5th International Conference on Control Engineering&Information Technology (CEIT-2017)
Proceeding of Engineering and Technology –PET
Vol.33 pp. 8-13

Reduced algorithmic complexity when learning Bayesian classifier structure using score-based algorithms

Mariem Kchaou[#1], Lilia Romdhane[*2], Heni Bouhamed[#3]

[#] *Faculty of Economics and Management of Sfax, Tunisia University, Tunisia, Sfax*

[1]maryouma.kchaou29@gmail.com

[3]heni_bouhamed@hotmail.fr

[*2] *Institut Pasteur of Tunis Software, Tunisia University, Tunisia, Sfax*

[2]lilia.romdhane@gmail.com

Abstract*—Bayesian networks are probabilistic graphical models, and are represented by a directed acyclic graph (DAG). In recent years, their popularity and their use has increased significantly in the biological and technological fields, such as the identification of printing problems on Windows (Microsoft), the real-time fault diagnosis for (Nasa), the diagnosis of faults or defects (Hewlet Packard, Intel, American Airlines), the shape recognition and data mining (Nasa) and the medical diagnosis (BiopSys, Radiology Department of the CHU of Tours, American Diabetes Association). The processing of a very large number of variables during the automatic learning of Bayesian network structure is classified as a NP-Difficult problem. In this context, any proposals to mitigate the impact of this problem are strongly solicited by the scientific community especially after the outbreak of the scourge "Big Data" in recent years. Thus, the main objective of this work is to implement and to improve an existing approach [1],[2] and to validate it on a fairly large real database to alleviate the algorithmic complexity of the learning process structure of the Bayesian network without losing information.

Keywords*— Machine learning; Bayesian networks; Classification; Structure learning; clustering; Algorithmic complexity.

I. INTRODUCTION

With the formalism of the Bayesian networks, one can construct an efficient classifier. It is a model with n variables and presented by n + 1 nodes [3],[4],[5]. The simplest model is that of the Bayesian naive classifier (CBN) [6]. It presents a central node that is called "class node" connected to the other nodes that are the descriptive variables (Figure 1).

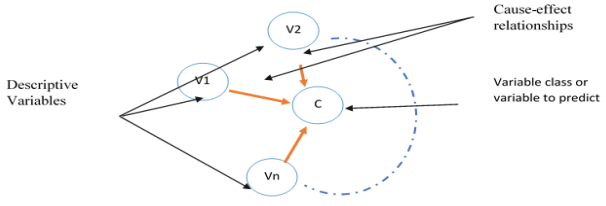

Fig. 1. Graphical representation of the naive Bayesian classifier.

This CBN is used when the learning set is of a moderate size, and the attributes that describe the instance are independent [6]. For this classifier, each instance of C (class variable) is described by the variables V1, V2, ..., Vn. For this type of classifier, the descriptive variables can contribute to the classifier in the same way but the relations between them are not represented. Indeed, the information of each attribute is separated from the information of the other attributes, which is far from perfect for a classification problem. This formalism has been enriched by several propositions considering the relationships between descriptive variables such as the tree-enhanced CBN [4], but the restriction of the number of parents of a node can present a default. As a result, many practitioners have focused on learning the Bayesian classifier (CB) structure using the same algorithms that are used for learning the structure of a classical Bayesian network, implying the consideration of the node of the variable to be predicted as an ordinary node [7],[8],[9],[10],[11],[12],[2]. (eg, .The major disadvantage of this approach is the super-exponential algorithmic complexity Indeed, the possible number of DAGs is expressed by the following recursive formula [13]:

$$r(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} r(n-i) = n^{2^{O(n)}} \quad (1)$$

,where n is the number of variables.

In this context, any proposals to mitigate the impact of this problem are strongly solicited by the scientific community especially after the outbreak of the scourge "Big Data" in recent years.

The rest of this article is organized as follows: in next section, we will present the Bayesian networks (BN). In Section 3, we will try to alleviate the algorithmic complexity of the process in question and consequently will reduce the execution time based on an existing approach [1], trying to improve it by adding the CBN as a starting structure, adapting it to quantitative variables, trying to overcome the local optima when learning Bayesian network structure and validating it on a large real identification database genes which are involved in the genodermatosis (that is any skin condition characterized by a specific mode of genetic transmission). Finally, we will conclude before suggesting relevant potential perspectives for future research.

ISSN 2356-5608

II. BAYESIAN NETWORK

Bayesian networks are probabilistic graphic models,

A Bayesian network B = (G, Θ) is defined by [14]:

- ststructure G = (V, E) which is a directed graph without a circuit (DAG) where V is the set of nodes which represent a set of random variables X = (X_1,…,X_n) and E is the set of arcs
- the parameters Θ = [P(X_i | Pa(X_i)] which are probability distributions for the verification of B at the Markov condition.

The Bayesian networks are graphical models, which represent the probabilistic relations for a set of random variables. They are very readable graphs of the probability laws joined between the variables (Pearl, 1985).

The distribution of probabilities on all variables is defined by [14]:

$$P(X) = P(X_1, X_2, \dots, X_n) = \prod_{i=1}^{n} P(X_i | Pa(X_i)) \quad (2)$$

Bayesian networks are based on the conditional probability and the Bayes theorem in order to calculate the probabilities of the different nodes of the network.

III. METHODOLOGY

In our work, we will try to validate the existing approach [2] on a fairly large real database with quantitative variables, trying to overcome the local optimums when learning the Bayesian classifier structure. Our approach is based on the concept of clustering the structure learning, so we propose in the following to present, first, the methodologies used for the clustering, then we will present our behavior for the clustering, learning of the structure as well as our proposals to avoid the local optimums when applying this process with heuristics.

A. Clustering of the variables

" The automatic classification is the most widespread of the descriptive techniques of data analysis and data mining. It is used when there is a large volume of data in which one seeks to distinguish the homogeneous subsets, and is capable of differentiated treatments and analyzes. " [15].

Indeed, two major types of classification algorithms are distinguished: one which is partitioning methods [16],[17] and the other is hierarchical ascending classification methods. It is known that the hierarchical ascending classification allows good results to be obtained even if the criterion of partitioning is not global. Indeed, this criterion depends on the classes already obtained, two variables in different classes can never be compared. These algorithms are quite complex (complexity of the order of O (n^3) [15]. Partitioning algorithms are favored to those who are hierarchical by the fact that there is a continuous improvement of the quality of the classes and also for their linear algorithmic complexities (for the most widespread) [15]. However, the number of the desired classes must be given in parameters to these algorithms, which present their major disadvantage.

It is noted that the database that will be processed during our work consists of mixed variables (mostly quantitative variables and some qualitative variables). Consequently, we will use a variant of the k-means algorithm [18] that treats these two types of variables simultaneously [19]. For the implementation of the clustering process, we will use the ClustOfVar package with the R language [20]. For the choice of the optimal class number, we will use a function (called stability), which is developed in the ClustOfVar package and based on the boostrap technique [20]: which will be presented in the following steps:

- The application of the hierarchical ascending classification algorithm on all the observations,
- The application of the hierarchical ascending classification algorithm on B bootstrap replications of n observations,
- For each replication, the partitions obtained from 2 to p-1 classes are compared to the partitions of the initial hierarchy with the use of the corrected Rand criterion [21]
- The average of the corrected Rand criteria for each partition obtained (from 2 to p-1) are represented graphically. The optimum class number will be the one preceding a sudden fall of the average of the corrected Rand criterion [20].

B. Learning of the structure

An automatic structure learning will be applied for each cluster by including the class variable. The resulting structures will be gathered around this variable to obtain the final structure.

First, we will try, to help the heuristics that will be used (see Subsection 3.2.1 and Subsection 3.2.2) by adding the structure of the Bayesian naïve classifier as a structure of departure. We will see later that this behavior has greatly improved the results (see Appendix B). In a second step, we will try to add the treatments to avoid the local optimums when using two algorithms tested during our work, namely the Hill Climbing [22],[23],[24],[25],[26],[27] and the Tabu Search[28],[29],[30],[31],[32],[33],[34],[22],[35],[36],[37], [23],[24],[25],[38].

1) The Hill Climbing

The Climbing algorithm (HC) (sometimes called simple local search) is the simplest form of the improvement methods. It starts from an initial state that can be an empty network or a randomly generated structure by iteratively performing the operations including adding, inverting, and deleting by maximizing the improvement of a score that reaches at an optimum [26] (See algorithm 1).

Algorithm1: Climbing algorithm (Hill Climbing (HC))

Notes:

G: a Bayesian network graph.

V: the set of nodes that represent a set of random variables.

ScoreG: the score of the Bayesian network.

G *: a modified Bayesian network graph.

ScoreG *: the score of the modified Bayesian network.

Algorithm : Hill Climbing

Choose a network structure G on V, usually (but not necessarily) empty.
Calculate the score of G, rated ScoreG = Score (G).
maxscore = ScoreG.
Repeat the following steps while maxscore increases:
To verify that adding, removing, or inverting an arc does not result in a cyclic network:
To calculate the modified network score G *, ScoreG * = Score (G *):
If ScoreG *> ScoreG then G = G * and ScoreG = ScoreG *.
To update the maxscore with the new ScoreG value.
Get the acyclic graphic directed G.

2) The Tabu Search

The Tabu Search developed by Fred Glover in 1986 [28], is a global optimization algorithm that controls the integrated heuristic technique. It provides a very flexible search behavior by using the adaptive memory.

Its basic principle is to continue the search whenever it encounters a local optimum. The process is performed using a memory that records the recent search history to avoid the cyclical path. This memory is a file with a variable size according to the choice of the user.

The passage from the current solution to a neighboring solution is accepted if it is not included in the Tabu queue (see algorithm 2).

Algorithm 2: Tabu Search algorithm

Notes:

G: a Bayesian network graph.
V: the set of nodes that represent a set of random variables.
L: an empty queue of the size n
ScoreG: the score of the Bayesian network.
G *: a modified Bayesian network graph.
ScoreG *: the score of the modified Bayesian network.

Algorithm : Tabu Search

To choose a network structure G on V, usually (but not necessarily) empty.
To define an empty queue L of size n (n is chosen by the user)
To calculate the score of G, rated ScoreG = Score (G).
maxscore = ScoreG.
L receives G
Repeat the following steps while maxscore increases:
Verify that adding, removing, or inverting an arc does not result in a cyclic network:
To calculate the modified network score G *, ScoreG * = Score (G *):
If ScoreG *> ScoreG then G = G * and ScoreG = ScoreG * and L = G.
To update the maxscore with the new ScoreG value.
To obtain the acyclic graphic directed G.

For the Climbing algorithm, we will use a method to restart (n times) the process to its end by disrupting the found structure (addition, inversion and deletion of arcs) in order to avoid an optimum local. For the Tabu Search algorithm, we will use a kind of queue to save the previous passages in order to avoid redoing a path which is already traveled.

IV. EXPERIMENTATION

A. DataBase

We will apply our approach to a real database (produced by Dr. Lilia Romdhane, University of Carthage, researcher at the Institut Pasteur in Tunis) which is composed of 224 variables with 1290 observations. The latter representing 648 genes implicating the Genodermatosis (any skin condition characterized by a specific mode of genetic transmission) and 642 genes implicated in other types of skin diseases [39][40]. The aim will be to construct a model to classify new genes in relation to their implications for Genodermatosis. During our work and for each test (there will be 10 learning tests and evaluations for each random division of the observation database into learning data and test data), each time 90% of the database of observations will be used for learning and the rest will be used during the testing phase.

B. Clustering of the variables

The application of the k-means algorithm is preceded by the use of the hierarchical ascending algorithm followed by the bootstrap technique for the estimation of the optimal number of clusters. According to the stability graph (see Figure 2), the corrected Rand average decreases as soon as the score is divided into three, so we decided to choose the partition in 2 that will be introduced in the k-means algorithm. The final result of clustering is presented in Table 2 (Appendix A).

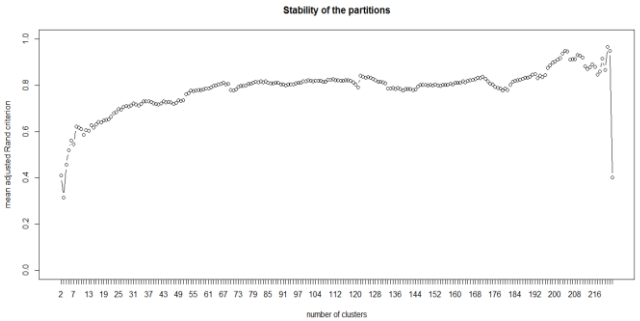

Fig. 2. Graph of partition stabilities from two to 224 variables

C. Learning of the structure

The machine used in this step is a personal computer with a Core i3 microprocessor with 3.58 gigabytes of RAM.

The results of good classifications obtained (see Appendix B for the comparison of the classificated results following the learning of the structure by adding the structure of the naive Bayesian classifier as a starting structure and the classification results following the learning of the structure without adding a starting structure In what follows we will discuss the results in terms of execution time for each method and the algorithm

used, followed by a summary table (Table 1) presenting all the results together.

From these 4 experiments, the gain in terms of execution time is certain, but it remains to verify the accuracy of the classification models according to our new approach.

TABLE 1
SUMMARY OF THE CPU EXECUTION TIMES FOR THE DIFFERENT ALGORITHMS AND THE VARIOUS SCORES TESTED.

Methods \ Algorithms		Hill Climbing with score aic-g	Hill Climbing with score bic-g	Tabu search with score aic-g	Tabu search with score bic-g
Classical Approach		80.02974 min	49.91092 min	65.33142 min	36.50834 min
Our approach	Cluster1	19.32753 min	13.73562 min	29.23134 min	10.96788 min
	Cluster2	2.591396 min	1.152383 min	2.102104 min	1.697241 min
	Sum	21,91892 min	14,88800 min	31,33344 min	12,66512 min

D. Inference

We randomly divided the database 10 times (90% of the database is still used for learning and the remaining 10% for the test) in order to test all the constructed models and thus, we compare the percentages of good classification after learning all the variables simultaneously with those obtained after them according to our new approach. The results in terms of percentages of good classification are represented in the form of histograms for each algorithm with its score whereas the numerical results are presented in the form of tables in Appendix C.

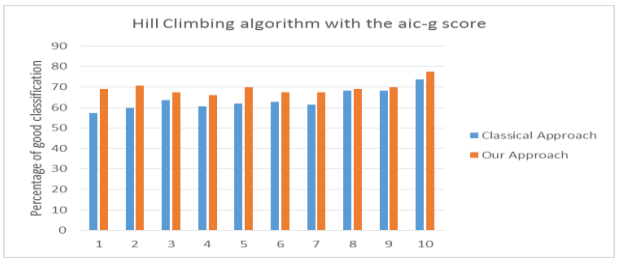

Fig. 3. Shows a comparison of the inference values following the learning of all the variables simultaneously and the learning according to our approach, using the Hill Climbing algorithm with the aic-g score.

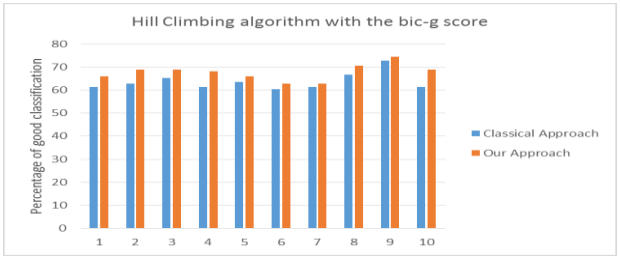

Fig. 4. Shows a comparison of the inference values following the learning of all the variables simultaneously and learning according to our approach, using the Hill Climbing algorithm with the bic-g score.

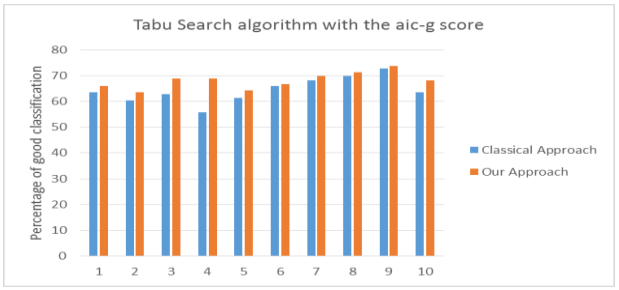

Fig. 5. Shows a comparison of the inference values following the learning of all the variables simultaneously and learning according to our approach, using the Tabu Search algorithm with the aic-g score.

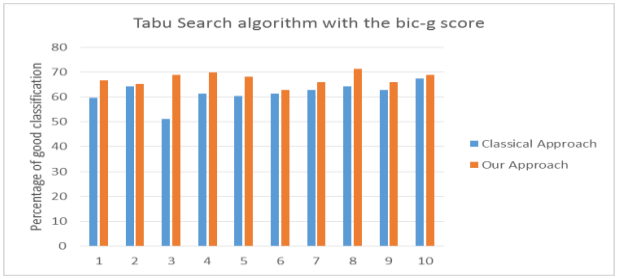

Fig. 6. Shows a comparison of the inference values following learning all the variables simultaneously and learning according to our approach, using the Tabu Search algorithm with the bic-g score.

We calculated the average of the percentages of good classification of the 10 tests following the learning of all the variables simultaneously and following the learning of the variables according to our approach (by applying the Hill Climbing and Tabu search algorithms with the aic-g and bic-g). The result is presented in the form of a histogram (Figure 7), while the numerical result is presented in a table in Appendix C.

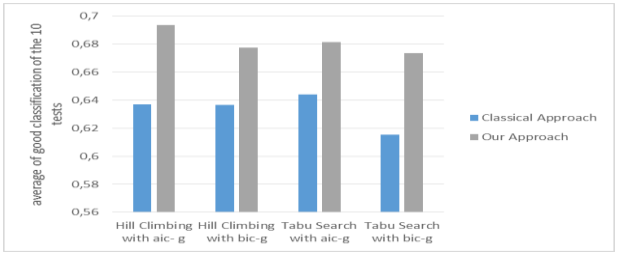

Fig. 7. Shows a comparison of the percentage averages of good classification of 10 tests following all variables learning simultaneously and following learning variables according to our approach (Using Both Hill Climbing and Tabu Search algorithms with aic-g and bic-g scores)

E. Discussion

During our work, we tried to validate (on a real database with quantitative variables) by improving an existing approach that aims to alleviate the algorithmic complexity when learning the Bayesian network structure. This method relies first of all on a step of automatic clustering of the variables and then on the learning of structures of the variables of each cluster with the class variable before gathering all these resulting structures around the latter.

In structure learning, we first tried to help the heuristics by adding the structure of the Bayesian naive classifier as the starting structure (The results of this behavior proved beneficial, see Appendix B). Then, we tried to add treatments to avoid local optimums when applying the two algorithms; Climbing and Tabu Search. For the Climbing algorithm, we used a method that allows the process to be restarted (several times) by disrupting the found structure (addition, inversion and deletion of arcs) in order to avoid optimum. For the Tabu Search algorithm, we used a kind of memory queue that records the recent history of the search in order to avoid a cyclical path.

Thanks to our new approach, we have succeeded in alleviating the algorithmic complexity of the Bayesian network structure learning process (see comparative table of execution times) not only without losing information but also by improving the percentages of good classifications compared to that found using the classical approach, which was not the case before our improvements [1]. Indeed, the different histograms as well as the different confusion matrices show an improvement of the results using our approach for all the tests performed, this improvement can be explained by the over-learning generated by the processing of all (several) variables simultaneously .

V. CONCLUSION

In this work we have tried to improve and validate an existing approach on a real database with quantitative variables whose goal is to alleviate the algorithmic complexity when learning Bayesian network structure. Our approach is based on the concept of clustering upstream of the learning of the structure of the variables of each cluster with the class variable before assembling all these resulting structures around the cluster to avoid local optimums when applying this process with heuristics.

Following these very positive results, the presented and tested behavior could be proposed as a solution allowing to distribute the calculations using the "Big Data" tools available (Example: Hadop, Spark etc ...). And in this sense, the next tasks to be undertaken following this work:

- To develop and to implement our new approach following the philosophy of the MapReduce[1] paradigm (Hadoop),
- To test our approach (implement with Mapreduce / Hadoop) on large masses of data distributed on HDFS (Hadoop File System) platforms,
- And to provide a solution with Apache Spark[2] to parallelize the processing of our approach and test on a cluster[3]

[1] An essential component of the Apache Hadoop software framework. It is a software framework for the distributed processing of large data sets on clusters of basic hardware. This is a subproject of the Apache Hadoop project. The framework takes care of the scheduling, the monitoring, and the re-running tasks that have failed.

Appendix A

TABLE 2

Clustering result of the variables in the database

Cluster 1 : 147 Variables	Cluster 2 : 77 variables
V1 ; V6 ; V7 ; V9 ; V10 ; V11 ; V12 ; V13 ; V14 ; V15 ; V17 ;V20 ; V22 ; V25 ; V28 ; V29 ; V30 ; V31 ; V32 ; V33 ; V35 ; V36 ; V37 ; V38 ; V39 ; V40 ; V41 ; V42 ; V44 ; V45 ; V46 ; V47 ; V48 ; V49 ; V50 ; V51 ; V52 ; V54 ; V55 ; V57 ; V59 ; V60 ; V61 ; V62 ; V63 ; V64 ; V65 ; V66 ; V67 ; V69 ; V70 ;V71 ; V72 ; V73 ; V74 ; V75 ; V77 ; V80 ; V81 ; V82 ; V83; V84 ; V85 ; V86 ; V87 ; V89 ; V92 ; V94 ; V96 ; V97 ; V100 ; V101 ; V104 ; V105 ; V106 ; V112 ; V113 ; V114 ; V116 ;V117 ; V121 ; V124 ; V125 ;V126 ; V127 ; V128 ; V129 ;V130 ; V131 ; V132 ; V133 ;V134 ; V137 ; V138 ; V142 ;V143 ; V144 ; V149 ; V151;V154;V155;V156;V157 ;V158;V160;V164; V166 ;V167 ; V170 ; V173 ; V174 ;V175 ; V176 ; V179 ;V182 ; V185 ; V187 ; V188 ; V190 ;V191 ; V193 ; V194 ; V196 ; V197 ; V198 ; V200 ; V202 ;V203 ; V206 ; V208 ; V209 ;V210 ; V212 ; V214 ; V215 ;V217 ; V218 ; V221 ; V223 ;V224 ; V95 ; V99 ; V103 ; V107 ;V111 ; V115	V2 ; V3 ; V4 ; V5 ; V8 ; V16 ; V18 ; V19 ; V21 ; V24 ; V26 ; V27 ; V34 ; V43 ; V53 ; V56 ; V58 ; V68 ; V76 ; V78 ; V79 ; V88 ; V90 ; V91 ; V93 ; V98 ; V102 ; V108 ; V109 ; V110 ; V118 ; V119 ; V120 ; V122 ; V123 ; V135 ; V136 ; V145 ; V146 ; V147 ; V148 ; V150 ; V152 ; V153 ; V159 ; V161 ; V162 ; V163 ; V165 ; V168 ; V169 ; V171 ; V172 ; V177 ; V178 ; V180 ; V181 ; V183 ; V184 ; V186 ; V189 ; V192 ; V195 ; V199 ; V201 ; V204 ; V205 ; V207 ; V211 ; V213 ; V216 ; V219 ; V220 ; V222 ; V139 ; V140 ; V141

Appendix B

TABLE 3

PERCENTAGE OF A GOOD CLASSIFICATION, AFTER THE APPLICATION OF THE ALGORITHMS WITH DIFFERENT SCORES.

	Without starting structure hc + aic	**Our approach hc + aic**	**Without starting structure hc + bic**	**Our approach hc + bic**	**Without starting structure tabu + aic**	**Our approach tabu + aic**	**Without starting structure tabu + bic**	**Our approach tabu + bic**
Test 1	62,79069767	68,99224806	64,34108527	65,89147287	58,13953488	65,89147287	61,24031007	66,66666667
Test 2	65,89147286	70,54263566	56,58914728	68,99224806	53,48837209	63,56589147	59,68992248	65,11627907
Test 3	61,24031007	67,44186047	65,89147286	68,99224806	62,01550387	68,99224806	60,46511627	68,99224806
Test 4	62,01550387	65,89147287	62,01550387	68,21705426	53,48837209	68,99224806	58,13953488	69,76744186
Test 5	66,66666666	69,76744186	64,34108527	65,89147286	57,36434108	64,34108527	56,58914728	68,21705426
Test 6	63,56589147	67,44186047	58,91472868	62,79069767	61,24031007	66,66666667	61,24031007	62,79069767
Test 7	62,79069767	67,44186047	54,26356589	62,79069767	61,24031007	69,76744186	63,56589147	65,89147286
Test 8	66,66666666	68,99224806	58,91472868	70,54263566	58,13953488	71,31782946	61,24031007	71,31782945
Test 9	60,46511627	69,76744186	70,54263565	74,41860465	66,66666666	73,64341085	53,48837209	65,89147286
Test 10	59,68992248	77,51937984	63,56589147	68,99224806	55,81395348	68,21705426	55,81395348	68,99224806

TABLE 4

AVERAGE CLASSIFICATION OF THE 10 TESTS, AFTER THE APPLICATION OF THE ALGORITHMS WITH THE DIFFERENT SCORES.

	Without starting structure hc + aic	Our approach hc + aic	Without starting structure hc + bic	Our approach hc + bic	Without starting structure tabu + aic	Our approach tabu + aic	Without starting structure tabu + bic	Our approach tabu + bic
Average classification of the 10 tests	0,631782945	0,693798449	0,619379844	0,677519379	0,587596899	0,681395348	0,591472868	0,673643410

[2] Is an open source engine developed specifically for large-scale data processing and analysis. Spark offers the ability to access data from a variety of sources including Hadoop Distributed File System (HDFS) , OpenStack Swift, Amazon S3, and Cassandra.

[3] Is a cluster of servers for performing Big Data data analyzes quickly and efficiently, the different computers that make up the cluster Discover how Hadoop clusters work

Appendix C

TABLE 5
PERCENTAGE OF GOOD CLASSIFICATION, AFTER THE APPLICATION OF THE ALGORITHMS WITH THE DIFFERENT SCORES.

	Classical Approach hc + aic	Our approach hc + aic	Classical Approach hc + bic	Our approach hc + bic	Classical Approach tabu + aic	Our approach tabu + aic	Classical Approach tabu + bic	Our approach tabu + bic
Test 1	57,36434109	68,99224806	61,24031008	65,89147287	63,56589147	65,89147287	59,68992248	66,66666667
Test 2	59,68992248	70,54263566	62,79069767	68,99224806	60,46511628	63,56589147	64,34108527	65,11627907
Test 3	63,56589147	67,44186047	65,11627907	68,99224806	62,79069767	68,99224806	51,1627907	68,99224806
Test 4	60,46511628	65,89147287	61,24031008	68,21705426	55,81395349	68,99224806	61,24031008	69,76744186
Test 5	62,01550388	69,76744186	63,56589147	65,89147286	61,24031008	64,34108527	60,46511628	68,21705426
Test 6	62,79069767	67,44186047	60,46511628	62,79069767	65,89147287	66,66666667	61,24031008	62,79069767
Test 7	61,24031008	67,44186047	61,24031008	62,79069767	68,21705426	69,76744186	62,79069767	65,89147286
Test 8	68,21705426	68,99224806	66,66666667	70,54263566	69,76744186	71,31782946	64,34108527	71,31782945
Test 9	68,21705426	69,76744186	72,86821705	74,41860465	72,86821705	73,64341085	62,79069767	65,89147286
Test 10	73,64341085	77,51937984	61,24031008	68,99224806	63,56589147	68,21705426	67,44186047	68,99224806

TABLE 6
AVERAGE CLASSIFICATION OF THE 10 TESTS, AFTER THE APPLICATION OF THE ALGORITHMS WITH THE DIFFERENT SCORES

	Classical Approach hc + aic	Our approach hc + aic	Classical Approach hc + bic	Our approach hc + bic	Classical Approach tabu + aic	Our approach tabu + aic	Classical Approach tabu + bic	Our approach tabu + bic
Average classification of the 10 tests	0,637209302	0,693798449	0,636434109	0,677519379	0,644186046	0,681395348	0,615503875	0,673643410

REFERENCES

[1] H. Bouhamed, A. Masmoudi, T. Lecroq, and A. Recifais, "Reducing the Structure Space of Bayesian Classifiers Using Some General Algorithms," *J. Math. Model. Algorithms Oper. Res.*, vol. 14, no. 2, pp. 197–237, 2015.

[2] H. Bouhamed, A. Masmoudi, T. Lecroq, and A. Recifais, "Structure space of Bayesian networks is dramatically reduced by subdividing it in sub-networks," *J. Comput. Appl. Math.*, vol. 287, no. March, pp. 48–62, 2015.

[3] S. Langley, P., Sage, "Induction of Selective Bayesian Classifiers," *Actes la Tenth Conf. Uncertain. Artif. Intell.*, pp. 399–406, 1994.

[4] M. Friedman, N., Geiger, D., Goldszmid, "Bayesian Network classifiers," *Mach. Learn.*, pp. 131–163, 1997.

[5] F. Pernkopf, "Bayesian network classifiers versus selective k-NN classifier," *Pattern Recognit.*, pp. 1–10, 2005.

[6] M. Domingos, P., Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Mach. Learn.*, pp. 103–130, 1997.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science (80-.).*, vol. 220, no. 4598, pp. 671–680, 1983.

[8] E. Cooper, G., Hersovits, "A Bayesian method for the induction of probabilistic networks from data," *Mach. Learn.*, pp. 309–347.

[9] S. Ezawa, K., Singh, M., Norton, "Learning goal oriented Bayesian networks for telecommunications risk management," *Actes la Thirteen. Int. Conf. Mach. Learn.*, pp. 139–147, 1996.

[10] F. Witten, H. I., Eibe, "Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations," *Morgan Kaufmann*, 1999.

[11] M. G. Madden, "A New Bayesian Network Structure for Classification Tasks," *Actes la 13th Irish Conf. Artif. Intell. Cogn. Sci.*, pp. 203–208, 2002.

[12] B. Malka, R., Lerner, "Classification of fluorescence in situ hybridization images using belief networks," *Pattern Recognit. Lett. 25*, pp. 1777–1785, 2004.

[13] R. W. Robinson, "Counting unlabeled acyclic digraphs," *Comb. Math. 622*, pp. 28–43, 1977.

[14] A. Pearl. J., Paz, "GRAPHOIDS : A graph-based logic for reasoning about relevance relations,Rapport technique, R_53_L, Cognitive Systems Laboratory, University of California, Los Angeles," 1985.

[15] S. Tufféry, "Data mining et statistique décisionnelle: l'intelligence des données," *Ed. Tech.*, 2010.

[16] E. M. Vigneau, E. and Qannari, "Clustering of variables around latent components," *Comm. Stat. - Simul Comput.*, p. 32(4):1131-1150, 2003.

[17] R. U. Dhillon IS, Marcotte EM, "Diametrical clustering for identifying anti-correlated gene clusters," *Bioinformatics*, vol. 19, no. 13, pp. 1612–1619.

[18] J. MacQueen, "Some methods for classification and analysis of multivariate observations Statistics," *Proc. Fifth Berkeley Symp. Math. Stat. Probab.*, vol. 1, pp. 281--297, 1967.

[19] J. Chavent, M., Kuentz, V., Saracco, "A partitioning method for the clustering of categorical variables," *Actes la IFCS'09, Classif. as a tool Res. Herman locarek-Junge, claus Weihs (Eds), Springer*, pp. 181–205, 2009.

[20] J. Chavent, M., Kuentz, V., Liquet, B., Saracco, "ClustOfVar: an R package for the clustering of variables," *R user Conf. Univ. Warwick Coventry UK*, pp. 63–72, 2011.

[21] A. Green, P., Kreiger, "A Generalized Rand-Index Method for Consensus Clustering of Separate Partitions of the Same Data Base," *J. Classif.*, pp. 63–89, 1999.

[22] M. D, "Learning Bayesian Network Model Structure from Data," *Ph.D. thesis, Sch. Comput. Sci. Carnegie-Mellon Univ. Pittsburgh, PA. Available as Tech. Rep. C.*, 2003.

[23] S. Q. Daly R, "Methods to Accelerate the Learning of Bayesian Network Structures," *Proc. 2007 UKWorkshop Comput. Intell.*, 2007.

[24] N. P. Russell SJ, "Artificial Intelligence: A Modern Approach," *Prentice Hall, 3rd Ed.*, 2009.

[25] N. A. Korb K, "Bayesian Artificial Intelligence. Chapman & Hall/CRC, 2nd edition," 2010.

[26] S. L. Radhakrishnan Nagarajan , Marco Scutari, *Bayesian Networks in R with Applications in Systems Biology*. 2013.

[27] M. A. Al-Beta, "b-Hill climbing: an exploratory local search," *Neural Comput. Appl.*, 2016.

[28] F. Glover, "Chemins futurs pour la programmation en nombre entier et les liens vers l'intelligence artificielle," *Rech. Inform. opérationnelle*, vol. 13, pp. 533–549, 1986.

[29] P. Hansen, "La guérison la plus raide de la décennie la plus légère pour la programmation combinatoire," *Congrès sur les méthodes numériques en Optim. Comb. Capri, Ital.*, 1986.

[30] C. R. Reeves, "Modern Heuristic Techniques for Combinatorial Problems," *John Wiley Sons, Inc*, 1993.

[31] M. Glover, F., Kelly, J. P., and Laguna, "Genetic Algorithms and Tabu search: Hybrids for Optimization," *Comput. Oper. Res.*, vol. 22, no. 1, pp. 111 – 134, 1995.

[32] M. Glover, F. and Laguna, "Tabu search," *Norwell, MA Kluwer Acad. Publ.*, 1997.

[33] D. Pham, D.T. and Karaboga, "Intelligent Optimisation Techniques – Genetic Algorithms, Tabu search, Simulated Annealing and Neural Networks," 2000.

[34] S. Hanafi, "On the Convergence of Tabu search," *J. Heuristics*, vol. 7, pp. 47 – 58, 2001.

[35] H. Ji, M. and Tang, "Global Optimizations and Tabu search Based on Mamory," *Appl. Math. Comput.*, vol. 159, pp. 449 – 457, 2004.

[36] D. Hertz, A., Taillard, E. and Werra, "A Tutorial on Tabu search," *Accessed on : http://www.cs.colostate.edu/~whitley/CS640/hertz92tutorial.pdf*.

[37] G. J. Hillier, F.S. and Lieberman, "Introduction to Operations Research," 2005.

[38] M. Islam, T., Shahriar, Z., Perves, M.A. and Hasan, "University Timetable Generator Using Tabu search.," *J. Comput. Commun.*, vol. 4, pp. 28–37, 2016.

[39] A. R. Lilia Romdhane, Amel Louhichi, Heni Bouhamed, Sonia Abdelhak, "Bioinformatic approach for genodermatosis genes study," *Proc. 1st Int. Conf. Eng. Sci. Biol. Med. Sci. Biol. Med.*, no. 88, 2013.

[40] A. R. Lilia Romdhane, Heni Bouhamed, Amel Louhichi, Sonia Abdelhak, "Skin genetic disease genes: Patterns and Prediction," *Proc. 2st Int. Conf. Eng. Sci. Biol. Med. Sci. Biol. Med.*