

# An model for structured the NoSQL databases based on machine learning classifiers

**Amine Benmakhlouf**

Computer, Networks, Mobility and Modeling Laboratory (IR2M), Department of Mathematics and Computer Science,  
Faculty of Science and Technology, University Hassan 1<sup>st</sup>, Settat, Morocco

## Article Info

### Article history:

Received Aug 28, 2024

Revised Oct 6, 2024

Accepted Nov 19, 2024

### Keywords:

Deep learning

Documents oriented database

Gradient

Machine learning

Neural networks

NoSQL

OLAP

## ABSTRACT

Today, the majority of data generated and processed in organizations is unstructured. NoSQL database management systems perform the management of this data. The problem is that these unstructured databases cannot be analyzed by traditional OLAP analytical treatments. The latter are mainly used in structured relational databases. In order to apply OLAP analyses on NoSQL data, the structuring of this data is essential. In this paper, we propose a model for structuring the data of a document-oriented NoSQL database using machine learning (ML). This method is broken down into three steps, first the vectorization of documents, then the learning via different ML algorithms and finally the classification, which guarantees that documents with the same structure will belong to the same collection. Therefore, the modeling of a data warehouse can be carried out in order to create OLAP cubes. Since the models found by learning allow the parallel computation of the classifier, our approach represents an advantage in terms of speed since we will avoid doubly iterative algorithms, which rely on textual comparisons (TC). A comparative study of the performances is carried out in this work in order to detect the most efficient methods to perform this type of classification.

*This is an open access article under the [CC BY-SA](#) license.*



## Corresponding Author:

Amine Benmakhlouf

Computer, Networks, Mobility and Modeling Laboratory (IR2M)

Department of Mathematics and Computer Science, Faculty of Science and Technology

Hassan 1<sup>st</sup> University

Settat, Morocco

Email: Amine.benmakhlouf@uhp.ac.ma

## 1. INTRODUCTION

Big data is characterized by the 3Vs [1], which are volume, velocity and variety. So it's a large amount of data arriving at a higher speed and with a lot of variety. This last characteristic means that the data collected in big data comes from several sources and is not necessarily structured. These large and varied datasets cannot be managed by traditional relational database management systems [2]. NoSQL represents an interesting alternative. These are non-relational database management systems (DBMSs) capable of handling a large amount of unstructured data with greater flexibility and scalability. The atomicity, coherence, isolation, and durability (ACID) principles are ensured in relational databases. These features ensure that information remains consistent during a transaction. The latter represents an isolated unit which is not affected by another transaction, it remains permanently in the system after validation. NoSQL databases also guarantee the distributiveness, flexibility (DF) principles. Distributing data across multiple servers makes it more accessible and increases the system's ability to perform well with larger workloads. It is for these reasons that NoSQL systems are solutions for storing and managing structured, semi-structured and

unstructured data. Despite the advantages of NoSQL databases, using online analytical processing (OLAP) multidimensional data analysis techniques is difficult. OLAP analysis is originally performed on relational databases. It can also be applied to NoSQL databases but with structured data [3]-[5]. Therefore, appropriate methods for analyzing unstructured data in NoSQL systems need to be developed. The most commonly used type of NoSQL database is the document-oriented database. In this type of database, data is stored in collections. Each collection is a set of documents and each document is a set of pairs (key, value), the keys are none other than the attributes. Documents in the same collection can have different attributes, hence the unstructured nature of a document-oriented NoSQL database. It is in this context that we propose, in this work, an approach to structure the data of a document-oriented NoSQL database and thus be able to extract OLAP cubes. Machine learning (ML) algorithms are used to obtain document classification models in different collections. Our method consists of three phases which are vectorization of documents, learning then prediction. This last phase will allow us to classify the documents into several collections. A parallel calculation can be carried out in this case since the models found, after training, can be applied to each document in the main collection. The training corpus will be composed of the names of the document attributes. ML is a branch of artificial intelligence that involves using algorithms to analyze data sets and identify trends or patterns. These models are then used to make predictions on new data. The algorithms used in this work are the classifiers: logistic regressions (LRs), Naives Bayes (NB), K-nearest neighbours (KNNs), multi layer perceptron (MLP), decision tree (DT), and support vector machines (SVMs). In order to evaluate the performance and effectiveness of these ML models in this type of multiclass classification, a study of metrics is carried out in this work.

## 2. MULTIDIMENSIONAL CONCEPTUAL MODEL

The multidimensional model includes fact tables associated with dimension tables [6]-[9]. The corresponding diagram E is given by:  $E = (F^E, D^E, start^E)$  with:

$F^E = \{F_1, F_2, \dots, F_n\}$  is a finite set of facts tables.

$D^E = \{D_1, D_2, \dots, D_m\}$  is a finite set of dimension tables

$start^E = F^E \rightarrow 2^{D^E}$  is a function that associates the facts  $F^E$  with sets of dimensions along which they can be analysed ( $2^{D^E}$  is the set of powers of  $D^E$ )

Each dimension  $D_i \in D^E$  is defined by  $(N^D, A^D, H^D)$  with:

$N^D$ : is the dimension name.

$A^D = \{a_1^D, a_2^D, \dots, a_p^D\}$  is the attributes set existing in the dimensions. There are simple and complex attributes composed of several attributes.

$H^D = \{h_1^D, h_2^D, \dots, h_s^D\}$  is the hierarchies set.

Each facte  $F \in F^E$  is defined by  $(N^F, M^F)$  with:

$N^F$  is the name of the table made.

$M^F = \{m_1^F, m_2^F, \dots, m_{|M^F|}^F\}$  is the measurement set. Aggregate functions are applied to the measurements.

A combination of dimensions represents the axes of analysis, while measures and their aggregations represent the analysis values.

## 3. DOCUMENT-ORIENTED NO-RELATIONAL LOGICAL MODEL

OLAP analysis on document-oriented NoSQL data warehouses has been the subject of several studies. All these studies were carried out on a set of structured data in NoSQL databases. For example the authors [10]-[13] worked on setting up a data warehouse with a document-oriented NoSQL system. They propose four document-oriented logic model approaches.

- In the first model, called flat denormalised (FM), all dimension attributes and all measurements are combined in a single document.

$$C^F = \{id_F, m_1^F, m_2^F, \dots, m_{|M^F|}^F, a_1^{D_1}, a_2^{D_1}, \dots, a_{|A^{D_1}|}^{D_1}, a_1^{D_2}, a_2^{D_2}, \dots, a_{|A^{D_2}|}^{D_2}\}$$

- In the second model, called nested denormalised (NM), the attribute values of the fact and dimension tables are stored in a single collection. In each document, the measurements from the fact table are grouped in a sub-document with the key NF. The attributes of each dimension  $D_i$  are also grouped in a sub-document identified by the key  $N^{D_i}$ . The model schema is defined by:

$$C^F = \left\{ \begin{array}{l} id_F, N^F: \{m_1^F, m_2^F, \dots, m_{|M^F|}^F\}, \\ N^{D_1}: \{a_1^{D_1}, a_2^{D_1}, \dots, a_{|A^{D_1}|}^{D_1}\}, \\ N^{D_2}: \{a_1^{D_2}, a_2^{D_2}, \dots, a_{|A^{D_2}|}^{D_2}\}, \dots \end{array} \right\}$$

- In the third model, called normalised split (SM), the data from the fact and dimension tables are stored in separate collections in order to remove redundancies. The fact F is stored in a collection CF and each dimension Di is stored in a collection CDi. The fact document contains foreign keys whose values come from the primary keys of the dimension documents. The model schema is defined by:

$$\begin{aligned} C^F &= \{id_F, m_1^F, m_2^F, \dots, m_{|M^F|}^F, id_{D_1}, id_{D_2}, \dots\} \\ C^{D_1} &= \{id_{D_1}, a_1^{D_1}, a_2^{D_1}, \dots, a_{|A^{D_1}|}^{D_1}\} \\ C^{D_2} &= \{id_{D_2}, a_1^{D_2}, a_2^{D_2}, \dots, a_{|A^{D_2}|}^{D_2}\} \end{aligned}$$

- In the fourth model, called hybrid (HM), the characteristics of the SM and NM models are combined. All the attributes of the fact and dimension tables are stored in a single collection, but keeping the same schema of the SN model. In each document of the CF collection, we store the attribute values of the fact table as well as the foreign keys whose values come from the primary keys of the dimension tables. These are stored in nested subdocuments in CF. the diagram is given by:

$$C^F = \left\{ \begin{array}{l} id_F, m_1^F, m_2^F, \dots, m_{|M^F|}^F, id_{D_1}, id_{D_2}, \dots, \\ N^{D_1}: \{id_{D_1}, a_1^{D_1}, a_2^{D_1}, \dots, a_{|A^{D_1}|}^{D_1}\}, \\ N^{D_2}: \{id_{D_2}, a_1^{D_2}, a_2^{D_2}, \dots, a_{|A^{D_2}|}^{D_2}\} \end{array} \right\}$$

These multidimensional logical models can only be obtained from a structured NoSQL database. i.e., in the case of document-oriented databases where all records have the same attributes. On the other hand, appropriate methods for analysing unstructured data in NoSQL systems must be developed. The main objective of our work is to propose an efficient approach for structuring the data in a NoSQL database so that it is suitable for multidimensional modelling.

#### 4. PROPOSED MODEL

In this article, we propose a model called “MLDS” (machine learning for data structuring) capable of structuring the data of a document-oriented NoSQL database. Since our problem is a supervised multiclass classification, we will apply ML and deep learning methods in order to study their performance in this type of problem. In the literature, several works have been carried out to classify documents in a document-oriented database. Amazal *et al.* [14] used the “Naïf Bayes” classification method. It is a supervised learning algorithm based on Bayes’ theorem. It is often used for classification of text and categorical data. This is a simple and fast algorithm, but it assumes conditional independence between features, which is not always realistic in practice. Davardoost *et al.* [15] combined this algorithm with the map-reduce programming method in order to adapt it to large amounts of data. The classification methods: DT, SVM, KNN, NB, MLP, and LR are used in our model. The main objective is to compare the performance of these methods in the classification of unstructured and complex data. The structure of our method will be described in this section. Figure 1 represents the different steps of the “MLDS” method to prepare data for OLAP processing.

The first phase is a training data preparation phase. NoSQL database needs to be converted to matrix. The documents from the document-oriented database are used as input data to the vectorization algorithm. A collection of a document-oriented database is composed of n documents. And each document is composed of a set of attributes. Since the data in a NoSQL database is not structured, the number of attributes differs from one document to another. Some attributes are present in documents while others are absent. The vectorization algorithm gives as output a binary matrix D composed of 0 and 1. The rows of this matrix represent the n documents in the collection, the columns represent the set of attributes of all the documents and the elements of the matrix are 0 or 1. 0 means the attribute is absent in the document and 1 means the attribute is present. A duplicate removal process is carried out on the matrix D in order to keep only unique vectors. The result will therefore be the training data for the deep neural network.

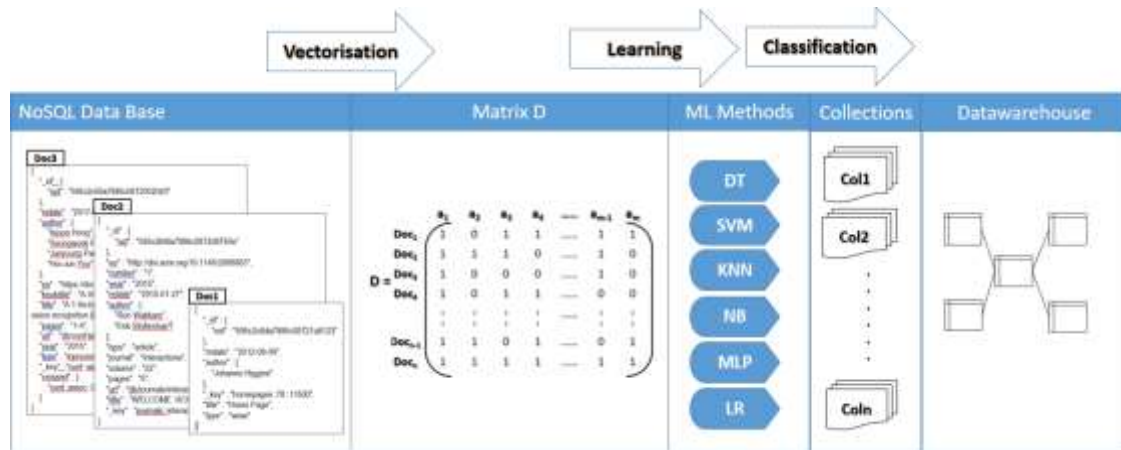


Figure 1. The phases of the MLDS model

The second phase is a learning phase. The vectors (rows of matrix D) are subjected to the different ML algorithms applied in the multiple class classifiers. Multi-class classification algorithms are used to classify data into more collections. Several methods have been developed based on neural networks, DTs, KNN, NB, SVMs, and LR to solve multi-class classification problems. To begin, we remove the duplicates from the matrix D. The set of unique vectors, thus obtained, will represent the training data. On the other hand, all of the vectors of the matrix D will represent the test data. The prediction step consists of using the model found by each learning method in order to determine which class the test data belongs to. Documents similar in structure are mapped to the same classes thus allowing extraction of OLAP cubes based on classification.

The third phase is the classification or prediction phase. The patterns found during the learning phase will be used to determine in which collection a document will be classified. Our method allows parallel classification of documents, which will significantly reduce the time required to classify all documents in the database. A comparative study of the classification processing time based on the learning models will be carried out in the experimentation part. Documents with a similar structure will be mapped to the same collection. Therefore, OLAP cubes can be extracted based on the classification.

To provide a visual overview of the proposed system, Figure 2 illustrates the architecture and workflow. The data preparation phase results in a training dataset composed by unique vectors after removing duplicate vectors. The model found after the learning phase is used on the entire matrix D to classify the documents.

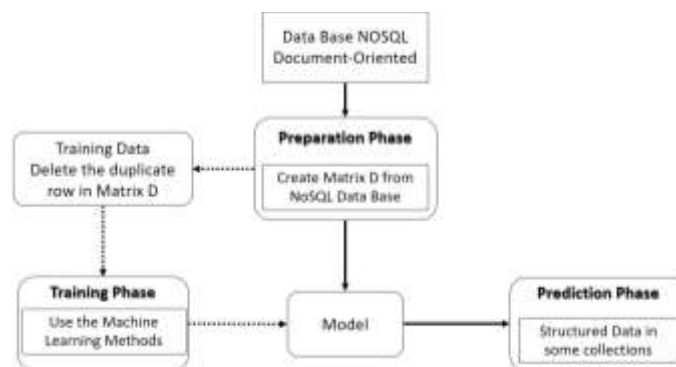


Figure 2. Flow diagram of the proposed solution

## 5. METHODS MACHINE LEARNING USED

### 5.1. Multinomial logistic regressions

LR is a supervised ML algorithm used to predict the probability of whether or not an instance belongs to a given class. This algorithm [16] uses a functional approach to find the binary response

probability based on a number of characteristics. The SoftMax function (1) can be used in multi-class classification problems where the goal is to predict a single label from multiple classes.

$$Softmax(z_i) = \frac{e^{(xw_i)}}{\sum_{j=0}^k e^{(xw_j)}}, z_i = xw_i \quad (1)$$

### 5.2. Naives Bayes classificatory

The NB algorithm [17] can be used for multi class classification with more two class. To classify a sample, we calculate the probability of each class and select the class with the highest probability. The NB classifier assumes that all input data features are independent, which is not true in reality. However, despite this simplifying hypothesis, this algorithm remains effective and performs well in many applications. Bayes' theorem allows us to calculate probability of each document belongs to each class  $P(C1/d)$ ,  $P(C2/d)$ , ...,  $P(Cn/d)$ . If  $P(Ck/d) = \max(P(C1/d), P(C2/d), \dots, P(Cn/d))$  then the class of the document  $d$  is  $Ck$ . Considering that the attributes are independent, the probability documents  $P(Ci|d)$  based on NB theory can be calculated according to (2).

$$P(C_i/d) = \frac{P(d/C_i)P(C_i)}{P(d)} \quad (2)$$

Avec:

- $P(Ci|d)$  is the posterior probability of the class ( $C_i$ , target) given the predictor ( $d$ , attributes).
- $P(C_i)$  is the prior probability of the class.
- $P(d|C_i)$  is the likelihood, which is the probability of the predictor given the class.
- $P(d)$  is the prior probability of the predictor.

If we assume that  $A = \{a_1, a_2, \dots, a_m\}$  is the attributes set of the document  $d$ , we are:

$$P(d/C_i)P(C_i) = P(C_i) \prod_{j=1}^m P(a_j/C_i) \quad (3)$$

### 5.3. K-nearest neighbours

KNN [18] is a ML algorithm that can be used for multi-class classification. In the context of multiclass classification, KNN aims to classify a data point based on the most frequent class among its KNN. The Euclidean distance ( $D_{ij}$ ) between two input vectors ( $V_i, V_j$ ) is given as:

$$D_{ij} = \sqrt{\sum_{k=1}^n (V_{ik} - V_{jk})^2} \quad (4)$$

This distance between the current entry and another data point is calculated for each data point in the dataset. The  $k$  elements are selected among those with the lowest distance. The classifier returns the majority class among these  $k$  data points as the classification for the entry point.

### 5.4. Multi layer perceptron

Deep learning is an advanced form of ML that uses neural networks to mimic the functioning of the human brain. The MLP [19] is a type of artificial neural network organized into several layers. A MLP has at least three layers: an input layer, at least one hidden layer, and an output layer. The technique called for in these neural networks is gradient backpropagation. During this propagation, input data is passed through the network layer by layer, with each layer performing a calculation based on the inputs it receives and passing the result to the next layer. Backpropagation is an algorithm used to train neural networks by adjusting the weights and biases of the network to minimize the loss function. Below is a mathematical explanation of this backpropagation method. In an artificial neural network, if the  $S_j$  are the inputs of the neuron  $n_i$  then the output is given by:

$$S_i = f(e_i) \text{ with } e_i = \sum_j W_{ij} S_j \quad (5)$$

$f$  is the activation function and  $W_{ij}$  are the synapse activation coefficients as shown in Figure 3. This method consists of calculating the gradient of the error in each neuron of the network. These errors will be corrected via back propagation of the gradient. This principle is effectively used in multilayer neural networks [20]-[23].



Figure 3. Neural network forward pass

Let  $S_i$  be the output obtained from the  $i$ th neuron of the output layer, and  $t_i$  the desired output. The squared error on the output neurons is given by:

$$E = \sum_i (t_i - S_i)^2 \quad (6)$$

The gradient method consists of evaluating the  $j$  activation coefficients of the  $i$ th neuron  $w_{ij}$  in the opposite direction of the gradient. According to (5), the evolution of the weight  $w_{ij}$  is:

$$\Delta W_{ij} = -\mu \frac{\partial E}{\partial W_{ij}} = -\mu \frac{\partial E}{\partial S_i} \frac{\partial S_i}{\partial e_i} \frac{\partial e_i}{\partial W_{ij}} \quad (7)$$

$0 \leq \mu \leq 1$ : learning constant

$$\frac{\partial e_i}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}} \sum_k W_{ik} S_k \quad (8)$$

since  $\frac{\partial W_{ij}}{\partial W_{ik}} = 0$  si  $k \neq j$ , we find:

$$\frac{\partial e_i}{\partial W_{ij}} = S_j \quad (9)$$

so, the evolution of the activation coefficient becomes:

$$\Delta W_{ij} = -\mu \delta_i S_j \text{ With: } \delta_i = \frac{\partial E}{\partial S_i} \frac{\partial S_i}{\partial e_i} \quad (10)$$

for the output layer the local gradient is given by:

$$\delta_i = f'(e_i)(t_i - S_i) \quad (11)$$

for hidden layers, a layer  $i$  influences the states of all the cells of the next layer  $k$ , the local gradient is given by:

$$\delta_i = f'(e_i) \sum_k \delta_k w_{ki} \quad (12)$$

we therefore obtain a recurring method for calculating the error signals of the cells of a layer from those of the following layer as shown in Figure 4.

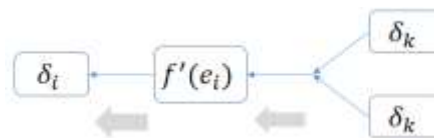


Figure 4. Neural network back pass

### 5.5. Decision tree

DT [24] is a supervised ML method that can be used for classification and regression problems. It is especially preferred for solving classification problems. It is a tree-structured classifier, where internal nodes

represent features of a dataset, branches represent decision rules, and each leaf node represents the result. A DT is made up of two nodes: the decision node and the leaf node. Decision nodes allow a decision to be made and have several branches, while leaf nodes give the result of these decisions and do not contain branches.

### 5.6. Support vector machines

SVM [25] are a set of supervised learning methods used for classification, regression, and outlier detection. The principle of SVM consists of carrying out classifications using hyperplanes (feature space). The latter make it possible to separate the data into several classes by specifying the boundary furthest possible from the data points (or maximum margin).

## 6. EXPERIMENTAL RESULTS

### 6.1. Comparison of metrics classification

The tests are carried out in a NoSQL mongodb database. We compare the performance of different ML methods. The metrics used in this comparison are: the metrics of each class (precision, recall, F1-score and support) and the macro metrics (macro precision, macro recall and macro F1-score). The classic “macro” metric represents the average of “per class” metrics. In mathematical terms, these metrics are given by the expression (13). Another important global metric is studied which is accuracy.

$$Macro\_Metric = \frac{1}{n_{class}} \sum_{i=1}^{n_{class}} Metric_{Class_i} \quad (13)$$

Accuracy measures the proportion of correctly classified cases out of the total number of objects in the dataset.

$$Accuracy = \frac{Correct\ Predictions}{All\ Predictions} \quad (14)$$

Precision for a given class in multiclass classification is the fraction of instances correctly classified as belonging to a specific class out of all instances that the model predicts to belong to that class.

$$Precision_{ClassA} = \frac{TP_{ClassA}}{TP_{ClassA} + FP_{ClassA}} \quad (15)$$

Recall in multiclass classification is the fraction of instances of a class that the model correctly classified among all instances of that class.

$$Recall_{ClassA} = \frac{TP_{ClassA}}{TP_{ClassA} + FN_{ClassA}} \quad (16)$$

F1-score takes into account both precision and recall measures by calculating their harmonic average. If we denote by P the precision and R the recall, we can represent the F1-score as follows:

$$F1\_score = \frac{2RP}{R+P} \quad (17)$$

support represents the number of actual occurrences of each class in the dataset. This is the number of instances in each class.

The dataset used is the computer bibliographic database known as DBLP. It is a database lists conference and journal articles. DBLP database JSON file contains unstructured information about publications, authors, and conferences. The documents forming this database do not have the same attributes. The number of documents used in the database is used as an evaluation metric. In this comparison, we were interested in NoSQL databases containing between 10,000 and 90,000 and between 100,000 and 600,000 unstructured documents. In this paper, we are content to represent the metrics of databases containing 10,000, 100,000 and 600,000. The results of these metrics are reported in Tables 1 to 3.

We note from this study that for the KNN, LR, NB, SVM learning models, all metrics are 100% regardless of the number of documents to be structured. On the other hand, for the MLP and DT models, the quality of the metrics decreases slightly from a quantity of documents greater than or equal to 30,000. For example, we have a precision of 0.69 for the MLP and DT, an average precision of 0.96 and 0.95 respectively for the MLP and DT and an F1-score of 0.95 for the MLP and DT.

Table 1. Up: the metrics report of the 83 classes of the classification database with 10,000 unstructured documents. Down: macro metrics results

Class	Precision						Recall						F1-score						Support					
	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM
1	0.00	0.00	1.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3047	3047	3047	3047	3047	3047
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3266	3266	3266	3266	3266	3266
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2286	2286	2286	2286	2286	2286
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	222	222	222	222	222	222
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	227	227	227	227	227	227
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	14	14	14	14	14	14
7	0.05	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.10	0.97	1.00	1.00	1.00	1.00	167	167	167	167	167	167
8	0.96	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.12	1.00	1.00	1.00	1.00	229	229	229	229	229	229
9	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	61	61	61	61	61	61
10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3	3	3	3	3	3
11	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	9	9	9	9	9	9
12	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	7	7	7	7	7	7
13	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3	3	3	3	3	3
14	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	14	14	14	14	14	14
15	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	42	42	42	42	42	42
.....																								
142	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1
143	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1
Classifier							MLP		DT		KNN		LR		NB		SVM							
Accuracy							100%		100%		100%		100%		100%		100%							
Macro precision							100%		100%		100%		100%		100%		100%							
Macro recall							100%		100%		100%		100%		100%		100%							
Macro F1-score							100%		100%		100%		100%		100%		100%							

Table 2. Up: the metrics report of the 143 classes of the classification database with 100,000 unstructured documents. Down: macro metrics results

Class	Precision						Recall						F1-score						Support					
	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM
1	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	1.00	1.00	1.00	30704	30704	30704	30704	30704	30704
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	32803	32803	32803	32803	32803	32803
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	22552	22552	22552	22552	22552	22552
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2204	2204	2204	2204	2204	2204
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2390	2390	2390	2390	2390	2390
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	108	108	108	108	108	108
7	0.05	0.95	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.10	0.97	1.00	1.00	1.00	1.00	1685	1685	1685	1685	1685	1685
8	0.96	0.06	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.12	1.00	1.00	1.00	1.00	2040	2040	2040	2040	2040	2040
9	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	631	631	631	631	631	631
10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	23	23	23	23	23	23
11	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	73	73	73	73	73	73
12	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	44	44	44	44	44	44
13	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	26	26	26	26	26	26
14	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	158	158	158	158	158	158
15	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	453	453	453	453	453	453
.....																								
142	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1
143	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1
Classifier							MLP		DT		KNN		LR		NB		SVM							
Accuracy							69%		69%		100%		100%		100%		100%							
Macro precision							95%		94%		100%		100%		100%		100%							
Macro recall							96%		96%		100%		100%		100%		100%							
Macro F1-score							95%		95%		100%		100%		100%		100%							

## 6.2. Structuring data

In Figure 5 we represent the number of collections generated by structuring data from the unstructured DBLP database. This study is carried out for different numbers of documents in the DBLP database and using the structuring ML models. all these models generate exactly the same document collections. These are well structured with the same attributes. We also see that the number of collections generated increases with the size of the unstructured database. But this increase will also depend on the degree of structuring of the documents which constitute the unstructured data base.

## 6.3. Comparison of structuring times

Another performance component studied in this work is the time required to structure NoSQL data by applying the prediction models found by the ML algorithms used. The results of this comparative study



are reported in the histograms of Figure 6. We can conclude that the ML methods: MPL, DL, LR, and SVM allow faster structuring of data compared to NB and KNN methods. For example for a database of 600,000 documents, the fastest structuring is carried out by the LR method with a time of 915.42 (s). On the other hand, the NB method records the longest time with 4096.81(s). This comparison shows that the LR method allows faster structuring, especially for large quantities of data.

Table 3. Up: the metrics report of the 239 classes of the classification database with 600,000 unstructured documents. Down: macro metrics results

Class	Precision					Recall					F1-score					Support																									
	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM	MLP	DT	KNN	LR	NB	SVM																	
1	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	1.00	1.00	1.00	184809	184809	184809	184809	184809	184809																	
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	196433	196433	196433	196433	196433	196433																	
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	134766	134766	134766	134766	134766	134766																	
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	13373	13373	13373	13373	13373	13373																	
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	14401	14401	14401	14401	14401	14401																	
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	626	626	626	626	626	626																	
7	0.05	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.10	0.97	1.00	1.00	1.00	1.00	9721	9721	9721	9721	9721	9721																	
8	0.96	0.91	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.12	1.00	1.00	1.00	1.00	12686	12686	12686	12686	12686	12686																	
9	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3560	3560	3560	3560	3560	3560																	
10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	144	144	144	144	144	144																	
11	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	379	379	379	379	379	379																	
12	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	214	214	214	214	214	214																	
13	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	192	192	192	192	192	192																	
14	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	967	967	967	967	967	967																	
15	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2690	2690	2690	2690	2690	2690																	
.....																																									
142	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1																	
143	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1	1	1	1	1	1																	
	Classifier						MLP						DT						KNN						LR						NB						SVM				
	Accuracy						69%						69%						100%						100%						100%						100%				
	Macro precision						97%						96%						100%						100%						100%						100%				
	Macro recall						97%						97%						100%						100%						100%						100%				
	Macro F1-score						97%						97%						100%						100%						100%						100%				

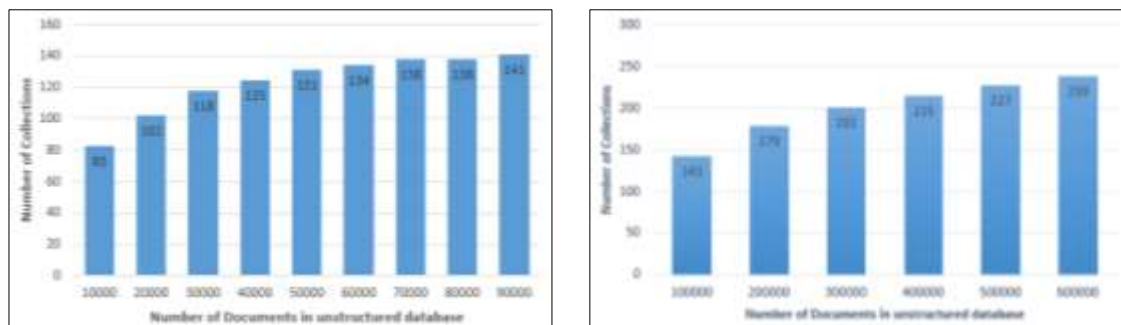


Figure 5. Collection number generated by data structuring using MLDS methods for different number of documents in the unstructured database

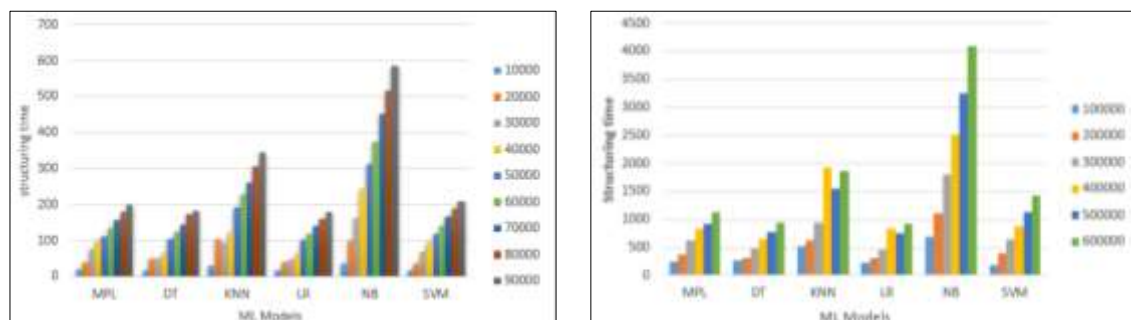


Figure 6. Structuring time using different ML models for different number of documents

In the previous study, the LR method showed good performance compared to other ML methods used. We will therefore study the advantage that the LR method can present in the structuring of NoSQL data compared to the classic method called textual comparison (TC). This method consists of using a doubly iterative algorithm to make TC of the attributes that compose each document of the unstructured database. On the other hand, in the prediction algorithms used on the models found by ML, we can exploit the advantages of parallel computing to structure the data. Figure 7 shows the results of the comparison of the structuring time of the NoSQL data of the two methods: the TC method TC and the ML method LR. This comparison is made for different quantities of unstructured data. We can see that for small quantities of data, the structuring time remains relatively low with the TC method. Example: we need 14.12 seconds to structure 10,000 documents with the TC method, while we need 20.60 seconds with the LR method. On the other hand, for large quantities of data, we see that the structuring time increases exponentially for the TC method, while the LR method shows lower execution times. Example: we need 1,498.65 seconds to structure 600,000 documents with the TC method, while we only need 915.42 seconds with the LR method. We can therefore conclude that the LR method shows these advantages for large quantities of NoSQL data.

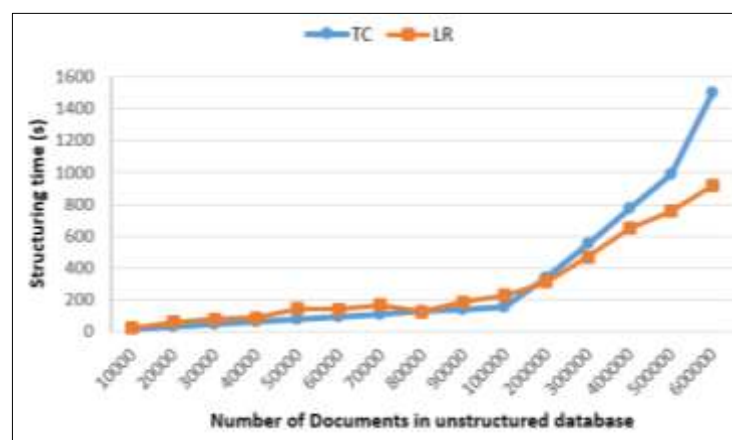


Figure 7. Structuring time of TC and LR methods for different amounts of data

## 7. CONCLUSION

Since big data is mainly made up of a large mass of unstructured data, their exploitation requires OLAP analytical calculations. This is why the development of innovative methods to structure this data has become a necessity. The model presented in this article exploits the advantages offered by ML methods. The application of these methods to datasets composed of attributes allowed us to generate classification models. These gave us the possibility of structuring data in a document-oriented NOSQL database. Each document in the NoSQL database having the same attributes will be stored in the same collection, allowing the use of OLAP cubes for data analysis. The model proposed by the logistic regression method shows high performance compared to other ML methods. It allows faster data structuring even for a large number of documents. Experience has also shown that, in the case of large data masses, the model found by the LR method allows faster data structuring compared to the classic TC method. As part of future work, we plan to introduce into our model the notion of data distributivity across multiple nodes of a cluster. This technique will allow more efficient management of a large quantity and high availability of information through distributed computing across multiple servers.





## REFERENCES

- [1] R. Lu, H. Zhu, X. Liu, J. Liu, and J. Shao, "Toward efficient and privacy-preserving computing in big data era," *IEEE Network*, vol. 28, no. 4, pp. 46–50, 2014, doi: 10.1109/MNET.2014.6863131.
- [2] R. Bruchez, *Les bases de données NoSQL et le big data (in France: NoSQL database and the big data)*, 2th ed. Paris, France: Eyrolles, 2015.
- [3] A. Sellami, A. Nabli, and F. Gargouri, "Graph NoSQL data warehouse creation," in *Proceedings of the 22nd International Conference on Information Integration and Web-based Applications & Services*, Nov. 2020, pp. 34–38, doi: 10.1145/3428757.3429141.
- [4] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "How can we implement a multidimensional data warehouse using NoSQL?," in *Enterprise Information Systems: 17th International Conference, ICEIS*, 2015, vol. 241, pp. 108–130, doi: 10.1007/978-3-319-29133-8\_6.

- [5] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "Implementation of multidimensional databases in column-oriented NoSQL systems," in *Advances in Databases and Information Systems: 19<sup>th</sup> East European Conference, ADBIS*, 2015, pp. 79–91, doi: 10.1007/978-3-319-23135-8\_6.
- [6] A. Cuzzocrea, R. Moussa, and G. Xu, "OLAP\*: effectively and efficiently supporting parallel OLAP over big data," in *Model and Data Engineering: Third International Conference, MEDI*, 2013, pp. 38–49, doi: 10.1007/978-3-642-41366-7\_4.
- [7] A. Cuzzocrea, L. Bellatreche, and I.-Y. Song, "Data warehousing and OLAP over big data," in *Proceedings of the sixteenth international workshop on Data warehousing and OLAP*, Oct. 2013, pp. 67–70, doi: 10.1145/2513190.2517828.
- [8] F. Ghozzi, F. Ravat, O. Teste, and G. Zurfluh, "Constraints for multidimensional models and languages (in France: Contraintes pour modèle et langage multidimensionnels)," *Ingénierie des systèmes d'information*, vol. 9, no. 1, pp. 9–34, Feb. 2004, doi: 10.3166/isi.9.1.9-34.
- [9] A. Hassan, F. Ravat, O. Teste, R. Tournier, and G. Zurfluh, "Differentiated multiple aggregations in multidimensional databases," in *Data Warehousing and Knowledge Discovery: 14th International Conference, DaWaK*, 2012, pp. 93–104, doi: 10.1007/978-3-642-32584-7\_8.
- [10] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "Implementing multidimensional data warehouses into NoSQL," in *Proceedings of the 17th International Conference on Enterprise Information Systems*, 2015, pp. 172–183, doi: 10.5220/0005379801720183.
- [11] M. Chavalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "Document-oriented data warehouses: models and extended cuboids, extended cuboids in oriented document," in *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, Jun. 2016, pp. 1–11, doi: 10.1109/RCIS.2016.7549351.
- [12] A. Benmakhlouf, "NoSQL Implementation of a conceptual data model: UML class diagram to a document oriented model," *International Journal of Database Management Systems*, vol. 10, no. 2, pp. 1–10, Apr. 2018, doi: 10.5121/ijdms.2018.10201.
- [13] Y. Hiyane, A. Benmakhlouf, and A. Marzouk, "Storing data in a document-oriented database and implemented from a structured nesting logical model," *International Journal of Database Management Systems*, vol. 12, no. 2, pp. 17–23, Apr. 2020, doi: 10.5121/ijdms.2020.12202.
- [14] H. Amazal, M. Ramdani, and M. Kissi, "A text classification approach using parallel Naive Bayes in big data context," in *Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications*, Oct. 2018, pp. 1–6, doi: 10.1145/3289402.3289536.
- [15] F. Davardoost, A. Babazadeh Sangar, and K. Majidzadeh, "An innovative model for extracting OLAP cubes from NOSQL database based on scalable Naïve Bayes classifier," *Mathematical Problems in Engineering*, vol. 2022, pp. 1–11, Apr. 2022, doi: 10.1155/2022/2860735.
- [16] S. Le Cessie and J. C. Van Houwelingen, "Ridge estimators in logistic regression," *Applied Statistics*, vol. 41, no. 1, pp. 191–201, 1992, doi: 10.2307/2347628.
- [17] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, 2013, pp. 338–345, [Online]. Available: <http://arxiv.org/abs/1302.4964>.
- [18] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991, doi: 10.1007/bf00153759.
- [19] F. Pedregosa *et al.*, "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [20] R. Rojas, "The backpropagation algorithm," in *Neural Networks*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 149–182.
- [21] H. N. Mpia and I. B. Nephtali, "Gradient back-propagation algorithm in the multi layer perceptron, foundations and case study," *International Journal of Innovation and Applied Studies*, vol. 32, pp. 271–290, 2021.
- [22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, MA, USA: The MIT Press, 2016.
- [23] M. Maynard, *Neural networks: introduction to artificial neurons, backpropagation and multilayer feedforward neural networks with real-world applications*. Independently published, 2020.
- [24] S. L. Salzberg, "C4.5: programs for machine learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993," *Machine Learning*, vol. 16, no. 3, pp. 235–240, Sep. 1994, doi: 10.1007/BF00993309.
- [25] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," *Neural Computation*, vol. 13, no. 3, pp. 637–649, Mar. 2001, doi: 10.1162/089976601300014493.

## BIOGRAPHIES OF AUTHORS



**Amine Benmakhlouf**     currently professor with the Faculty of Science and Technology in Settat (FST), Hassan 1<sup>st</sup> University. His research interests include big data and optimization in distributed systems. He obtained the Ph.D. degrees in 2003 from Hassan II University in Casablanca and the university accreditation to direct research in 2019 from Hassan I University in Settat. His latest publications cover the NoSQL database management systems. He can be contacted at email: [Amine.benmakhlouf@uhp.ac.ma](mailto:Amine.benmakhlouf@uhp.ac.ma).