Handling Tail-label problems in Multi-label Classification using in-situ Synthetic data generation

Pradyumna Kumar Sahoo
2018MSBDA016
Department of Data Science and Analytics
School of Mathematics,Statistics and Computational Science
Central University of Rajashthan

Agenda

- Abstract
- Introduction
- Major classification schemes
- Label Correlation & its orders
- Label-specific features and Class-dependent Labels
- LLSF-DL algorithm
- Label Imbalance – Types and measures
- Tail labels and Effective strategies
- MLSMOTE
- Expected Properties of an tail-label handling model
- Proposed Work – Algorithm, Analysis & Hyperparameters
- Experimental Setup – Evaluation Metrics, Results and Analyses
- Conclusions and Future Works

Abstract

- Labels with very few instances to learn from, called Tail-labels, is a critical problem faced in multilabel classification.
- Thus, to improve learning, oversampling algorithm MLSMOTE is employed on tail instances to generate new labeled instances in-situ for our base classifier LLSF-DL which is retrained on the modified dataset.
- This provides a standalone solution for learning of the classifier taking care of class imbalance along the way.

Introduction

Classification is a premiere field that occupies a large portion of research. Classification answers the question of assigning class(es) to the given input instance.

Instances can be anything from vectors to images to audio to text.

Major classification schemes:

- Binary Classification (BC)
- Multiclass Classification (MCC)
- Multilabel Classification (MLC)
- Multidimensional Classification (MDC)

Major classification schemes

Binary Classification (BC) :

An instance can belong one or the other class out of only two classes but not both.

Ex: One of the legendary movie, Rashomon can be classified as 20th century film or not.

Major classification schemes

Multiclass Classification (MCC) :

An instance can belong only one class out of many classes.

Ex: Rashomon can be classified into the group of ratings i.e. 7+,8+,9+ etc.

Major classification schemes

Multilabel Classification (MLC) :

An instance can belong one or more than one classes at the same time where each class is binary in nature.

Ex:Roshomon belongs to Crime, Drama, Mystery genre concurrently.

The movie may belong to more than one label but the labels are essentialy binary i.e. the movie either belongs to crime genre or it does not.

Major classification schemes

Multidimensional Classification (MDC) :

An instance can belong one or more than one classes at the same time but the classes need not be binary.

Ex: The movie’s labels represents whether the movie is a 20th century movie, who are the actors, which genres it belongs to, which month it has been released.

The month itself being a label consists of 12 different sublevels. This mixture of different objectives contributes towards the multidimensional aspect.

Label correlation

- The label-space grows exponentially with the number of labels.
- To cope with the challenge of exponential-sized output space, it is essential to facilitate the learning process by exploiting correlations (or dependency) among labels.

On the basis of capturing label correlation capacity, the algorithms can be grouped into three catagories :

- **First-order**: no correlation captured
- **Second-order**: pair-wise correaltion captured
- **Higher-order**: labelset-wise correlation captured

Label-specific features:

- In MLC, instances can have various labels associated with it. The features of those instances are obviously responsible for their labels.
- **But the question is which subset of features out of all are responsible for the individual labels when the instance has multiple ones??**
- Thus the features can be partitioned as per their role towards guiding instances towards a specific labels. These are called ***Label-specific features*.**
- **Correlated labels share the same label-specific features.**

Label-specific features - Example

Instances	Feature - 1	Feature - 2	Feature - 3	Label - 1	Label - 2	Label - 3
X1	0.5	0.3	1	1	1	0
X2	0.7	0.2	0	0	1	1

Suppose we know: (Feature-1,Feature-2) => Label-1

(Feature-2,Feature-3) => Label-2

Then (Feature-1,Feature-2) are L1-specific features while (Feature-2,Feature-3) are L2-specific features.

If L1 and L2 are found to be correlated, they share the same label-specific features, i.e. Feature-2 here.

Class-dependent labels:

In real world, labels are known to be not completely uncorrelated.Thus, a subset of labels can influence another subset as well as individual labels.

For example:

- Roshomon movie has genre labels – crime,drama,mystery.
- We usually find movies that are genre Crime associated with mystery.
- We seldom find crime and musical genre together.

The former example was related to pair-wise correlations (*second order*) while the later one depicts label correlation between a subset of labels with an individual one (*higher order*).

Those label or label subset that get influenced by other label or label subsets are called ***class-dependent labels.***

LLSF-DL Multilabel classifier

Learning Label-Specific Features and Class-Dependent Labels (LLSF-DL)[1] classifier is one of the SOTA algorithm for multi-label classification.
The objective function of LLSF-DL is designed in such a way that it learns label-specific data representation by considering both high order and second order label correlations class-dependent labels and it removes unnecessary label dependencies, preventing error propagation in learning.

$$\min_{W_x,W_y} \frac{1}{2}\|XW_x + YW_y - Y\|_F^2 + \frac{\alpha}{2}Tr(RW_x^T W_x) + \beta\|W_x\|_1 + \gamma\|W_y\|_1$$

- Weights of label-specific features for correlated labels are modelled pairwise by multiplying R (= 1- C,where C is the correlation matrix of Y) with $W_x^T W_x$. This product will be large for correlated labels only.

YW_Y is stacked with XW_x as guiding feature inside the main least square objective function.

- L1-regularisation of weight matrix W_y and W_x removes unnecessary label dependencies and uncorrelated label-specific features .The L1-norm makes the objective functions non-smooth, thus **Accelerated Proximal Gradient(APG)** method is used.
- And the optimised LLSF-DL algorithm return us weight matrix for X and Y. We then use the weight matrices from LLSF-DL to get our final predicted label matrix.

Label Imbalance

The labels do not have uniform representation.

Reasons:

- Learning from multi-label data is difficult because of the enormous output space, which exponentially scales as the number of class labels increases.

- Thus, distributions of class labels in the data need not be uniform, producing imbalanced datasets. Naturally, the class labels do not have uniform representation.

Types of Imbalance

Two types mainly:

- **Inter-label:**

For each label, when instances belonging to a certain label outnumbers other labels,

EX - Suppose we have 100 movies with genres i.e. g1,g2,g3 and num(g1>g2>g3) given. We find in the labelset matrix, g1 based movies are more than any other genre movies.

- **Intra-label:**

When discriminating instances are very few as compared to other instances for each label

Ex- Suppose g2-based movies are counted up, we find g2-based content is rare while negative instances are common.

Imbalance Measures

1. Imbalance Ratio per Label (IRLbl):

It is the ratio between the majority label and the label y. Most frequent label has value 1 while other labels have values >1. Higher imbalance is reflected by a higher IRLbl value.

|D| is total number of instances in multilabel dataset D, Y_i being i^{th} instance in Label matrix Y, y being any label

$$IRLbl(y) = \frac{\operatorname{argmax}_{y' = Y_1}^{Y_{|Y|}} \left(\sum_{i=1}^{|D|} h(y', Y_i)\right)}{\sum_{i=1}^{|D|} h(y, Y_i)}, \quad h(y, Y_i) = \begin{cases} 1, & y \in Y_i \\ 0, & y \notin Y_i. \end{cases}$$

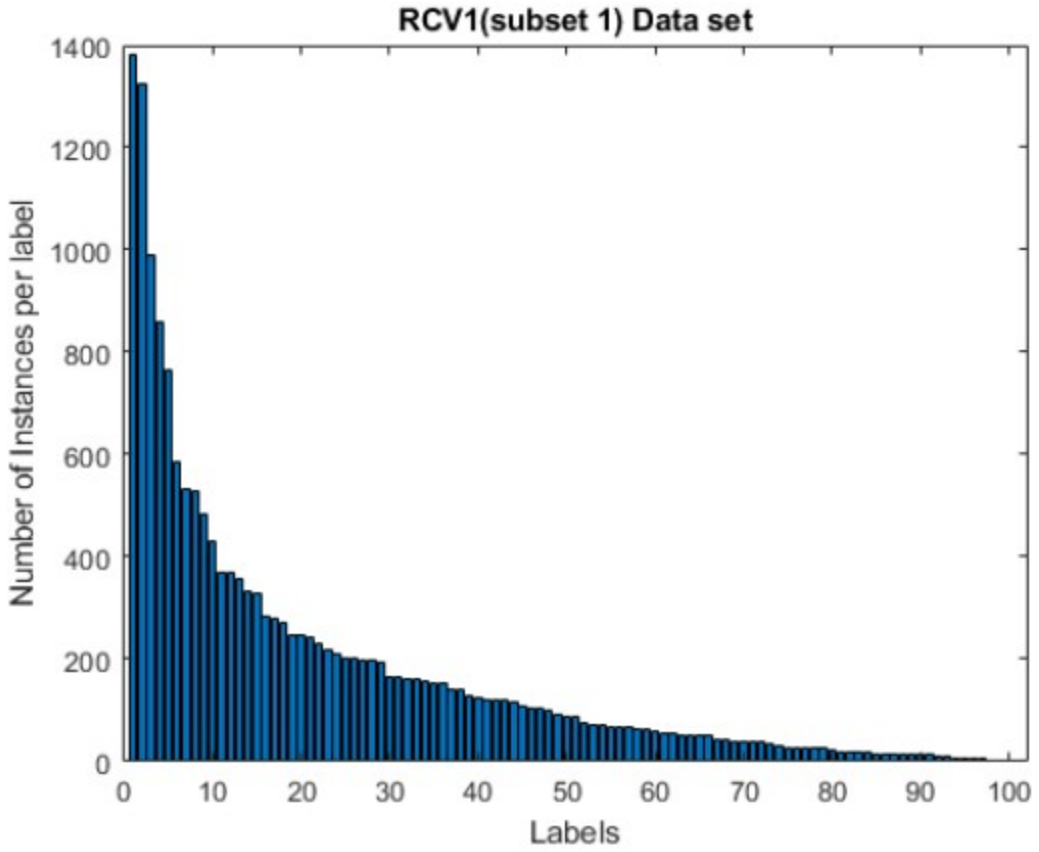
2. Mean Imbalance Ratio (MeanIR):

Represents the average imbalance in D.

$$MeanIR = \frac{1}{|Y|} \sum_{y = Y_1}^{Y_{|Y|}} (IRLbl(y)).$$

Tail-labels

- The intra-label imbalanced labels are famously called Tail-labels.
- The name has been stuck with it because they lie towards the end of the label frequency plot.
- Any label with IRLbl > MeanIR, are termed tail labels.



Approaches for Handling Tail-labels[3]:

Three most common approaches to deal with tail-labels are:

- **Re-sampling:**

- Data re-sampling re-balances class distributions by either deleting instances of the most frequent label(s) known as undersampling or adding new instances of the least frequent one called oversampling. It includes synthetic data generation too.This approach is independent of the classification algorithms.

- **Cost sensitive classification :**

- Tweaks the objective function of the constrained optimisation to provide equal footing for tail-labels.

- **Algorithmic adaptations:**

- Algorithmic adaptions includes transformative methods that converts the task at hand to a multiclass task as well as hybrid methods that mostly consist of ensembles etc.

Multi-label SMOTE[4]

Among synthetic data generation techniques, Synthetic Minority Oversampling Technique (SMOTE) stands apart from rest of the methods as SOTA in the field. One of the extension for multilabel samples is MLSMOTE which stands relevant in our case.

MLSMOTE’s working can be broke down into following parts:

- It separates tail-instances based upon imbalance ratio.
- And then generates new synthetic sample in the feature space over the lines joining a reference point and its neighbours.
- The labelsets are then assigned using majority voting among the reference point and its neighbours.

Expected Properties of an tail-label handling model

An optimal algorithm should have the follwing capabilities for handling tail labels:

- The algorithm can model higher order labelset correlations as well as pairwise labels correlation for better labelset learning.
- The algorithm should avoid error propagation meanwhile discard unnecessary label depdendencies.
- For efficient learning, only label-specific features should be used for responsible labels prediction.
- The algorithm should also capture imbalance among labels.
- Also, the algorithm should provide better generalisation over base models for both tail labels as well as for all labels.
- The algorithm should be scalable in terms of handling extremely large labelspace (Extreme Multilabel Classification) and time-efficient.

Our Algorithm

Input: Training Matrix $X \in \mathbb{R}^{m*d}$,Label Matrix $Y \in \{0,1\}^{m*l}$,Test Matrix $X_t \in \mathbb{R}^{n*d}$, Weight parameters α, β, γ, ρ, Threshold τ, tuning parameter κ, drop_majority $\in \{true, false\}$, no of nearest neighbours a

Output: Predicted Label Matrix $\hat{Y}_t \in \{0,1\}^{n*l}$

Find $MeanIR$
$Y_{tail\_labels} \leftarrow$ all tail labels for which $IRLbl \geq MeanIR$.
repeat
    if $drop\_majority == True$ **then**
        $Y \leftarrow Y_{tail\_label}$
    end if
    $X_{new}, Y_{new} \leftarrow MLSMOTE([X, Y], a)$
    $X \leftarrow X \cup X_{new}$
    $Y \leftarrow Y \cup Y_{new}$
    $\hat{Y} \leftarrow LLSF - DL(X, Y, X_t, \alpha, \beta, \gamma, \rho, \tau, \kappa)$
    return $\hat{Y}$
until Stopping condition reached

Algorithm specifications

- Our algorithm uses the feature selection strategy of LLSF_DL for assigning labels specific features to prediction of the associated labels.
- The L1-regularisation introduced in the former algorithm zeroes out weights of unnecessary labels while predicton for uncorrelated labels, hence, prevents error propagation.
- Global as well as local correlations among features a well as labels are captured explicitly using correlation terms in the objective function directly.
- Our algorithm has overall time- complexity $\mathbf{O(max(d^2l, l^3))}$. Here, *l* stands for number of labels, d stands for number of features.

Hyperparameter Selection:

We have used the default hyperparameter range as suggested by the authors in our experiment.

- For MLSMOTE:
- number of nearest neighbours to be selected for the reference point is kept as it is **5**.

- For LLSF_DL:

- α, β, γ have been searched in range $\{4^{-5}, 4^{-4}, 4^{-3}, ..., 4^{5}\}$
- ρ in {0.1,1,10}
- threshold is kept 0.5
- tuning parameter κ as 3.

- The hyperparameters of our model could not be tuned using Grid-search on our commodity hardware.
- We believe that even better performance can be achieved with carefully tuned hyperparameter set.

Dataset Information[5]

For validation of our algorithm, we have used standard MLDs. Datasets can be downloaded from mulan..

Data sets	Instances	Features	Domain	Labels	Tail Labels
Emotions	593	72	music	6	4
Genbase	662	1186	biology	27	8
Rcv1	6000	944	text	101	24
Recreation	5000	606	text(web)	22	7

Evaluation metrics:

Most of the MLC metrics are extension of their binary counterparts.These metrics fall into two categories:

1. ***Example-based:*** assess the difference between the true and predicted labelsets.Example-based metrics work by evaluating the learning system's performance on each test example separately, and then returning the mean value across the test set. Let q be number of labels

Hamming loss: Measures the average of misclassification frequency for instance-label pairs.Z_i denotes prediction, delta denotes symmetric difference.

$$\text{Hamming loss} = \frac{1}{|Y|}\sum_{i=1}^{|Y|}\frac{1}{l}[y'_i \triangle Y_i]$$

Subset Accuracy: Reports the percentage of predicted label sets that contain an error. Both incorrect and partially correct predictions are not considered as exact match is required.

$$\text{Subset Accuracy} = \frac{1}{|Y|}\sum_{n=1}^{|Y|}1(y'_n = y_n]$$

Evaluation metrics: (Continued)

2. *Label-based:* Learning system's performance is measured for each label separately but we can take average in macro/micro across all labels.Let B be a binary evaluation measure (e.g., precision,recall, accuracy, or F1-score), q be number of labels, tp – true positives, tn – true negatives, fp – false positive, fn – false negatives.

Macro approach: One metric is computed for each label, and the values are averaged over all labels. Macro averaged scores assign equal weightage to every label regardless of its frequency and is more influenced by the performance on rare categories like tail-labels.

Micro approach: It looks at predictions for all instances together (aggregating all the contingency tables) and then takes the average across all labels.Micro averaged scores assign equal weightage to every instance (per-example averaging) and tend to be dominated by the performance in most frequent labelsets.

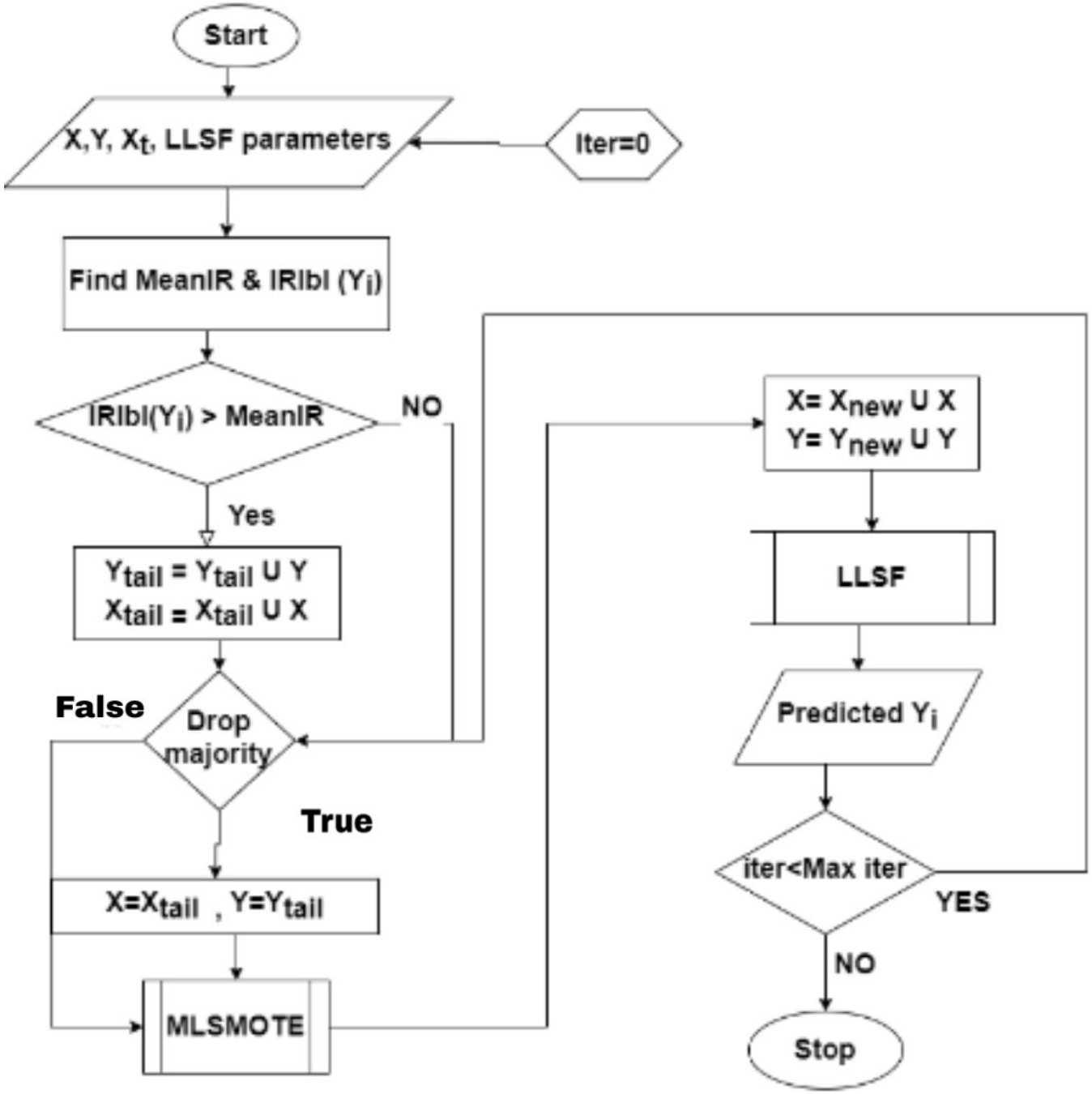
$$B_{macro} = \frac{1}{q}\sum_{i=1}^{q} B(tp_i, fp_i, tn_i, fn_i)$$

$$B_{micro} = B(\sum_{i=1}^{q} tp_i, \sum_{i=1}^{q} fp_i, \sum_{i=1}^{q} tn_i, \sum_{i=1}^{q} fn_i,)$$

Experimental Setup

- We wanted to know whether there are any explicit improvements in the classification of tail labels.
- Hence, we divided our experiment into two parts,namely, for overall data and for tail instances.
- In the former approach, we have used the given whole Y matrix while in the later part, we have trained the classifier only using the tail labels directly.
- The hamming loss and subset accuracy focus on instance-wise prediction performance while the label-based macro f1-measure and label-based macro f1-measure focus on label wise prediction performance.
- Out of all these metrics, label based macro F1- measure is very sensitive towards tail-labels.
- Hence, we have chosen this metric as our optimal metric while all others act as satisfying metrics.

Flowchart


Start
X,Y, Xt, LLSF parameters
Iter=0
Find MeanIR & IRlbl (Yi)
IRlbl(Yi) > MeanIR
NO
Yes
Ytail = Ytail U Y
Xtail = Xtail U X
False
Drop majority
True
X=Xtail , Y=Ytail
MLSMOTE
X= Xnew U X
Y= Ynew U Y
LLSF
Predicted Yi
iter<Max iter
YES
NO
Stop

Results:

HAMMING LOSS						
Datasets	For overall data			For minority instances		
	Iteration - 1	Iteration - 2	Iteration - 3	Iteration - 1	Iteration - 2	Iteration - 3
Emotions	0.2215	0.2313	0.2473	0.1981	0.2066	**0.2175**
Genbase	0.0024	0.0031	0.0032	0.0029	0.0032	0.0032
RCV1	0.0454	0.0507	0.0606	0.0028	0.0042	**0.0073**
Recreation	0.0560	0.0568	**0.0596**	0.0127	0.0164	**0.0185**

SUBSET ACCURACY						
Dataset	For overall data			For minority instances		
	Iteration - 1	Iteration - 2	Iteration - 3	Iteration - 1	Iteration - 2	Iteration - 3
Emotions	0.2514	0.2446	0.2108	0.3931	**0.4049**	0.3930
Genbase	0.9515	0.9380	0.9425	0.9788	0.9758	0.9758
RCV1	0.0220	0.0168	0.0093	0.9415	0.9218	0.8733
Recreation	0.3010	0.2756	0.2168	0.9226	0.9048	0.8916

Results (continued) :

LABEL-BASED MACRO F1-MEASURE

Dataset	For overall data			For minority instance		
	Iteration - 1	Iteration - 2	Iteration - 3	Iteration - 1	Iteration - 2	Iteration - 3
Emotions	0.6548	0.6496	0.6241	0.5686	0.6082	0.6353
Genbase	0.9744	0.9673	0.966	0.2967	0.3017	0.3017
RCV1	0.3751	0.352	0.2871	0.0026	0.0161	0.0125
Recreation	0.2932	0.3033	0.2413	0.121	0.1462	0.1256

LABEL-BASED MICRO F1-MEASURE

Dataset	For overall data			For minority instances		
	Iteration - 1	Iteration - 2	Iteration - 3	Iteration - 1	Iteration - 2	Iteration - 3
Emotions	0.7937	0.7746	0.7445	0.577	0.6075	0.6302
Genbase	0.9872	0.9872	0.986	0.6514	0.6533	0.6533
RCV1	0.4918	0.4655	0.3912	0.0245	0.0503	0.0582
Recreation	0.4557	0.4243	0.3422	0.1556	0.1552	0.1443

Analysis

1) The algorithm have performed marginally well in modelling tail labels for smaller datasets.

2) We have the best results from Emotions data set.

3) Increment in the label based macro f1-measure for all datasets shows that we are on the right track.

4) However, for overall dataset, our algorithm performs poorly in our optimal metric.

5) With each iteration, there is a decrease in performance.

Analysis (Continued)

Main reasons for poor performance are:

1. The base classifier LLSF_DL needs to be modified further to improve its prediction capacities.

2. Poor hyper-parameter tuning for overall model. Grid-search is unfeasible for an algorithm with time complexity $O(\max(d^2l, l^3))$ over commodity hardware.

3. The generated data may not be of high quality. It is because MLSMOTE generates new instances on the basis of features and after that the labels are assigned simply using majority voting.Thus, samples are generated based upon the marginal distribution $P(X|D)$. But these does not model inherent distribution of data which is a joint distribution $P(X,Y|D)$.

4. There is no check and balance mechanism inbetween data generation part and classification. Hence their assembly is performing poorly. To site a case, the MLSMOTE algorithm keeps generating new samples without any check implicitly effected by the classifier, hence the sample distribution of the overall dataset keeps getting skewed.This further introduces imbalances.

Conclusion

1. Though our algorithm is capable of handling tail labels, the experiments show that more improvements are needed.
2. The algorithm has increased label based macro F1- measure for all datasets. It is very effectively dealing with smaller datasets.
3. Further, each component of algorithm - data generation and classification are to be worked upon in order to avoid the previously stated drawbacks. At the best, the SOTA classifier -LLSF_DL alone is merely 38% - 40% accurate (rcv1,recreation) in predicting the labels. Hence, our prediction accuracy also suffers.
4. Similarly, the data generator should model the joint distribution of X and Y. Further, hyperparameter tuning should be carried on specialised hardwares. We believe these pointers for future works can improve our models' predictions.

Future Work:

1. Extending the research,the future course of action are being figured out.
2. Application of Generative Adverserial Networks to generate samples are showing great results in better quality data generation.
3. Similarly, recent papers on multilabel classifiers have performed multilabel classification by using non-sparse label specific features, give even better predictions.
4. Both the aspects are being evaluated before incorporating them.

References:

[1] J. Huang, G. Li, Q. Huang, and X. Wu. Learning label-specific features and class-dependent labels for multi-label classification. IEEE Transactions on Knowledge and Data Engineering, 28(12):3309–3323, 2016.

[2] Paula Branco, Luís Torgo, and Rita Ribeiro. A survey of predictive modeling on imbalanced domains. ACM Computing Surveys, 49, 08 2016.

[3] F. Charte, Antonio Rivera Rivas, María José Del Jesus, and Francisco Herrera.Addressing imbalance in multilabel classification: Measures and random resampling algorithms. Neurocomputing, 163, 04 2015.

[4] F. Charte, Antonio Rivera Rivas, María José Del Jesus, and Francisco Herrera.MLsmote: Approaching imbalanced multilabel learning through synthetic instance generation. Knowledge-Based Systems, pages –, 09 2015.

[5] http://mulan.sourceforge.net/datasets-mlc.html

THANK YOU

LLSF-DL Algorithm

Input: Training Matrix $X \in \mathbb{R}^{m*d}$, Label Matrix $Y \in \{0,1\}^{m*l}$, Weight parameters α, β, γ, ρ, Test Matrix $X_t \in \mathbb{R}^{n*d}$, Threshold τ, tuning parameter κ.

Output: Predicted Label Matrix $Y_t \in \{0,1\}^{n*l}$

Initialization:

$b_0 \leftarrow 1$, $b_1 \leftarrow b_0$, $t \leftarrow 1$

$W_x^{(0)} \leftarrow (X^T X + \rho I)^{-1} X^T Y$

$W_x^{(1)} \leftarrow W_x^{(0)}$

$W_y^{(0)} \leftarrow (Y^T Y + \rho I)^{-1} Y^T Y$

$W_y^{(1)} \leftarrow W_y^{(0)}$

$R \leftarrow \frac{Y^T Y}{||Y||_2^2}$ ▷ Cosine similarity

$W_f \leftarrow [3(||X^T X||_2^2 + ||X^T Y||_2^2 + ||\alpha R||_2^2) + 2(||Y^T Y||_2^2)]^{1/2}$

LLSF-DL Algorithm(Continued)

repeat

▷ Gradient descent

$W_x^{(t)} \leftarrow W_x^{(t)} + \frac{b_{t-1}-1}{b_t}(W_x^{(t)} - W_x^{(t-1)})$

$W_y^{(t)} \leftarrow W_y^{(t)} + \frac{b_{t-1}-1}{b_t}(W_y^{(t)} - W_y^{(t-1)})$

▷ Regularization

$G_x^{(t)} \leftarrow W_x^{(t)} - \frac{1}{L_f}[X^T X W_x + X^T Y W_y - X^T Y + \alpha W_x R]$

$G_y^{(t)} \leftarrow W_y^{(t)} - \frac{1}{L_f}[Y^T Y W_y + Y^T X W_x - Y^T Y]$

$W_x^{(t+1)} \leftarrow soft\_threshold(G_x^{(t)}, (\frac{\rho}{L_f}))$

$W_y^{(t+1)} \leftarrow soft\_threshold(G_y^{(t)}, (\frac{\rho}{L_f}))$

$b_{t+1} \leftarrow 0.5(1 + [4b_t^2 + 1]^{(1/2)})$

$t \leftarrow t + 1$

until Stopping criteria not reached

$\hat{Y}_t \leftarrow LLSF[X, Y, Xt, \alpha, \beta, \rho, \tau$

for $i \leftarrow 1, \kappa$ **do**

$S_t \leftarrow X_t W_x + \hat{Y}_t W_y$

$\hat{Y}_t \leftarrow sign(S_t - \tau)$

end for

$Y_t \leftarrow \hat{Y}_t$

MLSMOTE Algorithm

Input: Feature matrix X, Label matrix Y, Nearest neighbors K

$Z \leftarrow$ Concatenate the X and Y ▷ Data set for oversampling
$L \leftarrow$labels In Dataset(Z)
Calculate $MeanIR$ for Z by Eq 2.2
for each $Labels$ in L **do**
    $IRLbl_{label} \leftarrow$ calculate Imbalance ratio for each Label(D, label)
    if $IRLbl_{label} > MeanIR$ **then**
        ▷ Minority labels samples
        $Bag \leftarrow$ get All Instances Of Label(labels)
        for each $selected$ in Bag **do**
            $dist \leftarrow$ calculate Distance($selected$, Bag)
            sort Smaller To Largest($dist$)
            ▷ Neighbor set selection
            $Neighbors \leftarrow$ get Head Items($dist, k$)
            $refNeigh \leftarrow$ get Random Neighbor($Neighbors$)
            ▷ Generation of feature and label set
            $Synthetic\_sample \leftarrow$ **SYNTHETIC**($selected, refNeigh, neighbors$)
            $Z \leftarrow Z + Synthetic\_sample$
        end for
    end if
end for

MLSMOTE Algorithm (Continued)

21: **function SYNTHETIC**$(selected, refNeigh, neighbors)$
22: $Synthetic\_sample \leftarrow new\_Sample$ ▷ New empty instance
23: ▷ Assignment of feature set
24: **for each** $feature$ in$Synthetic_sample$ **do**
25: $diff \leftarrow refNeigh.feat - sample.feat$
26: $offset \leftarrow diff * randonInterval(0, 1)$
27: $value \leftarrow sample.feat + offset$
28: **end for**
29: ▷ Assignment of label set
30: $labe\_lCounts \leftarrow counts(sample.labels)$
31: $label\_Counts+ \leftarrow counts(neighbors.labels)$
32: $Labels \leftarrow labe\_lCounts > (k + 1)/2$
33: $Synthetic\_sample.Labels \leftarrow Labels$
34: **return** $Synthetic\_sample$
35: **end function**