On the Impact of Dataset Complexity and Sampling Strategy in Multilabel Classifiers Performance

Francisco Charte[1(✉)], Antonio Rivera[2], María José del Jesus[2], and Francisco Herrera[1]

[1] Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain
{fcharte,herrera}@ugr.es

[2] Department of Computer Science, University of Jaén, Jaén, Spain
{arivera,mjjesus}@ujaen.es

http://sci2s.ugr.es, http://simidat.ujaen.es

Abstract. Multilabel classification (MLC) is an increasingly widespread data mining technique. Its goal is to categorize patterns in several non-exclusive groups, and it is applied in fields such as news categorization, image labeling and music classification. Comparatively speaking, MLC is a more complex task than multiclass and binary classification, since the classifier must learn the presence of various outputs at once from the same set of predictive variables. The own nature of the data the classifier has to deal with implies a certain complexity degree. How to measure this complexness level strictly from the data characteristics would be an interesting objective. At the same time, the strategy used to partition the data also influences the sample patterns the algorithm has at its disposal to train the classifier. In MLC random sampling is commonly used to accomplish this task.

This paper introduces TCS (*Theoretical Complexity Score*), a new characterization metric aimed to assess the intrinsic complexity of a multilabel dataset, as well as a novel stratified sampling method specifically designed to fit the traits of multilabeled data. A detailed description of both proposals is provided, along with empirical results of their suitability for their respective duties.

Keywords: Multilabel classification · Complexity · Metrics · Partitioning

1 Introduction

Unlike multiclass and binary classification, where the classifier has to predict only one output, multilabel classification (MLC) must learn the associations between the patterns' features and several outputs at once. Each output indicates if a certain label is relevant to the data sample or not, thus the algorithms have to work with a set of binary predictions. Nowadays MLC is being applied to

F. Martínez-Álvarez et al. (Eds.): HAIS 2016, LNAI 9648, pp. 500–511, 2016.
DOI: 10.1007/978-3-319-32034-2_42

automate tag suggestion [1], categorize text documents [2], label incoming images [3], etc. A introduction to MLC and a recent review on MLC techniques and related topics can be found in [4,5], respectively.

Most of the aforementioned tasks involve working with large multilabel datasets (MLDs) having disparate numbers of input features, instances, labels, combinations of labels, etc. Undoubtedly some of these traits, such as the number of instances, determine at some extent the time necessary to train a classifier. Beyond this fact, it would be desirable to know in advance the difficulties a certain MLD can present and how its complexity can affect the classifier performance.

A second circumstance which potentially affects MLC algorithms performance is the way MLDs have been partitioned. There are MLDs containing only a few samples, sometimes only one, as representatives of rare labels. Random sampling, which is the mainstream strategy used in the multilabel field, can throw these few samples all on either the training or the test partition. Both cases will probably decrease the performance of the classifier.

The main aim of this paper is to study how the complexity of MLDs and the sampling strategy impacts classification results. For doing so, two proposals are introduced:

- A new characterization metric, called TCS (*Theoretical Complexity Score*), will allow to know the complexity of an MLD in advance, prior to use it to train a classifier. It is computed from the basic MLD traits.
- A novel stratified sampling method for partitioning datasets, aiming to improve label distribution among training and test partitions, thus providing the classifier a fairer representation of each label. It is built upon an stratification strategy, grouping instances containing labels with similar frequencies.

This paper is structured as follows. Section 2 explains how different complexity factors influence classification results and introduces the TCS metric. In Sect. 3 the problems of random sampling are described, and a new stratified sampling method is presented. The suitability of these two proposals is experimentally tested in Sect. 4. Lastly, in Sect. 5 some conclusions are drawn.

2 Assessing a Multilabel Dataset Complexity

Data complexity [6] is an intensively studied aspect in different fields, including classification. How to measure it and its influence in specific problems, such as imbalanced [7] learning and noise filtering [8], have been already faced in traditional classification. Regarding MLC, some studies related to imbalance [9] measurement and other complexity facts, such as the concurrence among frequent and infrequent labels [10], have been also published.

The interest here is to determine an intrinsic and easily interpretable complexity metric for MLDs. In this context, complexity has to be understood as the set of traits of the MLD that will make the learned model both more ineffective and inefficient.

2.1 Factors Influencing the Complexity of MLDs

In order to design such a metric firstly the main traits of any MLD and its implications in learning a MLC model have to be analyzed. The considered factors are the following:

– **Number of data instances:** The number of rows in an MLD determine the amount of available patterns to train and then test any model. While it is true that a larger quantity of data samples also implies more time devoted to training, having more instances does not necessarily means that the resulting model will be more complex. In fact, training a classifier with enough representative patterns is usually associated to a better performance.
– **Number of input features:** In machine learning the curse of dimensionality [11] is a very well-known problem. As the number of input features growths, so does the dimensions of the space where the patterns are located. Working in a high-dimensional space makes more difficult tasks such as measuring distances among patterns and finding analytical solutions. Most MLDs have a large number of features, thus it is a factor to be taken into account.
– **Number of labels:** In traditional classification the algorithms only have one output to learn, whether it is binary or multiclass. By contrast, MLDs have hundreds or thousands of labels. The larger is the number of labels the more complex would be the model to generate. There are many MLC methods based on binarization techniques [12–15], and the number of labels has a direct impact in both the time used to train each binary classifier and the complexity of the overall algorithm. This, no doubt, is another factor to consider.
– **Number of labelsets:** The labels in an MLD appear producing different combinations, usually known as labelsets. The number of distinct labelsets is another aspect to bear in mind, since there are many MLC methods [16–19] based on training multiclass classifiers using the labelsets as class identifiers. Some of them produce simpler subsets of labels through pruning, random combinations and clustering approaches. In general, the larger is the amount of different combinations the more complex will be the final solution.

2.2 Theoretical Complexity Score

Building on the premises just enumerated, the proposed TCS metric is computed as indicated in (1). Let f be the number of input features, k the number of labels and ls the number of distinct labelsets. The logarithm of the product of these three factors will provide a theoretical complexity score, based only on the basic traits of the MLD[1] and easier to interpret than the raw product.

$$TCS(D) = \log(f \times k \times ls) \tag{1}$$

[1] In practice there would be other factors also influencing the classifiers performance, such as data sparseness, imbalance levels, concurrence among rare and frequent labels, etc.

The main goals in defining this metric, in addition to assess the complexity of an MLD, were ease of computation and interpretation, providing a straightforward measurement.

If there were an extremely simple MLD, having only one input attribute, two labels (on the contrary it would not be multilabel), and four different labelsets (the number of combinations two labels can produce), its TCS value would be $\log(1 \times 2 \times 4) \approx 2$. Table 1 shows the number of attributes, labels, labelsets and TCS for twenty MLDs commonly used on the literature, ordered according to their TCS value. As can be seen emotions and scene, two of the most popular MLDs, are the simplest ones. The two MLDs from genetics/proteins field, genbase and yeast, are more complex. Multimedia datasets, such as mediamill and corel5k, are located at the middle of the table. Some of the MLDs coming from text media, such as delicious, bookmarks, EURLex, etc., appear as the most complex ones.

Table 1. MLDs ordered according to their theoretical complexity score

Dataset	TCS	Attributes	Labels	Labelsets
emotions	9.364	72	6	27
scene	10.183	294	6	15
yeast	12.562	103	14	198
genbase	13.840	1 186	27	32
cal500	15.597	68	174	502
medical	15.629	1 449	45	94
enron	17.503	1 001	53	753
reuters	17.548	500	103	811
mediamill	18.191	120	101	6 555
corel16k001	19.722	500	153	4 803
corel5k	20.200	499	374	3 175
stackex-cs	20.532	635	274	4 749
bibtex	20.541	1 836	159	2 856
tmc2007	21.093	49 060	22	1 341
eurlex-sm	21.646	5 000	201	2 504
eurlex-dc	21.925	5 000	412	1 615
rcv1subset1	22.313	47 236	101	1 028
delicious	22.773	500	983	15 806
bookmarks	22.848	2 150	208	18 716
eurlex-ev	26.519	5 000	3 993	16 467

3 Sampling Multilabel Datasets

Almost all studies and proposals in the multilabel field imply some classification experimentation. Hold out, 2 × 5 and 10 folds cross validation are among the most common schemes, always with the same strategy to chose the patterns included in train and test partitions, random sampling. Despite the fact that some other sampling strategies [20] have been described for some time, the random approach is still the most used option.

Random sampling does a good work in selecting training and test patterns when most labels have enough representation in the MLD. However, sometimes it could be a risky strategy. That some labels have only one or two patterns representing them in the MLD is quite usual. Random sampling can place all of them either in the training or the test partition. To avoid this problem a stratified sampling approach can be used.

3.1 Stratified Sampling of MLDs

Stratified sampling is a usual technique in cross validation [21] for traditional classification. Since only one class is assigned to each instance, it is possible to compute the distribution of each class in the whole dataset and then draw the equivalent proportion of samples for training and testing. On the contrary, samples in an MLD are associated to several labels at once. If one instance is chosen for the train partition because it holds a certain label, it must be taken into account that some other labels are also included in the operation since they jointly appear with the selected one.

In [20] a stratified iterative method for sampling MLDs is proposed. It goes label by label through the MLDs, choosing individual samples and updating a set of counters. Due to its iterative nature it is a slow method when compared with random sampling, specially with MLDs having thousands of labels. Nonetheless, the authors stated that it was able to improve the classifier performance while dealing with some MLDs.

3.2 Stratified Random Sampling Method

The method outlined in Algorithm 1 is a new proposal to partition MLDs. It follows a stratified random sampling approach, but unlike the one in [20] it is not iterative by label.

In line 3 a weight is computed for each instance in the MLD. It is obtained as the product of the relative frequencies of active labels in the data sample. If one or more rare labels appear in it, the score will be very low. On the contrary, the occurrence of one or more common labels will produce a higher value. The number of active labels also influences this score. The larger is the set of labels in the instance the lower will be the score. The goal is to group instances with a similar label distribution relying in a simple procedure.

Once the instances have been ordered according to their weight (line 5), they are divided into as many strata as folds have been requested. Each training

Algorithm 1. Partitioning method based on stratified random sampling

```
function STRATIFIED.KFOLDS(MLD $D$, Integer $nfolds$)
    for each $instance$ $i$ in $D$ do
        $D_{i_w} \leftarrow \prod freq(l \in D_i)$                ▷ Weight for each instance
    end for
    $D \leftarrow \mathrm{SortBy}(D_w)$                ▷ Sort instances according to their weight
    ▷ Group samples with similar weight in separate strata
    for $i = 1$ to $nfolds$ do
        $strata_i \leftarrow D_{|D|/nfolds*(i-1)} - D_{|D|/nfolds*i}$
    end for
    for $i = 1$ to $nfolds$ do                ▷ Generate nfolds folds
        for $j = 1$ to $nfolds$ do                ▷ Taking part of the samples in each stratum
            $trainfold_i \leftleftarrows \mathrm{drawRandomly}(strata_j, |D|/nfolds \times (nfolds - 1))$
            $testfold_i \leftleftarrows strata_j - trainfold_j$                ▷ Remainder samples in stratum
        end for
    end for
    return ($trainfold$, $testfold$)
end function
```

partition gets a portion of each stratum proportional to the number of samples in D and the number of folds. The remainder samples in the stratum are given to the test partition. The samples in each stratum are randomly picked.

4 Experimentation

Aiming to validate the usefulness of the two proposals made in the previous sections, five MLDs with diverse TCS values have been selected from Table 1. Those are emotions, yeast, enron, stackex-cs and delicious. All of them can be downloaded from the R Ultimate Multilabel Dataset Repository [22], and they can be partitioned randomly or using the stratified strategy described in Sect. 3 by means of the mldr.datasets[2] R package.

The datasets were partitioned using 10 fcv, once randomly and once with stratified random sampling. These partitions were given as input to tree multilabel classifiers, one based on binarization (BR [12]), one based on label combinations (LP [16]), and one on lazy learning adapted to multilabel data (ML-kNN [23]). From the results produced by the classifiers three general performance metrics, Accuracy (2), Precision (3) and Recall (4) have been computed to analyze the meaningfulness of the TCS metric. Another two more specific metrics, MacroPrecision and MacroRecall, have been obtained to compare the two sampling strategies. The macro-averaging strategy (5) allows the calculation of any standard performance metric label by label, then averaging to obtain the final measure. In these equations n is the number of instances in the MLD, Y_i the real labelset associated to i-th instance, Z_i the predicted one, k the number of

[2] https://cran.r-project.org/web/packages/mldr.datasets/index.html.

labels in $\mathcal{L}$, and TP, FP, TN and FN stand for *True Positives*, *False Positives*, *True Negatives* and *False Negatives*, respectively.

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{2}$$

$$Precision = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Z_i|} \tag{3}$$

$$Recall = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i|} \tag{4}$$

$$MacroMet = \frac{1}{k}\sum_{l\in\mathcal{L}} EvalMet(TP_l, FP_l, TN_l, FN_l) \tag{5}$$

4.1 Influence of Complexity in Classifier Performance

To analyze how the intrinsic complexity of each MLD influences the classifiers performance, Figs. 1, 2 and 3 shows for each classifier the Accuracy, Precision and Recall values along with TCS. The x-axis corresponds to the five MLDs ordered according to their complexity.

From these plots observation that higher TCS values are correlated to worse performances can be easily deducted. For the LP algorithm the three evaluation metrics show a similar behavior, whereas for BR and ML-kNN Precision seems to be less affected than Recall and Accuracy. To formally analyze this relationship, a Pearson correlation test was applied over the TCS and performance values for

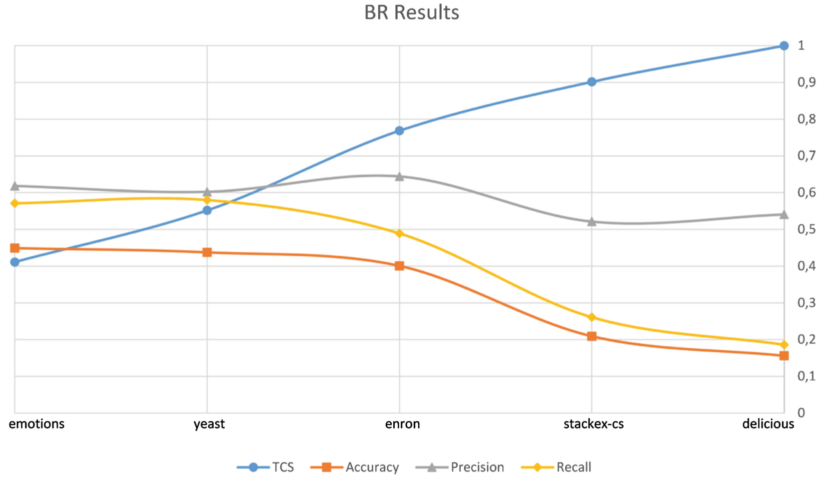

Fig. 1. Performance measures with respect to TCS values for the BR algorithm.

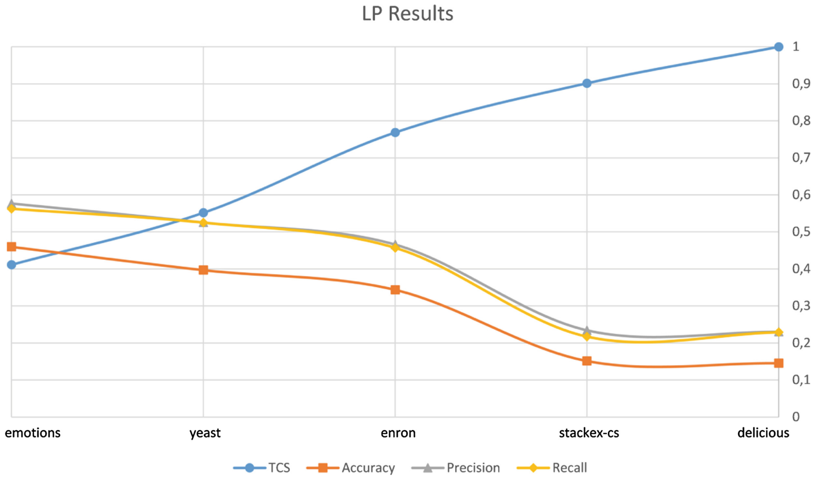

Fig. 2. Performance measures with respect to TCS values for the LP algorithm.

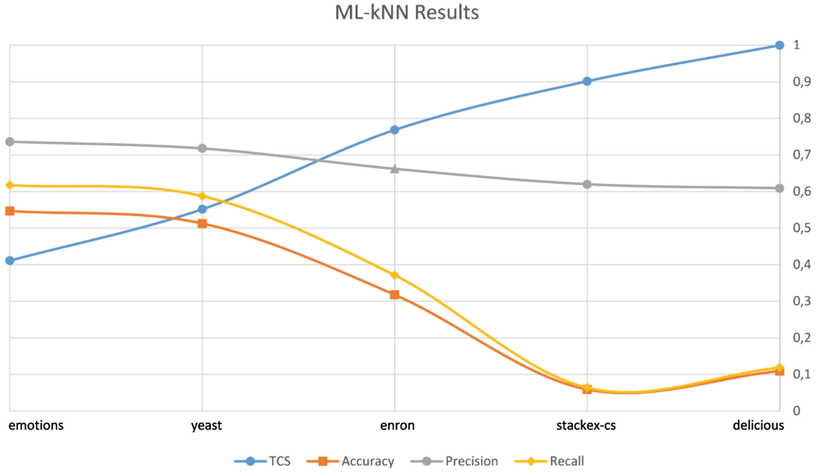

Fig. 3. Performance measures with respect to TCS values for the ML-kNN algorithm.

each metric and algorithm. The obtained results are the shown in Table 2. As can be seen, with the exception of Precision and BR, all the results are above 0.9 in absolute value, meaning that a strong correlation exists. The negative values imply an inverse relation, thus the higher is TCS the worse would be the result.

4.2 Influence of Sampling Strategy in Classifier Performance

Once the partitions for each MLD using the two sampling strategies are generated, it would be useful to know how potentially problematic cases affect each

Table 2. Pearson correlation test results

Algorithm	Accuracy	Precision	Recall
BR	−0, 91	−0, 67	−0, 93
LP	−0, 95	−0, 94	−0, 93
ML-kNN	−0, 96	−0, 99	−0, 95

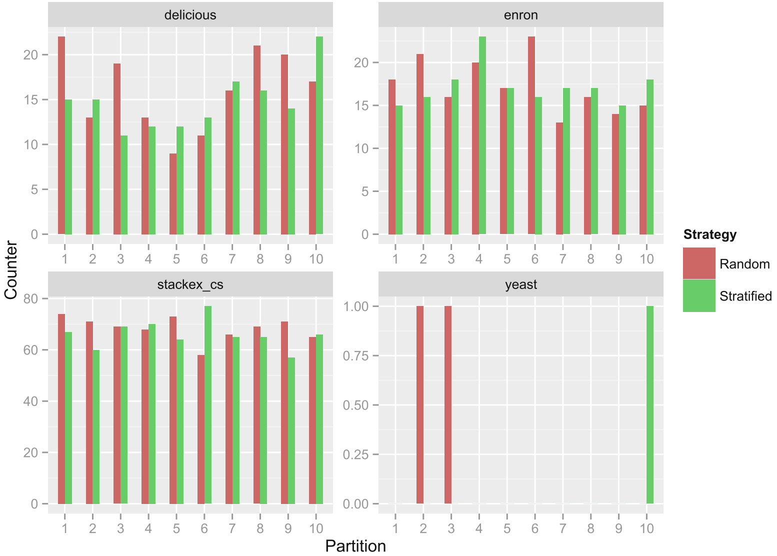

Fig. 4. Occurrences of problematic cases depending on the sampling strategy.

method. As mentioned above, some MLDs contain labels that could be considered as rare, since they only appear once or twice in the whole dataset. Using a 10 fcv scheme, it is easy that these singular cases fall down in the train partition almost always, leaving the test set with poor or null representation of these labels. The own sampling method guarantees that at least once they will appear in the test partition.

The plots in Fig. 4 show the amount of labels with one or none occurrences in the test set produced by each strategy. Here ”Random” refers to the classical random approach and ”Stratified” to the proposed stratified random sampling. The emotions MLD does not have any case, thus his plot would be empty. Looking at delicious, for instance, it can be verified that for folds 1, 3, 8 and 9 the random sampling clearly produces more problematic cases than the stratified approach. Only for fold 10 the result is definitely worse for the stratified strategy. With stackex-cs the differences appear to be smaller due to the y-axis scale. In more than half of the folds the stratified strategy worked better than the random one. The yeast MLD is not affected by the described problem as much as the other ones. There are two folds in which the random approach produced one

Table 3. Performance measures for each MLD, algorithm and sampling strategy

Dataset	Algorithm	Macro Precision		Macro Recall	
		Random	Stratified	Random	Stratified
delicious	BR	0.4882	**0.4919**	0.0696	**0.0705**
	LP	0.1113	**0.1114**	**0.1101**	0.1095
	ML-kNN	**0.6385**	0.6134	**0.0433**	0.0420
emotions	BR	0.5961	**0.5997**	0.5578	**0.5832**
	LP	**0.5761**	0.5573	0.5565	**0.5631**
	ML-kNN	**0.7439**	0.7122	**0.6051**	0.5837
enron	BR	0.4556	**0.4780**	**0.1684**	0.1667
	LP	**0.1855**	0.1696	**0.1657**	0.1536
	ML-kNN	0.5942	**0.6266**	0.0880	**0.0899**
stackex-cs	BR	**0.4026**	0.3864	**0.1160**	0.1156
	LP	**0.0964**	0.0937	**0.0911**	0.0847
	ML-kNN	**0.6242**	0.5876	**0.0200**	0.0184
yeast	BR	0.4425	**0.4576**	0.3817	**0.3971**
	LP	0.3764	**0.3784**	0.3762	**0.3814**
	ML-kNN	0.6783	**0.6803**	0.3503	**0.3515**

problematic case, against only one for the proposed stratified method. Lastly, enron has the most mixed situation, with large and small differences in both ways.

The results produced by the classifiers were, in general, better for the stratified strategy in those partitions where it produced less problematic cases. The same was applicable for the random approach. Since the results obtained from cross validation are always average values, these differences tend to compensate among them. These final evaluation measures are the shown in Table 3. Best values are highlighted in bold.

Overall there is a tie between the two strategies. Although there are MLDs working better with the stratified one, such as yeast, and others with the random alternative, such as stackex-cs, the remainder MLDs reflect a mixed behavior. Even though there are some noticeable differences between the results produced by the two strategies, most of them are in the order of a few thousandths.

5 Conclusions

The performance of a multilabel classifier is influenced by a plethora of circumstances, starting with the own model goodness, the learning process and the traits (imbalance, missing values, outliers, label concurrence, etc.) of the data used to train it. We hypothesized that two key aspects could be the inherent complexity of the data and the strategy used to partition the MLDs, and described two useful tools to face them.

With the proposed TCS metric the theoretical complexity of any MLD can be quickly and easily computed. As has been demonstrated with experimental results, a clear correlation between the TCS level and the performance of the tested MLC algorithms can be established. Therefore, this metric could be used to know in advance if an MLD would obtain better or worse classification results than others depending on their TCS values.

Regarding the sampling strategies to partition the datasets, the most used approach in MLC is the random way. It can produce some problems with certain MLDs, as has been explained, that could be solved with an stratified strategy. Such a method has been proposed, and its behavior has been compared with the standard random sampling. Although it clearly improved the balanced presence of rare labels among folds in some cases, the classifiers performance did not show fair overall differences. A further more extensive analysis, including additional MLDs, algorithms and sampling strategies, will be needed to determine which could be the best way for MLD partitioning.

Acknowledgments. This work was partially supported by the Spanish Ministry of Science and Technology under projects TIN2014-57251-P and TIN2012-33856, and the Andalusian regional projects P10-TIC-06858 and P11-TIC-7765.

References

1. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: QUINTA: a question tagging assistant to improve the answering ratio in electronic forums. In: EUROCON 2015 - International Conference on Computer as a Tool (EUROCON), pp. 1–6. IEEE (2015). doi:10.1109/EUROCON.2015.7313677
2. Klimt, B., Yang, Y.: The enron corpus: a new dataset for email classification research. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 217–226. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30115-8_22
3. Duygulu, P., Barnard, K., de Freitas, J.F.G., Forsyth, D.: Object recognition as machine translation: learning a lexicon for a fixed image vocabulary. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part IV. LNCS, vol. 2353, pp. 97–112. Springer, Heidelberg (2002). doi:10.1007/3-540-47979-1_7
4. Gibaja, E., Ventura, S.: A tutorial on multilabel learning. ACM Comput. Surv. **47**(3), 1–38 (2015). doi:10.1145/2716262
5. Gibaja, E., Ventura, S.: Multi-label learning: a review of the state of the art and ongoing research. Wiley Interdisc. Rev. Data Min. Knowl. Discovery **4**(6), 411–444 (2014). doi:10.1002/widm.1139
6. Ho, T.K., Basu, M.: Complexity measures of supervised classification problems. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 289–300 (2002)
7. Luengo, J., Fernández, A., García, S., Herrera, F.: Addressing data complexity for imbalanced data sets: analysis of smote-based oversampling and evolutionary undersampling. Soft. Comput. **15**(10), 1909–1936 (2011). doi:10.1007/s00500-010-0625-8
8. Sáez, J.A., Luengo, J., Herrera, F.: Predicting noise filtering efficacy with data complexity measures for nearest neighbor classification. Pattern Recogn. **46**(1), 355–364 (2013). doi:10.1016/j.patcog.2012.07.009

9. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: Addressing imbalance in multilabel classification: measures and random resampling algorithms. Neurocomputing **163**, 3–16 (2015). doi:10.1016/j.neucom.2014.08.091
10. Charte, F., Rivera, A., del Jesus, M.J., Herrera, F.: Concurrence among imbalanced labels and its influence on multilabel resampling algorithms. In: Polycarpou, M., Carvalho, A.C.P.L.F., Pan, J.-S., Woźniak, M., Quintian, H., Corchado, E. (eds.) HAIS 2014. LNCS, vol. 8480, pp. 110–121. Springer, Heidelberg (2014). doi:10.1007/978-3-319-07617-1_10
11. Bellman, R.: Dynamic programming and lagrange multipliers. Proc. Natl. Acad. Sci. U.S.A. **42**(10), 767 (1956)
12. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. Adv. Knowl. Discovery Data Min. **3056**, 22–30 (2004). doi:10.1007/978-3-540-24775-3_5
13. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. Artif. Intell. **172**(16), 1897–1916 (2008). doi:10.1016/j.artint.2008.08.002
14. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Mach. Learn. **73**, 133–153 (2008). doi:10.1007/s10994-008-5064-8
15. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. **85**, 333–359 (2011). doi:10.1007/s10994-011-5256-5
16. Boutell, M., Luo, J., Shen, X., Brown, C.: Learning multi-label scene classification. Pattern Recogn. **37**(9), 1757–1771 (2004). doi:10.1016/j.patcog.2004.03.009
17. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of the ECML/PKDD Workshop on Mining Multidimensional Data, Antwerp, Belgium, MMD 2008, pp. 30–44 (2008)
18. Read, J.: A pruned problem transformation method for multi-label classification. In: Proceedings of the 2008 New Zealand Computer Science Research Student Conference (NZCSRS 2008), pp. 143–150 (2008)
19. Tsoumakas, G., Vlahavas, I.P.: Random *k*-labelsets: an ensemble method for multilabel classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74958-5_38
20. Sechidis, K., Tsoumakas, G., Vlahavas, I.: On the stratification of multi-label data. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part III. LNCS, vol. 6913, pp. 145–158. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23808-6_10
21. Refaeilzadeh, P., Tang, L., Liu, H.: Cross-validation. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems, pp. 532–538. Springer, New York (2009). doi:10.1007/978-0-387-39940-9_565
22. Charte, F., Charte, D., Rivera, A., del Jesus, M.J., Herrera, F.: R ultimate multilabel dataset repository. In: Martínez-Álvarez, F., Troncoso, A., Quintián, H., Corchado, E. (eds.) HAIS 2016. LNCS (LNAI), vol. 9648, pp. 487–499 Springer, Switzerland (2016)
23. Zhang, M., Zhou, Z.: ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn. **40**(7), 2038–2048 (2007). doi:10.1016/j.patcog.2006.12.019