scikit-multilearn: A scikit-based Python environment for performing multi-label classification

Piotr Szymański piotr.szymanski@{pwr.edu.pl,illimites.edu.pl}
Institute of Informatics
Wrocław University of Technology
Wrocław, Poland

illimites foundation
Wrocław, Poland

Tomasz Kajdanowicz tomasz.kajdanowicz@pwr.edu.pl
Institute of Informatics
Wrocław University of Technology
Wrocław, Poland

Abstract

scikit-multilearn is a Python library for performing multi-label classification. The library is compatible with the scikit/scipy ecosystem and uses sparse matrices for all internal operations. It provides native Python implementations of popular multi-label classification methods alongside novel framework for label space partitioning and division. It includes graph-based community detection methods that utilize the powerful igraph library for extracting label dependency information. In addition its code is well test covered and follows PEP8. Source code and documentation can be downloaded from `http://scikit.ml` and also via `pip`. The library follows scikit's BSD licencing scheme.

Keywords: Python, supervised learning, multi-label classification

1. Multi-label classification

The task of single-label classification concerns assigning at most one class to a given object out of one (single-class classification) or many (multi-class classification) possible classes. Multi-label classification deals with the problem of assigning a subset of available labels to a given observation. Such problems appear in multiple domains - article and website classification, multimedia annotation, music categorization, discovery of genomics functionalities and are performed by either transforming a problem into a single/multi-class classification problem or by adapting a single/multi-class method to take multi-label information into account. An excellent introduction to the field has been provided by Tsoumakas et al. (2009).

Madjarov et al. (2012) divides approaches to multi-label classification into three groups: method adaptation, problem transformation and ensembles thereof. The first idea is to adapt single-label methods to multi-label classification by modifying label assigning fragments in a single-label method. An example of this is introducing a multi-label version of a decision function in Decision Trees.

Problem transformation concentrates on converting the multi-label problem to one or more single-label problems. The most prominent examples include classifying each label separately (Binary Relevance, Classifier Chains) and treating each label combination as a separate class in one multi-class problem (Label Powerset).

Both of those approaches suffer not only from standard problems of machine learning such as over- and underfitting but also from the issue of label imbalance (see Sun et al. (2009)) or numerical anomalies related to label ordering when Bayesian approaches are used for taking correlations into account (see Read et al. (2011)). Ensemble methods aim to correct this by learning multiple classifiers trained on label subspaces (RAkEL, HOMER), observation subsets with pruning and replacement, or analysing various label orderings (CC, PCC).

As with every applied science, research in this field required environments for performing experiments. The most prominent multi-label classification library to date (in terms of method count and popularity) is implemented in JAVA on top of the `WEKA` library (Hall et al. (2009)), which serves as a basis of a multi-label extension `MULAN` (Tsoumakas et al. (2011b)), which is extended by `MEKA` (Read et al. (2016)) with pruned sets and classifier chains methods.

2. The scikit-multilearn library

Scikit-multilearn follows the idea of building a multi-label classification library on top of an existing classification solution. In the case of Python, the obvious choice for a base library is the scipy stack with scikit-learn. The primary goal of `scikit-multilearn` is to provide an efficient Python implementation of as many multi-label classification algorithms as possible both to the open source community and commercial users of the Python data science stack of `scikit/scipy`.

scikit compatibility `scikit-multilearn` aims to be compatible with the Python data science stack. It follows the scikit API and requirements specified in check_estimator code. Multi-label classifiers in scikit-multilearn inherit scikit-learn's BaseEstimator class and the ClassifierMixin. They can be thus easily incorporated into scikit pipelines and cross validations, evaluation measures, and feature space transformators. It is easy to use scikit-learn's extensive feature-space manipulation methods, single-label classifiers and classification evaluation functions with scikit-multilearn.

Sparsity Multi-label classification problems often do not provide cases for all possible label combinations - in most benchmark sets there are less than 5% labels per row on average. The output space is thus very sparse. Scikit-multilearn operates on sparse matrices internally yielding a large memory and speed boost.

Label space division One of the significant contributions of the library is a framework for performing both flat and hierarchical ensemble classification with different label space division strategies Szymański et al. (2016). It provides a label co-occurence graph generator that divides the label space using community detection methods from igraph. This approach is easily extendable to a plethora of network libraries (ex. graphtool by Peixoto (2014)), other clustering solutions in Python or just scikit's cluster package.

Interfacing other frameworks The library provides scikit-compatible wrapper to reference libraries such as `MEKA`, `MULAN`, and `WEKA` through `MEKA` when a need arises to use methods that have not yet been implemented in Python natively. Using these libraries via scikit-multilearn wrapper does not induce a licensing (GPL-BSD) conflict.

BSD licensing As innovations in machine learning happen in both academia and companies, it is important to create a library that allows both communities to grow and profit from common work. Thus, scikit-multilearn is released under the BSD license to allow the for-profit ecosystem to use the library while opening the possibility of sharing development and maintenance tasks.

Performance on par with competition Scikit-multilearn is as fast as its competition. We have tested MEKA, MULAN and scikit-mulilearn on 9 well-cited benchmark multi-label classification sets using RAkELd by Tsoumakas et al. (2011a) - RAndom k-labELsets method that partitions the label space randomly and libSVM provided Support Vector Machinese as base classifier. Friedman-Iman-Davenport non-parametric test did not find statistically significant differences between mean ranks of evaluated software, with p-value 0.4853.

	meka	mulan	scikit.multilearn
CAL500	2.47	11.08	1.67
Corel5k	66.78	71.27	177.66
bibtex	530.14	971.97	1094.25
delicious	7014.17	6449.90	6661.15
enron	35.76	29.11	24.42
genbase	50.30	27.28	2.34
mediamill	11182.84	8396.57	2515.33
medical	5.49	2.78	4.93
tmc2007-500	1247.28	975.05	1601.57

Table 1: Running time (s) of RAkEL under scikit-multilearn, Meka and Mulan, with SVM, k = 6, 10-fold cross validations.

3. Using scikit-multilearn

The steps for using scikit-multilearn are very simple: loading the data, selecting the method and performing classification. The library supports loading matrices from all well-established formats thanks to `scipy` and operates on scipy/numpy representations internally, converting to dense matrices or lists of lists only if required by a scikit-base classifier, if one is employed in the problem transformation scenario.

`scikit-multilearn` provides three sub packages following the established categorization of multi-label methods: method adaptation approach (in `skmultilearn.adapt`), problem transformation approach (in `skmultilearn.pt`) and ensembles of the two (in `skmultilearn.ensemble`). To use multi-label adapted version of a single-label method one only needs to import it and instantiate an object of the class. A problem transformation approach takes a scikit-learn compatible base classifier and optional information whether the base classifier supports sparse input. Ensemble methods require a classifier and relevant method's parameters. The classifier is trained using the fit method, which takes the training input and output matrices, and classification is performed using the predict method on test observations - just as in scikit-learn. The meka wrapper provides a scikit compatible classifier class in skmultilearn.meka.

4. Documentation and Availability

`Scikit-multilearn` is hosted on Github and managed via the fork and pull-request paradigm. Both user and developer documentation is available at `http://scikit.ml/docs`. The code is released under BSD licence using Github and releases are available via PyPi. Implementation follows PEP8 and is maintained with a high level of test coverage. Development is managed on the Github repository `scikit-multilearn`[1] with an accompanying discussion group `scikit-multilearn-dev`[2].

5. Conclusions

The presented library - scikit-multilearn - is currently the most extensive scikit-compatible multi-label classification library. It provides implementations of both the most popular algorithms and new families of methods such as network-based label space division approaches. It is fast thanks to performing transformations on sparse matrices internally and using well-optimized methods from scikit as base classifiers. scikit-multilearn integrates well with the Python data science stack of scipy. It also provides wrappers to meka/weka/mulan - the standard Java classification stack - and allows easy use of those methods with the rest of the Python stack. scikit-multilearn is downloaded couple of hundred times every month and has already been used in several bachelor and master theses.

Acknowledgments

I would like to thank Tomasz Kajdanowicz for inspiration to work on this library, Wojciech Stachowski for initial versions of BR/ML-kNN & Classifier Chains implementations, Christian Schulze for asking multiple questions, reporting and fixing bugs, Felipe Almeida for bugfixes and testing, and Fernando Benites for providing the implementation of ML-ARAM Brucker et al. (2011). The work was partially supported by Fellowship co-financed by European Union within European Social Fund; The European Commission under the 7th Framework Programme, Coordination and Support Action, Grant Agreement Number 316097 [ENGINE]; The National Science Centre the research projects no. DEC-2013/09/B/ST6/02317 and UMO-2016/21/N/ST6/02382.

References

Florian Brucker, Fernando Benites, and Elena Sapozhnikova. Multi-label classification and extracting predicted class hierarchies. *Pattern Recognition*, 44(3):724 – 738,

1. `https://github.com/scikit-multilearn/scikit-multilearn`
2. `https://groups.google.com/d/forum/scikit-multilearn-dev`

2011. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2010.09.010. URL //www.sciencedirect.com/science/article/pii/S0031320310004504.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009. ISSN 1931-0145. doi: 10.1145/1656274.1656278. URL `http://doi.acm.org/10.1145/1656274.1656278`.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9): 3084–3104, September 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2012.03.004. URL `http://www.sciencedirect.com/science/article/pii/S0031320312001203`.

Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare. 1164194. URL `http://figshare.com/articles/graph_tool/1164194`.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, December 2011. ISSN 0885-6125, 1573-0565. doi: 10.1007/s10994-011-5256-5. URL `http://link.springer.com/article/10.1007/s10994-011-5256-5`.

Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. MEKA: A multi-label/multi-target extension to Weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016. URL `http://jmlr.org/papers/v17/12-164.html`.

Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04): 687–719, 2009.

Piotr Szymański, Tomasz Kajdanowicz, and Kristian Kersting. How is a data-driven approach better than random choice in label space division for multi-label classification? *Entropy*, 18(8):282, 2016. ISSN 1099-4300. doi: 10.3390/e18080282. URL `http://www.mdpi.com/1099-4300/18/8/282`.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer, 2009.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011a. ISSN 1041-4347. doi: 10.1109/TKDE.2010.164. URL `http://doi.ieeecomputersociety.org/10.1109/TKDE.2010.164`.

Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, and Ioannis Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011b. URL `http://jmlr.org/papers/volume12/tsoumakas11a/tsoumakas11a.pdf`.