SoftwareX 24 (2023) 101526

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

CLUSPLUS: A decision tree-based framework for predicting structured outputs

Matej Petković [a,b,1], Jurica Levatić [a,1], Dragi Kocev [a], Martin Breskvar [a,2], Sašo Džeroski [a,*,2]

[a] Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
[b] Faculty of Mathematics and Physics, Jadranska 21, 1000 Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 15 June 2023
Received in revised form 7 September 2023
Accepted 11 September 2023

Dataset link: https://github.com/knowledge-technologies/clus

Keywords:
Machine learning
Multi-target regression
Multi-label classification
Feature importance
Semi-supervised learning
Decision trees
Random forests

ABSTRACT

We present CLUSPLUS, a machine learning framework based on decision trees specialized for complex predictive modeling tasks. We provide the scientific community with an open source Java framework that unifies several major research directions in the machine learning field. The framework supports multi-target prediction, i.e., the simultaneous prediction of multiple continuous values, multiple discrete values, and hierarchically organized discrete values. Furthermore, CLUSPLUS enables state-of-the-art predictive performance via ensemble learning, exploitation of unlabeled data via semi-supervised learning, and data understanding via feature importance and building interpretable models. Out of a wide array of machine learning frameworks available today, very few support complex predictive modeling tasks and, to the best of our knowledge, none support all of the aforementioned functionalities.

Code metadata

Current code version	2.12.8
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00369
Permanent link to Reproducible Capsule	https://doi.org/10.24433/CO.0567198.v2
Legal Code License	GNU General Public License version 3
Code versioning system used	git
Software code languages, tools, and services used	Java
Compilation requirements, operating environments & dependencies	Java 1.8 JDK, Apache Maven 3.6
Link to developer documentation/manual	https://github.com/knowledge-technologies/clus/blob/main/ClusProject/docs/manual/clus-manual.pdf
Support email for questions	clus@ijs.si

1. Motivation and significance

The most common tasks in machine learning are classification and regression, where the goal is to predict a single discrete or numeric value (i.e., the target or output) associated with each example, from the descriptive features of the example. In many applications of machine learning, however, the target values to be predicted are inherently more complex [1,2]: multiple targets can be associated to each example and dependencies between targets can exist. The main types of such predictive tasks are: multi-target regression (where there are two or more numeric target variables), multi-label classification (where there are two or more binary target variables), and hierarchical multi-label classification (where classes are organized in a tree-shaped hierarchy or a directed acyclic graph).

Orthogonally to predicting structured target values, many machine learning applications require the learning of understandable models (i.e., knowledge extraction from the data), on one hand, and accurate models, on the other hand. The requirement for extracting knowledge from the data can be supported by

* Corresponding author.
E-mail addresses: martin.breskvar@ijs.si (Martin Breskvar), saso.dzeroski@ijs.si (Sašo Džeroski).
1 These authors contributed equally to this work.
2 These authors contributed equally to this work.

https://doi.org/10.1016/j.softx.2023.101526

learning interpretable models, such as decision trees. In the quest for more accurate models, ensembles (and in particular tree ensembles) have proved to be a powerful tool, but while being more accurate, they are more difficult to understand. Fortunately, feature importance estimation (and ranking) can provide insight into the data and the ensemble models built from the data.

To improve upon low predictive performance due to the scarcity of labeled training examples, semi-supervised learning [3] uses unlabeled examples, in addition to the labeled ones, in the learning process. It is useful in applications where few labeled examples are available due to expensive and/or time-consuming labeling procedures, while large numbers of unlabeled examples are easily available. Example applications include quantitative structure–activity relationship modeling, phonetic annotation of human speech, protein 3D structure prediction, and spam filtering.

Most publicly available machine learning toolboxes, such as Weka [4], Knime [5], or Orange [6] are designed to address predictive modeling tasks for primitive (unstructured) targets, i.e., classification and regression. A few that can handle the prediction of structured targets, such as Meka [7], Mulan [8], and scikit-learn [9], can handle only specific cases of structured targets. Typically, they cannot handle unlabeled data, i.e., missing target values, or even missing values of the independent variables.

Here we present CLUSPLUS, an open source machine learning framework for predicting structured outputs based on the predictive clustering paradigm. Building upon CLUS [10], it implements predictive clustering trees (PCTs) as the basic building blocks that embody the predictive clustering paradigm [11]. PCTs are generalized decision trees, capable of predicting simple unstructured targets (solving traditional classification and regression tasks) as well as several types of structured targets (solving the tasks of multi-target regression (MTR), multi-label classification (MLC), and hierarchical multi-label classification (HMLC)). CLUSPLUS also supports ensemble learning, feature ranking, and semi-supervised learning. To the best of our knowledge, no other machine learning toolbox offers such capabilities within a common framework. Thus, we expect CLUSPLUS to open possibilities for exploring new research avenues and application domains, as well as to provide the machine learning community with a baseline for benchmark comparisons for newly developed methods for the ever more popular tasks of predicting structured outputs.

2. Software description

CLUSPLUS is based on Predictive Clustering Trees (PCTs), which treat a decision tree as a hierarchy of clusters. The top-node contains all the data and is recursively partitioned into smaller clusters while descending the tree. PCTs are generated using a standard top-down induction of decision trees (TDIDT) algorithm described by Breiman [12].

The algorithm, outlined in Table 1, takes a set of examples (E) as input and produces a tree as output. It employs a heuristic (h) to select the tests (t), which is based on the reduction in variance resulting from the partitioning ($\mathcal{P}$) of instances associated with the tests (t) (refer to line 4 in the BestTest procedure in Table 1). This approach maximizes cluster homogeneity and enhances predictive performance by maximizing the variance reduction. In PCTs, the variance function (which evaluates the splits) and the prototype function (which calculates a label for each leaf node) are adjustable parameters that are instantiated to suit a specific learning task, such as multi-target prediction or semi-supervised learning (for more details see Refs. [13–15]).

Ensembles of PCTs [13] are constructed analogously to the bagging [16] and random forests [17] methods proposed by Breiman, as outlined in Table 2. In these methods, multiple

Table 1
The top-down induction algorithm for PCTs.

procedure PCT	procedure BestTest
Input: A dataset E	**Input:** A dataset E
Output: A predictive clustering tree	**Output:** the best test (t^*), its heuristic score (h^*) and the partition ($\mathcal{P}^*$) it induces on the dataset (E)
1: $(t^*, h^*, \mathcal{P}^*) = \text{BestTest}(E)$	1: $(t^*, h^*, \mathcal{P}^*) = (none, 0, \emptyset)$
2: **if** $t^* \neq none$ **then**	2: **for each** possible test t **do**
3: **for each** $E_i \in \mathcal{P}^*$ **do**	3: $\mathcal{P}$ = partition induced by t on E
4: $tree_i = \text{PCT}(E_i)$	4: $h = Var(E) - \sum_{E_i \in \mathcal{P}} \frac{\lvert E_i \rvert}{\lvert E \rvert} Var(E_i)$
5: **return** $\text{node}(t^*, \bigcup_i \{tree_i\})$	5: **if** $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ **then**
6: **else**	6: $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
7: **return** leaf(Prototype(E))	7: **return** $(t^*, h^*, \mathcal{P}^*)$

Table 2
The ensemble learning algorithms: bagging and random forests. Here, E is the set of the training examples, k is the number of trees in the forest, D is the number of features, and $f(D)$ is the size of the feature subset considered at each node during tree construction for random forests.

procedure Bagging(E, k)	**procedure** RForest(E, k, $f(D)$)
returns Forest	**returns** Forest
1: $F = \emptyset$	1: $F = \emptyset$
2: **for** $i = 1$ **to** k **do**	2: **for** $i = 1$ **to** k **do**
3: $E_i = bootstrap(E)$	3: $E_i = bootstrap(E)$
4: $T_i = PCT(E_i)$	4: $T_i = PCT\_rnd(E_i, f(D))$
5: $F = F \bigcup \{T_i\}$	5: $F = F \bigcup \{T_i\}$
6: **return** F	6: **return** F

bootstrap training datasets are obtained by randomly sampling training instances, with replacement, from the original training set (line 3 of Table 2). Additionally, in random forest, the PCT algorithm for tree construction (Table 1) is changed to *PCT_rnd* where at each node in the decision trees, a different random subset of the descriptive attributes (of size $f(D)$, e.g., $\sqrt{D}$) is considered (line 4 of Table 2).

CLUSPLUS is implemented in the Java programming language. Therefore, it is able to run wherever a Java runtime environment can be installed. The software is designed to be used as a stand-alone command-line tool. Its latest version (2.12.8) is available at: https://github.com/knowledge-technologies/clus.

2.1. Building the code

The following prerequisites must be met in order to successfully build the CLUSPLUS codebase. First, Java 1.8 JDK must be installed on the system that will be used to build CLUSPLUS (JRE is enough for running the built jar file). Second, Apache Maven 3.6 or newer is needed to actually perform the build. Finally, a git client is needed to clone the project repository. Once the prerequisites are met, the build process consists of downloading the code, navigating to the folder containing the *pom.xml* file, and building the code using *maven*, using the following commands:

```
git clone https://github.com/knowledge-technologies/clus
cd ClusProject
mvn clean package
```

When the build process finishes, the resulting executable .jar file will be available in the *./target* folder. In order to verify that the freshly-built .jar works properly, the following command can be executed:

```
java -jar target/clus-<version>-deps.jar
```

Running the jar file without providing any command-line parameters produces a short help message and terminates CLUSPLUS. Once built, the executable jar file can be copied and used anywhere, as long as the appropriate Java JRE (or JDK) is installed.

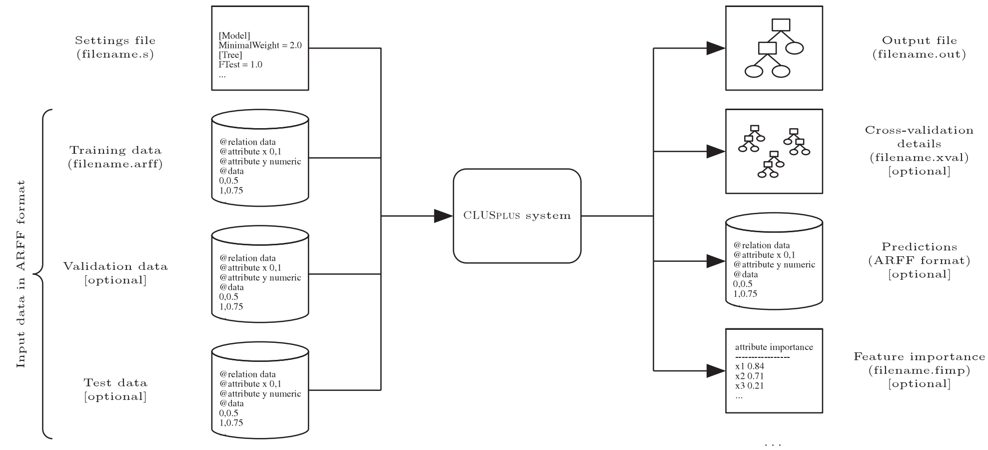

Fig. 1. Input and output files of CLUSPLUS.

2.2. *Running* CLUSPLUS

CLUSPLUS requires at least two input files (Fig. 1): (i) A settings file (e.g., `filename.s`) in the form of an INI config file,[3] and (ii) a dataset file in the attribute-relation file format[4] (`.arff`). The values of parameters for machine learning algorithms and path to the dataset(s) are specified in the settings file. Additionally, command-line switches are used to drive the behavior of the software, such as `-xval` to perform cross-validation or `-ssl` to perform semi-supervised learning. A complete description of the parameters and command-line switches is given in the user manual, located in the `docs/manual` folder within the repository. When the settings file and datasets have been prepared, CLUSPLUS can be executed by running the following command[5]:

```
java -jar clus.jar [switches] filename.s
```

The results of a CLUSPLUS run are written to an output file `filename.out`, which contains the values of parameter settings used for the run, followed by the evaluation metrics, and information about the learned models. The output files are explained in depth in the user manual.

2.3. *Development history*

The predictive clustering framework was first implemented within the TILDE software for learning logical decision trees from relational data [11]. This was followed by an implementation that worked on tabular data, named CLUS [10]. The two software packages were developed by Hendrik Blockeel and Jan Struyf from KU Leuven, who soon started collaborating with Sašo Džeroski from the Jožef Stefan Institute, Ljubljana (and his group) on applications [18] and further development of the software [19]. As a result of these joint efforts, CLUS was extended to handle multi-target regression problems [20] and hierarchical multi-label classification problems [21], as well as build tree ensembles (using the bagging and random forests approaches) [22] and was released on SourceForge.[6]

Between the years of 2013 and 2022, further development of CLUS was mainly carried out in Ljubljana within the group of Sašo Džeroski at the Jožef Stefan Institute. This development led to CLUSPLUS, described in the present article. The major novelties in CLUSPLUS as compared to CLUS include novel types of tree ensembles (based on extremely randomized trees [23], trees with random output subspaces [24] and rules with random output subspaces [25]), feature ranking (based on tree ensembles [26] and the distance-based RReliefF approach [27]), and semi-supervised learning (based on the self-training [28] and predictive clustering paradigms [29]).

3. Illustrative examples

We illustrate the use of CLUSPLUS through three related examples, where we showcase how to learn a decision tree in a supervised manner, a decision tree in a semi-supervised manner, and a random forest ensemble of trees in a semi-supervised manner. The task at hand is a multi-label classification task, where we need to predict the values of three logical functions y1, y2 and y3 of two binary variables (x1 and x2). y1 is the logical and, y2 the logical or, and y3 the exclusive or of the two inputs. The variable x3 is irrelevant. The design of CLUSPLUS allows us to use (almost) the same settings file `example.s` (Fig. 2, top left) in all three cases, while using command-line switches to control whether the algorithm will run in the ensemble and/or semi-supervised mode. The files mentioned here are provided in the code capsule and in the /ClusProject/docs/examples/ folder of the git repository https://github.com/knowledge-technologies/clus.

Learning a single decision tree. This is the most basic and default option of CLUSPLUS. If we simply call `java -jar clus.jar example.s` from the command line, it takes as input the mentioned settings file and the `example.arff` data file to produce the output in the `example.out` file (cf. middle-left and right-hand-side of Fig. 2). In the output file, the basic information for the run is specified, e.g., all the values of all the parameters, the list of the models, and some statistics for the run (e.g., induction time and error measures). Three models have been built: the Default (leaf) Model, a fully grown tree (Original Model) and its pruned version (Pruned Model).

The sections `Ensemble` and `SemiSupervised` of the settings file were ignored during this run, because the `-forest` and `-ssl` command-line switches were not used. Note also that there are some other possible sections of the parameter settings (e.g., `General`, where verbosity level and random seed can be specified), and there are many other parameters in the sections listed above (e.g., `TestSet` in section `Data`). However, since every parameter has its default value, we do not have to list them all in the settings file.

Semi-supervised learning of PCTs. In semi-supervised mode, CLUSPLUS can use partially labeled examples (where the values of

3 https://en.wikipedia.org/wiki/INI_file
4 https://www.cs.waikato.ac.nz/~ml/weka/arff.html
5 To simplify, we do not denote the .jar file version.
6 https://sourceforge.net/projects/clus/

```
--- example.s/exampleSSL.s --
[Data]
File = example.arff
[or exampleSSL.arff]
[Attributes]
Descriptive = 1-3
Target = 4-6

[Tree]
Heuristic = VarianceReduction

[Ensemble]
SelectRandomSubspaces = SQRT
EnsembleMethod = RForest
FeatureRanking = Genie3

[SemiSupervised]
SemiSupervisedMethod = PCT
-----------------------------
--- example.arff ------------
@relation example

@attribute x1 {0,1}
@attribute x2 {0,1}
@attribute x3 {0,1}
@attribute y1_x1ANDx2 {0,1}
@attribute y2_x1ORx2 {0,1}
@attribute y3_x1XORx2 {0,1}

@data
1,0,0,0,1,1
0,1,1,0,1,1
1,1,0,1,1,0
0,0,1,0,0,0
1,0,1,0,1,1
0,1,0,0,1,1
1,1,1,1,1,0
0,0,0,0,0,0
-----------------------------
--- example_ssl.arff (end) --
@data
1,0,0,0,1,1
0,1,1,0,1,1
1,1,0,1,1,0
0,0,1,0,0,0
1,0,1,?,1,1
0,1,0,?,1,?
1,1,1,?,?,?
0,0,0,?,?,?
-----------------------------
```

```
--- example.out ------------------------
Clus run example
****************
Date: 22/08/2022, 14:41
File: example.out
Attributes: 6 (input: 3, output: 3)

[Data]
File = example.arff
[Attributes]
Descriptive = 1-3
Target = 4-6
...
Run: 01
*******
Statistics
----------
FTValue (FTest): 1.0
Induction Time: 4.0000e-3 sec
Pruning Time: 2.0000e-3 sec
...
Training error
--------------
averageAUROC
   Default        : 5.000000e-1
   Original       : 1
   Pruned         : 1
...
Default Model
*************
[0,1,1] [6.0,6.0,4.0]: 8
Original Model
**************
x1 = 1
+--yes: x2 = 1
|       +--yes: [1,1,0] [2.0,2.0,2.0]: 2
|       +--no:  [0,1,1] [2.0,2.0,2.0]: 2
+--no:  x2 = 1
        +--yes: [0,1,1] [2.0,2.0,2.0]: 2
        +--no:  [0,0,0] [2.0,2.0,2.0]: 2
Pruned Model
************
x1 = 1
+--yes: x2 = 1
|       +--yes: [1,1,0] [2.0,2.0,2.0]: 2
|       +--no:  [0,1,1] [2.0,2.0,2.0]: 2
+--no:  x2 = 1
        +--yes: [0,1,1] [2.0,2.0,2.0]: 2
        +--no:  [0,0,0] [2.0,2.0,2.0]: 2
----------------------------------------
```

Fig. 2. Input and output files for CLUSPLUS on the logical function example.

some of the target variables are unknown) or unlabeled examples (where the values of all of the target variables are unknown). Note that the `exampleSSL.arff` data file contains such examples in the last four rows (cf. the bottom left of in the training data file Fig. 2). To build a PCT in SSL mode, the `-ssl` switch needs to be included into the command line: The command `java -jar clus.jar -ssl exampleSSL.s` would result in a single semi-supervised tree (output file `exampleSSL.out`).

Semi-supervised learning of a random forest of PCTs. When growing an ensemble of decision trees, e.g., random forests, we need to include the switch `-forest` in the command line: The command `java -jar clus.jar -forest -ssl exampleSSL.s` produces a random forest of SSL trees (output file `exampleSSLrf.out`). Since the settings also specify that a feature ranking should be computed (`Ensemble.FeatureRanking = Genie3`), we will additionally obtain an `exampleSSLTrees10Genie3.fimp` file where the feature importance values (as calculated by the Genie3 method) are listed.

4. Impact

As previously mentioned, CLUSPLUS is a machine learning toolbox with a unique set of capabilities, namely handling of several tasks of predicting structured outputs (i.e., multi-target regression, multi-label classification, and hierarchical multi-label classification), producing interpretable models, learning of ensemble models, providing feature importance scores and rankings, and exploiting unlabeled data via semi-supervised learning. The combination of these functionalities opens new (or poorly investigated) research directions, such as (semi-supervised) feature ranking for structured targets [15,26,30], or semi-supervised learning for hierarchical multi-label classification [14], and enables application of machine learning to problems ill-suited for conventional methods, such as learning from data partially labeled with structured targets [31]. CLUSplus was extensively compared to other available methods for complex predictive modeling tasks, demonstrating its competitive advantage in terms of predictive performance and scalability (see Refs. [13,32–34]).

The CLUSPLUS framework has enabled a large number of scholarly publications in many different scientific application domains, ranging from biology [35–44], chemistry [45,46], ecology [31, 47–54,54–58], image classification [58–61], space technologies [62–64], as well as to several methodological areas of computer science, namely decision tree and ensemble learning [13,20,21, 23,65–68], feature ranking [26,30,69,70], semi-supervised learning [28,29,71], and redescription mining [72,73].

The development of methods within the CLUSPLUS framework was a topic central to the Seventh Framework Programme project MAESTRA.[7] Furthermore, CLUSPLUS methods were used within the EU FET flagship Human Brain project.[8] Next, the use of CLUSPLUS methods yielded the winning solution to the European Space Agency's first machine learning competition that aimed to optimize the power consumption of the thermal system of the Mars Express spacecraft [62,63]. CLUSPLUS methods are included into the GalaxAI machine learning toolbox for interpretable analysis of spacecraft telemetry data from ESA's spacecraft MEX [74] and INTEGRAL [75]. CLUSPLUS was also applied in a wide range of diverse datasets and real-world problems demonstrating its effectiveness across different domains. For example, semi-supervised PCTs were applied to the medical problem of survival analysis [76] and quantitative structure–activity relationship (QSAR) modeling [45]. Ensembles of PCTs were applied to gene function prediction [44], drug design/repurposing for tuberculosis/salmonella [42], and for reversing lung fibrosis through inhibition of myofibroblast differentiation [46]. Finally, CLUSplus has been extensively used for modeling the ecology of different communities/biota, two recent examples being owls [77] and funghi found in urban sand [78].

The CLUS software,[9] the predecessor to CLUSPLUS, was downloaded 3085 times by individuals from 84 different countries. As we have described above, CLUSPLUS provides a number of unique additional functionalities. We thus expect that the CLUSPLUS framework will maintain and exceed the widespread use of its predecessor.

5. Conclusions

We present CLUSPLUS, a Java open source machine learning framework based on decision trees. The framework is specialized for predicting various types of structured outputs and provides a versatile set of functionalities, including the learning of interpretable models, ensemble learning, feature ranking, and semi-supervised learning, making it one of the most complete toolboxes for structured output prediction available. The modularity of the CLUSPLUS framework enables future extension to machine learning tasks or types of structured outputs not considered here, such as time-series prediction or multi-task learning.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

A link to the publicly available code repository is provided in the manuscript: https://github.com/knowledge-technologies/clus No data was used for the research described in the manuscript.

Acknowledgments

We are thankful to Celine Vens, Timo Aho, Hendrik Blockeel, Bernard Ženko, Jan Struyf, Leander Schietgaat, Aljaž Osojnik, Vanja Mileski, Valentin Gjorgjioski, Mitja Pugelj, Daniela Stojanova, and Tomaž Stepišnik for their contributions to the code of CLUS, most of which is included in CLUSPLUS.

A major part of the development of CLUSPLUS was supported by the Slovenian Research and Innovation Agency (ARRS; recently renamed to ARIS) via the Research Program Knowledge Technologies (grant P2-0103) and the European Commission (EC) via the project MAESTRA (grant 612944). In addition, SD is currently supported by the EC via the projects ASSAS (grant number 101059682), ELIAS (grant 101120237), INQUIRE (grant 101057499), PARC (grant 101057014), and TAILOR (grant 952215), as well as by ARRS via the projects J1-3033, J2-2505, J2-4452, J2-4660, J3-3070, J4-3095, J5-4575, J7-4636, J7-4637, and N2-0236.

References

[1] Kriegel H-P, Borgwardt K, Kröger P, Pryakhin A, Schubert M, Zimek A. Future trends in data mining. Data Min Knowl Discov 2007;15:87–97.
[2] Dietterich TG, Domingos P, Getoor L, Muggleton S, Tadepalli P. Structured machine learning: The next ten years. Mach Learn 2008;73(1):3–23.
[3] Chapelle O, Schölkopf B, Zien A. Semi-supervised learning. MIT Press; 2006.
[4] Witten IH, Frank E, Hall MA. Data mining: Practical machine learning tools and techniques. Morgan Kaufmann; 2006.
[5] Berthold MR, Cebron N, Dill F, Gabriel TR, Kötter T, Meinl T, et al. KNIME-the Konstanz information miner: version 2.0 and beyond. ACM SIGKDD Explor Newsl 2009;11(1):26–31.
[6] Demšar J, Curk T, Erjavec A, Gorup Č, Hočevar T, Milutinovič M, et al. Orange: Data mining toolbox in Python. J Mach Learn Res 2013;14:2349–53.
[7] Read J, Reutemann P, Pfahringer B, Holmes G. MEKA: A multi-label/multi-target extension to Weka. J Mach Learn Res 2016;17(21):1–5.
[8] Tsoumakas G, Spyromitros-Xioufis E, Vilcek J, Vlahavas I. Mulan: A java library for multi-label learning. J Mach Learn Res 2011;12:2411–4.
[9] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825–30.
[10] Blockeel H, Struyf J. Efficient algorithms for decision tree cross-validation. J Mach Learn Res 2002;3:621–50.
[11] Blockeel H, Raedt LD, Ramon J. Top-down induction of clustering trees. In: Proceedings of the 15th International conference on machine learning (ICML). Morgan Kaufmann; 1998, p. 55–63.
[12] Breiman L, Friedman J, Olshen R, Stone CJ. Classification and regression trees. Chapman & Hall/CRC; 1984.
[13] Kocev D, Vens C, Struyf J, Džeroski S. Tree ensembles for predicting structured outputs. Pattern Recognit 2013;46(3):817–33.
[14] Levatić J, Ceci M, Kocev D, Džeroski S. Semi-supervised predictive clustering trees for (Hierarchical) multi-label classification. 2022, arXiv.
[15] Petković M. Feature ranking for structured output prediction [Ph.D. thesis], Ljubljana, Slovenia: Jožef Stefan International Postgraduate School; 2020.
[16] Breiman L. Bagging predictors. Mach Learn 1996;24(2):123–40.
[17] Breiman L. Random forests. Mach Learn 2001;45(1):5–32.
[18] Blockeel H, Džeroski S, Grbović J. Simultaneous prediction of multiple chemical parameters of river water quality with TILDE. In: Proceedings of the 3rd European conference on principles and practice of knowledge discovery in databases (PKDD) - LNAI 1704. Springer; 1999, p. 32–40.
[19] Blokeel H, Bruynooghe M, Džeroski S, Ramon J, Struyf J. Hierarchical multi-classification. In: Proceedings of the International workshop on multi-relational data mining at KDD. 2002, p. 21–35.

[7] https://cordis.europa.eu/project/id/612944
[8] https://cordis.europa.eu/project/id/785907
[9] https://sourceforge.net/projects/clus/

[20] Struyf J, Džeroski S. Constraint based induction of multi-objective regression trees. In: Proceedings of the 4th International workshop on knowledge discovery in inductive databases (KDID) - LNCS, vol. 3933. Springer; 2006, p. 222–33.

[21] Vens C, Struyf J, Schietgat L, Džeroski S, Blockeel H. Decision trees for hierarchical multi-label classification. Mach Learn 2008;73(2):185–214.

[22] Kocev D, Vens C, Struyf J, Džeroski S. Ensembles of multi–objective decision trees. In: Proceedings of the 18th European conference on machine learning (ECML) – LNCS 4701. Springer; 2007, p. 624–31.

[23] Kocev D, Ceci M, Stepišnik T. Ensembles of extremely randomized predictive clustering trees for predicting structured outputs. Mach Learn 2020;109(11):2213–41.

[24] Breskvar M, Kocev D, Džeroski S. Ensembles for multi-target regression with random output selections. Mach Learn 2018;107:1673–709.

[25] Breskvar M, Džeroski S. Multi-target regression rules with Random Output Selections. IEEE Access 2021;9:10509–22.

[26] Petković M, Kocev D, Džeroski S. Feature ranking for multi-target regression. Mach Learn 2020;109(6):1179–204.

[27] Petković M, Kocev D, Džeroski S. Feature ranking with relief for multi-label classification: Does distance matter? In: Proceedings of the 21st International conference on discovery science (DS) - LNAI 11198. Springer International Publishing; 2018, p. 51–65.

[28] Levatić J, Ceci M, Kocev D, Džeroski S. Self-training for multi-target regression with tree ensembles. Knowl-Based Syst 2017;123:41–60.

[29] Levatić J, Kocev D, Ceci M, Džeroski S. Semi-supervised trees for multi-target regression. Inform Sci 2018;450:109–27.

[30] Petković M, Džeroski S, Kocev D. Feature ranking for semi-supervised learning. Mach Learn 2022;1–30.

[31] Nikoloski S, Kocev D, Levatić J, Wall DP, Džeroski S. Exploiting partially-labeled data in learning predictive clustering trees for multi-target regression: A case study of water quality assessment in Ireland. Ecol Inform 2021;61:101161.

[32] Madjarov G, Kocev D, Gjorgjevikj D, Džeroski S. An extensive experimental comparison of methods for multi-label learning. Pattern Recognit 2012;45:3084–104.

[33] Bogatinovski J, Todorovski L, Džeroski S, Kocev D. Comprehensive comparative study of multi-label classification methods. Expert Syst Appl 2022;203:117215.

[34] Mileski V, Džeroski S, Kocev D. Predictive clustering trees for hierarchical multi-target regression. In: Proceedings of the 16th International symposium on Advances in intelligent data analysis (IDA). Springer; 2017, p. 223–34.

[35] Slavkov I, Gjorgjioski V, Struyf J, Džeroski S. Finding explained groups of time-course gene expression profiles with predictive clustering trees. Mol Biosyst 2010;6:729–40.

[36] Stojanova D, Ceci M, Malerba D, Džeroski S. Using PPI network autocorrelation in hierarchical multi-label classification trees for gene function prediction. BMC Bioinformatics 2013;14:285.

[37] Schietgat L, Vens C, Struyf J, Blockeel H, Kocev D, Džeroski S. Predicting gene function using hierarchical multi-label decision tree ensembles. BMC Bioinformatics 2010;11(1):1–14.

[38] Levatić J, Brbić M, Perdih TS, Kocev D, Vidulin V, Šmuc T, et al. Phenotype prediction with semi-supervised classification trees. In: New frontiers in mining complex patterns. Cham: Springer International Publishing; 2018, p. 138–50.

[39] Babič MN, Zalar P, Ženko B, Džeroski S, Gunde-Cimerman N. Yeasts and yeast-like fungi in tap water and groundwater, and their transmission to household appliances. Fungal Ecol 2016;20:30–9.

[40] Skraban J, Dzeroski S, Zenko B, Mongus D, Gangl S, Rupnik M. Gut microbiota patterns associated with colonization of different Clostridium difficile ribotypes. PLoS One 2013;8(2):e58005.

[41] Zajc J, Džeroski S, Kocev D, Oren A, Sonjak S, Tkavc R, et al. Chaophilic or chaotolerant fungi: a new category of extremophiles? Front Microbiol 2014;5:708.

[42] Korbee CJ, Heemskerk MT, Kocev D, van Strijen E, Rabiee O, Franken KL, et al. Combined chemical genetics and data-driven bioinformatics approach identifies receptor tyrosine kinase inhibitors as host-directed antimicrobials. Nat Commun 2018;9(1):1–14.

[43] Mihelčić M, Šimić G, Babić Leko M, Lavrač N, Džeroski S, Šmuc T, et al. Using redescription mining to relate clinical and biological characteristics of cognitively impaired and Alzheimer's disease patients. PLoS One 2017;12(10):e0187364.

[44] Vidulin V, Šmuc T, Džeroski S, Supek F. The evolutionary signal in metagenome phyletic profiles predicts many gene functions. Microbiome 2018;6(1):1–21.

[45] Levatić J, Ceci M, Stepišnik T, Džeroski S, Kocev D. Semi-supervised regression trees with application to QSAR modelling. Expert Syst Appl 2020;158:113569.

[46] Ring NAR, Volpe MC, Stepišnik T, Mamolo MG, Panov P, Kocev D, et al. Wet-dry-wet drug screen leads to the synthesis of TS1, a novel compound reversing lung fibrosis through inhibition of myofibroblast differentiation. Cell Death Dis 2021;13(1):1–12.

[47] Džeroski S, Kobler A, Gjorgjioski V, Panov P. Using decision trees to predict forest stand height and canopy cover from LANDSAT and LIDAR data. In: Proceedings of the 20th International conference on informatics for environmental protection (EnviroInfo). Shaker Verlag; 2006, p. 125–33.

[48] Kampichler C, Džeroski S, Wieland R. Application of machine learning techniques to the analysis of soil ecological data bases: relationships between habitat features and Collembolan community characteristics. Soil Biol Biochem 2000;32(2):197–209.

[49] Stojanova D. Estimating forest properties from remotely sensed data by using machine learning [Master's thesis], Ljubljana, Slovenia: Jožef Stefan International Postgraduate School; 2009.

[50] Debeljak M, Kocev D, Towers W, Jones M, Griffiths B, Hallett P. Potential of multi-objective models for risk-based mapping of the resilience characteristics of soils: demonstration at a national level. Soil Use Manag 2009;25:66–77.

[51] Kocev D, Džeroski S, White M, Newell G, Griffioen P. Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. Ecol Model 2009;220(8):1159–68.

[52] Džeroski S, Demšar D, Grbović J. Predicting chemical parameters of river water quality from bioindicator data. Appl Intell 2000;13:7–17.

[53] Kocev D, Naumoski A, Mitreski K, Krstić S, Džeroski S. Learning habitat models for the diatom community in lake Prespa. Ecol Model 2010;221:330–7.

[54] Levatić J, Kocev D, Debeljak M, Džeroski S. Community structure models are improved by exploiting taxonomic rank with predictive clustering trees. Ecol Model 2015;306:294–304.

[55] Demšar D, Džeroski S, Larsen T, Struyf J, Axelsen J, Pedersen MB, et al. Using multi-objective classification to model communities of soil microarthropods. Ecol Model 2006;191(1):131–43.

[56] Kocev D, Naumoski A, Mitreski K, Krstić S, Džeroski S. Learning habitat models for the diatom community in Lake Prespa. Ecol Model 2010;221(2):330–7.

[57] Nikoloski S, Murphy P, Kocev D, Džeroski S, Wall DP. Using machine learning to estimate herbage production and nutrient uptake on Irish dairy farms. J Dairy Sci 2019;102(11):10639–56.

[58] Dimitrovski I, Kocev D, Loskovska S, Džeroski S. Fast and efficient visual codebook construction for multi-label annotation using predictive clustering trees. Pattern Recognit Lett 2014;38:38–45.

[59] Dimitrovski I, Kocev D, Loskovska S, Džeroski S. Hierarchical annotation of medical images. Pattern Recognit 2011;44(10–11):2436–49.

[60] Dimitrovski I, Kocev D, Loskovska S, Džeroski S. Hierarchical classification of diatom images using ensembles of predictive clustering trees. Ecol Inform 2012;7(1):19–29.

[61] Dimitrovski I, Kocev D, Loskovska S, Džeroski S. Improving bag-of-visual-words image retrieval with predictive clustering trees. Inform Sci 2016;329:851–65.

[62] Petković M, Boumghar R, Breskvar M, Džeroski S, Kocev D, Levatić J, et al. Machine learning for predicting thermal power consumption of the Mars Express Spacecraft. IEEE Aerosp Electron Syst Mag 2019;34:46–60.

[63] Breskvar M, Kocev D, Levatić J, Osojnik A, Petković M, Simidjievski N, et al. Predicting thermal power consumption of the Mars Express satellite with machine learning. In: Proceedings of 6th IEEE International conference on space mission challenges for information technology (SMC-IT). IEEE; 2017, p. 88–93.

[64] Petkovic M, Lucas L, Kocev D, Džeroski S, Boumghar R, Simidjievski N. Quantifying the effects of gyroless flying of the mars express spacecraft with machine learning. In: Proceedings of the 8th IEEE International conference on space mission challenges for information technology (SMC-IT). IEEE; 2019, p. 9–16.

[65] Kocev D, Struyf J, Džeroski S. Beam search induction and similarity constraints for predictive clustering trees. In: Proceedings of the 5th International workshop on knowledge discovery in inductive databases (KDID) - LNCS 4747. 2007, p. 134–51.

[66] Levatić J, Kocev D, Džeroski S. The importance of the label hierarchy in hierarchical multi-label classification. J Intell Inf Syst 2015;45(2):247–71.

[67] Breskvar M, Kocev D, Džeroski S. Ensembles for multi-target regression with random output selections. Mach Learn 2018;107(11):1673–709.

[68] Breskvar M, Kocev D, Džeroski S. Multi-label classification using random label subset selections. In: Proceedings of the 20th International conference on discovery science (DS) - LNAI 10558. Springer; 2017, p. 108–15.

[69] Slavkov I, Karcheska J, Kocev D, Džeroski S. HMC-ReliefF: feature ranking for hierarchical multi-label classification. Comput Sci Inf Syst 2018;15(1):187–209.

[70] Petković M, Džeroski S, Kocev D. Multi-label feature ranking with ensemble methods. Mach Learn 2020;109(11):2141–59.

[71] Levatić J, Ceci M, Kocev D, Džeroski S. Semi-supervised classification trees. J Intell Inf Syst 2017;49(3):461–86.
[72] Mihelčić M, Džeroski S, Lavrač N, Šmuc T. Redescription mining augmented with random forest of multi-target predictive clustering trees. J Intell Inf Syst 2018;50(1):63–96.
[73] Mihelčić M, Džeroski S, Lavrač N, Šmuc T. A framework for redescription set construction. Expert Syst Appl 2017;68:196–215.
[74] Kostovska A, Petković M, Stepišnik T, Lucas L, Finn T, Martinez-Heras J, et al. GalaxAI: Machine learning toolbox for interpretable analysis of spacecraft telemetry data. In: Proceedings of the 8th IEEE International conference on space mission challenges for information technology (SMC-IT). 2021, p. 44–52.
[75] Stepišnik T, Finn T, Simidjievski N, Southworth R, Belanger G, Martínez Heras JA, et al. Machine learning for effective spacecraft operation: Operating INTEGRAL through dynamic radiation environments. Adv Space Res 2022;69(11):3909–20.
[76] Roy B, Stepišnik T, The Pooled Resource Open-Access ALS, Vens C, Džeroski S, Clinical Trials Consortium, et al. Survival analysis with semi-supervised predictive clustering trees. Comput Biol Med 2022;141:105001.
[77] Ratajc U, Breskvar M, Džeroski S, Vrezec A. Differential responses of coexisting owls to annual small mammal population fluctuations in temperate mixed forest. Ibis 2022;164(2):535–51.
[78] Novak Babič M, Gunde-Cimerman N, Breskvar M, Džeroski S, Brandão J. Occurrence, diversity and anti-fungal resistance of fungi in sand of an urban beach in Slovenia—Environmental monitoring with possible health risk implications. J Fungi 2022;8(8):860.