A First Study on What MDL Can Do for FCA

Tatiana Makhalova[1,2], Sergei O. Kuznetsov[1], and Amedeo Napoli[2]

[1] National Research University Higher School of Economics,
3 Kochnovsky Proezd, Moscow, Russia
[2] LORIA, (CNRS – Inria – U. of Lorraine), BP 239
Vandœuvre-lès-Nancy, France
tpmakhalova@hse.ru, skuznetsov@hse.ru, amedeo.napoil@loria.fr

Abstract. Formal Concept Analysis can be considered as a classification engine able to build classes of objects with a description or concepts and to organize these concepts within a concept lattice. The concept lattice can be navigated for selecting significant concepts. Then the problem of finding significant concepts among the potential exponential number of concepts arises. Some measures exist that can be used for focusing on interesting concepts such as support, stability, and other. MDL (minimum description length) is also a good candidate that was rarely used in FCA by now for such objective. In this paper we argue that MDL can give FCA practitioners a good measure for selecting significant and representative concepts.

1 Introduction

Formal concept analysis (FCA) plays an important role in Data Mining and Machine Learning. Concept lattices support mainly unsupervised settings, improving tasks such as building taxonomies and ontologies, computing implications and association rules, clustering and solving classification tasks. These tasks in practice are coupled with the problem of exponential explosion of the number of formal concepts.

By now, to tackle this issue, a lot of different approaches have been proposed, including data pre- and postprocessing, background knowledge incorporation, computing approximate concepts (see [9] for an overview). As a result, one expects to get a small set of interesting, meaningful, non-redundant concepts [3].

In this paper, we focus on the characterization of such a small set of concepts. Instead of using an interesting measure in postprocessing step, we propose to rely on the minimum description length (MDL) principle [7], which allows one to select small sets of diverse and interpretable concepts. Providing the best lossless compression of the data, the MDL optimal sets of patterns (itemsets) automatically provides a balance between the quality of fit of the data and the complexity of the model without any user-defined parameters to be set [1].

In this paper we propose a first study on the application of the MDL principle in FCA settings. To the best of our knowledge, this is one of the first papers to study the effective use of MDL in the framework of FCA.

ISBN 978–80–244–5328–6, Department of Computer Science, Palacký University Olomouc, 2018.

The rest of the paper has the following structure. In Section 2 we remind the basic definition used in FCA and discuss how FCA can be used to solve classification problem. Section 3 introduces the MDL principle. In Section 4 we discuss what MDL can bring to FCA. Section 5 gives conclusion and directions for future work.

2 Preliminaries

In this section we recall the main notions that are used in the paper. We study attribute sets of formal concepts, which have several alternative names, such as itemset, intent, or pattern. We discuss the application of FCA in unsupervised and supervised settings.

2.1 FCA. Basic Notions

Here we briefly recall FCA terminology [5]. A formal context is a triple (G, M, I), where $G = \{g_1, g_2, ..., g_n\}$ is called a set objects, $M = \{m_1, m_2, \dots, m_k\}$ is called a set attributes and $I \subseteq G \times M$ is a relation called incidence relation, i.e. $(g, m) \in I$ if the object g has the attribute m. The derivation operators $(\cdot)'$ are defined for $A \subseteq G$ and $B \subseteq M$ as follows:

$$A' = \{m \in M \mid \forall g \in A : gIm\}$$
$$B' = \{g \in G \mid \forall m \in B : gIm\}$$

A' is the set of attributes common to all objects of A and B' is the set of objects sharing all attributes of B. An object g is said to contain a pattern (set of items) $B \subseteq M$ if $B \subseteq g'$. The double application of $(\cdot)'$ is a closure operator, i.e. $(\cdot)''$ is extensive, idempotent and monotone. Sets $A \subseteq G$, $B \subseteq M$, such that $A = A''$ and $B = B''$, are said to be closed.

A (formal) concept is a pair (A, B), where $A \subseteq G$, $B \subseteq M$ and $A' = B$, $B' = A$. A is called the (formal) extent and B is called the (formal) intent of the concept (A, B). A partial order $\leqslant$ is defined on the set of concepts as follows: $(A, B) \leq (C, D)$ iff $A \subseteq C$ $(D \subseteq B)$, a pair (A, B) is a subconcept of (C, D), while (C, D) is a superconcept of (A, B).

The number of formal concepts grows exponentially w.r.t. the size of a formal context, i.e. the number of objects in G and attributes in M. Thus, it becomes almost impossible to analyze and interpret the whole set of generated concepts. *Pattern discovery* techniques are designed to solve this problem. The goal of pattern discovery within the framework of FCA is to find a non-redundant set of concepts that are interesting w.r.t. specified constrains/interestingness criterion. The criterion can be applied to both intent and extent, whereas pattern discovery in general is related solely to the itemset assessment.

Example. Let us consider the toy example given in Table 1. We will use either D_1 or D_2 to compute classifiers and the remaining objects will be used to assess the quality of the classifiers. The set of attributes M includes columns $m_1, \dots, m_9$.

Table 1. An example of dataset.

data partitions		objects	m_1 : **4 legs**	m_2 : **hairs**	m_3 : **change size**	m_4 : **cold-resistant**	m_5 :**do not release** CO_2	m_6 : **black-white**	m_7 : **yellow-brown**	m_8 : **green**	m_9 : **gray**	**target** w: **animal**
D_2	D_1	g_1 dog	×	×	×			×				+
		g_2 cat	×	×	×				×			+
		g_3 frog	×		×					×		+
		g_4 car				×					×	−
		g_5 ball			×	×	×	×				−
		g_6 chair	×			×	×			×		−
		g_7 fur coat		×		×	×				×	−
		g_8 sunflower			×				×			−
		g_9 fish			×	×					×	+
		g_{10} leopard	×	×	×				×			+
		g_{11} table	×			×	×			×		−

The additional attribute *"target"*, i.e., class labels, is not taken into account under unsupervised settings.

Filtering concepts based on their extent and/or intent belongs to class of unsupervised problems, since any supplemental information is unavailable. In the next subsection we consider the problem of concept selection in supervised settings.

2.2 FCA under Supervised Settings: Concept-based Classifiers

In supervised settings along with the objects and their descriptions an additional attribute w is given. It specifies the class of an object. We denote the set of its values by ε.

We shall confine ourselves to two-class classification and study the simplest case, where each object belongs to a single class. In the defined settings $\varepsilon = \{+, -\}$. We consider the case where a set of objects $G = G_+ \cup G_-$ is divided into 2 disjoint subsets, i.e., a set of positive examples G_+, negative examples G_-. In practice, a set G_τ of unlabeled examples appears as well. The objects are described by attributes from M and the target attribute w is defined as follows: $gIw =$ " $+$ " for $g \in G_+$ and $gIw =$ " $-$ " for $g \in G_-$. The objects from $G_+ \cup G_-$ compose *training and test sets*, which are used to generate concepts and to estimate quality of classifiers, respectively.

It is assumed that there exists an unknown function that maps each object $g \in G$ (or its description $g' \subseteq M$) to an element in ε. The goal is to *reconstruct as accurately as possible the unknown function* using an observable subset of labeled objects. To do that, one builds classifiers, which can be constructed by means of FCA as follows.

For each concept (A, B) a class label $e \in \varepsilon$ is defined by majority of labeled objects in the extent, i.e. $class((A, B)) = \arg\max_{e \in \varepsilon} |A_e|/|A|$, where $A_e = G_e \cap A$. To classify an unlabeled object g w.r.t. a pair (B, e) we set the following *classification principle*:

$$(B, e)(g) = \begin{cases} e, & \text{if } B \subseteq g' \\ \emptyset, & \text{otherwise.} \end{cases} \tag{1}$$

Table 2. The values of quality measures (Formulas 2-5) for classifiers $\{$*"cold-resistant"*$\}^-$ and $\{$*"4 legs"*, *"change size"*$\}^+$ from the running example in Table 1. The attribute sets of the classifiers are intents of formal concepts computed on D_1. The set $\{g_8, g_9, g_{10}, g_{11}\}$ is used to assess classifiers.

classifier/ measure	$\{$*"cold-resistant"*$\}^-$	$\{$*"4 legs"*, *"change size"*$\}^+$
prec	$\lvert\{g_{11}\}\rvert/\lvert\{g_9, g_{11}\}\rvert = 1/2$	$\lvert\{g_{10}\}\rvert/\lvert\{g_{10}\}\rvert = 1$
recall	$\lvert\{g_{11}\}\rvert/\lvert\{g_8, g_{11}\}\rvert = 1/2$	$\lvert\{g_{10}\}\rvert/\lvert\{g_9, g_{10}\}\rvert = 1/2$
sup	$\lvert\{g_9, g_{11}\}\rvert/\lvert G^*\rvert = 1/2$	$\lvert\{g_{10}\}\rvert/\lvert G^*\rvert = 1/4$
acc	$\lvert\{g_{10}, g_{11}\}\rvert/\lvert G^*\rvert = 1/2$	$\lvert\{g_{10}\}\rvert/\lvert G^*\rvert = 1/4$

According to Formula 1, we get a non-empty response e from (B, e) if an object description g' contains attribute set B. To simplify notation we will write $\hat{B}^e$ instead of (B, e). In general, B could be any itemset, not necessarily closed.

The details on classification problem in terms of FCA can be found in [6, 8].

To identify the best classifier a test set $G^* \subseteq G_+ \cup G_-$ is used, we will write G^*_+ and G^*_- for subsets of positive and negative objects in the test set G^*, i.e., for $G^* \cap G_+$ and $G^* \cap G_-$, respectively. In our paper we estimate classifiers using the measures listed below and provide small examples of their usage (see Table 2). Precision measures how many correct answers are given by the classifier:

$$\text{prec}(\hat{B}^e, G^*) = \left|\left\{g \mid \hat{B}^e(g) = e, g \in G^*_e\right\}\right| / \left|\left\{g \mid \hat{B}^e(g) = e, g \in G^*\right\}\right|. \quad (2)$$

Recall measures how many objects from a target class are characterized by the classifier (i.e. whether a classifier is specific or general):

$$\text{recall}(\hat{B}^e, G^*) = \left|\left\{g \mid \hat{B}^e(g) = e, g \in G^*_e\right\}\right| / |G^*_e|. \quad (3)$$

Support measures how many objects can be classified (correctly or not):

$$\sup(\hat{B}^e, G^*) = \left|\left\{g \mid \hat{B}^e(g) = e, g \in G^*\right\}\right| / |G^*|. \quad (4)$$

The accuracy takes into account examples from the remaining classes $\varepsilon \setminus \{e\}$ unclassified by $\hat{B}^e$:

$$\text{acc}(\hat{B}^e, G^*) = \frac{\left|\left\{g \mid \hat{B}^e(g) = e, g \in G^*_e\right\} \cup \left\{g \mid \hat{B}^e(g) = \emptyset, g \in G^*_c, c \in \varepsilon \setminus \{e\}\right\}\right|}{|G^*|}. \quad (5)$$

However, other measures can also be examined [2], e.g., F1 score, AUC, etc.

In the next section we consider an ensemble of classifiers based on single concept-based classifiers defined in Formula 1.

2.3 Concept-based Classifiers

A set of the classifiers $\mathcal{S} = \left\{\hat{B}^e_j\right\}_{j \in J}$, $e \in \varepsilon$ with a rule for aggregation of their responses constitute an *ensemble of classifiers*. We call an ensemble *concept-based classifiers* (CBC) if the itemsets B are intents of formal concepts. As an aggregation rule the following principles might be chosen: class of the majority

of labels, classification if responses are the same class labels, the highest priority class from the set of responses.The best CBCs are those that ensure high accuracy and have a small size of $\mathcal{S}$.

Example. Let us turn back to the running example from Table 1. The supplementary information on class labels (the target attribute w) is given in column *"target"*. We consider the following set of classifiers:

$$\mathcal{S} = \{(\{\textit{"cold-resistant"}\}, -),\ (\{\textit{"4 legs"},\ \textit{"change size"}\}, +)\}.$$

As a rule for aggregation of responses we use the priority principle, let us suppose that "–" class has higher priority than "+", thus we can apply classifiers from "–" class and, if the object remains unclassified, we try to classify it with "+" class classifiers.

CBC works for g_{10} as follows. A classifiers with the highest priority ("–" class) are firstly applied. Since $\{$*"cold-resistant"*$\} \not\subseteq g'_{10}$, we get an empty response and turn to the classifiers of lower priority ("+" class). $\{$*"4 legs", "change size"*$\} \subseteq g'_{10}$, we get "+" response and classify g_{10} as an member of "+" class. The object g_{11} is classified as "–", since $\{$*"cold-resistant"*$\} \subseteq g'_{11}$. g_8 remains unclassified, since we get empty responses from all classifiers in $\mathcal{S}$, g_9 is misclassified by ($\{$*"cold-resistant"*$\}$, –).

Thus, a formal concept is a well-interpreted, quite intuitive and handy tool for describing subsets of objects both un- and supervised settings. However, as it was mentioned earlier, the huge number of generated concepts hampers interpretation of the results as well as its practical application.

In the next section we provide an approach that can be used to select a small set of diverse concepts.

3 Minimal Description Length Principle

3.1 Describing Data with MDL . Unsupervised Settings

The MDL principle in the context of pattern mining is formulated as follows: the best set of patterns is the set that best compresses the database [10].

The main element of this approach is a code table (CT), which is composed of "some" itemsets with their length. The best code table minimizes the total length in bits $L(D, CT) = L(D \mid CT) + L(CT \mid D)$, where $L(D \mid CT)$ is the length of the dataset encoded with the code table CT and $L(CT \mid D)$ is the length of the code table CT computed w.r.t. D. To encode an object g in a dataset one needs to select a subset of disjoint itemsets that cover all attributes of g. By $u(B) = |\{t \in D \mid B \in cover(t)\}|$ we denote the usage of itemset B in dataset D, i.e., how many times B is used to cover objects in D, where $U = \sum_{B \in CT} u(B)$ is the total usage of itemsets. The principles of building code tables will be discussed further.

To define the length of an itemset we use an optimal prefix code given by Shannon entropy, i.e., $l(B) = -logPr(B)$, where probability is defined as follows:

$Pr(B) = u(B)/U$. Thus, the itemsets with higher frequency have smaller code lengths. The details on itemset encoding and memory that is needed to store itemsets of code table are out of scope of this paper. Since we are interested in the length of the description rather than in the encoding itself (materialized codes), we consider simplified version of $L(CT, D)$ where only terms that characterize "specificity" of itemsets for the dataset are taken into account:

$$L(D \mid CT) = \sum_{g \in D} \sum_{B \in cover(g)} l(B) = -\sum_{B \in CT} u(B) \log \frac{u(B)}{U},$$

$$L(CT \mid D) = \sum_{B \in CT} code(B) + l(B),$$

where $code(B)$ is the length in bits to store itemset B in a code table.

Principles of computing a code table. A code table is computed in an incremental manner: starting from a set of single-attribute patterns, i.e. $\{\{m\} \mid m \in M\}$. The optimization procedure is based on adding a new itemset to the code table, correcting the information about usage of the other itemsets in the code table, recomputing the itemset lengths and re-encoding the data. At each iteration a new pattern can be added or not to the code table.

A set of candidates might be composed of any kind of patterns: arbitrary itemsets, closed ones, δ-itemsets, etc. The items in a set of candidates are sorted w.r.t. chosen interestingness measures, in particular, in Krimp [10] patterns are ordered by the length of itemset (its size of attribute set) and frequency.

Note that the problem of computing optimal code table implies exhaustive search for the best combination of patterns, and in practice some heuristics are used.

Example. Let us consider the principle of computing a CT using the running example. The iterative process of updating a CT for class "animal" and recomputing the cover of D_2 is described in Table 3. As candidates we use the set of concepts sorted by area, i.e., frequency $\times$ length, the ordered list of candidates with their areas is given in columns "Candidate set, area". Columns "CT" correspond to the CT at each iteration. The CT contains itemsets and the number of times each itemset is used to cover objects. The covering is given in columns "Data with covering". At the beginning it consists of single-attribute itemsets. The first object $m_1m_2m_3m_6$ is covered by four single-attribute itemsets. The first candidate is $m_1m_2m_3$ with an area equal to 6. At the next step this candidate is used to cover attributes in the dataset covered by single-attribute itemsets. With the chosen candidate, the covering for two objects is changed (compare the first two objects in columns "Data with covering" in "Step 0" and "Step 1"). Step by step a new itemset with the maximal area is added to the CT. If a new CT compresses the data better then the old one, the itemset is accepted to the CT, otherwise, it is removed from both the CT and the candidate set. If the CT is changed, the usage of itemsets in the CT and area for candidates are recomputed.

Table 3. An iterative procedure of computing a code table for class "animal" and the cover of D_2. The names "i" and "u" stand for an itemset and its usage in covering

Step 0				**Step 1**			
CT		**Data with**	**Candidate**	**CT**		**Data with**	**Candidate**
i	**u**	**covering**	**set, area**	**i**	**u**	**covering**	**set, area**
m_3	4	$(m_1)(m_2)(m_3)(m_6)$	$m_1m_2m_3, 6$	$m_1m_2m_3$	2	$(m_1m_2m_3)(m_6)$	$m_1m_3m_8, 3$
m_1	3	$(m_1)(m_2)(m_3)(m_7)$	$m_1m_3, 6$	m_3	2	$(m_1m_2m_3)(m_7)$	$m_3m_4m_9, 3$
m_2	2	$(m_1)(m_3)(m_8)$	$m_1m_2m_3m_6, 4$	m_1, m_4	1	$(m_1)(m_3)(m_8)$	$m_1m_3, 2$
m_4	1	$(m_3)(m_4)(m_9)$	$m_1m_2m_3m_7, 4$	m_6-m_9	1	$(m_3)(m_4)(m_9)$	
m_6-m_9	1		$m_1m_3m_8, 3$	m_2, m_5	0		
m_5	0		$m_3m_4m_9, 3$				
Step 2				**Step 3**			
CT		**Data with**	**Candidate**	**CT**		**Data with**	**Candidate**
i	**u**	**covering**	**set, area**	**i**	**u**	**covering**	**set, area**
$m_1m_2m_3$	2	$(m_1m_2m_3)(m_6)$	$m_3m_4m_9, 3$	$m_1m_2m_3$	2	$(m_1m_2m_3)(m_6)$	
$m_1m_3m_8$	1	$(m_1m_2m_3)(m_7)$		$m_1m_3m_8$	1	$(m_1m_2m_3)(m_7)$	
m_3, m_4	1	$(m_1m_3m_8)$		$m_3m_4m_9$	1	$(m_1m_3m_8)$	
m_6, m_7, m_9	1	$(m_3)(m_4)(m_9)$		m_6, m_7	1	$(m_3m_4m_9)$	
m_1, m_2, m_5, m_8	0			m_1-m_5	0		
				m_8-m_9	0		

3.2 MDL under Supervised Settings

Being purely unsupervised, the MDL principle can be adapted for usage in supervised settings. The idea is to find a compressed representation for objects using code tables of each target class separately. Classes have their own code tables. A code table consists of typical patterns and their lengths (the more typical patterns have shorter lengths). To classify a new object, its set of attributes is covered by itemsets from code tables of each class. Then, the encoding lengths for each class are computed and the class that corresponds to the minimal encoding length is assigned to the object. The length reflects typicality of an object for a particular class (code table).

Example. Consider classification with code tables "CT_1" and "CT_2" from Table 4 that have been computed on sets D_1 and D_2, respectively. The details on the computing of the code tables is out of the scope of this paper (see the Krimp algorithm [10]). Each column"CT_i" contains code tables CT_A and CT_{NA} for "animal" and "not animal" classes, respectively. Let us consider how an object g_9 is classified with the code tables from "CT_1". The main steps of the covering of g'_9 are given in Table 5. To find a covering we use a greedy strategy, i.e., we start from the first itemset in a code table and then stop iterating over itemsets when all attributes of the object are covered. To cover g'_9 with CT_{NA} we take the first itemset $m_1m_4m_5m_8$ (see Step 1 in Table 5), it does not cover g'_9, at the next step we take the second itemset m_4m_9, it covers a subset of g'_9 and m_3 remains uncovered (see Step 2). The iterations over itemsets from CT_{NA} continues until all the attributes of g_9 will be covered.

Classification with code tables from Table 4 are given in Table 6. The objects are covered by itemsets from tables of "animal" and "not animal" as it is described in Table 5). The class where the object has the shortest length is assigned to this object.

In this section we considered how the MDL principle is used to select patterns (itemsets) in un- and supervised settings. In the next section we study how MDL works in the FCA framework.

Table 4. Code tables $CT_{1,A}$ and $CT_{1,NA}$, $CT_{2,A}$ and $CT_{2,NA}$ computed on datasets D_1 and D_2, respectively. The lengths of itemsets are given with their relative size. A shorter itemset is more typical, i.e., more often used to cover the data on which they had been computed.

CT$_1$						**CT$_1$**					
code table "animal", CT_A			**code table "not animal" CT_{NA}**			**code table "animal" CT_A**			**code table "not animal" CT_{NA}**		
itemsets	**usage**	**length in CT**	**itemsets**	**usage**	**length in CT**	**itemsets**	**usage**	**length in CT**	**itemsets**	**usage**	**length in CT**
						$m_1m_2m_3$	2				
$m_1m_2m_3$	1		$m_1m_4m_5m_8$	1		$m_3m_4m_9$	1		$m_1m_4m_5m_8$	1	
$m_1m_3m_8$	1		m_4m_9	2		$m_1m_3m_8$	1		m_4m_9	2	
m_1	0		m_1	0		m_1	0		m_1	0	
m_2	0		m_2	1		m_2	0		m_2	1	
m_3	0		m_3	1		m_3	0		m_3	2	
m_4	0		m_4	1		m_4	0		m_4	1	
m_5	0		m_5	2		m_5	0		m_5	2	
m_6	1		m_6	1		m_6	1		m_6	1	
m_7	1		m_7	0		m_7	1		m_7	1	
m_8	0		m_8	0		m_8	0		m_8	0	
m_9	0		m_9	0		m_9	0		m_9	0	

Table 5. The steps of the covering process of object g_9 by itemsets from the code tables of classes "animal" and "not animal", $CT_{1,A}$ and $CT_{1,NA}$, respectively. To cover g_9' with $CT_{1,A}$ we try to use every itemset from the top, i.e., $m_1m_2m_3$, $m_1m_3m_8$, m_1, etc. We stop when all attributes are covered. The covering procedure for $CT_{1,A}$ and $CT_{1,NA}$ stops after m_9 and m_3, respectively, is being considered.

Covering with CT$_1$ for "animals"		
Step	**Used itemset (an attempt to cover)**	**Remaining attributes in g_9' to cover**
0	-	$m_3m_4m_9$
1	$m_1m_2m_3$	$m_3m_4m_9$
2	$m_1m_3m_8$	$m_3m_4m_9$
3	m_1	$m_3m_4m_9$
4	m_2	$m_3m_4m_9$
5	m_3	m_4m_9
6	m_4	m_9
	...	
11	m_9	$\{\emptyset\}$

Covering with CT$_1$ for "not animals"		
Step	**Used itemset (an attempt to cover)**	**Remaining attributes in g_9' to cover**
0		$m_3m_4m_9$
1	$m_1m_4m_5m_8$	$m_3m_4m_9$
2	m_4m_9	m_3
3	m_1	m_3
4	m_2	m_3
5	m_3	$\{\emptyset\}$

4 MDL in FCA: First Steps

To show that MDL can improve the practical application of FCA, in this section we discuss the results of experiments on the embedding of MDL within FCA. We used the discretized datasets from LUCS-KDD repository [4]. We split the data into 10 parts and in each of 10 experiments we use 9 of them as a training set and one as a test set. The average performance is reported in the paper. To compute code tables and to cover objects the Krimp algorithm [10] is used.

4.1 Descriptive Patterns. FCA in Unsupervised Settings

In unsupervised learning (where the target attribute is not given), one is interested in a small number of meaningful patterns. In our experiments we compute the set of closed itemsets and apply the MDL principle to them. For MDL we

Table 6. Classification with code tables (MDL-based approach) from Table 4. The class of the code table where an object has the shortest encoding length is assigned to the object

	Encoding with itemsets	animal	Encoding with itemsets	not animal	decision
CT_1	$g'_8 = (m_3)(m_7)$		$g'_8 = (m_3)(m_7)$		not animal
	$g'_9 = (m_3)(m_4)(m_9)$		$g'_9 = (m_3)(m_4m_9)$		not animal
	$g'_{10} = (m_1m_2m_3)(m_7)$		$g'_{10} = (m_1)(m_2)(m_3)(m_7)$		animal
	$g'_{11} = (m_1)(m_4)(m_5)(m_8)$		$g'_{11} = (m_1m_4m_5m_8)$		not animal
CT_2	$g'_{10} = (m_1m_2m_3)(m_7)$		$g'_{10} = (m_1)(m_2)(m_3)(m_7)$		animal
	$g'_{11} = (m_1)(m_4)(m_5)(m_8)$		$g'_{11} = (m_1m_4m_5m_8)$		not animal

Table 7. The parameters of sets of formal concepts and their proper MDL-subsets.

dataset	nmb. of obj.	nmb. of attr.	nmb. of concepts		avg. length of intent		dataset	nmb. of obj.	nmb. of attr.	nmb. of concepts		avg. length of intent	
			total	MDL	total	MDL				total	MDL	total	MDL
auto	205	135	67 557	57	8.83	**19.26**	horse colic	368	83	173 808	101	6.96	3.92
breast	699	16	642	24	7.36	**9.04**	iris	150	19	107	13	3.08	**3.92**
car	1 728	25	12 617	94	5.12	3.47	led7	3 200	24	1 937	152	4.60	**6.80**
chess	28 056	58	152 753	1 675	4.85	4.32	mushroom	8 124	90	181 945	**211**	15.23	**19.53**
dermatology	366	49	16 324	47	6.98	5.70	nursery	12 960	30	176 536	**392**	6.53	5.56
ecoli	336	29	694	25	5.49	**6.08**	page blocks	5 473	44	715	45	5.79	**10.27**
flare	1 389	38	16 303	106	6.82	**8.64**	pima indians	768	38	1 609	50	4.99	**5.86**
glass	214	46	4 704	50	5.06	4.32	ticTacToe	958	29	42 685	**160**	5.44	4.02
heart	303	50	36 708	54	7.14	5.09	wine	178	68	13 170	52	5.14	3.90
hepatitis	155	52	199 954	44	8.14	5.59	zoo	101	42	4 563	**17**	7.34	**12.24**

sort itemsets in the candidate set by "length" (the cardinality of intent) and "frequency" (the cardinality of extent).

MDL-optimal concepts are concepts whose intents are included in a code table with a non-empty usage. The results of the experiments are given in Table 7. For instance, the "auto" dataset consists of 205 objects and 135 binary attributes. The total number of formal concepts is 67 557, 57 of them are MDL-optimal.

Our experiments show that the selected itemsets might be shorter or longer on average than the itemsets in the whole set of closed concepts (see column "avg. length of intent" in Table 7). However, around 2% (12 % at most) of the concepts are selected with the MDL-principle. Thus, MDL can be considered as a threshold-free alternative for selection of interesting itemsets. It is important to notice that the subset of MDL-optimal itemsets is composed of diverse patterns and expert assumptions on interestingness of concepts can be embedded by ordering candidates w.r.t. particular interestingness measures. Since a greedy strategy is used to make a code table, one gets a set of diverse itemsets that are in agreement with interestingness.

4.2 Classifier Comparison. FCA in Supervised Settings

In this section we study both the accuracy of ensembles of classifiers $\mathcal{S}$ and their basic elements $\hat{B}$. We also compare the accuracy of the ensembles with commonly used classification methods, e.g., random forest, multilayer perceptron and support vector machine.

Comparison of single classifiers. Here we consider formal concepts as single classifiers. We study precision, recall (Formulas 2 and 3, respectively) and precision loss (Formula 6). Formula 6 is also used as a measure of overfitting, i.e.,

cases where accuracy on a test set is worse than on a training one. If precision loss is close to 1, we call it overfitting, the classifier makes much more errors on a test set than on a training set. If precision loss is close to 0, we get "expected" precision on a test set.

$$\text{prloss}(\hat{B}^e, G_+ \cup G_-) = \text{precision}(\hat{B}^e, G \setminus G^*) - \text{precision}(\hat{B}^e, G^*) \quad (6)$$

In Figure 1 we show the performance of single concepts in the following 2-dimensional spaces: "precision" and "precision loss" (blue shapes), "recall" and "precision loss" (green shapes). Due to lack of space we provide two typical kinds of distributions using results for "Breast cancer" and "Wine" datasets. Precision loss is shown on the vertical axis, recall and precision are combined in the horizontal axis. The pictures are density plots for the (sub)set of concepts in the chosen dimensions. Intense-colored regions correspond to the regions with high concentration of concepts.

Let us consider the classifiers of "Breast cancer" dataset (the first line in Figure 1). The first figure shows that most concepts have precision loss close to 0, i.e., they have similar precision both on training and test set. Classifiers with precision loss close to 0 are preferable, since they have the "expected" precision even on unobservable data. Let us consider the position on the horizontal axis. The blue shape ("precision" axis) is located close to 1, it means that most concepts have high precision on a training set. A long stretch of the green shape along the axis means that the set of classifiers consists of both specific (having relatively big intents) and general concepts (recall is from 0 to 1).

The second plot corresponds to MDL-optimal classifiers. In both spaces, i.e., "precision" and "precision loss" (blue shapes), "recall" and "precision loss" (green shapes), classifiers are concentrated in two points on the "precision loss" axis, the bottom shape is brighter than the upper one. It means that a lot of classifiers have similar precision on training and test sets, and there are several classifiers that have much smaller precision on a test set (they are overfitting classifiers). Since the blue shapes are located close to 0 on the horizontal axis, we conclude that the classifiers are very precise on a training set. The concentration of the green shapes around 0 on the horizontal axis means that most classifiers have recall close to 0, thus, the classifiers are very specific.

Classifiers of "Wine" dataset demonstrate another typical distribution of the classifiers. We can read the plots as it is done above. Here we discuss the key difference between two kinds of datasets.

In our experiments, the set of classifiers on the whole set of formal concepts was comprised of either mostly one type of classifiers with precision loss close to 0 or two types of classifiers: with precision loss close to 0 and to 1. Thus, the set of concept-classifiers contains either mostly non-overfitting (good) classifiers or non- and overfitting ones. MDL-based subset usually includes both non- and overfitted classifiers, all these classifiers are quite specific (have big intents).

A reasonable question arises: what is the accuracy of ensembles of classifiers built on such different types of concepts? In the next paragraph we examine the accuracy of ensembles of classifiers that are constituted by MLD-optimal subsets.

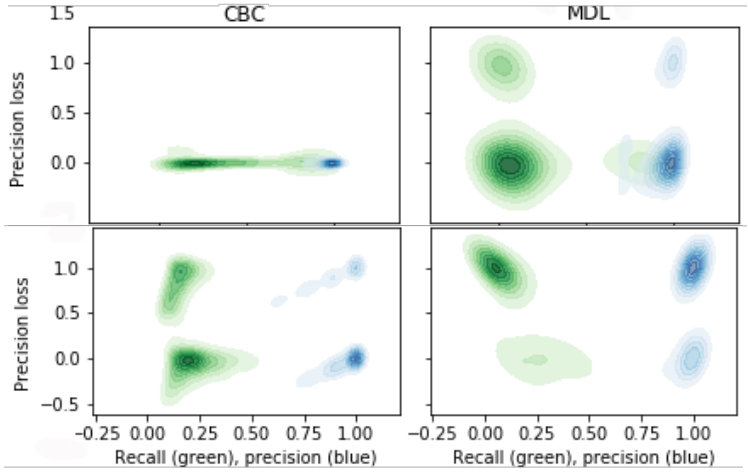

Fig. 1. Precision loss of the whole set of concepts (CBC), MDL-optimal subsets of classifiers for "Breast cancer" (top row) and "Wine" (bottom row) datasets

Comparison of ensembles of classifiers In this section we compare ensembles built on a set of formal concepts ("CBC") and its MDL-optimal subset ("MDL") with commonly used classification methods like random forests (RFs), multilayer perceptrons[3] (MLP) and support vector machines (SVM). We study the average accuracy on a test set, and the results are summarized in Table 8. As it was noticed above, we split our dataset into 10 folds (9:1 for train and test sets). The maximal accuracy over the 10 folds is also reported in the table. We pay our attention to the number of classifiers that constitute an ensemble. The smaller the number of classifiers the faster one can obtain the response and the better this response can be interpreted.

Our experiments show that among itemset-based classifiers (CBC, MDL, RF) MDL-based approach demonstrate quite good performances and have a much smaller set of classifiers. An ensemble with a small number of classifiers performs faster and is better interpretable. Thus, it is easier to classify with MDL-ensembles and understand the obtained results as well.

5 Conclusion

In this paper we have addressed the problem of selecting meaningful, non-redundant sets of formal concepts. We have proposed to use the MDL principle and to show how the expert understanding of interestingness might be incorporated into it.

The MDL principle ensures a good compression even when the set of formal concepts is huge (for example, 24 MDL-optimal among 6 432 concepts for "Breast cancer", 392 among 176 537 concepts for "nursery" dataset).

[3] We select the configuration that ensures the best accuracy among the following ones: (100:50:50),(100:50:25),(50:50:50)

Table 8. Performance of ensembles of classifiers

Dataset name	Classifier	Avg. accuracy	Max. acc.	Avg. rule numb.	Max. rule numb
breast cancer	CBC	0,86 ± 0,04	0,90	355,00 ± 7,10	361
	MDL	**0,94 ± 0,03**	0,99	**27,80 ± 0,98**	30
	RF	0,93 ± 0,04	0,99	30,60 ± 4,48	37
	MLP	**0,94 ± 0,03**	0,99	–	–
	SVM	0,93 ± 0,03	0,99	–	–
ecoli	CBC	0,84 ± 0,05	0,94	394,70 ± 14,47	410
	MDL	0,77 ± 0,10	0,85	**37,30 ± 2,10**	40
	RF	0,77 ± 0,05	0,85	1236,40 ± 532,54	1830
	MLP	0,84 ± 0,03	0,88	–	–
	SVM	**0,86 ± 0,04**	0,94	–	–
iris	CBC	0,93 ± 0,07	1,00	113,20 ± 5,57	119
	MDL	0,94 ± 0,06	1,00	18,50 ± 1,36	20
	RF	0,95 ± 0,07	1,00	169,40 ± 192,45	531
	MLP	0,93 ± 0,07	1,00	–	–
	SVM	0,93 ± 0,07	1,00	–	–

In supervised settings, MDL principle tends to choose the specific classifiers, some of them have a precision loss close to 0.9. However, the MDL principle ensures high classification accuracy.

One interesting direction for future work is to study how some interestingness measures, such as stability, might be embedded into the MDL approach. Another interesting direction is to study connection between the MDL principle and Pareto optimality.

Acknowledgements

The work was supported by the Russian Science Foundation under grant 17-11-01294 and performed at National Research University Higher School of Economics, Moscow, Russia.

References

1. Aggarwal, C.C., Han, J.: Frequent pattern mining. Springer (2014)
2. Baldi, P., Brunak, S., Chauvin, Y., Andersen, C.A., Nielsen, H.: Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics **16**(5), 412–424 (2000)
3. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Fast generation of best interval patterns for nonmonotonic constraints. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 157–172. Springer (2015)
4. Coenen, F.: The lucs-kdd discretised/normalised arm and carm data library (2003), http://www.csc.liv.ac.uk/ frans/KDD/Software/LUCS_KDD_DN
5. Ganter, B., Wille, R.: Formal concept analysis: Logical foundations (1999)
6. Ganter, B., Kuznetsov, S.O.: Formalizing hypotheses with concepts. In: International Conference on Conceptual Structures. pp. 342–356. Springer (2000)
7. Grünwald, P.D.: The minimum description length principle. MIT press (2007)
8. Kuznetsov, S.O.: Machine learning and formal concept analysis. In: Eklund, P. (ed.) Concept Lattices. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
9. Kuznetsov, S.O., Makhalova, T.: On interestingness measures of formal concepts. Information Sciences **442-443**, 202 – 219 (2018)
10. Vreeken, J., Van Leeuwen, M., Siebes, A.: Krimp: mining itemsets that compress. Data Mining and Knowledge Discovery **23**(1), 169–214 (2011)