Machine Learning and Formal Concept Analysis

Sergei O. Kuznetsov

All-Russian Institute for Scientific and Technical Information
Technische Universität Dresden

Abstract A model of learning from positive and negative examples is naturally described in terms of Formal Concept Analysis (FCA). In these terms, result of learning consists of two sets of intents (closed subsets of attributes): the first one contains intents that have only positive examples in the corresponding extents. The second one contains intents such that the corresponding extents contain only negative examples. On the one hand, we show how the means of FCA allows one to realize learning in this model with various data representation, from standard object-attribute one to that with labeled graphs. On the other hand, we use the language of FCA to give natural descriptions of some standard models of Machine Learning such as version spaces and decision trees. This allows one to compare several machine learning approaches, as well as to employ some standard techniques of FCA in the domain of machine learning. Algorithmic issues of learning with concept lattices are discussed. We consider applications of the concept-based learning, including Structure-Activity Relationship problem (in predictive toxicology) and spam filtering.

1 Introduction

Machine Learning is usually defined as a discipline "concerned with the question of how to construct computer programs that automatically improve with experience" [46]. Methods of FCA [62, 23] were from the very beginning (viz., attribute exploration and more later predicate, object and relational explorations) more oriented to human-machine interaction, thus being more along the lines of knowledge discovery: "The KDD process is interactive and iterative, involving numerous steps with many decision being made by the user" [13] and similar principles were e.g., declared in [29, 61]. However, in this paper we would like to relate FCA rather to mathematical models of machine learning, which underlie methods of knowledge discovery. The latter essentially consists in application-driven combination of various learning models under supervision of human experts, which also perform data selection at various stages of the discovery process.

From the very beginning, techniques related to extraction of knowledge from data were among the mainstream of FCA research. Implications between sets of attributes in formal contexts, as opposed to mathematically equivalent functional dependencies in databases, are drawn from datasets, whereas in the DBMS paradigm [41] the database dependencies are usually known in advance to a DB designer. Generating bases of implications from contexts can certainly be called a machine-learning procedure, the same holds for generation of bases of partial implications [42], which became known later as association rules in data mining.

One of the first models of machine learning that used lattices (closure systems) was the JSM-method[1] of automated hypothesis generation [16, 17]. In this model positive hypotheses are sought among intersections of positive example descriptions (object intents), same for negative hypotheses. Various additional conditions can

[1] called so in honor of the English philosopher John Stuart Mill, who introduced methods of inductive reasoning in 19th century.

be imposed on these intersections. For example, in Section 2 we consider so-called counterexample forbidding hypotheses, which are equivalent to implications with the value of the target attribute (positive or negative) in the consequence and closed set of attributes in the premise.

In terms of FCA, the system CHARADE described in [18] basically constructs rules of the form $A \to A''$. With the use of properties of Galois connections a sort of nonminimal base of implications is obtained. In system GRAND [50] learning with lattices proceeded as follows: At input the system has training positive and negative examples described by many-valued attributes. The system creates partial description and orders them by generality, completes the partial order to a lattice (which is known in the lattice theory as Dedekind-McNeille completion) and then finds implications (in the FCA sense) with consequences being values of the target attributes. In fact a base of implications with minimal premises is obtained, which however is not minimal in the number of implications. Some machine learning systems, e.g., those described in [55, 43] use various heuristics (based on allowances, accuracy, confidence, support, entropy, etc.) for feature selection and reduction of the number of possible concept-based dependencies learned from positive and negative examples. For example, Rulearner system [55] generates hypotheses (closed sets of attributes that are subsets of only some positive examples) with largest extents, which obviously belong to the set of minimal hypotheses. Among these the system looks for hypotheses with smallest cardinality. Additionally to this, the system deletes useless hypotheses, i.e., those that produce no classification of previously unclassified examples. Then the system deletes those instances that result only in useless hypotheses and repeat hypothesis generation for the new dataset. GALOIS system described in [8] realized clustering (i.e., unsupervised learning) based on concept lattices. For classifying a new object similarity between it and existing clusters (concepts) is computed as the number of common attributes. In 1990s the idea of a version space was elaborated by means of logical programming within the Inductive Logical Programming (ILP), where the notion of a subsumption lattice plays an important role [48]. In late 1990s the notion of a lattice of "closed itemsets" became important in the data mining community, see [13, 52].

The paper is organized as follows. In Section 2 we recall basic definitions of FCA and those related to concept-based hypotheses. In Section 3 we consider a learning model for pattern structures, i.e., for data that cannot be directly described by object-attribute matrices, but allow for a computable "meet" operation. In Section 4 we consider version spaces, a basic construction in machine learning, related to all possible classifiers compatible with a training sample, from the viewpoint of FCA. In Section 5 we show how decision trees can be naturally captured in terms of FCA. In Section 6 we discuss algorithmic issues of concept-based learning. Finally, in Section 7 we consider some applications of concept-based learning.

2 Basic definitions: concepts, implications, and hypotheses

First, to make the paper self-contained, we introduce standard definitions of Formal Concept Analysis (FCA) [23]. We consider a set M of "structural attributes", a set G of objects (or observations) and a relation $I \subseteq G \times M$ such that $(g, m) \in I$ if and only if object g has the attribute m. Such a triple $\mathbb{K} := (G, M, I)$ is called a *formal context*. Using the *derivation operators*, defined for $A \subseteq G$, $B \subseteq M$ by

$$A' := \{m \in M \mid gIm \text{ for all } g \in A\},$$
$$B' := \{g \in G \mid gIm \text{ for all } m \in B\},$$

we can define a *formal concept* (of the context $\mathbb{K}$) to be a pair (A, B) satisfying $A \subseteq G$, $B \subseteq M$, $A' = B$, and $B' = A$. A is called the *extent* and B is called the

intent of the concept (A, B). These concepts, ordered by

$$(A_1, B_1) \geq (A_2, B_2) \iff A_1 \supseteq A_2$$

form a complete lattice, called *the concept lattice* of $\mathbb{K} := (G, M, I)$.

Next, we use the FCA setting to describe JSM-hypotheses from [16, 17]. In addition to the structural attributes of M, we consider (as in [36, 19]) a *target attribute* $\omega \notin M$. This partitions the set G of all objects into three subsets: The set G_+ of those objects that are known to have the property ω (these are the *positive examples*), the set G_- of those objects of which it is known that they do not have ω (the *negative examples*) and the set G_τ of *undetermined examples*, i.e., of those objects, of which it is unknown if they have property ω or not. This gives three subcontexts of $\mathbb{K} = (G, M, I)$, the first two staying for the training sample:

$$\mathbb{K}_+ := (G_+, M, I_+), \quad \mathbb{K}_- := (G_-, M, I_-), \quad \text{and } \mathbb{K}_\tau := (G_\tau, M, I_\tau),$$

where for $\varepsilon \in \{+, -, \tau\}$ we have $I_\epsilon := I \cap (G_\varepsilon \times M)$ and the corresponding derivation operators are denoted by $(\cdot)^+$, $(\cdot)^-$, $(\cdot)^\tau$, respectively.

Intents, as defined above, are attribute sets shared by some of the observed objects. In order to form hypotheses about structural causes of the target attribute ω, we are interested in sets of structural attributes that are common to some positive, but to no negative examples. Thus, a *positive hypothesis* h for ω (called "counter-example forbidding hypotheses" in the JSM-method [16, 17]) is an intent of $\mathbb{K}_+$ such that $h^+ \neq \emptyset$ and $h \not\subseteq g^- := \{m \in M \mid (g, m) \in I_-\}$ for any negative example $g \in G_-$. An intent of $\mathbb{K}_+$ that is contained in the intent of a negative example is called a *falsified (+)-generalization. Negative hypotheses* are defined similarly. Hypotheses can be used to classify the undetermined examples: If the intent

$$g^\tau := \{m \in M \mid (g, m) \in I_\tau\}$$

of an object $g \in G_\tau$ contains a positive, but no negative hypothesis, then g^τ is *classified positively*. Negative classifications are defined similarly. If g^τ contains hypotheses of both kinds, or if g^τ contains no hypothesis at all, then the classification is contradictory or undetermined, respectively. In this case one can apply standard probabilistic techniques known in machine learning and data mining (majority vote, Bayesian approach, etc.)

In [35, 36] we argued that one can restrict to *minimal* (w.r.t. inclusion $\subseteq$) hypotheses, positive as well as negative, since an object intent obviously contains a positive hypothesis if and only if it contains a minimal positive hypothesis.

Example 1. Consider the following data table

G \ M	color	form	firm	smooth	**target**
1 apple	yellow	round	no	yes	+
2 grapefruit	yellow	round	no	no	+
3 kiwi	green	oval	no	no	+
4 plum	blue	oval	no	yes	+
5 toy cube	green	cubic	yes	yes	−
6 egg	white	oval	yes	yes	−
7 tennis ball	white	round	no	no	−

This dataset or *multivalued context* can be reduced to a context of the form presented above by *scaling* [23], e.g., as follows (scaling 1):

G \ M	w	y	g	b	f	$\overline{\text{f}}$	s	$\overline{\text{s}}$	r	o	$\overline{\text{o}}$	**target**
1 apple		×				×	×		×	×		+
2 grapefruit		×				×		×	×	×		+
3 kiwi			×			×		×		×		+
4 plum				×		×	×			×		+
5 toy cube			×		×		×				×	−
6 egg	×				×		×			×		−
7 tennis ball	×					×		×	×			−

Here we use the following abbreviations: "w" for white, "y" for yellow, "g" for green, "b" for blue, "s" for smooth, "f" for firm, "r" for round, "o" for oval, and "$\overline{\text{m}}$" for $m \in \{\text{w, y, g, b, s, f, r, o,}\}$. This context gives rise to the positive concept lattice in Fig. 1, where we marked minimal (+)-hypotheses and falsified (+)-generalizations. If we have an undetermined example `mango` with $\texttt{mango}^{\tau} = \{\text{y, } \overline{\text{f}}\text{, s, o}\}$ then it is classified positively, since $\texttt{mango}^{\tau}$ contains the minimal hypothesis $\{\overline{\text{f}}\text{, o}\}$ and does not contain any negative hypothesis.

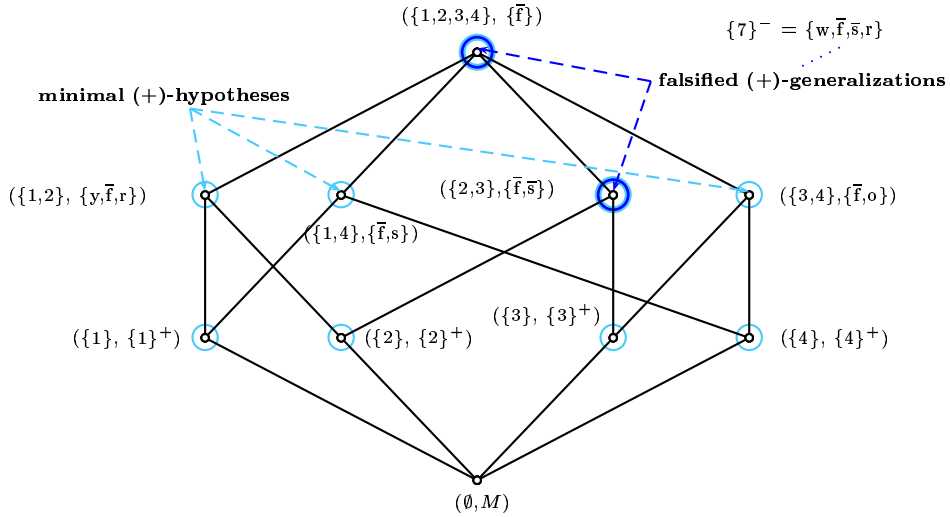


Fig. 1. Positive concept lattice for scaling 1

For this scaling we have two minimal negative hypotheses: {w} (supported by examples `egg` and `tennis ball` and {f, s} (supported by examples `toy cube` and `egg`. The context can be scaled differently, e.g. in this way (scaling 2):

G \ M	w	y	g	b	$\overline{\text{w}}$	$\overline{\text{y}}$	$\overline{\text{g}}$	$\overline{\text{b}}$	f	$\overline{\text{f}}$	s	$\overline{\text{s}}$	r	o	$\overline{\text{r}}$	$\overline{\text{o}}$	**target**
1 apple		×			×		×	×		×	×		×	×			+
2 grapefruit		×			×		×	×		×		×	×	×			+
3 kiwi			×		×	×		×		×		×		×	×		+
4 plum				×	×	×	×			×	×			×	×		+
5 toy cube			×		×	×		×	×		×				×	×	−
6 egg	×					×	×	×	×		×			×	×		−
7 tennis ball	×					×	×	×		×		×	×	×			−

This scaling gives rise to another positive concept lattice, all intents of which are (+)-hypotheses. The unique minimal hypothesis (corresponding to the top element of the concept lattice) is $\{\overline{\text{w}}, \overline{\text{f}}, \text{o}\}$. Two minimal negative hypotheses are $\{\overline{\text{y}}, \overline{\text{b}}, \overline{\text{r}}, \text{f}, \text{s}\}$ (supported by examples 5 and 6) and $\{\overline{\text{y}}, \overline{\text{g}}, \overline{\text{b}}, \text{w}, \text{o}\}$ (supported by examples 6 and 7).

3 Learning in Pattern Structures

3.1 Pattern Structures and Hypotheses Therein

Learning with descriptions given by logical formulas is systematically studied in ILP [48], with applications to learning with molecular graphs described by logical formulas [60, 6]. In FCA community several authors have considered the case where instead of having attributes the objects satisfy certain logical formulas [2, 9, 14] or they are described by labeled graphs [35, 37, 40]. In case of logical formulas shared attributes are replaced by common subsumers of the respective formulas. In [20] we showed how such an approach is linked to the general FCA framework.

Let G be some set, let $(D, \sqcap)$ be a meet-semilattice and let $\delta : G \to D$ be a mapping. Then $(G, \underline{D}, \delta)$ with $\underline{D} = (D, \sqcap)$ is called a *pattern structure*, provided that the set

$$\delta(G) := \{\delta(g) \mid g \in G\}$$

generates a complete subsemilattice $(D_\delta, \sqcap)$ of $(D, \sqcap)$, i.e., every subset X of $\delta(G)$ has an infimum $\sqcap X$ in $(D, \sqcap)$ and D_δ is the set of these infima. Each such complete semilattice has lower and upper bounds, which we denote by $\mathbf{0}$ and $\mathbf{1}$, respectively. There are two natural situations where the condition on the complete subsemilattice is automatically satisfied: when $(D, \sqcap)$ is complete, and when G is finite.

If $(G, \underline{D}, \delta)$ is a pattern structure, we define the derivation operators as

$$A^\diamond := \sqcap_{g \in A} \delta(g) \qquad \text{for } A \subseteq G$$

and

$$d^\diamond := \{g \in G \mid d \sqsubseteq \delta(g)\} \qquad \text{for } d \in D.$$

The elements of D are called *patterns*. The natural order on them is given, as usual, by

$$c \sqsubseteq d :\iff c \sqcap d = c,$$

and is called the *subsumption* order[2]. The operators $\diamond$ obviously make a Galois connection between the power set of G and $(D, \sqsubseteq)$. The pairs (A, d) satisfying

$$A \subseteq G, \quad d \in D, \quad A^\diamond = d, \quad \text{and} \quad A = d^\diamond$$

[2] Note that subsumption order on patterns defined in this way (a larger pattern subsumes a smaller pattern), being "intentional," is inverse to definitions of subsumption in logics, where it is "extensional" (a more general formula, covering more ground facts, subsumes a less general formula).

are called the *pattern concepts* of $(G, \underline{D}, \delta)$, with extent A and *pattern intent* d. For $a, b \in D$ the *pattern implication* $a \to b$ holds if $a^\diamond \sqsubseteq b^\diamond$. Similarly, for $C, D \subseteq G$ the *object implication* $C \to D$ holds if $C^\diamond \sqsubseteq D^\diamond$.

Since $(D_\delta, \sqcap)$ is complete, there is a (unique) operation $\sqcup$ such that $(D_\delta, \sqcap, \sqcup)$ is a complete lattice. It is given by

$$\sqcup X := \sqcap\{c \in D_\delta \mid \forall_{x \in X}\ x \sqsubseteq c\}.$$

A subset M of D is $\sqcup$-*dense* for $(D_\delta, \sqcap)$ if every element of D_δ is of the form $\sqcup X$ for some $X \subseteq M$. If this is the case, then with

$$\downarrow d := \{e \in D \mid e \sqsubseteq d\}$$

we get

$$c = \sqcup(\downarrow c \cap M) \qquad \text{for every } c \in D_\delta.$$

Of course, $M := D_\delta$ is always an example of a $\sqcup$-dense set.

If M is $\sqcup$-dense in $(D_\delta, \sqcap)$, then the formal context (G, M, I) with I given as $gIm :\Leftrightarrow m \sqsubseteq \delta(g)$ is called a *representation context* for $(G, \underline{D}, \delta)$. The following result from [20] can be proved by a standard application of the basic theorem of FCA [23].

Theorem 1 *Let $(G, \underline{D}, \delta)$ be a pattern structure and let (G, M, I) be a representation context of $(G, \underline{D}, \delta)$. Then for any $A \subseteq G$, $B \subseteq M$ and $d \in D$ the following two conditions are equivalent*

1. *(A, d) is a pattern concept of $(G, \underline{D}, \delta)$ and $B = \downarrow d \cap M$.*
2. *(A, B) is a formal concept of (G, M, I) and $d = \bigsqcup B$.*

Thus, the pattern concepts of $(G, \underline{D}, \delta)$ are in 1-1-correspondence with the formal concepts of (G, M, I). Corresponding concepts have the same first components (called *extents*). These extents form a closure system on G and thus a complete lattice, which is isomorphic to the concept lattice of (G, M, I).

An approach to generating pattern concepts and implications between objects can be made in lines of a procedure proposed in [2]. This procedure, called the *object exploration*, is the dual of the *attribute exploration* algorithm, which is standard in Formal Concept Analysis [23]. In the beginning of the exploration process one has the empty set of object implications and the set of extents E, consisting at the initialization step of the empty extent. One considers the set of implications of the form $A \to A''$ for $A \in E$ in the lexicographical order and asks an expert whether each particular implication holds. If the expert says yes, then either the set of implications or the set of extents are updated (dependent on the fact whether a set of objects is pseudoclosed or closed), if not, the expert should provide a counterexample that updates the current set of objects.

As the result of object exploration one obtains the context with the same concept lattice as the lattice of the subsumption hierarchy (in case of description languages this is given by the lattice of least common subsumers) and the stem base of object implications. The procedure proposed in [2] also applies to the general setting with an arbitrary semilattice D.

In [36, 37, 19] we considered a learning model from [17] in terms of Formal Concept Analysis. This model assumes that the cause of a *target property* resides in common attributes of objects that have this property.

For pattern structures this can be formalized as follows. Let $(G, \underline{D}, \delta)$ be a pattern structure together with an external target property ω. As in the case of standard contexts, the set G of all objects is partitioned into three disjoint sets w.r.t. ω: the

sets G_+, G_-, G_τ of positive, negative, and undetermined examples. This gives three pattern substructures of $(G, \underline{D}, \delta)$: $(G_+, \underline{D}, \delta)$, $(G_-, \underline{D}, \delta)$, $(G_\tau, \underline{D}, \delta)$.

A *positive hypothesis* h is defined as a pattern intent of $(G_+, \underline{D}, \delta)$ that is not subsumed by any pattern from $\delta(G_-)$ (for short: not subsumed by any negative example). Formally: $h \in D$ is a positive hypothesis iff

$$h^\diamond \cap G_- = \emptyset \text{ and } \exists A \subseteq G_+ : A^\diamond = h.$$

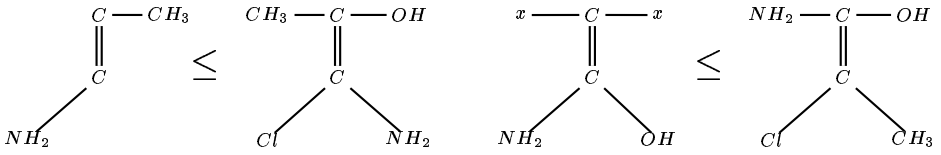
A *negative hypothesis* is defined accordingly.

A hypothesis in the sense of Section 2 [19] is obtained as a special case of this definition when $(D, \sqcap) = (2^M, \cap)$ for some set M. Hypotheses can be used for classification of undetermined examples as introduced in [17] in the following way. If $g \in G_\tau$ is an undetermined example, then a hypothesis h with $h \sqsubseteq \delta(g)$ is *for the positive classification* of g if h is positive and *for the negative classification* of g if it is a negative hypothesis. Classification of an example $g \in G_\tau$ is defined in the same way as in Section 2 (with $\subseteq$ replaced by $\sqsubseteq$).

Example 2. Consider a pattern structure based on the following ordered set P of graphs with labels from the set $\mathcal{L}$ with partial order $\preceq$. Each labeled graph Γ from P is a triple of the form $((V, l), E)$, where V is a set of vertices, E is a set of edges and $l: V \to \mathcal{L}$ is a label assignment function, taking a vertex to its label.
For two graphs $\Gamma_1 := ((V_1, l_1), E_1)$ and $\Gamma_2 := ((V_2, l_2), E_2)$ from P Γ_1 **dominates** Γ_2 or $\Gamma_2 \leq \Gamma_1$ if there exists a one-to-one mapping $\varphi: V_2 \to V_1$ such that it

- respects edges: $(v, w) \in E_2 \Rightarrow (\varphi(v), \varphi(w)) \in E_1$,
- fits under labels: $l_2(v) \preceq l_1(\varphi(v))$.

For example, if $\mathcal{L} = \{\mathrm{C}, \mathrm{NH_2}, \mathrm{CH_3}, \mathrm{OH}, x\}$ we can have



vertex labels are unordered

$x \preceq A$ for any vertex label $A \in \mathcal{L}$

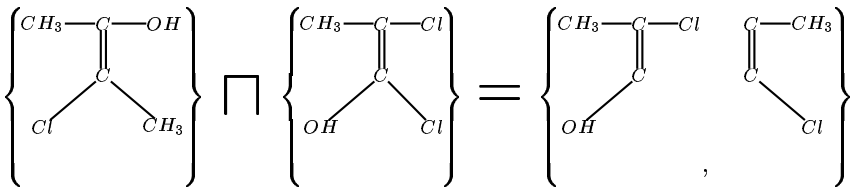
A meet operation $\sqcap$ on graph sets can then be defined as follows: For two graphs X and Y from P

$$\{X\} \sqcap \{Y\} := \{Z \mid Z \leq X, Y, \quad \forall Z_* \leq X, Y \quad Z_* \not\geq Z\},$$

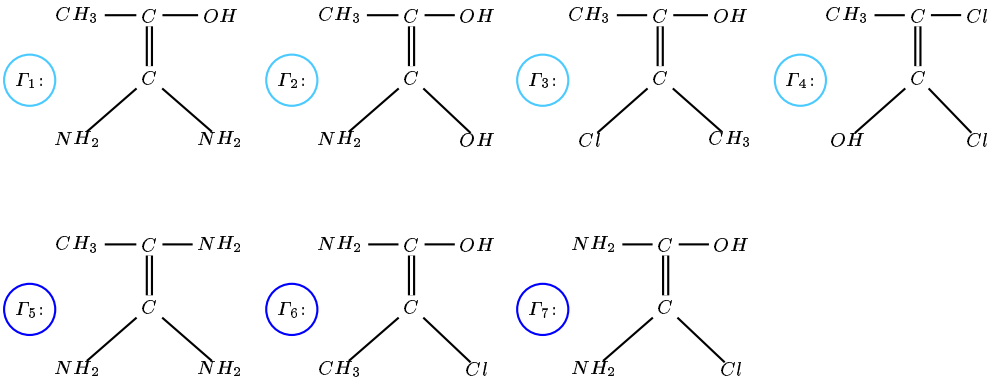
i.e., $\{X\} \sqcap \{Y\}$ is the set of all maximal common subgraphs of X and Y up to substitution of a vertex label by a vertex label smaller w.r.t. $\preceq$. The meet of nonsingleton sets of graphs is defined as

$$\{X_1, \ldots, X_k\} \sqcap \{Y_1, \ldots, Y_m\} := \mathrm{MAX}_\leq(\sqcup_{i,j}(\{X_i\} \sqcap \{Y_j\}))$$

for details see [35, 37, 20]. Here is an example of applying $\sqcap$ defined above:



Let positive examples be described by graphs $\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4$ and negative examples be described by graphs $\Gamma_5, \Gamma_6, \Gamma_7$:



then the lattice of the pattern structure $(G_+, \underline{D}, \delta)$, where $\underline{D}$ is the semilattice on graph sets and δ is a function taking an object to its graph description, is given in Fig. 2, where (+)-hypotheses and falsified (+)-generalizations are marked:

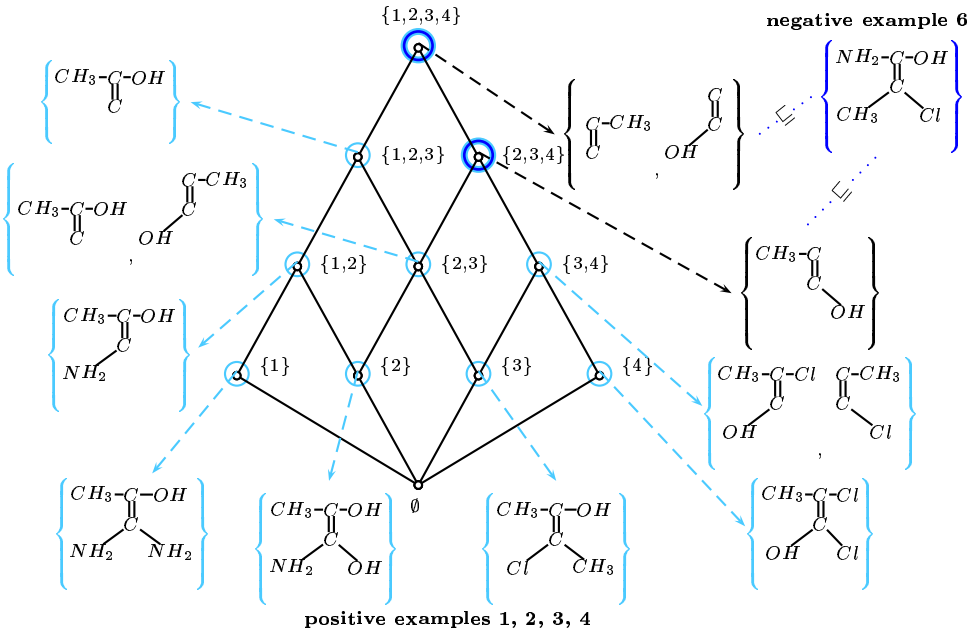


Fig. 2. The lattice of the positive pattern structure

3.2 Projections and projected hypotheses

Since for some pattern structures (e.g., for the pattern structure given by sets of graphs with labeled vertices) even computing subsumption relation may be NP-hard, in practical situations we need to look for some approximation tools, which would replace the patterns with simpler ones, even if that results in some loss of information. To this end we use a mapping $\psi: D \to D$ that replaces each pattern $d \in D$ by $\psi(d)$ such that the pattern structure $(G, \underline{D}, \delta)$ is replaced by $(G, \underline{D}, \psi \circ \delta)$. To distinguish two pattern structures, which we consider simultaneously, we use the symbol $^\diamond$ only for $(G, \underline{D}, \delta)$, not for $(G, \underline{D}, \psi \circ \delta)$. We additionally require that ψ is a kernel operator (or **projection**), i.e., that ψ is

monotone: if $x \sqsubseteq y$, then $\psi(x) \sqsubseteq \psi(y)$,
contractive: $\psi(x) \sqsubseteq x$, and
idempotent: $\psi(\psi(x)) = \psi(x)$.

This requirement seems to hold for any natural approximation mapping and projection thus defined has a nice property well-known in order theory: Any projection of a complete semilattice $(D, \sqcap)$ is $\sqcap$-preserving, i.e., for any $X, Y \in D$

$$\psi(X \sqcap Y) = \psi(X) \sqcap \psi(Y).$$

This helps us to describe how the lattice of pattern concepts changes when we replace $(G, \underline{D}, \delta)$ by its approximation $(G, \underline{D}, \psi \circ \delta)$. First, we note that $\psi(d) \sqsubseteq \delta(g) \Leftrightarrow \psi(d) \sqsubseteq \psi \circ \delta(g)$. Then, using the basic theorem of FCA (which, in particular allows one to represent every lattice as a concept lattice), we showed how the projected pattern lattice is represented by a context [20]:

Theorem 2 *For pattern structures $(G, \underline{D}, \delta_1)$ and $(G, \underline{D}, \delta_2)$ the following statements are equivalent:*

1. *$\delta_2 = \psi \circ \delta_1$ for some projection ψ of $\underline{D}$.*
2. *There is a representation context (G, M, I) of $(G, \underline{D}, \delta_1)$ and some $N \subseteq M$ such that $(G, N, I \cap (G \times N))$ is a representation context of $(G, \underline{D}, \delta_2)$.*

Again, the basic theorem helped us to "binarize" the initial data representation. However, to do this, we need first to compute the pattern lattice. Pattern structures are naturally ordered by projections: $(G, \underline{D}, \delta_1) \geq (G, \underline{D}, \delta_2)$ if there is a projection ψ such that $\delta_2 = \psi \circ \delta_1$. In this case, representation $(G, \underline{D}, \delta_2)$ can be said to be rougher than $(G, \underline{D}, \delta_1)$ and the latter to be finer than the former. In comparable pattern structures implications are related as follows: If $\psi(a) \to \psi(b)$ and $\psi(b) = b$ then $a \to b$ for arbitrary $a, b \in D$.

The properties of projection allow one to relate hypotheses in the original representation with those approximated by a projection. As in [20] we use the term "hypothesis" to those obtained for $(G, \underline{D}, \delta)$ and we refer to those obtained for $(G, \underline{D}, \psi \circ \delta)$ as *ψ-hypotheses*. There is no guarantee that the ψ-image of a hypothesis must be a ψ-hypothesis. In fact, our definition allows that ψ is the "null projection" with $\psi(d) = \mathbf{0}$ for all $d \in D$. (total abandoning of the data with no interesting hypotheses). However, if $\psi(d)$ is a (positive) hypothesis, then $\psi(d)$ is also a (positive) ψ-hypothesis. If we want to look another way round, we have the following: if $\psi(d)$ is a (positive) ψ-hypothesis, then $\psi(d)^{\diamond\diamond}$ is a (positive) hypothesis [20].

The set of all hypothesis-based classifications does not shrink when we pass from d to $\psi(d)$. Formally, if d is a hypothesis for the positive classification of g and $\psi(d)$ is a positive ψ-hypothesis, then $\psi(d)$ is for the positive classification of g.

The above observations show that we can generate hypotheses starting from projections. For example, we can select only those that can be seen in the projected data, which is suggested by the following theorem from [20]:

Theorem 3 *For any projection ψ and any positive hypothesis $d \in D$ the following are equivalent:*

1. *$\psi(d)$ is not subsumed by any negative example.*
2. *There is some positive ψ-hypothesis h such that $h^{\diamond\diamond} \sqsubseteq d$.*

3.3 Algorithmic problems of learning in concept lattices and pattern structures

Computing concept-based hypotheses can be hard. The number of concepts of a formal context, i.e., the size of a concept lattice, can be exponential in the size of a context (e.g., for the context $(A, A, \neq)$, which gives rise to a Boolean concept lattice) and the problem of computing the size of a concept lattice is #P-complete [34, 38]. All hypotheses can be generated by a polynomial delay algorithm, however a (cumulative) polynomial delay algorithm for minimal hypotheses is not known. In certain cases, e.g., when the number of attributes per object is bounded, the computation of hypotheses can be realized in polynomial time. In principle, any known algorithm for computing concepts (see, e.g., review [39]) can be adapted to computing hypotheses.

When a pattern structure is given, we may, for example, determine its concepts by computing all infima of subsets of D_δ and thereby all pattern concepts. To this end we can adapt some standard FCA algorithms, e.g., `Next Closure` [23]. Here one should take into account that performing a single closure may take exponential time. For example, already the problem of testing the $\sqsubseteq$ relation for a lattice on sets of labeled graphs from [35, 37, 20] is NP-complete (equivalent to SUBGRAPH ISOMORPHISM problem [24]), and computing $X \sqcap Y$ is even more difficult. A similar algorithm of this type was described in [37] for computing with sets of graphs. The time complexity of the algorithm is $O((\alpha + \beta|G|)|G||L|)$ and its space complexity is $O(\gamma|G||L|)$, where α is time needed to perform $\sqcap$ operation and β is time needed to test $\sqsubseteq$ relation and γ is the space needed to store the largest object from D_δ. Computing the line diagram of the set of all concepts, given the tree generated by the previous algorithm, takes $O((\alpha|G| + \beta|G|^2)|L|)$ time and $O((\gamma|G||L|)$ space [37].

4 Hypotheses, concept lattices, and decision trees

In this Section we consider the relation between decision trees, concept lattices and concept-based hypotheses. We describe a typical procedure of constructing a decision tree (see, e.g., [54]) in terms of concept lattices.

As input, a system constructing a decision tree receives descriptions of positive and negative examples (or positive and negative contexts). The root of the tree corresponds to the beginning of the process and is not labeled. Other vertices of the decision tree are labeled by attributes and edges are labeled by values of the attributes (e.g., 0 or 1 in case of binary contexts), each leaf is additionally labeled by a class + or −, meaning that all examples with attribute values from the path leading from the root to the leaf belong to a certain class, either + or −.

Systems like ID3 [54] (see also [46]) compute the value of the *information gain* (or negentropy) for each vertex and each attribute not chosen in the branch above. The attribute with the greatest value of the information gain (with the smallest entropy, respectively) "most strongly separates" objects from classes + and −. The algorithm sequentially extends branches of the tree by choosing attributes with the highest information gain. The extension of a branch stops when a next attribute value together with attributes above in the branch uniquely classify examples with

this value combination in one of classes + or −. In some algorithms, the process of extending a branch stops before this in order to avoid *overfitting*, i.e., the situation where all or almost all examples from the training sample are classified correctly by the resulting decision tree, but objects from test datasets are classified with many errors.

Now we consider decision trees more formally. Let the training data be described by the context $\mathbb{K}_{+-} = (G_+ \cup G_-, M, I_+ \cup I_-)$ with the derivation operator denoted by $(\cdot)'$. In FCA terms this context is called the *subposition* of $\mathbb{K}_+$ and $\mathbb{K}_-$. Assume for simplicity sake that for each attribute $m \in M$ there is an attribute $\overline{\mathrm{m}} \in M$, a "negation" of m: $\overline{\mathrm{m}} \in g'$ iff $m \notin g'$. A set of attributes M with this property is called *dichotomized* in FCA. We call a subset of attributes $A \subseteq M$ *noncontradictory* if either $m \notin A$ or $\overline{\mathrm{m}} \notin A$. We call a subset of attributes $A \subseteq M$ *complete* if for every $m \in M$ one has $m \in A$ or $\overline{\mathrm{m}} \in A$.

We would like first to avoid mentioning the use of any optimization functional like information gain for selecting attributes and consider construction of all possible decision trees. The construction of an arbitrary decision tree proceeds by sequentially choosing attributes. If different attributes $m_1, \ldots, m_k$ were chosen one after another, then the sequence $\langle m_1, \ldots, m_k \rangle$ is called a *decision path* if $\{m_1, \ldots, m_k\}$ is noncontradictory and there exists an object $g \in G_+ \cup G_-$ such that $\{m_1, \ldots, m_k\}' \subseteq g'$ (i.e., there is an example with this set of attributes). A decision path $\langle m_1, \ldots, m_i \rangle$ is a (proper) subpath of a decision path $\langle m_1, \ldots, m_k \rangle$ if $i \leq k$ ($i < k$, respectively). A decision path $\langle m_1, \ldots, m_k \rangle$ is called *full* if all objects having attributes $\{m_1, \ldots, m_k\}$ are either positive or negative examples (i.e., have either + or − value of the target attribute). We call a full decision path *irredundant* if none of its subpaths is a full decision path. The set of all chosen attributes in a full decision path can be considered as a sufficient condition for an object to belong to a class $\varepsilon \in \{+, -\}$. A decision tree is then a set of full decision paths.

In what follows, we shall use extensively the one-to-one correspondence between vertices of a decision tree and the related decision paths, representing the latter, when this does not lead to ambiguity, by their last chosen attributes. By *closure of a decision path* $\langle m_1, \ldots, m_k \rangle$ we mean the closure of the corresponding set of attributes, i.e., $\{m_1, \ldots, m_k\}''$. Now we relate decision trees with the covering relation graph of the concept lattice of the context $\mathbb{K} = (G, M, I)$, where the set of objects G is of size $2^{|M|/2}$ and the relation I is such that the set of object intents is exactly the set of complete noncontradictory subsets of attributes. In terms of FCA [23] the context $\mathbb{K}$ is the *semiproduct* of $|M|/2$ *dichotomic scales* or $\mathbb{K} = D_1 \between \ldots \between D_{|M|/2}$ (denoted by $\between_M D$ for short), where each dichotomic scale D_i stays for the pair of attributes (m,$\overline{\mathrm{m}}$).

In a concept lattice a sequence of concepts with decreasing extents we call a *descending chain*. If the chain starts at the top element of the lattice, we call it *rooted*.

Proposition 4. *Every decision path is a rooted descending chain in* $\underline{\mathfrak{B}}(\between_M D)$ *and every rooted descending chain consisting of concepts with nonempty extents in* $\underline{\mathfrak{B}}(\between_M D)$ *is a decision path.*

To relate decision trees to hypotheses introduced above we consider again the contexts $\mathbb{K}_+ = (G_+, M, I_+)$, $\mathbb{K}_- = (G_-, M, I_-)$, and $\mathbb{K}_{+-} = (G_+ \cup G_-, M, I_+ \cup I_-)$. The context $\mathbb{K}_{+-}$ can be much smaller than $\between_M D$ because the latter always has $2^{|M|/2}$ objects while the number of objects in the former is the number of examples. Also the lattice $\underline{\mathfrak{B}}(\mathbb{K}_{+-})$ can be much smaller than $\underline{\mathfrak{B}}(\between_M D)$.

Proposition 5. *A full decision path* $\langle m_1, \ldots, m_k \rangle$ *corresponds to a rooted descending chain* $\langle (m_1'', m_1'), \ldots, (\{m_1, \ldots, m_k\}'', \{m_1, \ldots, m_k\}') \rangle$ *of the line diagram of* $\underline{\mathfrak{B}}(\mathbb{K}_{+-})$ *and the closure of each full decision path* $\langle m_1, \ldots, m_k \rangle$ *is a hypothesis,*

either positive or negative. Moreover, for each minimal hypothesis h, there is a full irredundant path $\langle m_1, \ldots, m_k \rangle$ such that $\{m_1, \ldots, m_k\}'' = h$.

This proposition also illustrates the difference between hypotheses and irredundant decision trees. The former correspond to "most cautious" (most specific) learning in the sense that they are least general generalizations of descriptions of positive examples (or object intents, in terms of FCA). The shortest decision paths (for which in no decision tree there exist full paths with proper subsets of attribute values) correspond to the "most courageous" learning (often referred to as "most discriminant" in machine learning community): being the shortest possible rules, they are most general generalizations of positive example descriptions. However, it is not guaranteed that for a given training set resulting in a certain set of minimal hypothesis there is a decision tree such that minimal hypotheses are among closures of its paths (see Example 3 below). In general, to obtain all minimal hypotheses as closures of decision paths one needs to consider several decision trees, not all of them being optimal w.r.t. a procedure based on the information gain functional (like ID3 or C4.5). The issues of generality of generalizations and, in particular, the relation between most cautious (most specific) and most courageous (most general) generalizations, are naturally captured in terms of version spaces, which we consider in the next section.

In real systems for the construction of decision trees like ID3 or C4.5 the process of constructing a decision path is driven by the information gain functional: a next chosen attribute should have maximal information gain. For dichotomized attributes the information gain is defined for a pair of attributes $m, \overline{m} \in M$. Given a decision path $\langle m_1, \ldots, m_k \rangle$

$$\mathrm{IG}(m) := -\frac{|A'_m|}{|G|}\mathrm{Ent}(A_m) - \frac{|A'_{\overline{m}}|}{|G|}\mathrm{Ent}(A_{\overline{m}}),$$

where $A_m := \{m_1, \ldots, m_k, m\}$, $A_{\overline{m}} := \{m_1, \ldots, m_k, \overline{m}\}$, and for $A \subseteq M$

$$\mathrm{Ent}(A) := -\sum_{\varepsilon \in \{+,-\}} p(\varepsilon \mid A) \cdot \log_2 p(\varepsilon \mid A),$$

$\{+,-\}$ are values of the target attribute and $p(\varepsilon \mid A)$ is the conditional sample probability (for the training set) that an object having a set of attributes A belongs to a class $\varepsilon \in \{+,-\}$. If the derivation operator $(\cdot)'$ is associated with the context $(G_+ \cup G_-, M, I_+ \cup I_-)$, then, by definition of the conditional probability, we have

$$p(\varepsilon \mid A) = \frac{|A' \cap G_\varepsilon|}{|A'|} = \frac{|(A'')' \cap G_\varepsilon|}{|(A'')'|} = p(\varepsilon \mid A'')$$

by the property of the derivation operator $(\cdot)'$: $(A'')' = A'$. This observation implies that instead of considering decision paths, one can consider their closures without affecting the values of the information gain. In terms of lattices this means that instead of the concept lattice $\underline{\mathfrak{B}}(\mathbb{X}_M\ D)$ one can consider the concept lattice of the context $\mathbb{K}_{+-} = (G_+ \cup G_-, M, I_+ \cup I_-)$. Another consequence of the invariance of IG w.r.t. closure is the following fact: If implication $m \to n$ holds in the context $\mathbb{K}_{+-} = (G_+ \cup G_-, M, I_+ \cup I_-)$, then an IG-based algorithm will not choose attribute n in the branch below chosen m and will not choose m in the branch below chosen $\overline{n}$.

Example 3. Consider the training set from Example 1. The decision tree obtained by the IG-based algorithm is given in Fig. 3. Note that attributes f and w has the same IG value (a similar tree with f at the root is also optimal), the IG-based algorithms usually take the first attribute with the same value of IG.

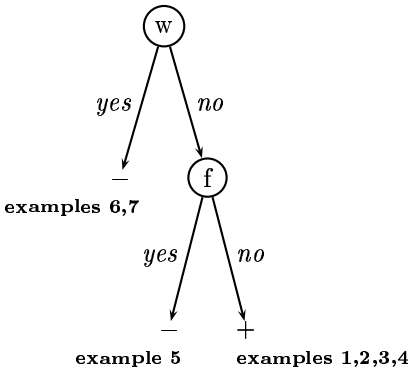


Fig. 3. A decision tree for the dataset from Example 1

The decision tree in Fig. 3 corresponds to three implications $\{\mathrm{w}\} \to -$, $\{\overline{\mathrm{w}}, \mathrm{f}\} \to -$, $\{\overline{\mathrm{w}}, \overline{\mathrm{f}}\} \to +$, such that closures of their premises make the corresponding negative and positive hypotheses for the second scaling from Example 1. Note that the hypothesis $\{\overline{\mathrm{w}}, \mathrm{f}\}''$ is not minimal, since there is a minimal hypothesis $\{\mathrm{f}\}''$ contained in it. The minimal hypothesis $\{\mathrm{f}\}''$ corresponds to a decision path of the mentioned IG-based tree with the attribute f at the root.

5 Version spaces vs. concept-based hypotheses

5.1 Version spaces

The term "version space" was coined by T. Mitchell [44–46] to denote a variety of models compatible with the training sample of positive and negative examples. Version spaces can be defined in different ways: e.g., in terms of sets of maximal and minimal elements [44, 45] or in terms of minimal elements and sets of negative examples [59]. They can also be defined in terms of some matching predicate. These representations are equivalent, however transformations from one into another are not always polynomially tractable. We will start from the representation with matching predicates, in terms slightly modified as compared with [44, 59], in order to avoid collision of FCA terminology and that of machine learning.

- An *example language* L_e (elsewhere also called *instance language*) by means of which the examples (instances) are described. This language describes a *set E of examples.*
- A *classifier language* L_c describing the possible classifiers (elsewhere called *concepts*). This language describes a set C of *classifiers.*
- A *matching predicate* $M(c, e)$ that defines if a classifier c does or does not *match* an example e: We have $M(c, e)$ iff e is an example of classifier c. The set of classifiers is (partially) ordered by a *subsumption order*: for $c_1, c_2 \in L_c$ the classifier c_1 subsumes c_2 or $c_1 \sqsupseteq c_2$ if c_1 corresponds to a more specific description and thus, covers less objects than c_2:

$$c_1 \sqsupseteq c_2 : \Longleftrightarrow \forall_{e \in E}\ M(c_1, e) \to M(c_2, e).$$

  The corresponding strict order $\sqsupset$ is called *proper subsumption.*
- Sets E_+ and E_- of *positive* and *negative examples* of a *target attribute* with $E_+ \cap E_- = \emptyset$. The target attribute is not explicitly given.
- *consistency predicate* $\mathrm{cons}(c)$:
  $\mathrm{cons}(c)$ holds if for every $e \in E_+$ the matching predicate $M(c, e)$ holds and for every $e \in E_-$ the negation $\neg M(c, e)$ holds. The set of all consistent classifiers is called the *version space*

$$\mathrm{VS}(L_c, L_e, M(c, e), E_+, E_-).$$

The learning problem is then defined as follows:

Given $L_c, L_e, M(c,e), E_+, E_-$.
Find the version space $\mathrm{VS}(L_c, L_e, M(c,e), E_+, E_-)$.

In the sequel, we shall usually fix L_c, L_e, and $M(c,e)$ and write $\mathrm{VS}(E_+, E_-)$ or even just VS for short. Version spaces are often considered in terms of *boundary sets* proposed in [45]. They can be defined if the language L_c is *admissible*, i.e., if every chain in the subsumption order has a minimal and a maximal element. In this case,

$$\mathrm{GVS}(L_c, L_e, M(c,e), E_+, E_-) := \mathrm{MIN}_{\sqsubseteq}(\mathrm{VS}) := \{c \in \mathrm{VS} \mid \neg\exists c_1 \in \mathrm{VS}\, c_1 \sqsubset c\},$$
$$\mathrm{SVS}(L_c, L_e, M(c,e), E_+, E_-) := \mathrm{MAX}_{\sqsubseteq}(\mathrm{VS}) := \{c \in \mathrm{VS} \mid \neg\exists c_1 \in \mathrm{VS}\, c \sqsubset c_1\}.$$

If a version space VS is fixed, we also use notation G(VS) and S(VS) for short. According to [46], the ideal result of learning a target attribute is the case where the version space consists of a single element. Otherwise, the target attribute is said to be *learned partially*.

The elements of the version space can be used as potential classifiers for the target attribute: A classifier $c \in \mathrm{VS}$ *classifies* an example positively if c matches e and negatively else. Then, all positive examples are classified positively, all negative examples are classified negatively, and undetermined examples may be classified either way. If it is assumed that E_+ and E_- carry sufficient information about the target attribute, we may expect that an undetermined example is likely to have the target attribute if it is classified positively by a large percentage of the version space (cf. [46]). We say that an example e is *α-classified* (or α%-classified) if no less than $\alpha \cdot |\,\mathrm{VS}\,|$ classifiers classify it positively. This means, e.g., that 100%-classification of e takes place if e is matched by all elements of SVS and negative classification of e (0%-classification) takes place if e is not matched by any element of GVS.

5.2 Version spaces in terms of Galois connections

As we showed in [21] the basic properties of general version spaces can easily be expressed with Galois connections, which underlie basic definitions of Formal Concept Analysis [23]. Consider the formal context (E, C, I), where E is the set of examples containing the disjoint sets of observed positive and negative examples: $E \supseteq E_+ \cup E_-$, $E_+ \cap E_- = \emptyset$, C is the set of classifiers and the relation I corresponds to the matching predicate $M(c,e)$: for $c \in C$, $e \in E$ the relation eIc holds iff $M(c,e) = 1$. The complementary relation, $\bar{I}$, corresponds to the negation: $e\bar{I}c$ holds iff $M(c,e) = 0$. As shown in [21]

$$\mathrm{VS}(E_+, E_-) = {E_+}^I \cap {E_-}^{\bar{I}}.$$

This characterization of version spaces implies immediately the property of *merging version spaces*, proved in [30]: For fixed L_c, L_e, $M(c,e)$ and two sets E_{+1}, E_{-1} and E_{+2}, E_{-2} of positive and negative examples one has

$$\mathrm{VS}(E_{+1} \cup E_{+2}, E_{-1} \cup E_{-2}) = \mathrm{VS}(E_{+1}, E_{-1}) \cap \mathrm{VS}(E_{+2}, E_{-2}).$$

This follows from the relation $(A \cup B)' = A' \cap B'$, which holds for a derivation operator $(\cdot)'$ of an arbitrary context.

The classifications produced by classifiers from the version space are characterized as follows. The set of all 100%-classified examples defined by the version space $\mathrm{VS}(E_+, E_-)$ is given by

$$({E_+}^I \cap {E_-}^{\bar{I}})^I.$$

In particular, if one of the following conditions is satisfied, then there cannot be any 100%-classified undetermined example:

1. $E_- = \emptyset$ and ${E_+}^{II} = E_+$,
2. $({E_+}^I \cap {E_-}^{\bar{I}})^I = E_+$.

The set of examples that are classified positively by at least one element of the version space $\mathrm{VS}(E_+, E_-)$ is given by

$$E \setminus ({E_+}^I \cap {E_-}^{\bar{I}})^{\bar{I}}.$$

5.3 Version spaces for classifier semilattices

In the preceding section we showed that the language of FCA and Galois connections is a convenient means for describing version spaces in general case, for unspecified order relation on the set of classifiers. Now we would like to consider a very important special case where the ordered set $(C, \leq)$ of classifiers given in terms of some language L_c makes a meet-semilattice w.r.t. $\wedge$ meet operation. This assumption is quite natural and realistic, e.g., classifiers given as logical formulas form a meet semilattice when the set of these formulas is closed under conjunction. Classifiers given as sets of attributes show the same effect if arbitrary subsets of attribute are allowed as classifiers, too. This also covers the case of attributes with values. In the setting of [46], for example, each attribute takes one of possible values, either constant or "wildcard" $*$, the latter being the shortcut for universal quantification over constant values of this attribute. Examples are given by conjunctions of attribute values. A classifier c matches example e if all attribute values of c that do not coincide with the corresponding values of e are wildcards.

In [21] we proved that in case where the classifiers, ordered by subsumption, form a complete semilattice, the version space is a complete subsemilattice for any sets of examples E_+ and E_-. For the case where the set of classifiers C makes a complete semilattice $(C, \sqcap)$, we can consider a *pattern structure* $(E, (C, \sqcap), \delta)$, where E is a set (of "examples"), δ is a mapping $\delta : E \to C$, $\delta(E) := \{\delta(e) \mid e \in E\}$. The subsumption order can be reconstructed from the semilattice operation: $c \sqsubseteq d \iff c \sqcap d = c$.

The version space may be empty, in which case there are no classifiers separating positive examples from negative ones. This happens, e.g., if there is a *hopeless* positive example (an outlier), by which we mean an element $e_+ \in E_+$ having a negative counterpart $e_- \in E_-$ such that every classifier which matches e_+ also matches e_-. An equivalent formulation of the hopelessness of e_+ is that $(e_+)^{\diamond\diamond} \cap E_- \neq \emptyset$.

Theorem 1 *Suppose that the classifiers, ordered by subsumption, form a complete meet-semilattice $(C, \sqcap)$, and let $(E, (C, \sqcap), \delta)$ denote the corresponding pattern structure. Then the following are equivalent:*

1. *The version space $\mathrm{VS}(E_+, E_-)$ is not empty.*
2. *$(E_+)^{\diamond\diamond} \cap E_- = \emptyset$.*
3. *There are no hopeless positive examples and there is a unique minimal positive hypothesis h_{min}.*

In this case, $h_{min} = (E_+)^{\diamond}$, and the version space is a convex set in the lattice of all pattern intents, ordered by subsumption, with maximal element h_{min}.

In case where conditions 1-3 are satisfied, the set of training examples is often referred to as *separable* in machine learning. The theorem gives access to an algorithm for generating the version space. For example, in [21] we use a modification of a standard **Next Closure** [23] algorithm for generating all formal concepts of a formal context to generate the version space as a convex set of the type described in Theorem 1.

According to [23] a subset $A \subseteq M$ can be defined as a *proper premise of an attribute* $m \in M$ if $m \notin A$, $m \in A''$ and for any $A_1 \subset A$ one has $m \notin A_1''$. In particular we can define a *positive proper premise* as a proper premise of the target attribute ω. In [21] we generalized this notion to include the possibility of the unknown value of a target attribute (for an undetermined example): $d \in L_c$ is a *positive proper predictor with respect to examples* E_+, E_-, and E_τ if the following conditions 1-3 are satisfied:

1. $d^\diamond \subseteq E_+ \cup E_\tau$,
2. $\exists g \in E_+ : g \in d^\diamond$ (or $d^\diamond \cap E_+ \neq \emptyset$),
3. $\forall d_1$ such that $d \sqsubseteq d_1$ and $d \neq d_1$, the relation $d_1^\diamond \not\subseteq E_+ \cup E_\tau$ holds.

In the case where $E_\tau = \emptyset$, satisfaction of condition 2 of the definition follows from condition 1 and a proper predictor is just a *proper premise* [23] of the target attribute. The set of all positive proper predictors for a pattern structure $\Pi = (E, (C, \sqcap), \delta)$ and sets of positive and negative examples E_+ and E_- will be denoted by $PP_+(\Pi, E_+, E_-)$.

By $H_+(\Pi, E_+, E_-)$ we denote the set of positive hypotheses, by $VS(\Pi, E_+, E_-)$ we denote the version space for the pattern structure $\Pi = (E, (C, \sqcap), \delta)$ and sets of positive and negative examples E_+ and E_-. Then the proper predictors and hypotheses are related to the boundaries of the version space as follows [21]:

(1) $PP_+(\Pi, E_+, E_-) = \mathrm{MAX}_\sqsubseteq(\bigcup_{F_+ \subseteq E_+} GVS(\Pi, F_+, E_-))$,

(2) $H_+(\Pi, E_+, E_-) = \bigcup_{F_+ \subseteq E_+} SVS(\Pi, F_+, E_-)$.

To sum up the relation of concept-based hypotheses with version spaces, we can say the following:

The major drawback of the version spaces where classifiers are defined syntactically is the very likely situation when - in case of too restrictive choice of the classifiers - there is no classifier that matches all positive examples (so-called "collapse of the version space"). This can easily happen for example when classifiers are just conjunctions of attribute value assignments and "wildcards" $*$, a case mentioned above. In other words: The situation discussed in Theorem 1, which presupposes that there are classifiers that match all positive and no negative examples, is too narrow. If the expressive power is increased syntactically, e.g., by introducing disjunction, then the version space tends to become trivial, while the most specific generalization of positive examples becomes "closer" to or just coincide with the set of positive examples. A syntactical restriction to conjunctions of k-term disjunctions was proposed in [57]. Hypotheses as we defined them in terms of concepts and patterns structures offer another sort of "context-restricted" disjunction: not all disjunctions are possible, but only those of minimal hypotheses (that are equivalent to certain conjunctions of attributes), which express similarities of examples. As for the relation of version spaces (with example descriptions given by consjunctions of attribute values) to decision trees, for any $c \in \mathrm{G(VS)}$ there is a decision tree with a path whose set of attributes coincide with c, but in general not all paths of decision trees are in G(VS).

A systematical elaboration of the idea of version spaces in logical terms by means of logical programming is going on in Inductive Logic Programming (ILP, see [47, 48]). Classifiers there are logical formulas with deducibility order on them. Standard inductive operators of ILP, called V- and W-operators, are based on the idea of inverting resolution. The application of these operators can be translated in the language of FCA generally in terms of association rules (partial implications) and for certain cases in terms of implications.

6 Applications of concept-based hypotheses

Starting from early 1980s JSM-hypotheses (equivalent to concept-based hypotheses from Section 2) were used in several applied domains, including biosciences, technical diagnostics, sociology, document dating, spam filtering and so on. Most numerous experiments were carried out in applied pharmacology or Structure-Activity Relationship domain, which deals with predicting biological activity of chemical compounds with known molecular structure. JSM-hypotheses were generated for antitumor [53], antibacterial, antileprous, hepatoprotective [7], plant growth-stimulating, cholesterase-inhibitine, toxic and carcinogenic activities, see review [4]. A freeware system QuDA [25, 26], which incorporates several data mining techniques also presents a possibility of generating JSM-hypotheses. JSM-hypotheses were used for making predictions at two international competitions: that for predcitive toxicology [28, 5] and that for spam filtering [10].

6.1 Competition on Predictive Toxicology

The program of a workshop on Predictive Toxicology Challenge (PTC) [28], (at the joint 12th European Conference on Machine Learning and the 5th European Conference on Principles of Knowledge Discovery in Databases) consisted in a competition of machine learning programs for generation of hypothetical causes of toxicity from positive and negative examples of toxicity. The organizers (Machine Learning groups of the Freiburg University, Oxford University, and University of Wales) together with toxicology experts (US Environmental Protection Agency, US National Institute of Environmental and Health Standards) provided the participants with training and test samples.

The training sample consisted of descriptions of 185 molecular graphs of 409 chemical compounds with indication of whether a compound is toxic or not for a particular sex/species group out of four possible groups: male mice, female mice, male rats and female rats. For each group there were about 120 to 150 positive examples and 190 to 230 negative examples of toxicity with indication of whether a substance is toxic for four sex/species groups: {male, female} $\times$ {mice, rats} (for some groups a substance can be neither a positive nor a negative example because of ambiguous laboratory results). The test sample provided by the Food and Drug Administration (FDA) consisted of 185 substances for which forecasts of toxicity should be made (actually, (non)toxicity of substances was known to organizers). Twelve research groups (world-wide) participated in PTC, each with up to 4 prediction models for every sex/species group.

The competition consisted of the following stages: 1. Encoding of chemical structures in terms of attributes, 2. Generation of classification rules, 3. Prediction by means of classification rules. All results of each stage were made public by the organizers. In particular, encodings of chemical structures made by a participant were made available to all participants. The evaluation was ROC diagrams where each predictive model was represented in a two-dimension space with coordinates related to the rate of (in)correctly predicted toxicity: percentage of substances from the test sample with correctly predicted toxicity (true positive classification rate) and that of incorrectly predicted (false positive classification rate).

The following learning models were used by the participants: structural regression tree (STR) [33] based on combination of statistical methods for constructing regression trees and inductive logic programming (ILP); tree induction for predictive toxicology (TIPT) [3], which is an adaptation of the standard C4.5 algorithm, ILP algorithm PROGOL [47] that realizes inverse entailment for generalizing positive examples w.r.t. a partial domain theory; LRD model based on Distill algorithm [58], which is a combination of the method of Disjunctive Version Spaces (DiVS) [57] with

the method of stochastic choice of certain model parameters; the OUCL-2 model used the C4.5 algorithm for construction decision trees, where some attributes were constructed by means of ILP and WARMR methods [32] (the latter is a modernization of the Apriori algorithm [1] by generating DATALOG queries levelwise up to a certain level, choosing those satisfied by a sufficient number of examples); OAI model used a combination of rules generated by C4.5 with Bayes classification followed by voting; LEU3 model based on the Inductive constraint logic (ICL) algorithm [12], which used a mutagenesis theory preconstructed by PROGOL; LEU1 model used an algorithm for inducing decision trees; LEU2 model based on the MACCENT system with the use of association rules found by WARMR. MACCENT model [11] also uses DATALOG queries, first finding constraints for conditional distributions for the membership to positive and negative classes and then finding the distributon with the use of the maximum entropy principle.

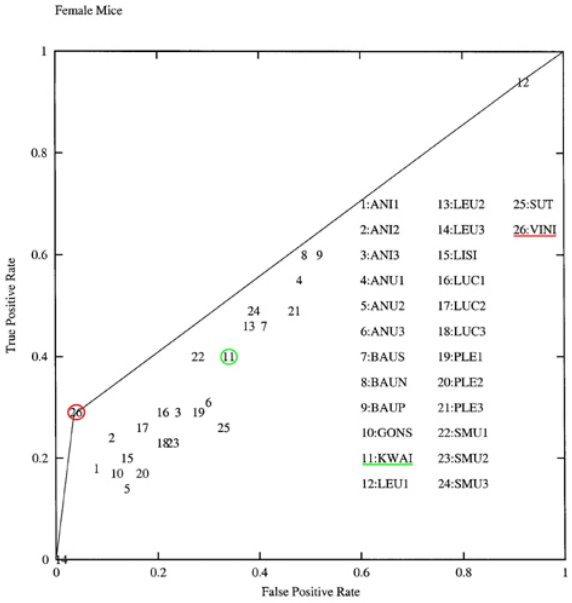


Fig. 4. ROC-diagram for classifications in the `female mice` group

As measured by ROC diagrams, the performance of the learning program from VINITI (Moscow) [5] based on JSM-hypotheses defined in Section 2 turned out to be Pareto-optimal (ROC diagrams allow for several incomparable "best" results) for all sex/species groups, see e.g. model number 26 in Fig. 4, among all classification rules generated by learning models participating in the competition in terms of the relative number of false and true positive classifications made by hypotheses generated by a learning program (see [28] for details).

6.2 Spam filtering

The first successful applications of concept-based hypotheses for filtering spam was reported in [15]. In April-May 2003 Technical University Chemnitz, European Knowledge Discovery Network, and PrudSys AG organized the Data Mining Cup (DMC) competition for students specializing in Machine Learning [10]. 514 participants from 199 Universities of 38 countries received training datasets with 8000 e-mail messages some part of which (39%) was qualified as spam (positive examples) and the rest (61%) as nonspam (negative examples). The test dataset contained 11177 messages. Both datasets were described by 833 attributes, including

832 binary ones and one numeric one. The only numeric attribute (ID) reflected the (unique) incoming number of the e-mail within the company that provided the data. Thus no two different e-mails had the same value for this attribute.

The participants were to generate a classifier for distinguishing spam from not spam by using various machine learning techniques. An important condition of the competition was that the error of classifying a nonspam message from the test set as a spam one should not be greater than 1%. There were only 74 participants whose learning models did not exceed this level. The learning models were than ranked according to the rate of incorrect classification of spam messages as nonspam ones.

Seven students from Computer Science and Mathematics Faculties of the Darmstadt University of Technology took part in the competition. Three of the students obtained solutions within the best 20 in the competition. The sixth place (the best among the Darmstadt group) was taken by a solution of F. Hutter, which used "Naive Bayes" approach with boosting and when the confidence level of the classification (i.e., the rate of correct classifications in the training sample itself) was less than 90% utilized concept-based hypotheses with support (i.e., number of examples in the hypothesis extent) ≥ 20. When hypotheses refused to classify a message from the test set, the model used classifiers based on majority votes among decision trees, the Naive Bayes classifier and a neural network. The sixteenth and seventeenth places in the competition were also taken by the students from Darmstadt. Their models combined concept-based hypotheses, decision trees and Naive Bayes appoaches using the majority vote strategy.

During the data preparation stage, the participants from Darmstadt "cleaned off" the ID attribute (the serial number of an e-mail message). It turned out later that the ID attribute, unique for each e-mail, implicitly indicated the time when the e-mail was received: the last 4000 e-mails were spam (obtained on holidays when business corespondence was temporarile canceled). 5 most successful models used this to turn ID into time attribute with few values (roughly, "before holidays" and "during the holidays"). The results obtained with concept-based hypotheses could have been even better if this consideration had been taken into account at the preprocessing stage.

7 Conclusions

The application of the lattice theory and FCA in machine learning shows that the basic notions in lattice-based learning are that of a concept, concept intent (closed itemset), implication, and association rule (partial implication). We presented several machine learning models from the concept lattice viewpoint, including version spaces, decision trees, and JSM- (or concept-based) hypotheses. It is shown that concept-based hypotheses tend to be "more cautious" than those obtained by decision trees. On the other hand, they introduce a kind of restricted disjunction (over a certain subset of concept intents) for purely conjunctive version spaces. A next interesting link between FCA and machine learning will be relating FCA-based learning with methods of ILP. We also discussed algorithmic problems of concept-based and pattern-based hypotheses, as well as applications of concept-based hypotheses in predictive toxicology and spam filtering.

Acknowledgments

I thank Bernhard Ganter, Peter Grigoriev, Sergei Obiedkov, and Mikhail Samokhin for heplful discussions and attention to this work.

References

1. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo, Fast Discovery of Association Rules, in *Advances in Knowledge Discovery and Data Mining*, 1996, 307-328.
2. F. Baader and R. Molitor, Building and Structuring Description Logic Knowledge Spaces Using Least Common Subsumers and Concept Analysis, in *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 292-305.
3. D. Bahler and D.W. Bristol, The induction of rules for predicting chemical carcerogenesis in rodents, in Intelligent Systems for Molecular Biology, L. Hunter, D. Searls, J. Shavlik, Eds., Menlo Park, CA, AAAI/MIT Press, 1993, 29-37.
4. V.G. Blinova, Results of Application of the JSM-method of Hypothesis Generation to Problems of Analyzing the Relation "Structure of a Chemical Compound - Biological Activity," *Autom. Docum. Math. Ling.*, vol. 29, no. 3, pp. 26-33, 1995.
5. V.G. Blinova, D.A. Dobrynin, V.K. Finn, S.O. Kuznetsov, and E.S. Pankratova, Toxicology analysis by means of the JSM-method, *Bioinformatics* 2003, vol. 19, pp. 1201-1207.
6. M. Botta, A. Giordana, L. Saitta, and M. Sebag, Relational Learning as Search in a Critical Region, *Journal of Machine Learning Research*, 2003, **4**, 431-463.
7. A.P. Budunova, V.V. Poroikov, V.G. Blinova, and V.K. Finn, The JSM-method of hypothesis generation: Application for analysis of the relation "Structure - hepatoprotective detoxifying activity", Nauchno-Tekhnicheskaya Informatsiya, no. 7, pp.12-15, 1993 [in Russian].
8. C. Carpineto, G. Romano, A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval, *Machine Learning*, 1996, Vol. 24, pp. 95-122.
9. L. Chaudron and N. Maille, Generalized Formal Concept Analysis, in *Proc. 8th Int. Conf.on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 357-370.
10. Data Mining Cup (DMC), http://www.data-mining-cup.de
11. L. Dehaspe and L. De Raedt, Mining Association Rules in Multiple Relations, Proc. 7th Int. Workshop on Inductive Logic Programming, LNAI, vol. 1297, 1997, 125-132.
12. L. De Raedt, W. van Laer, Inductive Constraint Logic, Proc. 5th Workshop on Algorithmic Learning Theory, LNCS, vol. 997, 1995, 80-94.
13. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From Data Mining to Knowledge Discovery: An Overview, In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Eds. Advances in Knowledge Discovery and Data Mining, AAAI Press, 1996, 3-33.
14. S. Férré and O. Ridoux, A Logical Generalization of Formal Concept Analysis, in *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000.
15. S. Férré and O. Ridoux, The Use of Associative Concepts in the Incremental Building of a Logical Context in *Proc. 10th Int. Conf. on Conceptual Structures, ICCS'2002*, U. Priss, D. Corbet, G. Angelova, Eds., Lecture Notes in Artificial Intelligence, **2393**, 2002, 299-313.
16. V.K. Finn, On Machine-Oriented Formalization of Plausible Reasoning in the Style of F. Backon–J. S. Mill, *Semiotika Informatika*, **20** (1983) 35-101 [in Russian].
17. V.K. Finn, Plausible Reasoning in Systems of JSM Type, *Itogi Nauki i Tekhniki, Seriya Informatika*, **15**, 54-101, 1991 [in Russian].
18. J.G. Ganascia, CHARADE: A rule system learning system, In: *Proc. of the 10th International Joint Conference on Artificial Intelligence (IJCAI-87)*, Milan, Italy, August 23-28 (1987), 345-347, 1987.
19. B. Ganter and S. Kuznetsov, Formalizing Hypotheses with Concepts, *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'00*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 342-356.
20. B. Ganter and S. Kuznetsov, Pattern Structures and Their Projections, *Proc. 9th Int. Conf. on Conceptual Structures, ICCS'01*, G. Stumme and H. Delugach, Eds., Lecture Notes in Artificial Intelligence, **2120** 2001, pp. 129-142.

21. B. Ganter and S.O. Kuznetsov, Hypotheses and Version Spaces, *Proc. 10th Int. Conf. on Conceptual Structures, ICCS'01*, A.de Moor, W. Lex, and B.Ganter, Eds., Lecture Notes in Artificial Intelligence, **2746** 2003, pp. 83-95.
22. B. Ganter and K. Reuter, Finding all closed sets: a general approach, *Order*, **8**, 283-290, 1991.
23. B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundations*, Springer, 1999.
24. M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York, Freeman, 1979.
25. P.A. Grigoriev and S.A. Yevtushenko, Elements of an Agile Discovery Environment, in Proc. 6th International Conference on Discovery Science (DS 2003), Eds. G. Grieser, Y. Tanaka, and A. Yamamoto, Lecture Notes in Artifical Intelligence, **2843**, 309-316, 2003.
26. P.A. Grigoriev, S.A.Yevtushenko, and G.Grieser, *QuDA, a data miner's discovery enviornment*, Tehcnical Report AIDA 03 06, FG Intellektik, FB Informatik, Technische Universitaet Darmstadt, September 2003, http://www.intellektik.informatik.tu-darmstadt.de/ peter/QuDA.pdf.
27. C.A. Gunter, T.-H. Ngair, D. Subramanian, The Common Order-Theoretic Structure of Version Spaces and ATMSs, *Artificial Intelligence* **95**, 357-407, 1997.
28. C. Helma, R.D. King, S. Kramer, A. Srinvasan, Proc. of the Workshop on Predictive Toxicology Challegnge at the 5th Conference on Data Mining and Knowledge Discovery (PKDD'01), Freiburg (Germany), 2001, September 7, http://www.predictive-toxicology.org/ptc/
29. J. Hereth, G. Stumme, R. Wille, and U. Wille, Conceptual Knowledge Discovery and Data Analysis, *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'98*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 421-437.
30. H. Hirsh, Generalizing Version Spaces, *Machine Learning* **17**, 5-46, 1994.
31. H. Hirsh, N. Mishra, and L. Pitt, Version Spaces Without Boundary Sets, in *Proc. of the 14th National Conference on Artificial Intelligence (AAAI97)*, AAAI Press/MIT Press, 1997.
32. R.D. King, A. Srinivasan, and L. Dehaspe, WARMR: A Data Mining tool for chemical data, J. Computer-Aided Molecular Design, vol. 15, 2001, 173-181.
33. S. Kramer, Structural Regression Trees, Proc. *13th National Conference on Artificial Intelligence* (AAAI-96), 812-819, AAAI Press, 1996.
34. S.O. Kuznetsov, Interpretation on Graphs and Complexity Characteristics of a Search for Specific Patterns, *Nauchn. Tekh. Inf., Ser. 2 (Automat. Document. Math. Linguist.)* no. 1, 23-27, 1989.
35. S.O. Kuznetsov, JSM-method as a machine learning method, *Itogi Nauki i Tekhniki, ser. Informatika*, vol. 15, pp.17-50, 1991 [in Russian].
36. S.O. Kuznetsov and V.K. Finn, On a model of learning and classification based on similarity operation, *Obozrenie Prikladnoi i Promyshlennoi Matematiki* **3**, no. 1, 66-90, 1996 [in Russian].
37. S.O. Kuznetsov, Learning of Simple Conceptual Graphs from Positive and Negative Examples. In: J. Zytkow, J. Rauch (eds.), Proc. *Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD'99*, Lecture Notes in Artificial Intelligence, **1704**, pp. 384-392, 1999.
38. S.O. Kuznetsov, On Computing the Size of a Lattice and Related Decision Problems, Order, 2001, **18**(4), pp. 313-321.
39. S.O. Kuznetsov, S.A. Obiedkov, Comparing performance of algorithms for generating concept lattices, *J. Exp. Theor. Artif. Intell.*, 2002, vol. 14, nos. 2-3, pp. 189-216.
40. M. Liquiere and J. Sallantin, Structural Machine Learning with Galois Lattice and Graphs, *Proc. Int. Conf. Machine Learning ICML'98*, 1998.
41. D. Maier, *The Theory of Relational Databases*, Comput. Sci. Press, Potomac, MD, 1983.
42. M. Luxenburger, Implications partielle dans un contexte, *Math. Sci. Hum.*, 1991.
43. P. Njiwoua, E. Mefu Nguifo, Forwarding the choice of bias. LEGAL-F: Using Feature Selection to Reduce Complexity of LEGAL, in *Proc. of BENELEARN-97*, 1997, pp. 89-98.

44. T. Mitchell, Version Space: An Approach to Concept Learning, PhD thesis, Stanford University, 1978.
45. T. Mitchell, Generalization as Search, *Artificial Intelligence* **18**, no. 2, 1982.
46. T. Mitchell, *Machine Learning*, The McGraw-Hill Companies, 1997.
47. S. Muggleton, Inverse Entailment and Progol, *New Generation Computing*, Special Issue on Inductive Logic Programming, vol. 13 (3-4), 1995, 245-286.
48. S.-H. Nienhuys-Cheng and R. de Wolf, *Foundations of Inductive Logic Programming*, Lecture Notes in Artificial Intelligence, vol. 1228, 1997.
49. G.D. Oosthuizen and D.R. McGregor, Induction Through Knowledge Normalization, in *Proc. 8th. European Conference on Artificial Intelligence*, Munich, 1988.
50. G.D. Oosthuizen, The use of of a lattice in Knowledge Processing, PhD Thesis, University of Strathclyde, Glasgow, 1988.
51. G.D. Oosthuizen, The application of Concept Lattices to Machine Learning, Univeristy of Pretoria, Tech. Rep. CSTR 94/01, 1994.
52. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, Efficient Minining of Association Rules Based on Using Closed Itemset Lattices, *J. Inf. Systems*, **24**, 1999, pp. 25-46.
53. D.V. Popov, V.G. Blinova and E.S. Pankratova, Drug deisgn. JSM-Method of hypothesis generation for predicting antitumor activity and toxic effects forecasts with respect to plant products, Proc. *5th Int. Conf. on Chemistry and Biotechnology of Biologically Active Natural Products*, Varna (Bulgaria), September 18-23, 1989, vol. 2, pp. 437-440.
54. J.R. Quinlan, Induction on Decision Trees, *Machine Learning*, **1**, No. 1, 81-106 (1986).
55. M. Sahami, Learning Classification Rules Using Lattices, *Proc. 8th European Conference on Machine Learning*, N. Lavrac, S. Wrobel, Eds., pp. 343-346, 1995.
56. M. Sebag, Using Constraints to Building Version Spaces, in L. de Raedt and F. Bergadano, eds., *Proc. of the European Conference on Machine Learning (ECML-94)*, pp. 257-271, Springer, 1994.
57. M. Sebag, Delaying the Choice of Bias: A Disjunctive Version Space Approach, in L. Saitta ed., *Proc. of the 13th International Conference on Machine Learning*, pp. 444-452, Morgan Kaufmann, 1996.
58. M. Sebag, C. Rouveroil, Tractable induction and classification in first-order logic via stochastic matching, Proc. *15th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1997, 888-893.
59. E.N. Smirnov and P.J. Braspenning, Version Space Learning with Instance-Based Boundary Sets, in H. Prade, ed., Proc. of *13th European Conference on Artificial Intelligence*, J. Wiley, Chichester, 460-464, 1998.
60. A. Srinivasan, S.H. Muggleton, M.J.E. Sternberg, and R.D. King, Theories for mutagenicity: a study in first order and feature-based induction, *Artificial Intelligence*, 1996, **85**, 277-299.
61. G. Stumme, R. Wille, U. Wille, Conceptual Knowledge Discovery in Databases Using Formal Concept Analysis Methods, In: J. Zytkow, M. Quafofou, ed., Proc. *2nd European Symposium on PKDD'98. Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Artificial Intelligence, **1510**, Springer 1998, 450-458.
62. R. Wille, Restructuring Lattice Theory: an Approach Based on Hierarchies of Concepts, In: *Ordered Sets* (I. Rival, ed.), Reidel, Dordrecht–Boston, 445-470, 1982.

References

1. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo, Fast Discovery of Association Rules, in *Advances in Knowledge Discovery and Data Mining*, 1996, 307-328.
2. F. Baader and R. Molitor, Building and Structuring Description Logic Knowledge Spaces Using Least Common Subsumers and Concept Analysis, in *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 292-305.
3. D. Bahler and D.W. Bristol, The induction of rules for predicting chemical cancerogenesis in rodents, in Intelligent Systems for Molecular Biology, L. Hunter, D. Searls, J. Shavlik, Eds., Menlo Park, CA, AAAI/MIT Press, 1993, 29-37.

4. V.G. Blinova, Results of Application of the JSM-method of Hypothesis Generation to Problems of Analyzing the Relation "Structure of a Chemical Compound - Biological Activity," *Autom. Docum. Math. Ling.*, vol. 29, no. 3, pp. 26-33, 1995.
5. V.G. Blinova, D.A. Dobrynin, V.K. Finn, S.O. Kuznetsov, and E.S. Pankratova, Toxicology analysis by means of the JSM-method, *Bioinformatics* 2003, vol. 19, pp. 1201-1207.
6. M. Botta, A. Giordana, L. Saitta, and M. Sebag, Relational Learning as Search in a Critical Region, *Journal of Machine Learning Research*, 2003, **4**, 431-463.
7. A.P. Budunova, V.V. Poroikov, V.G. Blinova, and V.K. Finn, The JSM-method of hypothesis generation: Application for analysis of the relation "Structure - hepatoprotective detoxifying activity", Nauchno-Tekhnicheskaya Informatsiya, no. 7, pp.12-15, 1993 [in Russian].
8. C. Carpineto, G. Romano, A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval, *Machine Learning*, 1996, Vol. 24, pp. 95-122.
9. L. Chaudron and N. Maille, Generalized Formal Concept Analysis, in *Proc. 8th Int. Conf.on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 357-370.
10. Data Mining Cup (DMC), http://www.data-mining-cup.de
11. L. Dehaspe and L. De Raedt, Mining Association Rules in Multiple Relations, Proc. 7th Int. Workshop on Inductive Logic Programming, LNAI, vol. 1297, 1997, 125-132.
12. L. De Raedt, W. van Laer, Inductive Constraint Logic, Proc. 5th Workshop on Algorithmic Learning Theory, LNCS, vol. 997, 1995, 80-94.
13. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From Data Mining to Knowledge Discovery: An Overview, In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Eds. Advances in Knowledge Discovery and Data Mining, AAAI Press, 1996, 3-33.
14. S. Férré and O. Ridoux, A Logical Generalization of Formal Concept Analysis, in *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'2000*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000.
15. S. Férré and O. Ridoux, The Use of Associative Concepts in the Incremental Building of a Logical Context in *Proc. 10th Int. Conf. on Conceptual Structures, ICCS'2002*, U. Priss, D. Corbet, G. Angelova, Eds., Lecture Notes in Artificial Intelligence, **2393**, 2002, 299-313.
16. V.K. Finn, On Machine-Oriented Formalization of Plausible Reasoning in the Style of F. Backon–J. S. Mill, *Semiotika Informatika*, **20** (1983) 35-101 [in Russian].
17. V.K. Finn, Plausible Reasoning in Systems of JSM Type, *Itogi Nauki i Tekhniki, Seriya Informatika*, **15**, 54-101, 1991 [in Russian].
18. J.G. Ganascia, CHARADE: A rule system learning system, In: *Proc. of the 10th International Joint Conference on Artificial Intelligence (IJCAI-87)*, Milan, Italy, August 23-28 (1987), 345-347, 1987.
19. B. Ganter and S. Kuznetsov, Formalizing Hypotheses with Concepts, *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'00*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 342-356.
20. B. Ganter and S. Kuznetsov, Pattern Structures and Their Projections, *Proc. 9th Int. Conf. on Conceptual Structures, ICCS'01*, G. Stumme and H. Delugach, Eds., Lecture Notes in Artificial Intelligence, **2120** 2001, pp. 129-142.
21. B. Ganter and S.O. Kuznetsov, Hypotheses and Version Spaces, *Proc. 10th Int. Conf. on Conceptual Structures, ICCS'01*, A.de Moor, W. Lex, and B.Ganter, Eds., Lecture Notes in Artificial Intelligence, **2746** 2003, pp. 83-95.
22. B. Ganter and K. Reuter, Finding all closed sets: a general approach, *Order*, **8**, 283-290, 1991.
23. B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundations*, Springer, 1999.
24. M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York, Freeman, 1979.
25. P.A. Grigoriev and S.A. Yevtushenko, Elements of an Agile Discovery Environment, in Proc. 6th International Conference on Discovery Science (DS 2003), Eds. G. Grieser, Y. Tanaka, and A. Yamamoto, Lecture Notes in Artifical Intelligence, **2843**, 309-316, 2003.

26. P.A. Grigoriev, S.A.Yevtushenko, and G.Grieser, *QuDA, a data miner's discovery enviornment*, Tehcnical Report AIDA 03 06, FG Intellektik, FB Informatik, Technische Universitaet Darmstadt, September 2003, http://www.intellektik.informatik.tu-darmstadt.de/ peter/QuDA.pdf.
27. C.A. Gunter, T.-H. Ngair, D. Subramanian, The Common Order-Theoretic Structure of Version Spaces and ATMSs, *Artificial Intelligence* **95**, 357-407, 1997.
28. C. Helma, R.D. King, S. Kramer, A. Srinvasan, Proc. of the Workshop on Predictive Toxicology Challegnge at the 5th Conference on Data Mining and Knowledge Discovery (PKDD'01), Freiburg (Germany), 2001, September 7, http://www.predictive-toxicology.org/ptc/
29. J. Hereth, G. Stumme, R. Wille, and U. Wille, Conceptual Knowledge Discovery and Data Analysis, *Proc. 8th Int. Conf. on Conceptual Structures, ICCS'98*, G. Mineau and B. Ganter, Eds., Lecture Notes in Artificial Intelligence, **1867**, 2000, pp. 421-437.
30. H. Hirsh, Generalizing Version Spaces, *Machine Learning* **17**, 5-46, 1994.
31. H. Hirsh, N. Mishra, and L. Pitt, Version Spaces Without Boundary Sets, in *Proc. of the 14th National Conference on Artificial Intelligence (AAAI97)*, AAAI Press/MIT Press, 1997.
32. R.D. King, A. Srinivasan, and L. Dehaspe, WARMR: A Data Mining tool for chemical data, J. Computer-Aided Molecular Design, vol. 15, 2001, 173-181.
33. S. Kramer, Structural Regression Trees, Proc. *13th National Conference on Artificial Intelligence* (AAAI-96), 812-819, AAAI Press, 1996.
34. S.O. Kuznetsov, Interpretation on Graphs and Complexity Characteristics of a Search for Specific Patterns, *Nauchn. Tekh. Inf., Ser. 2 (Automat. Document. Math. Linguist.)* no. 1, 23-27, 1989.
35. S.O. Kuznetsov, JSM-method as a machine learning method, *Itogi Nauki i Tekhniki, ser. Informatika*, vol. 15, pp.17-50, 1991 [in Russian].
36. S.O. Kuznetsov and V.K. Finn, On a model of learning and classification based on similarity operation, *Obozrenie Prikladnoi i Promyshlennoi Matematiki* **3**, no. 1, 66-90, 1996 [in Russian].
37. S.O. Kuznetsov, Learning of Simple Conceptual Graphs from Positive and Negative Examples. In: J. Zytkow, J. Rauch (eds.), Proc. *Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD'99*, Lecture Notes in Artificial Intelligence, **1704**, pp. 384-392, 1999.
38. S.O. Kuznetsov, On Computing the Size of a Lattice and Related Decision Problems, Order, 2001, **18**(4), pp. 313-321.
39. S.O. Kuznetsov, S.A. Obiedkov, Comparing performance of algorithms for generating concept lattices, *J. Exp. Theor. Artif. Intell.*, 2002, vol. 14, nos. 2-3, pp. 189-216.
40. M. Liquiere and J. Sallantin, Structural Machine Learning with Galois Lattice and Graphs, *Proc. Int. Conf. Machine Learning ICML'98*, 1998.
41. D. Maier, *The Theory of Relational Databases*, Comput. Sci. Press, Potomac, MD, 1983.
42. M. Luxenburger, Implications partielle dans un contexte, *Math. Sci. Hum.*, 1991.
43. P. Njiwoua, E. Mefu Nguifo, Forwarding the choice of bias. LEGAL-F: Using Feature Selection to Reduce Complexity of LEGAL, in *Proc. of BENELEARN-97*, 1997, pp. 89-98.
44. T. Mitchell, Version Space: An Approach to Concept Learning, PhD thesis, Stanford University, 1978.
45. T. Mitchell, Generalization as Search, *Artificial Intelligence* **18**, no. 2, 1982.
46. T. Mitchell, *Machine Learning*, The McGraw-Hill Companies, 1997.
47. S. Muggleton, Inverse Entailment and Progol, *New Generation Computing*, Special Issue on Inductive Logic Programming, vol. 13 (3-4), 1995, 245-286.
48. S.-H. Nienhuys-Cheng and R. de Wolf, *Foundations of Inductive Logic Programming*, Lecture Notes in Artificial Intelligence, vol. 1228, 1997.
49. G.D. Oosthuizen and D.R. McGregor, Induction Through Knowledge Normalization, in *Proc. 8th. European Conference on Artificial Intelligence*, Munich, 1988.
50. G.D. Oosthuizen, The use of of a lattice in Knowledge Processing, PhD Thesis, University of Strathclyde, Glasgow, 1988.

51. G.D. Oosthuizen, The application of Concept Lattices to Machine Learning, Univeristy of Pretoria, Tech. Rep. CSTR 94/01, 1994.
52. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, Efficient Minining of Association Rules Based on Using Closed Itemset Lattices, *J. Inf. Systems*, **24**, 1999, pp. 25-46.
53. D.V. Popov, V.G. Blinova and E.S. Pankratova, Drug deisgn. JSM-Method of hypothesis generation for predicting antitumor activity and toxic effects forecasts with respect to plant products, Proc. *5th Int. Conf. on Chemistry and Biotechnology of Biologically Active Natural Products*, Varna (Bulgaria), September 18-23, 1989, vol. 2, pp. 437-440.
54. J.R. Quinlan, Induction on Decision Trees, *Machine Learning*, **1**, No. 1, 81-106 (1986).
55. M. Sahami, Learning Classification Rules Using Lattices, *Proc. 8th European Conference on Machine Learning*, N. Lavrac, S. Wrobel, Eds., pp. 343-346, 1995.
56. M. Sebag, Using Constraints to Building Version Spaces, in L. de Raedt and F. Bergadano, eds., *Proc. of the European Conference on Machine Learning (ECML-94)*, pp. 257-271, Springer, 1994.
57. M. Sebag, Delaying the Choice of Bias: A Disjunctive Version Space Approach, in L. Saitta ed., *Proc. of the 13th International Conference on Machine Learning*, pp. 444-452, Morgan Kaufmann, 1996.
58. M. Sebag, C. Rouveroil, Tractable induction and classification in first-order logic via stochastic matching, Proc. *15th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1997, 888-893.
59. E.N. Smirnov and P.J. Braspenning, Version Space Learning with Instance-Based Boundary Sets, in H. Prade, ed., Proc. of *13th European Conference on Artificial Intelligence*, J. Wiley, Chichester, 460-464, 1998.
60. A. Srinivasan, S.H. Muggleton, M.J.E. Sternberg, and R.D. King, Theories for mutagenicity: a study in first order and feature-based induction, *Artificial Intelligence*, 1996, **85**, 277-299.
61. G. Stumme, R. Wille, U. Wille, Conceptual Knowledge Discovery in Databases Using Formal Concept Analysis Methods, In: J. Zytkow, M. Quafofou, ed., Proc. *2nd European Symposium on PKDD'98. Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Artificial Intelligence, **1510**, Springer 1998, 450-458.
62. R. Wille, Restructuring Lattice Theory: an Approach Based on Hierarchies of Concepts, In: *Ordered Sets* (I. Rival, ed.), Reidel, Dordrecht–Boston, 445-470, 1982.