Kaggle Competition
PLaStiCC Astronomical Classification Challenge

Author
Ioannis Prapas

CONTENTS

1 Introduction 3
1.1 PLaStiCC Astronomical Classification competition 3
1.2 The data 3
1.2.1 Light Curve time-series 3
1.2.2 Metadata 4
1.2.3 Training/Test set difference 4
1.3 Class distribution 6
1.3.1 Training set 6
1.3.2 Test set 6
1.4 Evaluation Metric 7
2 My Approach 7
2.1 General Idea 7
2.2 Feature Calculation 7
2.2.1 Time width features 8
2.2.2 Flux features 8
2.2.3 Color features 8
2.2.4 Flux-flux_err ratio features 8
2.2.5 Absolute Magnitude 8
2.3 Feature Selection 9
2.3.1 Unused features 9
2.3.2 Feature importance and overfitting elimination 9
2.4 Training phase 9
2.5 Unknown class predictions 11
3 Team Merge 11
4 Conclusion and Contribution 12
5 References 13
6 APPENDIX A - Light Curve Visualizations 14

1 INTRODUCTION

This report aims to present my experience participating in an astronomical object classification challenge[1] hosted by Kaggle and Large Synoptic Survey Telescope (LSST)[2]. Kaggle is the most popular data science competition platform, in which people from all over the world gather to compete in data science challenges.

1.1 PLaStiCC Astronomical Classification competition

The Photometric LSST Astronomical Time-series Classification Challenge (PLaStiCC) [1] is a competition, in which participants are asked to classify simulated astronomical time-series data. These simulations are based on what is expected to come from the LSST, which is now being build high in the deserts of northern Chile on a mountain called Cerro Pachon. When put in place, it will use an 8-meter telescope equipped with a 3-billion-pixel camera to image the entire Southern sky roughly every few nights and over a ten-year duration. The influx of data will be unprecedented, so LSST is asking Kaggle Data scientists to the rescue.

Important note: Every participant is allowed at most 5 submissions per day.

1.2 The data

The time-series data of this challenge are called light curves and are the result of difference imaging: two images are taken of the same region on different nights and then the images are subtracted from each other and the flux (measurement of brightness) of the light source is computed. Light curves for each object come in six different passbands, which include ultra-violet, optical and infrared regions of the light spectrum. The challenge is to analyze the time-series data along with some metadata of the astronomical sources and determine a probability that each object belongs to each of 15 classes, 14 of which are present in the training set. There is one unknown class to detect, as LSST is expected to find some astronomical sources, never observed before.

1.2.1 Light Curve time-series

The time-series are given in a table format with the following columns:

object_id: Primary key of the time series. (Will be used to join with the metadata table)

mjd: the time in Modified Julian Date (MJD) of the observation. The MJD is a float number, representing the number of days from midnight on November 17, 1858.

passband: The specific LSST passband integer, such that u, g, r, i, z, y = 0, 1, 2, 3, 4, 5 in which it was viewed.

flux: the measured flux (brightness) in the passband of observation as listed in the passband column.

flux_err: the uncertainty on the measurement of the flux

detected: If detected equals 1, the object's brightness is significantly different at the 3σ level relative to the reference template. Otherwise, it is 0.

[1] https://www.kaggle.com/c/PLAsTiCC-2018
[2] https://www.lsst.org/

1.2.2 Metadata

object_id: the Object ID, unique identifier (given as int32 numbers).

ra: right ascension, sky coordinate: longitude, in degrees.

decl: declination, sky coordinate: latitude, in degrees.

gal l: Galactic longitude, in degrees.

gal b: Galactic lattitude, in degrees

hostgal specz: the spectroscopic redshift of the source. This is an extremely accurate measure of redshift, provided for the training set and a small fraction of the test set.

hostgal photoz: The photometric redshift of the host galaxy of the astronomical source. While this is meant to be a proxy for hostgal specz, there can be large differences between the two and hostgal photoz should be regarded as a far less accurate version of hostgal specz. This is calculated by the providers and the way it is calculated is not provided.

hostgal photoz err: The uncertainty on the hostgal photoz

distmod: The distance (modulus) calculated from the hostgal photoz since this redshift. Computing it requires knowledge of General Relativity, and assumed values of the dark energy and dark matter content of the Universe, as mentioned in the introduction section.

mwebv = MW E(B-V): this 'extinction' of light is a property of the Milky Way (MW) dust along the line of sight to the astronomical source and is thus a function of the sky coordinates of the source ra, decl. Fluxes are corrected from this effect.

target: The class of the astronomical source. This is provided in the training data. Correctly correctly assigning classification probabilities to the test objects is the goal of the challenge.

ddf: A Boolean flag to identify the object as coming from the DDF survey area (with value ddf = 1 for the DDF). While the DDF fields are contained within the full WFD survey area, the DDF fields have significantly smaller uncertainties, given that the data are provided as additions of all observations in each night. Objects in these DDF patches will have light-curve points that are extremely well determined and therefore have small errors in flux

1.2.3 Training/Test set difference

The training data follow the description above and have the properties and light curves of a set of 7848 astronomical sources and are meant to represent the brighter objects for which obtaining expensive spectroscopy is possible. The test data represent all the data for which no spectroscopy is given, and is a much larger set of around 3.5 million objects. Therefore, the test data have 'NULL' entries for the hostgal specz column for all but a few percent of object in the test data. Of course, the target column is 'NULL' for all test data. Moreover, according to the organizers, the training data properties are non-representative of distributions of the test data set. The training data are mostly composed of nearby, low-redshift, brighter objects while the test data contain more distant (higher redshift) and fainter objects. Therefore, there are objects in the test data that do not have counterparts in the training data.

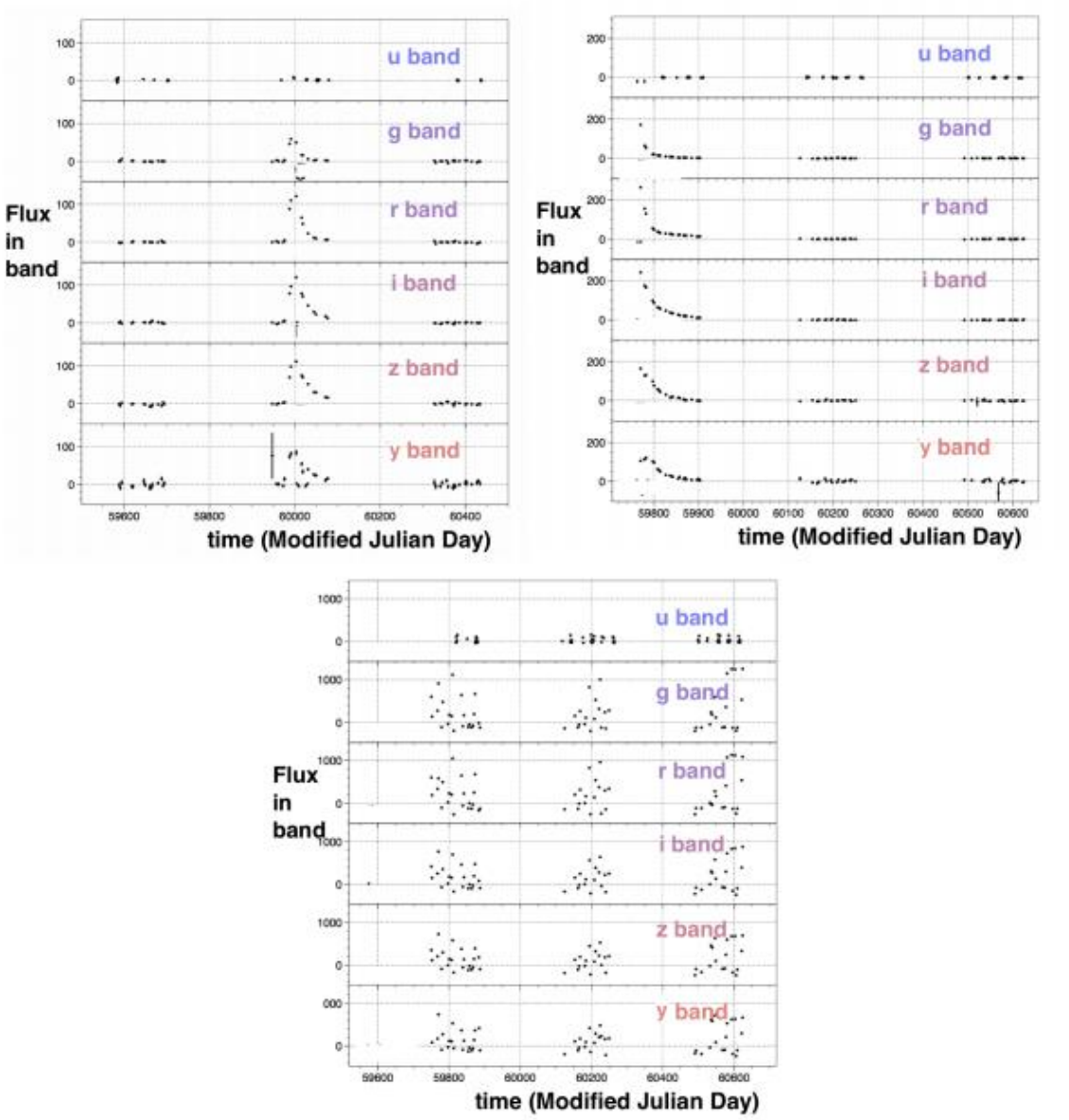

Figure 1: Example light curves included in the dataset. Taken from [1]

1.3 Class distribution

1.3.1 Training set

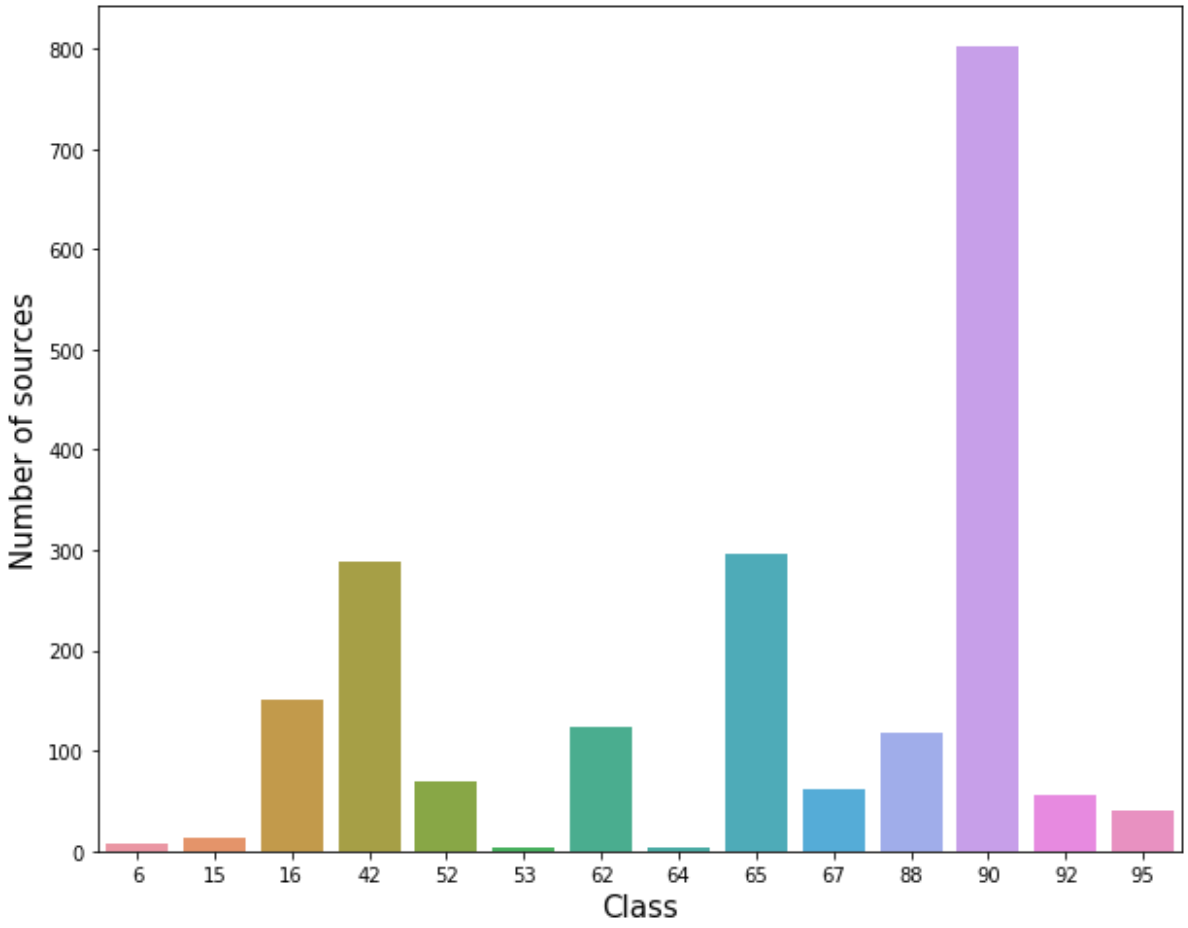

Figure 2: Training set distribution figure[i]

As we can see in Figure 1, the class distribution is highly imbalanced. Class 90 is by far the majority class in the training set, while for classes 53, 64 6 and 15 we have only very few examples.

1.3.2 Test set

The class distribution on the test set is not given, but the Kaggle community tried to find out by probing the leaderboard. This means, checking your submission score, when setting your submission to predict only one class at a time. The results for this was that every class was distributed equally, except for class_64, class_15 and class_99 (unknown class), which were almost double in size. One caveat here is that the leaderboard score is calculated based on the 33% of the data and only when the competition ends, one can know her final score.

1.4 Evaluation Metric

$$\text{Log Loss} = -\left(\frac{\sum_{i=1}^{M} w_i \cdot \sum_{j=1}^{N_i} \frac{y_{ij}}{N_i} \cdot \ln p_{ij}}{\sum_{i=1}^{M} w_i} \right)$$

where N is the number of objects in the class set, M is the number of classes, $\boldsymbol{ln}$ is the natural logarithm, $\boldsymbol{y_{ij}}$ is 1 if observation i belongs to class j and 0 otherwise, $\boldsymbol{p_{ij}}$ is the predicted probability that observation i belongs to class j. $\boldsymbol{w_i}$ is the weight of class i.

The competition uses a weighted multi-class logarithmic loss. The effect is such that each class is roughly equally important for the final score. Details about the choice of this metric are given in [2].

2 MY APPROACH

In the course of the competition my approach developed a lot. To name a few of the techniques I used that didn't work out as well as my latest approach, I tried using Support Vector Machines, Neural Networks, Principal Component Analysis for Dimensionality reduction. In the following sections, I will be presenting the latest version of my approach, which uses Gradient Boosting Machines (LighGBM[3] implementation) [3] as a model.

2.1 General Idea

My approach to the problem was to calculate meaningful features from the light curves and use these features to train a LightGBM classification model. Meaningful features in the context of classification are the ones that help a model distinguish between different classes. To find them, I had to look at light curves of different objects and find out how they differ from each other. A visualization of the different curves lies in the APPENDIX A section of the report.

2.2 Feature Calculation

Early in the competition, I came to the realization that the test set is massive and calculating features on it would take a lot of time. To tackle this, I decided to incrementally add features to my model and always store the calculation of any new features.

The feature calculation has been guided by:

- looking at the light curves (See some examples in APPENDIX A of the report)
- researching for useful features for time-series
- researching for useful features for light-curves
- kernels and discussions in the Kaggle platform
- the cross-validation score and leaderboard score

In total, I calculated more than 500 different features, only a small subset of which was eventually useful. The calculation of the features was aided in some cases by the tsfresh [4]

[3] https://github.com/Microsoft/LightGBM

python library. Following are some of the different features I calculated and the majority of which, turned out to be useful.

2.2.1 Time width features

- *mjd_diff_detected*: Time difference between the last detected flux and the first one. This feature is good to differentiate between periodic and aperiodic events.
- *Mjd_width_max_decay div_{N}*: Time of decay of a light curve from maximum value to N% of maximum

2.2.2 Flux features

- *Slope_after_max{i}*: slope term of linear fit after maximum
- *Slope_before_max{i}*: slope term of linear fit after maximum
- *Intercept_before_max{i}*: intercept term of linear fit before maximum value for passband i
- *Intercept_after_max{i}*: intercept term of linear fit before maximum value for passband i
- *Time-Series Autocorrelation*
- *Fourrier Coefficients*
- Basic statistics per passband and in total: *maximum, minimum, mean, median, skewness, kurtosis*.

2.2.3 Color features

Combination of maximum and intercept after max per passband:

```
1. for i in range(6):
2.     for j in range(i+1, 6):
3.         df['{0}{1}__feature'.format(i,j)] = df['{0}__feature'.format(i)] / df['{0}__
   feature'.format(j)]
```

2.2.4 Flux-flux_err ratio features

Basic statistics per passband and in total: maximum, minimum, mean, median, skewness, kurtosis.

2.2.5 Absolute Magnitude

Absolute magnitude during maximum flux is a distinguishing term between different types of astronomical objects.

Its calculation is defined as follows, according to Wikipedia[4]:

$$M = -2.5 * \log_{10}\left(\frac{F_{max}}{F_0}\right) - distmod$$

From this, we don't know the F_0 term which is the zero-point Magnitude for a given filter, which means that our calculated magnitude, will differ from the actual one by a constant value per filter. Also, the $F_{max} = flux_{maximum}$ that we calculate is only an approximation of the actual maximum. Still this becomes a very useful feature for our model.

[4] https://en.wikipedia.org/wiki/Absolute_magnitude

2.3 Feature Selection

2.3.1 Unused features

As the universe is isotropic in every direction and the fluxes have been corrected for differences in dust concentration and atmospheric effected, I do not expect the following angle features to be helpful to my model and therefore I remove them from the training set:

ra, decl, gal l, gal b

Moreover, the spectroscopically calculated redshift (**hostgal_specz**) could be a very useful feature for my training. However, it is very rarely present on the test set, thus I must also not use it. At one point, I had the idea to regress this value using all the other features, but due to lack of time, I dropped the idea.

2.3.2 Feature importance and overfitting elimination

LGBM is a tree-based model, and after training it, you can calculate the importance of each feature, during tree splitting. My first rough approach to feature selection was to take the N most important features. After having a more limited number of features, I tried to remove the features that overfitted my training set. This means that I removed the features that gave a better local score, without adding anything to my leaderboard score. LGBM is invariant to correlated features, so I didn't try to eliminate correlations.

2.4 Training phase

For the training phase, I do a 5-Fold Cross Validation and train 5 models. In the testing phase, I am using an averaged prediction of these 5 trained models. Next figure shows the confusion matrix of the cross-validation predictions of my best single LGBM model.

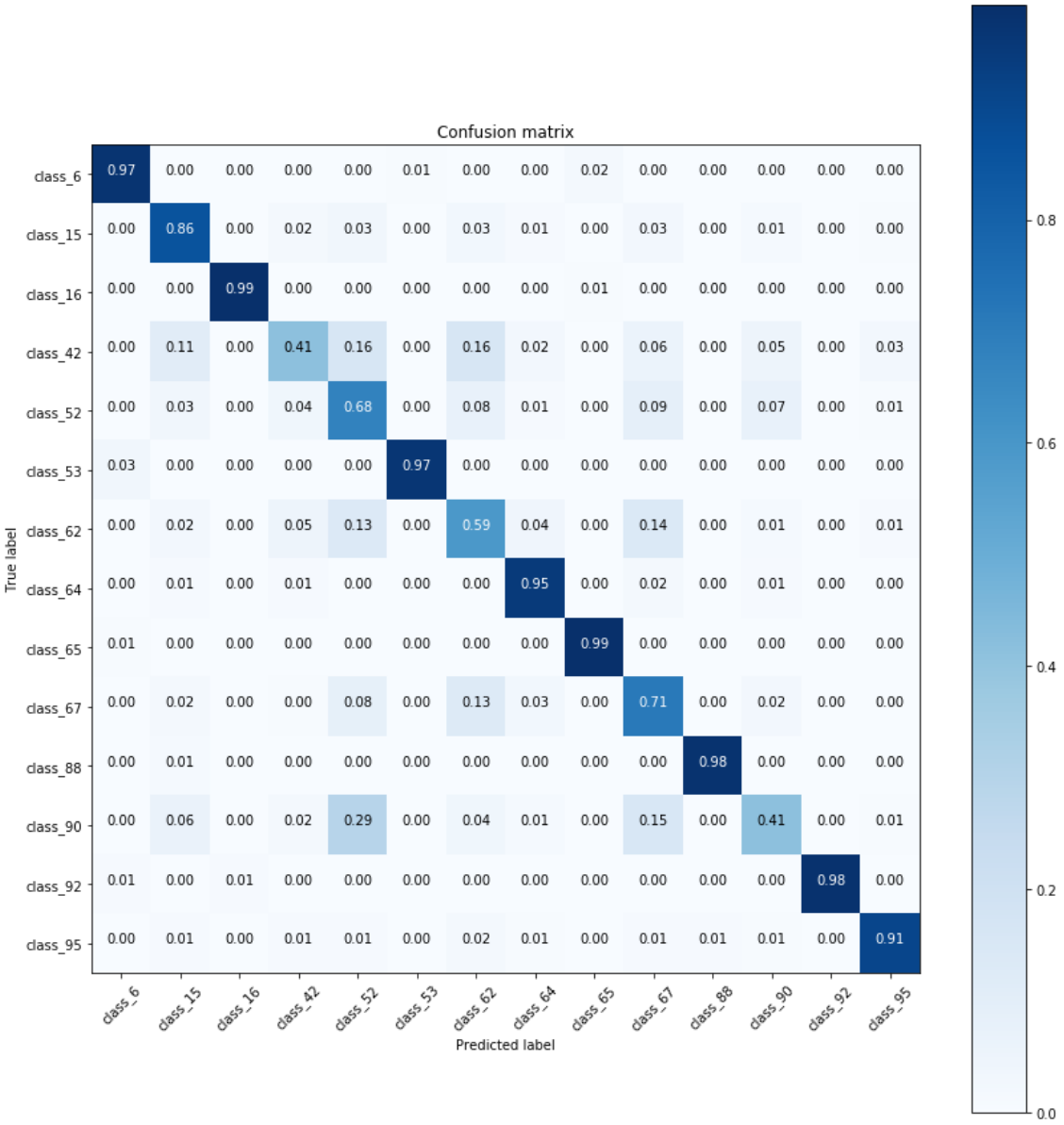

Figure 3 Confusion matrix of cross validation predictions

In the confusion matrix (Figure 3), we can see that there are some hard classes (42, 52, 62, 67, 90) which get confused with each other. A better look at the light curves of those items shows that they consist of aperiodic single events, probably different types of supernovae that burst once and then they fade out. Figure 4 shows two examples of different class 90 objects. In the top one, the event is present in the measurements, while in the bottom example, the characteristic burst-type event is not present and therefore very hard to identify.

Another point to make from the cross-validation confusion matrix is that because of the differences between the training and the test set (discussed in section 1.2.3) in this challenge, it is not safe to trust the cross-validation score so much as it is very easy to overfit.

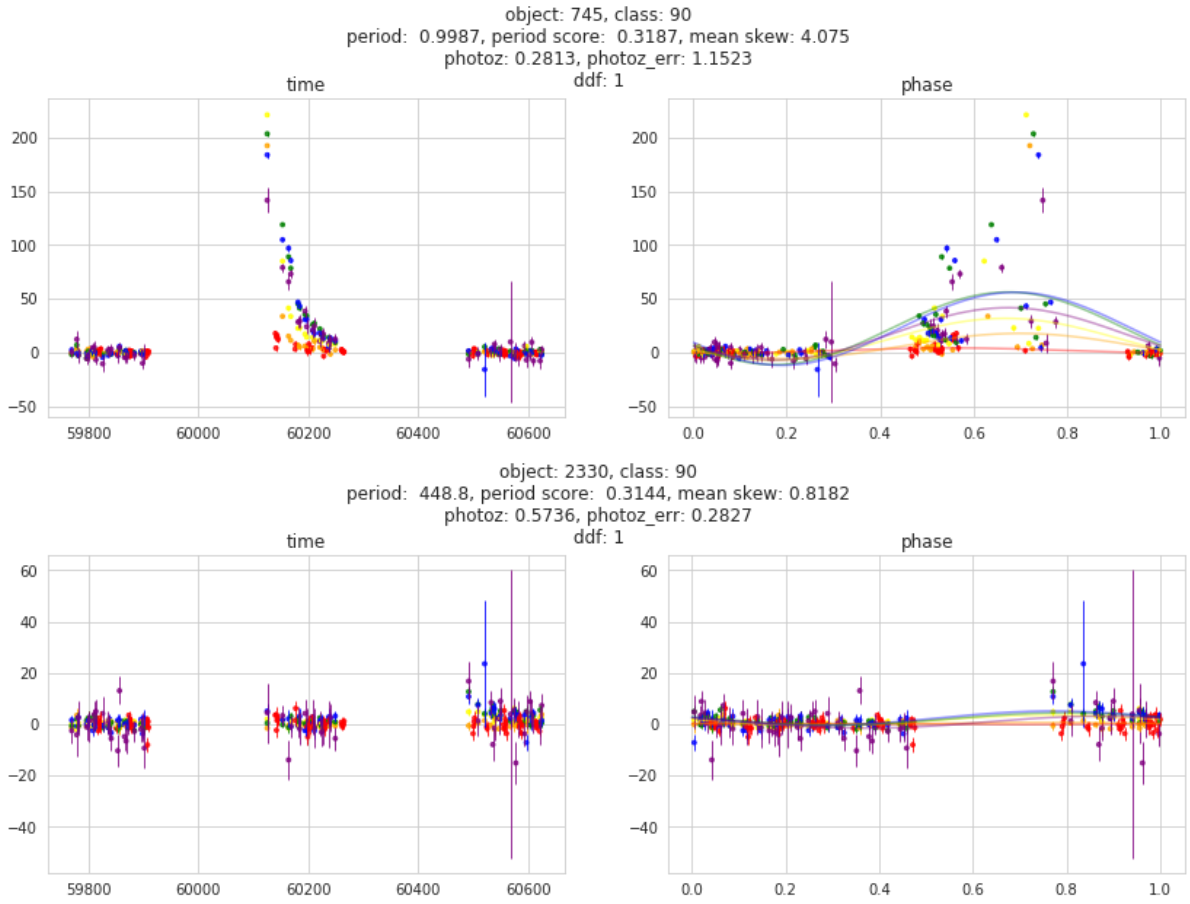

Figure 4 Example of class 90 light curve(left) and frequency fit (right). Top is a common example of single aperiodic event present in the measurements. Bottom shows an example where the single event is not present.

2.5 Unknown class predictions

Class 99 is a class of objects that are not present in the training set and it might be comprised by many kinds of astronomical objects. It is therefore a very hard problem to predict something that you have never seen. This problem is part of the general class of problems called Novelty Detection. Two very common techniques for this is using One-Class SVM or Isolation Forests. After trying these techniques without success, I found out at the Kaggle post section for the competition that not even the leaders in the leaderboard tried to explicitly detect these instances. I ended up, predicting this class by taking the multiplication of the opposite probabilities of an object being part of any of the other classes:

$$P_{class_{99}} = \prod (1 - P_{class_i})$$

3 TEAM MERGE

At a certain point near the end I was out of time and ideas, so I decided to accept the offer of a fellow Kaggler (Max Halford) with similar score to form a team. Then, we added another Kaggler (adityasinha) to the team. Just before teaming up my best score was 0.945, Max's best score was 0.965 and adityasinha's best score was also 0.945 (lower is better). Sadly, I didn't have anything to gain from the features my teammates had calculated, but I helped Adityasinha lower his loss to 0.918, with the time-width features around maximum. I

managed to lower my loss to 0.900 by introducing SMOTE [5] with the python library imblearn [6] to manage the imbalance of classes in the training set. This approach was introduced to me by a public kernel[ii] in kaggle. The good thing is that since we had different approaches, just by averaging the predictions of our best models we got a score of 0.856, which raised as from the 50th to the 16th position in the leaderboard. This was just one week before the competition ending and we finished at the 22nd position with the same score as none of us had time to implement more ideas. We didn't experience any significant shake-up moving from position 21st on the public leaderboard to position 22nd in the private one. This performance earned us a Kaggle silver medal.

4 CONCLUSION AND CONTRIBUTION

In general, this competition has been a huge learning experience for me. More specifically:

- I learned to use the powerful LGBM, a true hammer for data science.
- I learned different techniques to deal with imbalanced data.
- It was my first time dealing with astronomical or time-series data. Researching about the extraordinary stuff that comprise the universe has been truly interesting.
- It was my first Kaggle competition ever and I competed head-to-head with some of the best data scientists.

In the end though, I did not only learn, but I also contributed to the Kaggle community, by actively participating in the forum discussions and publishing my code. These contributions earned me:

1. A **silver medal** for my ranking in the competition
2. A **gold medal** for a high-scoring kernel[5] I published which at the time of writing this report has received 100 upvotes and at has been forked almost 400 times. This has been by far my biggest contribution as it inspired multiple public kernels afterwards and was a serious drive for improving the score of all the teams in the leaderboard.
3. **3 silver and 24 bronze medals** for my contributions in the discussions.

[5] https://www.kaggle.com/iprapas/ideas-from-kernels-and-discussion-lb-1-135

5 REFERENCES

[1] T. Allam Jr, A. Bahmanyar, R. Biswas, M. Dai, L. Galbany, R. Hlo{\v{z}}ek, E. E. O. Ishida, S. W. Jha, D. O. Jones, R. Kessler and others, "The Photometric LSST Astronomical Time-series Classification Challenge (PLAsTiCC): Data set," *arXiv preprint arXiv:1810.00001,* 2018.

[2] A. Malz, R. Hlo{\v{z}}ek, T. Allam Jr, A. Bahmanyar, R. Biswas, M. Dai, L. Galbany, E. Ishida, S. Jha, D. Jones and others, "The Photometric LSST Astronomical Time-series Classification Challenge (PLAsTiCC): Selection of a performance metric for classification probabilities balancing diverse science goals," *arXiv preprint arXiv:1809.11145,* 2018.

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, 2017.

[4] M. Christ, N. Braun, J. Neuffer and A. W. Kempa-Liehr, "Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh--A Python package)," *Neurocomputing,* 2018.

[5] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of artificial intelligence research,* vol. 16, pp. 321-357, 2002.

[6] G. Lema{{\^i}}tre, F. Nogueira and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *Journal of Machine Learning Research,* vol. 18, pp. 1-5, 2017.

[8] R. Kessler, J. P. Bernstein, D. Cinabro, B. Dilday, J. A. Frieman, S. Jha, S. Kuhlmann, G. Miknaitis, M. Sako, M. Taylor and others, "SNANA: A public software package for supernova analysis," *Publications of the Astronomical Society of the Pacific,* vol. 121, p. 1028, 2009.

6 APPENDIX A - LIGHT CURVE VISUALIZATIONS

These visualizations follow a modification of the code of a public Kaggle kernel[iii]. They are handpicked to not be too noisy, so that the main characteristics of each class are visible. Real data are much more complex.

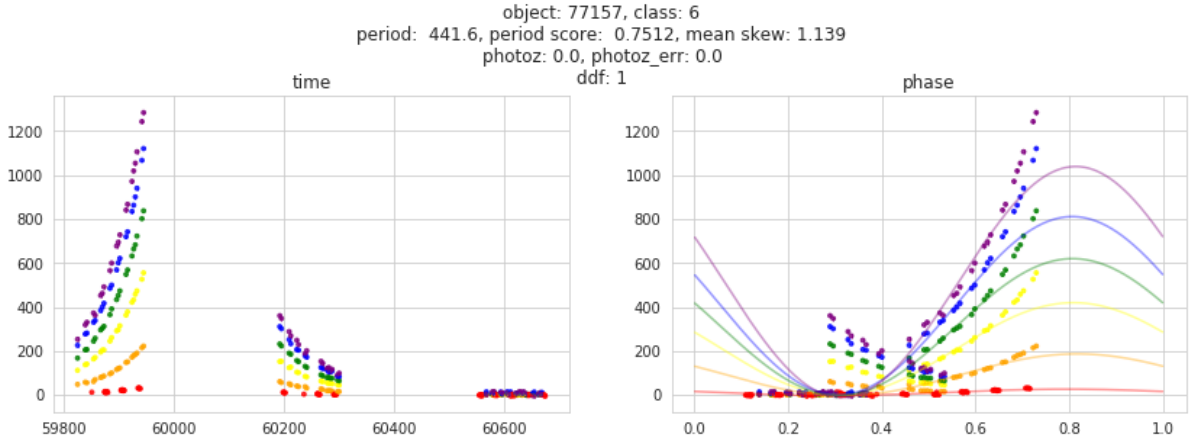

Figure 5: Class 6 object

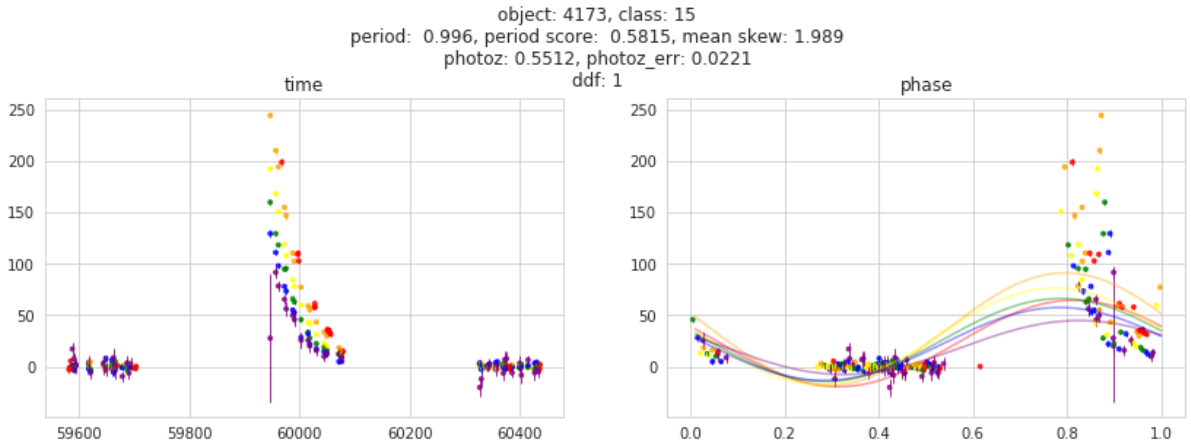

Figure 6: Class 15 object

Figure 7: Class 16 object

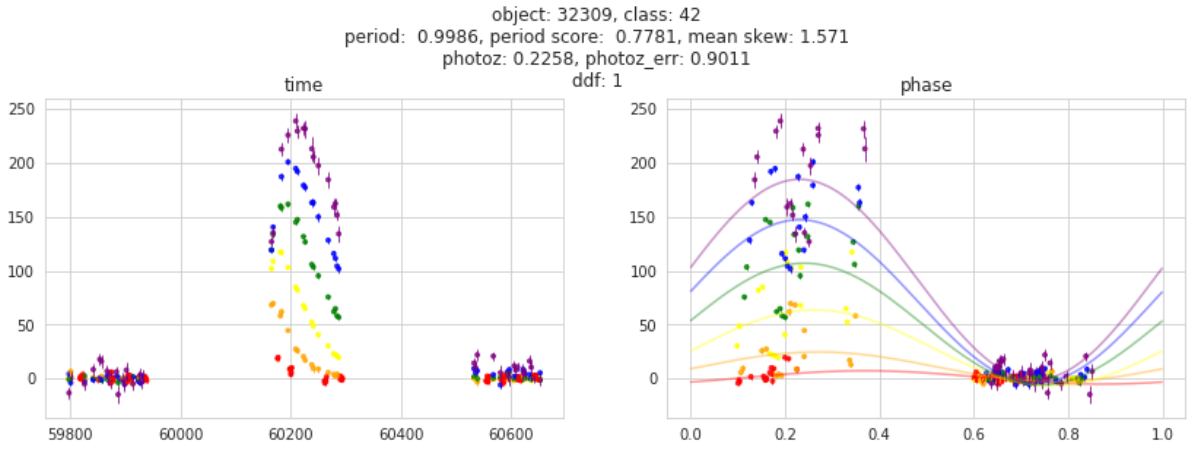

Figure 8: Class 42 object

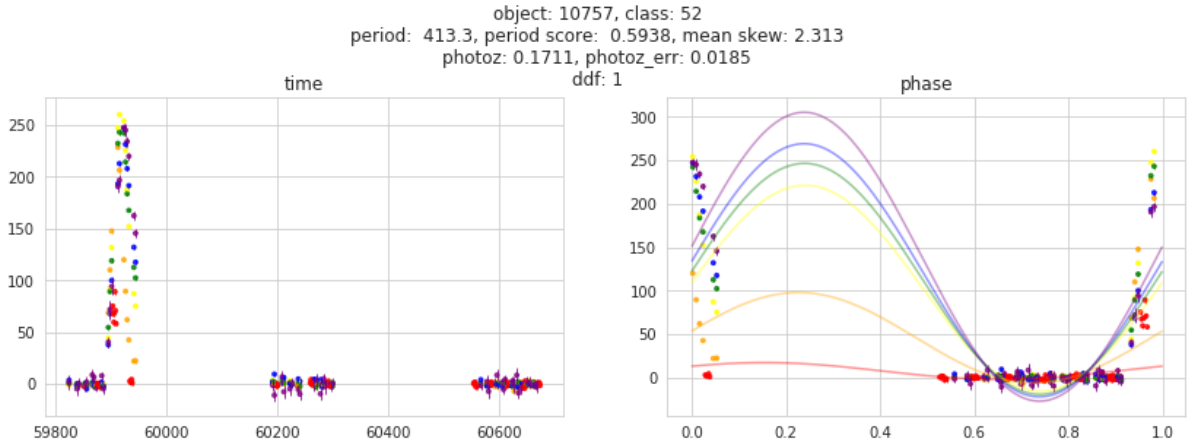

Figure 9: Class 52 object

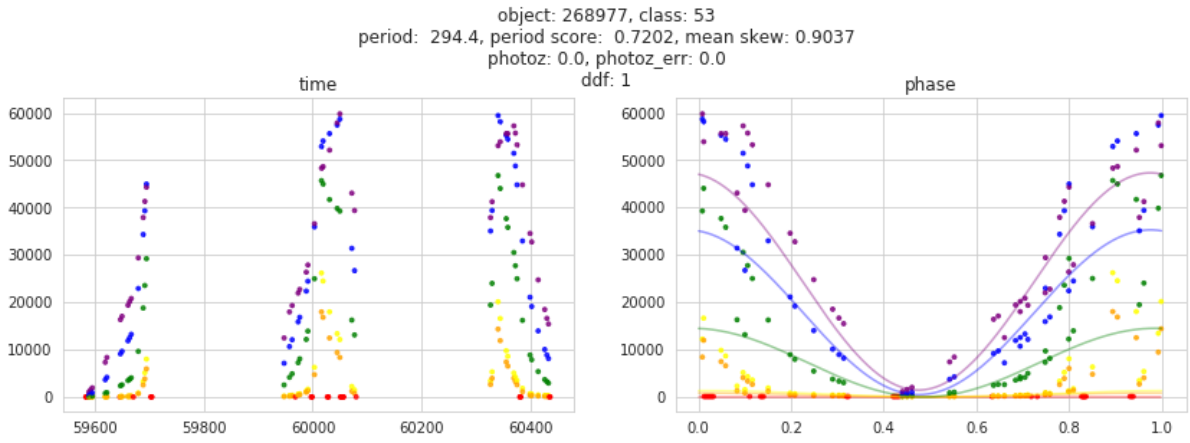

Figure 10: Class 53 object

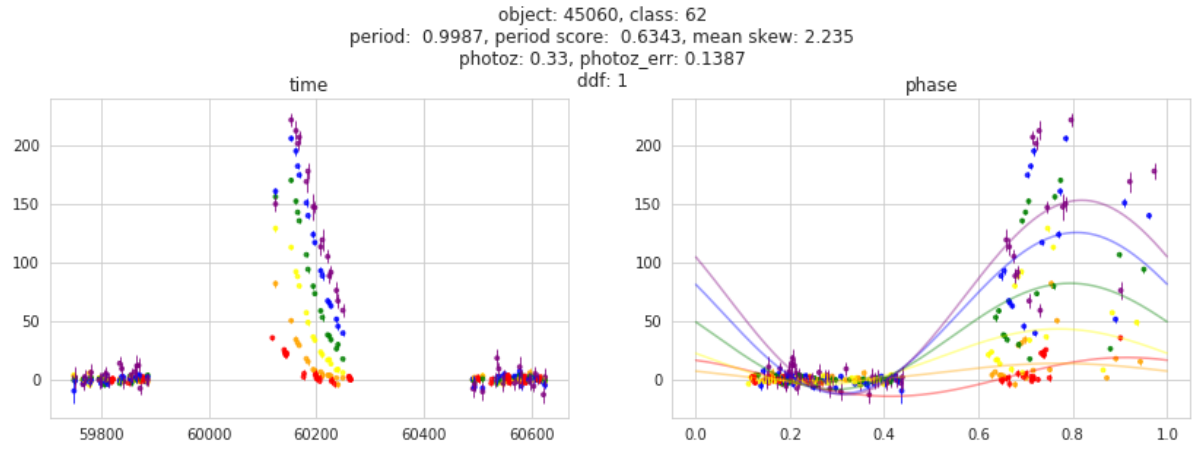

Figure 11: Class 62 object

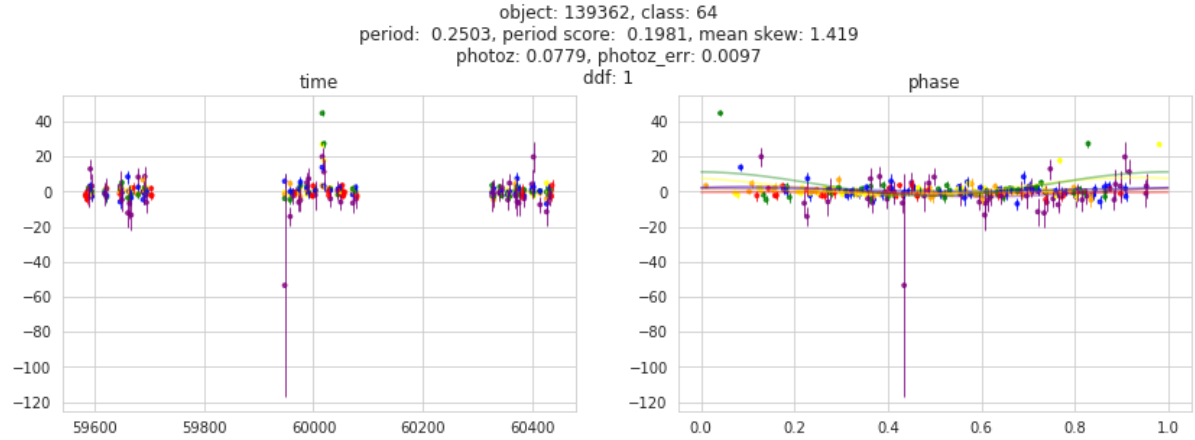

Figure 12: Class 64 object

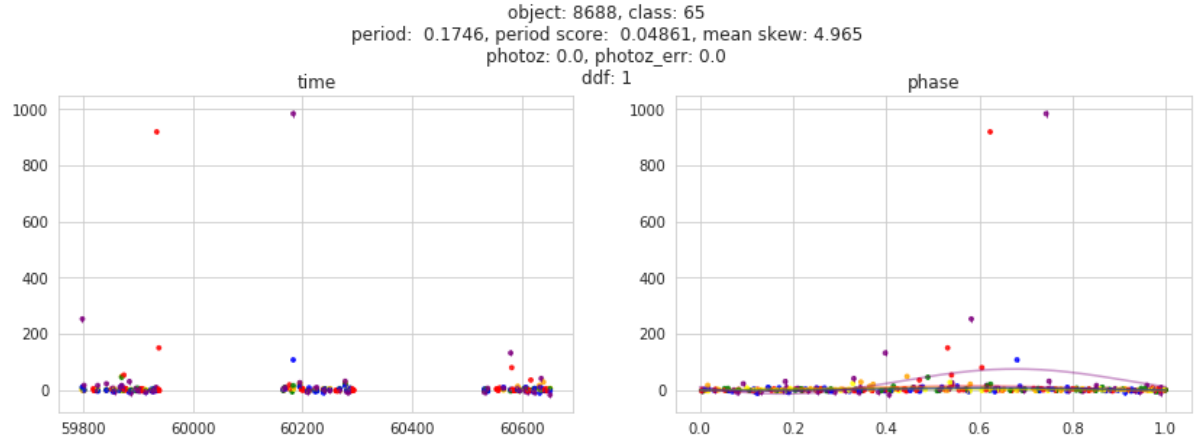

Figure 13: Class 65 object

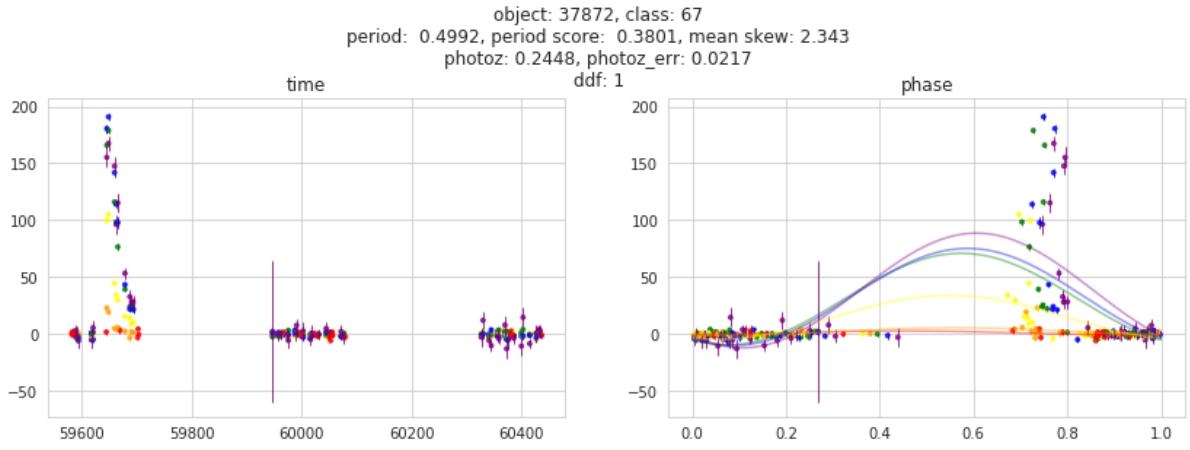

Figure 14: Class 67 object

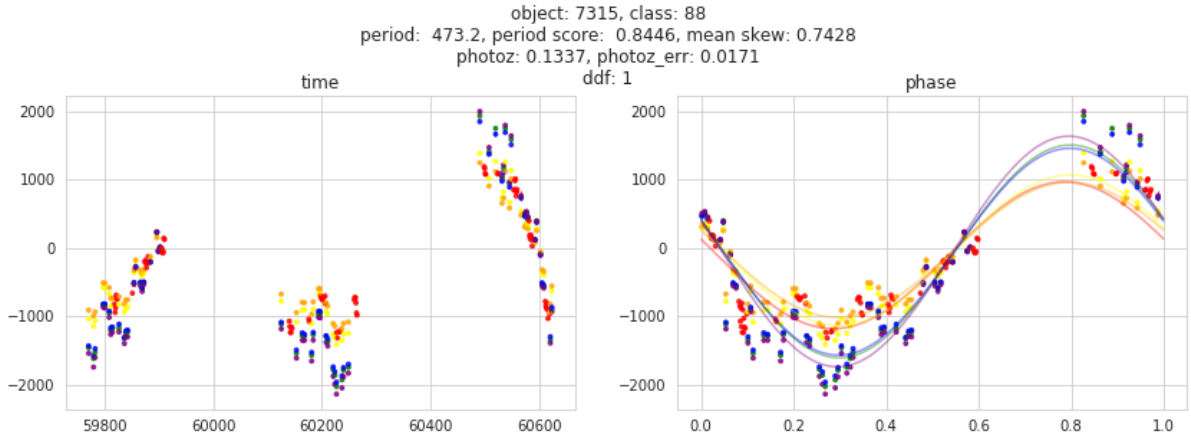

Figure 15: Class 88 object

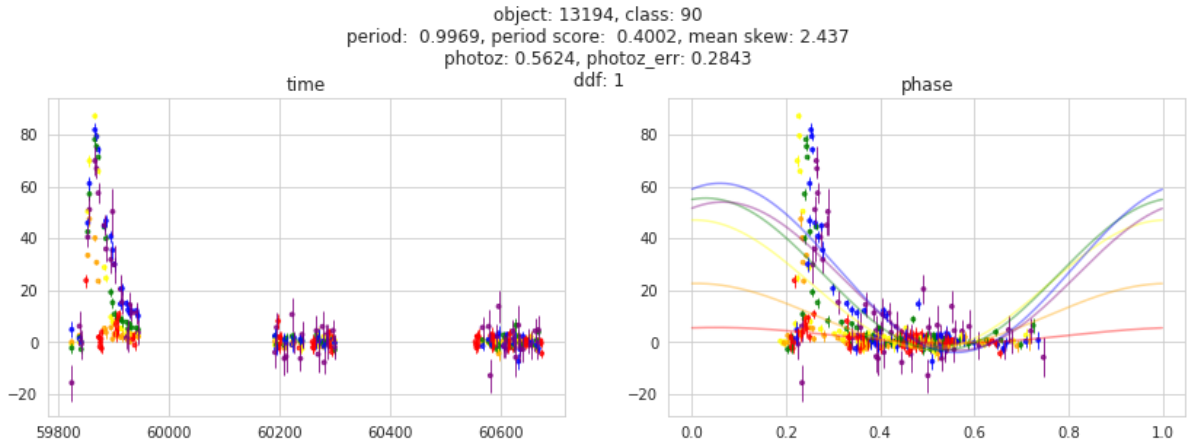

Figure 16: Class 90 object

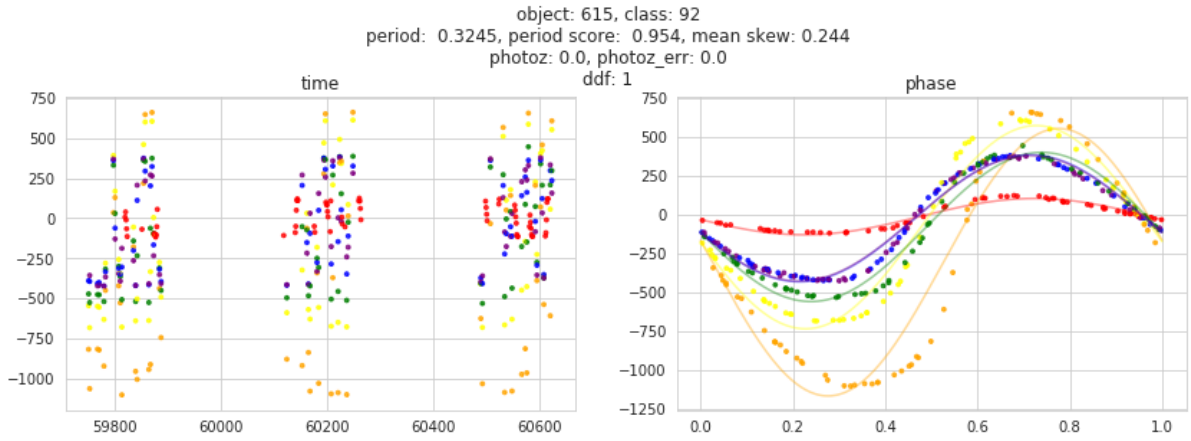

Figure 17: Class 92 object

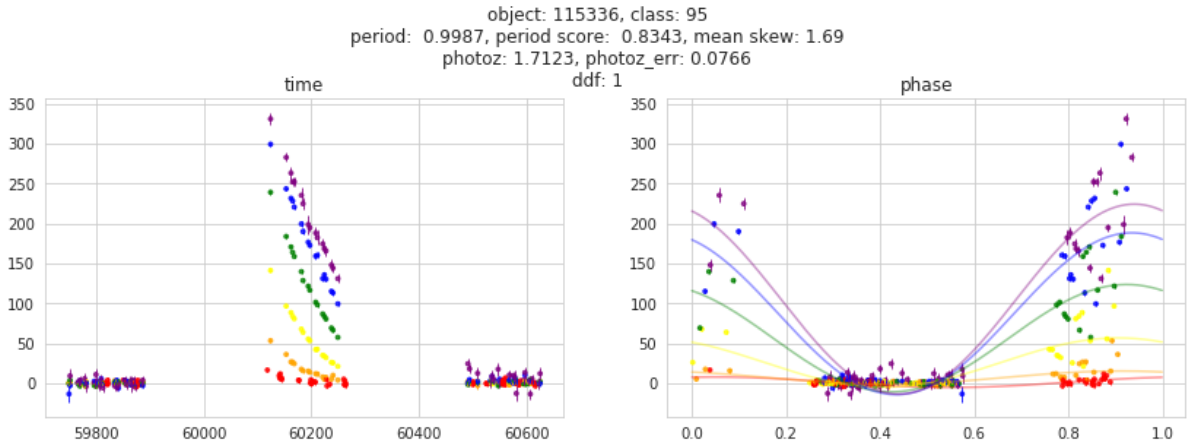

Figure 18: Class 95 object

[i] https://www.kaggle.com/hrmello/dataset-overview-exploration-and-comments
[ii] https://www.kaggle.com/jimpsull/collaboratingwithkagglecommunity-1-037-lb
[iii] https://www.kaggle.com/mithrillion/all-classes-light-curve-characteristics-updated