Preventing Beam Pollution: Defining an Empirical Protocol to Improve Beam Search Lattice Traversal

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";

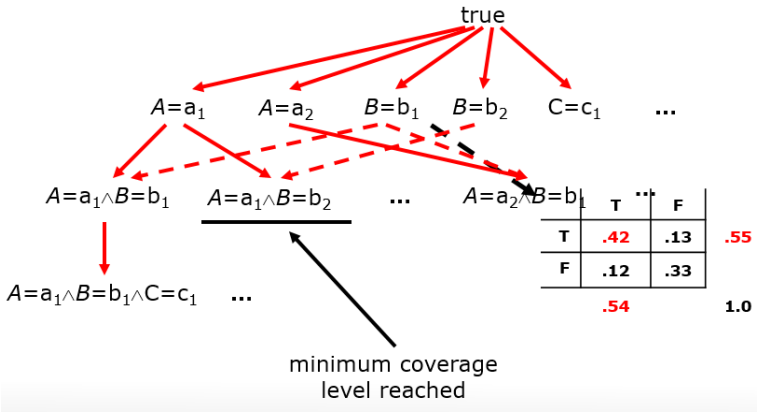
Subgroup Discovery **[4, 9, 3]** : defines it in a supervised manner as "unusual distribution of a single target columm";

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.

A canonical way to traverse this search lattice is through *Beam Search* [2, Algorithm 1]: a top-down algorithm where we start with the most general subgroup there is: no conditions, so the subgroup encompasses all records in the dataset. We then generate all possible refinements (conjoining to the current condition list a single condition on a single attribute) of this seed subgroup as candidates for the next level: their qualities are determined, and the best w (to be set by the user) are kept as the *beam* for the next level. On the next level each subgroup in the beam is refined into a new set of candidates, and this process repeats until the maximum depth d (to be set by the user) is reached. Hence, Beam Search starts from the top, and makes a level-wise downward traversal through the lattice until a certain maximum depth is reached; each level beyond depth 1 is only partially traversed.

The Beam Pollution Problem

The core idea behind Beam Search is attractive: we essentially have a multi-pronged greedy strategy. Since we cannot expand all the search nodes in the top level we must resort to heuristics, but if we were to go pure greedy (expanding only the single most promising search node) we are very likely to miss interesting subgroups. Beam Search is meant to provide the best of both worlds: by expanding the w (typically: 10 to 100) most promising search nodes, we are more likely to cover all interesting subgroups while keeping the tractability under control.

A problem here is that this parameter w is fiendishly difficult to tune, and its correct setting is very sensitive to the underlying structure of the dataset as a whole. Let's illustrate the problem with a concrete dataset. Let's say that the dataset is about people, and that for the target concept we're interested in both smokers and people of retirement age are exceptionally-behaving subgroups; these factors amplify each other for even better-scoring subgroups. On search depth $d = 1$, our beam may look something like this:

Rank	Subgroup
1.	age ≥ 67
2.	occupation = retired
3.	age ≥ 72
4.	age ≥ 62
5.	smoker = yes
6.	age ≥ 77
7.	age ≥ 57
8.	owns_house = yes
9.	has_children = yes
10.	yearly_income = high
$\vdots$	$\vdots$

This already illustrates two problems. The top-two subgroups are strongly correlated: retirement age in the Netherlands is around 67 years, so these subgroups are almost identical. This is inevitable in large datasets. The numeric attributes pose another problem: how does one determine the right level of discretization? Too finegrained means that our beam instantly pollutes, too coarse means that potentially interesting subgroups go undetected. Notice how ranks 8-10 contain some wider groups that point at middle-aged and older people; these signals are not as strong as the ones from the age, occupation, and smoker attributes, but they do belong a little lower-ranked in the beam of a certain width.

On search depth $d = 2$, we find the following:

Rank	Subgroup
1.	age ≥ 67 AND smoker = yes
2.	occupation = retired AND smoker = yes
3.	age ≥ 67 AND occupation = retired
4.	age ≥ 72 AND smoker = yes
5.	age ≥ 72 AND occupation = retired
6.	age ≥ 62 AND smoker = yes
7.	age ≥ 62 AND occupation = retired
8.	age ≥ 57 AND occupation = retired
9.	age ≥ 52 AND occupation = retired
10.	age ≥ 22 AND occupation = retired
$\vdots$	$\vdots$

The top-two subgroups are still almost identical, but this combination of factors is the strongest signal in the dataset so it's not necessarily bad that we find this. But diversity of the subgroups in the beam has decreased quite a bit. Notice how the last four subgroups are basically all the same: if enough half-intervals of age end up in the beam on search depth 1, these all become flattened into the same subgroup by recombination with the occupation = retired condition on level 2, and if this subgroup is good enough to enter the beam, it will do so with all its variants.

On search depth $d = 3$, things break down irreparably:

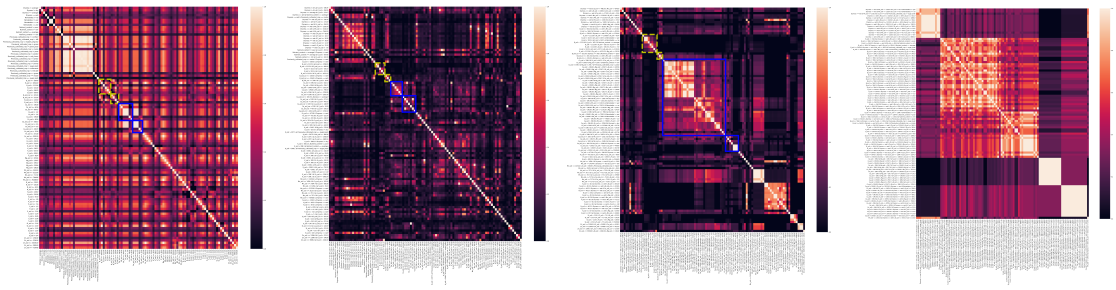
Rank	Subgroup
1.	age ≥ 67 AND smoker = yes AND income $\leq \$600K$
2.	occupation = retired AND smoker = yes AND income $\leq \$600K$
3.	age ≥ 67 AND smoker = yes AND occupation $\neq$ carpenter
4.	age ≥ 67 AND smoker = yes AND occupation $\neq$ train driver
5.	age ≥ 67 AND smoker = yes AND occupation $\neq$ tech entrepreneur
6.	age ≥ 67 AND smoker = yes AND occupation $\neq$ quarterback
7.	age ≥ 67 AND smoker = yes AND income $\leq \$500K$
8.	occupation = retired AND smoker = yes AND income $\leq \$500K$
9.	age ≥ 67 AND smoker = yes AND income $\leq \$450K$
10.	occupation = retired AND smoker = yes AND income $\leq \$450K$
⋮	⋮

Only variants of the first two subgroups on the previous level remain. Two problems appear. The variable occupation is a high-cardinality categorical: many occupations exist, and if the pattern language allows for $\neq$ constraints, then many near-copies of the same subgroup can be generated that will inevitably pollute the beam. But even if we forbid such constraints, this does not prevent the other problem: at some point we will have identified the strongest subgroups in the dataset, and on subsequent levels, the algorithm will try to find ways to generate small tweaks to those subgroups in order to maximize quality. At some point, the beam will inevitably become polluted.

What to Do?

It is hard to make a general statement about when a refinement of a subgroup is useful and when it is spurious. This makes it almost impossible to make an individual decision on the subgroup level. But we can try to tackle this problem, perhaps, by looking at the composition of the beam on a collective level: are these indeed sufficiently different prongs of the greedy strategy we are trying to pursue here?

We made a first attempt in an as of yet unpublished paper. Of the 100 subgroups that ended up in the beam, we determined the mutual Jaccard indices: of each pair, determine the overlap of their coverage (records belonging to the subgroup). Lower is better: a value of zero means that these subgroups are disjoint; a value of one means that these subgroups are identical. In the figure below, you find the heatmaps for the first four search levels:



On depth 1, there is some overlap, but this is not weird: with 100 subgroups in the beam, and a dataset with not that crazy many attributes, we will find some overlap there. On depth 2, almost the entire off-diagonal figure turns very dark. This is ideal: many very disjoint subgroups in the beam. On depth 3, some clusters begin to appear but this is still largely okay. On depth 4, we see all these blocks of white: our beam gets polluted by clumps of identical subgroups. We probably have gone too far.

I firmly believe that this tells us *something*. I am not quite as convinced precisely *what* it tells us. On depth 4, have we gone too far? How can we turn such pictures into a hard decision? And how does this decision depend on problem parameters such as:

- number of records;
- chosen search width;
- maximum search depth;
- total number of attributes;
- fraction of attributes that is numeric;
- chosen discretization strategy for numeric attributes [8];
- cardinality of the categorical attributes;
- correlation between the attributes;
- kind of target interaction that we aim for and the quality measure we employ to determine exceptionality.

The ultimate goal of this project would be a hard protocol: can we make a decision when the beam search is polluted in such a way that digging deeper is senseless (make a stopping criterion on the search depth)? Failing that, can we come up with a better beam selection strategy?

Current Status

Many papers exist to perform Diverse Subgroup Set Selection: finding a good but diverse set of subgroups as final result of a local pattern mining run. For a subgroup discovery setting, two papers exist [6, 7] that connect the beam composition to optimality in ROC space, but ROC analysis only makes sense if a single binary target attribute exists and we want something more generic.

Requirements

You need an analytic mind and a willingness to plug away at solutions. This is a seemingly simple problem with no solution, but that means that many intelligent people have tried many things already and failed. If we can come up with a well-motivated protocol, I expect this project to lead to a publication at a top-level data mining or machine learning conference, or perhaps 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] Michael Mampaey, Siegfried Nijssen, Ad Feelders, Arno J. Knobbe: Efficient Algorithms for Finding Richer Subgroup Descriptions in Numeric and Nominal Data. ICDM 2012: 499-508

[7] Marvin Meeng, Wouter Duivesteijn, Arno J. Knobbe: ROCsearch - An ROC-guided Search Strategy for Subgroup Discovery. SDM 2014: 704-712

[8] 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)

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