Package 'corels'

June 5, 2020

Type Package

Title R Binding for the 'Certifiably Optimal RulE ListS (Corels)' Learner

Version 0.0.2

Date 2020-05-31

Author Dirk Eddelbuettel

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description The 'Certifiably Optimal RulE ListS (Corels)' learner by Angelino et al described in <arXiv:1704.01701> provides interpretable decision rules with an optimality guarantee, and is made available to R with this package. See the file 'AUTHORS' for a list of copyright holders and contributors.

License GPL (>= 2)

Imports Rcpp (>= 1.0.2)

LinkingTo Rcpp

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-06-05 11:00:02 UTC

R topics documented:

corels-package *R Binding for the 'Certifiably Optimal RulE ListS (Corels)' Learner*

Description

The 'Certifiably Optimal RulE ListS (Corels)' learner by Angelino et al described in <arXiv:1704.01701> provides interpretable decision rules with an optimality guarantee, and is made available to R with this package. See the file 'AUTHORS' for a list of copyright holders and contributors.

Package Content

Index of help topics:

```
corels                  Corels interace
corels-package          R Binding for the 'Certifiably Optimal RulE
                        ListS (Corels)' Learner
```

Maintainer

Dirk Eddelbuettel <edd@debian.org>

Author(s)

Dirk Eddelbuettel

corels *Corels interace*

Description

R Interface to 'Certifiably Optimal RulE ListS (Corels)'

Usage

```
corels(rules_file, labels_file, log_dir, meta_file = "", run_bfs = FALSE,
  calculate_size = FALSE, run_curiosity = FALSE, curiosity_policy = 0L,
  latex_out = FALSE, map_type = 0L, verbosity_policy = 0L,
  max_num_nodes = 100000L, regularization = 0.01,
  logging_frequency = 1000L, ablation = 0L)
```

Arguments

`rules_file`	Character variable with file name for training data; see corels documentation and data section below.
`labels_file`	Character variable with file name for training data labels; see corels documentation and data section below.
`log_dir`	Character variable with logfile directory name
`meta_file`	Optional character variable with file name for minor data with bit vector to support equivalent points bound (see Theorem 20 in Section 3.14).
`run_bfs`	Boolean toggle for 'breadth-first search'. Exactly one of 'breadth-first search' or 'curiosity_policy' *must* be specified.
`calculate_size`	Optional boolean toggle to calculate upper bound on remaining search space size which adds a small overheard; default is to not do this.
`run_curiosity`	Boolean toggle
`curiosity_policy`	Integer value (between 1 and 4) for best-fist search policy. Exactly one of 'breadth-first search' or 'curiosity_policy' *must* be specified. The four different prirization schemes are chosen, respectively, by values of one for prioritize by curiousity (see Section 5.1 of the paper), two for prioritize by the lower bound, three for prioritize by the objective or four for depth-first search.
`latex_out`	Optional boolean toggle to select LaTeX output of the output rule list.
`map_type`	Optional integer value for the symmetry-aware map. Use zero for no symmetry-aware map (this is also the default), one for permutation map, and two for the captured vector map.
`verbosity_policy`	Optional character variable one containing one or more of the terms 'rule', 'label', 'minor', 'samples', 'progress', 'loud', or 'silent'.
`max_num_nodes`	Integer value for the maximum trie cache size; execution stops when the number of node isn trie exceeds this number; default is 100000.
`regularization`	Optional double value, default is 0.01 which can be thought of as a penalty equivalent to misclassifying 1% of the data when increasing the length of a rule list by one association rule.
`logging_frequency`	Optional integer value with default of 1000.
`ablation`	Integer value, default value is zero, one excludes the minimum support bounds (see Section 3.7), two excludes the lookahead bound (see Lemma 2 in Section 3.4).

Details

CORELS is a custom discrete optimization technique for building rule lists over a categorical feature space. The algorithm provides the optimal solution with a certificate of optimality. By leveraging algorithmic bounds, efficient data structures, and computational reuse, it achieves several orders of magnitude speedup in time and a massive reduction of memory consumption. This approach produces optimal rule lists on practical problems in seconds, and offers a novel alternative to CART and other decision tree methods.

Value

A constant bool for now

References

Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. *Learning Certifiably Optimal Rule Lists for Categorical Data.* JMLR 2018, http://www.jmlr.org/papers/volume18/17-716/17-716.pdf Nicholas Larus-Stone, Elaine Angelino, Daniel Alabi, Margo Seltzer, Vassilios Kaxiras, Aditya Saligrama, Cynthia Rudin. *Systems Optimizations for Learning Certifiably Optimal Rule Lists*. SysML 2018 http://www.sysml.cc/doc/2018/54.pdf Nicholas Larus-Stone. *Learning Certifiably Optimal Rule Lists: A Case For Discrete Optimization in the 21st Century. Senior thesis 2017. https://dash.harvard.edu/handle/1/38811502. Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, Cynthia Rudin. *Learning certifiably optimal rule lists for categorical data*. KDD 2017, https://www.kdd.org/kdd2017/papers/view/learning-certifiably-optimal-rule-lists-for-categorical-data.

See Also

The corels C++ implementation at https://github.com/nlarusstone/corels, the website at https://github.com/nlarusstone/corels and the Python implementation at https://github.com/fingoldin/pycorels.

Examples

```
library(corels)

logdir <- tempdir()
rules_file <- system.file("sample_data", "compas_train.out", package="corels")
labels_file <- system.file("sample_data", "compas_train.label", package="corels")
meta_file <- system.file("sample_data", "compas_train.minor", package="corels")

stopifnot(file.exists(rules_file),
          file.exists(labels_file),
          file.exists(meta_file),
          dir.exists(logdir))

corels(rules_file, labels_file, logdir, meta_file,
       verbosity_policy = "silent",
       regularization = 0.015,
       curiosity_policy = 2,   # by lower bound
       map_type = 1)     # permutation map
cat("See ", logdir, " for result file.")
```

Index