Light Curve Classification for LSST EPO

Ryan J. McCall

May 25, 2018

Overview

The LSST alert stream, once online, will produce an inordinate amount of astronomical data requiring automated machine learning analysis. In particular the EPO is interested in the classification of *light curves (LCs)*, time series of light measurements from variable stars or other varying celestial objects. In this report, I summarize my research on light curve classification techniques. During this project Ben Emmons and I prepared several datasets for experimentation. Working with a proven light curve classification technique, I ran several supervised and unsupervised ML experiments. These experiments were coded as a machine learning pipeline in Python in the Light Curve Machine Learning (LCML) Github repository[1]. Here I report my findings on some initial experiments using a feature-extraction-based approach. The project was halted in May 2018 because it was determined that the LSST alert stream will not be coming online in the near future.

Assumptions

While it may be possible to train ML models online, based on incoming data, here I focused on a batch training paradigm involving static datasets. These datasets provide photometry in units of magnitude and are taken as a proxy for actual LSST alert stream photometry, which is expected to come in both units of flux and magnitude [8].

Secondly, while some datasets may provide multiple color bands of data for the same astronomical object, I treated each distinct band as a separate light curve. In contrast, more advanced multi-band features can be computed with two color bands that are time-aligned. This multi-band analysis was deemed beyond the scope of this project.

Datasets

For this work a simple .csv dataset format was used. Individual light curves were concatenated together monolithically into a single (large) file. Each row in the file represents a single data point in a time series and has the following columns: unique ID, class label, time value, magnitude value, and magnitude error value. Next I describe two labeled datasets used in this work, MACHO and OGLE3. We also obtained a third labeled dataset of Kepler Exoplanets, however it was not evaluated.

MACHO. MACHO stands for "massive astrophysical compact halo object" and is any kind of astronomical body that might explain the apparent presence of dark matter in galaxy halos. This particular dataset comes from work classifying 611 eclipsing binary stars in the MACHO project photometry database [9]. I constructed a labeled dataset of MACHO photometry called from the database tables found on the MACHO TAP server [2]. Initially, to view the server data I tested out database queries and viewed results using the TOPCAT application[3]. Next I wrote scripts to access the same tables using the STILTS command line tool[4]. The resultant .csv file has a unique ID column that is the hyphen-separated combination of observation field, tile number, sequence number, and color band. The file is about 2.2GB and contains 42,325 light curves. Based on selection criteria described later on (see section Preprocessing), only 21,056 were deemed useable. One related side note here is that additional unlabeled MACHO LCs were found on the MACHO website[5]. Ben Emmons converted

1 https://github.com/lsst-epo/light_curve_ml
2 http://machotap.asvo.nci.org.au/ncitap/tap
3 http://www.star.bris.ac.uk/~mbt/topcat/
4 http://www.star.bris.ac.uk/~mbt/stilts/
5 http://macho.nci.org.au/macho_photometry/

them into large .csv files, however, I did not make use of these during this project, although they do appear viable.

OGLE3. This dataset is based on the OGLE3 Catalog of variable stars[6]. The script used to obtain this data may be found in the LCML repository[7]. The variable stars come from three possible fields: Galactic bulge (BLG), Large Magellanic Cloud (LMC), or Small Magellanic Cloud (SMC). There are 8 possible categories: anomalous Cepheids, classical Cepheids, type II Cepheids, delta Scuti stars, Long Period Variables, Double Period Variables, R CrB stars, and RR Lyrae stars. Most objects have light curve data in the I (infrared) and V (visual) bands. Times are given in Heliocentric Julian Day (HJD-2450000) and are accompanied by magnitudes and errors. The dataset involves 399,679 unique objects (represented by OGLE3 IDs) for a total of 786,650 light curves, since objects may have data in multiple bands. Some light curves were too short, for example I found that only 465,325 (59.15%) had a length of at least 60. Kim and Bailer-Jones [4] have suggested a length of at least 80 for periodic variables.

Existing Work and Approach

Some challenges of light curve classification include the presence of arbitrary temporal gaps and nonuniform sampling times. One proven approach to light curve analysis involves "feature extraction" as was first described by Richards et al. [1] and later extended by Nun & Protopapas [2] and Kim and Bailer-Jones [4]. A feature is an function taking light curve data and outputting a single scalar value. At a minimum the input data includes time and magnitude values but may also require the error in magnitude as well as magnitude and/or error data in additional color bands. Features range from simplistic (mean and std) to more exotic statistical measurements selected to be robust against temporal gaps and sampling issues (see [1], [2], [4]). Some features specifically assume periodicity [4]. Typically around 60 or so of these features are computed for each light curve yielding a feature vectors. These feature vectors can then be analyzed using known machine learning techniques. Following this "Richards" family of approaches, I also used random forests for supervised classification. I also experimented with unsupervised clustering approaches including mini-batch *k*-means and agglomerative clustering.

For the feature extraction implementation I chose the `feets` Python library[8], an actively maintained fork of the original `fats` library[9] by Nun. I used `feets` version 0.4 for which there were 63 distinct features. While I determined that some features were computationally expensive (e.g., CAR_mean, CAR_sigma, CAR_tau), I settled on using all features in experimentation. For the supervised and unsupervised learning algorithms, including random forests, clustering, and related metrics, I used the `scikit-learn` Python library[10].

The Random (decision) forest technique constructs a multitude of decision trees at training time. Multiple trees are merged together to get more accurate and stable predictions. Each decision tree models decisions and their consequences. Nodes represent Boolean tests of an attribute (e.g., is feature *5 > 0.7*?). Branches represent test outcomes and leaf nodes represent class labels. Paths from the root to leaf nodes thus encode classification rules. The technique generates *M* decision trees from *M* random subsets of features. The classifier output is the mode of the output of all the decision trees.

The advantages of random forests are accuracy and the flexibility to handle large input dimensionality. They tend to not overfit, require data scaling, and tolerate missing values. The disadvantages are that the resultant model is a difficult-to-understand "black box", prediction

6 http://ogledb.astrouw.edu.pl/~ogle/CVS/
7 Source file lcml.data.acquisition.ogle.download_ogle3.py
8 https://github.com/carpyncho/feets
9 https://github.com/isadoranun/FATS
10 http://scikit-learn.org/stable/index.html

generation can be slower than other techniques, and the model size can be large in terms of memory.

Light Curve Machine Learning (LCML) Github Repository
The follow description pertains to version 0.0.1 of the `lcml` Python project[11]. To orchestrate the analysis outlined above, I developed a basic machine learning pipeline with supervised and unsupervised versions. The details are given in the repository's README file. The pipeline makes use of the lightweight sqlite3 database to save intermediate work and avoid RAM issues. A brief summary of the main stages follows:

Loading. A general function is used to load a source dataset .csv file parsing out individual light curves and loading them into a "raw" light curve table.

Preprocessing. Raw light curves are preprocessed in the following manner. First, known bogus values and statistical outliers are removed from each light curve. Afterwards, if the light curve's length is NOT sufficiently long (typically 80) it is discarded. Next, surviving light curves can optionally have their magnitude and error values standardized to a mean of 0 and a standard deviation of 1. The resulting "cleaned" light curves are all stored to a separate database table.

Feature Extraction. Here cleaned LCs are run through the extraction library using multiprocessing. This is a computationally expensive step. For instance, on a laptop computing all the features for a light curve takes roughly a second for every 100 data points in the time series. Feature vectors are, again, stored in a separate "features" database table.

Feature Postprocessing. On rare occasions, feature values are undefined (ie. nan, infinity). In such situations I elected to impute a value of 0. Also, at this juncture, there is the option to perform feature standardization, normalizing the feature values to have mean 0 and standard deviation of 1.

Serialization. For both the supervised and unsupervised pipelines the winning model is serialized to disk. Additionally, all model hyperparameters and relevant experimental parameters are saved to disk in a separate metadata file.

Supervised Pipeline

Model Selection and Evaluation. For the supervised learning case the training set is used to perform *k*-fold cross validation on various model hyperparameters. The implementation uses `sklearn`'s `GridSearchCV` function to perform the search. Because the datasets have imbalanced classes, the weight-averaged F1 score is used to determine the best model. Some other metrics are also calculated for reference: accuracy, micro-averaged F1, macro-averaged F1, and the confusion matrix. Next the held-out test set is used to evaluate the winning model using the aforementioned metrics.

Unsupervised Pipeline

Model Selection and Evaluation. The unsupervised evaluation phase tests several experimental variables including the number of components used in dimensionality reduction and the number of clusters assumed by the clustering algorithms. For each dimensionality reduction condition, I tested clustering algorithms (possibly for a range of clusters). I evaluate the clusters produced using several "external" (using class labels) metrics and two "internal" (using no class label information) metrics.

11 https://github.com/lsst-epo/light_curve_ml

Note the external metrics are only possible when the datasets have been labeled and the labels are not used by the clustering algorithms at all. Sklearn provides several external unsupervised metrics including adjusted mutual information, adjusted rand score, Fowlkes-Mallows score, and the V-measure with its intuitive components homogeneity and completeness. For completeness, I compute all these measures in the implementation. The V-measure however, does not normalized with regards to random labeling. Since the number of classes may be more than 10 (e.g., MACHO has 12), I therefore chose to evaluate clustering performance based on adjusted mutual information, which is normalized against chance. Also, for completeness, I included the internal metrics Calinkski-Harabaz and Silhoutte Score but did not make major decisions based on them.

Experiments

The following experiments were run on AWS EC2 instances (t2.2xlarge) having 8 CPU cores to leverage multi-core processing.

Supervised learning experiment.

1. MACHO Dataset. I tested supervised classifier learning on the MACHO dataset described earlier. I explored the following key random forest hyperparameters: 1) `n_estimators`, from 50 to 1600, steps of 50, 2) `max_features`, from 5 to 30, steps of 2, 3) `min_samples_leaf`, 1-3, and 4) `criterion` both "gini" and "entropy." The baseline performance I obtained was 0.863 weight-average F1. The best hyperparameters found are given in Table 1. I additionally tested the best hyperparameter set found for the baseline test in two additional tests. The LC standardized test differed by taking all light curves and standardizing the magnitudes and magnitude errors to mean 0 and std 1 as suggested by [5]. The LC and feature standardization test additionally standardized each feature to have mean 0 and std 1 across the dataset. The results were not appreciably better for these additional manipulations, perhaps due to the ability of random forests to perform well regardless of feature range.

Description	**Weight-Averaged F1**	**Estimators**	**Max features**	**Min samples leaf**	**Criterion**
Baseline	0.863	1500	28	2	entropy
LC standardized	0.837	1500	28	2	entropy
LC and features standardized	0.833	0	28	2	entropy

Table 1. Supervised Classifier Learning Results for MACHO Dataset.

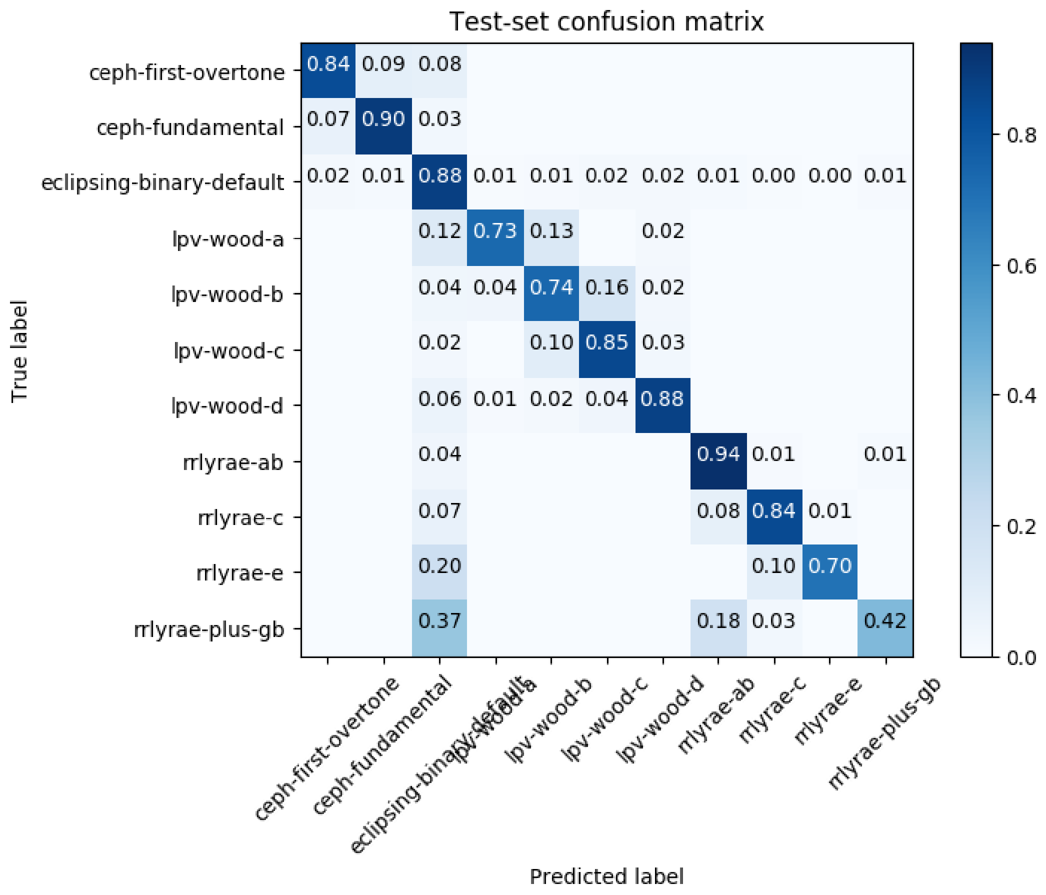

Figure 1. Confusion Matrix for Supervised Classifier on MACHO Dataset.

Feature Name	**Importance**
PeriodLS	0.251
Mean	0.200
SlottedA_length	0.101
Autocor_length	0.045
Freq1_harmonics_amplitude_1	0.032
StetsonK_AC	0.026
Beyond1Std	0.023
CAR_tau	0.021
CAR_mean	0.016
Psi_eta	0.015

Table 2. Top Ten Features by Importance for MACHO-trained Random Forest Classifier.

2. OGLE3 Dataset. For the OGLE-3 I explored the same hyperparameters over similar ranges. Only the `n_estimators` hyperparameter value changed from MACHO. The results are given in Table 3 and the confusion matrix in Figure 2.

Description	Weight-Averaged F1	Estimators	Max features	Min samples	Criterion
Baseline	0.997	700	28	2	entropy

Table 3. Supervised Classifier Learning Results for OGLE-3 Dataset.

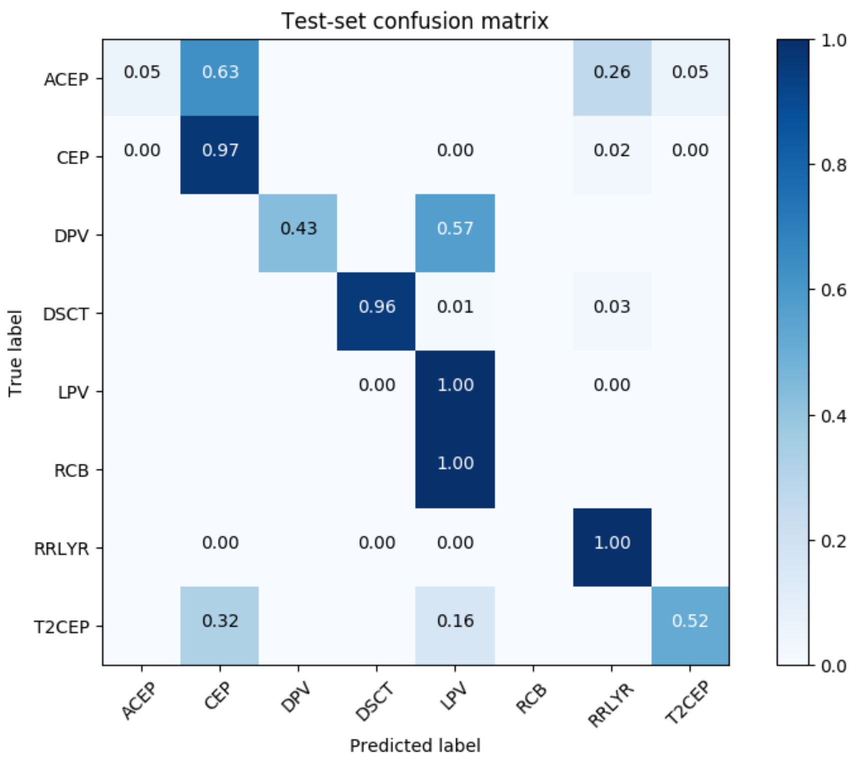

Figure 2. Confusion Matrix for Supervised Classifier on OGLE-3 Dataset.

Unsupervised learning experiment. I explored the feasibility of performing unsupervised learning on the MACHO dataset. In particular I tried out using dimensionality reduction on the feature vectors followed by some appropriate clustering algorithms. Clustering algorithms do not do well on high dimensional data and the feature vectors have length 63. I tested out two dimensionality reduction algorithms: Principle Components Analysis (PCA), which makes use of class labels, and the Linear Discriminant Analysis (LDA), which does not. Both algorithms have a *components* hyperparameter specifying the desired reduced dimension. I test a range of components from 3 to 19.

For clustering algorithms there are several choices. Hierarchical methods tend to assume that objects closer together are more related than objects far apart. Centroid-based methods represent clusters by a central vector. Given the hierarchical nature of some of the categories and the generality of centroid-based approaches I decided to test out agglomerative clustering and mini-batch *k*-means clustering. Through prior testing I settled on a reasonable set of hyperparameters for the these two algorithms, e.g., I determined Ward linkage produced the best results for agglomerative clustering.

In order to keep the number of experimental variables limited and manageable I restricted the number of clusters assumed by the clustering algorithms to 12, which is number of classes in the MACHO dataset. The best result using PCA reduction occurred with 17 components and agglomerative clustering, where the adjusted mutual information (AMI) score was 0.350. The best result using LDA reduction occurred with 10 components and agglomerative clustering, where AMI was 0.422. Note AMI can have a maximum value of 1.0 corresponding with perfect performance.

I also tested combining PCA and LDA together for a two-step dimensionality reduction process.

I tried both PCA then LDA and LDA then PCA. In both cases, for the first algorithm, I tried a range of components from 4 up to 19. Define as the number of components used for the first algorithm by *c1*. For the second algorithm I also tried a range of components from 3 to *c1 – 1*. The best result obtained for the two-step experiment was an AMI of 0.42237 which occurred for an LDA reduction to 11 followed by a PCA reduction to 10. This score is comparable to the best LDA-only result.

Conclusions

Overall these limited supervised learning experiments showed that the random forest classifier with the Richardson et al. features did quite well on the MACHO and OGLE-3 dataset. There was little effort required in adapting the classifier from MACHO to OGLE-3. While performance was acceptable, future work in this area may consider encoder-decoder recurrent neural networks to boost performance [5].

The results were less promising for unsupervised learning. The approaches I tested, mini-batch *k*-means and agglomerative clustering (using Ward linkage) did not do particularly well for unsupervised clustering on the MACHO dataset. The best baseline score I received was around 0.2 AMI. I found that LDA dimensionality reduction helped improved this score to around 0.422 AMI. Combining LDA with PCA did not yield appreciable differences. Future work in this area might consider training with a larger dataset. To what extent would additional data improve performance?

Bibliography

[1] Richards, J. W. *et al*.. On machine-learned classification of variable stars with sparse and noisy time-series data. *The Astrophyiscal Journal* **733**, 1 (2011).

[2] Nun, I. *et al.* FATS: Feature Analysis for Time Series. *ArXiv e-prints* (2015). 1506.00010.

[3] Nun, I., Pichara, K., Protopapas, P. & Kim, D.-W. Supervised Detection of Anomalous Light Curves in Massive Astronomical Catalogs. *Astrophysical Journal* **793**, 23 (2014). 1404.4888.

[4] Kim, D.-W. & Bailer-Jones, C. A. A package for the automated classification of periodic variable stars. *Astronomy & Astrophysics* **587**, A18 (2016).

[5] Naul, B., Bloom, J. S., Pérez, F., & van der Walt, S. A recurrent neural network for classification of unevenly sampled variable stars. *Nature Astronomy* **2**, 151–155 (2018).

[6] Drake, A. J. *et al.* The Catalina Surveys Periodic Variable Star Catalog. *The Astrophysical Journal Supplement Series* **213**, 9 (2014).

[7] Alcock, C., Allsman, R. A., Alves, D., Axelrod, T. S., Becker, A. C., Bennett, D. P. The MACHO Project LMC Variable Star Inventory.V.Classification and Orbits of 611 Eclipsing Binary Stars. *Astronomical Journal* **114**, p.326 (1997).

[8] Jurić, M. et al. Large Synoptic Survey Telescope (LSST) Data Products Definition Document. Retrieved from https://github.com/lsst/LSE-163 (2018).

[9] MACHO project data services. Retrieved from http://macho.nci.org.au/