# DistClassiPy

# A new distance-based classifier to find rare gems in time-domain astronomy.

We developed a method and a package to classify light curves using 18 distance metrics, with:

- State-of-the-art accuracy
- Significant computational gains
- Augmented interpretability
- Customizable to your problem

**Siddharth Chaini**,[1] Ashish Mahabal,[2] Ajit Kembhavi,[3] Federica B. Bianco[1]

[1]University of Delaware, [2]Caltech, [3]IUCAA, Pune

## Introduction

The rise of synoptic sky surveys has ushered in a new era of big data in time-domain astronomy, making data science essential. While tree-based methods (e.g., Random Forest) are the standard in astrophysical classification, the direct use of distance metrics has remained unexplored.

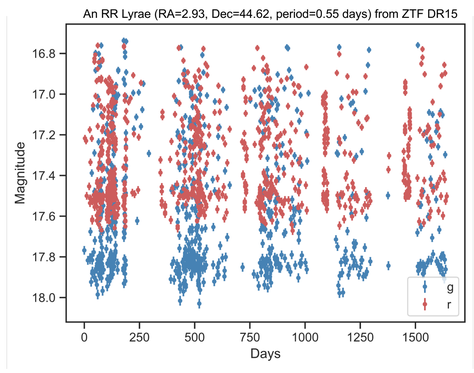
We looked at 18 different distance metrics to classify light curves of variable stars and transients and make recommendations for efficient and physically interpretable classification algorithms.

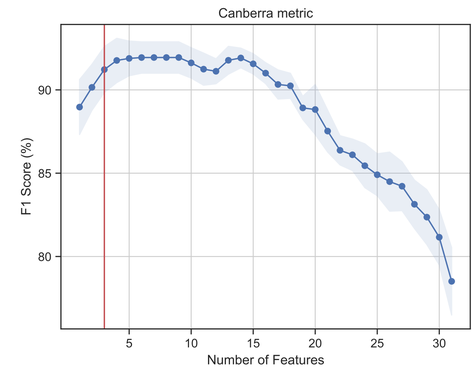
## Data & Feature Extraction

We use light curves from the Zwicky Transient Facility (ZTF) including four types of variable stars: Cepheid (CEP), RR Lyrae (RR), RR Lyrae Type c (RRc), Deta Scuti (DSCT).

We extract 112 features from these light curves using the lc_classifier python package (ALeRCE, Jainaga+2021).

We perform dimensionality reduction tailored to the classification problem using Sequential Feature Selection. This results in a handful set of essential features.



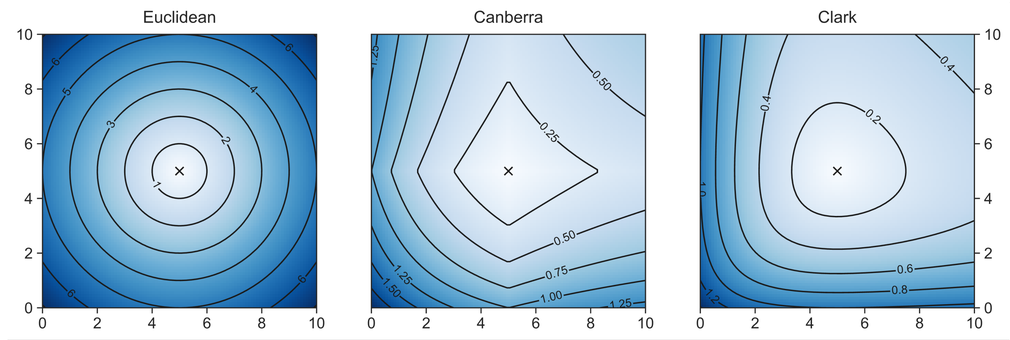




## Distance Metrics

A distance metric is any mathematical function that obeys the following rules:

- $d(x, y) = 0 \iff x = y$ (identity of indiscernibles)
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality)

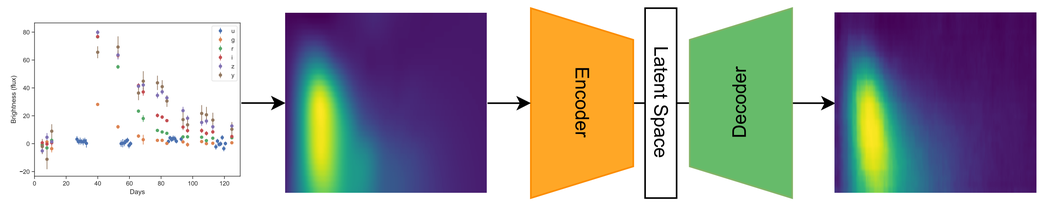
We can use different distance metrics for machine learning!
*e.g.*, Euclidean, Canberra, Clark, etc.




## What's next?

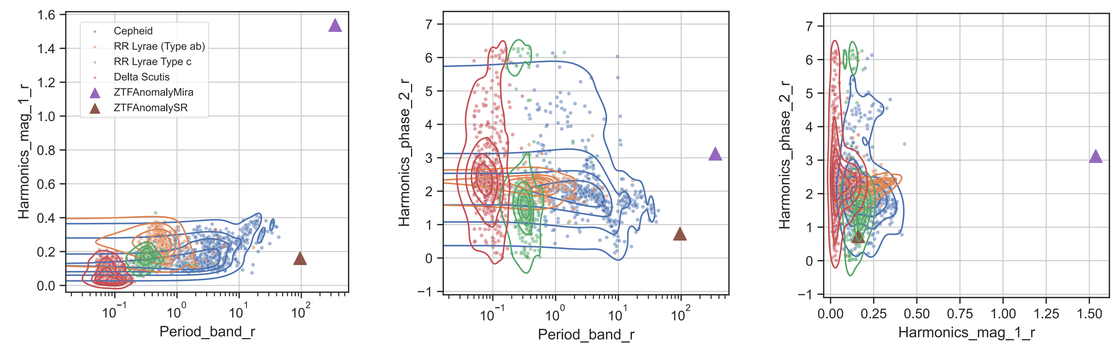
### 1. Expanding DistClassiPy to transients: designing feature extraction

For transient classification, we need to extract different features. Using the *PLAsTiCC* & *ELAsTiCC* (Hlozek+2020; Narayan+2023) datasets, we preprocess the light curves with 2D Gaussian Processes, and generate features with an Autoencoder – the neural network latent space will be our new feature set.




### 2. Hunting for anomalies with DistClassiPy

An anomaly may be far away from all other or close to multiple clusters. Our classifier can be used for anomaly detection: anomalies should have ambiguous classifications sitting at the tail of known distribution families under multiple distance choices.




## Reference

[1] Chaini, S. et al. (2024), arXiv:2403.12120 (under review, A&C).

## Algorithm

**Training:**

1: **for** each class $C$ in the training set **do**
2: **for** each feature $F$ in class $C$ **do**
3: Calculate the median $M_F^C$ of feature $F$ in class $C$.
4: Calculate the standard deviation $\sigma_F^C$ of feature $F$ in class $C$.
5: **end for**
6: Save the median set $\{M_F^C\}$ and standard deviation set $\{\sigma_F^C\}$ for class $C$.
7: **end for**

**Testing**:

1: Choose a distance metric.
2: **for** each test object **do**
3: **for** each class $C$ in the training set **do**
4: **for** each feature $F$ in class $C$ **do**
5: Scale the test object feature by dividing it by $\sigma_F^C$, which was calculated in the training step.
6: Scale the median set $\{M_F^C\}$ features by dividing them by $\sigma_F^C$.
7: **end for**
8: Calculate the distance $D_C$ between the scaled test object and the median set $\{M_F^C\}$ for class $C$, scaled by $\{\sigma_F^C\}$.
9: Save $D_C$ for class $C$.
10: **end for**
11: Choose the class $C_{\min}$ for which the distance $D_{C_{\min}}$ is the smallest.
12: Assign class $C_{\min}$ as the predicted class for the test obkect.
13: **end for**

## Results

### Classification Performance

We evaluated our classifier on 558 objects each across the 4 classes (CEP, RR, RRc, DSCT), for all 18 distance metrics.

| Classifier | $F_1$ Score | Time Taken |
|---|---|---|
| DistClassiPy with Clark | 92% | 0.004 s |
| DistClassiPy with Canberra | 91% | 0.026 s |
| Random Forest | 92% | 0.188 s |

Canberra and Clark performed as well as a Random Forest - but faster!

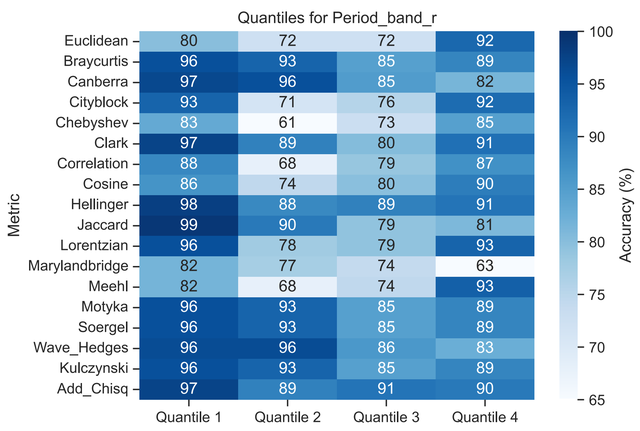
Note: The computational time was computed by training & testing on a mock dataset consisting of 5000 samples and 100 features and was averaged over 5 iterations.

### But DistClassiPy is Customizable

Different distance metrics perform better for different kinds of objects.

For *e.g.*, different metrics perform differently for objects having a different $r$-band period.

So, you can choose the distance metric that's best for your object of interest!




SCAN ME