International Journal of Artificial Intelligence and Machine Learning
Volume 10 • Issue 2 • July-December 2020

Efficient Closure Operators for FCA-Based Classification

Nida Meddouri, Caen-Normandy University, France & GREYC, ENSICAEN, Caen, France
https://orcid.org/0000-0002-7815-630X

Mondher Maddouri, College of Business, University of Jeddah, Saudi Arabia

ABSTRACT

Knowledge discovery in databases (KDD) aims to exploit the large amounts of data collected every day in various fields of computing application. The idea is to extract hidden knowledge from a set of data. It gathers several tasks that constitute a process, such as: data selection, pre-processing, transformation, data mining, visualization, etc. Data mining techniques include supervised classification and unsupervised classification. Classification consists of predicting the class of new instances with a classifier built on learning data of labeled instances. Several approaches were proposed such as: the induction of decision trees, Bayes, nearest neighbor search, neural networks, support vector machines, and formal concept analysis. Learning formal concepts always refers to the mathematical structure of concept lattice. This article presents a state of the art on formal concept analysis classifier. The authors present different ways to calculate the closure operators from nominal data and also present new approach to build only a part of the lattice including the best concepts. This approach is based on Dagging (ensemble method) that generates an ensemble of classifiers, each one represents a formal concept, and combines them by a voting rule. Experimental results are given to prove the efficiency of the proposed method.

KEYWORDS

Classification Rules, Closure Operator, Dagging, Data Mining, Ensemble method, Formal Concept Analysis, Machine Learning, Nominal Concept

INTRODUCTION

The classification approach, which is based on formal concept analysis, is a symbolic approach allowing the extraction of correlations, reasons and rules according to the concepts discovered from data. Many learning methods based on Formal Concept Analysis are proposed, such as: JSM-method (Blinova, Dobrynin, Finn, Kuznetsov & Pankratova, 2003), CLANN (Tsopze, Mephu-Nguifo & Tindo, 2007)), CITREC (Douar, Latiri & Slimani, 2008), NAVIGALA (Visani, Bertet & Ogier, 2011), HMCS-FCA-SC (Ferrandin et al, 2013), SPFC (Ikeda & Yamamoto, 2013) and MCSD-FCA-PS (Buzmakov et al, 2016). Unfortunately, this approach encountered some problems such as exponential complexity (in the worst case), a high error rate and over-fitting (Meddouri & Maddouri, 2008,2010). Most of them handle only binary data. The construction of the all concepts can be either exhaustive or non-contextual. There is absence of the adaptive selection of concepts (Meddouri & Maddouri, 2008).

For these reasons, we focused in our research on ensemble methods used to improve the error rate of any single learner. We proposed BFC (MeddouriI & Maddouri, 2009) and BNC (Meddouri

DOI: 10.4018/IJAIML.2020070105

& Maddouri, 2010) methods based on sequential learning (Boosting). All the data are considered in each learning step and weights are assigned to learning instances. However, it was proved that sequential learning (Boosting) is not interesting, insufficient for a more efficient classifier as Decision Tree (Meddouri & Maddouri, 2010). Other ensemble learning methods exists, and they are based on parallel learning. The difference between these two ensemble methods, derives from how to select data for learning. They are distinguished by the data sampling techniques as Bootstrapping used to learn the classifiers from particular subsets. The particularity of learning from a Bootstrap is to combine hard learning instances to misleading instances in the training set (unlike the sequential approach) (Breiman, 96a, 96b). The best known method, which is based on this type of learning is Dagging (Disjoint samples aggregating) (Kotsiantis, Anyfantis, Karagiannopoulus & Pintelas, 2007) that creates a number of disjoint groups and stratified data from the original learning data set (Ting & Witten, 1997), each considered as a subset of learning. The classifier is built on this learning sets. The predictions are then obtained by combining the classifiers outputs by majority voting (Ting & Witten, 1997). This method has shown its importance in recent work (Meddouri, Khoufi & Maddouri, 2014). Then, we propose to use this technique in this work to study the classifier ensembles based on formal concepts, since, no study has focused on the formal concepts in the context of parallel learning.

In section 2, we present a state of the art on Formal Concept Analysis. In section 3, we propose classifiers using closure operators based on Formal Concept Analysis. In the section 4, an experimental study is presented to evaluate the performance of nominal classifiers based on different closure operators. An experimental study is also presented showing the importance of parallel learning compared to single learning for classifiers based on Formal Concept Analysis.

FORMAL CONCEPT ANALYSIS AND CLASSIFICATION

Definition

A formal context is a triplet $\left\langle \mathcal{I}, \mathcal{A}, \mathcal{R} \right\rangle$, where $\mathcal{I} = \left\{ \mathrm{i}_1, \mathrm{i}_2, \ldots, \mathrm{i}_\mathrm{n} \right\}$ is a set of *n* instances, $\mathcal{A} = \left\{ a_1, a_2, \ldots, a_m \right\}$ a set of *m* binary attributes and $\mathcal{R}$ is a binary relation defined between $\mathcal{I}$ and $\mathcal{A}$. $\mathcal{R}\left(i_k, a_l\right) = 1$ means that k^{th} instance i_k verifies the l^{th} attribute a_l in relation $\mathcal{R}$ (Stumme, Ganter & Wille, 2005). The context is often represented by a cross-table or a binary-table as shown in Table 1[1].

Let $X \subseteq \mathcal{I}$ and $Y \subseteq \mathcal{A}$ be two finite sets. For both sets X and Y, operators $\varphi\left(X\right)$ and $\delta\left(Y\right)$ are defined as:

- $\varphi\left(X\right) = \left\{ y \mid \forall x, x \in X \sim and \sim R\left(x, y\right) = 1 \right\}$
- $\delta\left(Y\right) = \left\{ x \mid \forall y, y \in Y \sim and \sim R\left(x, y\right) = 1 \right\}$

Operator φ defines the properties shared by all elements of X. Operator δ defines the instances which share the same attributes included in Y. Operators φ and $'$ define the Galois connection between sets $\mathcal{I}$ and $\mathcal{A}$ (Stumme, Ganter & Wille, 2005). An example from the formal context Weather of Table 1, we consider $X = \left\{ i_1, i_2 \right\}$ and $Y = \left\{ a_4, a_8 \right\}$, so $\varphi\left(X\right) = \left\{ a_1, a_4 \right\}$ and $\delta\left(Y\right) = \left\{ i_1, i_3, i_{13} \right\}$.

The closure operators are $X'' = \delta \circ \varphi\left(X\right)$ and $Y'' = \varphi \circ \delta\left(Y\right)$. Finally, the closed sets X and Y are defined by $Y = \varphi \circ \delta\left(Y\right)$ and $X = \delta \circ \varphi\left(X\right)$ (Stumme, Ganter & Wille, 2005). An example from the previous formal context of Table 1, we have:

Table 1. Illustration of the formal context (Weather data under binary format seen in Table 2)

I\A	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	Play
i_1	1	0	0	1	0	0	0	1	No
i_2	1	0	0	1	0	0	0	0	No
i_3	0	1	0	1	0	0	0	1	Yes
i_4	0	0	1	0	1	0	0	1	Yes
i_5	0	0	1	0	0	1	1	1	Yes
i_6	0	0	1	0	0	1	1	0	No
i_7	0	1	0	0	0	1	1	0	Yes
i_8	1	0	0	0	1	0	0	1	No
i_9	1	0	0	0	0	1	1	1	Yes
i_{10}	0	0	1	0	1	0	1	1	Yes
i_{11}	1	0	0	0	1	0	1	0	Yes
i_{12}	0	1	0	0	1	0	0	0	Yes
i_{13}	0	1	0	1	0	0	1	1	Yes
i_{14}	0	0	1	0	1	0	0	0	No

Table 2. Specification of binary attributes

Attributes	Signification
a_1	Outlook=sunny
a_2	Outlook=overcast
a_3	Outlook=rainy
a_4	Temperature=hot
a_5	Temperature=mild
a_6	Temperature=cool
a_7	Humidity
a_8	Windy

- $\left\{i_5, i_{10}\right\}'' = \delta \circ \varphi\left(\left\{i_5, i_{10}\right\}\right) = \delta\left(\left\{a_3, a_7, a_8\right\}\right) = \left\{i_5, i_{10}\right\}$.
- $\left\{a_3, a_7\right\}'' = \varphi \circ \delta\left(\left\{a_3, a_7\right\}\right) = \varphi\left(\left\{i_5, i_6, i_{10}\right\}\right) = \left\{a_3, a_7\right\}$.

A formal concept of the context $\left\langle \mathcal{I}, \mathcal{A}, \mathcal{R} \right\rangle$ is a pair (X, Y) where $X \subseteq \mathcal{I}$, $Y \subseteq \mathcal{X}$, $\varphi\left(X\right) = Y$ and $\delta\left(Y\right) = X$. Sets X and Y are called, respectively, the extent (domain) and intent (co-domain) of the formal concept (Stumme, Ganter, & Wille, 2005). For example, ($\left\{i_1, i_2\right\}, \left\{a_1, a_4\right\}$) is a formal concept from Weather context (Table 1). The set of attributes common to i_1 and i_2 is $\left\{a_1, a_4\right\}$. The set of instances that share both a_1 and a_4 is $\left\{i_1, i_2\right\}$. Contrariwise, ($\left\{i_2, i_3\right\}, \left\{a_4\right\}$) is not a formal concept since $\varphi\left(\left\{i_2, i_3\right\}\right) = \left\{a_4\right\}$ and $\delta\left(\left\{a_4\right\}\right) = \left\{i_1, i_2, i_3, i_{13}\right\}$.

From the formal context $\left\langle \mathcal{I}, \mathcal{A}, \mathcal{R} \right\rangle$, we can extract all possible concepts organized as a complete lattice (called Galois lattice (Stumme, Ganter & Wille, 2005)). We define the following partial order relation '$\ll$' between two concepts as: ($X\_\{1\}, Y\_\{1\}) \ll (X_2, Y_2)$ if and only if ($X_1 \subseteq X_2$) and ($Y_2 \subseteq Y_1$). The concepts (X_1, Y_1) and (X_2, Y_2) are represented by nodes in the lattice diagram.

Formal Concept Analysis Based Classification

A classification method must determine the class of new instances. The Galois lattice can be used in classification as a search space in which we evolve level to another, by validating the characteristics associated to the new instance (Visani, Bertet & Ogier, 2011). Many classification methods were proposed in the literature using Galois lattices (Trabelsi, Meddouri & Maddouri, 2016).

Exhaustive Classification Methods

Using only one single classifier to generate all the formal concepts, is an exhaustive way to build a learning model based on Galois lattices of formal concepts. Many classification methods exist in the literature using complete lattice of concepts such as JSM-method (Blinova, Dobrynin, Finn, Kuznetsov & Pankratova, 2003), NAVIGALA (Visani, Bertet & Ogier, 2011), HMCS-FCA-SC (Ferrandin et al., 2013) and SPFC (Ikeda & Yamamoto, 2013). These recent methods carried out the validation of the characteristics associated to each concept in the lattices level by level. The navigation in the lattice of concepts starts from the minimal concept where all the concepts of the lattice are considered as candidates without having an idea on their validity. However, they vary according to the criteria used for concepts selection and the size of lattices outlining formal concepts (Trabelsi, Meddouri, & Maddouri, 2016). There are three common limitations for systems based on concept lattice: the complexity (time and space) of generating the lattice is exponential, the navigation in huge search space is hard (Meddouri & Maddouri, 2008) and the used data is binary. For these reasons, many researchers focused on sub-lattice-based classification.

Other methods can build a sub-lattice of concepts, which reduces their theoretical complexity and their times of execution. A sub-lattice is a mathematical structure which represents a part of the full lattice in a selective way (Stumme, Ganter & Wille, 2005) (Trabelsi, Meddouri & Maddouri, 2016). Classification based on sub-lattice is similar to that started from a complete lattice. The major difference between complete lattice and sub-lattice-based classification is the number of concepts generated. However, their limitation is the possible loss of information in a condensed data representation or a partial reproduction of the full lattice. Systems like IPR (Maddouri, 2004), CLANN (Tsopze, Mephu-Nguifo & Tindo, 2007) and CITREC (Douar, Latiri & Slimani, 2008), MCSD-FCA-PS (Buzmakov et al, 2016) are characterized by the ability to build a part of the concept lattice and induce classification rules.

Adaptive Classification Methods

Generating many classifiers from the same model and combining them by a fusion technique is an adaptive way to build a learning model. Various methods have been proposed based on sequential approach (Boosting) such as BFC (MeddouriI & Maddouri, 2009), BNC (Meddouri & Maddouri, 2010) and others based on parallel approach (Bagging) such as DNC (Meddouri, Khoufi & Maddouri, 2014), FPS-FCA (Kuznetsov, 2013) and RMCS (Kashnitsky & Ignatov, 2014).

Boosting is an adaptive approach, which makes it possible to correctly classify an object that can be badly classified by an ordinary classifier. The main idea of Boosting is to build many classifiers who complement each other, in order to build a more powerful classifier. At first, it selects a subset of instances from the learning data set (different subset from the training data set in each iteration). Then, it builds a classifier using the selected instances. Next, it evaluates the classifier on the learning data set, and it starts again T times (T is the number of generated classifiers). AdaBoost (Adaptive Boosting) is the most well-known method of Boosting for classifiers generation and combination.

In parallel approach, Bagging is based on Bootstraps. Each classifier is trained on a set of n' training instances ($n' < n$), drawn randomly with replacement from the original training set of size n. Such a training set is called a Bootstrap replicate of the original set. Each Bootstrap replicate contains, on average, 63.2% of the original training set, with many instances appearing several times. Predictions on the new instances are made by taking the majority vote of the ensemble. The particularity of these training sets is to reduce the impact of hard instances to learn (called outliers and misleaders) (Skurichina & Duin, 1998).

In the literature of vote methods, the majority vote can turn good classifiers to almost optimal (Breiman, 1996a, 1996b). Bagging is typically applied to learning algorithms that are unstable, i.e., a small change in the training set leads to a noticeable change in the model produced (Melville & Mooney, 2005). Because each ensemble member is not exposed to the same set of instances, they are different from each other. By voting the predictions of each of these classifiers, Bagging seeks to reduce the error due to variance of the base classifier. Bagging of stable learners, such as Naive Bayes, does not reduce error (Melville & Mooney, 2005). The authors of (Kuncheva, Skurichina & Duin, 2002) report that parallel learning improves the performance of unstable classifier such as neural networks and decision trees. They report that Bagging is not very beneficial for improving the performance of a linear classifier on large data. It will be then advantageous to use these methods with unstable classifier which is the case of classifiers based on formal concepts (Meddouri, Khoufi & Maddouri, 2014).

In the literature of data sampling methods, stratified sampling has proved to be efficient.

Disjoint and stratified data sets are more representative of the original training data base (Ting & Witten, 1997). Learning from stratified data samples allows to generate a more efficient classifier than those generated from the weighted data in the case of sequential learning classifiers. Dagging has the particularity to learn in parallel from stratified data sets (Kotsiantis, Anyfantis, Karagiannopoulus & Pintelas, 2007).

PROPOSED METHODS LABEL

A nominal (multi-valued) context is a quadruple $\left\langle \mathcal{I}_{nom}, \mathcal{A}_{nom}, \mathcal{V}, \mathcal{R}_{nom} \right\rangle$, where:

- $\mathcal{I}_{nom} = \left\{ \mathrm{i}_{1_{\mathrm{nom}}}, \mathrm{i}_{2_{\mathrm{nom}}}, \ldots, \mathrm{i}_{\mathrm{k}_{\mathrm{nom}}}, \ldots, \mathrm{i}_{\mathrm{n}_{\mathrm{nom}}} \right\}$ is the set of n_{nom} instances.
- $\mathcal{A}_{nom} = \left\{ a_{1_{nom}}, a_{2_{nom}}, \ldots, a_{l_{nom}}, \ldots, a_{m_{nom}} \right\}$ is the set of m_{nom} attributes.
- $\mathcal{V} = \left\{ v_{1^l}, v_{2^l}, \ldots v_{p^l}, \ldots, v_{*^l} \right\}$ is the set of values
- $\mathcal{R}_{nom}$ is a relation defined between $\mathcal{I}_{nom}$, $\mathcal{A}_{nom}$ and $\mathcal{V}$. So $\mathcal{R}_{nom}$ is a set of triples.

Each triple $\left(i_{k_{nom}}, a_{l_{nom}}, v_{p^l} \right)$ means that $\mathrm{v}_{\mathrm{p}^{\mathrm{l}}}$ is a value taken by the l^{th} nominal attribute $a_{l_{nom}}$ on k^{th} nominal instance $i_{k_{nom}}$ ($R\left(i_{k_{nom}}, a_{l_{nom}}, v_{p^l} \right)$ exist) (Stumme, Ganter & Wille, 2005). Table 3 represents the nominal (multi-valued) context of Weather.

We denote by n_{nom} the number of nominal instances $\mathcal{I}_{nom}$ and m_{nom} the number of nominal attributes $\mathcal{A}_{nom}$ with:

$$\mathcal{A}_{nom} = \left\{ a_{l_{nom}} \mid \exists i_{k_{nom}} \in \mathcal{I}_{nom}, \exists a_{l_{nom}} \in \mathcal{A}_{nom}, \mathcal{A}_{nom}\left(i_{k_{nom}} \right) = a_{l_{nom}} \right\} \tag{1}$$

Table 3. Illustration of the multi-valued context (Weather data under nominal format)

	Outlook	Temperature	Humidity	Windy	Play
i_1	sunny	hot	high	false	No
i_2	sunny	hot	high	true	No
i_3	overcast	hot	high	false	Yes
i_4	rainy	mild	high	false	Yes
i_5	rainy	cool	normal	false	Yes
i_6	rainy	cool	normal	true	No
i_7	overcast	cool	normal	true	Yes
i_8	sunny	mild	high	false	No
i_9	sunny	cool	normal	false	Yes
i_{10}	rainy	mild	normal	false	Yes
i_{11}	sunny	mild	normal	true	Yes
i_{12}	overcast	mild	high	true	Yes
i_{13}	overcast	hot	normal	false	Yes
i_{14}	rainy	mild	high	true	No

A pertinent nominal concept within the data set is extracted by selecting the nominal attribute which maximize the measure of Informational Gain (*IG*) calculated from the learning context.

$$IG\left(a_{l_{nom}}, \mathcal{I}_{nom}\right) = E\left(\mathcal{I}_{nom}\right) - \sum_{p=1}^{*l} \frac{S\left(v_p^l\right)}{N} E\left(v_p^l\right) \tag{2}$$

$a_{l_{nom}}$ is represented by ($*^l$) different values. The Information Gain $IG\left(a_{l_{nom}}, \mathcal{I}_{nom}\right)$ of the nominal attribute $a_{l_{nom}}$ is calculated from the entropy function: *E()*. $E\left(\mathcal{I}_{nom}\right)$ calculates the entropy of the whole nominal instances $\mathcal{I}_{nom}$. $E\left(v_p^l\right)$ calculates the entropy of a value v_p^l of the $a_{l_{nom}}$ on $\mathcal{I}_{nom}$. *S()* calculates the pertinence of a value v_p^l of the $a_{l_{nom}}$ on $\mathcal{I}_{nom}$. n_{nom} represents the number of nominal instances from $\mathcal{I}_{nom}$.

Proposition 1: From a nominal context (multi-valued), the δ operator is set by:

$$\delta\left(v_p^l\right) = \left\{i_{k_{nom}} \in \mathcal{I}_{nom} \mid R\left(i_{k_{nom}}, a_{l_{nom}}, v_{p^l}\right) exist\right\} \tag{3}$$

Proposition 2: From a nominal context (multi-valued), the Æ operator is set by:

$$\varphi\left(\left\{i_{nom}\right\}\right) = \left\{v_{p^l} \mid \forall i_{k_{nom}}, i_{k_{nom}} \in \left\{i_{nom}\right\} \text{and } \exists a_{l_{nom}} \in \mathcal{A}_{nom} \mid R\left(i_{k_{nom}}, a_{l_{nom}}, v_{p^l}\right) exist\right\} \tag{4}$$

Then, we look for the other attributes describing all the extracted instances (using the closure operator $\delta \circ \varphi\left(v_p^l\right)$).

Learning Classification Rules Based on Nominal Concepts

We construct nominal concepts associated with one or each value v_p^l of the nominal attribute $a_{l_{nom}}\left(\left\{\delta\left(v_p^l\right)\right\},\left\{\delta \circ \varphi\left(v_p^l\right)\right\}\right)$. A classifier is obtained by seeking the majority class associated with the extent of one or each concept ($\delta\left(v_p^l\right)$). It induces a classification rule. The condition part of each rule is made up by the conjunction of the attributes included in the intent: $\delta \circ \varphi\left(v_p^l\right)$. The conclusion part of the rule is made up by the majority class.

We consider here 4 variants to learn classification rules based on Nominal Concepts. These 4 variants differ by the way we calculate the closure operator $\delta \circ \varphi$.

- CpNC_COMV: Classifier pertinent Nominal Concept based on Closure Operator for Multi-Values of the pertinent nominal attribute.
- CpNC_CORV: Classifier pertinent Nominal Concept based on Closure Operator for Relevant-Values of the pertinent nominal attribute.
- CaNC_COMV: Classifier all Nominal Concept based on Closure Operator for Multi-Values of the pertinent nominal attribute.
- CaNC_CORV: Classifier all Nominal Concept based on Closure Operator for Relevant-Values of the pertinent nominal attribute.

Classifier Based on Pertinent Nominal Concepts (CpNC)

Closure Operator on Each Value From the Pertinent Attribute (CpNC_COMV)

Once the pertinent nominal attribute is selected ($a_{l_{nom}}^*$), we extract associated instances to each value v_p^l from this attribute ($\delta\left(v_p^l\right)$). Then, we look for the other attributes describing all the extracted instances (using the closure operator $\delta \circ \varphi\left(v_p^l\right)$). We construct nominal concepts associated with each value v_p^l of the nominal attribute $a_{l_{nom}}^*\left(\left\{\delta\left(v_p^l\right)\right\},\left\{\delta \circ \varphi\left(v_p^l\right)\right\}\right)$. A classifier is obtained by seeking the majority class associated with the extent of each concept ($\delta\left(v_p^l\right)$). It induces a classification rule. The condition part of each rule is made up by the conjunction of the attributes included in the intent: $\delta \circ \varphi\left(v_p^l\right)$. The conclusion part of the rule is made up by the majority class.

Algorithm 1: Algorithm of Classifier pertinent Nominal Concept based on Closure Operator for multi-values of the pertinent nominal attribute (CpNC_COMV)

Input: Sequence of n_{nom} instances $\mathcal{I}_{nom}=\left\{\left(i_{1_{nom}}, y_1\right), \ldots,\left(i_{n_{nom}}, y_k\right)\right\}$ with labels $\mathcal{K}=\left\{y_1, .., y_k\right\}$.

Output: h_{CpNC_COMV} a classifier.

Begin

From $\mathcal{I}_{nom}$, find the attribute $a_{l_{nom}}^*$ having the best IG value using (2);

For each nominal value v_p^l of $a_{l_{nom}}^*$, calculate the closure associated to each v_p^l based on (3) and (4) to construct pertinent

nominal concepts ($\{\delta\left(v_p^l\}\right)$, $\delta \circ \varphi\left(\left\{v_p^l\right\}\right)$);
Determine the majority class y associated with the extent of each pertinent concept ($\delta\left(v_p^l\right)$);
Induce and combine the new classification rules into h_{CpNC_COMV};
Return h_{CpNC_COMV};
End

Closure Operator on the Relevant Value of the Pertinent Attribute (CpNC_CORV)

Back to *CpNC_COMV* and once the pertinent nominal attribute ($a^*_{l_{nom}}$) is selected, we extract instances associated to more relevant value v^{*l}_{*p} of this attribute. So we construct only one pertinent concept associated to the most relevant value v^{*l}_{*p} of the pertinent attribute $a^*_{l_{nom}}$ $\left(\left\{\delta\left(a^*_{l_{nom}}\right)\right\},\left\{\delta \circ \varphi\left(v^{*l}_{*p}\right)\right\}\right)$. A *CpNC_CORV* classifier is obtained by seeking the majority class associated to the extent of the obtained pertinent concept ($\left\{\delta\left(v^{*l}_{*p}\right)\right\}$)It induces only one classification rule. The condition part of this rule is made up by the conjunction of the attributes included in the intent: $\left\{\delta \circ \varphi\left(v^{*l}_{*p}\right)\right\}$. The conclusion part of this rule is made up by the majority class.

Algorithm 2: Algorithm of Classifier pertinent Nominal Concept based on Closure Operator for the relevant value of the pertinent nominal attribute (CpNC_CORV)

Input: Sequence of n_{nom} instances $\mathcal{I}_{nom} = \left\{\left(i_{1_{nom}}, y_1\right),\ldots,\left(i_{n_{nom}}, y_k\right)\right\}$ with labels $\mathcal{K} = \left\{y_1,..,y_k\right\}$.
Output: h_{CpNC_CORV} a classifier.
Begin
From $\mathcal{I}_{nom}$, find the attribute $a^*_{l_{nom}}$ having the best IG value using (2) and its more relevant value v^{*l}_{*p};
From v^{*l}_{*p}, calculate the closure based on (3) and (4) to construct a pertinent nominal concept $\left(\left\{\delta\left(v^{*l}_{*p}\right)\right\},\left\{\delta \circ \varphi\left(v^{*l}_{*p}\right)\right\}\right)$;
Determine the majority class y associated with the extent of the obtained pertinent concept $\left(\left\{\delta\left(v^{*l}_{*p}\right)\right\}\right)$;
Induce the classification rule to h_{CpNC_CORV};
Return h_{CpNC_CORV};
End

Classifier Based on all Nominal Concepts (CaNC)

CaNC consider the whole of training instances and use nominal attributes. While *CpNC* calculate only the closure of the pertinent attribute, *CaNC* consider all nominal attributes and calculate the closure associated to each one.

Closure Operator on Each Value from the All Nominal Attributes

On each nominal attribute ($a_{l_{nom}}$) from the nominal context, we extract associated instances to each value v_p^l from this attribute, using $\delta\left(v_p^l\right)$ (3). Then we extract one or many common nominal values associated with one or many common nominal attributes, using $\delta \circ \varphi\left(v_p^l\right)$ (4). So, we construct many nominal concepts associated with each value v_p^l of the nominal attribute $a_{l_{nom}}$: $\left(\left\{\delta\left(v_p^l\right)\right\},\left\{\delta \circ \varphi\left(v_p^l\right)\right\}\right)$. A *CaNC_COMV* classifier is obtained by investigating the majority class associated to the extent of each nominal concept $\left(\left\{\delta\left(v_p^l\right)\right\}\right)$. It induces many classification rules. The condition part of each rule is made up of the conjunction of the attributes included in the intent: $\left\{\delta \circ \varphi\left(v_p^l\right)\right\}$. The conclusion part of the rule is made up of the majority class.

Algorithm 3: Algorithm of Classifier pertinent Nominal Concept based on Closure Operator for multi-values of each nominal attribute (CaNC_COMV)

Input: Sequence of n_{nom} instances $\mathcal{I}_{nom} = \left\{\left(i_{1_{nom}}, y_1\right), \ldots, \left(i_{n_{nom}}, y_k\right)\right\}$ with labels $\mathcal{K} = \left\{y_1, .., y_k\right\}$

Output: h_{CaNC_COMV} a classifier.

Begin

For each $a_{l_{nom}}$ and for each nominal value v_p^l, calculate the closure associated to v_p^l based on (3) and (4) to construct nominal concepts $\left(\left\{\delta\left(v_p^l\right)\right\},\left\{\delta \circ \varphi\left(v_p^l\right)\right\}\right)$;

Determine the majority class y associated with the extent of each nominal concept $\left(\left\{\delta\left(v_p^l\right)\right\}\right)$;

Induce and combine the classification rules into h_{CaNC_COMV} ;

Return $h_{\backslash CaNC\_COMV}$;

End

Closure Operator on the Relevant Value from Each Nominal Attribute

On each nominal attribute ($a_{l_{nom}}$) from the nominal context, we extract associated instances to its relevant value v_{*p}^l, using $\delta\left(v_{*p}^l\right)$ (3). Then we extract one or many common nominal values associated with one or many common nominal attributes, using $\delta \circ \varphi\left(\left\{v_{*p}^l\right\}\right)$ (4}). So, we construct one nominal concept associated with the pertinent value v_{*p}^l of the nominal attribute $a_{l_{nom}}$: $\left(\delta\left(v_{*p}^l\right), \delta \circ \varphi\left(v_{*p}^l\right)\right)$. A CaNC_CORV classifier is obtained by investigating the majority class associated to the extent of each nominal concept ($\left\{\delta\left(v_{*p}^l\right)\right\}$). It induces many classification rules. The condition part of each rule is made up by the conjunction of the attributes included in the intent: $\left(\left\{\delta \circ \varphi\left(v_{*p}^l\right)\right\}\right)$. The conclusion part of the rule is made up by the majority class.

Algorithm 4: Algorithm of Classifier pertinent Nominal Concept based on Closure Operator for a relevant value of each nominal attribute (CaNC_CORV)

Input: Sequence of n_{nom} instances $\mathcal{I}_{nom} = \left\{\left(i_{1_{nom}}, y_1\right), \ldots, \left(i_{n_{nom}}, y_k\right)\right\}$ with labels $\mathcal{K} = \left\{y_1, .., y_k\right\}$.

Output: h_{CaNC_CORV} a classifier.

Begin

For each $a_{l_{nom}}$ and for its relevant nominal value v_{*p}^{l}, calculate the closure associated to $\left(v_{*p}^{l}\right)$ based on (3) and (4) to construct nominal concepts $\left(\left\{\delta\left(v_{*p}^{l}\right)\right\}, \left\{\delta \circ \varphi\left(v_{*p}^{l}\right)\right\}\right)$;

Determine the majority class y associated with the extent of each obtained concept $\left\{\delta\left(v_{*p}^{l}\right)\right\}$;

Induce and combine the classification rules into h_{CaNC_CORV} ;

Return h_{CaNC_CORV} ;

End

Dagging FCA Based Classifiers

Recently, a great number of researches in machine learning have been concerned with ensemble learning of classifiers that allow the improvement of a single learner performances (Meddouri, Khoufi & Maddouri, 2014; Kuznetsov, 2013; Kashnitsky & Ignatov, 2014). The two principal reasons for this success are probably the simplicity of implementation and the recent theorems relative to the boundaries, the margins, or to the convergence (Meddouri & Maddouri, 2010).

In (Meddouri & Maddouri, 2010), authors have found that the sequential learning is beneficial for classifiers having *Decision Tree* structure such as *J48* and *Id3*. In (Meddouri, Khoufi & Maddouri, 2012), authors noticed that classifier based on *Formal Concept Analysis* is not good enough with the sequential learning on data sets of different sizes. In (Breiman, 1996a, 1996b) and (Breiman, 1999), the author has shown, theoretically and experimentally, the importance and the reliability of the parallel ensemble approach.

In the literature, stratified sampling has proved to be efficient (Ting & Witten, 1997). Learning from stratified data samples allows to generate more efficient classifier than those generated from the weighted data in the case of sequential learning classifiers. *Dagging* has the particularity to learn classifiers in parallel way from stratified data sets. We propose to exploit this variant of parallel learning method to generate classifiers based on nominal concepts. To generate T classifiers, we execute T times the learning algorithm on various disjoint and stratified sets of learning instances. Each set of learning instances is satisfied to have a similar distribution to the initial set. The samples are obtained by drawing n'_{nom} instances randomly without replacement in the training sample $\mathcal{I}_{nom}$, with $n'_{nom} < n_{nom}$. These samples respect the distribution of learning instances as classes. The principle *of Dagging Classifiers Nominal Concept* is then to take several disjoint and stratified samples $\left\{\mathcal{I}_{nom}^{\Theta_1}, \ldots, \mathcal{I}_{nom}^{\Theta_T}\right\}$. On each of which, the *Classifier Nominal Concept* is built to get a collection of classifiers $\left\{h_1, \ldots, h_T\right\}$ and to combine them by majority voting rule (Ting & Witten, 1997).

We propose to exploit the advantages of *Dagging* (kotsiantis, Anyfantis, karagiannopoulos & Pintelas, 2007) to improve the performance of proposed *Classifiers Nominal Concept*: *CpNC* and *CaNC*. Our objective here is to study the behavior of proposed nominal classifiers in parallel learning using Dagging.

EXPERIMENTAL STUDY

In this section, we are going to compare experimentally the proposed *Classifiers Nominal Concept*: *CpNC_COMV*, *CpNC_CORV*, *CaNC_COMV* and *CaNC_CORV*. To compare the proposed methods, we consider their classification error rates and training time. Further comparison will include the *Dagging* of the proposed classifiers.

We used well-known data sets from UCI Machine Learning Repository (Asuncion & Newman, 2007)). The chosen data sets contain continuous attributes. We discretize them with a WEKA[2] filter. The used filter[3] is an instance filter that converts a range of numeric attributes into nominal attributes. This transformation of data must be used by our proposed classifiers. These data sets are presented in Table 4. The Duplicated Data column present the ratio between the number of duplicated vectors of instances (attributes) and the total number of vectors in each data set. The Diversity Data column present the ratio between the number of different vectors of instances (attributes) and the total number of vectors in each data set (Haghighi, Vahedian & Yazdi, 2011).

The performance of classifiers generated is evaluated in terms of error rates. To calculate these rates, 10 Cross-validation method is used in *WEKA* whose principle is to divide each data set on 10 subsets. In turn, each subset used for testing and the other subsets for learning (Kohavi, 1995).

Table 4. Characteristics of data sets used

Data Sets		Instances	Attributes	Classes	Data Duplicated	Data Diversity
1.	Anneal	898	38	6	53,34	67,93
2.	Car	1728	6	4	0	100
3.	CMC	1473	9	3	51,6	64,83
4.	Ecoli	336	7	8	88,69	20,24
5.	Haberman	306	3	2	98,69	15,36
6.	Iris	150	4	3	95,33	16
7.	Kdd_synthetic_control	600	60	6	0	100
8.	Kr-vs-Kp	3196	36	2	0	100
9.	Lymphography	148	18	4	1,35	99,32
10.	Molecular-biology_Promoters	106	57	2	0	100
11.	Nursery	12960	8	5	0	100
12.	Page-blocks	5473	10	5	85,53	23,1
13.	Postoperative-Patient	90	8	3	18,89	88,89
14.	Sonar	208	60	2	7,69	95,19
15.	Spectrometer	531	101	48	3,01	97,36
16.	Tae	151	5	3	99,34	7,28
17.	Tic-Tac-Toe	958	9	2	0	100
18.	Vowel	990	13	11	62,63	61,21
19.	Waveform	5000	40	3	0	100
20.	Wine	178	13	3	38,76	71,91

In the next subsection, we will try to provide answers to the following questions: What is the best proposed Classifier Nominal Concept? Which closure operator it is based on? Is Dagging able to ameliorate the performance of the proposed Classifier Nominal Concept based on Formal Concept Analysis?

Comparison of Proposed Classifiers Nominal Concept

Table 5 present the error rates of the proposed classifiers nominal concept such as: *CpNC_COMV*, *CpNC_CORV*, *CaNC_COMV* and *CaNC_CORV*. As shown in Table 5, *CpNC_CORV* has the specific ability to reduce the error rates compared to the others proposed methods (average of 11.94%). These results show that *CpNC_CORV* is better than *CpNC_COMV* (average of 34.14%) and holds the best error rates for all the data sets. *CaNC_COMV* and *CaNC_CORV* produced higher error rates than the rest of the proposed classifiers (respectively average of 38.3% and 44.1%).

Table 6 present the training time for the proposed classifiers nominal concept. *CpNC_CORV* and *CpNC_COMV* are the faster compared to *CaNC_COMV* and *CaNC_CORV*. We report that *CpNC_CORV* is 4.61 times faster than *CaNC_CORV*. Also, *CaNC_COMV* is 16.67 times slower than *CpNC_CORV*.

Table 5. Error rates of proposed classifiers nominal concept

Data Sets		CpNC_CORV		CpNC_COMV		CaNC_CORV		CaNC_COMV	
		Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.
1.	Anneal	1.43	0.96	22.81	3.63	23.83	0.55	23.83	0.55
2.	Car	8.69	7.12	29.98	0.16	25.25	2.66	29.98	0.16
3.	Cmc	23.96	5.42	55.89	3.02	57.36	0.31	57.30	0.25
4.	Ecoli	7.41	3.81	34.67	4.95	57.44	1.18	53.53	3.50
5.	Haberman	14.96	4.96	24.87	5.34	18.33	4.13	26.47	1.08
6.	Iris	0.00	0.00	4.00	4.64	9.73	6.59	11.93	8.33
7.	Kdd_Synthetic_Control	9.35	3.40	42.35	4.56	59.75	4.94	34.73	4.27
8.	Kr-vs-Kp	33.95	1.72	33.95	1.72	47.75	0.13	47.75	0.13
9.	Lymphography	8.43	7.51	24.63	11.86	44.90	2.67	44.30	3.32
10.	Molecular-Biology-Promoter	13.50	8.97	27.53	10.94	50.00	3.54	27.84	14.21
11.	Nursery	13.55	2.84	29.03	0.99	55.74	3.58	51.76	1.83
12.	Page-Blocks	0.64	0.34	8.99	0.66	10.23	0.04	10.23	0.04
13.	Postoperatie-Patient-Data	19.78	9.94	27.89	6.61	28.89	5.47	28.89	5.47
14.	Sonar	4.78	4.36	27.86	9.19	30.62	10.47	26.05	8.11
15.	Spectrometer	26.25	6.33	72.30	3.35	82.88	3.43	75.72	2.54
16.	Tae	18.73	13.18	55.48	12.33	64.61	6.19	56.85	8.67
17.	Tic-Tac-Toe	9.60	2.42	30.06	4.31	34.66	0.41	34.66	0.41
18.	Vowel	13.88	2.87	65.41	3.22	84.70	3.39	72.57	4.60
19.	Waveform	7.25	1.04	43.18	1.24	54.12	2.23	36.18	3.07
20.	Wine	2.74	4.24	21.93	8.71	41.24	9.44	15.39	8.11
	Average	11.94		34.14		44.10		38.30	

Table 6. Training time (milliseconds) of proposed classifier nominal concept

Data Sets		CpNC_CORV		CpNC_COMV		CaNC_CORV		CaNC_COMV	
		Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.
1.	Anneal	2.19	5.45	**1.72**	4.91	20.16	7.79	33.28	7.25
2.	Car	1.56	4.71	**1.41**	4.49	1.72	4.91	4.38	7.05
3.	Cmc	**1.41**	4.49	**1.41**	4.49	2.66	5.90	6.41	7.72
4.	Ecoli	0.78	3.42	0.78	3.42	**0.31**	2.20	3.28	6.40
5.	Haberman	**0.00**	0.00	0.47	2.68	0.47	2.68	1.09	4.01
6.	Iris	**0.00**	0.00	0.31	2.20	0.63	3.08	0.16	1.56
7.	Kdd_Synthetic_Control	**1.72**	4.91	3.13	6.28	15.16	3.48	84.06	10.60
8.	Kr-vs-Kp	**4.53**	7.13	5.63	7.54	34.06	7.16	58.75	8.35
9.	Lymphography	**0.16**	1.56	**0.16**	1.56	0.78	3.42	2.34	5.61
10.	Molecular-Biology-Promoter	**0.31**	2.20	0.47	2.68	2.19	5.45	7.34	7.84
11.	Nursery	**12.34**	6.77	13.44	6.29	19.69	6.89	48.91	6.14
12.	Page-Blocks	**5.94**	7.62	7.34	7.84	17.50	5.10	49.22	6.43
13.	Postoperatie-Patient-Data	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.47	2.68
14.	Sonar	**0.78**	3.42	1.09	4.01	6.25	7.69	27.66	6.61
15.	Spectrometer	**4.38**	7.05	4.84	7.26	31.25	3.85	155.00	7.91
16.	Tae	0.31	2.20	**0.16**	1.56	0.47	2.68	0.94	3.73
17.	Tic-Tac-Toe	0.78	3.42	**0.63**	3.08	2.34	5.61	3.13	6.28
18.	Vowel	**1.41**	4.49	1.56	4.71	4.53	7.13	14.06	4.71
19.	Waveform	**12.66**	6.16	14.84	4.65	77.34	7.15	359.38	12.37
20.	Wine	0.63	3.08	**0.31**	2.20	1.25	4.26	3.75	6.71
	Average	**2.59**		2.98		11.94		43.18	

In our experiments, the performance of theses classifiers is not correlated with the diversity of training data. *CaNC_COMV* and *CaNC_CORV* have the particularity to generate many classification rules, at least, each one is associated to each attribute. Theirs higher error rates are due to the no-pertinent generated nominal concept.

Dagging of Proposed Classifiers Nominal Concept

To study the performance of Dagging using proposed classifiers nominal concept, we generated sets of 11 classifiers (Meddouri, Khoufi, & Maddouri, 2014). We reported their error rates in Table 7 and their training time in Table 8.

From Table 7, we report that Dagging ameliorates the error rates of *CpNC_CORV* and *CpNC_COMV* by respectively average of 3,85% and 9,96%. The error rates of *CaNC_COMV* deteriorate by average of 2,77%. The error rates of *CaNC_COpRV* is slightly improved by average of 0,77%.

In addition, we report that Dagging produces the best error rates with *CpNC_CORV* (average of 8.09%). These results show that Dagging of *CpNC_CORV* holds the best error rates for all the data sets (average of 43.33%). *Dagging CpNC_CORV* is more efficient than *Dagging CpNC_COMV* (average of 25.13%). *Dagging* of *CaNC_COMV* and *CaNC_CORV* produced the highest error rates (respectively average of 43.33% and 41.07%) compared to the rest of the classifiers. In conclusion, we can note from these experiments that parallel learning is interesting for *CpNC*, especially for *CpNC_CORV*.

Table 7. Performance of Dagging using proposed methods based on Classifier Nominal Concept

Data Sets		Dagging							
		CpNC_CORV		CpNC_COMV		CaNC_CORV		CaNC_COMV	
		Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.
1.	Anneal	6.07	5.67	16.83	3.81	21.57	2.05	21.57	2.05
2.	Car	6.06	7.47	31.31	2.97	26.21	2.25	29.98	0.16
3.	Cmc	9.99	5.65	52.59	4.59	57.11	1.11	56.99	1.24
4.	Ecoli	4.89	4.78	25.23	10.71	57.35	1.32	55.50	3.36
5.	Haberman	4.51	4.96	19.37	8.14	13.96	5.41	24.41	3.53
6.	Iris	0.67	3.73	3.13	6.93	11.27	9.32	19.93	11.93
7.	Kdd_Synthetic_Control	5.10	3.11	36.20	9.24	56.27	5.91	33.98	5.19
8.	Kr-vs-Kp	33.33	3.09	33.70	1.90	47.37	0.51	47.37	0.51
9.	Lymphography	5.85	6.51	9.29	9.85	40.18	8.09	39.58	7.75
10.	Molecular-Biology-Promoter	0.27	1.56	0.27	1.56	41.13	13.16	41.33	13.18
11.	Nursery	13.95	2.95	29.03	0.99	56.07	3.29	47.09	2.74
12.	Page-Blocks	0.77	0.94	7.90	0.94	10.23	0.05	10.23	0.04
13.	Postoperatie-Patient-Data	11.11	11.05	16.22	13.16	24.44	8.79	24.56	8.83
14.	Sonar	3.75	5.03	6.24	7.81	44.66	10.76	42.67	10.70
15.	Spectrometer	11.96	5.24	42.98	10.38	85.10	3.65	77.91	4.74
16.	Tae	7.76	9.46	19.05	12.59	60.87	10.16	58.80	9.86
17.	Tic-Tac-Toe	11.46	4.22	33.21	6.58	34.49	1.66	34.61	0.48
18.	Vowel	12.88	3.73	69.23	6.11	84.27	3.99	79.88	4.36
19.	Waveform	8.56	2.18	44.90	2.10	57.81	2.98	36.23	3.01
20.	Wine	2.77	4.91	5.98	8.13	36.26	11.88	38.78	10.31
	Average	8.09		25.13		43.33		41.07	

Table 8 presents the training time of *Dagging* for the proposed classifiers nominal concept. *Dagging* of *CpNC_CORV* and *CpNC_COMV* (respectively average of 18.16 ms and 18.84 ms) are the faster compared to the *Dagging* of *CaNC_COMV* and *CaNC_CORV* (respectively average of 27.57 ms and 64.88 ms). We report that *Dagging CpNC_CORV* is 1.51 times faster than *Dagging CaNC_CORV*. Also, *Dagging CaNC_COMV* is 3.57 times slower than *Dagging CpNC_CORV*.

Similarly, to the previous experiments, the performance of these classifier ensembles is not correlated with the diversity of training data. In conclusion, the diversity of data is not correlated with the performance of classifiers nominal concept based on *Formal Concept Analysis*.

Comparison With State-of-the-Art Classification Methods

Table 9 present the error rates of different classification methods from the literature (*Bayes Net*, *Naive Bayes, SVM*[4], *IB1*[5], *Decision Stump, Hoeffding Tree, C4.5*[6], *Random Forest* and *Random Tree*). As shown in this table, *Dagging CpNC_CORV* has the specific ability to reduce the error rates with the less error rate for 12 data sets. *Dagging CpNC_CORV* has the best error rates for most of the data sets (average of 8.09%), then *Random Forest* (average of 14.54%) compared to *Decision Stump* (average of 40.08%).

Table 8. Training time (milliseconds) of Dagging using proposed methods

Data Sets		Dagging							
		CpNC_CORV		CpNC_COMV		CaNC_CORV		CaNC_COMV	
		Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.
1.	Anneal	**16.88**	5.29	17.03	5.48	33.75	6.17	53.13	8.01
2.	Car	**15.31**	4.43	**15.31**	4.43	14.53	4.58	17.81	5.45
3.	Cmc	**14.53**	5.09	15.47	4.71	15.47	3.51	20.47	7.91
4.	Ecoli	**13.59**	5.28	13.75	5.10	14.22	4.49	15.78	3.51
5.	Haberman	**12.81**	6.03	13.75	5.57	**12.81**	6.03	13.75	5.10
6.	Iris	**13.13**	5.76	14.06	5.21	14.06	5.21	13.28	6.03
7.	Kdd_Synthetic_Control	**21.09**	7.49	23.28	7.85	35.78	7.79	131.41	12.38
8.	Kr-vs-Kp	**19.38**	6.71	20.63	7.65	53.13	8.31	80.63	7.60
9.	Lymphography	**13.13**	6.56	14.53	4.01	13.91	5.39	15.63	4.97
10.	Molecular-Biology-Promoter	**14.22**	5.01	16.09	6.06	18.13	6.17	32.03	5.15
11.	Nursery	**26.25**	8.85	26.41	8.80	35.63	7.72	59.22	8.10
12.	Page-Blocks	**21.72**	7.66	21.88	7.69	32.34	5.09	62.81	5.87
13.	Postoperatie-Patient-Data	13.28	5.61	**13.13**	6.17	14.38	6.91	13.28	5.61
14.	Sonar	**18.75**	7.02	20.00	7.05	26.56	7.20	68.44	8.54
15.	Spectrometer	39.22	8.74	**38.75**	8.15	59.69	7.81	239.38	9.65
16.	Tae	13.59	6.14	13.44	5.45	**13.28**	5.61	13.75	5.10
17.	Tic-Tac-Toe	**13.91**	5.39	15.63	3.14	15.31	3.13	16.88	5.29
18.	Vowel	**16.09**	3.48	17.34	5.39	18.91	6.40	27.97	6.40
19.	Waveform	31.72	3.48	**31.25**	6.66	93.75	8.60	383.91	19.14
20.	Wine	**14.53**	5.09	15.16	4.13	15.78	5.66	17.97	7.15
	Average	**18.16**		18.84		27.57		64.88	

Table 10 present the training time (in milliseconds) of different classification methods cited previously, compared to the *Dagging* of *CpNC_CORV*. As shown in this table, *IB1* has the best training time (average of 0.15 ms), then *Naive Bayes* (average of 2.07 ms) compared to *SVM* (average of 636.38 ms). As shown in Table 9, *IB1*, *Naive Bayes* and *SVM* didn't have the best error rates (respectively average of 17.74%, 22.09% and 17.44%). *Dagging CpNC_CORV*, *Random Forest* and *SVM* holds the best error rates (respectively average of 8.09%, 14.54% and 17.44%). *Dagging CpNC_CORV* is 17.13 times faster than *Random Forest* and 36.17 times faster than *SVM*. However, *Dagging CpNC_CORV* is slower than *IB1*, *Naive Bayes* and *Decision Stump* (which have the worst error rates).

CONCLUSION

Formal Concept Analysis is an interesting formalism to study machine learning and classification methods. It allows a full construction of the concepts and the dependence relationships between concepts in order to build a lattice of formal concepts. Many classification methods based on exhaustive or combinatory approach exists in the literature of classification based on Formal Concept Analysis. We have presented a learning method: Dagging classifiers based on nominal concepts. Our method

Table 9. Error rates of classification methods

Data Sets		Dagging CpNC_CORV		Bayes Net		Naive Bayes		SVM		IB1		Decision Stump		Hoeffding Tree		J48		Random Forest		Random Tree	
		Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.	Err.	Dev.
1.	Anneal	6,07	5,67	4,05	2,19	13,41	3,31	2,54	1,6	0,87	1,06	22,83	3,66	23,83	0,55	1,43	1,04	**0,41**	0,7	1,65	1,57
2.	Car	6,06	7,47	14,39	2,56	14,54	2,56	6,38	1,93	6,95	1,59	29,98	0,16	14,54	2,56	7,78	2,01	**5,33**	1,74	16,04	3,51
3.	Cmc	**9,99**	5,65	49,98	4,02	49,52	4,22	51,34	4,18	56,04	3,58	57,3	0,25	50,35	4,24	48,56	3,88	49,56	3,89	53,58	3,92
4.	Ecoli	**4,89**	4,78	18,45	4,48	14,5	5,46	16,52	5,17	19,34	6,16	35,38	1,94	16,19	5,94	17,17	5,73	14,31	5,15	21,06	6,6
5.	Haberman	**4,51**	4,96	28,07	4,96	24,64	5,4	27,03	2	32,5	6,59	28,43	3,95	25,95	4,34	27,84	4,7	31,85	6,29	34,4	7,22
6.	Iris	**0,67**	3,73	6,8	5,92	4,47	5,02	3,73	4,58	4,6	4,8	33,33	0	4,73	5,47	5,27	5,3	5,33	5,01	6,73	4,97
7.	Kdd_Synthetic	5,1	3,11	3,12	2,21	5,45	2,56	**0,93**	1,19	3	2,25	66,73	0,33	5,45	2,56	7,48	3,3	1,5	1,62	10,87	4,47
8.	Kr-vs-Kp	33,33	3,09	12,19	1,9	12,21	1,91	4,21	1,34	3,88	1,12	33,95	1,72	6,32	1,5	**0,56**	0,37	0,83	0,48	3,85	1,64
9.	Lymphography	**5,85**	6,51	14,36	8,17	16,87	8,89	13,52	7,68	18,31	8,73	24,69	11,34	24,29	11,01	24,16	11,05	16,99	9,46	26,82	11,17
10.	Molecular-Biology	**0,27**	1,56	8,97	9,2	8,46	9,09	7,77	7,49	19,41	10,75	29,52	8,81	8,46	9,09	19,2	11,51	8,61	8,02	33,06	14,85
11.	Nursery	13,95	2,95	9,69	0,73	9,7	0,72	6,92	0,62	1,61	0,32	33,75	0,04	5,7	0,6	2,82	0,46	**0,94**	0,32	5,92	1,31
12.	Page-Blocks	**0,77**	0,94	6,46	1,02	9,99	2,08	7,16	0,66	3,92	0,74	6,87	0,7	9,94	1,65	3,01	0,6	2,47	0,64	3,66	0,79
13.	Postoperatie-Data	**11,11**	11,05	34,11	9,77	31,89	8,15	32,67	8,02	37,56	9,31	29,89	7,01	29,22	6,05	30,22	6,9	38,11	10,51	43,67	12,85
14.	Sonar	**3,75**	5,03	23,29	9,4	32,29	8,66	23,4	8,27	13,83	8,45	27,75	9,44	31,79	8,95	26,39	9,34	16,9	8,66	27,74	9,95
15.	Spectrometer	**11,96**	5,24	53,65	6,23	57,94	5,99	52,46	5,67	62,45	6,02	81,36	1,22	58,09	6	52,5	5,69	43,11	5,63	59,34	6,37
16.	Tae	**7,76**	9,46	53,68	7,74	45,99	11,02	46,4	11,86	37,11	12,81	62,02	9,07	46,13	10,96	42,59	12,64	32,46	11,35	33,87	11,49
17.	Tic-Tac-Toe	11,46	4,22	30,41	4,39	30,36	4,4	1,67	1,28	**1,02**	0,98	30,06	4,31	30,36	4,4	14,72	3,18	3,22	1,94	20,17	4,94
18.	Vowel	12,88	3,73	41,24	5,46	37,1	4,38	29,39	3,86	**0,95**	1,04	82,53	0,82	37,13	4,39	19,8	4,36	1,9	1,54	17,01	3,98
19.	Waveform	**8,56**	2,18	20,04	1,43	19,99	1,45	13,52	1,53	26,59	1,82	43,18	1,25	20	1,43	24,75	1,9	14,84	1,47	27,41	1,91
20.	Wine	2,77	4,91	1,35	2,56	2,54	3,7	**1,24**	2,73	4,88	4,34	42,09	6,09	2,48	3,77	6,8	5,9	2,14	3,17	7,82	6,4
	Average	**8.09**		21.72		22.09		17.44		17.74		40.08		22.55		19.15		14.54		22.73	

Table 10. Training time (milliseconds) of classification methods

Data Sets		Dagging CpNC_ CORV		Bayes Net		Naive Bayes		SVM		IB1		Decision Stump		Hoeffding Tree		J48		Random Forest		Random Tree	
		Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.	Time	Dev.
1.	Anneal	16,56	5,36	2,97	6,16	0,78	3,42	57,81	8,74	**0,16**	1,56	0,78	3,42	27,81	6,87	10,31	8,07	64,53	7,9	0,47	2,68
2.	Car	14,53	5,55	0,47	2,68	**0,16**	1,56	104,38	11,3	0,47	2,68	0,47	2,68	4,22	6,97	1,72	4,91	73,28	8,51	0,47	2,68
3.	Cmc	14,38	4,8	1,41	4,49	0,31	2,2	223,44	16,54	**~0**	~0	0,94	3,73	5,16	7,38	11,88	7,06	245,94	11,46	3,59	6,61
4.	Ecoli	14,38	4,26	0,63	3,08	**~0**	~0	46,09	7,15	**~0**	~0	0,31	2,2	7,81	7,85	1,09	4,01	46,56	5,87	0,78	3,42
5.	Haberman	13,59	5,73	0,47	2,68	**~0**	~0	5,63	7,54	**~0**	~0	**~0**	~0	**~0**	~0	0,31	2,2	23,59	7,85	0,47	2,68
6.	Iris	13,59	5,73	**~0**	~0	**~0**	~0	5,16	7,38	**~0**	~0	**~0**	~0	0,63	3,08	**~0**	~0	9,69	7,62	**~0**	~0
7.	Kdd_Synthetic	20,78	8,33	15	4,4	2,81	6,03	35,63	8,91	**0,16**	1,56	5,63	7,54	59,22	6,77	25,47	7,9	171,41	7,52	3,13	6,28
8.	Kr-vs-Kp	18,44	6,03	6,09	7,66	0,94	3,73	606,72	58,13	**0,16**	1,56	1,41	4,49	26,09	7,38	11,88	8,04	272,19	20,97	3,13	6,28
9.	Lymphography	13,28	6,03	0,31	2,2	0,31	2,2	13,91	6,62	0,16	1,56	**~0**	~0	2,19	5,45	0,47	2,68	13,28	5,61	0,31	2,2
10.	Molecular-Biology	13,28	5,61	0,16	1,56	**~0**	~0	7,34	7,84	0,16	1,56	0,16	1,56	1,41	4,49	0,16	1,56	9,84	7,58	0,16	1,56
11.	Nursery	25,16	7,97	5,31	7,44	1,41	4,49	9010,31	461,6	**0,94**	3,73	1,88	5,1	55,94	7,75	16,25	4,93	623,28	35,46	5,31	7,44
12.	Page-Blocks	20	7,05	16,41	4,08	5,78	7,58	74,22	11,83	**0,31**	2,2	7,66	7,85	64,84	8,71	70,47	10,3	1234,22	40,07	20,31	7,2
13.	Postoperatie-Data	12,03	6,61	**~0**	~0	**~0**	~0	5,63	7,54	**~0**	~0	0,16	1,56	0,47	2,68	0,63	3,08	10,16	7,49	0,31	2,2
14.	Sonar	19,84	6,97	2,34	5,61	1,41	4,49	5,47	7,49	**~0**	~0	1,25	4,26	5,31	7,44	6,09	7,66	45,47	5,01	0,78	3,42
15.	Spectrometer	36,72	8,42	50,94	6,89	5,63	7,54	1832,97	107,15	**~0**	~0	21,88	7,69	1528,91	37,71	98,91	8,02	600,47	24,05	10,31	7,44
16.	Tae	12,5	6,28	0,16	1,56	**~0**	~0	6,72	7,77	0,16	1,56	**~0**	~0	0,16	1,56	0,78	3,42	23,59	7,85	0,16	1,56
17.	Tic-Tac-Toe	13,28	5,61	0,16	1,56	**~0**	~0	103,28	17,61	**~0**	~0	0,16	1,56	2,03	5,28	1,72	4,91	57,66	7,59	0,94	3,73
18.	Vowel	15,47	3,51	4,38	7,05	1,09	4,01	264,84	11,41	**~0**	~0	2,03	5,28	27,97	6,77	15	4,4	236,41	8,21	3,91	6,8
19.	Waveform	30	6,91	42,81	8,2	20,47	7,26	311,72	47,08	**0,31**	2,2	26,88	8,62	129,53	10,49	268,91	13,92	2246,25	43,3	36,25	7,33
20.	Wine	13,91	5,83	**~0**	~0	0,31	2,2	6,41	7,72	**~0**	~0	0,16	1,56	1,88	5,1	0,78	3,42	18,91	6,4	0,63	3,08
	Average	17.59		7.5		2.07		636.38		**0.15**		3.59		97.58		27.14		301.34		4.57	

has the particularity to construct a classification rule form the closure operator of a selected attribute. The proposed method suggests enlarging the application fields of Formal Concept Analysis on a type of data other than the binary one (i.e nominal).

In this paper, we have proposed 4 variants of our method. The *CpNC* which calculate only the closure of the pertinent attribute. The *CaNC* which consider all nominal attributes and calculates the closure associates to each one. *CpNC* and *CaNC* consider the whole of training instances and use nominal attributes. Each one has two ways to use the closure operator. Since a nominal attribute has many nominal values, we proposed the *CpNC_CORV* which is a classifier of pertinent nominal concept based on closure operator for the relevant values of the pertinent nominal attribute. Furthermore, *CpNC_COMV* has the particularity to consider a closure operator for multi-values of the pertinent nominal attribute. Besides, we have proposed the *CpNC_CORV* and *CpNC_COMV* which have the particularity to consider all the nominal concepts. The experimental results have shown that *CpNC_CORV* reduces the error rates compared to *CpNC_COMV*, *CaNC_CORV* and *CaNC_COMV*.

To improve the performances of our proposed approach, we use ensemble methods. Previous researches recommend a parallel learning by *Dagging* for classifiers based on *Formal Concept Analysis* instead of the sequential learning. We have presented a variant of *Dagging* to generate ensemble of classifiers based on Formal Concept Analysis. In parallel learning, few classifiers are enough to reach better performance. We have made an experimental study to show the interest of the *Dagging* of the proposed method by using known data sets. *Dagging* of *CpNC_CORV* reached good precision compared to known methods like *J48* and *SVM*. We report that even in *Dagging CpNC_CORV* is better than other variants. Accordingly, we recommend using the closure operator on the relevant value of the pertinent attribute.

More experiments are possible on larger data sets with other ensemble methods, such as *Random Forests*. Many improvements on the ensemble methods can be brought. *DNC* has adopted majority vote for classifier combination. A variety of voting rules already exist. Hence, a study of these rules can be beneficial to improve the performance of our method.

REFERENCES

Asuncion, A., & Newman, D. J. (2007). UCI machine learning repository.

Blinova, V., Dobrynin, D. A., Finn, V., Kuznetsov, S., & Pankratova, E. (2003). Toxicology analysis by means of the JSM-method. *Bioinformatics (Oxford, England)*, *19*(07), 1201–1207. doi:10.1093/bioinformatics/btg096 PMID:12835262

Breiman, L. (1996). Bagging predictors. *Journal of Machine Learning*, *24*(2), 123–140. doi:10.1007/BF00058655

Breiman, L. (1996). *Bias, variance, and arcing classifiers. Technical report*. University of California.

Breiman, L. (1999). Prediction games and arcing algorithms. *Journal of Neural Computation*, *11*(7), 1493–1517. doi:10.1162/089976699300016106 PMID:10490934

Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S. O., Napoli, A., & Raissi, C. (2016). On mining complex sequential data by means of FCA and Pattern Structures. *International Journal of General Systems*, *45*(2), 135–159. doi: 10.1080/03081079.2015.1072925

Douar, B., Latiri, C. C., & Slimani, Y. (2008). Approche hybride de classification supervisee a base de treillis de galois: application a la reconnaissance de visages. In Actes des 8mes Journes Francophones en Extraction et Gestion des Connaissances, volume E-11 of Revue des Nouvelles Technologies de l'Information (pp. 309-320). Cepadues-Editions.

Ferrandin, M., Nievola, J. C., Enembreck, F., Scalabrin, E. E., Kredens, K. V., & Avila, B. C. (2013). Hierarchical classification using fca and the cosine similarity function. In *Proceedings on the International Conference on Artificial Intelligence* (pp. 1-7). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Ikeda, M., & Yamamoto, A. (2013). Classification by selecting plausible formal concepts in a concept lattice. In *Workshop on Formal Concept Analysis meets Information Retrieval* (pp. 22-35).

Kashnitsky, Y., & Ignatov, D. (2014). Can FCA-based recommender system suggest a proper classifier? *CEUR Workshop Proceedings*, *1257*, 17–26.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *In Proceedings of the 14th international joint conference on Artificial Intelligence* (pp 1137-1143). Morgan Kaufmann.

Kotsianti, S. B., & Kanellopoulos, D. (2007). Combining bagging, boosting and dagging for classification problems. In *Proceedings of the 11th International Conference on Knowledge-Based Intelligent Information and Engineering Systems* (pp. 493-500). Springer. doi:10.1007/978-3-540-74827-4_62

Kotsiantis, S. B., Anyfantis, D., Karagiannopoulos, M., & Pintelas, P. E. (2007). Local dagging of decision stumps for regression and classification problems. In *Proceedings of 15th IEEE Mediterranean Conference on Control and Automation* (pp. 1-6). Academic Press.

Kuncheva, L., Skurichina, M., & Duin, R. P. W. (2002). An experimental study on diversity for bagging and boosting with linear classifiers. *Journal of Information Fusion*, *3*(4), 245–258. doi:10.1016/S1566-2535(02)00093-3

Kuznetsov, S. O. (2004). Fitting pattern structures to knowledge discovery in big data. In *International Conference on Formal Concept Analysis* (pp. 254-266). Springer.

Maddouri, M. (2004). Towards a machine learning approach based on incremental concept formation. *Journal of Intelligent Data Analysis*, *8*(3), 267–280. doi:10.3233/IDA-2004-8304

Meddouri, N., N & Maddouri, M. (2009). Boosting formal concepts to discover classification rules. In *Proceeding of the 22rd International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems* (pp. 501-510). Springer. doi:10.1007/978-3-642-02568-6_51

Meddouri, N., Khoufi, H., & Maddouri, M. (2012). Diversity analysis on boosting nominal concepts. In *Proceedings of Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining* (Part I, pp. 306-317). Springer.

Meddouri, N., Khoufi, H., & Maddouri, M. (2014). Parallel learning and classification for rules based on formal concepts. *Procedia Computer Science*, *35*, 358-367.

Meddouri, N., & Maddouri, M. (2009). Classification methods based on formal concept analysis. In *Proceedings of the 6th International Conference on Concept Lattices and Their Applications* (pp 9-16). Academic Press.

Meddouri, N., & Maddouri, M. (2010). Adaptive learning of nominal concepts for supervised classification. In *Proceedings of the 14th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems* (pp. 121-130). Springer. doi:10.1007/978-3-642-15387-7_16

Melville, P., & Mooney, R. J. (2005). Creating diversity in ensembles using artificial data. *Journal of Information Fusion: Special Issue on Diversity in Multi Classifier Systems*, *6*(1), 99–111. doi:10.1016/j.inffus.2004.04.001

Salkhordeh Haghighi, M., Vahedian, A., & Sadoghi Yazdi, H. (2011). Creating and measuring diversity in multiple classifier systems using support vector data description. *Applied Soft Computing*, *11*(8), 4931–4942. doi:10.1016/j.asoc.2011.06.006

Skurichina, M., & Duin, R. P. W. (1998). Bagging for linear classifiers. *Journal of Pattern Recognition*, *31*(7), 909–930. doi:10.1016/S0031-3203(97)00110-6

Stumme, G., Ganter, B., & Wille, R. (2005). *Formal Concept Analysis: Foundations and Applications*. Springer.

Ting, K. M., & Witten, I. H. (1997). Stacking bagged and dagged models. In *Proceedings of 14th International Conference on Machine Learning* (pp. 367-375). Academic Press.

Trabelsi, M., Meddouri, N., & Maddouri, M. (2016) New taxonomy of classification methods based on formal concepts analysis. In *Proceedings of the 5th International Workshop "What can FCA do for Artificial Intelligence"?* (pp. 113-120). Academic Press.

Tsopze, N., Mephu Nguifo, E., & Tindo, G. (2007). Clann: Concept latticebased artificial neural network for supervised classification. In *Proceedings of the 5th International Conference on Concept Lattices and Their Applications* (pp. 812-821). Springer.

Visani, M., Bertet, K., & Ogier, J. M. (2011). Navigala: An original symbol classifier based on navigation through a Galois Lattice. *International Journal of Pattern Recognition and Artificial Intelligence*, *25*(4), 449–473. doi:10.1142/S0218001411008634

ENDNOTES

1 The data sets are selected from local copy of the sample Weka databases at http://storm.cis.fordham.edu/~gweiss/data-mining/datasets.html

2 Available at http://www.cs.waikato.ac.nz/ml/Weka

3 weka.filters.unsupervised.attribute.Discretize

4 In this work, the *SMO* module of *WEKA* with a default parameter setting is used to perform classification via the *SVM*

5 In this work, the *IBk* module of *WEKA* with default parameter settings is used to perform classification via the *Nearst-neighbour* classifier

6 In this work, the *J48* module of *WEKA* with a default parameter setting is used to perform classification via the *C4.5*

Nida Meddouri obtained her PhD in Computer Science at the University of Tunis El Manar in 2015. In 2019, he was hired as a contractual teaching and research associate at the University of Caen Normandy and is currently a senior researcher at GREYC (Normandie Univ, UNICAEN, ENSICAEN, CNRS - UMR GREYC, 14000 Caen, France). He writes and presents numerous publications on machine learning based on formal concept analysis and ensemble methods, and is the author of Learning Sets of Classifications Rules by Formal Concept Analysis (2015, Faculty of Mathematical, Physical and Natural Sciences of Tunis, University of Tunis El Manar, Tunisia).