Documentation: MIP Function — Subgroup discovery from multi-resolution data

A subgroup discovery tool that can use ontological domain knowledge (RDF graphs) in the learning process. Subgroup descriptions contain terms from the given domain knowledge and enable potentially better generalizations.

Metadata

Category	MIP function, data analysis algorithm
Maintainers	Anže Vavpetič
Homepage	https://github.com/anzev/hedwig/tree/hbp
Documentation	http://source.ijs.si/hbp/mipfunctions/raw/master/doc/r-hedwig.pdf https://github.com/anzev/hedwig/tree/hbp
Support	https://github.com/anzev/hedwig/issues
Source Code	https://github.com/anzev/hedwig/tree/hbp
License	MIT
Current Version	0.3.1
All Versions	0.3.1

Description

This MIP function implements the Hedwig algorithm [1,2, 3]. Given a data set consisting of examples (e.g., patients) described in terms of several descriptive binary attributes (e.g., clinical variables) labelled with a single target attribute (e.g., a potential biological marker) the algorithm produces a set of rules, i.e., subgroup descriptions or subgroups. In contrast to standard subgroup discovery, Hedwig can also exploit additional domain knowledge encoded as RDF graphs: these can be simple facts or ontologies like the Gene Ontology or a combination of several types of domain knowledge. These RDF graphs represent additional relationships of the attributes describing the examples and can be used to automatically generate generalizations not possible only with the “bottom level” descriptive attributes. Currently it is unclear if such hierarchical information will be available within the MIP, so the Hedwig algorithm (as implemented in the MIP function) at the moment generates subgroup descriptions only from bottom level attributes.

References

1. VAVPETIČ, Anže, KRALJ NOVAK, Petra, GRČAR, Miha, MOZETIČ, Igor, LAVRAČ, Nada. Semantic data mining of financial news articles. Editors: FÜRNKRANZ, Johannes (ur.), et al. Discovery science : 16th International Conference, DS 2013, Singapore, October 6-9, 2013. proceedings, (Lecture notes in computer science, ISSN 0302-9743, Lecture notes in artificial intelligence, 8140). Berlin, Heidelberg: Springer, 2013, vol. 8140, pages 294-307.
2. ADHIKARI, Prem Raj, VAVPETIČ, Anže, KRALJ, Jan, LAVRAČ, Nada, HOLLMÉN, Jaakko. Explaining mixture models through semantic pattern mining and banded matrix visualization. Editors: DŽEROSKI, Sašo (ur.), et al. Discovery science : 17th International Conference, DS 2014, Bled, Slovenia, October 8-10 : proceedings, (Lecture notes in computer science, ISSN 0302-9743, Lecture notes in artificial intelligence, 8777). Heidelberg [etc.]: Springer, 2014, vol. 8777, pages 1-12.
3. ADHIKARI, Prem Raj, VAVPETIČ, Anže, KRALJ, Jan, LAVRAČ, Nada, HOLLMÉN, Jaakko. Explaining mixture models through semantic pattern mining and banded matrix visualization. *Machine Learning Journal*, 2016, *in press*.

Usage

Input: A set of labelled examples and (potentially) background hierarchical information about the descriptive attributes, and binary relations among examples.

Parameters: The query that selects the examples to analyse with the provided attributes (PARAM_query). Target attribute name (PARAM_target_att).

Output: A set of descriptive rules (subgroup descriptions) encoded in JSON. The rules are typically not used for prediction, but for inspection by domain experts.

Example

An example test case can be defined for the test database from the HBP functions-repository as follows. We provide a test script to work with this example here.

Input data:

```
PARAM_query="select * from func"

PARAM_target_att="tsnr"
```

Here we select all of the attributes from the *func* table and (arbitrarily) select *tsnr* as the target variable. Other parameters are currently set to viable defaults and will be exposed if needed.

Example output:

The output of the MIP function is encoded in a JSON string. On this small example, the algorithm produces a single statistically significant rule. In human-readable form it is as follows (along with some statistics):

```
'22.929=<tsnr<25.72895'(X) <-- annotated_with(X, 3.10265=<fwhm<3.1378), annotated_with(X, 0.0253=<gsr<0.0018)

[cov=4, pos=4, prec=1.000, lift=3.444, pval=0.002]
```

The algorithm works only with binary attributes and nominal targets, so the attributes were all discretized beforehand by the algorithm. As mentioned, in this case we have no hierarchical information available, so the *annotated_with* predicates contain only the base attributes such as *fwhm* and *qsr*.