Chapter 3
Case Studies and Metrics

Abstract Multilabel classification techniques have been applied in many real-world situations in the last two decades. Each one represents a different case study for MLC, using one or more MLDs. After the general overview provided in Sect. 3.1, this chapter begins by briefly describing in Sect. 3.2 the most usual case studies found in the literature. As a result, a full list of available MLDs will be obtained, and the usual characterization metrics are explained and put in use with them in Sect. 3.3. Then, a practical use case is detailed in Sect. 3.4, running a simple MLC algorithm over a few MLDs. Lastly, the usual performance evaluation metrics for MLC are introduced in Sect. 3.5 and they are used to analyze the results obtained from this experiment.

3.1 Overview

The main application fields of MLC were introduced in the previous chapter from a global perspective. The goal in this chapter was to delve into each one of these fields, enumerating every one of the publicly available MLDs and stating where they come from. In addition to this basic reference information, it would be interesting to get some general characteristics for each MLD. For doing so, most of the characterization metrics described in the literature are going to be introduced, along with their formulations and discussion about their usefulness. Several extensive tables containing each measurement for every MLD will be provided.

In the following chapters, several dozens of MLC algorithms will be described, and some of them will be experimentally tested. Therefore, how to conduct such an experiment, and the way the results can be assessed to evaluate the algorithms' performance, are fundamental aspects. Once the available MLDs and their main traits are known, a basic kNN-based MLC algorithm is introduced and it is run to process some of these MLDs.

Multilabel predictive performance evaluation metrics have to deal with the presence of multiple outputs, taking into consideration the existence of predictions which are partially correct or wrong. As will be expounded, these metrics can be grouped

F. Herrera et al., *Multilabel Classification*,
DOI 10.1007/978-3-319-41111-8_3

into several categories according to distinct criteria. Then, most of the MLC evaluation metrics are explained along with their formulation, using them to assess the results obtained from the previous experiments.

3.2 Case Studies

In the previous chapter, the main application fields for MLC were portrayed. Attending to the grouping criterion then established, in this section most of the case studies found in the specialized literature will be enumerated. Table 3.1 summarizes these case studies, giving their original references and the place they can be downloaded from.[1]

Some of these case studies have associated several MLDs, whose names and characteristics will be analyzed later. The same MLD can be available in different formats,[2] for instance MULAN, MEKA, and KEEL, depending on the repository the user refers to.

The following subsections cover each MLC application field. The case studies are alphabetically enumerated inside each category conforming to the name of the MLD or set of MLDs belonging to them.

3.2.1 Text Categorization

Categorizing text documents into one or more categories is a very usual need. It is the task at the root of MLC. This is the reason for the existence of many datasets associated with this use case. The case studies mentioned below have been used in a considerable portion of the multilabel literature. Some of them have associated more than one MLD.

- `20ng`: This dataset, known as *20 Newsgroups*, has its origin in the task [28] of filtering news group messages. The dataset has become a classical problem for testing text clustering and text-labeling algorithms. It contains over a thousand entries for each one of 20 different news groups, making a total of almost 20 000 data instances. Some of the news groups are closely related, so some messages were cross-posted to more than one group. The input attributes, there are more than a thousand, are the terms extracted from all the documents. For each instance, those terms appearing in the message are set to 1, while the others are set to 0. This representation is known as boolean bag-of-words (BoW) model. There are

[1] All datasets are available at RUMDR (*R Ultimate Multilabel Dataset Repository)* [10], from which can be downloaded and exported to several file formats.

[2] The differences among the main file formats, all of them derived from the ARFF format used by WEKA, and how to use each of them, will be detailed in Chap. 9.

Table 3.1 Case studies and their categories and references

Case study	Category	References	Download
20ng	Text	[28]	[33]
bibtex	Text	[26]	[3, 43]
birds	Sound	[7]	[43]
bookmarks	Text	[26]	[3, 43]
cal500	Sound	[44]	[43]
corel	Image	[5, 20]	[3, 43]
delicious	Text	[40]	[3, 43]
emotions	Sound	[48]	[3, 33, 43]
enron	Text	[27]	[3, 33, 43]
EUR-Lex	Text	[30]	[43]
flags	Image	[24]	[43]
genbase	Gen/Bio	[19]	[3, 43]
imdb	Text	[32]	[33]
langlog	Text	[31]	[33]
mediamill	Video	[35]	[3, 9, 43]
medical	Text	[18]	[3, 33, 43]
nus-wide	Image	[17]	[43]
ohsumed	Text	[25]	[33]
rcv1v2	Text	[29]	[3, 9, 43]
reuters	Text	[31]	[33]
scene	Image	[6]	[3, 9, 33, 43]
slashdot	Text	[32]	[33]
stackexchange	Text	[15]	[12]
tmc2007	Text	[37]	[9, 43]
yahoo	Text	[47]	[43]
yeast	Gen/Bio	[21]	[3, 9, 33, 43]

20 labels, corresponding to the news groups the messages have been taken from. Only a handful of instances are assigned to more than one label.

- bibtex: Introduced in [26] as part of a tag recommendation task, it contains the metadata for bibliographic entries. The words that presented in the papers' title, authors names, journal name, and publication date were taken as input attributes. The full vocabulary consisted in 1 836 features. The data origin is Bibsonomy,[3] a specialized social network where the users can share bookmarks and BibTeX entries assigning labels to them. bibtex is the dataset generated from the data contained in the BibTeX entries, being associated with a total of 159 different

[3] http://www.bibsonomy.org.

labels. The boolean BoW model is used to represent the documents, so all features are binary indicating if a certain term is relevant to the document or not.

- `bookmarks`: This MLD comes from the same source [26] that the previous one. In this case, the data are obtained from the bookmarks shared by the users. Specifically, the URL of the resource, its title, date, and description are included into the dataset. The vocabulary consisted in 2 150 different terms, used as input features. The tags assigned to the bookmarks by the users, a total of 208, are taken as labels. The main difference between `bibtex` and `bookmarks` is the size of the MLD, having the latter more than ten times the number of instances that the former.
- `delicious`: The authors of this dataset [40] are the same of the previous one, and its nature is also similar to `bookmarks`. This time the links to Web pages were taken from the del.icio.us[4] portal. The page content for a set of popular tags was retrieved and parsed, and the resulting vocabulary was filtered to avoid non-frequent words. As a result, an MLD with almost a thousand labels was generated. The goal of the authors was proposing an MLC method able to deal with a so large number of labels.
- `enron`: The Enron corpus is a large set of email messages, with more than half a million entries, from which a dataset for automatic folder assignment research was generated [27]. The `enron` MLD is a subset of the previous dataset, with only 1 701 instances. Each one has as input features a BoW obtained from the email's fields, such as the subject and the body of the message. The labels correspond to the folders in which each message was stored into by the users. A total of 53 labels are considered.
- `EUR-Lex`: This case study is made up of three MLDs. The primary source is the European Union's database of legal documents, which includes laws, agreements, and regulations. Each document is classified in accordance with three criteria, EUROVOC descriptors, directory codes, and subject matters. For doing so, the header of the document indicates which descriptors, codes, and matters are relevant. Therefore, there are three multilabel tasks to accomplish. From this database, the authors of [30] generated the `eurlex-dc`, `eurlex-ev`, and `eurlex-sm` MLDs.[5] Unlike in most cases, reduction techniques were not applied aiming to obtain a limited number of labels. As a result, the `eurlex-ev` MLD has almost 4 000 of them. The three datasets have the same instances with the same set of 5 000 input features. These contain, in the version used in [30], the TF-IDF representation instead of BoW as the previous ones.
- `imdb`: The aim of this study [32] was to automatically classify movies into the proper genres, i.e., drama, comedy, adventure, or musical, among others. A total of 28 genres are considered. The input features were generated from the text gathered from the IMDB[6] database for each movie, relying in a boolean BoW

[4] https://delicious.com/.

[5] Additional information about how these MLDs were produced, including the software to do so, can be found at http://www.ke.tu-darmstadt.de/resources/eurlex.

[6] http://imdb.org.

representation. These texts contained a summary of the movies' plot, with a vocabulary made up of a thousand terms. Containing more than 120 000 instances, it is one of the largest MLDs publicly available.

- `langlog`: Introduced in [31], this MLD has been created from the posts published into the Language Log Forum,[7] a Web site for discussing language-related topics. As many other text MLDs, this also follows the boolean BoW model, with a total of 1 460 input features. The blog entries are categorized by 75 different labels.
- `medical`: The documents processed to produce this MLD are anonymized clinical texts, specifically the free text where the patient symptoms are described. A portion of the total corpus described in [18] was used to generate the MLD, with the text transformed into a BoW per document. The labels, a total of 45, are the codes from the International Classification of Diseases, precisely ICD-9-CM[8] codes.
- `ohsumed`: The origin of this dataset [25] is the Medline database, a text corpus from almost three hundred medical journals. The Ohsumed collection is a subset of the Medline dataset compiled in the Oregon Health Science University. The title and abstract texts of each article were processed and represented as BoW, producing a set of thousand input features. Each document is linked to one or more of the 23 main categories of the MeSH diseases ontology.[9] These categories are the labels appearing in the 13 929 instances that the MLD consists in.
- `rcv1v2`: This case study consists of five MLDs, being each one of them a subset of the original RCV1-v2 (*Reuters Corpus Volume 1 version 2*). The RCV1 text corpus was generated from the full text of English news published by Reuters along one year, from August 20, 1996, to August 19, 1997. Version 2 of this corpus is a corrected version introduced in [29]. Each entry was classified according to three categories, such as topic codes, industry codes, and region codes. A total of 101 different labels are considered. The vocabulary used as input features has 47 236 terms, represented as TF-IDF values. The full RCV1 corpus had 800 000 documents. 6 000 of them are provided in each one of the five subsets.
- `reuters`: Introduced in [31], it is also a subset of the RCV1 corpus. In this case, a feature selection method has been applied, taking only 500 input attributes instead of the more than 47 000 in `rcv1v2`. The goal was to work with more representative features. At the same time, the reduced set of attributes improves the speed of the learning process.
- `slashdot`: The source this MLD was generated from is Slashdot,[10] a well-known news portal mainly focused in technology and in science. The MLD was generated [32] taking the text from the news title and summary, producing a boolean BoW for each entry. The vocabulary has 1 079 terms. The tags used for categorize these entries, a total of 22, act as labels.
- `stackexchange`: The case study faced in [15] is a tag suggestion task for questions posted in specialized forums, specifically forums from the Stack Exchange

[7] http://languagelog.ldc.upenn.edu/nll/.

[8] http://www.cdc.gov/nchs/icd/icd9cm.htm.

[9] https://www.nlm.nih.gov/mesh/indman/chapter_23.html.

[10] http://slashdot.org.

network.[11] Six MLDs were generated from six different forums, devoted to topics such as cooking, computer science, and chess. The title and body of each question was text-mined, producing a frequency BoW. The tags assigned by the users to their questions were used as labels. The vocabulary for each forum is specific, being made of between 540 and 1 763 terms. These worked as input attributes. The labels are specific as well, ranging its number from 123 to 400.

- `tmc2007`: This dataset bore as a result of the SIAM Text Mining Workshop[12] in 2007 [37]. As many other text datasets, boolean BoW was chosen as a way of representing the terms appearing in documents. Those were aviation safety reports, in which certain problems during flights were described. The vocabulary consists of 49 060 different words, used as input features. Each report is tagged into one or more categories from a set of 22. These are the labels in the MLD.
- `yahoo`: The authors of [47] compiled for their study the Web pages referenced in 11 out of the 14 main categories of the classical Yahoo![13] Web index. Therefore, 11 MLDs are available for this case study. All of them use the boolean BoW representation, with features obtained from the pages referenced in the index. The number of words goes from 21 000 to 52 000, depending on the MLD. The subcategories that the pages belong to are used as labels. The number of labels is in the range 21–40.

3.2.2 Labeling of Multimedia Resources

Although text resources were the first ones to demand automated classification mechanisms, recently the need for labeling other kind of data, such as images, sounds, music, and video, has experimented a huge growth. By contrast with the case studies enumerated in the previous section, in which a common representation as BoW (whether they contain boolean values, frequencies, or TF-IDF values) is used, the following ones resort to disparate embodiments.

- `birds`: This MLD emerges from the case study described in [7], where the problem of identifying multiple birds species from acoustic recordings is tackled. The researchers used hundreds of sound snippets, recorded in nature at times of day with high bird activity. Between 1 and 5 different species appear in each snippet. The audio was processed with a 2D time-frequency segmentation approach, aiming to separate syllables overlapping in time. As a result, a set of features with the statistic profile of each segment is produced. Since a sound can be made up of several segments, the produced dataset is a multiinstance multilabel dataset.
- `cal500`: Tagging music tracks with semantic concepts is the task faced in [44], from which the `cal500` MLD is derived. The researchers took five hundred

[11] http://stackexchange.com/.

[12] http://web.eecs.utk.edu/events/tmw07/.

[13] http://web.archive.org/web/19970517033654/http://www9.yahoo.com/.

songs, from unique singers, and defined a vocabulary aimed to define aspects such as the emotions produced by the song, the instruments and vocal qualities, and music genre. These concepts, a total of 174, are used as labels. Each music track is assigned at least 3 of them and the average is above 26, which is a quite high number in the multilabel context. The input features were generated by sound segmentation techniques. A distinctiveness of this MLD is that no two instances are assigned the same combination of labels.

- `corel`: The original Corel dataset was used in two different case studies [5, 20] by the same authors, from which several MLDs have been obtained. The Corel dataset has thousands of images categorized into several groups. In addition, each picture is assigned a set of words describing its content. These pictures were segmented by the authors using the normalized cuts method, generating a set of blobs associated with one or more words. The input features, 500 in total, are the vectors resulting from the segmentation process. In [20] (`corel5k`), 5 000 instances were taken and there are 374 labels, since a minimum of occurrences was not established. The posterior study in [5] (`corel16k`) used 138 111 instances grouped into 10 subsets. A minimum of occurrences for each label was set, limiting its number to 153–174 depending on the subset.
- `emotions`: The origin of this dataset is the study conducted in [48], whose goal is to automatically identify the emotions produced by different songs. A hundred songs from each one of seven music styles were taken as input. The authors used the software tool described in [46] to extract from each record a set of rhythmic features and another one with timbre features. The union of these sets, after a process of feature selection, is used as input attributes. The songs were labeled by three experts, using the six main emotions of the Tellegen-Watson-Clark abstract emotional model. Only those songs where the assigned labels coincide were retained, reducing the number of instances from the original 700 to 593.
- `flags`: This MLD is considered as a toy dataset, since it only has 194 instances with a set of 19 inputs features and 7 labels. The original version can be found in the UCI repository.[14] In [24] several of its attributes, the ones indicating which colors appear in the flag or if it contains a certain image or text, were defined as labels. The remainder attributes, including the zone and land mass the country belongs to, its area, religion, population, etc., are established as input features.
- `mediamill`: It was introduced in [35] as a challenge for video indexing. The data consist of a collection of video sequences, taken from the TREC Video Retrieval Evaluation,[15] from which a set of 120 features have been extracted. This set of features is the concatenation of several similarity histograms extracted from the pixels of each frame. The goal was to discover what semantic concepts are associated with each entry, among a set of 101 different labels. Some of these concepts refer to environments, such as road, mountain, sky, or urban, others to physical objects, such as flag, tree, and aircraft. A visual representation of these 101 concepts can be fond in [35].

[14] https://archive.ics.uci.edu/ml/datasets/Flags.

[15] http://www-nlpir.nist.gov/projects/trecvid/.

- `nus-wide`: The famous Flickr[16] social network, in which millions of users publish their photographs every day, is the origin for the NUS-WIDE dataset, created by NUS's Lab for Media Search. Each image was segmented extracting color histogram, correlation histogram, edge direction, textures, etc. The resulting MLD has 269 648 instances, and two versions of the MLD with different features representation are available. The first one, known as `nus-wide-BoW`, used clustering to produce a 500 dimensional vector of visual words (real values). The second one, named `nus-wide-VLAD`, the vectors have 128 dimensions and are encoded as cVLAD+ features [36] (real values). In both, each instance has an initial attribute containing the name of the file where the image was stored into. Each image was manually annotated using a 81 items vocabulary, with terms such as animal, house, statue, and garden. These are the labels of the MLD.
- `scene`: This MLD is also related to image labeling, specifically to scene classification. The set of pictures was taken from the Corel dataset and some personal ones by the authors [6] were also included. The MLD is made up of 400 pictures for each main concept, beach, sunset, field, fall foliage, mountain, and urban. Therefore, six non-exclusive labels are considered. The images are transformed to the CIE Luv color space, known for being perceptually uniform, and latter segmented into 49 blocks, computing for each one of them values such as the mean and variance. The result is a vector of 294 real-value features in each instance.

3.2.3 Genetics/Biology

This is the area with less publicly available datasets, which is not surprising due to its complexity. There are two MLDs, one focused in predicting the class of proteins and another one for classifying genes in line with their functional expression.

- `genbase`: The authors of [19] produced this MLD compiling information for 662 different proteins. The Prosite access number[17] was used to identify the 1 185 motif patterns and profiles used as input features. All of them are nominal, taking only the YES or NO values. This way the motifs and profiles present in each protein are indicated. 27 different protein classes are considered, being each protein associated with one or more of them. The PDOC protein class identifiers are used as label names. Something to be taken into account while using this MLD is the presence of one additional feature, the first one, that uniquely identifies each instance.
- `yeast`: In this case [21], the goal was to predict the functional expression for a set of genes. The input features for each gene come from microarray expression data, with a 103 real values vector per instance. A subset of 14 functional classes, whose origin is the Comprehensive Yeast Genome Database,[18] are selected and

[16] https://www.flickr.com/.

[17] http://prosite.expasy.org/prosite.html.

[18] http://www.ncbi.nlm.nih.gov/pubmed/15608217.

used as labels. Since each gene can express more than one function at once, in fact this is the usual situation, the result is a dataset with multilabel nature.

3.2.4 Synthetic MLDs

Even though there are a quite large collection of MLDs publicly available, in some situations it can be desirable to work with datasets that have certain characteristics. For instance, if we were designing an algorithm to deal with noisy data it would be interesting to test it with MLDs having different noise levels. This is a trait that could be modeled by generating custom synthetic datasets.

Despite the aforementioned need, which has been demanded by several authors in some papers, there is a lack of tools to produce synthetic MLDs when compared with utilities with the same aim for traditional classification. In most cases, internal programs are used to generate these artificial datasets, and only the characteristics of the data are explained. Fortunately, there are some exceptions, such as the Mldatagen program[19] described in [38].

Since they are created by a program, an a priori limit in the number of MLDs that can be created does not exist. They can hold any number of instances, attributes, and labels but, unlike the enumerated in the previous sections, they do not represent any real situation.

3.3 MLD Characteristics

Before attempting to build a classification model to solve a specific problem, it is important to analyze the main characteristics of the data available to accomplish this task. Understanding the inner traits of the data usually will allow the selection of the best algorithm, parameters, etc. Revealing these traits is the aim of the specific characterization metrics for MLDs defined in the literature.

In the following subsections, many of the available characterization metrics are defined, providing their mathematical expressions, and detailing their usefulness. Many of them will be further applied to the MLDs associated with the previous case studies, and certain facts will be discussed. The nomenclature stated in Sect. 2.2 will be used in all equations.

[19] http://sites.labic.icmc.usp.br/mldatagen/.

3.3.1 Basic Metrics

The main difference between traditional and multilabel classification comes from the fact that in the latter each instance is associated with a set of labels. This is the reason behind the first specific metrics designed for MLDs, whose purpose is to assess the *multilabelness* of the data, in other words determining the extent at which the samples in the dataset have more than one label.

An obvious way to calculate such a measure consists in counting the number of labels relevant to every instance in the dataset, then averaging the sum to know the mean number of labels per instance. This simple metric was introduced in [39] as label cardinality or simply *Card* (3.1).

$$Card\,(D) = \frac{1}{n}\sum_{i=1}^{n}|Y_i| \tag{3.1}$$

In this context, n denotes the number of instances in the MLD D, Y_i the labelset of the ith instance, and k the total number of labels considered in D. The higher is the *Card* level, the larger is the number of active labels per instance. As a consequence, MLDs with low *Card* values, near 1.0, would denote that most of its samples have only one relevant label. Therefore, it would be a dataset with little multilabelness nature. On the opposite side, high *Card* values state that the data are truly multilabeled. As a general rule, high *Card* values are linked to MLDs which have large sets of labels, yet the contrary is not always true.

Since *Card* is a metric influenced by the size of the set of labels used by each MLD, and it is expressed using the number of labels as measurement unit, a normalized version (3.2) was also proposed. By dividing *Card* by the number of labels in the MLD, a dimensionless metric, known as label density (*Dens*), is obtained. Usually, a high *Dens* value indicates that the labels in the MLD are well represented in each instance. By contrast, low *Dens* values denote more dispersion, with only a small subset of the labels present in most instances.

$$Dens\,(D) = \frac{1}{k}\frac{1}{n}\sum_{i=1}^{n}|Y_i| \tag{3.2}$$

Another way of assessing the multilabelness of a dataset would be by means of the P_{min} metric (3.3) introduced in [45]. This is simply the percentage of instances in the MLD with only one active label. Intuitively, a high P_{min} value would denote that a large proportion of instances are single labeled.

$$P_{min}\,(D) = \sum_{y' \in Y/|y'|=1}\frac{|y'|}{n} \tag{3.3}$$

Subsets of the labels in the set $\mathcal{L}$ appear in the instances of D forming labelsets. Theoretically 2^k different labelsets could exists, but in practice the number of unique (distinct) labelsets is limited by the number of instances in D. Thus, the number of unique combinations is limited by the expression $min(n, 2^k)$. The effective number of distinct labelsets in an MLD is an indicator of the uniformity in the labels distribution among the samples. The higher the number is, the more irregularly the labels appear in the data instances. The number of distinct labelsets is also known as label diversity (*Div*), and it can also be normalized dividing it by the number of instances.

Furthermore, the frequency of each labelset appears in the MLD may be also an interesting information. Even though the total number of distinct labelsets is not high, if many of them only appear once, associated with one instance, this could lead to some difficulties during the learning process. In addition, the analysis of the labelsets provides information related to dependencies among labels.

Besides the previous ones, in [11] other standard statistical metrics, such as the coefficient of variation, kurtosis, and skewness, are used to characterize how the labels in an MLD are distributed. The joint use of all these metrics can help in gaining insight into this problem.

3.3.2 Imbalance Metrics

The presence of class imbalance in a dataset is a challenge for most learning algorithms. This problem will be analyzed in Chap. 8 in the context of MLC. As will be seen, most MLDs suffer from label imbalance. This means that some labels are much more frequent than others, and it is being an aspect interesting to appraise due to its impact in classification results. Three different metrics to assess label imbalance are proposed in [14], named *IRLlbl* (3.4), *MaxIR* (3.5) and *MeanIR* (3.6). In (3.4), the operator $[\![expr]\!]$ denotes de Iverson bracket. It will return 1 if the expression inside is true or 0 otherwise.

$$IRLbl(l) = \frac{\max\limits_{l' \in L} \left(\sum_{i=1}^{n} [\![l' \in Y_i]\!] \right)}{\sum_{i=1}^{n} [\![l \in Y_i]\!]}. \tag{3.4}$$

With the *IRLbl* metric, it is possible to know the imbalance level of one specific label. This is computed as the proportion between the number of appearances of the most common label and the considered label. Therefore, for the most common label $IRLbl = 1$. For least frequent labels, the level always will be greater than 1. The higher the *IRLbl*, the rarer is the label presence in the MLD. The goal of the *MaxIR* metric was obtaining the maximum imbalance ratio. In other words, the proportion of the most common label against the most rare one.

$$MaxIR = \max_{l \in L} (IRLbl(l)) \tag{3.5}$$

$$MeanIR = \frac{1}{k} \sum_{l \in L} IRLbl(l). \tag{3.6}$$

Usually a global assessment of the imbalance in the MLD is desired. This metric, named *MeanIR*, is calculated by averaging the *IRLbl* of all labels. Despite the usefulness of this metric by itself, some dispersion measure, such as standard deviation or coefficient of variation, should also be included. A high *MeanIR* could be due to a relatively high *IRLbl* for several labels, but also by cause of extreme imbalance levels for only some labels. In this context, the *CVIR* (3.7) metric provides the additional information needed to know the cause.

$$CVIR = \frac{IRLbl\sigma}{MeanIR}, \quad IRLbl\sigma = \sqrt{\frac{1}{k-1} \sum_{l \in L} (IRLbl\,(l) - MeanIR)^2} \tag{3.7}$$

3.3.3 Other Metrics

Besides the already aforementioned, some other characterization metrics have been proposed in the literature to assess specific qualities of the MLDs. In [13], the *SCUMBLE* metric is introduced as a way to measure the concurrence among very frequent and rare labels. A score is individually computed for each instance (3.8). This score is based on the Atkinson index [4] and the *IRLbl* metric introduced in the previous section. The former is an econometric measure aimed to evaluate income inequalities among the population. In this context, monetary quantities have been replaced by imbalance ratios, provided by the *IRLbl* metric. The result is a value in the [0, 1] range indicating if all the labels in the instance have similar frequencies in the MLD, low values, or by the contrary there are significant differences, the result would be a higher value. The global *SCUMBLE* measure (3.9) is obtained by averaging the score for all instances in the MLD. How these metrics have been the foundation for developing new MLC algorithms will be explained in Chap. 8. As a general rule, higher *SCUMBLE* values denote harder MLDs to learn from.

$$SCUMBLE_{ins}\,(i) = 1 - \frac{1}{\overline{IRLbl_i}} \left(\prod_{l \in L} IRLbl_{il} \right)^{(1/k)} \tag{3.8}$$

$$SCUMBLE\,(D) = \frac{1}{n} \sum_{i=1}^{n} SCUMBLE_{ins}\,(i) \tag{3.9}$$

The *TCS* (3.10) metric is presented in [16] aiming to facilitate a theoretical complexity indicator. It is calculated as the product of the number of input features, number of labels, and number of different label combinations. To avoid working with very large values, whose interpretation and comparison would be not easy, the *log* function is used to adjust the scale of the previous product. The goal was to determine which MLDs would present a harder work to the preprocessing an learning algorithms. Unlike *SCUMBLE*, *TCS* values are not upper bounded. The higher the value, the more complex would be to process the MLD.

$$TCS(D) = \log(f \times k \times |unique\ labelsets|) \tag{3.10}$$

3.3.4 Summary of Characterization Metrics

Once the main characterization metrics have been defined, they can be used to analyze the MLDs corresponding to the case studies enumerated in Sect. 3.2. Tables 3.2, 3.3, and 3.4 summarize most of these metrics for the MLDs corresponding to case studies from the text, multimedia, and genetics fields, respectively. The columns show, from left to right, **Dataset**: name of the MLD, **n**: number of instances, **f**: number of input attributes, **k**: number of labels, **LSet**: number of distinct labelsets, **Card**: label cardinality (*Card*), **Dens**: label density (*Dens*), **MeanIR**: mean imbalance ratio (*MeanIR*), and **SCUMBLE**: imbalanced labels concurrence level (*SCUMBLE*).

The MLDs from text case studies clearly share a common trait, as almost all of them have a high number of input features, in the range of thousands of them with few exceptions. This is due to the techniques used to mining the text, which produce large collections of words and their frequencies. Many of them also have several hundreds of labels. This, when combined with a large number of instances, also produces a huge amount of labelsets. It is the case with MLDs such as `bookmarks` or `delicious`. Comparatively, the number of features, labels, and labelsets is lower in the datasets coming from multimedia and genetics case studies.

Regarding the *Card* metric that indicates the mean number of labels per instance, most MLDs are in the $[1, 5]$ interval. Some MLDs, such as `20ng`, `langlog`, `slashdot`, `yahoo-reference`, `birds`, and `scene`, are only slightly above 1, meaning that most of its instances are associated with only one label. These would be the less representative cases of what should be a multilabel scenario, since they are closer to a multiclass one. There are a pair of extreme cases in the opposite side. The *Card* values for `delicious` and `cal500` are above 19 and 26, respectively. These MLDs are truly multilabel, with a remarkable average number of active labels in each instance. Halfway between the previous utmost scenarios, the remainder MLDs present the most typical *Card* values, between 2 and 5 labels per instance in average.

Dens is a metric closely related to *Card*. In general, most MLDs have *Dens* values below 0.1. Only those with a very limited set of labels, such as `emotions`, `flags`, or `scene`, or a very high *Card*, such as `cal500`, get a high label density. Therefore,

Table 3.2 Main characteristics of MLDs from text classification case studies

Dataset	n	f	k	LSet	Card	Dens	MeanIR	SCUMBLE
20ng	19 300	1 006	20	55	1.029	0.051	1.007	0.000
bibtex	7 395	1 836	159	2 856	2.402	0.015	12.498	0.094
bookmarks	87 856	2 150	208	18 716	2.028	0.010	12.308	0.060
delicious	16 105	500	983	15 806	19.017	0.019	71.052	0.532
enron	1 702	1 001	53	753	3.378	0.064	73.953	0.303
eurlex-dc	19 348	5 000	412	1 615	1.292	0.003	268.930	0.048
eurlex-ev	19 348	5 000	3 993	16 467	5.310	0.001	396.636	0.420
eurlex-sm	19 348	5 000	201	2 504	2.213	0.011	536.976	0.182
imdb	120 919	1 001	28	4 503	2.000	0.071	25.124	0.108
langlog	1 460	1 004	75	304	1.180	0.016	39.267	0.051
medical	978	1 449	45	94	1.245	0.028	89.501	0.047
ohsumed	13 929	1 002	23	1 147	1.663	0.072	7.869	0.069
rcv1subset1	6 000	47 236	101	1 028	2.880	0.029	54.492	0.224
rcv1subset2	6 000	47 236	101	954	2.634	0.026	45.514	0.209
rcv1subset3	6 000	47 236	101	939	2.614	0.026	68.333	0.208
rcv1subset4	6 000	47 229	101	816	2.484	0.025	89.371	0.216
rcv1subset5	6 000	47 235	101	946	2.642	0.026	69.682	0.238
reuters	6 000	500	103	811	1.462	0.014	51.980	0.052
slashdot	3 782	1 079	22	156	1.181	0.054	17.693	0.013
stackex-chemistry	6 961	540	175	3 032	2.109	0.012	56.878	0.187
stackex-chess	1 675	585	227	1 078	2.411	0.011	85.790	0.262
stackex-coffee	225	1 763	123	174	1.987	0.016	27.241	0.169
stackex-cooking	10 491	577	400	6 386	2.225	0.006	37.858	0.193
stackex-cs	9 270	635	274	4 749	2.556	0.009	85.002	0.272
stackex-philosophy	3 971	842	233	2 249	2.272	0.010	68.753	0.233
tmc2007	28 596	49 060	22	1 341	2.158	0.098	15.157	0.175
tmc2007-500	28 596	500	22	1 172	2.220	0.101	17.134	0.193
yahoo-arts	74 840	23 146	26	599	1.654	0.064	94.738	0.059
yahoo-business	11 214	21 924	30	233	1.599	0.053	880.178	0.125
yahoo-computers	12 444	34 096	33	428	1.507	0.046	176.695	0.097
yahoo-education	12 030	27 534	33	511	1.463	0.044	168.114	0.042
yahoo-entertainment	12 730	32 001	21	337	1.414	0.067	64.417	0.039
yahoo-health	9 205	30 605	32	335	1.644	0.051	653.531	0.092
yahoo-recreation	12 828	30 324	22	530	1.429	0.065	12.203	0.030
yahoo-reference	8 027	39 679	33	275	1.174	0.036	461.863	0.049
yahoo-science	6 428	37 187	40	457	1.450	0.036	52.632	0.058
yahoo-social	12 111	52 350	39	361	1.279	0.033	257.704	0.049
yahoo-society	14 512	31 802	27	1 054	1.670	0.062	302.068	0.096

Table 3.3 Main characteristics of MLDs from multimedia resources classification case studies

Dataset	n	f	k	LSet	Card	Dens	MeanIR	SCUMBLE
birds	645	260	19	133	1.014	0.053	5.407	0.033
cal500	502	68	174	502	26.044	0.150	20.578	0.337
corel5k	5 000	499	374	3 175	3.522	0.009	189.568	0.394
corel16k001	13 766	500	153	4 803	2.859	0.019	34.155	0.273
corel16k002	13 761	500	164	4 868	2.882	0.018	37.678	0.288
corel16k003	13 760	500	154	4 812	2.829	0.018	37.058	0.285
corel16k004	13 837	500	162	4 860	2.842	0.018	35.899	0.277
corel16k005	13 847	500	160	5 034	2.858	0.018	34.936	0.285
corel16k006	13 859	500	162	5 009	2.885	0.018	33.398	0.290
corel16k007	13 915	500	174	5 158	2.886	0.017	37.715	0.282
corel16k008	13 864	500	168	4 956	2.883	0.017	36.200	0.289
corel16k009	13 884	500	173	5 175	2.930	0.017	36.446	0.298
corel16k010	13 618	500	144	4 692	2.815	0.020	32.998	0.279
emotions	593	72	6	27	1.868	0.311	1.478	0.011
flags	194	19	7	54	3.392	0.485	2.255	0.061
mediamill	43 907	120	101	6 555	4.376	0.043	256.405	0.355
nus-wide-BoW	269 648	501	81	18 430	1.869	0.023	95.119	0.171
nus-wide-VLAD	269 648	129	81	18 430	1.869	0.023	95.119	0.171
scene	2 407	294	6	15	1.074	0.179	1.254	0.000

Table 3.4 Main characteristics of MLDs from genetics/proteomics classification case studies

Dataset	n	f	k	LSet	Card	Dens	MeanIR	SCUMBLE
genbase	662	1 186	27	32	1.252	0.046	37.315	0.029
yeast	2 417	103	14	198	4.237	0.303	7.197	0.104

this value is useful to know how sparse are the labelsets in the MLD. Higher *Dens* values will denote labelsets with more active labels than the lower ones.

As can be stated by glancing at the column with the *MeanIR* values, most MLDs show noteworthy imbalance levels. The mean proportion between the frequency of labels are higher to 1:100 in many cases, with some drastic occasions such as eurlex-sm, yahoo-health, or yahoo-business, whose *MeanIR* is above 500. There is only a handful of MLDs that could be considered as balanced, including 20ng, emotions, flags and scene. How this remarkably high imbalance levels can influence the learning methods, and how this difficulty has been faced in the literature, will be the main topics in Chap. 8.

The right-most column in these three tables shows the *SCUMBLE* value for each MLD. Attending to what was stated in [13], values well above 0.1 in this metric designate MLDs in which a significant proportion of rare labels jointly appear with very frequent ones, in the same instances. As can be seen, this is the case for many

Table 3.5 MLDs sorted according to their theoretical complexity score

Rank	Dataset	TCS	f	k	LSet
1	flags	8.879	19	7	54
2	emotions	9.364	72	6	27
3	scene	10.183	294	6	15
4	yeast	12.562	103	14	198
5	birds	13.395	260	19	133
6	genbase	13.840	1 186	27	32
7	20ng	13.917	1 006	20	55
8	slashdot	15.125	1 079	22	156
9	cal500	15.597	68	174	502
10	medical	15.629	1 449	45	94
11	tmc2007-500	16.372	500	22	1 172
12	langlog	16.946	1 004	75	304
13	ohsumed	17.090	1 002	23	1 147
14	stackex-coffee	17.446	1 763	123	174
15	enron	17.503	1 001	53	753
16	reuters	17.548	500	103	811
17	mediamill	18.191	120	101	6 555
18	imdb	18.653	1 001	28	4 503
19	stackex-chess	18.779	585	227	1 078
20	yahoo-business	18.848	21 924	30	233
21	nuswide-VLDA	19.076	129	81	18 430
22	yahoo-entertainment	19.238	32 001	21	337
23	stackex-chemistry	19.473	540	175	3 032
24	yahoo-health	19.609	30 605	32	335
25	corel16k010	19.638	500	144	4 692
26	yahoo-recreation	19.684	30 324	22	530
27	yahoo-reference	19.702	39 679	33	275
28	yahoo-arts	19.703	23 146	26	599
29	corel16k001	19.722	500	153	4 803
30	corel16k003	19.730	500	154	4 812
31	corel16k004	19.791	500	162	4 860
32	corel16k002	19.805	500	164	4 868
33	corel16k005	19.814	500	160	5 034
34	corel16k006	19.821	500	162	5 009
35	corel16k008	19.847	500	168	4 956
36	stackex-philosophy	19.905	842	233	2 249
37	corel16k009	19.919	500	173	5 175
38	corel16k007	19.922	500	174	5 158
39	yahoo-education	19.956	27 534	33	511

(continued)

Table 3.5 (continued)

Rank	Dataset	TCS	f	k	LSet
40	yahoo-computers	19.993	34 096	33	428
41	corel5k	20.200	499	374	3 175
42	yahoo-science	20.337	37 187	40	457
43	yahoo-social	20.418	52 350	39	361
44	nuswide-BoW	20.433	501	81	18 430
45	stackex-cs	20.532	635	274	4 749
46	bibtex	20.541	1 836	159	2 856
47	yahoo-society	20.623	31 802	27	1 054
48	tmc2007	21.093	49 060	22	1 341
49	stackex-cooking	21.111	577	400	6 386
50	eurlex-sm	21.646	5 000	201	2 504
51	eurlex-dc	21.925	5 000	412	1 615
52	rcv1subset4	22.082	47 229	101	816
53	rcv1subset3	22.223	47 236	101	939
54	rcv1subset5	22.230	47 235	101	946
55	rcv1subset2	22.239	47 236	101	954
56	rcv1subset1	22.313	47 236	101	1 028
57	delicious	22.773	500	983	15 806
58	bookmarks	22.848	2 150	208	18 716
59	eurlex-ev	26.519	5 000	3 993	16 467

of the MLDs shown in the previous tables. Some of them, such as the MLDs coming from the Corel image database, enron and delicious, stand out with *SCUMBLE* values as high as 0.532. This means that those MLDs would be specially harder for preprocessing and learning algorithms.

A metric which does not appear in the previous tables is *TCS*. Since it provides a score of the theoretical complexity of the MLDs, it is more useful to look at it after sorting the MLDs by the *TCS* column, instead of alphabetically. The result is shown in Table 3.5. Along the mentioned score, the number of features, labels, and labelsets are also presented. From this table, it is easy to deduct that some of the MLDs previously described as toy datasets present the lower theoretical complexity, with *TCS* values around 10. Unsurprisingly, the text MLDs appear as the most complex ones, due to their large sets of features and labels. Remember that *TCS* values are logarithmic, so a difference of only one unit implies one order of magnitude lower or higher.

Obviously, the MLDs could also be ordered by their *Card*, *MeanIR*, *SCUMBLE* or any other metric values, depending on which traits of the data the interest is on. It is easy to do so using the tools described in Chap. 9.

3.4 Multilabel Classification by Example

At this point, the source, nature, and main characteristics of a large set of MLDs have been already introduced. The characterization metrics have been applied over the MLDs, obtaining the measures shown in the previous tables. Before going into the study of the evaluation metrics, whose goal was to assess the predictive performance of a classifier, some predictions would be needed. This way we could get a glimpse of the values returned by these metrics. For this reason, this section is devoted to demonstrate how to conduct an example of multilabel classification job.

Even though the description of MLC algorithms is the main topic of further chapters, in the following subsection a specific algorithm is introduced to be able to complete the task. The outputs provided by this algorithm are then evaluated by means of different multilabel evaluation metrics.

3.4.1 The ML-kNN Algorithm

One of the simplest approaches to classification is that of kNN. Once a new data sample is given, a kNN classifier looks for its k-nearest neighbors. For doing so, the distance (in some f-dimensional space) between the features of the new sample and all instances in the dataset is computed. Once the closer instances have been gathered, their classes are used to predict the one for the new sample. Since kNN does not create any model, only when a new sample arrives the classifier does some work, it is usually known as a lazy [1] method. It is also frequently referred as instance-based learning [2].

ML-kNN [49] is an adaptation of the kNN method to the multilabel scenario. Unlike the classic kNN algorithm, ML-kNN is not so lazy. It starts by building a limited model that consists of two pieces of information:

- The a priori probabilities for each label. These are simply the number of times each label appears in the MLD divided by the total number of instances. A smoothing factor is applied to avoid multiplying by zero.
- The conditional probabilities for each label, computed as the proportion of instances with the considered label whose k-nearest neighbors, also have the same label.

These probabilities are independently computed for each label, facing the task as a collection of individual binary problems. Therefore, the potential dependencies among labels are fully dismissed by this algorithm.

After this limited training process, the classifier is able to predict the labels for new instances. When a new sample arrives, it goes through the following steps:

- First, the k-nearest neighbors of the given sample are obtained. By default the $L^2 - norm$ (Euclidean distance) is used to measure the similarity between the reference instance and the samples in the MLD.
- Then, the presence of each label in the neighbors is used as evidence to compute maximum a posteriori (MAP) probabilities from the conditional ones obtained before.
- Lastly, the labelset of the new sample is generated from the MAP probabilities. The probability itself is provided as a confidence level for each label, thus making possible to also generate a label ranking.

The reference MATLAB implementation for the ML-kNN algorithm is supplied by the author at his own Web site.[20] There is also available a Java implementation in MULAN. The latter has been used in order to conduct the experimentation described below.

3.4.2 Experimental Configuration and Results

Five MLDs have been chosen to run the ML-kNN algorithm. Two of them are from the text domain (`enron` and `stackex-cs`), two more from the multimedia field (`emotions` and `scene`), and the last one comes from the biology domain (`genbase`). Attending to their *TCS* measure `emotions` and `scene`, ranked at positions 2 and 3 in Table 3.5, would be the easier cases. A little harder would be `genbase` (6th), followed by `enron` (15th) and finally `stackex-cs` (45th) which, theoretically, would be the most difficult MLD in this collection.

The MLDs were partitioned following a 2×5 strategy. This means that there are two repetitions with 5 folds, and that for each run 80 % (4/5) of instances are used for training and 20 % (1/5) for testing. Therefore, a total of 10 runs are made for each MLD. Random sampling was used to select the instances in each fold. The full set of folds for the aforementioned five MLDs is available in the book repository [12].

From each run, a set of predictions are obtained from the classifier. These can be assessed using many performance evaluation metrics (they will be described in the next section), getting a set of values for each metric/fold. These values are then averaged, obtaining the mean indicators which are usually reported in most papers, sometimes along with their deviations. Table 3.6 shows all these values, whose interpretation will be further provided as the evaluation metrics are described.

[20] http://cse.seu.edu.cn/people/zhangml/Resources.htm#codes.

Table 3.6 Classification results produced by ML-kNN assessed with several evaluation metrics

	stackex-cs	emotions	enron	genbase	scene
Accuracy ↑	0.0540	0.5391	0.3156	0.9440	0.6667
AvgPrecision ↑	0.3009	0.7990	0.6280	0.9860	0.8648
Coverage ↓	77.9260	1.7715	13.2092	0.6110	0.4797
F-measure ↑	0.5900	0.7776	0.5898	0.9776	0.9593
HammingLoss ↓	0.0091	0.1940	0.0524	0.0048	0.0869
MacroF-measure ↑	0.1999	0.6225	0.4284	0.9357	0.7378
MacroPrecision ↑	0.5866	0.7279	0.5568	0.9795	0.8149
MacroRecall ↑	0.0169	0.5981	0.0808	0.6787	0.6808
MicroAUC ↑	0.8481	0.8565	0.9002	0.9893	0.9405
MicroF-measure ↑	0.1065	0.6652	0.4715	0.9458	0.7331
MicroPrecision ↑	0.6289	0.7217	0.6613	0.9934	0.8137
MicroRecall ↑	0.0583	0.6186	0.3671	0.9031	0.6673
OneError ↓	0.6571	0.2799	0.3070	0.0129	0.2269
Precision ↑	0.6157	0.7182	0.6616	0.9956	0.8252
RLoss ↓	0.1522	0.1608	0.0929	0.0072	0.0786
Recall ↑	0.0582	0.6184	0.3654	0.9454	0.6836
SubsetAccuracy ↑	0.0165	0.2968	0.0564	0.9132	0.6243

The arrow at the right of each metric name indicates whether lower values are better (↓) or the opposite (↑).

Disparate plot designs can be used to graphically represent those final values, being bar plots and line plots among the most frequent ones. When the interest is in comparing a group of cases, in this occasion how the classifier has performed with each MLD in accordance with several metrics, a radar chart (also known as spider plot) can be useful. In Figs. 3.1 and 3.2, this type of representation has been used to show the results produced by ML-kNN. Each vertex corresponds to a metric.[21] The points belonging to an MLD are connected so that a polygon is generated. The larger is the area of the polygon, the better is the result with a certain MLD.

Through the observation of these two plots, despite the details of each metric are not yet known, the following facts can be deducted:

- The performance with `emotions` and `scene`, with share a very similar *TCS* value, is very much alike.
- The results for he previous two MLDs are clearly better than for `enron`, which has a higher *TCS* score.
- The worst results are in general attributable to `stackex-cs`, the most complex MLD according to the *TCS* metric.

[21] The values of metrics such as *HammingLoss*, *OneError*, and *RankingLoss* have been complemented as the difference with respect to 1, aiming to preserve the principle of assigning a larger area to better values.

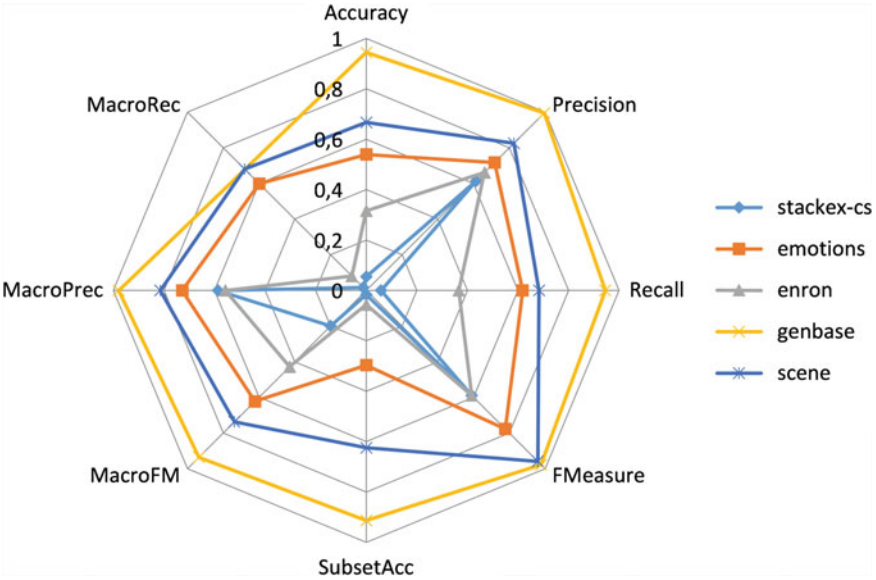

Fig. 3.1 Classification results produced by ML-kNN (part 1)

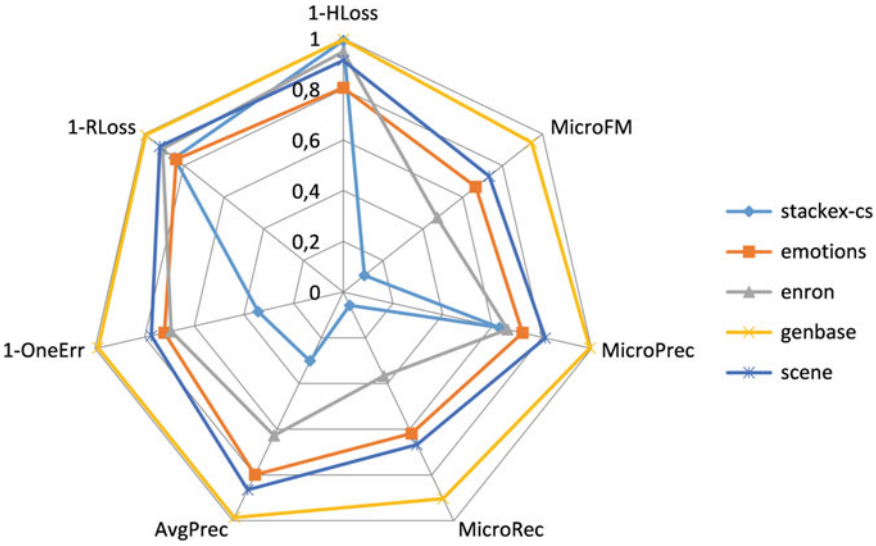

Fig. 3.2 Classification results produced by ML-kNN (part 2)

- The `genbase` results are not in line with previous appraisals, since it obtains the best results in all measures. This could be due to the existence of an attribute, named `protein`, containing a code that uniquely identifies each protein in the MLD. This feature would allow the classifier to easily locate the closest instances, producing a prediction that would be not so precise without that knowledge.

Overall, the previous plots seem to denote that the easier is the MLD, the better the classifier performs. This rule, as can be seen with the exception of `genbase`, can be broken depending on the MLDs specificities. Moreover, these results correspond to one classifier only, so they must be taken cautiously.

In order to complete the judgment of these results, it would be essential to gain an understanding of each individual evaluation metric. In the last section of this chapter, the details regarding how the performance of classifier can be assessed are provided, including additional discussion related to the values in Table 3.6.

3.5 Assessing Classifiers Performance

The output of any multilabel classifier consists of the labelset predicted for each test instance. When working in the traditional scenario, with only one class as output, the prediction only can be correct or wrong. A multilabel prediction, by contrast, can be fully correct, partially correct/wrong (at different degrees), or totally wrong. Applying the same metrics used in traditional classification is possible, but usually it is excessively strict. This is the reason for using specific evaluation metrics, able to take into consideration the cases between the two extremes.

Currently, more than twenty distinct performance metrics have been defined in the literature, and some of them quite specific aimed to hierarchical multilabel classification. All multilabel evaluation metrics can be grouped conforming to two criteria:

- **How the prediction is computed**: A measurement can be made by instance or by label, giving as a result two different groups of metrics:
  - **Example-based metrics**: These metrics [22, 23, 34] are calculated separately for each instance and then averaged dividing between the number of samples.
  - **Label-based metrics**: In contrast to the previous group, the label-based metrics [42] are computed independently for each label before they are averaged. For doing so, two different strategies [41] can be applied:
    Macro-averaging: The metric is calculated individually for each label and the result is averaged dividing by the number of labels (k).
    Micro-averaging: The counters of hits and misses for each label are firstly aggregated, and then the metric is computed only once.
- **How the result is provided**: The output produced by a multilabel classifier can be a binary bipartition of labels or a label ranking. Some of them provide both results.

- **Binary bipartition**: A binary bipartition is a vector of 0 s and 1 s indicating which of the labels belonging to the MLD are relevant to the processed sample. There are metrics that operate over these bipartitions, using the counters of true positives, true negatives, false positives, and false negatives.
- **Label ranking**: The output is a list of labels ranked according to some relevance measure. A binary bipartition can be obtained from a label ranking by applying a threshold, usually given by the classifier itself. However, there are performance metrics that work with raw rankings to compute the measurement, instead of using counters of right and wrong predictions.

In the two following subsections, the example-based and label-based metrics commonly used in the literature are described, providing their mathematical formulation. Where applicable, each metric description is completed with a discussion of the results produced by the experimentation with ML-kNN in the previous section.

3.5.1 Example-Based Metrics

These are the performance metrics which are firstly evaluated by each instance and then averaged according to the number of instances considered. Therefore, the same weight is assigned to every instance in the final score, whether they contain frequent or rare labels.

3.5.1.1 Hamming Loss

Hamming loss is probably the most commonly used performance metric in MLC. This is not surprising, as it is easy to calculate as can be seen in (3.11). The Δ operator returns the symmetric difference between Y_i, the real labelset of the ith instance, and Z_i, the predicted one. The $|r|$ operator counts the number of 1 s in this difference, in other words the number of miss predictions. The total number of mistakes in the n instances is aggregated and then normalized taking into account the number of labels and number of instances.

$$HammingLoss = \frac{1}{n}\frac{1}{k}\sum_{i=1}^{n}|Y_i \Delta Z_i| \tag{3.11}$$

Since the mistakes counter is divided by the number of labels, this metric will result in different assessments for the same amount of errors when used with MLDs having disparate labelset lengths. This is the main reason for the low *HammingLoss* value of the `stackex-cs` when compared to `emotions` or `scene`. The former has a large number of labels, while the others only have six. Therefore, this metric is

an indicator of committed errors by the classifier proportional to the labelset length. We can compare the results of `emotions` and `scene`, both have the same number of labels, and conclude that ML-kNN has performed better with the latter (lower value) than the former.

3.5.1.2 Accuracy

In the multilabel field, *Accuracy* is defined as (3.12) the proportion between the number of correctly predicted labels and the total number of active labels, in the both real labelset and the predicted one. The measure is computed by each instance and then averaged, as all example-based metrics.

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{3.12}$$

The *Accuracy* for `genbase` is very high, due to the reason previously explained. As shown in Fig. 3.1, the values for `emotions` and `scene` are very similar again, although with a slight advantage to the latter. The obtained *Accuracy* cannot be considered as good in the case of `enron`, and even less with the `stackex-cs` MLD. It must be remembered that this MLD had the highest *TCS* of the five case studies. Therefore, that it gets the worst classification performance was within the expected.

3.5.1.3 Precision, Recall, and F-Measure

Precision (3.13) is considered one of the more intuitive metrics to assess multilabel predictive performance. It is calculated as the proportion between the number of labels correctly predicted and the total number of predicted labels. Thus, it can be interpreted as the percentage of predicted labels which are truly relevant for the instance. This metric is usually used in conjunction with *Recall* (3.14) that returns the percentage of labels correctly predicted among all truly relevant labels. That is, the ratio of true labels is given as output by the classifier.

$$Precision = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Z_i|} \tag{3.13}$$

$$Recall = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i|} \tag{3.14}$$

The jointly use of *Precision* and *Recall* is so common in the information retrieval (IR) field that a metric combining them is defined. It is known as *F-measure* (3.15)

and computed as the harmonic mean of the previous ones. This way a weighted measure of how many relevant labels are predicted and how many of the predicted labels are relevant is obtained.

$$F\text{-}measure = 2 * \frac{Precision * Recall}{Precision + Recall}. \tag{3.15}$$

By observing the right side of Fig. 3.1, where *Precision*, *Recall*, and *F-measure* are depicted, that `scene` and `emotions` are once again very close can be stated, though `scene` results are a bit better. With `enron`, it can be seen that *Precision* has a higher value than *Recall*, a far greater fact in the case of `stackex-cs`. This means that for these MLDs a high proportion of the labels included in the prediction are relevant labels, but that there are many other true labels which are not predicted by the classifier. Looking at the *F-measure* values, the same correlation between the theoretical complexity (*TCS* value) of each MLD and classification performance assessment can be deducted.

3.5.1.4 Subset Accuracy

This is possibly the most strict evaluation metric. It is also known as classification accuracy and labelset accuracy, since full labelsets, the predicted and the real one, are compared for full equality as can be seen in (3.16). The larger is the labelset, the lower the likelihood that the classifier produces exactly the correct output. Therefore, for MLDs with large sets of labels that low *SubsectAccuracy* values are obtained is something usual.

$$SubsetAccuracy = \frac{1}{n}\sum_{i=1}^{n}[\![Y_i = Z_i]\!] \tag{3.16}$$

Apart from the atypical case of `genbase`, the *SubsectAccuracy* for the MLDs used in the previous experimentation reflects the problems the classifier had with each one of them. While `scene` values are not bad, the performance with `emotions` was far worse. As could be expected, due to their large sets of labels, `enron` and `stackex-cs` show the worst results.

3.5.1.5 Ranking-Based Metrics

All the example-based metrics described above work over binary partitions of labels, so they need a labelset as output from the classifier. By contrast, the explained here need a ranking of labels, so a confidence degree or belonging probability of each label is needed.

In the following equations, $rank(x_i, l)$ is defined as a function that for the x_i instance and the relevant label $l \in \mathcal{Y}$, whose position is known, returns l's confidence degree into the Z_i prediction returned by the classifier.

The *AvgPrecision* (Average precision) metric (3.17) determines for each label in an instance, the proportion of relevant labels that are ranked above it in the predicted ranking. The goal was to know how many positions have to be checked, in average, before a non-relevant label is found. Therefore, the larger is the *AvgPrecision* measure obtained, the better would be performing the classifier.

$$AveragePrecision = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{|\{y'|rank(x_i,y') \leq rank(x_i,y), y' \in Y_i\}|}{rank(x_i,y)} \tag{3.17}$$

The *Coverage* metric (3.18) counts the number of steps to going through the ranking provided by the classifier until all the relevant labels are found. The lower is the mean number of steps for the MLD, value returned by *Coverage*, the better is performing the classifier. As can be shown in (3.18), this measure is not normalized, so it is not upper bounded. As happens with other multilabel classification metrics, *Coverage* is influenced for the size of the set of labels in each MLD. The larger is this set, the higher usually is the mean number of steps to walk-through the ranking.

$$Coverage = \frac{1}{n}\sum_{i=1}^{n}\underset{y\in Y_i}{\text{argmax}}\langle rank(x_i,y)\rangle - 1 \tag{3.18}$$

As the previous one, *OneError* (3.19) is a performance metric to minimize. The expression which follows the summation returns 1 if the top-ranked label in the prediction given by the classifier does not belong to the real labelset. The number of miss predictions is accumulated and averaged. The result is the percentage of cases in which the most confident label for the classifier is a false positive.

$$OneError = \frac{1}{n}\sum_{i=1}^{n}[\![[\underset{y\in Z_i}{\text{argmax}}\langle rank(x_i,y)\rangle \notin Y_i]]\!]. \tag{3.19}$$

The *RLoss* (Ranking loss) metric takes all possible combinations of relevant and non-relevant labels for an instance and counts (3.20) how many times a non-relevant label is ranked above a relevant one in the classifier prediction. The counting is normalized dividing by the product of relevant and non-relevant labels in the instance and then averaged by the number of assessed instances. The lower is the *RLoss* measure, the better is performing the classifier.

$$RLoss = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i|.|\overline{Y_i}|}|y_a, y_b : rank(x_i,y_a) > rank(x_i,y_b), (y_a,y_b) \in Y_i \times \overline{Y_i}| \tag{3.20}$$

Observing the *AvgPrecision* in Table 3.6, it can be seen that with the exception of `stackex-cs`, ML-kNN performed quite well with the other four MLDs. Looking at the *Coverage* row, the values for `stackex-cs` and `enron` stand out. Since they have more labels, the number of steps to complete before getting all relevant labels is higher. The *OneError* values are quite similar for `emotions`, `scene` and `enron`, while for `stackex-cs` is much higher. This denotes that for the latter MLD the top-ranked label was not usually relevant. Lastly, considering the *RLoss* values a different scenario is observed. In this case, the worst results are obtained from `emotions`, though `stackex-cs` is very close. Although `emotions` only has six labels, there are a significant amount of predictions made by ML-kNN in which non-relevant labels are ranked above the relevant ones.[22]

3.5.2 Label-based Metrics

All the performance metrics enumerated in the previous section are evaluated individually for each instance, and then averaged dividing by the number of considered instances. Therefore, each data sample is given the same weight in the final result. On the contrary, label-based metrics can be computed by means of two different averaging strategies. These are usually known as *macro-averaging* and *microaveraging*.

Any of the metrics obtained from a binary partition of labels, such as *Precision*, *Recall* or *F-measure*, can be also computed using these strategies. For doing so, the generic formulas in (3.21) and (3.22) are used. *EvalMet* would be one of the metrics just mentioned. In this context, TP stands for *True Positives*, FP for *False Positives*, TN for *True Negatives*, and FN for *False Negatives*.

$$MacroMet = \frac{1}{k}\sum_{l\in\mathcal{L}} EvalMet(TP_l, FP_l, TN_l, FN_l) \tag{3.21}$$

$$MicroMet = EvalMet(\sum_{l\in\mathcal{L}} TP_l, \sum_{l\in\mathcal{L}} FP_l, \sum_{l\in\mathcal{L}} TN_l, \sum_{l\in\mathcal{L}} FN_l) \tag{3.22}$$

In the macro-averaging approach, the metric is evaluated once per label, using the accumulated counters for it, and then the mean is obtained dividing by the number of labels. This way the same weight is assigned to each label, whether it is very frequent or very rare.

[22] It must be taken into account that ML-kNN does not generate a real ranking of labels as prediction, but a binary partition. The ranking is generated from the posterior probabilities calculated for each label. With so few labels in `emotions`, it is possible to have many ties in these probabilities, so the positions in the ranking could be randomly determined in some cases.

On the contrary, the microaveraging strategy first adds the counters for all labels and then computes the metric only once. Since the predictions where rare labels appear are combined with that made for the most frequent ones, the former are usually diluted among the latter. Therefore, the contribution of each label to the final measure is not the same.

In addition to label-based metrics computed from binary partitions, those calculated from labels rankings are also available. The area under the ROC (*Receiver Operating Characteristic*) curve (AUC) can be computed according to the macro- and (3.23) and micro- (3.24) averaging approaches

$$MacroAUC = \frac{1}{k}\sum_{l\in\mathcal{L}} \frac{|\{x', x'' : rank(x', y_l) \geq rank(x'', y_l), (x', x'') \in X_l \times \overline{X_l}\}|}{|X_l|.|\overline{X_l}|},$$
$$X_l = \{x_i | y_l \in Y_i\}, \overline{X_l} = \{x_i | y_l \notin Y_i\} \tag{3.23}$$

$$MicroAUC = \frac{|\{x', x'', y', y'' : rank(x', y') \geq rank(x'', y''), (x', y') \in S^+, (x'', y'') \in S^-\}|}{|S^+|.|S^-|},$$
$$S^+ = \{(x_i, y) | y \in Y_i\}, S^- = \{(x_i, y) | y \notin Y_i\} \tag{3.24}$$

Analyzing the results in Table 3.6 corresponding to the label-based metrics, some interesting conclusions can be drawn. The *MacroF-measure* for `genbase` is clearly under the *MicroF-measure*. In the both cases, the same basic metric is used, *F-measure*, but with a different averaging strategy. From this observation, it can be deducted that one or more miss predicted rare labels exist in this MLD. By looking at Table 3.4 that `genbase` has a remarkable imbalance level can be confirmed, the existence of some rare labels is a fact. On the other hand, the *MicroAUC* values for all the MLDs are above the 0.8 level, which is the threshold from which usually the results are considered as good. The values for `enron`, `genbase`, and `scene` even surpass the 0.9 limit and can be regarded as excellent.

In addition to the groups of metrics already explained here, several more can be found, in general much more specific, in the specialized literature. For instance, there are metrics for evaluating the performance in hierarchical multilabel classification such as *Hierarchical loss* [8]. It is based on *Hamming loss*, but considering the level of the hierarchy where the miss predictions are made.

References

1. Aha, D.W. (ed.): Lazy Learning. Springer (1997)
2. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Mach. Learn. **6**(1), 37–66 (1991)
3. Alcala-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F.: KEEL multi-label dataset repository. http://sci2s.ugr.es/keel/multilabel.php
4. Atkinson, A.B.: On the measurement of inequality. J. Econ. Theory **2**(3), 244–263 (1970)
5. Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D.M., Jordan, M.I.: Matching words and pictures. J. Mach. Learn. Res. **3**, 1107–1135 (2003)
6. Boutell, M., Luo, J., Shen, X., Brown, C.: Learning multi-label scene classification. Pattern Recogn. **37**(9), 1757–1771 (2004)
7. Briggs, F., Lakshminarayanan, B., Neal, L., Fern, X.Z., Raich, R., Hadley, S.J.K., Hadley, A.S., Betts, M.G.: Acoustic classification of multiple simultaneous bird species: a multi-instance multi-label approach. J. Acoust. Soc. Am. **131**(6), 4640–4650 (2012)
8. Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Incremental algorithms for hierarchical classification. J. Mach. Learn. Res. **7**, 31–54 (2006)
9. Chang, C.C., Lin, C.J.: LIBSVM data: multi-label classification repository. http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html
10. Charte, F., Charte, D., Rivera, A.J., del Jesus, M.J., Herrera, F.: R Ultimate multilabel dataset repository. In: Proceedings of 11th International Conference on Hybrid Artificial Intelligent Systems, HAIS'16, vol. 9648, pp. 487–499. Springer (2016)
11. Charte, F., Rivera, A., del Jesus, M.J., Herrera, F.: LI-MLC: a label inference methodology for addressing high dimensionality in the label space for multilabel classification. IEEE Trans. Neural Netw. Learn. Syst. **25**(10), 1842–1854 (2014)
12. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: Multilabel classification. Problem analysis, metrics and techniques book repository. https://github.com/fcharte/SM-MLC
13. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: Concurrence among Imbalanced labels and its influence on multilabel resampling algorithms. In: Proceedings of 9th International Conference on Hybrid Artificial Intelligent Systems, HAIS'14, vol. 8480. Springer (2014)
14. 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)
15. 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: Proceedings of IEEE International Conference on Computer as a Tool, EUROCON'15, pp. 1–6. IEEE (2015)
16. Charte, F., Rivera, A.J., del Jesus, M.J., Herrera, F.: On the impact of dataset complexity and sampling strategy in multilabel classifiers performance. In: Proceedings of 11th International Conference on Hybrid Artificial Intelligent Systems, HAIS'16, vol. 9648, pp. 500–511. Springer (2016)
17. Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: NUS-WIDE: a real-world web image database from National University of Singapore. In: Proceedings of 8th ACM international Conference on Image and Video Retrieval, CIVR'09, pp. 48:1–48:9. ACM (2009)
18. Crammer, K., Dredze, M., Ganchev, K., Talukdar, P.P., Carroll, S.: Automatic code assignment to medical text. In: Proceedings of Workshop on Biological, Translational, and Clinical Language Processing, BioNLP'07, pp. 129–136. Association for Computational Linguistics (2007)
19. Diplaris, S., Tsoumakas, G., Mitkas, P., Vlahavas, I.: Protein classification with multiple algorithms. In: Proceedings of 10th Panhellenic Conference on Informatics, PCI'05, vol. 3746, pp. 448–456. Springer (2005)
20. Duygulu, P., Barnard, K., de Freitas, J., Forsyth, D.: Object recognition as machine translation: learning a Lexicon for a fixed image vocabulary. In: Proceedings of 7th European Conference on Computer Vision, ECCV'02, vol. 2353, pp. 97–112. Springer (2002)
21. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Advances in Neural Information Processing Systems, vol. 14, pp. 681–687. MIT Press (2001)

22. Ghamrawi, N., McCallum, A.: Collective multi-label classification. In: Proceedings of 14th ACM International Conference on Information and Knowledge Management, CIKM'05, pp. 195–200. ACM (2005)
23. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. Adv. Knowl. Discov. Data Min. **3056**, 22–30 (2004)
24. Gonçalves, E.C., Plastino, A., Freitas, A.A.: A genetic algorithm for optimizing the label ordering in multi-label classifier chains. In: Proceedings of 25th IEEE International Conference on Tools with Artificial Intelligence, ICTAI'13, pp. 469–476. IEEE (2013)
25. Joachims, T.: Text categorization with suport vector machines: learning with many relevant features. In: Proceedings of 10th European Conference on Machine Learning, ECML'98, pp. 137–142. Springer (1998)
26. Katakis, I., Tsoumakas, G., Vlahavas, I.: Multilabel text classification for automated tag suggestion. In: Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD'08, pp. 75–83 (2008)
27. Klimt, B., Yang, Y.: The enron corpus: a new dataset for email classification research. In: Proceedings of 15th European Conference on Machine Learning, ECML'04, pp. 217–226. Springer (2004)
28. Lang, K.: Newsweeder: learning to filter netnews. In: Proceedings of 12th International Conference on Machine Learning, ML'95, pp. 331–339 (1995)
29. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: RCV1: a new benchmark collection for text categorization research. J. Mach. Learn. Res. **5**, 361–397 (2004)
30. Mencia, E.L., Fürnkranz, J.: Efficient pairwise multilabel classification for large-scale problems in the legal domain. In: Proceedings of 11th European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD'08, pp. 50–65. Springer (2008)
31. Read, J.: Scalable multi-label classification. Ph.D. thesis, University of Waikato (2010)
32. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. **85**, 333–359 (2011)
33. Read, J., Reutemann, P.: MEKA multi-label dataset repository. http://sourceforge.net/projects/meka/files/Datasets/
34. Schapire, R.E., Singer, Y.: Boostexter: a boosting-based system for text categorization. Mach. Learn. **39**(2–3), 135–168 (2000)
35. Snoek, C.G.M., Worring, M., van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.M.: The challenge problem for automated detection of 101 semantic concepts in multimedia. In: Proceedings of 14th ACM International Conference on Multimedia, MULTIMEDIA'06, pp. 421–430 (2006)
36. Spyromitros-Xioufis, E., Papadopoulos, S., Kompatsiaris, I.Y., Tsoumakas, G., Vlahavas, I.: A comprehensive study over vlad and product quantization in large-scale image retrieval. IEEE Trans. Multimedia **16**(6), 1713–1728 (2014)
37. Srivastava, A.N., Zane-Ulman, B.: Discovering recurring anomalies in text reports regarding complex space systems. In: Aerospace Conference, pp. 3853–3862. IEEE (2005)
38. Tomás, J.T., Spolaôr, N., Cherman, E.A., Monard, M.C.: A framework to generate synthetic multi-label datasets. Electron. Notes Theoret. Comput. Sci. **302**, 155–176 (2014)
39. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. Int. J. Data Warehouse. Min. **3**(3), 1–13 (2007)
40. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of ECML/PKDD Workshop on Mining Multidimensional Data, MMD'08, pp. 30–44 (2008)
41. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook, pp. 667–685. Springer (2010)
42. Tsoumakas, G., Vlahavas, I.: Random k-Labelsets: an ensemble method for multilabel classification. In: Proceedings of 18th European Conference on Machine Learning, ECML'07, vol. 4701, pp. 406–417. Springer (2007)
43. Tsoumakas, G., Xioufis, E.S., Vilcek, J., Vlahavas, I.: MULAN multi-label dataset repository. http://mulan.sourceforge.net/datasets.html

44. Turnbull, D., Barrington, L., Torres, D., Lanckriet, G.: Semantic annotation and retrieval of music and sound effects. IEEE Trans. Audio Speech Lang. Process. **16**(2), 467–476 (2008)
45. Turner, M.D., Chakrabarti, C., Jones, T.B., Xu, J.F., Fox, P.T., Luger, G.F., Laird, A.R., Turner, J.A.: Automated annotation of functional imaging experiments via multi-label classification. Front. Neurosci. **7** (2013)
46. Tzanetakis, G., Cook, P.: Musical genre classification of audio signals. IEEE Trans. Speech Audio Process. **10**(5), 293–302 (2002)
47. Ueda, N., Saito, K.: Parametric mixture models for multi-labeled text. In: Proceedings of 15th Annual Conference on Neural Information Processing Systems, NIPS'02, pp. 721–728 (2002)
48. Wieczorkowska, A., Synak, P., Raś, Z.: Multi-label classification of emotions in music. In: Intelligent Information Processing and Web Mining, AISC, vol. 35, chap. 30, pp. 307–315 (2006)
49. Zhang, M., Zhou, Z.: ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn. **40**(7), 2038–2048 (2007)