Patterns via Clustering as a Data Mining Tool

Lars Lumpe[1] and Stefan E. Schmidt[2]

Institut für Algebra,
Technische Universität Dresden
larslumpe@gmail.com[1], midt1@msn.com[2]

Abstract. Research shows that pattern structures are a useful tool for analyzing complex data. In our paper, we present a new general framework of using pattern structures as a data mining tool and as an application of the new framework we show a way to handle a classification problem of red wines.

1 Introduction

Pattern structures within the framework of formal concept analysis have been introduced in [4]. Since then they have been the subject of further investigations like in [2, 9, 11, 10, 12] and have turned out to be a useful tool for analyzing various real-world applications (cf. [4–8]). In this paper, we want to present a new application. But first we will introduce a general way to construct a pattern structure. Then we are going to use several clustering algorithm to find important pattern in it. Further we will use this pattern to build a model to solve a classification problem. In particular, we will take the dataset from [3] and train an algorithm to predict the quality of red wines.

2 Preliminaries

We start with some general definitions:

Definition 1 (restriction)**.** Let $\mathbb{P} := (P, \leq_{\mathbb{P}})$ be a poset, then for every set U the poset

$$\mathbb{P} \mid U := (U, \leq_{\mathbb{P}} \cap (U \times U))$$

is called the **restriction** of $\mathbb{P}$ onto U.

If we consider a poset of patterns, it often arises as a dual of a given poset.

Definition 2 (opposite or dual poset)**.** Let $\mathbb{P} := (P, \leq_{\mathbb{P}})$ be a poset. Then we call

$$\mathbb{P}^{op} := (P, \geq_{\mathbb{P}}) \text{ with } \geq_{\mathbb{P}} := \{(q, p) \in P \times P \mid p \leq q\}$$

the **opposite or dual** of $\mathbb{P}$.

Definition 3 (Interval Poset)**.** Let $\mathbb{P} := (P, \leq_{\mathbb{P}})$ be a poset. Then we call

$$\mathrm{Int}\mathbb{P} := \{[p, q]_{\mathbb{P}} \mid p, q \in P\} \text{ with } [p, q]_{\mathbb{P}} := \{t \in P \mid p \leq_{\mathbb{P}} t \leq_{\mathbb{P}} q\}$$

the **set of all intervals** of $\mathbb{P}$, and we refer to $\mathbb{I}\mathrm{nt}\mathbb{P} := (\mathrm{Int}\mathbb{P}, \subseteq)$ as the interval poset of $\mathbb{P}$.

Remark: (i) Let $\mathbb{P}$ be a poset. Then $\mathbb{I}\mathrm{nt}\mathbb{P}$ is a lower bounded poset, that is, $\emptyset$ is the least element of $\mathbb{I}\mathrm{nt}\mathbb{P}$, provided $\mathbb{P}$ has at least 2 elements. Furthermore if $\mathbb{P}$ is a (complete) lattice than so is $\mathbb{I}\mathrm{nt}\mathbb{P}$.
(ii) If A is a set of attributes than $(\mathbb{R}^A, \leq) := (\mathbb{R}, \leq)^A$ is a lattice. With (i) it follows that $\mathbb{I}\mathrm{nt}(\mathbb{R}^A, \leq)$ is a lower bounded lattice.

Definition 4 (kernel operator)**.** A **kernel operator** on a poset $\mathbb{P} := (P, \leq)$ is a map $\gamma : P \to P$ such that for all $x, y \in P$:

$$kx \leq y \Leftrightarrow kx \leq ky \tag{1}$$

A subset ζ of P is called a **kernel system** in $\mathbb{P}$ if for every $x \in P$ the restriction of $\mathbb{P}$ onto $\{t \in \zeta \mid t \leq x\}$ has a greatest element.

Remark: A closure operator on $\mathbb{P} := (P, \leq)$ is defined as a kernel operator on $\mathbb{P}^d$, and a closure system in $\mathbb{P}$ is defined as a kernel system in $\mathbb{P}^d$.

The main definitions of FCA are based on a binary relation I between a set of so called objects G and a set of so called attributes M. However, in many real-world knowledge discovery problems, researchers have to deal with data sets that are much more complex than binary data tables. In our case, there was a set of numerical attributes, such as the amount of acetic acid, density, the amount of salt, etc., describing the quality of a red wine. To deal with this kind of data, pattern structures are a useful tool.

Definition 5 (pattern setup, pattern structure)**.** A triple $\mathcal{P} = (G, \mathbb{D}, \delta)$ is a **pattern setup** if G is a set, $\mathbb{D} = (D, \sqsubseteq)$ is a poset, and $\delta : G \to D$ is a map. In case every subset of $\delta G := \{\delta g \mid g \in G\}$ has an infimum in $\mathbb{D}$, we will refer to $\mathcal{P}$ as **pattern structure**.

For pattern structures, an important complexity reduction is often provided by so-called o-projections:

Proposition 1 (o-projection)**.** For a pattern structure $\mathcal{P} := (G, \mathbb{E}, \varepsilon)$ and a kernel operator κ on $\mathbb{E}$, the triple

$$opr(\mathcal{P}, \kappa) := (G, \mathbb{D}, \delta)$$

with $\mathbb{D} := \mathbb{E}|D$ where $D := \kappa E$ and

$$\delta : G \to D, g \mapsto \kappa(\varepsilon g)$$

is a pattern structure, called the **o-projection** of $\mathcal{P}$ via κ (see [2]).

3 Construction of a pattern structure

The following definition establishes the connection between pattern setups and pattern structures.

Definition 6 (embedded pattern structure)**.** Let $\mathbb{E}$ be a complete lattice and $\mathcal{P} := (G, \mathbb{D}, \delta)$ a pattern setup with $\mathbb{D} := \mathbb{E}|D$. Then we call

$$\mathcal{P}_e := (G, \mathbb{E}, \mathbb{D}, \overline{\delta}) \text{ with } \overline{\delta} : G \to L, g \mapsto \delta g$$

the **embedded pattern structure**. We say the pattern setup $\mathcal{P}$ is **embedded** in the pattern structure $(G, \mathbb{E}, \overline{\delta})$.

This definition shows, that it is possible to build a pattern structure from a given pattern stup. The construction below is another demonstration of how to build a pattern structure from a pattern setup.

Construction 1. Let G be a set and let $\mathbb{L} := (L, \leq)$ be a poset, then for every map $\varrho : G \to L$ the **elementary pattern structure** is given by:

$$\mathcal{P}_\varrho := (G, \mathbb{E}, \varepsilon) \text{ with } \mathbb{E} := (2^L, \supseteq) \text{ and } \varepsilon : G \to 2^L, g \mapsto \{\varrho g\}.$$

Hence, the pattern setup $(G, \mathbb{L}, \varrho)$ is embedded in the pattern structure $(G, \mathbb{E}, \varepsilon)$. In many cases this construction leads to a large set of patterns. Therefore we need the following: Let ζ be a closure system in $\mathbf{2}^L := (2^L, \subseteq)$, that is, ζ is a kernel system in $\mathbb{E}$ and let $\gamma : 2^L \to 2^L$ be the associated closure operator of ζ w.r.t. $\mathbf{2}^L$. Thus, γ is a kernel operator on $\mathbb{E}$. Then $(G, \mathbb{D}, \delta)$ is a pattern structure for $\mathbb{D} := \mathbb{E}|\zeta$ and

$$\delta : G \to D, g \mapsto \gamma\{g\}.$$

Indeed the map

$$\psi : 2^L \to \zeta, X \mapsto \gamma X$$

is a residual map from $\mathbb{E}$ to $\mathbb{D}$ with $\delta = \psi \circ \varepsilon$.
By the above proposition, $(G, \mathbb{D}, \delta)$ is a pattern structure, since

$$opr(\mathcal{P}_\varrho, \gamma) = (G, \mathbb{D}, \delta)$$

is the o-projection of $\mathcal{P}_\varrho$ via γ.

4 Connection to Data Mining and to our Dataset

In this section we describe how we use the construction 1 to handle a typical data mining classification problem. We train a model to predict classes of red wines. But first we give an insight to our data.

4.1 Red Wine Dataset

To apply our previous results on a public data set we choose the red wine data set from [3]. There are 1599 examples of wines, described by 11 numerical attributes. The input includes objective tests (e.g. fixed acidity, sulphates, PH values, residual sugar, chlorides, density, alcohol...) and the output is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between 0 (very bad) and 10 (very excellent). For our purpose, we established a binary distinction where every wine with a quality score above 5 is classified "good" and all below as "bad". This led to a set of 855 positive and 744 negative examples. We split the data into a training set (75% of the examples) and a test set (25% of the examples).

4.2 Describing the Proceeding

Many data mining relevant data sets (like the red wine dataset) can be described via an evaluation matrix:

Definition 7 (evaluation map, evaluation setup)**.** Let G be a finite set, M a set of attributes and $\mathbb{W}_m := (W_m, \leq_m)$ a complete lattice for every attribute $m \in M$. Further, let

$$W := \prod_{m \in M} W_m \text{ and } \mathbb{W} := \prod_{m \in M} \mathbb{W}_m.$$

Then, a map

$$\alpha : G \to W, g \mapsto \prod_{m \in M} \{\alpha_m g\}$$

such that

$$\alpha_m : G \to W_m, g \mapsto w_m$$

is called **evaluation map**. We call $\mathfrak{E} := (G, M, \mathbb{W}, \alpha)$ an **evaluation setup**.

Example 1. *In the wine data set in [3] we can interpret the wines as a set G, the describing attributes as the set M and $\mathbb{W}_m$ as the numerical range of attribute m with the natural order.*

In the above example the the evaluation map

$$\alpha : G \times M \to W$$

assigns to every wine bottle the values of all attributes $m \in M$. This map is a good starting point for an elementary pattern structure with

$$L := \prod_{m \in M} W_m \text{ and } \mathbb{L} := \prod_{m \in M} \mathbb{W}_m.$$

Thus, $\mathbb{E} := (2^L, \supseteq)$ is the dually ordered power set of vectors with values of the attributes, which describe the wine. On $\mathbb{E}$ we installed the following kernel operator

$$\gamma : 2^L \to 2^L, X \mapsto [inf_{\mathbb{L}} X, sup_{\mathbb{L}} X]_{\mathbb{L}},$$

This leads to the o-projection of the elementary pattern structure $\mathcal{P}_\varrho$ via γ, that is,

$$(G, \mathbb{D}, \delta) := opr(\mathcal{P}_\varrho, \gamma).$$

As a matter of fact,

$$\mathbb{D} = (D, \supseteq) \text{ with } D = \text{Int}\mathbb{L}$$

is the dual interval lattice of $\mathbb{L}$ and the map δ is given by

$$\delta : G \to D,\ g \mapsto \{\varrho g\}.$$

Often the dual power set lattice $\mathbb{E}$ is too large for applications. Therefore, we concentrate on relevant patterns in $\mathbb{D}$, that is, in the dual interval lattice of $\mathbb{E}$.

To identify important patterns in $\mathbb{E}$ for the red wine classification, we looked at the positive examples of the training set and combined the results of different clustering algorithms implemented in python. In particular, we used a *k-means* algorithm, *k-medoids* algorithm (with metrices Mahalanobis, Euclidean and correlation), a *Gausian Mixture Model* and a *Bayesian Gaussian Mixture Model* to cluster the good red wines. Furthermore we interprete the leaves of decision trees (with Gini Impurity and entropy as splitting measure) as cluster of wines to find important patterns in $\mathbb{E}$ for our case. The same cluster algorithm can lead to different output clusters; this is a result of the different metrics, which were used to measure the distance and the randomly choosen starting points of the algorithms. Hence we tried every algorithm 5 times. For every attempt we used different specifications. The number of clusters for the *k-medoids*, *k-means*, *Gausian Mixture Model* and *Gausian Mixture Model* algorithm is set randomly between 2 and 50. For the decision trees we set the number of examples in a leaf to at least 100. This leads to more than 700 clusters in $\mathbb{E}$.
Via the kernel operator on $\mathbb{E}$:

$$\gamma : 2^L \to 2^L, X \mapsto [inf_{\mathbb{L}} X, sup_{\mathbb{L}} X]_{\mathbb{L}}$$

we get patterns in $\mathbb{D}$ of the clusters in $\mathbb{E}$. Since γ is a closure operator on the power set $\mathbf{2}^L := (2^L, \subseteq)$ we can think of the patterns as closures of clusters.
On the next step we eliminated all clusters with less than 100 wines. Then we looked at the ratio of good examples (wines with a scoring of 5 or better) and all examples (good and bad) in the patterns and took the five patterns with the best ratio. These patterns are listed below. The range of the attributes is printed in red. For a better interpretability we scaled every attribute to the range $[0, 1]$.

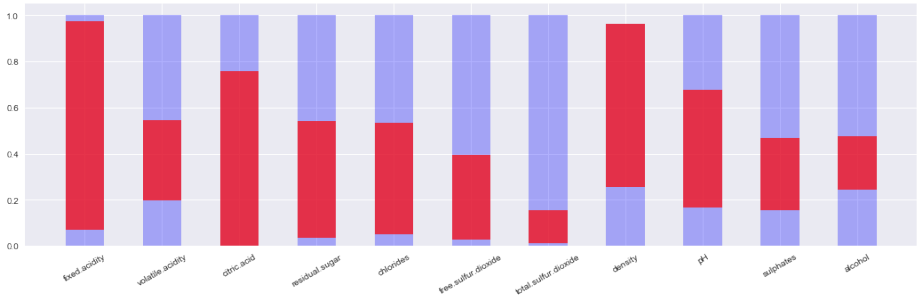

Fig. 1. Interval 1: decision tree (entropy), 108 wines, 108 good and 0 bad

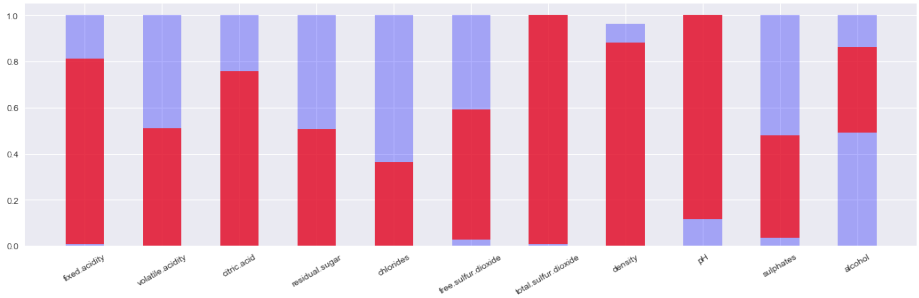

Fig. 2. Interval 2: decision tree (entropy), 158 wines, 158 good and 0 bad

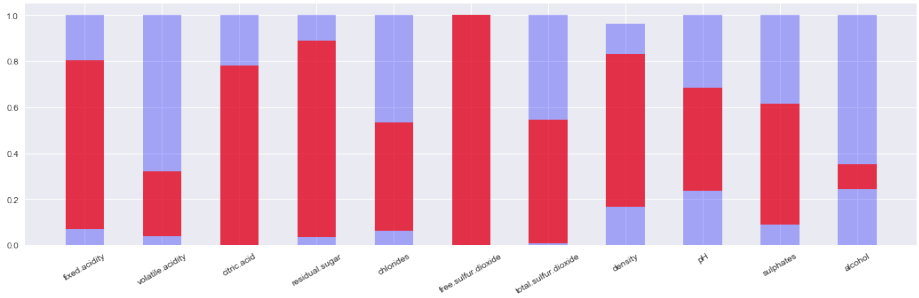

Fig. 3. Interval 3: decision tree (gini), 128 wines, 127 good and 1 bad

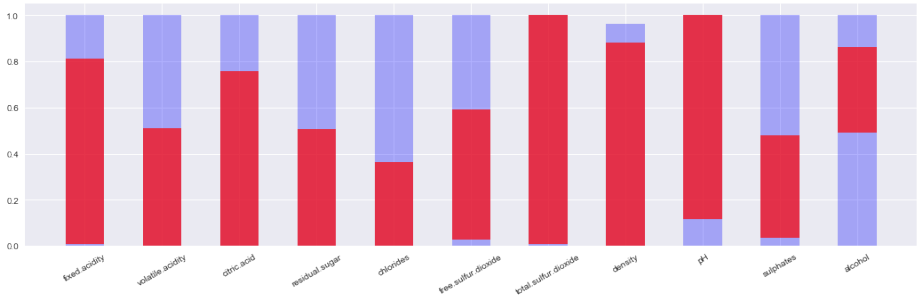

Fig. 4. Interval 4: k-medoids (mahalanobis), 163 wines, 160 good and 3 bad

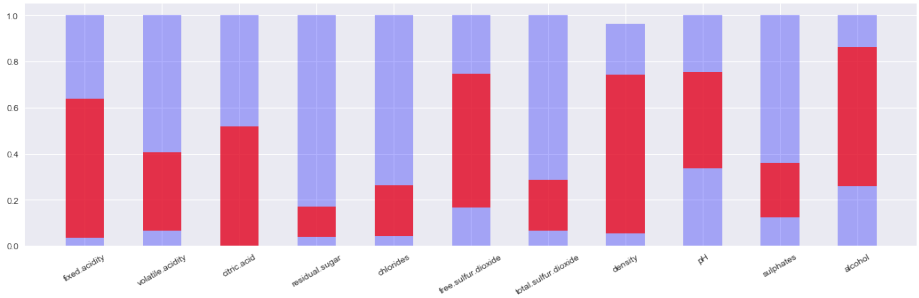

Fig. 5. Interval 5: k-medoids (mahalanobis), 216 wines, 209 good and 7 bad

Combining these 5 intervals to predict the classes of the test set leads to the following confusion matrix:

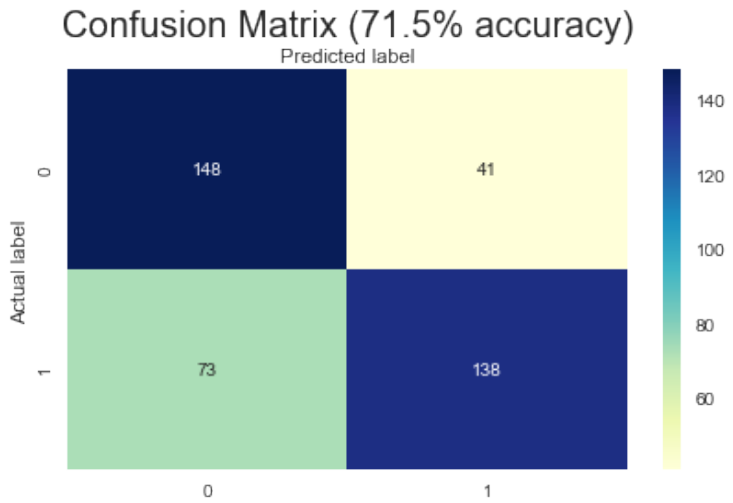

Fig. 6. Combination of the 5 intervals

The following table presents a comparison of our method to other algorithms.

Our method is easy to interprete and leads to the second best precision of all listed algorithms, but the recall value is the worst under all methods. More patterns would probably lead to a better recall, but likely worsen the precision. Further investigations are needed to find the best collections of patterns for different usecases (e.g. maximize accuracy). The here presented proceeding is just an example of building a model from our framework. Hopefully, further investigations show, that it is possible to create stronger models with our framework.

Model	Accuracy	Presicion	Recall	F1-score
cluster pattern	**71,5**%	**77,1**%	**65,5**%	**70,7**%
K-nearest neighbors	74,2%	73,0%	81,0%	76,9%
Support Vector	73,5%	75,6%	73,4%	74,5%
Naive Bayes	70,0%	69,2%	77,8%	73,2%
logistic regression	73,2%	73,6%	76,8%	75,2%
Random Forest	78,0%	80,0%	77,7%	78,8%

Table 1. Comparison of different classifier algorithms

5 Conclusion

We introduced a new general framework for the application of pattern structures. Then we gave an example how this general framework can be used to predict the quality of red wines. In the presented way the pattern structures can be a useful tool in analysing data. As shown here they are capable to give good predictions and the good interpretability makes them even more powerful.

References

1. T.S. Blyth, M.F.Janowitz (1972), Residuation Theory, Pergamon Press, pp. 1-382.
2. A. Buzmakov, S. O. Kuznetsov, A. Napoli (2015) , Revisiting Pattern Structure Projections. Formal Concept Analysis. Lecture Notes in Artificial Intelligence (Springer), Vol. 9113, pp 200-215.
3. P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis (2009), Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems 47(4), pp. 547-553.
4. B. Ganter, S. O. Kuznetsov (2001), 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 (Springer), Vol. 2120, pp. 129-142.
5. T. B. Kaiser, S. E. Schmidt (2011), Some remarks on the relation between annotated ordered sets and pattern structures. Pattern Recognition and Machine Intelligence. Lecture Notes in Computer Science (Springer), Vol. 6744, pp 43-48.
6. M. Kaytoue, S. O. Kuznetsov, A. Napoli, S. Duplessis (2011), Mining gene expression data with pattern structures in formal concept analysis. Information Sciences (Elsevier), Vol.181, pp. 1989-2001.
7. S. O. Kuznetsov (2009), Pattern structures for analyzing complex data. In H. Sakai et al. (Eds.). Proceedings of the 12th international conference on rough sets, fuzzy sets, data mining and granular computing (RSFDGrC'09). Lecture Notes in Artificial Intelligence (Springer), Vol. 5908, pp. 33-44.
8. S. O. Kuznetsov (2013), Scalable Knowledge Discovery in Complex Data with Pattern Structures. In: P. Maji, A. Ghosh, M.N. Murty, K. Ghosh, S.K. Pal, (Eds.). Proc. 5th International Conference Pattern Recognition and Machine Intelligence (PReMI'2013). Lecture Notes in Computer Science (Springer), Vol. 8251, pp. 30-41.
9. L. Lumpe and S. E. Schmidt (2015), A Note on Pattern Structures and Their Projections. Formal Concept Analysis. Lecture Notes in Artificial Intelligence (Springer), Vol. 9113, pp 145-150.

10. L. Lumpe, S. E. Schmidt (2016), Morphisms Between Pattern Structures and Their Impact on Concept Lattices, FCA4AI@ ECAI 2016, pp 25-34.
11. L. Lumpe, S. E. Schmidt (2015), Pattern Structures and Their Morphisms. CLA 2015, pp 171-179.
12. L. Lumpe, S. E. Schmidt (2016), Viewing Morphisms Between Pattern Structures via Their Concept Lattices and via Their Representations, International Symposium on Methodologies for Intelligent Systems 2017, pp. 597-608.
13. H. S. Park, C. H. Jun (2009), A simple and fast algorithm for K-medoids clustering. Expert systems with applications 36(2), pp. 3336-3341.