Mon. Not. R. Astron. Soc. **000**, 1–17 (2017) Printed 20 October 2017 (MN LATEX style file v2.2)

Machine learning search for variable stars

Ilya N. Pashchenko[1⋆], Kirill V. Sokolovsky[2,3,1]†, Panagiotis Gavras[2]‡

[1]*Astro Space Center of Lebedev Physical Institute, Profsoyuznaya St. 84/32, 117997 Moscow, Russia*
[2]*IAASARS, National Observatory of Athens, Vas. Pavlou & I. Metaxa, 15236 Penteli, Greece*
[3]*Sternberg Astronomical Institute, Moscow State University, Universitetskii pr. 13, 119992 Moscow, Russia*

Accepted XXXX Month XX. Received 2017 Month XX; in original form 2017 Month XX

ABSTRACT

Photometric variability detection is often considered as a hypothesis testing problem: an object is variable if the null-hypothesis that its brightness is constant can be ruled out given the measurements and their uncertainties. Uncorrected systematic errors limit the practical applicability of this approach to high-amplitude variability and well-behaving data sets. Searching for a new variability detection technique that would be applicable to a wide range of variability types while being robust to outliers and underestimated measurement uncertainties, we propose to consider variability detection as a classification problem that can be approached with machine learning. We compare several classification algorithms: Logistic Regression (*LR*), Support Vector Machines (*SVM*), *k* Nearest Neighbors (*kNN*) Neural Nets (*NN*), Random Forests (*RF*) and Stochastic Gradient Boosting classifier (*SGB*) applied to 18 features (variability indices) quantifying scatter and/or correlation between points in a light curve. We use a subset of OGLE-II Large Magellanic Cloud (LMC) photometry (30265 light curves) that was searched for variability using traditional methods (168 known variable objects identified) as the training set and then apply the *NN* to a new test set of 31798 OGLE-II LMC light curves. Among 205 candidates selected in the test set, 178 are real variables, 13 low-amplitude variables are new discoveries. We find that the considered machine learning classifiers are more efficient (they find more variables and less false candidates) compared to traditional techniques that consider individual variability indices or their linear combination. The *NN*, *SGB*, *SVM* and *RF* show a higher efficiency compared to *LR* and *kNN*.

Key words: methods: data analysis, methods:statistical, stars: variables: general

1 INTRODUCTION

A variety of astrophysical phenomena manifest themselves with optical variability. The incomplete list includes accretion, ejection, explosions, gravitational lensing, stellar magnetic activity, pulsations and eclipses. Historically, variable objects were mostly identified by comparing their brightness recorded at a pair of images (Hoffmeister, Richter & Wenzel 1990). The photographic images were compared with a blink-comparator or by placing a positive image of a photographic plate taken at one epoch on top of the negative plate taken at a different epoch. Difference image analysis (DIA; Alard & Lupton 1998, Bramich et al. 2016) can be thought of as a modern software implementation of this idea. The pair-wise image comparison has the obvious drawback that it can detect only high-amplitude variability: the object's brightness difference between the two images should be a few times larger than measurement errors associated with individual images. To detect low-amplitude variability one needs to construct and analyze object's light curve containing multiple measurements in order to effectively average-out individual measurement errors. The multiple measurements may be performed using DIA, point spread function (PSF) fitting, or aperture photometry. The effect of considering multiple measurements altogether instead of pair-wise is illustrated by the large number of high-amplitude δ Scuti/SX Phoenicis stars (HADS) found using digitized photographic plates by Kolesnikova et al. (2010). These plates were earlier searched for variability by comparing pairs of images, but this search failed to identify the majority of HADS variables despite having a comparable accuracy of individual measurements.

Detection of variability in a light curve may be considered a hypothesis testing problem (Eyer 2005, Huber, Everett & Howell 2006, de Diego 2010, Piquard et al. 2001): an object is variable if the null-hypothesis that its brightness is constant can be ruled out. Uncorrected systematic errors and corrupted measurements limit the practical applicability of this approach to well-behaving data sets. Tests that take into account not only the measurements themselves, but also the order (Tamuz, Mazeh & North 2006, Figuera Jaimes et al. 2013, Ferreira Lopes & Cross 2016)

⋆ E-mail: in4pashchenko@gmail.com
† kirx@kirx.net
‡ pgavras@noa.gr

and times (Stetson 1996, Zhang et al. 2003) at which the measurements were taken were also proposed. The variability detection threshold for these tests often has to be determined empirically for a given data set. Sokolovsky et al. (2017) investigated 24 "variability indices" (also referred to as "light curve features") – statistical characteristics quantifying scatter and correlation between points in a light curve. The ability of these indices used individually or in a linear combination to discriminate variable objects from non-variable ones was compared using multiple real and simulated data sets.

In this paper we explore a new variable star selection technique that outperforms all the individual (or linearly combined) indices considered by Sokolovsky et al. (2017). This is achieved by finding useful non-linear combinations of these indices. We consider variability detection not as a hypothesis testing problem, but as a binary classification problem (variable vs. non-variable objects) and apply machine learning techniques to solve it. The proposed technique does not depend critically on accurate photometric error estimates and is not sensitive to individual outlier measurements[1]. It can be applied to *any* large set of light curves given a representative subset of these light curves has been manually classified as variable and non-variable ones. This subset is used to train a machine learning (ML) classifier that will process the rest of the data.

While preparing this paper we learned that the General Variability Detection module of the Gaia Variability Analysis pipeline (Eyer et al. 2017) is using a Random Forest classifier trained on multiple variability indices computed for variables identified in the OGLE-IV Gaia south ecliptic pole field by Soszyński et al. (2012). Earlier, Shin, Sekora & Byun (2009), Shin et al. (2012) proposed to use multiple variability indices together combining them with an infinite Gaussian mixture model. The method of Pawlak et al. (2016), while focusing solely on eclipsing binaries, is similar in spirit to the method proposed here. The authors use a set of features computed by the BLS period-finding algorithm (Kovács, Zucker & Mazeh 2002) as an input for the Random Forest classifier trained on OGLE-III eclipsing binaries (Graczyk et al. 2011) in one of the OGLE-IV (Udalski, Szymański & Szymański 2015) fields. Elorrieta et al. (2016) used machine learning to identify RRab stars in the VVV survey data (Minniti et al. 2010). Pérez-Ortiz et al. (2017) propose a set of light curve features robust to individual outlier measurements and use them to compare multiple machine learning algorithms on classified OGLE-III light curves. Taking into account the experience of authors listed above, we suggest the following points that we try to justify in this work:

- Machine learning can be used for variability detection *in general*, not only for extracting variable objects of specific types, one type at a time.
- This general variability detection problem is tractable for many different supervised learning algorithms.
- Systematic search for optimal hyperparameters of a learning algorithm is needed to achieve its best performance.
- Variability search with machine learning is effective even with modest training sample size containing hundreds of known variables. The sample may be highly imbalanced (few variables and many constant stars).

The paper is structured as follows: Section 2 describes the test data, Section 3 describes the proposed variable object selection technique and Section 4 discusses the results of its application to the test data while Section 5 summarizes our findings.

[1] We use light curve features (MAD, IQR, $1/\eta$; Table 2) that are by design not sensitive to outliers (Sokolovsky et al. 2017) and do not depend on the estimated photometric errors. While this is not the case for other features (σ, $\chi^2_{\rm red}$,...) these features will end up having less predictive power compared to the robust features if the sensitivity to outliers or the incorrectly estimated errors constitute a problem in the given data set. ML techniques described in Section 3.2 include procedures (bagging, dropout, appropriate choice of loss function) designed to minimize dependency on individual objects with outlier lightcurve feature values (that may result from corrupted photometry). The OGLE-II and $TF1$ data we use for tests are plagued with outlier measurements which do not end up having a critical impact on our ability to identify variable objects (Sections 4.1 and 4.3).

2 INPUT DATA

The primary input for variability search is a set of time-series brightness measurements collected for a number of sources – a set of light curves. The light curves may be quite diverse even within one data set. They may have different number of measurements as not all sources are detected and successfully measured in each image. Measurements of different sources may be influenced to a different extent by systematic effects that depend on source color or its position on an image. Some measurements get corrupted by random events such as cosmic ray hits or object's image falling on a bad pixel. Light curves of variable sources may show a variety of patterns depending on the variability type and period (or typical timescale for non-periodic variables). To characterize such diverse light curves in a uniform way we extract a set of light curve features (or "variability indices"). We use VaST code (Sokolovsky & Lebedev 2017) to extract the features while other feature extraction codes are also publicly available (Nun et al. 2015, Kim & Bailer-Jones 2016, Christ, Kempa-Liehr & Feindt 2016). The features computed by VaST are meant to be used for variability *detection* while the other codes are mainly concerned with features useful for *classification* of detected variables, but there is a great deal of overlap between the features useful for these two tasks. In Section 2.1 we describe the photometric data set used for our tests and discuss its inherent biases in Section 2.2. In Section 2.3 we present the utilized set of light curve features.

2.1 Light curves

As input data we use a small subset of publicly available Optical Gravitational Lensing Experiment phase two (OGLE-II) PSF fitting I-band photometry of the field LMC_SC20 towards the Large Magellanic Cloud (LMC; Szymanski 2005). OGLE-II observations are conducted with the 1.3 m Warsaw telescope at the Las Campanas Observatory, Chile (Udalski, Kubiak & Szymanski 1997). Public OGLE-II photometry was used earlier to test new variability detection techniques by Shin & Byun (2007). OGLE-II LMC data were searched for various specific types of variable objects including microlensing events (Wyrzykowski et al. 2009), variable red giants (Soszynski et al. 2004, Kiss & Bedding 2003, Soszynski et al. 2005), RR Lyrae stars (Soszynski et al. 2003), eclipsing binaries (Wyrzykowski et al. 2003), cataclysmic variables (Cieslinski et al. 2003), quasars (Eyer 2002), Cepheids (Udalski et al. 1999). Zebrun et al. (2001) constructed a comprehensive catalog of candidate variables (of all types) detected with DIA. The field was also covered by later phases of the OGLE project (Udalski et al.

2008, Udalski, Szymański & Szymański 2015) as well as other time-domain surveys including VMC (Cioni et al. 2011), EROS (Tisserand et al. 2007), MACHO (Alcock et al. 2000, Becker et al. 2005). Overall, the test field is well-studied for variability.

The LMC_SC20 data set was manually searched for variable objects by Sokolovsky et al. (2017). The authors identified 20 new variable stars hinting that variability detectable in OGLE-II data is still not fully explored. These findings also highlight the fact that in practice, one cannot expect to have a complete sample of variables stars by just searching catalogs of known variables, even in such well-studied sky region as the LMC.

The use of the LMC_SC20 data set allows us to directly compare the effectiveness of the variability detection technique proposed here to the techniques discussed by Sokolovsky et al. (2017). Specifically, we want to compare the results obtained with machine learning to the results of variability search by visual inspection of light curves, which is the most reliable, but labor-intensive way of identifying variable objects (hence the relatively small size of our training sample). The sample consists of 30265 sources with high-quality (percentage of good measurements Pgood $\geqslant 98$; see Section 4.1 in Szymanski 2005) light curves each having 262 to 268 points; among them 168 variable sources of various types. This data set is further randomly split into subsets multiple times in order to find the most promising variable object selection technique as described in Section 3. The full LMC_SC20 data set from Sokolovsky et al. (2017) is used to train the selected best classifier before applying it to a new data set that was not previously searched for variability by us. The new data set consists of 31798 OGLE-II PSF I-band light curves from the adjacent field LMC_SC19 selected by applying the same quality cut (Pgood $\geqslant 98$ resulting in 262–268 light curve points). Three variable objects (and 893 non-variable ones) located in the overlapping sky region present both in LMC_SC19 and LMC_SC20 data sets.

Table 1 presents the distribution of variability types available in the training (LMC_SC20) data set and recovered from the blind test data set (LMC_SC19, Section 4.3). We adopted a published classification of variable objects whenever possible: eclipsing binaries from Wyrzykowski et al. (2003), Graczyk et al. (2011), Kim et al. (2014), red giant variables from Soszynski et al. (2005), Fraser, Hawley & Cook (2008), Soszyński et al. (2009b), Spano et al. (2011), RR Lyrae variables from Soszynski et al. (2003), Cepheids from Udalski et al. (1999), candidate Be stars from Sabogal et al. (2005), QSO candidates from Eyer (2002), Kim et al. (2012), Kozłowski et al. (2013), δ Scuti stars from Poleski et al. (2010). Soszynski et al. (2004) classified 1546 periodic red giants in the LMC as candidate ellipsoidal variables following the suggestion by Wood et al. (1999), Wood (2000) that one of the five period-luminosity sequences observed in LMC red giants may represent binary systems rather than a mode of pulsations. Considering that *i)* physical interpretation of this sequence as binary systems is not unambiguous; *ii)* in practice, the light curve shapes of these objects are indistinguishable from light curves of some semiregular variables; *iii)* eclipsing variables with periods > 10 d showing strong ellipsoidal variations often have bluer colors than the candidate ellipsoidal variables with no eclipses; for the purpose of this work we group the candidate ellipsoidal variables with other variable red giants in Table 1. The lists of candiate Be stars of Sabogal et al. (2005) and QSO candidates of Eyer (2002) have 99 common objects 11 of which are among the variable objects in our data sets. In Table 1 we combine them under the label "blue irregular variables".

While the discussion below is based on the OGLE-II data, we also performed a similar analysis of two other data sets investigated by Sokolovsky et al. (2017) that were collected with different telescopes and processed using different source extraction and photometry software: Kr (Lapukhin, Veselkov & Zubareva 2013, 2016) and $TF1$ (Burdanov et al. 2016, Popov et al. 2015, Burdanov, Krushinsky & Popov 2014). The results obtained with Kr and $TF1$ are consistent with the ones presented in Sections 4 and 5. The main focus in our investigation was set on the OGLE-II LMC_SC20 data set as many other OGLE-II light curves are readily available for variability search with the technique described here.

Table 1. Types of variable objects in the blind test (LMC_SC19) and training (LMC_SC20) data sets.

Type	LMC_SC19	LMC_SC20
eclipsing binaries	36	54
variable red giants (L/M/SR/ELL)	54	52
RR Lyrae-type variables	56	26
Cepheids (classical and Type II)	17	20
blue irregular variables (GCAS/BE/QSO)	22	13
δ Scuti stars	1	3
total	186	168

2.2 Sources of bias in the training sample

The training sample may be biased as the list of known variables in the LMC_SC20 data set may not be exhaustive (and therefore some variable objects may be incorrectly labeled as non-variable). We try to minimize this by conducting our own variability search (used also in our previous work, Sokolovsky et al. 2017) based on visual inspection of light curves instead of relying on published lists of variables (Sec. 2.1), however this is still likely an approximation to the complete list of (detectable) variables in the used set of light curves.

Another source of bias is the limited size of our training sample (Table 1) that does not nearly represent all variability types and all possible variations in amplitude and period or variability time scale within each type. This translates in a non-trivial way to an incomplete coverage of the variability features (introduced in Sec. 2.3) parameter space occupied by variable objects (see the discussion of learning curves in Sec. 4.2.1). The severity of this problem is hard to quantify a priori. Positive results of variability search in the unseen data described in Sec. 4.3 indicate that this is not a critical issue. This may partly be attributed to the fact that (while relying on the assumption that variable objects are rare) the section of the variability features parameter space occupied by non-variable objects should be sampled well with ~ 30000 example non-variable sources in the training set.

2.3 Variability features

We initially considered 24 features listed in Table 2 (a detailed discussion of these features is presented by Sokolovsky et al. 2017). Many of them are highly correlated (see Figure 1)[2], in fact some represent the same quantity computed using different weighting

[2] *biokit* Python package was used to generate the plot `https://github.com/biokit/biokit`

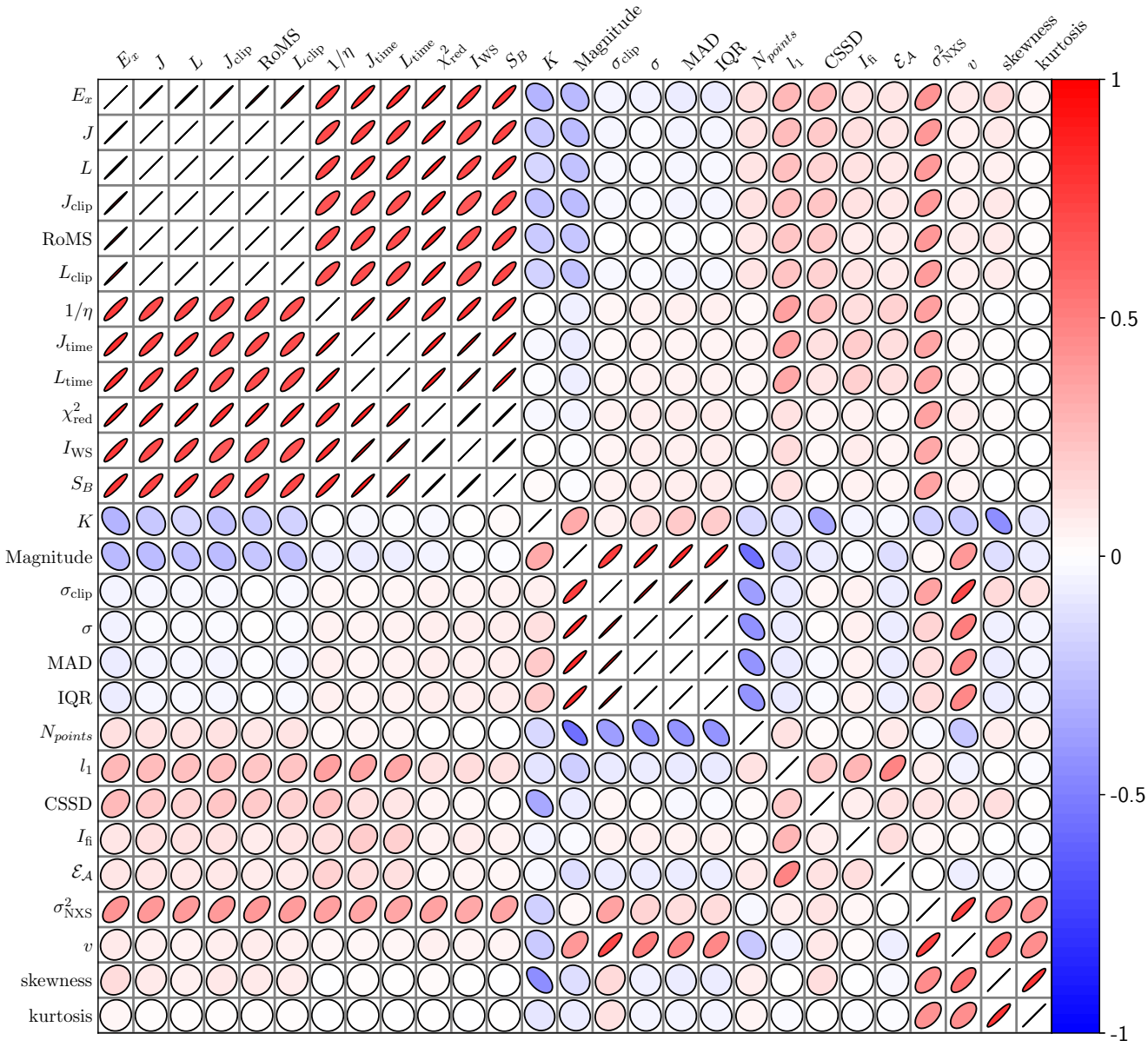

Figure 1. Correlation between the light curve features. Color and orientation of each ellipse represent the sign (red and rotated 45 degrees clockwise from vertical - positive while blue and rotated counterclockwise - negative) and eccentricity with color depth code the value of the Pearson correlation coefficient between the corresponding features (see the color bar). A nearly circular shape and white color indicate close to zero correlation between a pair of features while a narrow red (blue) ellipse indicates high positive (negative) correlation between features.

or clipping schemes (σ-$\sigma_{\rm clip}$, Stetson's K-kurtosis, J-$J_{\rm time}$-$J_{\rm clip}$, L-$L_{\rm time}$-$L_{\rm clip}$; see Section A). We dropped the features $\sigma_{\rm clip}$, L, $L_{\rm clip}$, $J_{\rm clip}$, MAD, $L_{\rm time}$ which are highly correlated to other corresponding features with $r > 0.995$ (Figure 1). The choice which feature to keep among a few highly correlated ones was done in a quasi-random fashion. When processing a really large set of light curves, it would be wise to consider computational costs of features and keep the one that requires less time to calculate. We checked that the number of remaining features is reasonable using *Principal Component Analysis* (PCA; Pearson 1901). Most (95%) of the variance in features can be explained by 10 PCA-components (Figure 2). This suggests that at least 10 original features are needed to describe most of the variance in the data. We also dropped the features *CSSD* and l_1 that in the implementation of Sokolovsky et al. (2017) appeared to be less-informative for variability search. We tried to log-transform positive features (such as σ or *IQR*) to make their distribution closer to the normal but it has not resulted in higher performance for any of the tested algorithms. Ensemble trees methods used in our work, Random Forests (*RF*; Section 3.2.4) and Stochastic Gradient Boosting (*SGB*; Section 3.2.5), are invariant to one-to-one transformations of the input feature data. Our pre-processing procedure includes scaling features by centering and standardization for all methods except *RF* and *SGB*. We note that to prevent overestimation of classification performance, the data pre-processing and the feature selection should be done in a way that prevents any information leakage from the sample used to

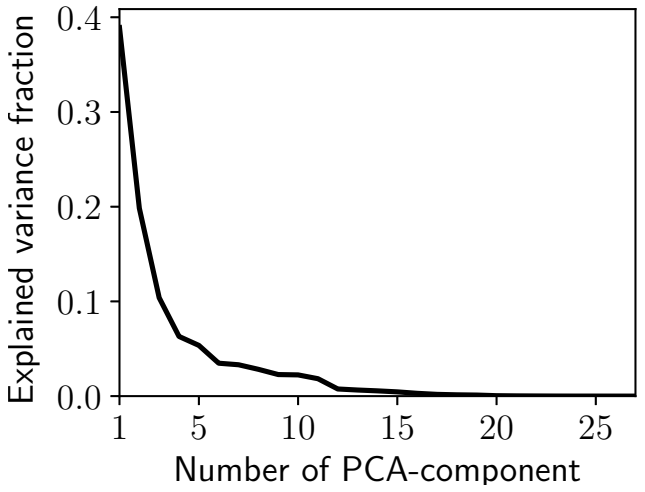

Figure 2. Fractional variance explained by each of the PCA-component. Also known as *scree plot* (Cattell 1966). Most of the variance can be explained by 10 PCA components confirming that many light curve features listed in Table 2 are correlated (see also Figure 1 and Section 4.2).

Table 2. Light curve features (variability indices). Features correlated with other features with $r > 0.995$ for the LMC_SC20 data set (and excluded from the final analysis) are marked with italic

Index	Reference
weighted standard deviation – σ	Kolesnikova et al. (2008)
clipped σ – $\sigma_{\rm clip}$	Section A1
median abs. deviation – MAD	Zhang et al. (2016)
interquartile range – IQR	Sokolovsky et al. (2017)
reduced χ^2 statistic – $\chi^2_{\rm red}$	de Diego (2010)
robust median statistic – RoMS	Rose & Hintz (2007)
norm. excess variance – $\sigma^2_{\rm NXS}$	Nandra et al. (1997)
norm. peak-to-peak amp. – v	Sokolovsky et al. (2009)
autocorrelation – l_1	Kim et al. (2011)
inv. von Neumann ratio – $1/\eta$	Shin, Sekora & Byun (2009)
Welch-Stetson index – $I_{\rm WS}$	Welch & Stetson (1993)
flux-independent index – $I_{\rm fi}$	Ferreira Lopes et al. (2015)
Stetson's J index	Stetson (1996)
time-weighted Stetson's $J_{\rm time}$	Fruth et al. (2012)
clipped Stetson's $J_{\rm clip}$	Section A2
Stetson's L index	Stetson (1996)
time-weighted Stetson's $L_{\rm time}$	Fruth et al. (2012)
clipped Stetson's $L_{\rm clip}$	Section A2
consec. same-sign dev. – CSSD	Shin, Sekora & Byun (2009)
S_B statistic	Figuera Jaimes et al. (2013)
excursions – E_x	Parks et al. (2014)
excess Abbe value – $\mathscr{E}_{\mathscr{A}}$	Mowlavi (2014)
Stetson's K index	Stetson (1996)
kurtosis	Friedrich, Koenig & Wicenec (1997)
skewness	Friedrich, Koenig & Wicenec (1997)

evaluate performance to the one used to build the classifier (e.g. Smialowski, Frishman & Kramer 2010).

3 VARIABLE STARS IDENTIFICATION AS CLASSIFICATION PROBLEM

We tackle the problem of variable stars identifications as classification problem. Classification is a *supervised* learning problem where one has a set of objects X, a set of responses Y and some unknown dependence $f : X \mapsto Y$ (*target function*, e.g. Hastie, Tibshirani & Friedman 2001, Vorontsov 2013). The problem is to find (*learn*) an algorithm (decision function) $f^\star$ that approximates target function f for all X given only the subsample of all objects - $X_{train,i}$ with known responses $Y_{train,i}$ (called the *training sample*). Depending on the nature of Y, the problem can be formulated as regression ($Y = \mathbb{R}$), binary ($Y = \{0,1\}$) or K-class classification ($Y = \{0,1,...,K\}$). The objects are characterized by a set of *features* $\phi_j : X \mapsto D_j$, where D_j could be $\{0,1\}$ (binary feature), $|D_j| < \infty$ (nominal or ordinal feature if finite D_j could be ordered) or $D_j = \mathbb{R}$ (qualitative feature). The choice of features that capture properties related to object's class is crucial for good classification. The chosen set of features determines the maximum classification performance that could be achieved for a given problem.[3].

When building a classifier $f^\star$ that provides high quality predictions on new unclassified data and given a set of features (that constrains the maximum achievable quality of classification) one has to decide what family of algorithms $f^\star(\theta)$, parametrized by some parameter vector θ, to use. The main concern is that if the chosen algorithm is not flexible enough to approximate f then it cannot approach the highest possible performance of classification on new data no matter how large training sample is used to learn the parameters θ. In this case, the algorithm prediction has high bias and it is said that the algorithm is *underfitting*. If the algorithm is too complex ($f^\star$ has many unconstrained parameters θ) it can spend some of its degrees of freedom on learning noisy patterns specific to a given finite training sample. Thus algorithm's predictions on new data become unstable, sensitive to small changes in training data. In that case, the predictions have high dispersion and it is said that the algorithm is *overfitting*[4]. In both cases the algorithm's ability to generalize (that is to provide good quality classification of new data) becomes low. This trade-off governed by the algorithm's complexity is called the *Bias-Variance trade-off* (Hastie, Tibshirani & Friedman 2001).

One can constrain the complexity of $f^\star$ or tune some other high-level algorithm property (e.g. algorithm behavior during training) to reduce the dependence of its predictions on the used finite training sample (i.e. algorithm dispersion[5]). To see how much the algorithm is overfitting one has to apply it to some classified data that are not part of the training sample. Parameters that determine the algorithm performance on new data but cannot be learned using training data alone are called *hyperparameters* (HP).

In summary, each algorithm has a set of conventional parameters θ (e.g. coefficients of features in regression, Section 3.2.2, weights of neurons in *NN*, Section 3.2.6) that are learned from the training sample and hyperparameters that have to be set before training (e.g. number of trees in *RF*, Sec. 3.2.4 or number of

[3] Classifier with such performance is called *Bayes classifier* (Hastie, Tibshirani & Friedman 2001) and its (maximum achievable) error rate is called *Bayesian rate*. It should be noted that it is quite theoretical construction because it uses generally unknown posterior probability of class membership $P(Y|X)$ for making its predictions.

[4] Overfitting could also be the result of training sample being unrepresentative of the parent population, e.g. when training data set is small or has wrongly classified objects. In general, any algorithm that has high performance on data used to train it but lower performance on new data is said to overfit.

[5] High-bias algorithms could also have significant dispersion, e.g. multivariate linear regression with highly correlated independent variables (features).

hidden layers or number of neurons in each hidden layer in *NN*, Section 3.2.6). The hyperparameters include not only the complexity parameters (capacity to learn, e.g. depth of a decision tree, number of hidden layers and neurons in each layer in neural network, number of basis learners in ensemble, value of regularization that penalizes too complex models), but also parameters that control the process of algorithm training, e.g. speed of learning (the learning rate in gradient descent methods of learning neural networks). The optimal set of hyperparameters for a given algorithm largely depends on the data set and might differ even between training samples of different sizes.

3.1 Performance metric

To decide which variability detection technique works best, we need to define what exactly do we mean by "best", in other words – adopt an appropriate performance metric. As we deal with a highly imbalanced data set (non-variable stars outnumber variable ones by a factor of ~ 100, Section 2.1), *accuracy* defined as the ratio of correct predictions to the total number of cases evaluated, despite being most intuitive performance metric is not a proper measure of classification algorithm performance. A high accuracy score could be obtained by just labeling all target objects with the majority class (Kononenko & Bratko 1991, Valverde-Albacete & Peláez-Moreno 2014). To avoid this, one considers *Precision*, $P = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP})$ and *Recall*, $R = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$, as well as their harmonic mean known as F_1-score

$$F_1 = 2PR/(P+R),$$

where TP is the number of true positives (i.e. true variables classified as variables), FP is the number of false positives (non-variables classified as variables) and FN is the number of false negatives (true variables classified as non-variables; Rijsbergen 1974).

Suppose we test a classifier using it to select candidate variables from a set of light curves for which we already know the right answer - which light curve shows variability and which does not. Then P is the probability that a randomly chosen object from the list of candidates is a true variable while R is the probability that a randomly chosen true variable is in the list of candidates. In reality there is a trade-off between high values of R and P, i.e. recovery of all positive objects (true variables) and contamination by false positives (objects that algorithm wrongly classifies as variables). F_1 is a useful compromise: it has a high value when both R and P are high, that is when the classifier does not miss many true variables *and* the majority of objects classified as variables are actual variables.

Most classification algorithms instead of class labels (e.g. variable/non-variable) return probabilities p_i of i-th object representing a certain class[6]. To assign class membership to objects being classified one has to choose a threshold value $p_{\mathrm{threshold}}$ such that objects with probability p_i of belonging to the class Y are assigned to that class if $p_i > p_{\mathrm{threshold}}$. P, R and F_1 do depend on the adopted threshold value. This can be utilized if the cost of false positives and false negatives is different. For example, if when visually inspecting a list of candidate variables we are willing to look at ten false candidates for every true variable, then for us the cost of false positives is ten times lower than the cost of false negatives. If the cost of false positives is high (e.g. if we want to obtain a list of candidates with the majority of the objects representing true variability) then P is a suitable performance metric, if the cost of false negatives is high (e.g. if we want to recover as many true variables as possible) than R can be used. Alternatively, one may use

$$F_\beta = (1+\beta^2)RP/(R+\beta^2 P),$$

a score that attaches β times as much importance to R as P (Rijsbergen 1974). In case of equal costs F_1 works best.

To characterize the model's performance over all possible thresholds (i.e. under different values of FP/FN cost ratios), Area Under *ROC* Curve (*AUC*; Fawcett 2006) may be used as a performance metric. ROC-curve is a plot of R against *False Positive Rate* $FPR = \mathrm{FP}/(\mathrm{FP}+\mathrm{TN})$, where TN is the number of true negatives (true non-variable stars correctly classified as non-variables). For binary classification *AUC* is the probability that given one positive and one negative example at random, the classifier rank the positive example above the negative one.

As shown by Saito & Rehmsmeier (2015) in case of highly imbalanced data *AUC* weakly depends on the algorithm performance (mainly because it considers the number of TN) and other metrics (such as Area Under Precision-Recall Curve - *AUPRC*) should be used instead. To compare methods in similar to Sokolovsky et al. (2017) manner we decided to search hyperparameters that maximize F_1-score using the default threshold value of 0.5.

3.2 Classifiers

We tried several classifiers: Logistic Regression (*LR*), Support Vector Machines with Radial Basis Functions (*SVM*), k Nearest Neighbors (*kNN*), Neural Nets (*NN*), Random Forests (*RF*) and Stochastic Gradient Boosting classifier (*SGB*). These algorithms use different assumptions about classes and target function and use different heuristics and methods to tackle the problem of classification. We use `scikit-learn` Python package (Pedregosa et al. 2011) implementation of *SVM*, *RF*, *kNN*, `XGBoost`[7] (Chen & Guestrin 2016) implementation of *SGB* and `Keras`[8] library for *NN*–classification. In the following we briefly describe these classifiers and their hyperparameters. More information may be found in the official documentation of *scikit-learn*, *XGBoost* and *Keras*.

3.2.1 *k Nearest Neighbors (kNN)*

kNN method is based on the hypothesis that similar objects usually share the same class. The notion of "similarity" is defined in terms of distance between objects in feature space. The object class predicted by *kNN* is the class chosen by the majority of k closest neighbors. Despite being quite simple (no need to fit model or learn anything) this method is very effective especially in situation where the hypothesis holds and number of samples is relatively high. The algorithm is nonparametric i.e. it's decision surface (boundary between classes in features space) can be arbitrary complex and approximate any underlying dependence f (Section 3) given enough training data. The optimized hyperparameters are number of neighbors k and *weights* - the type of weighting being used. We tried

[6] Actually they return some proxy of probability. To make it probability one has to *calibrate* (Zadrozny & Elkan 2002) classifier by comparing predicted and true frequency of classes for some independent data set.

[7] https://xgboost.readthedocs.io/en/latest/

[8] https://keras.io/

uniform weights and weights inversely proportional to euclidean distance to neighbor[9].

3.2.2 Logistic Regression (LR)

LR is a generalized regression model used in cases of binary (or categorical, in general) response variable. It differs from the standard linear regression with continuous response by the use of *link function* that transforms linear combinations of features to binary response variable. *LR* models the $\text{logit}(p) = \log(p/(1-p))$ of posterior class probability membership p as a linear combination of features. Setting some threshold value of p allows one to make the response binary. We optimized two hyperparameters: C that defines the level of regularization used (default $L2$-regularization was used which penalizes complexity by adding term to objective function being minimized that consists of sum of squares of feature coefficients) and relative weights of classes.

3.2.3 Support Vector Machine (SVM)

Linear *SVM* is searching for the *optimal separating hyperplane* in the features space that separates classes best in terms of maximum distance from closest objects of both classes to the hyperplane (Vapnik 1996) thus maximizing the *margin* between classes. This hyperplane is defined by a (usually) small number of objects in feature space that are close to decision surface (*support vectors*) and that are the hardest to classify. For classification problems with classes that can't be separated using linear surface the use of special kernels reduces the problem to finding the optimal separating hyperplane in enlarged (even infinite-dimensional for some kernels) transformed feature space without explicitly transforming features (Boser, Guyon & Vapnik 1992). We optimized: kernel type - linear (*linear*), polynomial (*poly*) and Radial Basis Function kernel (rbf), degree of polynomial kernel for kernel *poly*, C - "soft margin" regularization penalty parameter (it determines the relative influence of wrongly classified points - points on the "wrong" side of the optimal hyperplane), *gamma* - kernel coefficient and the relative weights of classes.

3.2.4 Random Forest (RF)

RF is an ensemble method. Ensemble methods use the predictions of several weak learners[10] and combines them at once or sequentially to make more efficient predictions than individual learners. *RF* uses the *bagging* (bootstrap aggregation Brieman 1996) method that combines many weak learners with high variance (which are overfitting, e.g. too flexible/complex themselves, Section 3) trained on bootstrap samples[11] of training data thus reducing variance of the final estimator. It usually uses a deep decision tree (tree with many branches) as weak learner. An example decision tree classifier is presented in Figure 3. We use a shallow tree with an easy-to visualize structure. Hyperparameters of this tree were also optimized for maximum performance as measured by $F_1 = 0.69$ (see

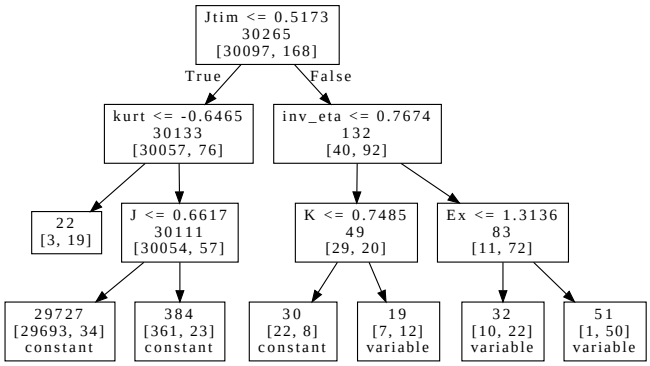

Figure 3. An example decision tree for LMC_SC20 data set. Nodes of the tree show the cuts on variability feature (Table 2) values used to make a decision at each node. The numbers in each node are the number of all objects considered in this node, the number of non-variable and variable objects.

3.3 for details of measuring F_1). *RF* also uses the idea of *random subspace selection* (Ho 1998) (also known as attribute or feature bagging) that is similar to bagging but instead of subsampling training objects it consists of using random subsets of features for creating and growing individual decision trees. This prevents *RF* from being focused on small number of highly informative features that could loose their predictive power on unseen data.

The optimized hyperparameters are: $n\_estimators$ - the number of decision trees to use in the ensemble, $max\_features$ - the number of features to use in search of best split of the node, max_depth - the maximum depth of the individual trees, $min\_samples\_split$ - the minimum number of samples in the node of the decision tree required to make split, $min\_samples\_leaf$ - the minimum number of samples required to be in the leaf (that is terminal) node of each tree and relative weights of classes.

3.2.5 Stochastic Gradient Boosting (SGB)

The idea of *boosting* (Schapire 1990) is to incrementally built classifier by re-weighting training examples giving more weight to the misclassified objects. Boosting combines weak learners with high bias (which are underfitting, i.e. not flexible/complex enough to approximate underlying relation themselves) sequentially (shallow decision trees in our work) reducing bias of the final estimator. Gradient boosting treats boosting as optimization algorithm and generalizes the boosting method to arbitrary differentiable objective functions (Friedman 2001, Mason et al. 1999). To prevent overfitting, boosting can be combined with bagging and random subset selection (*stochastic gradient boosting*) by using only a subsample of training data on each iteration (Friedman 2002) and a subset of features to decide which should be used for splitting tree node or creating another tree. We optimized the following hyperparameters: $learning\_rate$ - the scale value for the prediction of each tree (shrinkage); model complexity parameters: max_depth - the maximum depth of the individual trees, *gamma* - the minimum objective function reduction required to make a further partition on a leaf node of the tree, min_child_weigth - the minimum sum of weights of all examples in a child of split required to make further splits, max_delta_step - the maximum delta step allowed for each tree's weight estimation to be; parameters that makes predictions to be more robust to noise: $subsample$ - subsample ratio of the training instances, that is fraction of the training data set drawn at

[9] We also experimented with some non-euclidean metrics supported in *scikit-learn* Python package, e.g. *chebyshev*, *manhattan* distances, but their usage resulted in degraded performance.

[10] Weak learner is an algorithm performing not much better than random guessing

[11] Bootstrap sample is a sample of the same size as the original one drawn with replacement from it.

random without replacement at each iteration, *col_sample_bytree* - the subsample ratio of columns (features) when constructing each tree, *col_sample_bylevel* - subsample ratio of columns (features) for each split, in each level, *scale_pos_weigth* - relative weights of classes, parameter that controls the model complexity through regularization: *reg_lambda* - $L2$-regularization term on weights. The parameter *n_estimators* - number of decision trees in model - was determined as iteration since which performance measure being used (F_1, but see 3.3 for details) have not improved during the following 30 iterations (*early stopping* rule).

3.2.6 Neural Net (NN)

We used a fully connected neural network topology and checked one and two hidden layers. Though we did not expect complex decision surface geometry for our problem, we decided to try two hidden layers but include regularization by means of constrains on neurons weights and *dropout*[12] technique to prevent overfitting. The input and the hidden layer(s) both had *rectified linear units* (Nair & Hinton 2010) activation functions (Haykin 1999) and the output layer had a sigmoid activation function for probabilistic predictions. The neuron weights were initialized using the normal distribution. The weights update used *Stochastic Gradient Decent* (*SGD*) method on subsets (minibatches) of training data. We optimized the following hyperparameters: *i)* network architecture parameters – number of hidden layers and neurons in each hidden layer (size of the input layer was determined by the number of features); *ii)* regularization parameters – the value of the dropout at each layer (except output) and the maximum sum of weights for each layer; *iii)* parameters of *SGD* (not specific to *NN*) – the initial learning rate *lr*, the decay rate *decay*, rate of decreasing learning rate (learning rate schedule), *momentum* - parameter that determines the "inertia" of neurons weights update with *SGD*, *batch_size* - number of data points to use for calculating updates of neurons weights; *iv)* *class_weight* - relative weights of classes. *nb_epochs* - number of epochs, that is the number of times all training data were used for updating network weights - was determined by the early stopping rule.

3.3 Hyperarameters tuning

Each algorithm's hyperparameters (listed in Table 3) were tuned using the *Tree of Parzen Estimators (TPE)* algorithm (Bergstra et al. 2011) implemented in *hyperopt*[13]. TPE is a Bayesian approach to optimization that models conditional probability $p(\lambda|c)$, where λ - the values of hyperparameters, c - some loss function (criterion one desires to minimize) by two Gaussian Mixture Models. One ($l(\lambda)$) is fitted to the hyperparameters values associated with the smallest (best) values of loss function and other ($g(\lambda)$) is fitted to the hyperparameter values for all other values of loss function. New candidates are considered the ones with the lowest value of g/l.

As noted in Section 3, hyperparameters shouldn't be learned from training data. To prevent overfitting we used 4-fold *Cross-Validation* (Hastie, Tibshirani & Friedman 2001) (*CV*) during hyperparameters search. Each trial with proposed by *TPE* values of hyperparameters data was split into 4 folds. The split was made by preserving the proportion of classes in both samples ("stratified" split). Three of four folds were combined into training sample where the classifier with trial hyperparameters values was fitted and one fold became the evaluation sample that was used to evaluate the F_1-score. This combination of folds in training/evaluation samples was done 4 times in such a way that each of the 4 folds was used as evaluation sample once. To properly combine individual F_1-scores of 4 splits to one value we first found TP and FP for each split, sum them and then calculate F_1-score using obtained values. This pro-

Table 3. Variability selection algorithms and their hyperparameter values that maximize the F_1^{CV} for the test data set LMC_SC20.

Algorithm	Secion	Hyperparameter	Value	F_1^{CV}
		Machine learning algorithms		
kNN	3.2.1	*n_neighbors*	6	0.68
		weights	*distance*	
LR	3.2.2	C	50.78	0.68
		class_weight	2.65	
SVM	3.2.3	*kernel*	*rbf*	0.80
		C	25.05	
		gamma	0.017	
		class_weight	2.93	
RF	3.2.4	*n_estimators*	1400	0.77
		max_depth	16	
		max_features	5	
		min_samples_split	16	
		min_samples_leaf	2	
		class_weight	28	
SGB	3.2.5	*learning_rate*	0.085	0.79
		max_depth	6	
		min_child_weigth	2.36	
		subsample	0.44	
		colsample_bytree	0.35	
		colsample_bylevel	0.76	
		gamma	4.16	
		scale_pos_weight	4.09	
		max_delta_step	2	
		reg_lambda	0.09	
NN	3.2.6	num. of hidden layers	1	0.81
		num. neurons in hidden layer	13	
		dropout on input layer	0.00	
		dropout on hidden layer	0.17	
		sum of weights, input layer	9.04	
		sum of weights, hidden layer	5.62	
		learning_rate	0.20	
		decay_rate	0.001	
		momentum	0.95	
		class_weight	2.03	
		batch_size	1024	
		Traditional methods		
J_{time}^a		selection threshold	5.3σ	0.59
L^b		selection threshold	6.5σ	0.53
PCA_1^c		selection threshold	7.4σ	0.49
median[d]				0.43

[a] J_{time} is the variability index (Table 2) with the highest F_1-score for LMC_SC20, but some short-period variables cannot be recovered with this index. [b] L index has the highest F_1-score in this data set among the indices that may recover all known variables. [c] Admixture coefficient of the first PCA component used as a composite variability index (a linear combination of individual indices, see Sokolovsky et al. 2017 for details). [d] The last line in the table presents the median F_1-score of all variability indices compared by Sokolovsky et al. (2017).

[12] Dropout is a regularization method for *NN* where a randomly selected fraction of neurons do not participate in weights update. That helps to avoid overfitting as shown by (Srivastava et al. 2014)

[13] `http://hyperopt.github.io/hyperopt/`

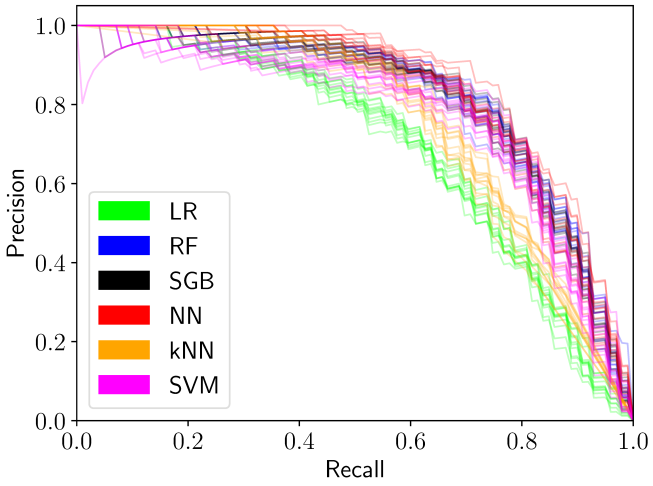

Figure 4. Precision-Recall curves for 6 algorithms with 12 different splits of data set into folds during CV. Nearly identical performance is apparent for the four best algorithms.

cedure (unlike direct averaging of F_1-scores of each split) is nearly free of bias due to highly imbalanced data sets (Forman & Scholz 2010). Such value F_1^{CV} (cross-validation estimate of F_1-score) is an estimate of the algorithm prediction performance (as measured by F_1 metric) on the independent (unseen) data set. This is the quantity that was subject to maximization using TPE algorithm.

We did a couple of thousands iterations of *TPE* on classifiers that have many hyperparameters and several hundreds on the rest. It takes a couple of days of computing time on a Core i5 desktop to find the best hyperparameters for the *RF*, *SGB* and *NN* algorithms (the computing time was less for the other algorithms). For *NN* and *SGB* we first fixed learning rate on some default values (0.2 and 0.1) and searched for best hyperparameters. Then secondary search was made with the found hyperparameters fixed but now searching for the best learning rate. For hyperparameters that were set and not fitted, we tried a few other choices manually, specifically: *kNN* with different distance metrics, *NN* with more hidden layers than 2. We also tried $L1$-regularization for *LR* (with poor performance that could be attributed to the features correlation).

4 RESULTS AND DISCUSSION

4.1 Algorithms performance comparison

The best values of F_1^{CV} obtained for each algorithm along with corresponding values of tuned hyperparameters are presented in Table 3. As expected for small training data set the performance of classifiers depends on the way the data are split in folds during CV. Figure 4 shows the *Precision-Recall* curves for each of the 6 algorithms. The hyperparameters used are the best for one (common to all algorithms) of the CV splits, that was the result of the fixed random seed used. The different curves of the same color show the effect of different CV splits on each algorithm performance. Clearly *SVM*, *RF*, *GB* and *NN* performance is nearly equal.

LR showed the worst performance as indicated by its Precision-Recall curve in Figure 4 and the low value of F_1^{CV}. Note however, that the algorithm's F_1-score (as measured by F_1^{CV}) is still above the values reached by the traditional selection based on individual variability indices (Table 3). Low performance can be understood as the *LR* is a linear model that separates classes with a linear decision surface that could result in high bias in case of classes that aren't linearly separable (e.g. when nonlinear features combinations predict data better).

kNN also showed lower performance compared to the other classifiers that could result from the presence of class outliers (training objects surrounded by objects of different class in the features space), that is especially pronounced in case of highly imbalanced data sets used. Moreover, large number of features promote the *curse of dimensionality* (Hughes 1968) - a phenomenon that in high dimensional space, all vectors become remote from a given vector equally far (Beyer et al. 1999). One has to mention that the classes of variable and non-variable stars are very inhomogeneous. The class of "variables" includes objects of various types (ecipsing binaries, pulsating variables) changing their brightness with different amplitudes and on various timescales. The class of "non-variables" includes non-variable objects with properly measured brightness as well as the few objects with corrupted measurements that have high values of the variability indexes but do not pass visual inspection of their light curves. Thus, the "similarity hypothesis" (see 3.2.1) may fail in this case. Finally, inclusion of some noisy features could also lead to degraded performance. We tested the latter possibility by adding an extra data prepossessing step: selecting K_{best} best features as measured by *ANOVA* F-value between features and class (Guyon & Elisseeff 2003, Nadir, Othman & Ahmed 2014) and found best $K_{best} = 16$ but only with marginal (0.002) gain in F_1^{CV}.

Formally, the highest F_1^{CV} was obtained by *NN*. The best *NN* architecture consists of a fully-connected network structure with one input layer with 18 neurons (that is determined by number of features used), one hidden layer with 13 neurons (both with Rectified Linear Units activation functions) and output layer with sigmoid activation function. No dropout and relaxed weights constrains are preferred by the best model for the input layer.

We also compare classifiers on Kr and $TF1$ data sets described in Sokolovsky et al. (2017). After excluding the most correlated features (with $r > 0.995$) we were left with 20 and 24 features, respectively. The performance of all considered algorithms on the first data set is nearly equal ($F_1^{CV} = 0.88$ for *kNN*, 0.90 for *LR*, *RF* and *SVM*, 0.91 for *SGB* and 0.92 for *NN*) and on second data set the relative performance is about the same as for LMC_SC20 data set, but with lower overall level (resulting from a larger number of corrupted measurements in this data set) with the best $F_1^{CV} \approxeq 0.78$ achieved by the *NN* classifier.

4.2 Testing further modifications to the algorithms

4.2.1 Learning curves and feature pre-conditioning for *LR*

To explore the possibilities of further increasing the algorithms performance we first considered *learning curves* (Raschka 2015) – the dependence of classifier performance (measured by F_1-score) on the amount of training data used (Figure 5). For all the considered classifiers except *LR*, the learning curves show that the F_1-score on the training data set is higher than the one obtained on the independent validation data set and the later is still increasing at maximal training sample size. This indicates that using a larger training set should further increase performance of these algorithms. On the other hand, *LR* shows comparable relatively low F_1 values on training and validation sets. These two characteristic types of learning curves correspond respectively to high-variance (in our case – *kNN*, *SVM*, *RF*, *SGB*, *NN*) and high-bias (*LR*) algorithms (see Section 3) for the used data set.

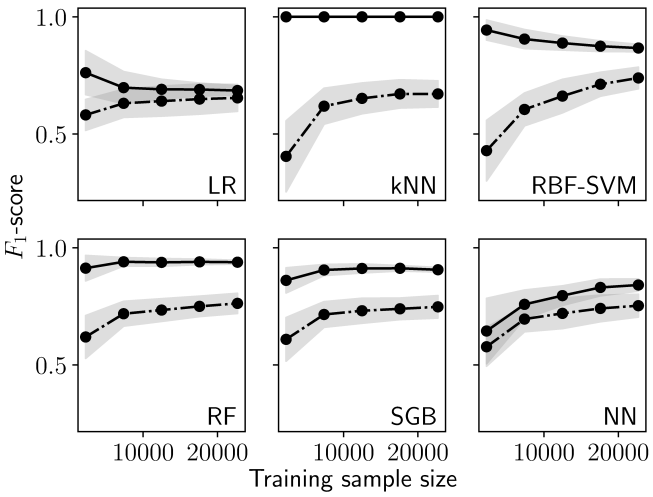

Figure 5. Learning curves for the LMC_SC20 data set. Solid lines denote F_1-score on training sample, dot-dashed - cross-validation estimate of F_1-score on unseen data. Shaded regions show uncertainty estimated using 40 different splits of data set in training and validation sample. Two typical learning curve shapes are evident. *LR* reveals comparable relatively low F_1 values on both training and validation sets that remain constant with growing training sample size. This is the sign of a bias of the classifier. The learning curves of the other classifiers show F_1 on the training data set higher than F_1 obtained on the independent validation data set (i.e. classifiers are overfitting) that is increasing with training sample size (that implies absence of a bias).

To improve the performance of *LR* we tried to reduce its bias by accounting for non-linear features interactions. First, we excluded not only the highly correlated ($r > 0.995$, Section 2.3) features but also the features that show low $F_{1\ \max}$ in the original paper by Sokolovsky et al. (2017) – $\sigma^2_{\rm NXS}$ and ν (Table 2) and the features with lowest rank (as measured by feature coefficients in regression) that were lowering the maximum achievable CV estimate of F_1-score using *Recursive Feature Elimination* method (kurtosis and $\sigma^2_{\rm NXS}$ again). Then instead of raw features we used their second order polynomial combinations and several first PCA-components of raw features (the number was determined by *TPE*-search optimizing F_1^{CV}). This resulted in performance ($F_1^{CV} = 0.78$) comparable to that of other classifiers. We conclude that *LR* may work as well as the other considered algorithms, but requires a special preparation of the input data.

4.2.2 **Exclude uninformative features**

We tried to exclude two features (kurtosis and skewness) that have the least relation to variability class (as reported by *ANOVA F*-value between label/feature) from the input of the best classifier, *NN*, to check if the removal of these most noisy features increases performance of the *NN* classifier. After excluding the features, we repeated the TPE search for optimal hyperparametes. The resulting *NN* has marginally ($\Delta F_1^{CV} \approx 0.005$) degraded performance and simpler architecture (11 instead of 13 neurons on hidden layer, stronger regularization via dropouts and weight constrains). Excluding kurtosis and skewness from the input of the third-best *SGB* classifier also results in the slightly decreased performance ($\Delta F_1^{CV} \approx 0.01$). This suggests that even the least-important of the considered features contain some minimal useful information that can be taken into account by the best classifiers *NN* and *SGB*.

To test how many features are necessary to obtain high F_1-scores we used *SGB* method as it is pretty straightforward to get the importance of features using this algorithm (Hastie, Tibshirani & Friedman 2001). Although we used hyperparameters tuned for 18 features after successively excluding the least important features, we found that with 9 features (J, $J_{\rm time}$, I, *Magnitude*, IQR, $1/\eta$, kurtosis, skewness, I_{sgn}) we can still obtain F_1^{CV} as high as 0.77 and using only 3 (J, kurtosis, I) results in F_1^{CV} = 0.62.

We also tried to use several PCA-components (Figure 2) as features instead of the original features listed in Table 2. The expectation was that using several first PCA-components we may reduce the noise introduced by a number of (nearly) uninformative features. For this test we used *RF* classifier and added the number of used PCA-components to the list of optimized hyperparameters (see Table 3). We allowed *max_features* to vary from 3 to 5 and number of PCA-components from 5 to 18. The best value of F_1^{CV} was 0.75 with 18 PCA-components and *max_features* = 4. Thus the classifier doing its best when using essentially all features. The degraded performance could be attributed to PCA keeping only linear combinations of features.

4.2.3 **Ensemble combining multiple classifiers**

We have tried to combine individual algorithms predictions using different ensembling methods. To approximate the case of using unseen data we used different random seed when splitting the sample on train/test splits during cross-validation estimation of F_1-score. This results in slightly worse performance of the algorithms that used HP optimized with different CV-splits.

First we used *Hard Voting* of individual algorithms when the class that obtains the majority votes of individual classifiers is chosen. We used as all algorithms with weights equal to their F_1^{CV} during HP optimization as only four with the highest performance (*NN*, *SVM*, *SGB* and *RF*). This resulted in F_1^{CV} estimates slightly higher than the best values for individual algorithms used in voting (with corresponding gains in F_1^{CV} 0.007 and 0.004).

As the predictions of individual algorithms are uncalibrated we tried *ranking average* the probability outputs of individual learners. Using all classifiers resulted in degraded performance (-0.018) relative to the best individual classifier. Averaging ranks of predictions of four best-performing algorithms gives the same F_1^{CV} (0.0006). At the same time using two of the least performance algorithms in averaging brings some improvement relative to their individual score (0.032).

Also we combined class and probabilistic predictions of individual algorithms using higher-lever (meta) algorithm – *LR* using the *Stacking Generalization* or *stacking* method (Wolpert 1992) both alone and with original (*lower-level*) features[14]. Most improvement (0.007) was done with using only class predictions of four best classifiers. This could be the result of uncalibrated probabilistic outputs of the base algorithms.

We also attribute insignificant improvements of this ensembling methods to high correlation between predictions of individual algorithms (see Figure 6) (Sollich & Krogh 1996). This is because all classifiers HP were tuned to have highest F_1^{CV} using the same CV splits of the training data. Using different CV splits during HP optimization for each of the algorithm or larger training sample (that will allow calibration of the algorithms probabilistic

[14] We used *mlxtend* Python package (Raschka 2016).

	LR	kNN	RF	SGB	SVM	NN
LR	1	.76	.80	.80	.76	.76
kNN	.76	1	.79	.83	.80	.79
RF	.80	.79	1	.89	.83	.84
SGB	.80	.83	.89	1	.84	.82
SVM	.76	.80	.83	.84	1	.86
NN	.76	.79	.84	.82	.86	1

Figure 6. Pearson correlation coefficient between algorithms predictions estimated using CV on LMC_SC20 data set. This values are obtained with one fixed split of the data sample in train/test samples used in CV. Depending on the split, the presented values are changing with std 0.01-0.02 as estimated using 30 different splits.

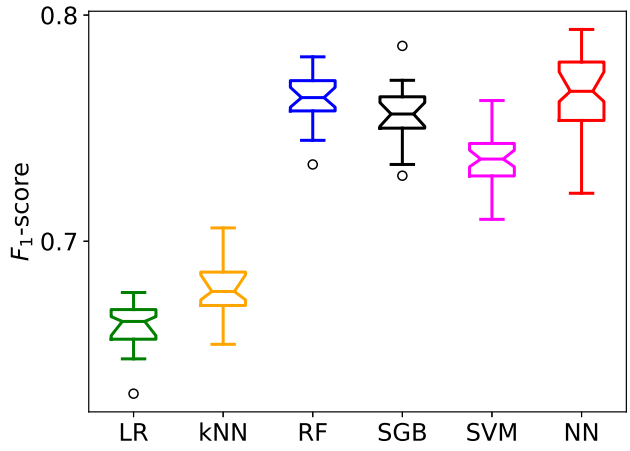

Figure 7. Boxplot of F_1^{CV} values obtained by classifiers with optimized HP (see Table 3) for 30 different CV splits of LMC_SC20 data on train/evaluation splits (not including the one used for HP tuning). The box extends from the lower to upper quartile values of the data, with a line at the median and narrowing of the box denotes confidence band on median. The whiskers extend from the box to 1.5 of interquantile range to show the range of the data. Points outside of the whiskers are considered as outliers.

outputs) will make using the ensembling methods more effectively (Ting & Witten 1999, Sigletos et al. 2005).

4.2.4 *Possible future improvements*

As can be seen from the learning curves presented at Figure 5, all high performance classifiers would benefit from increasing the amount of training data. Also, larger sample of variables also will allow one to calibrate classifiers and combine multiple classifiers probabilistic output using e.g. stacking (Section 4.2.3). Finally, as discussed in Section 4.3, larger sample size will help to escape overfitting due to small-sized training samples that could be unrepresentative of the general population.

A promising way to achieve larger training set size could be the artificial enlargement of training data (*data augmentation*; see e.g. Hoyle et al. 2015) by introducing possible variations to known constant/variable stars light curves (e.g. changing variability amplitude, noise level, addition of instrumental trends (see 4.3), etc.). According to Section 3 another promising way for improvement is engineering new features that quantify the object's image shape profile and position on a CCD chip, proximity to other detected objects, correlation of magnitude measurements with external parameters such as seeing and airmass, periodicity in light variations, shape of the period-folded light curve, etc.

4.3 Blind test on the new data set

The actual performance on unseen data is hard to estimate. As our data sample is quite small, we didn't hold out some part of it for testing classifiers on the unseen data (Hastie, Tibshirani & Friedman 2001). Performance on unseen data should be slightly lower than the estimations obtained using CV on the original data set (Table 3). We estimate the effect of this by considering distributions of F_1^{CV} values obtained by classifiers with best HP from Table 3 for 30 different splits of LMC_SC20 data sample on train/evaluation splits (not including the one used for HP tuning; Figure 7). The obtained F_1^{CV} values are lower than the best F_1^{CV} values presented in Table 3 up to 0.05 for *SVM* and nearly the same for *LR* (-0.01) and the same for *kNN*. The typical error estimated using variance of F_1^{CV} for different splits is 0.01.

On the other hand, CV estimate of prediction performance are pessimistic because only some portion of data is used to fit model (e.g. 75% in our case of 4-fold CV). Thus F_1 on new data set with the size of LMC_SC20 will be higher for high-variance algorithms (all except *LR*). The value of this bias can be estimated using learning curves (Figure 5; Hastie, Tibshirani & Friedman 2001). Interesting, that *SVM* that demonstrated the highest drop of performance on new CV splits should gain the most performance from the enlarging training sample according to its learning curve (Fig. 5).

Finally, if LMC_SC20 is not representative to the overall variable stars population, then we expect degraded performance of classifiers on new unseen data sampled from that population (i.e. overfitting). See discussion in Section 2.2. This and first item could be reduced with larger sample size.

We have tested *NN* classifier with chosen best hyperparameters on unseen data set consisting of 31798 stars (field LMC_SC19, Section 2). *NN* was fitted on whole training data set (LMC_SC20) with found best hyperparameters and its predictions were evaluated. We used default threshold (0.5) as this was the value used for hyperparameters optimization. The predicted variables were checked in existing catalogues (Section 2.1) and by visual inspection. Among the 205 candidates classified as variable stars, 178 occurred to be real variables (TP), 27 were considered FP.

The true variables/false candidates division may not be perfect, it involves the following assumptions:

- If a candidate variable is matched with a catalog, it is considered TP. We neglect the possibility that an object may have no detectable variations in OGLE-II data while being detected as variable by another survey.
- We consider TP candidates that are not matched to the catalogs of know variables, but upon visual inspection are identified as variable stars of a known type (Figure 8).
- We consider FP all candidates showing a continuous brightness increase or decline if they are not matched with known variables from the catalogs (lower right panel of Figure 10). This is

done to exclude possible long term instrumental trends and apparent variations caused by proper motion (Eyer & Woźniak 2001). It is possible that some true variables showing long-term brightness changes may be misattributed to instrumental trends and mislabeled as FP.

- We consider FP candidates showing elevated scatter in their light curves (compared to other objects of similar brightness), while showing no detectable periodicity in these variations (Figure 10). Specifically, we consider FP objects showing non-periodic dimming or brightening on a timescale shorter then the typical observing cadence. Young stellar objects and flare stars may show this type of behaviour. Hot/cold pixels underneath the star image may also produce light curves of these shapes. The inspection of images associated with individual measurements (that are not available to us) is necessary to judge if the measurements of a given object are reliable. We choose to exclude candidates showing this type of behaviour from the list of confirmed variables.

Among the 178 confirmed variable objects in LMC_SC19, 12 have never been reported as variable before. Table 4 presents the list of newly identified variables, their colors from Udalski et al. (2000) and the suggested classification according to the GCVS scheme (Samus' et al. 2017). Table 4 also lists one new variable, LMC_SC19_184609, that was not selected as a candidate variable by the final run of the *NN* classifier. This variable was identified by us during a test run with hyperparameters of the *NN* classifier differing from the ones listed in Table 3 (but some other variables were missed in this run). In order to obtain a more exhaustive list of variables one needs to lower the classifier's threshold or optimize its hyperparameters using a different performance metric (as discussed in Section 3.1). This will come at a price of increased number of false candidates that have to be rejected during visual inspection. The need to find an optimal trade-off between the rate of false candidates and search completeness is common to all variability detection techniques. Machine learning techniques considered here provide a more favorable ratio of true variables to false detections compared to the traditional methods (Table 3).

The light curves of the new variables are presented in Figure 8. The period search was performed using Deeming (1975) discrete Fourier transform method implemented in the online period search tool[15]. These newly identified variables give an idea of what kind of variables are missed by previous variability searches in LMC (Section 2.1): they have low amplitudes $\Delta I \lesssim 0.25$, many are periodic with long periods $\gtrsim 30^d$.

Eleven variable sources discovered with DIA by Zebrun et al. (2001) had no classification suggested in the literature. In order to account for these variables in Table 1, we classify them (Table 5) based on their light curves (Figure 9) and colors measured by Udalski et al. (2000).

Figures 8 and 9 present light curves of some of the variables correctly identified by the *NN* classifier – TP. Figure 10 illustrates light curves of objects that we believe were incorrectly selected by the *NN* classifier as candidate variables – FP. Eight known variables were not detected by the *NN* classifier (FN; Figure 12), three of them are eclipsing binaries identified by (Wyrzykowski et al. 2003, Graczyk et al. 2011) and the rest are RR Lyrae stars (Soszynski et al. 2003, Soszyński et al. 2009a). Figure 11 presents example light curves that were correctly identified by the classifier as non-variable (TN) while these objects have elevated values of some variability features and therefore would ap-

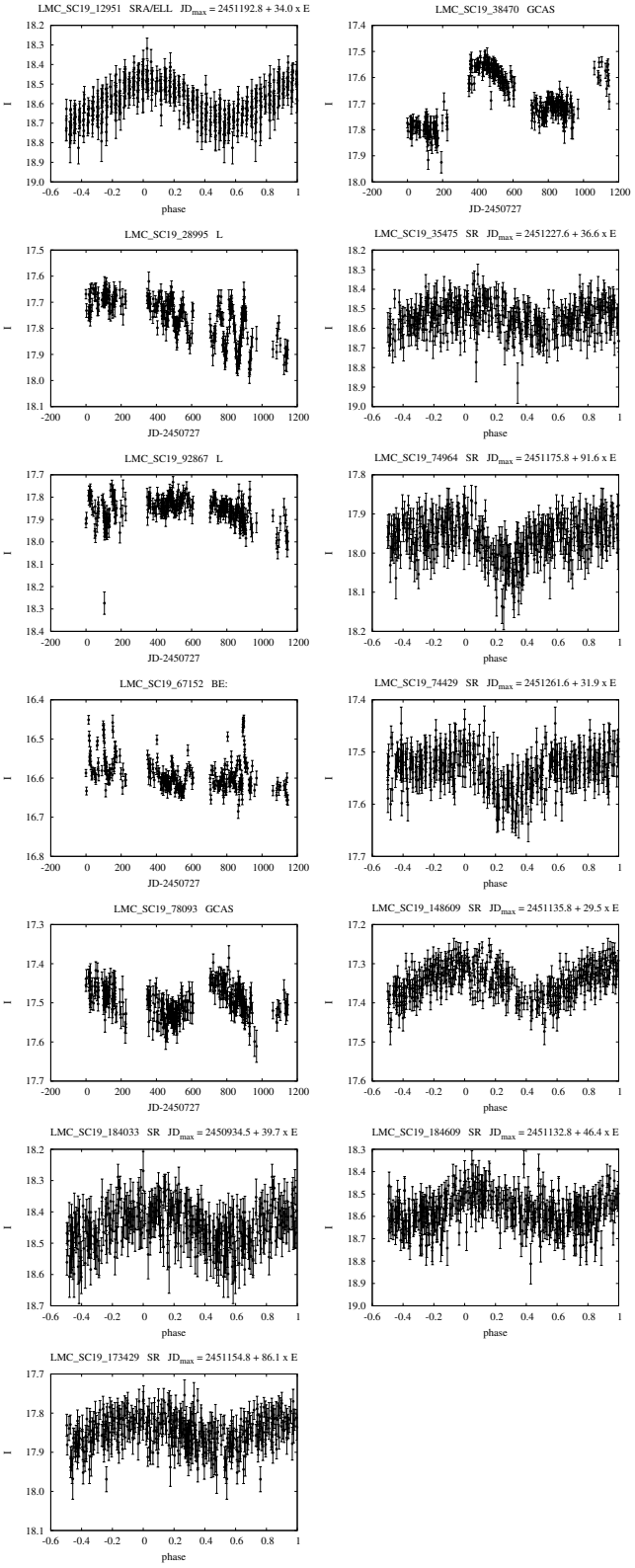

Figure 8. Light curves of the newly identified variable stars listed in Table 4. These are examples of true positives (TP): candidate variables identified by the *NN* classifier that passed visual inspection. Light curves of periodic variables are phase folded with the indicated light elements. For non-periodic variables the light curves are plotted as a function of time.

[15] http://scan.sai.msu.ru/lk/

Table 4. New variable stars identified in the field LMC_SC19 using the *NN* classifier with hyperparameters resulting in the best F_1-score for LMC_SC20.

Name	Position (J2000)	I-band range	Type	Light elements	$B-V$	$V-I$	Remarks
LMC_SC19_12951	05:42:40.86 −70:47:08.7	18.50–18.70	SRA/ELL	$JD_{max} = 2451192.8 + 34.0 \times E$	1.047	1.120	(1)
LMC_SC19_38470	05:42:41.10 −70:18:07.2	17.55–17.80	GCAS		0.039	0.040	
LMC_SC19_28995	05:42:42.43 −70:28:34.8	17.70–17.90	SR	$JD_{max} = 2451623.7 + 70.2 \times E$	1.364	1.500	(2)
LMC_SC19_18475	05:42:54.55 −70:23:19.7	18.50–18.60	SR	$JD_{max} = 2451227.6 + 36.6 \times E$	1.120	1.297	
LMC_SC19_92867	05:43:13.34 −70:15:23.2	17.80–18.00	L		1.203	1.285	(3)
LMC_SC19_74964	05:43:17.78 −70:36:02.7	17.95–18.10	SR	$JD_{max} = 2451175.8 + 91.6 \times E$	1.134	1.171	
LMC_SC19_67152	05:43:24.27 −70:44:16.3	16.45–16.65	BE:		−0.007	−0.002	(4)
LMC_SC19_74429	05:43:37.06 −70:37:03.3	17.50–17.60	SR	$JD_{max} = 2451261.6 + 31.9 \times E$	1.271	1.364	
LMC_SC19_78093	05:43:41.45 −70:32:19.9	17.45–17.60	GCAS		0.065	0.124	
LMC_SC19_184033	05:44:54.88 −70:18:02.3	18.40–18.50	SR	$JD_{max} = 2450934.5 + 39.7 \times E$	0.979	1.096	
LMC_SC19_148609	05:44:52.60 −71:01:38.1	17.30–17.40	SR	$JD_{max} = 2451135.8 + 29.5 \times E$	1.104	1.206	
LMC_SC19_184609	05:45:00.36 −70:17:26.8	18.50–18.60	SR	$JD_{max} = 2451132.8 + 46.4 \times E$	0.444	1.349	(5)
LMC_SC19_173429	05:45:01.34 −70:31:23.1	17.80–17.90	SR	$JD_{max} = 2451154.8 + 86.1 \times E$	0.967	1.041	

(1) $2''$ from an X-ray source 1WGA J0542.6−7047. (2) Periodic variations with changing amplitude are superimposed on a long-term declining trend. (3) The faint outlier point in the light curve (Figure 8) is likely not real. (4) Irregular flares lasting 10–20^d superimposed on a slow declining trend. (5) Found in one of the test run with hyperparameter values different from the ones listed in Table 3.

Table 5. Classification of the variable stars discovered with DIA.

Name	Position (J2000)	I-band range	Type	Light elements	$B-V$	$V-I$	Remarks
LMC_SC19_28805	05:42:47.47 −70:28:49.6	15.85–15.90	BE		0.125	0.355	(1)
LMC_SC19_32187	05:42:59.07 −70:26:01.0	16.10–16.15	BE		0.028	0.021	
LMC_SC19_41313	05:43:00.57 −70:15:45.8	16.35–16.45	L		0.551	0.912	
LMC_SC19_111203	05:43:53.34 −70:49:07.4	16.05–16.30	GCAS			0.004	
LMC_SC20_21197	05:45:21.69 −70:50:21.3	16.50–16.80	GCAS		0.020	0.017	
LMC_SC20_13936	05:45:22.51 −70:57:24.2	16.50–16.55	SR	$JD_{max} = 2451290.6 + 170.0 \times E$	1.572	1.464	
LMC_SC20_83505	05:45:49.71 −70:43:18.3	16.70–16.80	SR	$JD_{max} = 2450856.8 + 70.3 \times E$	0.927	1.108	(2)
LMC_SC20_134793	05:46:29.70 −70:43:56.8	17.00–17.05	SR	$JD_{max} = 2451657.6 + 53.0 \times E$	1.423	1.173	(3)
LMC_SC20_112813	05:46:31.25 −71:09:13.6	17.65–17.90	SR	$JD_{max} = 2451092.8 + 21.1 \times E$	0.926	1.175	(4)
LMC_SC20_131397	05:46:54.52 −70:45:01.4	17.50–17.65	SR	$JD_{max} = 2451256.6 + 51.5 \times E$	1.498	1.172	(2,5)
LMC_SC20_188685	05:47:02.33 −70:40:37.2	17.30–17.55	GCAS		−0.090	−0.032	

(1) B0IIIe spectral type according to Reid & Parker (2012). (2) Periodic brightness variations superimposed on a rising trend. (3) Three faint outliers are likely not real. (4) Periodic variations superimposed on a long-term wave. (5) Periodic variations stop around JD2450900 and reappear around JD2451800.

pear as false candidates in a variability search based on individual features (rather than their ML-based combination used here). As the light curves of FP and TN show high scatter of brightness measurements while showing no periodicity, it is most likely that the measurements are corrupted and do not reflect true brightness variations of these objects. Additional information, first of all - visual inspection of the images is required to identify one of few effects corrupting measurements of these objects.

4.4 Applicability to other photometric data sets

The suggested approach to variability detection should be applicable to any large set of light curves given that:

(i) a subset of these light curves is *a priori* classified into variable and non-variable ones,
(ii) both classes include hundreds of examples or more,
(iii) the examples are representative of variability types and measurement artifacts found in the studied set of light curves.

These requirements are easily satisfied for surveys covering a large fraction of the sky as they include many previously known variable stars of various types listed in the GCVS and the AAVSO International Variable Star Index (VSX[16]; Watson 2006). The photometric data suitable for the ML-based variability search are collected by a number of surveys including ASAS (Pojmanski 2002) and ASAS-SN (Shappee et al. 2014, Kochanek et al. 2017), CRTS (Drake et al. 2009), DES (Abbott et al. 2016), Gaia (Eyer et al. 2017) HATNet (Bakos et al. 2004), KELT (Pepper et al. 2007), MASCARA (Talens et al. 2017), NMW (Sokolovsky, Korotkiy & Lebedev 2014), NSVS (Woźniak et al. 2004), Pan-STARRS (Kaiser et al. 2010, Chambers et al. 2016), PTF (Law et al. 2009), SuperWASP (Butters et al. 2010), TrES (Alonso et al. 2007), VVV (Minniti et al. 2010) with even more ambitious surveys being developed, among them LSST (Ivezic et al. 2008), NGTS (Chazelas et al. 2012), PLATO (Rauer et al. 2014), TESS (Ricker et al. 2014), ZTF (Laher et al. 2017). The survey parameters such as photometric accuracy, observing cadence, single or multi-color observations, number of measurements per object in a single filter and magnitude range have an impact on the ability to discover various types of variable objects. The suggested ML-based variability detection approach

[16] `https://www.aavso.org/vsx/`

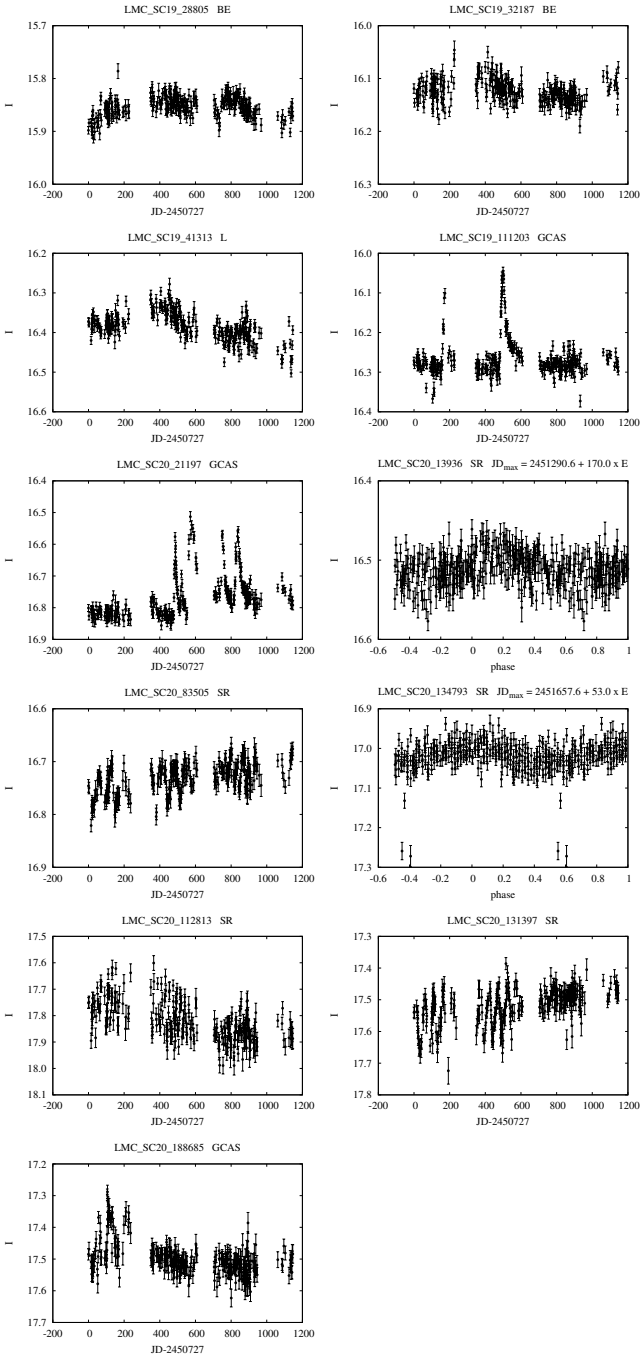

Figure 9. Light curves of variable stars with no previous reported classification (Table 5). Variability of these stars was discovered with DIA. The light curves are phased with the inidcated light elements for LMC_SC20_13936 and LMC_SC20_134793 and plotted as a function of time for the remaining stars.

is applicable regardless of the specifics of the survey's observing strategy.

Space photometry surveys such as Kepler (Borucki et al. 2010) and CoRoT (Auvergne et al. 2009) are capable of detecting brightness variations caused by magnetic activity (faculae, star spost; e.g. Shapiro et al. 2016) in Sun-like stars (Basri, Walkowicz & Reiners 2013) blurring the boundary between "variable" and "non-variable" stars. The question "is there any detectable variability" may still be relevant for the fainter stars observed in these surveys. One may be interested in identifying stars more variable than the Sun (McQuillan, Aigrain & Roberts 2012) or the ones showing periodic variability (Debosscher et al. 2009, 2011) – these problems require a different set of light curve features than the ones considered here. The variability detection approach presented here will likely not be useful for space astroseismology missions like MOST (Walker et al. 2003), BRITE (Weiss et al. 2014, Pablo et al. 2016, Popowicz et al. 2017) and the upcoming transit photometry mission CHEOPS (Broeg et al. 2013) as they observe (with superior accuracy) only one or few stars at a time.

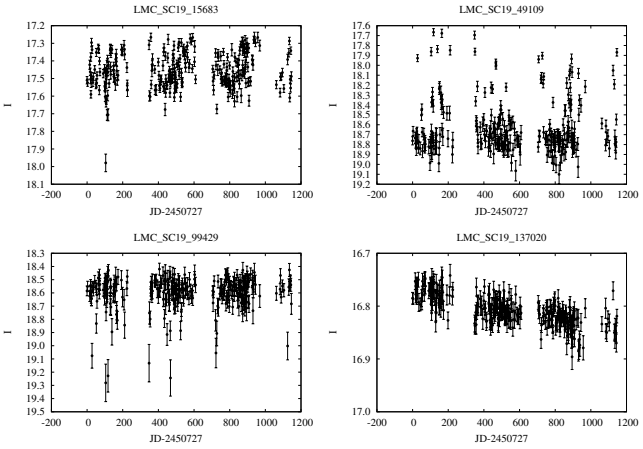

Figure 10. Example light curves of candidate variables rejected during visual inspection (FP).

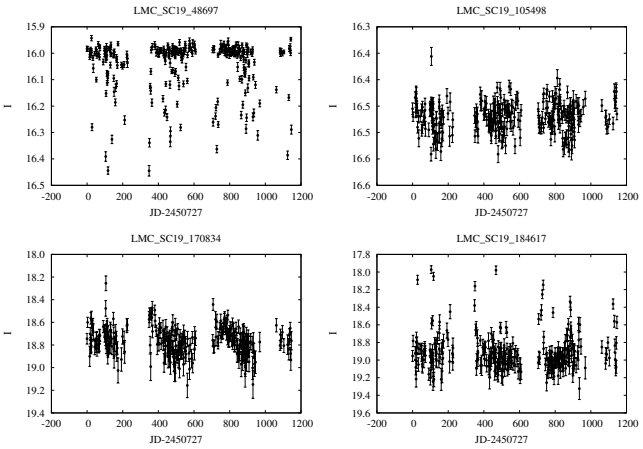

Figure 11. Example light curves having elevated values of individual variability indexes that were *correctly rejected* by the *NN* classifier (TN).

When applying the ML-based variability detection to new data sets, some light curve features listed in Table 2 may lose their predictive power while some that are found to be the least informative for the OGLE-II data set could become useful. When designing a variability detection procedure for a new set of photometric observations, it is desirable to go through the full path (Section 5) of feature selection/filtering, choosing multiple ML-algorithms, tuning their HP, checking for possible over/underfitting using learning curves before choosing the best algorithm and its HP values. The resulting classification performance will be different from the one reported in Table 3 and could be both better or worse depending on

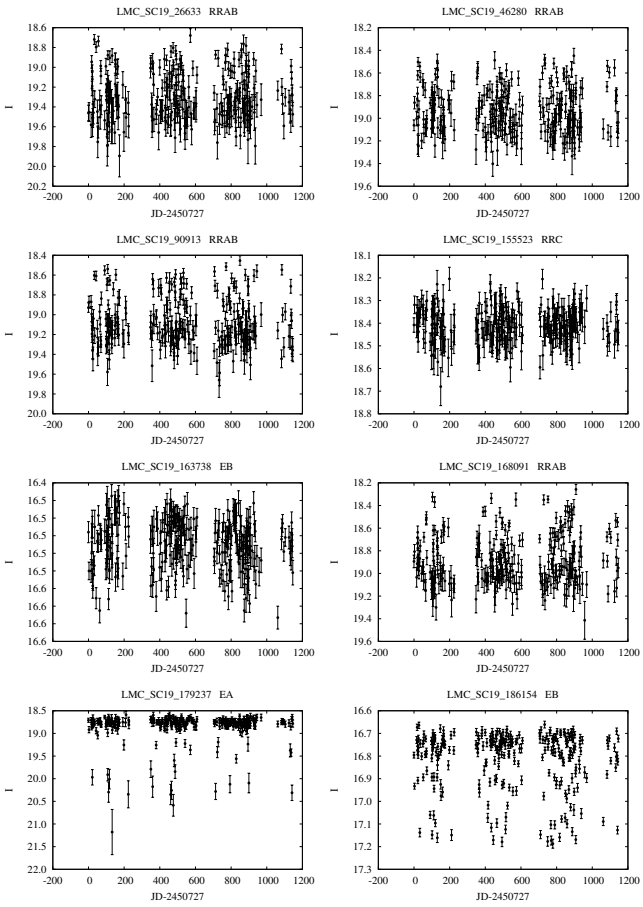

Figure 12. Light curves of known variables missclassified as non-variable by the *NN* (FN). While all these variables are periodic, we plot here their light curves as a function of time rather than phase to highlight similarities with some FP (Fig. 10) and TN (Fig. 11). Recall that none of the utilized variability features (Table 2) includes information about the period or the phased light curve shape.

sample size, light curve quality and the exact set of features used for classification.

5 CONCLUSIONS

We explore a novel approach to selecting variable objects from a set of light curves. The basic idea is to treat variability detection as a two-class classification problem (variable vs. non-variable objects) despite the intrinsic inhomogeneity of these classes and solve it with machine learning. The procedure may be summarized as:

(i) Search a representative subset of all light curves for variability using traditional methods, e.g. by visually inspecting light curves of all outliers in variability feature – magnitude plots. It is important to get reasonable confidence that the variability search in the subset is exhaustive. This will be our training subset.

(ii) For each light curve compute a set of features (Table 2) that highlight some or all types of variability while hiding unimportant differences between the light curves (i.e. the difference in the number of measurements).

(iii) Choose a machine learning algorithm and tune its hyperparameters on the training subset using cross-validation as described in Section 3.3. Table 3 presents an example of how the optimal hyperparameter values may look like. One may control the trade-off between the completeness of variability search and the rate of false detections by selecting performance metrics (e.g. F_β instead of F_1, Section 3.1) maximized during the optimal hyperparameters search.

(iv) Train the algorithm with the optimized hyperparameters on the whole training subset.

(v) Apply the algorithm to the full set of light curves and inspect the ones classified as variable. One may control the false detections rate at this stage by changing the classifier threshold.

This procedure works even with a highly imbalanced training subsample of a modest size: 168 variables among 30265 OGLE-II light curves (Section 2.1; see also the cross-validation scores in Figure 5). Application to an independent set of 31798 OGLE-II light curves resulted in selection of 205 candidate variables, 27 of which turned out to be false detections and 178 – real variables (12 of them new, Table 4, Figure 8).

To directly compare traditional variability search methods to the machine learning algorithms considered here, we restricted ourselves to the data sets used by Sokolovsky et al. (2017) who compared effectiveness of various variability indices (features). In terms of F_1-score (Table 3), all machine learning algorithms tested here outperform each individual variability index as well as their linear combination. *NN*, *SVM*, *SGB* and *RF* algorithms show the best performance (Figure 4). In addition to the OGLE-II data discussed in details here, these conclusions are confirmed on two other data sets from Sokolovsky et al. (2017) that were collected with different telescopes and processed using different source extraction and photometry software (Section 4.1). To improve the variable objects selection results even further, one needs to use a larger training sample and engineer additional features that would quantify the object's image shape, it's proximity to other detected objects and periodicity in light variations. The suggested ML-based variability detection technique should be applicable to any large ($\gtrsim 10^4$) set of light curves given that a representative sub-sample of these light curves is a priori classified as "constant" or "variable" by other means (Section 4.4).

ACKNOWLEDGMENTS

We thank the anonymous referees for helpful comments. We thank Dr. Laurent Eyer for pointing out the hypothesis-testing approach to the problem of variability detection, Dr. Antonios Karampelas, Dr. Nikolay Samus, Dr. Maria Ida Moretti for critically reading this manuscript. KVS and PG are supported by the European Space Agency (ESA) under the "Hubble Catalog of Variables" program, contract No. 4000112940. This research has made use of the International Variable Star Index (VSX) database, operated at AAVSO, Cambridge, Massachusetts, USA. This research has made use of the VizieR catalogue access tool, CDS, Strasbourg, France. The original description of the VizieR service is presented by Ochsenbein, Bauer & Marcout (2000). We also relied on the catalog matching capabilities of TOPCAT (Taylor 2005) and catalog and image visualization with Aladin sky atlas (Bonnarel et al. 2000). This research has made use of NASA's Astrophysics Data System.

REFERENCES

Abbott T. et al., 2016, MNRAS, 460, 1270

Alard C., Lupton R. H., 1998, ApJ, 503, 325
Alcock C. et al., 2000, ApJ, 542, 281
Alonso R. et al., 2007, in Astronomical Society of the Pacific Conference Series, Vol. 366, Transiting Extrapolar Planets Workshop, Afonso C., Weldrake D., Henning T., eds., p. 13
Auvergne M. et al., 2009, A&A, 506, 411
Bakos G., Noyes R. W., Kovács G., Stanek K. Z., Sasselov D. D., Domsa I., 2004, PASP, 116, 266
Basri G., Walkowicz L. M., Reiners A., 2013, ApJ, 769, 37
Becker A. C. et al., 2005, in IAU Symposium, Vol. 225, Gravitational Lensing Impact on Cosmology, Mellier Y., Meylan G., eds., pp. 357–362
Bergstra J. S., Bardenet R., Bengio Y., Kégl B., 2011, in Advances in Neural Information Processing Systems, pp. 2546–2554
Beyer K., Goldstein J., Ramakrishnan R., Shaft U., 1999, When Is "Nearest Neighbor" Meaningful?, Beeri C., Buneman P., eds., Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 217–235
Bonnarel F. et al., 2000, A&AS, 143, 33
Borucki W. J. et al., 2010, Science, 327, 977
Boser B. E., Guyon I. M., Vapnik V. N., 1992, in Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92, ACM, New York, NY, USA, pp. 144–152
Bramich D. M., Horne K., Alsubai K. A., Bachelet E., Mislis D., Parley N., 2016, MNRAS, 457, 542
Brieman L., 1996, Machine Learning, 24, 2, 123
Broeg C. et al., 2013, in European Physical Journal Web of Conferences, Vol. 47, European Physical Journal Web of Conferences, p. 03005
Burdanov A. Y. et al., 2016, MNRAS, 461, 3854
Burdanov A. Y., Krushinsky V. V., Popov A. A., 2014, Astrophysical Bulletin, 69, 368
Butters O. W. et al., 2010, A&A, 520, L10
Cattell R. B., 1966, Multivariate Behavioral Research, 1, 2, 245, pMID: 26828106
Chambers K. C. et al., 2016, ArXiv e-prints
Chazelas B. et al., 2012, in Proc. SPIE, Vol. 8444, Ground-based and Airborne Telescopes IV, p. 84440E
Chen T., Guestrin C., 2016, ArXiv:1603.02754
Christ M., Kempa-Liehr A. W., Feindt M., 2016, ArXiv:1610.07717
Cieslinski D., Diaz M. P., Mennickent R. E., Pietrzyński G., 2003, PASP, 115, 193
Cioni M.-R. L. et al., 2011, A&A, 527, A116
de Diego J. A., 2010, AJ, 139, 1269
Debosscher J., Blomme J., Aerts C., De Ridder J., 2011, A&A, 529, A89
Debosscher J. et al., 2009, A&A, 506, 519
Deeming T. J., 1975, Ap&SS, 36, 137
Drake A. J. et al., 2009, ApJ, 696, 870
Elorrieta F. et al., 2016, A&A, 595, A82
Eyer L., 2002, Acta Astron., 52, 241
Eyer L., 2005, in ESA Special Publication, Vol. 576, The Three-Dimensional Universe with Gaia, Turon C., O'Flaherty K. S., Perryman M. A. C., eds., p. 513
Eyer L. et al., 2017, ArXiv:1702.03295
Eyer L., Woźniak P. R., 2001, MNRAS, 327, 601
Fawcett T., 2006, Pattern Recognition Letters, 27, 8, 861 , rOC Analysis in Pattern Recognition
Ferreira Lopes C. E., Cross N. J. G., 2016, A&A, 586, A36
Ferreira Lopes C. E., Dékány I., Catelan M., Cross N. J. G., Angeloni R., Leão I. C., De Medeiros J. R., 2015, A&A, 573, A100
Figuera Jaimes R., Arellano Ferro A., Bramich D. M., Giridhar S., Kuppuswamy K., 2013, A&A, 556, A20
Forman G., Scholz M., 2010, ACM SIGKDD Explorations Newsletter, 12, 1, 49
Fraser O. J., Hawley S. L., Cook K. H., 2008, AJ, 136, 1242
Friedman J. H., 2001, Ann. Statist., 29, 5, 1189
Friedman J. H., 2002, Computational Statistics & Data Analysis, 38, 4, 367 , nonlinear Methods and Data Mining
Friedrich S., Koenig M., Wicenec A., 1997, in ESA Special Publication, Vol. 402, Hipparcos - Venice '97, Bonnet R. M., Høg E., Bernacca P. L., Emiliani L., Blaauw A., Turon C., Kovalevsky J., Lindegren L., Hassan H., Bouffard M., Strim B., Heger D., Perryman M. A. C., Woltjer L., eds., pp. 441–444
Fruth T. et al., 2012, AJ, 143, 140
Graczyk D. et al., 2011, Acta Astron., 61, 103
Guyon I., Elisseeff A., 2003, J. Mach. Learn. Res., 3, 1157
Hastie T., Tibshirani R., Friedman J. H., 2001, The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations. New York: Springer-Verlag, p. 533
Haykin S., 1999, Neural Networks: A Comprehensive Foundation, International edition. Prentice Hall
Ho T., 1998, IEEE Transactions on Pattern Analysis and Machine Intelligence., 20, 8, 832
Hoffmeister C., Richter G., Wenzel W., 1990, Variable stars
Hoyle B., Rau M. M., Bonnett C., Seitz S., Weller J., 2015, Monthly Notices of the Royal Astronomical Society, 450, 1, 305
Huber M. E., Everett M. E., Howell S. B., 2006, AJ, 132, 633
Hughes G., 1968, IEEE Transactions on Information Theory, 14, 1, 55
Ivezic Z. et al., 2008, ArXiv e-prints
Kaiser N. et al., 2010, in Proc. SPIE, Vol. 7733, Ground-based and Airborne Telescopes III, p. 77330E
Kim D.-W., Bailer-Jones C. A. L., 2016, A&A, 587, A18
Kim D.-W., Protopapas P., Alcock C., Byun Y.-I., Khardon R., 2011, in Astronomical Society of the Pacific Conference Series, Vol. 442, Astronomical Data Analysis Software and Systems XX, Evans I. N., Accomazzi A., Mink D. J., Rots A. H., eds., p. 447
Kim D.-W., Protopapas P., Bailer-Jones C. A. L., Byun Y.-I., Chang S.-W., Marquette J.-B., Shin M.-S., 2014, A&A, 566, A43
Kim D.-W., Protopapas P., Trichas M., Rowan-Robinson M., Khardon R., Alcock C., Byun Y.-I., 2012, ApJ, 747, 107
Kiss L. L., Bedding T. R., 2003, MNRAS, 343, L79
Kochanek C. S. et al., 2017, ArXiv e-prints
Kolesnikova D. M., Sat L. A., Sokolovsky K. V., Antipin S. V., Belinskii A. A., Samus' N. N., 2010, Astronomy Reports, 54, 1000
Kolesnikova D. M., Sat L. A., Sokolovsky K. V., Antipin S. V., Samus N. N., 2008, Acta Astron., 58, 279
Kononenko I., Bratko I., 1991, Machine Learning, 6, 1, 67
Kovács G., Zucker S., Mazeh T., 2002, A&A, 391, 369
Kozłowski S. et al., 2013, ApJ, 775, 92
Laher R. R. et al., 2017, ArXiv e-prints
Lapukhin E. G., Veselkov S. A., Zubareva A. M., 2013, Peremennye Zvezdy Prilozhenie, 13
Lapukhin E. G., Veselkov S. A., Zubareva A. M., 2016, Peremennye Zvezdy Prilozhenie, 16
Law N. M. et al., 2009, PASP, 121, 1395
Mason L., Baxter J., Bartlett P., Frean M., 1999, in Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99, MIT Press, Cambridge, MA, USA, pp. 512–518
McQuillan A., Aigrain S., Roberts S., 2012, A&A, 539, A137
Minniti D. et al., 2010, New A, 15, 433
Mowlavi N., 2014, A&A, 568, A78
Nadir E., Othman I., Ahmed O., 2014, Research Journal of Applied Sciences, Engineering and Technology, 7, 625
Nair V., Hinton G. E., 2010, in Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, Omnipress, USA, pp. 807–814
Nandra K., George I. M., Mushotzky R. F., Turner T. J., Yaqoob T., 1997, ApJ, 476, 70
Nun I., Protopapas P., Sim B., Zhu M., Dave R., Castro N., Pichara K., 2015, ArXiv:1506.00010
Ochsenbein F., Bauer P., Marcout J., 2000, A&AS, 143, 23
Pablo H. et al., 2016, PASP, 128, 12, 125001
Palaversa L. et al., 2013, AJ, 146, 101
Parks J. R., Plavchan P., White R. J., Gee A. H., 2014, ApJS, 211, 3
Pawlak M. et al., 2016, ArXiv:1612.06394
Pearson K., 1901, Philosophical Magazine Series 6, 2, 11, 559
Pedregosa F. et al., 2011, Journal of Machine Learning Research, 12, 2825
Pepper J. et al., 2007, PASP, 119, 923

Pérez-Ortiz M. F., García-Varela A., Quiroz A. J., Sabogal B. E., Hernández J., 2017, ArXiv:1707.04560
Piquard S., Halbwachs J.-L., Fabricius C., Geckeler R., Soubiran C., Wicenec A., 2001, A&A, 373, 576
Pojmanski G., 2002, Acta Astron., 52, 397
Poleski R. et al., 2010, Acta Astron., 60, 1
Popov A. A., Burdanov A. Y., Zubareva A. M., Krushinsky V. V., Avvakumova E. A., Ivanov K., 2015, Peremennye Zvezdy Prilozhenie, 15
Popowicz A. et al., 2017, ArXiv:1705.09712
Raschka S., 2015, Python Machine Learning. Packt Publishing
Raschka S., 2016, Mlxtend
Rauer H. et al., 2014, Experimental Astronomy, 38, 249
Reid W. A., Parker Q. A., 2012, MNRAS, 425, 355
Ricker G. R. et al., 2014, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 9143, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, p. 20
Rijsbergen C. V., 1974, Journal of Documentation, 30, 4, 365
Rose M. B., Hintz E. G., 2007, AJ, 134, 2067
Sabogal B. E., Mennickent R. E., Pietrzyński G., Gieren W., 2005, MNRAS, 361, 1055
Saito T., Rehmsmeier M., 2015, PLOS ONE, 10, 3, 1
Samus' N. N., Kazarovets E. V., Durlevich O. V., Kireeva N. N., Pastukhova E. N., 2017, Astronomy Reports, 61, 80
Schapire R. E., 1990, Machine Learning, 5, 2, 197
Shapiro A. I., Solanki S. K., Krivova N. A., Yeo K. L., Schmutz W. K., 2016, A&A, 589, A46
Shappee B. J. et al., 2014, ApJ, 788, 48
Shin M.-S., Byun Y.-I., 2007, in Astronomical Society of the Pacific Conference Series, Vol. 362, The Seventh Pacific Rim Conference on Stellar Astrophysics, Kang Y. W., Lee H.-W., Leung K.-C., Cheng K.-S., eds., p. 255
Shin M.-S., Sekora M., Byun Y.-I., 2009, MNRAS, 400, 1897
Shin M.-S., Yi H., Kim D.-W., Chang S.-W., Byun Y.-I., 2012, AJ, 143, 65
Sigletos G., Paliouras G., Spyropoulos C. D., Hatzopoulos M., 2005, J. Mach. Learn. Res., 6, 1751
Smialowski P., Frishman D., Kramer S., 2010, Bioinformatics, 26, 3, 440
Sokolovsky K., Korotkiy S., Lebedev A., 2014, in Astronomical Society of the Pacific Conference Series, Vol. 490, Stell Novae: Past and Future Decades, Woudt P. A., Ribeiro V. A. R. M., eds., p. 395
Sokolovsky K. V. et al., 2017, MNRAS, 464, 274
Sokolovsky K. V., Kovalev Y. Y., Kovalev Y. A., Nizhelskiy N. A., Zhekanis G. V., 2009, Astronomische Nachrichten, 330, 199
Sokolovsky K. V., Lebedev A. A., 2017, ArXiv:1702.07715
Sollich P., Krogh A., 1996, Advances in Neural Information Processing Systems, 8, 190
Soszynski I. et al., 2004, Acta Astron., 54, 347
Soszynski I. et al., 2005, Acta Astron., 55, 331
Soszyński I. et al., 2012, Acta Astron., 62, 219
Soszynski I. et al., 2003, Acta Astron., 53, 93
Soszyński I. et al., 2009a, Acta Astron., 59, 1
Soszyński I. et al., 2009b, Acta Astron., 59, 239
Spano M., Mowlavi N., Eyer L., Burki G., Marquette J.-B., Lecoeur-Taïbi I., Tisserand P., 2011, A&A, 536, A60
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, Journal of Machine Learning Research, 15, 1929
Stetson P. B., 1996, PASP, 108, 851
Szymanski M. K., 2005, Acta Astron., 55, 43
Talens G. J. J., Spronck J. F. P., Lesage A.-L., Otten G. P. P. L., Stuik R., Pollacco D., Snellen I. A. G., 2017, A&A, 601, A11
Tamuz O., Mazeh T., North P., 2006, MNRAS, 367, 1521
Tang S., Grindlay J., Los E., Servillat M., 2013, PASP, 125, 857
Taylor M. B., 2005, in Astronomical Society of the Pacific Conference Series, Vol. 347, Astronomical Data Analysis Software and Systems XIV, Shopbell P., Britton M., Ebert R., eds., p. 29
Ting K. M., Witten I. H., 1999, J. Artif. Int. Res., 10, 1, 271
Tisserand P. et al., 2007, A&A, 469, 387
Udalski A., Kubiak M., Szymanski M., 1997, Acta Astron., 47, 319
Udalski A., Soszynski I., Szymanski M., Kubiak M., Pietrzynski G., Wozniak P., Zebrun K., 1999, Acta Astron., 49, 223
Udalski A., Szymanski M., Kubiak M., Pietrzynski G., Soszynski I., Wozniak P., Zebrun K., 2000, Acta Astron., 50, 307
Udalski A., Szymanski M. K., Soszynski I., Poleski R., 2008, Acta Astron., 58, 69
Udalski A., Szymański M. K., Szymański G., 2015, Acta Astron., 65, 1
Valverde-Albacete F. J., Peláez-Moreno C., 2014, PLoS ONE, 9, 1, e84217+
Vapnik V., 1996, The Nature of Statistical Learning Theory. New York: Springer-Verlag
Vorontsov K., 2013, Machine learning. lecture course. `http://www.MachineLearning.ru/wiki`
Walker G. et al., 2003, PASP, 115, 1023
Watson C. L., 2006, Society for Astronomical Sciences Annual Symposium, 25, 47
Weiss W. W. et al., 2014, PASP, 126, 573
Welch D. L., Stetson P. B., 1993, AJ, 105, 1813
Wolpert D. H., 1992, Neural Networks, 5, 241
Wood P. R., 2000, PASA, 17, 18
Wood P. R. et al., 1999, in IAU Symposium, Vol. 191, Asymptotic Giant Branch Stars, Le Bertre T., Lebre A., Waelkens C., eds., p. 151
Woźniak P. R. et al., 2004, AJ, 127, 2436
Wyrzykowski Ł. et al., 2009, MNRAS, 397, 1228
Wyrzykowski L. et al., 2003, Acta Astron., 53, 1
Zadrozny B., Elkan C., 2002, in Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02, ACM, New York, NY, USA, pp. 694–699
Zebrun K. et al., 2001, Acta Astron., 51, 317
Zhang M., Bakos G. Á., Penev K., Csubry Z., Hartman J. D., Bhatti W., de Val-Borro M., 2016, PASP, 128, 3, 035001
Zhang X.-B., Deng L.-C., Xin Y., Zhou X., 2003, Chinese J. Astron. Astrophys., 3, 151

APPENDIX A: CLIPPED LIGHT CURVE FEATURES

Corrupted photometric measurements result in outlier points in a light curve (Sec. 2, see for example LMC_SC19_92867 in Fig. 8 and LMC_SC20_134793 in Fig 9) that may alter the light curve feature values while having no relation to object's variability. One way to minimize this problem is to apply clipping to the light curve before computing the feature values. Kim & Bailer-Jones (2016) perform σ-clipping before computing all the light curve features used for classification of periodic variable stars. As we are concerned with detection of non-periodic stars (as well as periodic ones) that may show variability only occasionally, we do not apply σ-clipping. Instead, for a few features that are most sensitive to outlier light curve points we compute both their unclipped and clipped versions (Table 2) as outlined below.

A1 VAST-style clipped $\sigma - \sigma_{\rm clip}$

This clipped statistic was used for variability detection in the early versions of the VAST code. From each light curve we drop 5 per cent of brightest and 5 per cent of faintest points, but not more than 5 points from each side and compute the unweighted standard deviation

$$\sigma_{\rm clip} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(m_i - \bar{m})^2}$$

where N is the number of points in the clipped light curve, $\bar{m}$ is the mean magnitude of the set m_i of magnitude measurements remaining after clipping. In many data sets $\sigma_{\rm clip}$ proved to be a more useful variability indicator than σ computed over the non-clipped light curve. It is also more sensitive then MAD and IQR (Table 2) to rare variability events (flares, eclipses). Similar clipping schemes based on removing a predefined percentage or number of brightest and faintest points were applied by Palaversa et al. (2013), Tang et al. (2013).

A2 Clipped Stetson's indices $J_{\rm clip}$ and $L_{\rm clip}$

Stetson (1996) suggested variability detection statistics J and L that rely on observations taken close in time being grouped into pairs. If both observations in a pair deviate in the same direction from the mean brightness, this indicates the light curve is smooth (as expected for an object varying on a timescale longer than the time difference between the observations in the pair). Sokolovsky et al. (2017) suggested a modified versions of these variability indices, $J_{\rm clip}$ and $L_{\rm clip}$, that did not form a pair if the magnitude difference between the two observations was larger than a predefined limit (indicating that one of the observations in the pair might be corrupted). The clipping in these indices is done on the magnitude difference in pairs, not on the original light curve. This modification however did not result in a considerable performance improvement compared to the original J and L when tested on real data Sokolovsky et al. (2017).

Stetson (1996) advocates for iterative re-weighting as an alternative to clipping. This allows one avoid having a sharp boundary between the observations that are "in" or "out". In the original J and L definitions iterative re-weighting is applied only to the mean magnitude calculation, but not to the observations that form pairs.

APPENDIX B: VARIABILITY FEATURE – MAGNITUDE PLOTS

Figure B1 presents plots of selected individual variability features (Table 2) as a function of OGLE I magnitude. Such plots are typically used to identify variable objects by selecting a magnitude-dependent cut-off for an individual index and visually inspecting light curves of all objects above the cut-off (e.g. Sokolovsky et al. 2017).

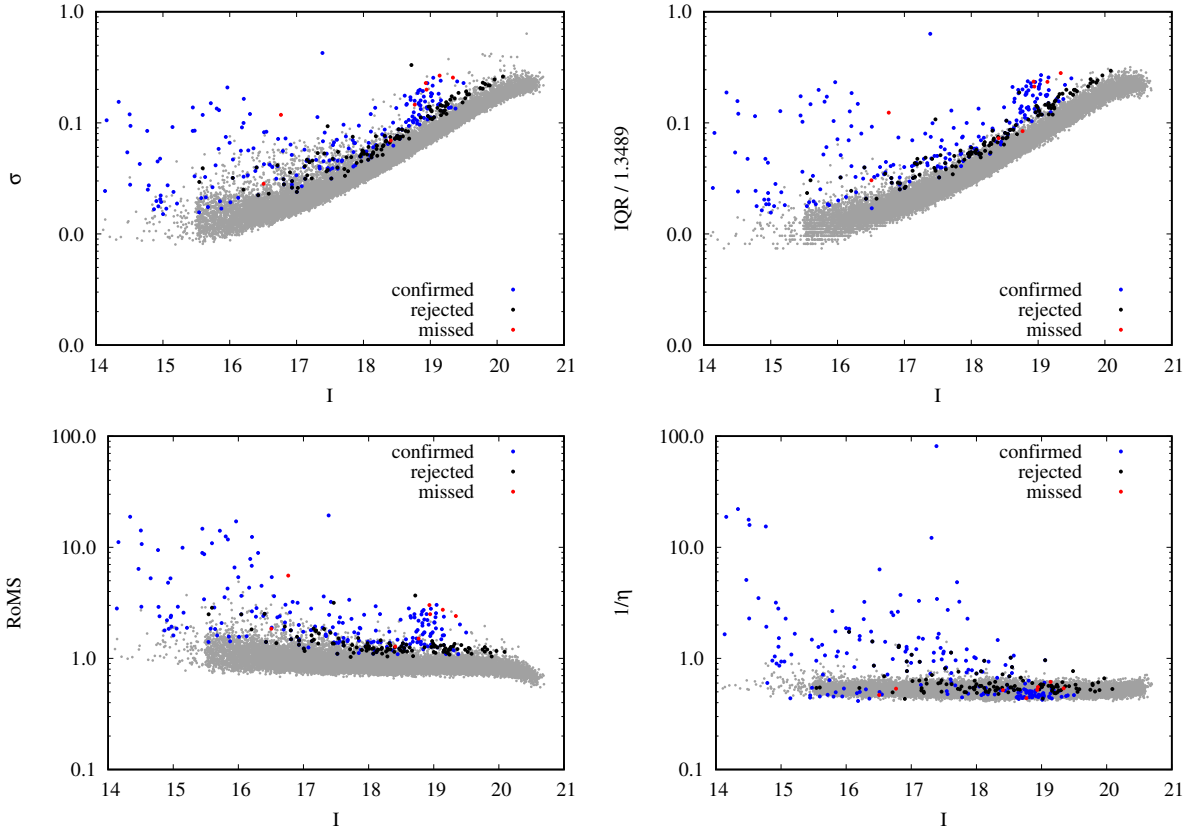

Figure B1. Variability feature vs. *I* magnitude plots showing all objects in grey and highlighting candidate variables selected by the *NN* classifier and confirmed by visual inspection (see example light curves in Figures 8 and 9), rejected after visual inspection (Figure 10) as well as the known variable stars missed by the *NN* classifier (Figure 12). The IQR is scaled to σ of the Gaussian distribution so the numerical values of the two upper plots may be compared directly.