Hosenie, Z, Lyon, R, Stappers, B, Mootoovaloo, A and McBride, V

Imbalance learning for variable star classification

Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **493,** 6050–6059 (2020)
Advance Access publication 2020 March 13

doi:10.1093/mnras/staa642

Imbalance learning for variable star classification

Zafiirah Hosenie,[1★] Robert Lyon,[1,2] Benjamin Stappers,[1]
Arrykrishna Mootoovaloo[3] and Vanessa McBride[4,5,6]

[1] *Jodrell Bank Centre for Astrophysics, Department of Physics and Astronomy, The University of Manchester, Manchester M13 9PL, UK*
[2] *Department of Computer Science, Edge Hill University, Ormskirk Lancashire L39 4QP, UK*
[3] *Imperial Centre for Inference and Cosmology (ICIC), Imperial College, Blackett Laboratory, Prince Consort Road, London SW7 2AZ, UK*
[4] *Department of Astronomy, University of Cape Town, Private Bag X3, Rondebosch 7701, South Africa*
[5] *South African Astronomical Observatory, PO Box 9, Observatory 7935, South Africa*
[6] *IAU Office of Astronomy for Development, Cape Town, Observatory 7935, South Africa*

Accepted 2020 February 27. Received 2020 February 27; in original form 2019 October 23

ABSTRACT

The accurate automated classification of variable stars into their respective subtypes is difficult. Machine learning–based solutions often fall foul of the imbalanced learning problem, which causes poor generalization performance in practice, especially on rare variable star subtypes. In previous work, we attempted to overcome such deficiencies via the development of a hierarchical machine learning classifier. This ‘algorithm-level’ approach to tackling imbalance yielded promising results on Catalina Real-Time Survey (CRTS) data, outperforming the binary and multiclass classification schemes previously applied in this area. In this work, we attempt to further improve hierarchical classification performance by applying ‘data-level’ approaches to directly augment the training data so that they better describe underrepresented classes. We apply and report results for three data augmentation methods in particular: *R*andomly *A*ugmented *S*ampled *L*ight curves from magnitude *E*rror (`RASLE`), augmenting light curves with Gaussian Process modelling (`GpFit`) and the Synthetic Minority Oversampling Technique (`SMOTE`). When combining the ‘algorithm-level’ (i.e. the hierarchical scheme) together with the ‘data-level’ approach, we further improve variable star classification accuracy by 1–4 per cent. We found that a higher classification rate is obtained when using `GpFit` in the hierarchical model. Further improvement of the metric scores requires a better standard set of correctly identified variable stars, and perhaps enhanced features are needed.

Key words: methods: data analysis – stars: variables: general.

1 INTRODUCTION

Astronomy is now in an era dominated by an explosion of big data, produced with current and future surveys, such as OGLE (Udalski et al. 2008; Udalski, Szymański & Szymański 2015), CRTS (Drake et al. 2017), and Kepler (Koch et al. 2010) among others; thus, relying solely on visual inspection for classification is becoming impractical. Therefore, automatic classification pipelines are required to categorize an unprecedented amount of variable star light curves into known or unknown classes for various astrophysical applications. Accordingly, Machine Learning (ML) has heavily been studied to solve classification problems, for instance, uncovering aberrant phenomena encountered in observations, also known as unsupervised anomaly detection (Chen et al. 2018; Zong et al. 2018) and automatic classification of variable stars (Kim & Bailer-Jones 2016; Benavente, Protopapas & Pichara 2017; Mahabal et al. 2017; Narayan et al. 2018; Pashchenko, Sokolovsky & Gavras 2018; Tsang & Schultz 2019; Zorich, Pichara & Protopapas 2020).

However, a major issue that impedes the successful automated classification of astronomical data is known as the imbalanced learning problem. This occurs when we wish to organize data into distinct groups known as ‘classes’, using examples to guide a process known as ‘classification’. When there is a large distributional difference between the number of examples belonging to each class, minority, and majority classes form. When the imbalance between the minority and majority classes is large, problems can arise when attempting to build standard machine learning classification algorithms, ultimately resulting in poor categorization performance. This happens as such algorithms are usually optimized to achieve maximum accuracy. However, this is trivially achievable in imbalanced data sets by always assigning the majority class label when making predictions. This leads to biased classifiers that obtain high predictive accuracy for majority class, but poor predictive accuracy for minority classes, which are more often than not, the focus of our interest.

★ E-mail: zafiirah.hosenie@gmail.com

Imbalanced learning problems occur in many domains, for instance in fraudulent phone call identification (few calls are fraudulent, Fawcett & Provost 1996), or text classification (in cases where there is either more positive or more negative sentiments). In astronomy, this issue becomes acute, given that data sets must often be searched for rare or unusual phenomena that may not be accurately defined in advance. This problem impacts the classification of variable stars in particular, as some types of variable star are uncommon, making it difficult to build systems to be able to recognize them. In astronomy, several works have tried to address the problem of class imbalance to date (Hoyle et al. 2015; Lochner et al. 2016; Narayan et al. 2018; Revsbech, Trotta & van Dyk 2018; Agarwal et al. 2019).

There are two approaches for dealing with class imbalance problems (He & Garcia 2008). The first are generally known as ‘algorithm level’ approaches. These seek to modify classification algorithms directly to better accommodate imbalanced class distributions. This can involve, for example, adapting the learning function at the heart of the algorithm to favour metrics other than accuracy during training and also applying hyperparameter tuning while training the algorithm (see Section 4.4). Algorithm level approaches make an implicit assumption – that is, the data are sufficiently descriptive and statistically characteristic of the classes under consideration, and changes to the algorithm alone will enable this data to yield good classification performance.

Alternatively, ‘data level’ approaches seek to modify the data given to a classification algorithm, with the aim of improving classification performance. Data level approaches can be as simple as balancing training data artificially via an appropriate sampling method, or as complex as generating artificial samples to balance the training set. Data level approaches assume that classification algorithms will be capable of separating the classes under consideration, given appropriate training data. Hybrid approaches mix the two techniques when faced with difficult problems. For instance, in some cases modifying an algorithm will not produce the improvement expected, if the classification problem at hand exhibits excessive class overlap, disjuncts, or is affected by small sample sizes (i.e. some classes are genuinely rare). Whilst in some cases trying to balance training sets will not work if the information content of the training samples is too low to allow a classifier to delineate effective class boundaries.

In previous work, we attempted to develop a variable star classifier together with various techniques of feature selection and feature importance, and ran into the imbalanced learning problem. To overcome this, we attempted to modify the algorithms used for classification, and ultimately proposed a successful hierarchical classification system. We compared the hierarchical system (using seven features) with the UPSILON package (Kim & Bailer-Jones 2016) (using 16 features). Whilst hierarchical system was effective, recall on minority classes could be stubbornly low relative to majority classes. In other domains, such problems are overcome by balancing the training distribution directly. This approach implies the minority class is sufficiently described in the training data to solve the imbalance, and further that the classifier used is sensitive to the class size. We believe this to be the case, thus we proceed similarly. We present a hybrid approach to overcoming imbalances, which represents a principled and pragmatic approach to this problem. Thus in this work, we improve the Hosenie et al. (2019, hereafter H19) classification scheme by adding a sufficient amount of data, such that each class has an equal amount of training examples. This can be achieved by simulating more data or gathering more real data (which is often difficult).

Table 1. Sample size of classes in CRTS data. The class distribution is extremely imbalanced, such as Ecl are overrepresented.

Types of variable stars	N_{Objects}
RRab (fundamental mode)	4325
RRc (first overtone mode)	3752
RRd (multimode)	502
Blazhko (long-term modulation)	171
Contact and semidetached binary: Ecl	18803
Detached binary: EA	4509
Rotational: Rot	3636
Long period variable: LPV	1286
δ-Scuti	147
Anomalous Cepheids: ACEP	153
Type-II Cepheids: Cep-II	153

Balancing training sets directly can be difficult. Fortunately, techniques such as Synthetic Minority Oversampling Technique (SMOTE; Chawla et al. 2002), random values drawn from the Gaussian distribution (Peterson et al. 1998), and Gaussian Processes (GPs; Rasmussen & Williams 2005) modelling (GpFit) can simplify the problem to a large extent by simulating light curves. GPs have been used in several works to synthetically augment biased supernova training sets (Lochner et al. 2016; Narayan et al. 2018; Revsbech et al. 2018), variable stars (Faraway et al. 2016; Castro, Protopapas & Pichara 2018; Martínez-Palomera et al. 2018), and light-curve detrending (Aigrain, Parviainen & Pope 2016).

In this work, we are concerned only with periodic variable star classification and we present GPs for augmenting periodic variable star data using folded light curves. Secondly, we propose a new method, *R*andomly *A*ugmented *S*ampled *L*ight curves from magnitude *E*rror (RASLE[1]) to periodic variable star data for the first time, which synthetically augments the training set by sampling from the magnitude errors. We then compare the three data augmentation methods (SMOTE, GpFit, & RASLE) and their utility for improving variable star classification, trained with either a Random Forest (RF; Breiman 2001) classifier or eXtreme Gradient Boosting (XGBoost; Chen & Guestrin 2016) classifier. Finally, we incorporate a Bayesian optimization approach to find the best hyperparameters for the RF and XGBoost in the hierarchical classification (HC) scheme. We achieve an improvement of 1–4 per cent in terms of balanced-accuracy and G-mean scores at all levels in the HC, compared to the results of H19.

The structure of the paper is as follows. In Section 2, we describe the data set used in our analysis, while in Section 3, the three data augmentation algorithms used are explored. In Section 4, we provide a description of the various stages in the hierarchical classification pipeline; in Section 5, we present the classification results, and finally, we conclude in Section 6.

2 DATA

The Catalina Real-Time Transient Survey (CRTS; Drake et al. 2017) has produced a catalogue of periodic variable stars from 6 yr of optical photometry from the Siding Spring Survey. We consider only 11 classes from the CSDR2[2] data set as presented in Table 1

[1] After the preparation of this manuscript, we learnt that another team Gabruseva, Zlobin & Wang (2019) has come up with a similar method independently.

[2] Catalina Surveys Data Release 2.

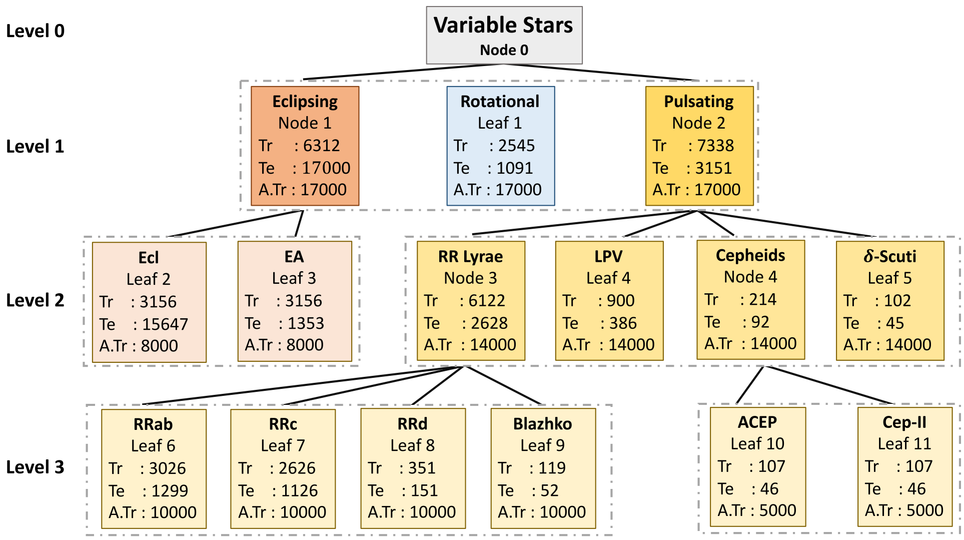

Figure 1. Hierarchical Tree classification with automated light curves augmentation for CRTS Data. The number of training examples (real LCs) is represented by *Tr*, the number of training examples after augmentation (both real and synthetic LCs) is represented by *A.Tr* and the number of test examples (real LCs) is represented by *Te*. At level 1, the real LCs in the training set are augmented and the dotted square represents a trained model (RF/XGBoost classifier). During testing phase, the classified examples in the test set move down the hierarchy at level 2. Afterwards, real LCs in the training set in level 1 moves to their respective branches at level 2. The real LCs are augmented and features are extracted. This process is repeated until it reaches all leaves in the hierarchy.

for our analysis. From Table 1, we observe that the data are heavily imbalanced. Thus to simplify our experimentation, we reduced the size of the largest class (Ecl) via random undersampling. We downsample this class to 4509 (this makes the number of Ecl examples comparable to the next biggest class, EA) and the remaining Ecl light curves (LCs) are then used for prediction. This is why the number of samples available for testing exceeds those for training as shown in Fig. 1.

3 DATA AUGMENTATION

While the undersampling methods (i.e. downsample Ecl and developing the hierarchical system) help to address some of the class imbalance issues, they are themselves insufficient, as minority class performance was not good enough for our purposes. We therefore provide three ways to oversample the data, a form of data augmentation necessary as some of the classes still outnumber other classes (see *Tr* values in Fig. 1). We augment the data via the generation of artificial data in order to increase the number of training samples by generating similar but not identical examples. In principal, the more data we have, the better our ML models will be as this technique helps to reduce overfitting. In this work, we consider three methods of augmentation: (i) SMOTE, (ii) RASLE, and (iii) GpFit.

3.1 Synthetic minority oversampling technique

The Synthetic Minority Oversampling Technique (SMOTE) inserts artificially generated minority class examples into a data set by operating in ‘feature space’ rather than ‘data space’. This technique helps to balance the overall class distribution. The standard implementation of SMOTE utilizes *k*-nearest neighbours (Buturovic 1993) to group similar class objects and to determine which class categories are in the minority class and need oversampling. To generate a new synthetic example, the *k*-nearest neighbour method is further used by first selecting an example in the minority class. The collection of feature values describing this example, it’s feature vector, is then combined with the feature vectors of one of its *k*-nearest neighbours chosen at random. The difference between the vectors of these two examples is computed and subsequently multiplied by a random number drawn between 0 and 1. This produces an entirely new synthetic feature vector. This process is repeated until enough synthetic examples have been created. Finally, the new augmented training set is comprised of both the synthetic examples and the real minority examples. In our pipeline, we utilize the ‘regular-SMOTE’ algorithm from the imbalanced-learn[3] (Lemaître, Nogueira & Aridas 2017) package.

3.2 *R*andomly *A*ugmented *S*ampled *L*ight curves from magnitude *E*rrors

The artificial examples generated by standard SMOTE may not truly represent data recorded during observations. One way around this is to generate artificial samples from existing data points in a more scientifically valid way. That is we randomly sample a selection of rare class examples, take their primary characteristics, and generate new examples from them by perturbing them in a principled way. We do this using the *R*andomly *A*ugmented *S*ampled *L*ight curves from magnitude *E*rrors (RASLE) method.

The application of RASLE is employed on unfolded-LCs; that is, each variable star is described by its time, magnitude, and error in magnitude. Using this information, we generate new light curves in the following way. Let us consider a probability distribution that can be concisely represented by a normal distribution. The probability distribution function (*pdf*) can be interpreted as going over the magnitude space vertically with the horizontal axis showing the probability that some value will occur. To construct the *pdf*, we make an assumption that the magnitude follows a normal distribution with mean, μ, to be equal to the true magnitude and the standard

[3] https://imbalanced-learn.readthedocs.io/en/stable/index.html

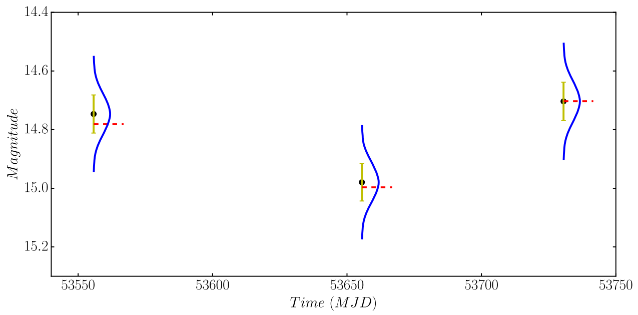

Figure 2. Generating new light curves by random sampling from a normal distribution. The true magnitude along with its error bars is shown in black and yellow. We assume a normal distribution with mean equal to the true magnitude and with sigma equal to the error in magnitude. We randomly draw one sample (red-dashed line) from each normal distribution to produce a completely new light curve.

deviation, σ, to be equal to the error in magnitude. For each data point at a specific time, we sample a single magnitude from the *pdf*. Each sampled magnitude is assigned the same time as in the original data. Fig. 2 shows an example of a light curve with the magnitude and error bars drawn for three specific times. The *pdf* of the magnitude is shown in blue and one magnitude is sampled randomly from the *pdf* shown in dotted red lines. The generated light curve is given the new (random) sampled magnitude with the same time value as in the original data.

3.3 Modelling light curves with GP

An ideal case for data augmentation is to use a well-defined model of the classes under consideration to create synthetic data. However, there is no available model valid for all the different variable stars considered. We therefore build a model describing variable stars using GPs (Rasmussen & Williams 2005) applied to CRTS data. We then use this model to generate artificial light curves, allowing us to augment our training data through the addition of new examples in a principled way, using the distributions of existing data to create them.

A GP is a distribution over functions. It is defined by a mean $\mu(t)$ and a covariance (kernel) function $c(t, t^{'})$ and is given as

$$f(t) \sim \mathrm{GP}(\mu(t), c(t, t')). \tag{1}$$

When the function f is computed at points t, the marginal distribution follows a multivariate normal distribution (Rasmussen & Williams 2005). The kernel function, c, takes two inputs and shows the similarity between them. When evaluating Bayesian inference, having the set of known function values for the training sets $\boldsymbol{f_x}$, and the set of known function values for the test sets $\boldsymbol{f_y}$, are normally distributed and is given as follows:

$$\begin{bmatrix} \boldsymbol{f_x} \\ \boldsymbol{f_y} \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_{f_x} \\ \boldsymbol{\mu}_{f_y} \end{bmatrix}, \begin{bmatrix} \boldsymbol{C}_{f_x f_x} & \boldsymbol{C}_{f_x f_y} \\ \boldsymbol{C}_{f_x f_y} & \boldsymbol{C}_{f_y f_y} \end{bmatrix} \right), \tag{2}$$

where the means of the training and test set are denoted by $\boldsymbol{\mu}_{f_x}$ and $\boldsymbol{\mu}_{f_y}$, respectively, and likewise $\boldsymbol{C}_{f_x f_x}$, $\boldsymbol{C}_{f_y f_y}$, $\boldsymbol{C}_{f_x f_y}$ represent the training, test, and train-test covariances/kernels. The conditional distribution, $\boldsymbol{f_x} \mid \boldsymbol{f_y} = \mathcal{P}$, is given by

$$\mathcal{P} \sim \mathcal{N}\left(\boldsymbol{C}_{f_x f_y} \boldsymbol{C}^{-1}_{f_y f_y} \left(\boldsymbol{f_y} - \boldsymbol{\mu}_y\right) + \boldsymbol{\mu}_{f_x}, \boldsymbol{C}_{f_x f_x} - \boldsymbol{C}_{f_x f_y} \boldsymbol{C}^{-1}_{f_y f_y} \boldsymbol{C}^{\mathsf{T}}_{f_x f_y} \right). \tag{3}$$

For a specific set of testing samples, equation (3) represents the posterior distribution. For a set of training examples $\mathcal{D}$, the posterior distribution is described by Rasmussen & Williams (2005)

$$\begin{aligned} f_y \mid \mathcal{D} &\sim \mathrm{GP}(\mu_{\mathcal{D}}, c_{\mathcal{D}}), \\ \mu_{\mathcal{D}}(t) &= \mu(t) + \boldsymbol{c}^{\mathrm{T}}_{T_s t} \boldsymbol{C}^{-1}(f_y - \mu), \\ c_{\mathcal{D}}(t, t') &= c(t, t') - \boldsymbol{c}^{\mathrm{T}}_{T_s t} \boldsymbol{C}^{-1} \boldsymbol{c}^{\mathrm{T}}_{T_s t'}, \end{aligned} \tag{4}$$

where the covariance vector between every training sample, T_s and t, is $\boldsymbol{c}_{T_s t} = c(T_s, t)$. The choice of the covariance function is established, based on the knowledge of the domain. In our case, we want to model light curves, so we require a kernel that can demonstrate both small fluctuations and smooth variations. Given the different characteristics of the various stars, an appropriate choice of the kernel in this work is the Matern 5/2 kernel given by

$$C_{\mathrm{Matern52}}(\Upsilon) = \left(1 + \frac{\sqrt{5}\Upsilon}{\ell} + \frac{5\Upsilon^2}{3\ell}\right) \exp\left(-\frac{\sqrt{5}\Upsilon}{\ell}\right), \tag{5}$$

where Υ and ℓ are the kernel hyperparameters; that is, Υ controls the degree of smoothness and ℓ is the characteristic length scale. We employ the GP regression using `George` (Ambikasaran et al. 2014) with kernel hyperparameters randomly initialized. Using our data and these randomly initialized hyperparameters, the negative log likelihood is calculated. Afterwards, these hyperparameters for the kernel are optimized (i.e. finding the best values for these parameters) using the Limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS; Fletcher 1987) optimization algorithm by minimizing the negative log likelihood.

The kernel with the optimized parameters is then used to fit the GP from which we sample synthetic light curves to augment our training set. Before fitting a GP to our data, we first convert the LCs from time distribution to phase distribution (folded-light curves) where the data is at the detected period for each variable star. We then randomly sampled synthetic LCs from the GP model to form the augmented training set. We show an example of `GpFit` on the folded-LCs for the different variable stars in Fig. 3 and the bottom plot illustrates 3 synthetic LCs randomly drawn from `GpFit`. We then unfolded the phases back into time space and used those synthetic LCs together with the original LCs as the training set.

4 METHOD DESCRIPTION

Drawing heavily from H19, we outline the general approach used to classify variable stars. In this study, we use RF and XGBoost classifiers. We use these classifiers for two reasons. Firstly, to ensure that results presented here are comparable with previous work. Secondly, because they have proven to be robust against the issues associated with class imbalance (Chen et al. 2004; Wang, Deng & Wang 2019). We then provide an overview of the HC scheme, together with the various stages we adopt to build the ML pipeline. Similar to H19, we pre-processed the light curves by applying a sigma-clipping method prior to any analysis.

4.1 Stage 1: hierarchical tree classifiers

H19's HC uses the astrophysical properties of the various sources to construct a tree-based structure to represent the different classes (Fig. 1). Each node/leaf represents a class – identified by the label inside the node/leaf – and the edges represent the relationship

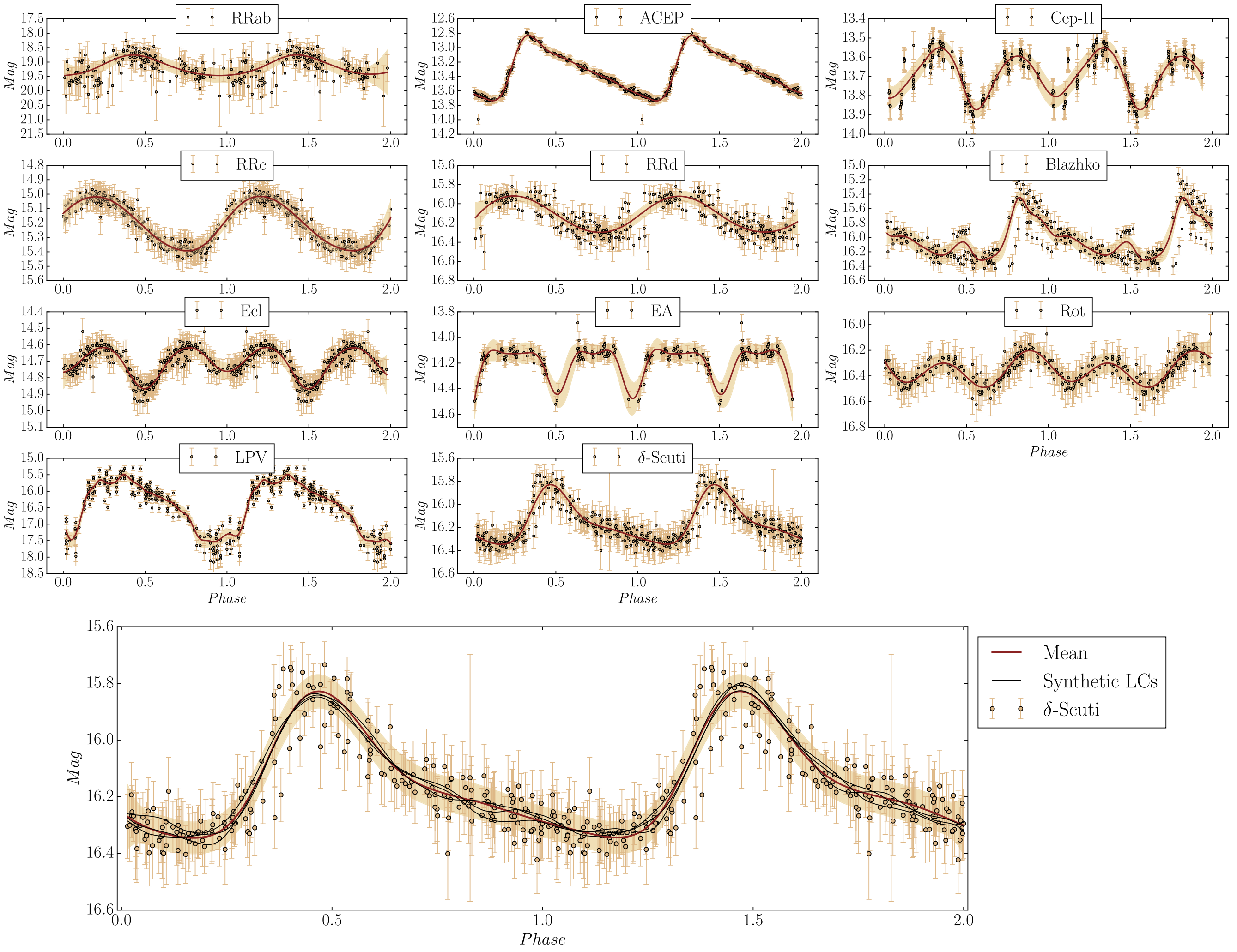

Figure 3. GPs offer a flexible approach to produce a smooth model of periodic light curves reported in magnitudes as a function of phase. This is demonstrated with model fits for each example of variable stars considered in the CRTS data set. The data points are illustrated in black-rounded dots along with the error bars. The mean of the GP fit is shown in brown with three standard deviation away from the mean, shown in shaded pale brown. In the bottom panel, the black lines represent three randomly drawn samples from the GP fit. These randomly sampled light curves, also known as synthetic LCs together with real LCs, are used in the training set.

between the superclass and subclass. For the HC, we use XGBoost and RF and then report the one that provides the best classification performance. XGBoost is a boosting algorithm and is a tree-based model which became popular since its inception in the ML community in 2016. XGBoost works in the same way as Gradient Boosting Decision Tree (GBDT, Friedman 2001). GBDT is an ensemble classification system that iteratively adds simple decision tree classifiers. The first classifier of the ensemble system is trained on the data, while the successive classifiers are trained on the errors of the predecessor classifiers. Unlike, in GBDT, XGBoost parallelizes this process/task and gives a substantial boost in speed. In addition, this classifier controls overfitting by using the regularization techniques, L1-norm (Tibshirani 1996) and L2-norm (Ng 2004). While a RF is simply an addition of decision trees that aggregate tree decisions. In astronomy, XGBoost has recently been used by Mirabal et al. (2016) who implemented this classifier for unknown point source classification in the *Fermi*-LAT catalogue and for the separation of pulsar signals from noise (Bethapudi & Desai 2018). In addition, XGBoost has also been applied for variable star classification (Sesar et al. 2017; Pashchenko et al. 2018; Kgoadi et al. 2019).

4.2 Stage 2: level-wise data augmentation in HC

Since the training set is still imbalanced after aggregating subclasses into superclasses, we use the three data augmentation techniques described in Section 3. Each technique is applied and tested independently in our HC based ML pipeline. For the SMOTE approach, features (the mean magnitude, standard deviation, skewness, kurtosis, mean-variance, amplitude and period) described in H19 are extracted from the real LCs. Then, SMOTE automatically balances the class distribution via the creation of synthetic examples sampled over the feature space, such that the size of the minority class equals the size of the majority class, cancelling the imbalance out. For example, considering level 1 in Fig. 1, the majority class is Pulsating, consisting of 7338 examples. Therefore, SMOTE adds new examples of the other two minority classes (eclipsing 6312 and rotational 2545) ensuring they both contain 7338 examples. This

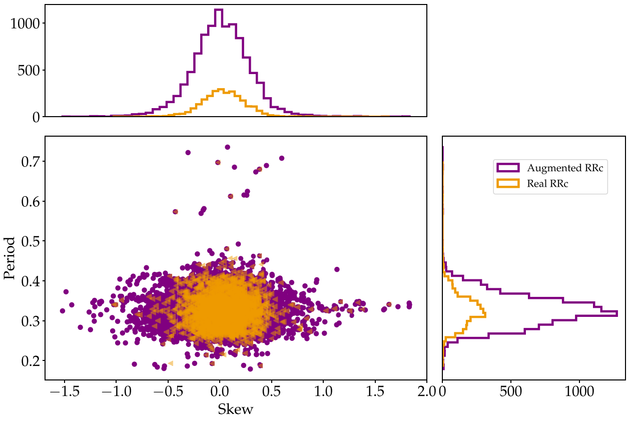

Figure 4. Period versus Skew distribution for real and synthetic LCs generated using `GpFit`.

process is repeated for each branch and level in the HC, where the training set is directly balanced according to the size of the majority class prior to data augmentation.

While for the `GpFit` and `RASLE` cases, we are generating new light curves based on real LCs, thus generating new synthetic LC examples. Therefore, our training set will consist of both real and synthetic LCs, whilst we test our ML pipeline with only real LCs. These two techniques can be used to oversample both the majority and minority class. The number of training examples after augmentation, *A.Tr* used for each level is given in Fig. 1. Afterwards, features are extracted from these LCs as discussed below.

4.3 Stage 3: feature extraction

In this work, similarly to H19, our features are based on 6 intrinsic statistical properties relating to location (mean magnitude), scale (standard deviation), variability (mean variance), morphology (skew, kurtosis, amplitude), and time (period). These features are highly interpretable, and robust against bias (Hosenie et al. 2019). For the `GpFit` and `RASLE` approach, the first six features are extracted directly from the augmented training set (containing both real and synthetic LCs) using the FATS library (Nun et al. 2015). Whilst for the period feature, the real LCs in the training set are assigned their respective period from the ascii-catalogue (Drake et al. 2017) and the synthetic LCs are assigned a period calculated by the method discussed in Section 4.3.1. For the test set we use only real LCs, hence the six features are extracted directly from the LCs and their period is obtained from the data catalogue. Therefore, we have 7 features that describe each variable star. Fig. 4 shows the distribution of the two most important features as investigated in H19 (period and skew) for real and synthetic LCs. We observe that the synthetic LCs show similar characteristics compared to the real LCs.

4.3.1 Period for augmented LCs

A synthetic LC is given a period based on the uncertainty in the estimated period of the real LC. In this case, the estimated period, *T*, is obtained from Drake et al. (2017). The associated uncertainty, σ_T for a given period is calculated as follows. A periodic signal is detected in a periodogram by the presence of a peak with a certain width and height. In Fourier perspective, we assume that there is a direct relationship between the precision with which a peak's frequency can be detected and the width of this peak; often known as the half-width at half-maximum (VanderPlas 2018) and is given by

$$\upsilon_{\frac{1}{2}} \approx \frac{1}{T}. \tag{6}$$

This can be viewed as interpreting the periodogram with the least-square method; that is, the inverse of the curvature of the peak is determined with the uncertainty (Ishak 2017). In the Bayesian perspective, this translates to a Gaussian curve fit to the exponentiated peak (Smith & Erickson 2012; Bretthorst 2013). Let us consider a periodogram with maximum value $P_{\max} = P(\upsilon_{\max})$, such that

$$P(\upsilon_{\max} \pm \upsilon_{\frac{1}{2}}) = \frac{P_{\max}}{2}. \tag{7}$$

Hence, the Bayesian uncertainty is calculated by approximating the exponentiated peak as a Gaussian, that is,

$$\exp\left[\mathrm{P}\left(\upsilon_{\max} \pm \delta\upsilon\right)\right] \propto \exp\left[\frac{-\delta\upsilon^2}{\left(2\sigma_\upsilon^2\right)}\right]. \tag{8}$$

The above equation can then be written as follows, and we obtain the uncertainty in frequency in equation (9).

$$\frac{P_{\max}}{2} \approx P_{\max} - \frac{\upsilon_{\frac{1}{2}}^2}{\left(2\sigma_\upsilon^2\right)}; \qquad \frac{\upsilon_{\frac{1}{2}}^2}{2\sigma_\upsilon^2} \approx \frac{P_{\max}}{2};$$

$$\sigma_\upsilon \approx \frac{\upsilon_{\frac{1}{2}}}{\sqrt{P_{\max}}}, \tag{9}$$

where $\delta\upsilon \approx \upsilon_{\frac{1}{2}}$. Considering the signal-to-noise ratio $\varphi = \mathrm{rms}[\frac{y_n - \mu}{\sigma_n}]$, where μ is the mean magnitude, y_n and σ_n is the magnitude and error in magnitude for each data point, respectively. We can then write the following equation for a well-fitted model:

$$P_{\max} \approx \frac{\varphi^2 N}{2}. \tag{10}$$

We then substitute equation (10) in equation (9) and the uncertainty in frequency can be written as:

$$\sigma_\upsilon \approx \upsilon_{\frac{1}{2}} \sqrt{\frac{2}{\varphi^2 N}}, \tag{11}$$

where $\upsilon_{\frac{1}{2}} \approx \frac{1}{T}$, N is the number of data points and φ is the signal to noise. We now compute the uncertainty in period by taking the derivative of σ_υ,

$$\frac{\partial\upsilon}{\partial T} \approx -\frac{1}{T^2}; \qquad \partial T = -T^2\sigma_\upsilon; \qquad \sigma_T^2 = T^4\sigma_\upsilon^2.$$

Hence, the uncertainty in period is then obtained using equation (12).

$$\sigma_T = T^2\sigma_\upsilon \tag{12}$$

where σ_T will be Gaussian if σ_υ is very small. A period value is given to each synthetic LC (generated either with `GpFit` or `RASLE`), by randomly sampling from a normal distribution with mean, *T* (the true period of the real LC from which the synthetic LCs are generated) and within 1σ-confidence interval, being σ_T using equation (12). An example of associating a period to an augmented LC is shown in Fig. 5.

4.4 Stage 4: training with Bayesian optimization

We first randomly split our data into training (70 per cent) and testing sets (30 per cent). The training set moves through the first level in

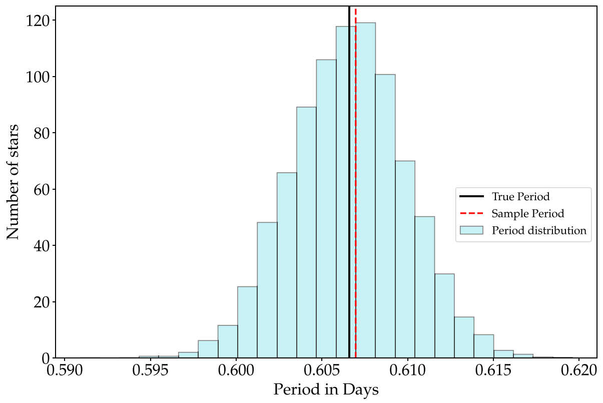

Figure 5. For each synthetic LC, a period value (red vertical line) is randomly sampled from a normal distribution, with mean T being the true period of the real LC and σ_T being the computed uncertainty of the period, *T*.

the HC scheme discussed in Section 4.1. The training examples are then augmented using one of the three data augmentation techniques and features are extracted where appropriate. Afterwards, the model (see dotted square at level 1 in Fig. 1) is trained using either the RF or XGBoost classifier, as required. We then use a Bayesian optimization approach to find the best hyperparameters for the ML algorithm. It has been demonstrated for large parameter spaces that Bayesian optimization, also known as sequential model-based optimization (SMBO; Hutter, Hoos & Leyton-Brown 2011) performs better than either manual or randomized grid searches (Bergstra, Yamins & Cox 2013). It is one of the most efficient techniques for hyperparameter optimization of ML algorithms.

In this work, we used SMBO techniques compared to H19, who used a randomized grid-search for hyperparameter optimization. Before applying the above methods, we perform a stratified cross validation. The training data is split into fivefolds, where four different folds are kept for training each time and an independent fold is used for validation. We then use the SMBO method, `HyperOpt` (Bergstra et al. 2013) to find the best hyperparameters on the fourfolds and validated the model on the independent fold. We then evaluate our trained model based on balanced-accuracy, G-mean, precision, recall, and F1-scores, on real LCs in the test set.

5 ANALYSIS AND RESULTS

This paper is mostly concerned with learning from an imbalanced class distribution. The problem is typically addressed using the following approaches.

(i) *Data level*: We employ three approaches to the HC scheme in such a way that the class distributions are rebalanced directly; that is, it is a first proof of principle application of a level-wise augmentation in Hierarchical taxonomy, where we resample the original data set to achieve a desired balancing.

(ii) *Algorithm level*: We focus on using two different algorithms (RF and XGBoost), together with a Bayesian optimization algorithm for hyperparameter tuning, to achieve improved performance on the minority class examples.

The HC algorithm is trained on both real and artificially augmented data and tested on real data. We show the results of the three data augmentation techniques in Table 2. We assess the consistency of the results based on balanced-accuracy and G-mean scores. The shaded blue colour represents the augmentation methods, which when applied together with the HC classifier, yielded improved results over H19. We found that `GpFit` achieves the best performance measures compared to H19 at all levels in the HC. When using the `GpFit` method, we found that our RF implementation performs best at all HC levels when compared to H19 and we highlight this result in grey. In addition, we found that XGBoost, similarly to the RF, provides good performance for variable star classification. Moreover, in H19, we show that the HC model is neither underfitting nor overfitting by plotting precision-recall curves at different levels. In this paper, we assess the consistency of the results using `GpFit` and RF by plotting the Receiver Operator Characteristic (ROC) curve for each class (see Fig. 6). We note that classification performance is very good. The area under the ROC curve (AUC) values are greater than 0.95 for several classes, except for Rotational, RRd, and Blazhko. The reasons for these misclassification are further discussed in Section 5.1.

We improve upon the result obtained in H19. For instance, the balanced-accuracy increases from 61 to 65 per cent in level 1, from 86 to 88 per cent at level 2 for the eclipsing node, from 86 to 87 per cent for subclasses of RR Lyrae at level 3, and finally from 81 to 83 per cent for Cepheids at level 3. To check the consistency and robustness of our new approach, we perform an additional step. We use different splits ($K = 5, 6, ...,10$) during cross-validation and predict on the 30 per cent test set. With these analyses, we obtain an uncertainty on the metric scores considered, for example for Cepheids at level 3, a 0.83 ± 0.02 balanced-accuracy and 0.91 ± 0.01 G-mean score are obtained. We obtain similar results at different levels in the hierarchy. In these analyses, we observe that we have not made a huge improvement to H19, in terms of minority classes and we explain the various reasons that might lead to this outcome in Section 5.1.

5.1 Impact of imbalance on classification performance

Training a classifier upon imbalanced data does not guarantee poor generalization performance (Galar et al. 2011). Regardless of imbalance, if the features or the training data themselves are discriminative enough to provide a clear separation between the different classes, then classifiers will likely generalize well. However, there are three main characteristics of imbalanced data sets that make it hard for a classifier to discriminate the minority from the majority classes. These are as follows:

(i) Small sample sizes (He & Garcia 2008; Galar et al. 2011)

(ii) Class inseparability (Japkowicz & Stephen 2002; Galar et al. 2011; see Figs 7a and 8)

(iii) Small disjuncts (see Fig. 7b)

Ultimately, the training data showing these characteristics conspire to make it hard for any classifier to build an optimal decision boundary, leading to suboptimal classifier performance. These characteristics are seen at some levels in the HC. In this section, we illustrate these effects at level 3 using the subclasses of RR Lyrae. Fig. 7(a) shows that some classes have overlapping characteristics, which leads to poor performance. We observe similar characteristics (class-overlapping) for the subclasses of RR Lyrae in Fig. 8(a), even after balancing the classes in the training set. These overlapping characteristics are due to the fact that there are no physical distinction between some of the subclasses. As can be seen in Fig. 8(a), RRab and RRc classes can reasonably be separated based on their period alone. RRab are variable stars pulsating in

Table 2. Evaluation metrics used to summarize the HC pipeline with the application of three methods of data augmentation. We present the balanced-accuracy and G-mean scores level-wise to evaluate our model. H19 results are presented in bold text in the table. It is seen that the HC pipeline performs fairly well with data augmentation, achieving G-mean scores above ∼80 per cent at all levels. The shaded blue represents the augmentation methods that outperform H19. We observe that at all levels, `GpFit` together with RF, performs better than H19 and it is represented in shaded grey. The '∼' represents a single value for the computed average metrics by taking into consideration the overall classes.

Augmentation techniques	Classifiers	G Mean	Balanced-accuracy
First level: eclipsing, rotational, and pulsating classification			
H19 (no augmentation)	**RF**	**0.78/0.78/0.86 (∼0.79)**	**0.59/0.60/0.75 (∼0.61)**
	XGBoost	0.80/0.77/0.89 (∼0.81)	0.63/0.59/0.80 (∼0.65)
SMOTE	RF	0.80/0.78/0.89 (∼0.81)	0.63/0.60/0.79 (∼0.65)
	XGBoost	0.82/0.76/0.89 (∼0.83)	0.66/0.57/0.79 (∼0.68)
RASLE	RF	0.82/0.77/0.89 (∼0.83)	0.66/0.58/0.79 (∼0.68)
	XGBoost	0.80/0.75/0.89 (∼0.81)	0.63/0.56/0.79 (∼0.65)
GpFit	RF	0.80/0.75/0.89 (∼0.81)	0.63/0.56/0.78 (∼0.65)
Second level: RR Lyrae, LPV, Cepheids, and δ-Scuti			
H19 (no augmentation)	**RF**	**0.99/1.00/0.97/1.00 (∼0.99)**	**0.98/0.99/0.93/1.00 (∼0.98)**
SMOTE	XGBoost	0.99/1.00/1.00/0.95 (∼0.99)	0.97/0.99/1.00/0.90 (∼0.97)
	RF	0.99/1.00/1.00/0.96 (∼0.99)	0.97/0.99/1.00/0.92 (∼0.97)
	XGBoost	0.99/1.00/0.99/0.93 (∼0.99)	0.98/1.00/0.98/0.85 (∼0.98)
RASLE	RF	0.99/1.00/1.00/0.94 (∼0.99)	0.98/0.98/1.00/0.88 (∼0.98)
	XGBoost	0.99/1.00/0.99/0.95 (∼0.99)	0.97/0.99/0.97/0.99 (∼0.98)
GpFit	RF	0.99/1.00/1.00/0.97 (∼0.99)	0.97/0.99/1.00/0.93 (∼0.98)
Second level: Ecl and EA			
H19 (no augmentation)	**RF**	**0.93/0.93 (∼0.93)**	**0.86/0.86 (∼0.86)**
	XGBoost	0.94/0.94 (∼0.94)	0.88/0.88 (∼0.88)
SMOTE	RF	0.94/0.94 (∼0.94)	0.88/0.88 (∼0.88)
RASLE	XGBoost	0.93/0.93 (∼0.93)	0.85/0.85 (∼0.85)
	RF	0.93/0.93 (∼0.93)	0.85/0.86 (∼0.86)
	XGBoost	0.93/0.93 (∼0.93)	0.88/0.88 (∼0.88)
GpFit	RF	0.94/0.94 (∼0.94)	0.87/0.88 (∼0.88)
Third level: RRab, RRc, RRd, and Blazhko			
H19 (no augmentation)	**RF**	**0.97/0.92/0.65/0.44 (∼0.92)**	**0.94/0.85/0.40/0.18 (∼0.86)**
SMOTE	XGBoost	0.95/0.92/0.67/0.58 (∼0.91)	0.91/0.83/0.42/0.31 (∼0.83)
	RF	0.95/0.82/0.47/0.33 (∼0.91)	0.91/0.82/0.47/0.33 (∼0.83)
	XGBoost	0.96/0.95/0.56/0.53 (∼0.92)	0.93/0.89/0.30/0.26 (∼0.87)
RASLE	RF	0.97/0.95/0.52/0.52 (∼0.92)	0.94/0.90/0.25/0.25 (∼0.87)
	XGBoost	0.97/0.93/0.57/0.44 (∼0.92)	0.94/0.86/0.30/0.17 (∼0.85)
GpFit	RF	0.97/0.93/0.56/0.41 (∼0.92)	0.94/0.87/0.32/0.26 (∼0.87)
Third level: ACEP and Cep-II			
H19 (no augmentation)	**RF**	**0.90/0.90 (∼0.90)**	**0.82/0.80 (∼0.81)**
SMOTE	XGBoost	0.88/0.88 (∼0.88)	0.78/0.76 (∼0.77)
	RF	0.88/0.88 (∼0.88)	0.78/0.76 (∼0.77)
	XGBoost	0.88/0.88 (∼0.88)	0.77/0.78 (∼0.77)
RASLE	RF	0.88/0.88 (∼0.88)	0.77/0.78 (∼0.78)
	XGBoost	0.88/0.88 (∼0.88)	0.78/0.78 (∼0.78)
GpFit	RF	0.91/0.91 (∼0.91)	0.84/0.82 (∼0.83)

fundamental mode, RRc stars pulsate in the first overtone while RRd stars simultaneously pulsate in the fundamental and first overtone. Therefore, RRd's form part of both RRab and RRc variable stars at the same time. In addition, Blazhko stars are found among RRab stars (Jurcsik et al. 2009), RRc stars (Netzel et al. 2018), and even RRd stars (Jurcsik et al. 2015). This explains the poor performance of the classifier for separating RRd and Blazhko stars, even after balancing the classes. In addition, we also present a *t*-distributed stochastic neighbour embedding (t-SNE; van der Maaten & Hinton 2008) of the minority classes (Blazhko, δ-Scuti, ACEP & Cep-II) in Fig. 8(b) after augmenting them using the `GpFit` method. The result shown in Fig. 8(a) does not differ when we perform multiple runs with different parameters. Each time we find small disjuncts in the feature space, showing characteristics similar to those shown in Fig. 7(b), thus making it difficult for the classifier to construct a decision boundary.

In this paper, we found that training the HC with class-balanced data has the effect of improving balanced-accuracy and G-mean scores. However, the minority classes are still misclassified. Although these results suggest that balancing the class distribution is not sufficient for classifying the minority classes, their capacity to prevent overfitting and increase the recall rate makes them appealing.

Another reason that leads to misclassification – the lack of a standard set of correctly classified (i.e. where the ground truth is certain) variable star example useful for training. Drake et al. (2017)

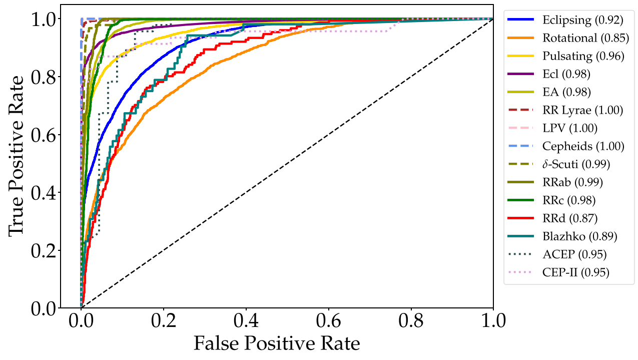

Figure 6. Receiver operating characteristic (ROC) curves for each node in the hierarchical model. Each curve represents a different variable star class with the area under the ROC curve (AUC) score in brackets. This metric is computed on the 30 per cent of the data set used for testing.

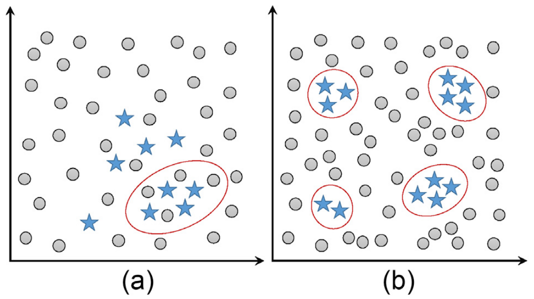

Figure 7. Demonstration of (a) class inseparability and (b) small disjuncts in feature space.

investigated the level of agreement of their classifications with the International Variable Star Index (VSX; Watson, Henden & Price 2006). They found that

(i) VSX has not classified any of their Blazhko stars, but instead simply classify them as RRab stars,

(ii) VSX classified many of their contact binaries as detached and semi-detached binaries,

(iii) most of their rotational stars (spotted or ellipsoidal variables) have been classified as contact binaries, and

(iv) most of their RRd stars have been misclassified as other stars (RRab, RRc) by VSX.

We observe similar misclassifications when using our automated HC pipeline, even after balancing the classes. With the presence of so many misclassified objects, we can plausibly say neither Drake et al. (2017) or VSX can be considered as providing ground truth. Therefore, there is a real need to have a standard set of correctly identified variable stars that can be utilized for training automated machine learning methods. It is imperative to train these sophisticated ML-based algorithms with accurately calibrated priors in order to obtain reliable classification outputs.

6 CONCLUSION

In this paper, we present a new approach for tackling the problem of imbalanced data: a level-wise data augmentation in a hierarchical classification framework. Our code is publicly available at https://github.com/Zafiirah13/Imbalance-Learning-for-Variable-Star-Classification-using-Machine-Learning. Through an empirical investigation, we demonstrate three techniques for augmenting data; that is, `SMOTE`, `RASLE`, and `GpFit` are applied to variable star data. We show that using RF and `GpFit` together can effectively improve recall rates, hence increasing the balanced-accuracy and G-mean scores by 1–4 per cent. Although, the results show that even after balancing the training set level-wise, such approaches do not prevent the misclassification of the minority class, though their capacity to increase other metrics (e.g. recall) still makes their application appealing. Perhaps, the misclassification occurs because these objects are just not easily separable and we observe similar misclassifications in this paper as determined by Drake et al. (2017) when they compared their results with VSX. Therefore, it is imperative to have correctly labelled data that can accurately be used to train automated ML pipeline in order to output reliable classification performance.

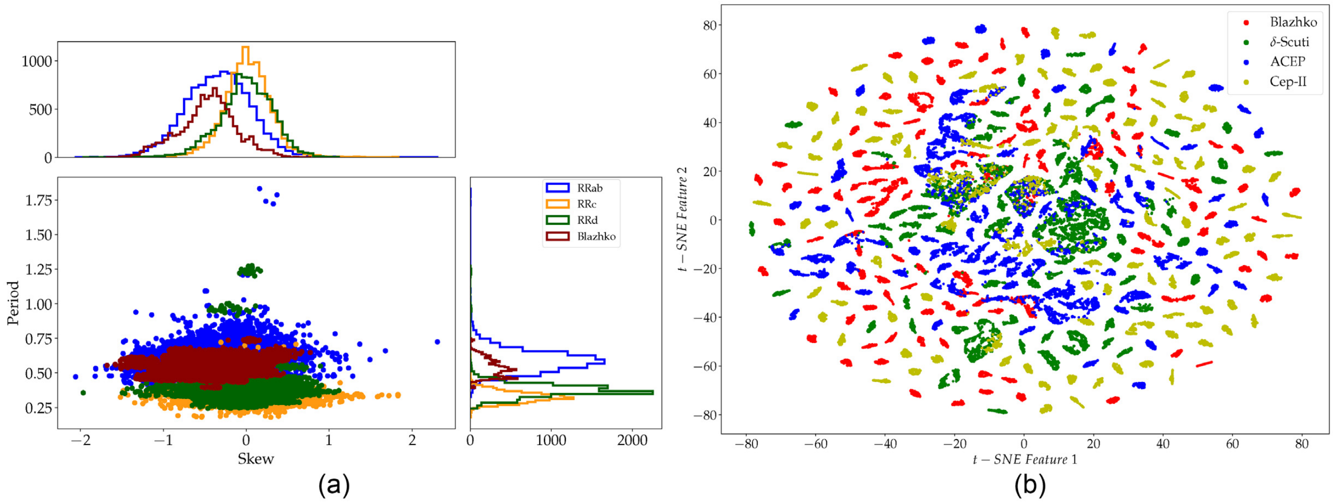

Figure 8. (a) Shows the Period-Skew distribution of RRab, RRc, RRd, and Blazhko after augmenting each respective class to 10 000 examples. We note that the classes are still overlapping in the feature space, even after the augmentation process. (b) Illustrates small disjoints in feature space using *t*-distributed stochastic neighbour embedding (t-SNE) visualization in the small sample size data (Blazhko, δ-Scuti, ACEP, and Cep-II), after augmentation. No distinct separation is seen within the feature space.

ACKNOWLEDGEMENTS

We thank the referee for useful comments and suggestions for improving this paper. ZH acknowledges support from the UK Newton Fund as part of the Development in Africa with Radio Astronomy (DARA) Big Data project delivered via the Science & Technology Facilities Council (STFC). BWS acknowledges funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 694745). AM is supported by the Imperial President's PhD Scholarship. VMB acknowledges funding from the National Research Foundation (grant nos 98969 and 119446).

REFERENCES

Agarwal D., Aggarwal K., Burke-Spolaor S., Lorimer D. R., Garver-Daniels N., 2019, preprint (arXiv:1902.06343)

Aigrain S., Parviainen H., Pope B., 2016, MNRAS, 459, 2408

Ambikasaran S., Foreman-Mackey D., Greengard L., Hogg D. W., O'Neil M., 2015, IEEE transactions on pattern analysis and machine intelligence, 38, 252

Benavente P., Protopapas P., Pichara K., 2017, ApJ, 845, 147

Bergstra J., Yamins D., Cox D. D., 2013, Proceedings of the 12th Python in Science Conference. Citeseer, p. 13

Bethapudi S., Desai S., 2018, Astron. Comput., 15, 23

Breiman L., 2001, Mach. Learn., 45, 5

Bretthorst G. L., 2013, Bayesian Spectrum Analysis and Parameter Estimation. Vol. 48. Springer Science & Business Media

Buturovic L. J., 1993, Pattern Recognit., 26, 611

Castro N., Protopapas P., Pichara K., 2018, ApJ, 155, 16

Chawla N. V., Bowyer K. W., Hall L. O., Kegelmeyer W. P., 2002, J. Artif. Intell. Res., 16, 321

Chen T, Guestrin C, 2016, preprint (arXiv:1603.02754)

Chen C. et al., 2004, Using random forest to learn imbalanced data, Vol. 110. Univ. California, Berkeley, p. 24

Chen H., Diethe T., Twomey N., Flach P. A., 2018, in ESANN

Drake A. J. et al., 2017, MNRAS, 469, 3688

Faraway J., Mahabal A., Sun J., Wang X.-F., Wang Y. G., Zhang L., 2016, Stat. Anal. Data Mining: ASA Data Sci. J., 9, 1

Fawcett T., Provost F., 1996, Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. p. 8

Fletcher R., 1987, John Wiley & Sons

Friedman J., 2001, Ann. Statist., 29, 1189

Gabruseva T., Zlobin S., Wang P., 2019, preprint (arXiv:1909.05032)

Galar M., Fernandez A., Barrenechea E., Bustince H., Herrera F., 2011, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 42. IEEE, p. 463

He H., Garcia E. A., 2008, IEEE Transactions on Knowledge & Data Engineering. IEEE, p. 1263

Hosenie Z., Lyon R. J., Stappers B. W., Mootoovaloo A., 2019, MNRAS, 488, 4858

Hoyle B., Rau M. M., Bonnett C., Seitz S., Weller J., 2015, MNRAS, 450, 305

Hutter F., Hoos H. H., Leyton-Brown K., 2011, International Conference on Learning and Intelligent Optimization. Springer, p. 507

Ishak B., 2017, in Ivezić Ž., Connolly A. J., Van der Plas J. T., Gray A., eds, Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data. Princeton University Press

Japkowicz N., Stephen S., 2002, Intell. Data Anal., 6, 429

Jurcsik J. et al., 2009, MNRAS, 400, 1006

Jurcsik J. et al., 2015, ApJS, 219, 25

Kgoadi R., Engelbrecht C., Whittingham I., Tkachenko A., 2019, preprint (arXiv:1906.06628)

Kim D.-W., Bailer-Jones C. A., 2016, A&A, 587, A18

Koch D. G. et al., 2010, ApJ, 713, L79

Lemaître G., Nogueira F., Aridas C. K., 2017, J. Mach. Learn. Res., 18, 1

Lochner M., McEwen J. D., Peiris H. V., Lahav O., Winter M. K., 2016, ApJS, 225, 14

Mahabal A., Sheth K., Gieseke F., Pai A., Djorgovski S. G., Drake A., Graham M., 2017, IEEE Symposium Series on Computational Intelligence. IEEE, p. 2757

Martínez-Palomera J. et al., 2018, AJ, 156, 186

Mirabal N., Charles E., Ferrara E., Gonthier P. L., Harding A. K., Sánchez-Conde M. A., Thompson D. J., 2016, ApJ, 825, 69

Narayan G., Zaidi T., Soraisam M. D., Wang Z., Lochner M., Matheson T., Saha A., Yang S. et al., 2018, ApJS, 236, 9

Netzel H., Smolec R., Soszyński I., Udalski A., 2018, MNRAS, 480, 1229

Ng A. Y., 2004, Proceedings of the Twenty-First International Conference on Machine Learning. p. 78

Nun I., Protopapas P., Sim B., Zhu M., Dave R., Castro N., Pichara K., 2015, preprint (arXiv:1506.00010)

Pashchenko I. N., Sokolovsky K. V., Gavras P., 2018, MNRAS, 475, 2326

Peterson B. M., Wanders I., Horne K., Collier S., Alexander T., Kaspi S., Maoz D., 1998, PASP, 110, 660

Rasmussen C. E., Williams C. K. I., 2005, Gaussian processes for machine learning. The MIT Press

Revsbech E. A., Trotta R., van Dyk D. A., 2018, MNRAS, 473, 3969

Sesar B. et al., 2017, AJ, 153, 204

Smith C. R., Erickson G., 2012, Maximum-Entropy and Bayesian Spectral Analysis and Estimation Problems: Proceedings of the Third Workshop on Maximum Entropy and Bayesian Methods in Applied Statistics, August 1–4, Vol. 21, 1983. Springer Science & Business Media, Wyoming, USA

Tibshirani R., 1996, J. R. Stat. Soc. Ser. B (Methodol.), 58, 267

Tsang B. T.-H., Schultz W. C., 2019, ApJ, 877, L14

Udalski A., Szymanski M., Soszynski I., Poleski R., 2008, preprint (arXiv:0807.3884)

Udalski A., Szymański M., Szymański G., 2015, preprint (arXiv:1504.05966)

VanderPlas J. T., 2018, ApJS, 236, 16

van der Maaten L., Hinton G., 2008, J. Mach. Learn. Res., 9, 2579

Wang C., Deng C., Wang S., 2019, preprint (arXiv:1908.01672)

Watson C. L., Henden A. A., Price A., 2006, Society for Astronomical Sciences Annual Symposium. p. 47

Zong B., Song Q., Min M. R., Cheng W., Lumezanu C., Cho D., Chen H., 2018

Zorich L., Pichara K., Protopapas P., 2020, MNRAS, 492, 2897

This paper has been typeset from a TEX/LATEX file prepared by the author.