Surveying to Bring Order in the Jungle of Supervised Local Pattern Mining Implementations

TU/e supervisor: dr. W. Duivesteijn

Introduction to Local Pattern Mining

In contrast with building a one-size-fits-none global model for the entire dataset, Local Pattern Mining is a data mining subfield that acknowledges that a dataset may encompass multiple groups that each behave in their own way. The goal of Local Pattern Mining is to discover interesting subsets in a dataset. The subsets must satisfy two conditions:

- they must be *interpretable*;
- they must display *exceptional behavior*.

By *interpretable*, we typically mean that we are only interested in subsets that can be defined as a conjunction of a few conditions on columns of the dataset at hand. Hence, we do not want to find "this group of ten datapoints is interesting", but instead:

$$\text{smoking} = \text{yes} \wedge \text{age} \leq 25 \rightsquigarrow \text{(some exceptional behavior)}$$

Such subsets are called *subgroups*. Returning only subgroups that can be defined in terms that the dataset owner can understand, ensures that we find things that are actionable: we can build a policy on this sort of knowledge.

Exceptional behavior can be defined in many many ways. A wild variety of Local Pattern Mining methods exist, typically distinguished by what choice they make here. For instance:

Frequent Itemset Mining **[1]** : defines it in an unsupervised manner as "high frequency";

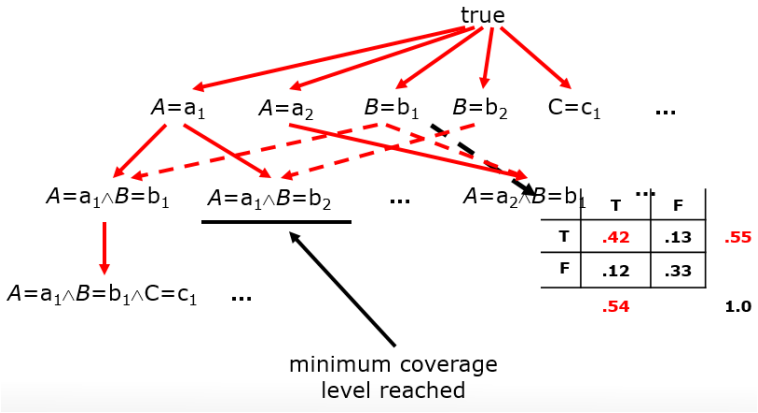
Subgroup Discovery **[4, 7, 3]** : defines it in a supervised manner as "unusual distribution of a single target columm";

Exceptional Model Mining **[5, 2]** : defines it in a supervised manner as "unusual interaction between several target columns".

Still, within those definitions, many parameters can be chosen and altered to end up with a sprawling cornucopia of interesting mining tasks and results.

Search Lattice

The subgroup search space can be represented as a lattice of candidate subgroups, where each level of the search consists of specializations of subgroups from the layer above:



When the dataset consists of many attributes with many distinct values, the number of subgroups that can be defined explodes, and the lattice above becomes ridiculously (intractably) wide. This often prohibits exhaustive search.

The Project

Many algorithms, some more intelligent, some more generic, some more fast, some more all-encompassing, have been introduced for the supervised versions of Local Pattern Mining. This project aims to create some order in this implementation jungle, by creating a categorical survey of these implementations. We need to discuss:

- which lattice search strategies are supported? Top-down versus bottom-up search, or pattern sampling? Exhaustive or heuristic or anytime algorithm with or without guarantees? Is manual stopping possible?
- which types of target models are included in the implementation?
- which types of refinements are possible in the subgroup definition language? Do we in- or exclude $\neq$ constraints for categorical variables, and set-valued constraints? For numerical variables, do we allow value-equivalence constraints, half intervals, full intervals?
- which discretization strategies [6] are included in the implementation?
- does the algorithm leverage the embarrassingly parallel nature of level-wise search?

This is just off the top of my head. It is quite likely that many more interesting distinctions between implementations can be made.

Current Status

It is *very likely* that the following summary misses many relevant implementations. But to give you a head start: I know that some German researchers maintain a Python package called pysubgroup, and they published papers about this (making this by far the best-documented implementation). An overlapping set of German researchers maintained VIKAMINE, another group of German researchers made an implementation for RapidMiner, Slovenian researchers worked with Orange (relevance not guaranteed), French researchers developed their own implementations, Spanish researchers created evolutionary algorithms, and a group from Leiden originally worked with a Java implementation. All kinds of overlapping implementations were generated here at TU/e too. Most of these implementations are at best not that well documented beyond the research papers in which the implementations were used. Any order we can bring into this chaos is progress, for no order currently exists.

Requirements

The main task of a student in this project is to dig through other people's software implementations and associated papers. So, some affinity with reading source code in multiple languages is desired. A good survey should be publishable in a journal.

References

[1] Rakesh Agrawal, Ramakrishnan Srikant: Fast Algorithms for Mining Association Rules in Large Databases. VLDB 1994: 487-499

[2] Wouter Duivesteijn, Ad Feelders, Arno J. Knobbe: Exceptional Model Mining - Supervised descriptive local pattern mining with complex target concepts. Data Min. Knowl. Discov. 30(1): 47-98 (2016)

[3] Francisco Herrera, Cristóbal J. Carmona, Pedro González, María José del Jesus: An overview on subgroup discovery: foundations and applications. Knowl. Inf. Syst. 29(3): 495-525 (2011)

[4] Willi Klösgen: Explora: A Multipattern and Multistrategy Discovery Assistant. Advances in Knowledge Discovery and Data Mining 1996: 249-271

[5] Dennis Leman, Ad Feelders, Arno J. Knobbe: Exceptional Model Mining. ECML/PKDD (2) 2008: 1-16

[6] Marvin Meeng, Arno J. Knobbe: For real: a thorough look at numeric attributes in subgroup discovery. Data Min. Knowl. Discov. 35(1): 158-212 (2021)

[7] Stefan Wrobel: An Algorithm for Multi-relational Discovery of Subgroups. PKDD 1997: 78-87