifier would have predicted all the negatives as positive and all the positives as negative.

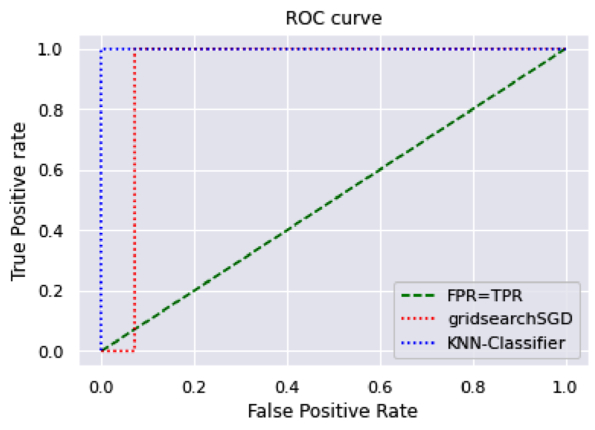


Fig. 6. ROC of SGDC GridSearchCV and KNN

In Fig. 7, we have AUC = 1, then the classifier is able to distinguish perfectly all the positive and negative class points correctly. If, however, the AUC had been 0, then the classifier would have predicted all negatives as positives and all positives as negatives.

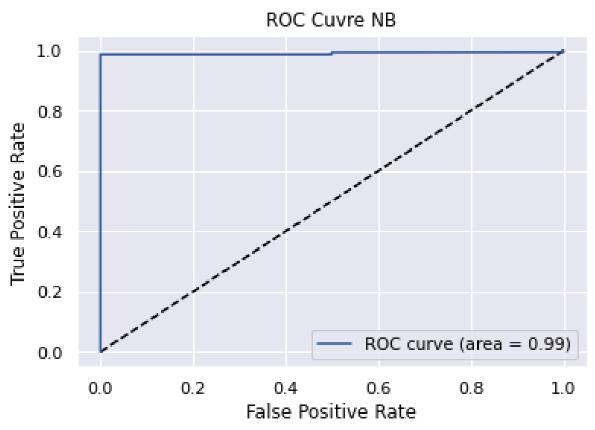


Fig. 7. ROC of NB

In Fig. 8., we have AUC = 1, then the classifier is able to distinguish perfectly all the positive and negative class points correctly. If, however, the AUC had been 0, then the classifier would have predicted all negatives as positives and all positives as negatives.

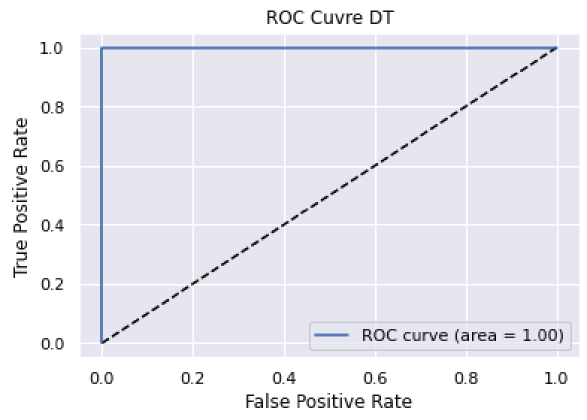


Fig. 8. ROC of DT

Fig. 9. shows that we can clearly see that the training score is always around the maximum of 1.0 and that the validation score, could be increased with more training samples of 0.88.

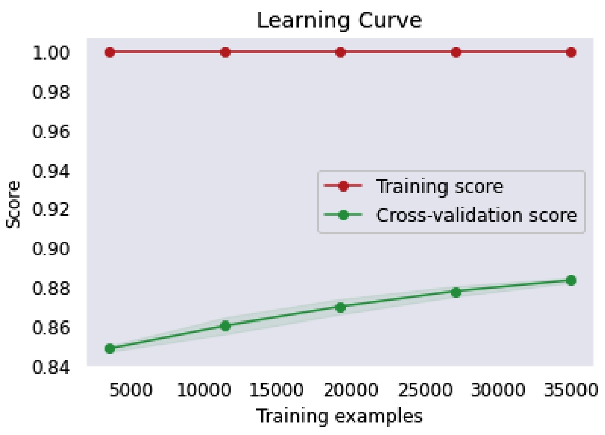


Fig. 9. Learning curve of DT

In Fig. 10., we can see clearly that the training score is still around the maximum of score 0.92 and the validation score could be increased with more training samples of the maximum score is 0.92.

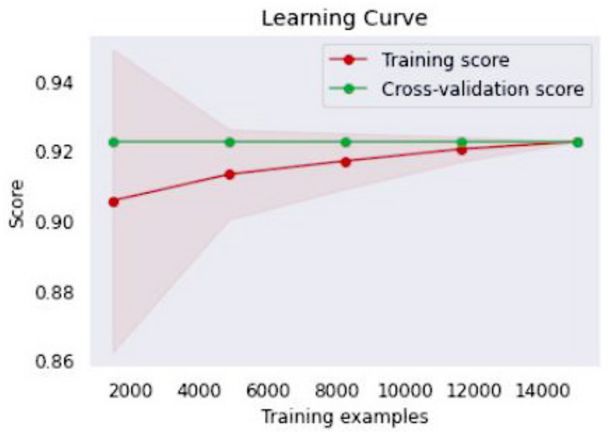


Fig. 10. Learning curve of SVM

Fig. 11. clearly shows that the training score is always around the maximum of 0.0 and that the validation score could be lowered with more training samples.

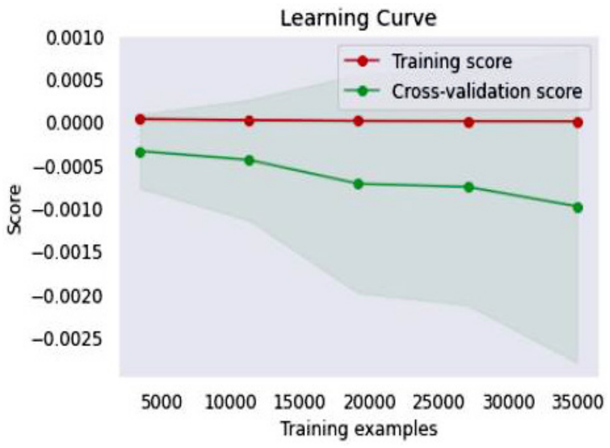


Fig. 11. Learning curve of LR

In Fig. 12., we use this curve to select the optimal number of clusters by fitting the model with a range of values for K in the K-means algorithm. We have shown a line drawn between SSE (Sum of Squared Errors) and the number of clusters to find the point representing the "elbow point" (this is a point after which the SSE or inertia starts to decrease linearly). The point on the SES / Inertia graph where the SES or inertia starts to decrease linearly is the elbow point. In this Fig. 12., it can be noted that it is at the number of clusters = 6 that the SSE starts to decrease linearly.

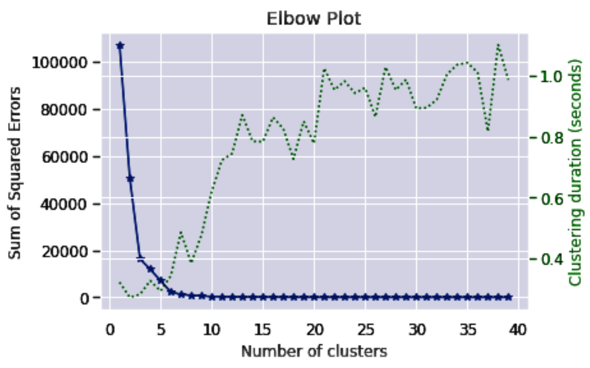


Fig. 12. Sum Squared Errors vs Number of clusters

In Fig. 13., we have AUC = 0.5, then the classifier is not able to distinguish between the Positive and Negative class points. This means that the classifier predicts a random class or a constant class for all data points.

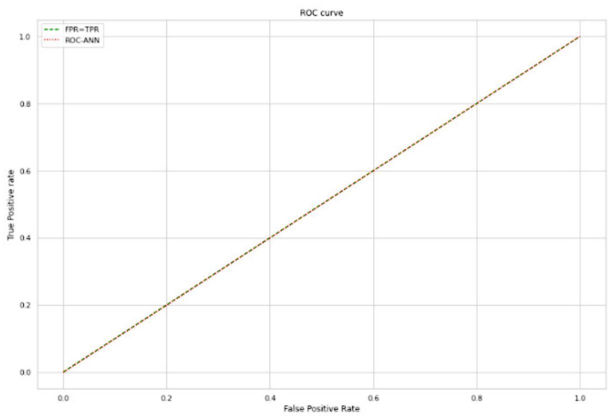


Fig. 13. ROC of ANN

Fig. 14., shows that the training score is still around the maximum score of 1.0 and the validation score could be increased with more training samples of the maximum score is 1.0.

In Fig. 15, we get the index of the first five products correlated with the product Id chosen by the user, then we see the ratings that the user could give to these 5 most correlated products by made.

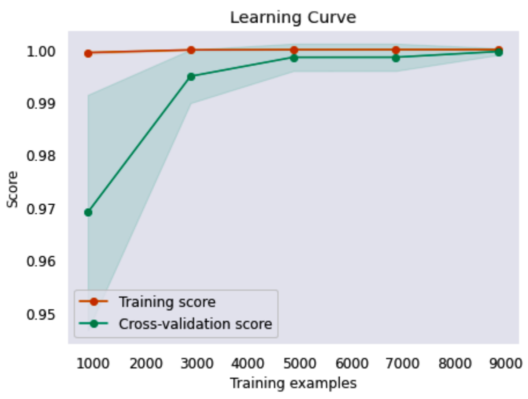
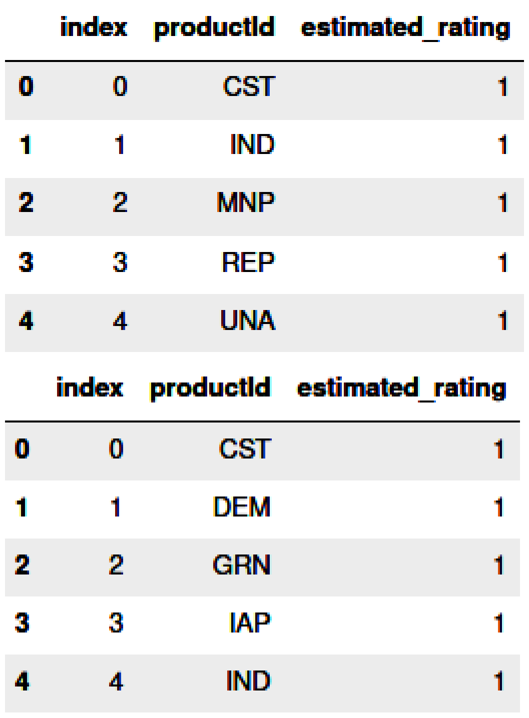


Fig. 14. Learning curve of Pearson



	index	productId	estimated_rating
0	0	CST	1
1	1	IND	1
2	2	MNP	1
3	3	REP	1
4	4	UNA	1

	index	productId	estimated_rating
0	0	CST	1
1	1	DEM	1
2	2	GRN	1
3	3	IAP	1
4	4	IND	1

Fig. 15. Results of the hybrid approach

6. Conclusion

The growth of IoT devices generating a large amount of data has made cloud computing an indispensable technology to facilitate data storage, processing, analysis and easy access. As the usage is diversified, it is very important that it is always in an optimal way. Several methods such as recommender systems can be used to improve its use. Recommender systems are known to provide relevant suggestions to a user based on his preferences or needs. However, with the rapid progress of technology. Therefore, current recommender systems rely on the Internet of Things (IoT) to achieve certain goals. In addition, there is too much information available on the Internet that needs to be analyzed or calculated for recommendation purposes. However, with the increase in the volume of data, the system performance gradually degrades. Therefore, a distributed environment such as the cloud is needed to perform and store all the computations on a single system. Based on this study, we observe that all the above areas are interdependent.

In this work, we have a comparative study on the performance of fourteen models applied to the election dataset among others: Rocchio, Latent Semantic Analysis, K-Nearest Neighbors, Naïve Bayes, Decision Tree Classifier, Support Vector Machine, Linear Regression, K-means, Slope One, Non-matrix Factorization, Singular Decomposition Vector, Co-clustering, Pearson Correlation and Artificial Neural Network. The evaluation of the algorithms is based on precision, recall, f1-score, accuracy, RMSE, MSE, MAE and area under the curve. The results, shows that K-NN and SGDC GridSearchCV perform best in terms of all evaluation parameters.

AUTHORS

Aristide Ndayikengurukiye* – Mohammed V University, Rabat, Morocco. e-mail: lemure.dede@gmail.com.

Abderrahmane Ez-Zahout – Mohammed V University, Rabat, Morocco. e-mail: abderrahmane.ezzahout@um5.ac.ma.

Akou Aboubakr – Mohammed V University, Rabat, Morocco. e-mail: aboubakar_aakaou@um5.ac.ma.

Youssef Charkaoui – Mohammed V University, Rabat, Morocco. e-mail: youssef_charkaoui@um5.ac.ma.

Omary Fouzia – Mohammed V University, Rabat, Morocco.

* Corresponding author

REFERENCES

[1] R. M. Frey, R. Xu and A. Ilic, "A Novel Recommender System in IoT". In: *2015 5th International Conference on the Internet of Things (IOT 2015)*, 2015, 10.3929/ETHZ-A-010561395.

[2] L. Mathiassen, "Collaborative practice research", *Information Technology & People*, vol. 15, no. 4, 2002, 321–345, 10.1108/09593840210453115.

[3] R. P. Adhikary, "Effectiveness of Content–based Instruction in Teaching Reading", *Theory and Practice in Language Studies*, vol. 10, no. 5, 2020, 10.17507/tpls.1005.04.

[4] S. Sharma, A. Sharma, Y. Sharma and M. Bhatia, "Recommender system using hybrid approach". In: *2016 International Conference on Computing, Communication and Automation (ICCCA)*, 2016, 219–223, 10.1109/CCAA.2016.7813722.

[5] H. J. Zimmermann, "Fuzzy set theory", *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 3, 2010, 317–332, 10.1002/wics.82.

[6] A. Gosain and S. Dahiya, "Performance Analysis of Various Fuzzy Clustering Algorithms: A Review", *Procedia Computer Science*, vol. 79, 2016, 100–111, 10.1016/j.procs.2016.03.014.

[7] T. Di Noia, R. Mirizzi, V. C. Ostuni and D. Romito, "Exploiting the web of data in model-based recommender systems". In: *Proceedings of the sixth ACM conference on Recommender systems - RecSys '12*, 2012, 253–256, 10.1145/2365952.2366007.

[8] A. A. Abdallat, A. I. Alahmad, D. A. A. Amimi and J. A. AlWidian, "Hadoop MapReduce Job Scheduling Algorithms Survey and Use Cases", *Modern Applied Science*, vol. 13, no. 7, 2019, 10.5539/mas.v13n7p38.

[9] M. Uta, A. Felfernig, V.-M. Le, A. Popescu, T. N. T. Tran and D. Helic, "Evaluating recommender systems in feature model configuration". In: *Proceedings of the 25th ACM International Systems and Software Product Line Conference – Volume A*, 2021, 58–63, 10.1145/3461001.3471144.

[10] N. Bhalse and R. Thakur, "Algorithm for movie recommendation system using collaborative filtering", *Materials Today: Proceedings*, 2021, 58–63, 10.1016/j.matpr.2021.01.235.

[11] K. Inagaki, S. Takaki, Y. Honda, M. Inoue, N. Mori, H. Kawakami, Y. Kawakami, M. Kawakami, H. Okamoto, K. Tuji, M. Tuge and K. Chayama, "A case of hepatitis E virus infection in a patient with primary biliary cholangitis", *Acta Hepatologica Japonica*, vol. 58, no. 3, 2017, 183–190, 10.2957/kanzo.58.183.

[12] M. Pazzani and D. Billsus, "Learning and Revising User Profiles: The Identification of Interesting Web Sites", *Machine Learning*, vol. 27, no. 3, 1997, 313–331, 10.1023/A:1007369909943.

[13] H. Christian, M. P. Agus and D. Suhartono, "Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TF-IDF)", *ComTech: Computer, Mathematics and Engineering Applications*, vol. 7, no. 4, 2016, 285–294, 10.21512/comtech.v7i4.3746.

[14] R. Madani, A. Ez-Zahout and A. Idrissi, "An Overview of Recommender Systems in the Context of Smart Cities". In: *2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech)*, 2020, 1–9, 10.1109/CloudTech49835.2020.9365877.

[15] V. Vaidhehi and R. Suchithra, "A Systematic Review of Recommender Systems in Education", *International Journal of Engineering & Technology*, vol. 7, no. 3.4, 2018, 188–191, 10.14419/ijet.v7i3.4.16771.

[16] Y. Ge, S. Zhao, H. Zhou, C. Pei, F. Sun, W. Ou and Y. Zhang, "Understanding Echo Chambers in E-commerce Recommender Systems". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, 2261–2270, 10.1145/3397271.3401431.

[17] A. Felfernig, V.-M. Le, A. Popescu, M. Uta, T. N. T. Tran and M. Atas, "An Overview of Recommender Systems and Machine Learning in Feature Modeling and Configuration". In: *15th International Working Conference on Variability Modelling of Software-Intensive Systems*, 2021, 1–8, 10.1145/3442391.3442408.

[18] X. Zheng, L. D. Xu and S. Chai, "QoS Recommendation in Cloud Services", *IEEE Access*, vol. 5, 2017, 5171–5177, 10.1109/ACCESS.2017.2695657.

[19] Y. R. Shrestha and Y. Yang, "Fairness in Algorithmic Decision-Making: Applications in Multi--Winner Voting, Machine Learning, and Recommender Systems", *Algorithms*, vol. 12, no. 9, 2019, 10.3390/a12090199.

[20] I. Palomares, C. Porcel, L. Pizzato, I. Guy and E. Herrera-Viedma, "Reciprocal Recommender Systems: Analysis of state-of-art literature, challenges and opportunities towards social recommendation", *Information Fusion*, vol. 69, 2021, 103–127, 10.1016/j.inffus.2020.12.001.

[21] D. S. Tarragó, C. Cornelis, R. Bello and F. Herrera, "A multi-instance learning wrapper based on the Rocchio classifier for web index recommendation", *Knowledge-Based Systems*, vol. 59, 2014, 173–181, 10.1016/j.knosys.2014.01.008.

[22] R. M. Suleman and I. Korkontzelos, "Extending latent semantic analysis to manage its syntactic blindness", *Expert Systems with Applications*, vol. 165, 2021, 10.1016/j.eswa.2020.114130.

[23] R. J. Kuo, C.-K. Chen and S.-H. Keng, "Application of hybrid metaheuristic with perturbation-based K-nearest neighbors algorithm and densest imputation to collaborative filtering in recommender systems", *Information Sciences*, vol. 575, 2021, 90–115, 10.1016/j.ins.2021.06.026.

[24] S. Renjith, A. Sreekumar and M. Jathavedan, "An extensive study on the evolution of context-aware personalized travel recommender systems", *Information Processing & Management*, vol. 57, no. 1, 2020, 10.1016/j.ipm.2019.102078.

[25] R. Panigrahi and S. Borah, "Rank Allocation to J48 Group of Decision Tree Classifiers using Binary and Multiclass Intrusion Detection Datasets", *Procedia Computer Science*, vol. 132, 2018, 323–332, 10.1016/j.procs.2018.05.186.

[26] J. Hamidzadeh, E. Rezaeenik and M. Moradi, "Predicting users' preferences by Fuzzy Rough Set Quarter-Sphere Support Vector Machine", *Applied Soft Computing*, vol. 112, 2021, 10.1016/j.asoc.2021.107740.

[27] K. Luo, S. Sanner, G. Wu, H. Li and H. Yang, "Latent Linear Critiquing for Conversational Recommender Systems". In: *Proceedings of The Web Conference 2020*, 2020, 2535–2541, 10.1145/3366423.3380003.

[28] A. Yassine, L. Mohamed and M. Al Achhab, "Intelligent recommender system based on unsupervised machine learning and demogra-

phic attributes", *Simulation Modelling Practice and Theory*, vol. 107, 2021, 10.1016/j.simpat.2020.102198.

[29] Q.-X. Wang, X. Luo, Y. Li, X.-Y. Shi, L. Gu and M.-S. Shang, "Incremental Slope-one recommenders", *Neurocomputing*, vol. 272, 2018, 606–618, 10.1016/j.neucom.2017.07.033.

[30] T. V. R. Himabindu, V. Padmanabhan and A. K. Pujari, "Conformal matrix factorization based recommender system", *Information Sciences*, vol. 467, 2018, 685–707, 10.1016/j.ins.2018.04.004.

[31] A. L. Vizine Pereira and E. R. Hruschka, "Simultaneous co-clustering and learning to address the cold start problem in recommender systems", *Knowledge-Based Systems*, vol. 82, 2015, 11–19, 10.1016/j.knosys.2015.02.016.

[32] F. Maazouzi, H. Zarzour and Y. Jararweh, "An Effective Recommender System Based on Clustering Technique for TED Talks", *International Journal of Information Technology and Web Engineering (IJITWE)*, vol. 15, no. 1, 2020, 35–51, 10.4018/IJITWE.2020010103.

[33] C. Djellali and M. Adda, "A New Hybrid Deep Learning Model based-Recommender System using Artificial Neural Network and Hidden Markov Model", *Procedia Computer Science*, vol. 175, 2020, 214–220, 10.1016/j.procs.2020.07.032.