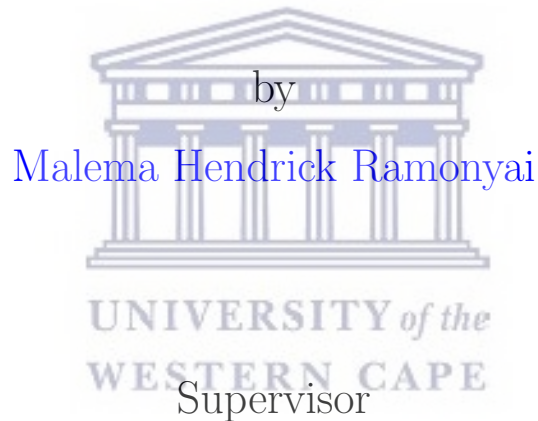


UNIVERSITY OF THE WESTERN CAPE

Application of Anomaly Detection Techniques to Astrophysical Transients



Dr Michelle Lochner

A thesis submitted in fulfillment for the degree
of Master of Science

in the
Department of Physics and Astronomy

February 2022

<http://etd.uwc.ac.za/>

Declaration of Authorship

I, Malema Hendrick Ramonyai, declare that the thesis titled *APPLICATION OF ANOMALY DETECTION TECHNIQUES TO ASTROPHYSICAL TRANSIENTS* is my own work, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged by complete references.



Date:

Signed:

UNIVERSITY *of the*
WESTERN CAPE

Abstract

Master of Science

Application of Anomaly Detection Techniques to Astrophysical Transients

by [Malema Hendrick Ramonyai](#)

We are fast moving into an era where data will be the primary driving factor for discovering new unknown astronomical objects and also improving our understanding of the current rare astronomical objects. Wide field survey telescopes such as the Square Kilometer Array (SKA) and Vera C. Rubin observatory will be producing enormous amounts of data over short timescales. The Rubin observatory is expected to record ~ 15 terabytes of data every night during its ten-year Legacy Survey of Space and Time (LSST), while the SKA will collect ~ 100 petabytes of data per day. Fast, automated, and data-driven techniques, such as machine learning, are required to search for anomalies in these enormous datasets, as traditional techniques such as manual inspection will take months to fully exploit such datasets.

In this work, we aim to answer two questions: 1) how do we generalise the anomaly detection process (i.e., how do we automate the process from data processing to anomaly detection with minimum programming), and 2) how do we personalise interesting anomalies? To do this, we employed unsupervised anomaly detection techniques incorporated in the **ASTRONOMALY** framework, to search for anomalies in optical astrophysical transient and variable light curve datasets (a plot of brightness against time of an astronomical object). **ASTRONOMALY** is a flexible framework designed to generalise the anomaly detection process and personalise interesting anomalies to a human expert. We extend the **ASTRONOMALY** framework to operate with light curve data, and employed two machine learning algorithms: isolation forest (iForest) and local outlier factor (LOF), incorporated in the framework, to search for anomalies in the **MANTRA** and **PLAsTiCC** light curve data. We then used the active learning technique incorporated in **ASTRONOMALY**, to personalise interesting anomalies in both datasets.

We found that iForest detects two types of anomalies in the **MANTRA** data: anomalies triggered by artefacts (bogus) and those triggered by interesting astrophysical processes (e.g., stellar flares). However, active learning is successful in flagging most of the bogus anomalies, which implies that our technique is promising in detecting anomalies in big datasets. We also found that LOF performs better than iForest in detecting anomalies in the **PLAsTiCC** data; however, on average, the performance of both algorithms as measured by both recall and rank weighted scores was subpar. We also found that different feature extractors results in different classes of objects detected as anomalies. Our work highlights the importance of both feature extraction and active learning for anomaly detection in large samples of variable and transient light curves. Lastly, we conclude from our work that the **feets** feature extractor is not ideal for anomaly detection with **PLAsTiCC**-like data.

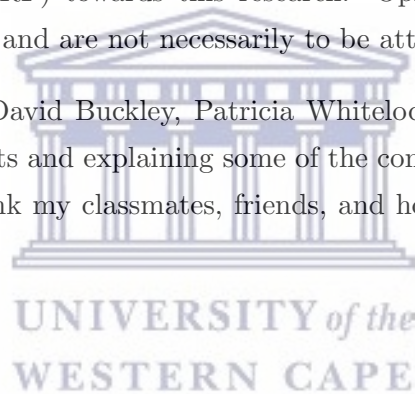
Acknowledgements

This work is dedicated to my parents: Maphefo Annah Ramonyai, and the late Petrus Tawana Ramonyai. You are indeed more than a blessing, and your endless advice, calls, and motivations impacted this work. A special thanks to my siblings (Chula Ramonyai, Evelyn Ramonyai, and Amos Ramonyai) for motivating me to keep pushing even when I was on my least abilities. A special thanks to Thinus and his wife for the advice and words of wisdom.

A special thanks to my supervisor, Dr. Michelle Lochner. This work would not be complete if you did not motivate me to keep going. Thanks for giving me the best guidance and, most importantly, for reviewing my work. I would also like to thank Verlon Etsebeth and Zwido Khangale for reviewing my work. A special thanks to the center for radio cosmology at UWC for funding this work.

I acknowledge support from the South African Radio Astronomy Observatory (SARAO) and the National Research Foundation (NRF) towards this research. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NRF.

A special thanks to Sara Webb, David Buckley, Patricia Whitelock, and Retha Pretorius for their help in analysing some of my results and explaining some of the complicated concepts covered in this work. Finally, I would like to thank my classmates, friends, and housemates for being there when I needed advice.



Preface

The use of machine learning algorithms has emerged dramatically in the past few decades. This is due to the big datasets that come with the developing technology. A group of machine learning algorithms called anomaly detection algorithms are ideal tools to search for anomalies in these big datasets. However, using these techniques is not as simple as it sounds. The challenges I encountered in applying them include choosing the best algorithm for a given dataset and finding the best tool to extract features from the input data.

However, overall, my masters experience at the University of the Western Cape was fun and exciting. I got an opportunity to learn about some of the hot topics in the fourth industrial revolution (machine learning) and astronomy (transient astronomy); and also learned to link the two fields into a practical research problem: applying anomaly detection techniques to astrophysical transients.

Chapters 1 and 2 are the literature review Chapters. Transient astronomy is discussed in Chapter 1, which covers the most common techniques used to acquire data in optical astronomy and the common variable and transient events. Machine learning is discussed in Chapter 2, covering the algorithms used in this work (random forest, local outlier factor, and isolation forest), and also included is the application of anomaly detection in transient astronomy.

Chapter 3 covers the tools used in this work, which includes the feature extractors (`feets` and `avocado`) and the `ASTRONOMALY` framework. It also include the steps taken in extending `ASTRONOMALY` to operate with time series data. Chapters 4 and 5 are the research chapters, which discusses the application of the upgraded framework on variable and transient time series data from real observations (Chapter 4) and simulated (Chapter 5). Chapter 6 discusses conclusions drawn from Chapters 4 and 5.

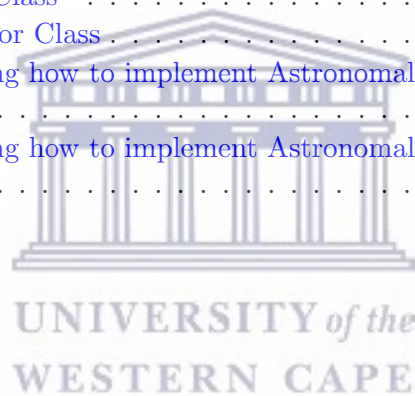
Contents

| | |
|---|-------------|
| Declaration of Authorship | i |
| Abstract | ii |
| Acknowledgements | iii |
| Preface | iv |
| List of Figures | viii |
| List of Tables | x |
| Abbreviations | xi |
| Physical Constants | xii |
| 1 Introduction to Time Domain Astronomy | 1 |
| 1.1 Taking astronomical observations | 2 |
| 1.1.1 Photometric Observations | 2 |
| 1.1.2 Spectroscopic Observations | 4 |
| 1.2 Variable Stars | 6 |
| 1.2.1 Extrinsic Variables | 6 |
| 1.2.1.1 Eclipsing Variables and Transiting Exoplanets | 6 |
| 1.2.1.2 Rotating Variables | 7 |
| 1.2.2 Intrinsic Variables | 10 |
| 1.2.2.1 Pulsating Variables | 11 |
| 1.2.2.2 Cataclysmic Variables | 14 |
| 1.3 Transient Events | 16 |
| 1.3.1 Supernovae | 18 |
| 1.3.2 Other Types of Supernovae | 22 |
| 1.3.3 Tidal Disruption Events | 23 |
| 1.3.4 Gamma Ray Bursts | 24 |
| 1.3.5 Kilonovae | 26 |
| 1.3.6 Active Galactic Nuclei | 27 |
| 1.3.7 Flare Stars | 28 |



| | | |
|----------|--|------------|
| 1.4 | Optical Surveys and the Big Data Challenge | 29 |
| 2 | Introduction to Machine Learning | 31 |
| 2.1 | Supervised Learning | 32 |
| 2.1.1 | Classification and Regression | 33 |
| 2.1.2 | Random Forest-An Example of a Classification Algorithm | 34 |
| 2.1.3 | Building a Model and Analysing its Performance | 36 |
| 2.2 | Unsupervised Learning | 39 |
| 2.2.1 | Clustering, Association and Dimensionality Reduction | 39 |
| 2.2.2 | Anomaly Detection | 40 |
| 2.2.2.1 | Isolation Forest | 41 |
| 2.2.2.2 | Local Outlier Factor | 43 |
| 2.3 | Active Learning | 44 |
| 2.4 | Anomaly Detection Applied to Transient Astronomy | 46 |
| 3 | Methodology | 52 |
| 3.1 | ASTRONOMALY | 53 |
| 3.1.1 | Data Management and Preprocessing | 54 |
| 3.1.2 | Feature Extraction | 57 |
| 3.1.2.1 | Feature Extraction With feets | 57 |
| 3.1.2.2 | Feature extraction with avocado | 69 |
| 3.1.3 | Data post-processing | 73 |
| 3.1.4 | Anomaly Detection | 73 |
| 3.1.5 | The Frontend Web Interface | 74 |
| 3.1.6 | Active Learning | 76 |
| 3.1.7 | Analysing the Performance of Unsupervised Anomaly Detection Algorithms | 77 |
| 3.2 | Follow-up Studies on Detected Anomalies | 78 |
| 4 | Application of Astronomy on the Catalina Real-Time Transient Survey data | 80 |
| 4.1 | The Data | 81 |
| 4.2 | An Overview of the Methodology | 84 |
| 4.2.1 | Feature Extraction | 84 |
| 4.2.2 | Anomaly Detection and Active Learning | 86 |
| 4.3 | Results | 88 |
| 4.3.0.1 | Analysis of the Top 12 anomalies after Active learning | 90 |
| 4.4 | Discussion | 98 |
| 5 | Application of ASTRONOMALY on the Photometric LSST Astronomical Time-Series Classification Challenge data | 100 |
| 5.1 | The Photometric LSST Astronomical Time-Series Classification Challenge | 100 |
| 5.2 | The Data | 102 |
| 5.2.1 | The Samples | 106 |
| 5.3 | Astronomy applied to a small PLAsTiCC sample | 106 |
| 5.3.1 | The Methodology | 107 |
| 5.3.1.1 | Data Pre-processing and Feature Extraction | 107 |
| 5.3.1.2 | Anomaly Detection, Active Learning, and Classification | 110 |
| 5.3.2 | Results | 110 |

| | | |
|----------|--|------------|
| 5.4 | Astronomy applied to a large sample with all classes in the PLAsTiCC Data | 114 |
| 5.4.1 | The methodology | 115 |
| 5.4.2 | Results | 115 |
| 5.5 | Discussion | 118 |
| 5.5.1 | Comparing Anomalies Detected from the feets and Avocado samples | 120 |
| 5.5.2 | Scaling Astronomy | 120 |
| 6 | Conclusions | 122 |
| | | |
| | Bibliography | 125 |
| | | |
| A | Objects with Unclear Labels in the MANTRA Data | 146 |
| | | |
| B | The code developed when extending Astronomy to Operate with Light Curve Data | 148 |
| B.1 | The Light Curve Reader Class | 148 |
| B.2 | The feets Feature Extractor Class | 160 |
| B.3 | An Example script showing how to implement Astronomy to search for anomalies in the MANTRA data. | 164 |
| B.4 | An Example script showing how to implement Astronomy to search for anomalies in the PLAsTiCC data. | 167 |



List of Figures

| | | |
|------|--|----|
| 1.1 | An example taxonomy tree for transients and variable objects (Förster et al. 2021). | 1 |
| 1.2 | Example filters used by telescopes | 3 |
| 1.3 | An example light curve of an eclipsing binary system | 7 |
| 1.4 | Phased example light curve of a rotating variable star | 8 |
| 1.5 | Example r-band light curves of microlensing events | 9 |
| 1.6 | Similar to Figure 1.5 | 10 |
| 1.7 | The HR diagram showing the locations of known variable stars. | 11 |
| 1.8 | Phased example light curves of pulsating variable stars | 13 |
| 1.9 | Schematic representation of a binary cataclysmic variable system (Warner 2003). | 15 |
| 1.10 | Example light curves of cataclysmic variables | 16 |
| 1.11 | Example light curves of different variable and transient events in different observation bands | 17 |
| 1.12 | The classification of SNe based on their spectral features | 19 |
| 1.13 | Example light curve shapes of different SNe. | 20 |
| 1.14 | Spectra of different SNe | 21 |
| 1.15 | Example light curves of known SNIa | 22 |
| 1.16 | Examples light curves of TDEs (Velzen et al. 2019). | 24 |
| 1.17 | GRBs duration against number of detected burst. | 25 |
| 1.18 | Examples light curves of GRBs | 26 |
| 1.19 | Example near-infrared light curves of KNe | 27 |
| 1.20 | Example light curves of a blazar | 28 |
| 1.21 | Example light curves of an M dwarf flare star. | 29 |
| 1.22 | Schematic image of the Rubin Observatory camera | 30 |
| 2.1 | Types of machine learning problems. | 31 |
| 2.2 | Schematic example of two decision trees | 34 |
| 2.3 | An illustration of an overfitted mapping function | 37 |
| 2.4 | Confusion metrics of a binary classification problem between target label A and B. | 39 |
| 2.5 | Demonstration that iForest requires few random splits to isolate an anomaly | 42 |
| 2.6 | Demonstration of the LOF anomaly scores based on the local densities of objects. | 44 |
| 3.1 | The Astronomy pipeline. | 54 |
| 3.2 | A screenshot of the ASTRONOMY frontend visualisation tab | 75 |
| 3.3 | A screenshot of the ASTRONOMY frontend anomaly scoring tab | 76 |
| 3.4 | A flow diagram showing the follow-up studies procedure | 79 |
| 4.1 | Example light curve from the MANTRA dataset. | 82 |
| 4.2 | A bar graph showing a list of labels present in the MANTRA data. | 83 |

| | | |
|------|---|-----|
| 4.3 | Histogram showing the light curve duration of objects in the MANTRA data in units of MJD. | 83 |
| 4.4 | Histogram showing the distribution of some of the computed features for different classes. | 85 |
| 4.5 | Scatter plot of some of the calculated features. Note that all the plots have the same legend. | 86 |
| 4.6 | Correlation metric of the computed features from <code>feets</code> | 87 |
| 4.7 | A violin plot of the computed anomaly scores from iForest for every input class in the MANTRA data. | 88 |
| 4.8 | Light curves of random objects in the CRTS data. | 89 |
| 4.9 | The top 12 anomalies as by the raw anomaly scores. | 90 |
| 4.10 | The top 12 anomalies after active learning. | 92 |
| 4.11 | Top 12 anomalies flagged by active learning. | 93 |
| 4.12 | Light curve of IRAS 04188+0122. | 94 |
| 4.13 | Anomalous spectra of IRAS 04188+0122 | 95 |
| 4.14 | Light curve of SN 2013cv. | 96 |
| 4.15 | Light curve of 1FGL J0050.0-0446. | 97 |
| 4.16 | Same as Figure 4.14 but for <i>CSS130509</i> | 98 |
| 5.1 | A plot showing the distribution of the six passbands in the LSST data. | 101 |
| 5.2 | Example light curves of two randomly selected objects from the PLAsTiCC data. | 102 |
| 5.3 | Example light curves from the <code>KN_RRL_sample</code> | 107 |
| 5.4 | The flow diagram of the pipeline used for data pre-processing in the PLAsTiCC data. | 108 |
| 5.5 | Recall and RWS for anomalies in the <code>KN_RRL_sample</code> | 111 |
| 5.6 | Confusion matrix from the random forest classifier, class 1 is for the KNe and class 0 is for the RRL. | 111 |
| 5.7 | The scatter plots showing the relationship between some of the important features as by random forest (see figure 5.8) and random features in the data. We can see from left panel and middle panel plots, that there is a distinct difference between the KNe and RR Lyrae labels. | 113 |
| 5.8 | The top 30 important features as by the random forest classifier. | 114 |
| 5.9 | Same as figure 5.5, but for the <code>all_labels_sample</code> | 115 |
| 5.10 | A bar graph showing the top 200 anomalous classes detected from both the <code>feets_sample</code> and <code>avocado_sample</code> | 116 |
| 5.11 | A histogram showing the flux and flux error distributions of the top 3 anomalous labels from the <code>avocado_sample</code> | 117 |
| 5.12 | Same as figure 5.11 but from the <code>feets_sample</code> | 118 |
| 5.13 | A histogram showing the host galaxy photometric redshift distribution for the top 200 anomalies detected from both the <code>feets_sample</code> and <code>avocado_sample</code> | 119 |

List of Tables

| | | |
|-----|--|-----|
| 1.1 | Different pulsating variable stars and their properties taken from Catelan and Smith 2015; Percy 2007; Eyer and Mowlavi 2008. Objects with regular, irregular, and semi-irregular light curves shapes are those with periodic, non-periodic, and semi-period light curves, respectively. | 12 |
| 1.2 | Table of cataclysmic variables classes and their characteristics (Robinson 1976). | 15 |
| 3.1 | Descriptions of features covered in this work. Note that feets has more features which are not covered here. We only cover those that we computed based on the input data we have. Their references can be found in the footnotes at the end of the table. | 63 |
| 3.2 | List of features computed from the <i>avocado</i> code. | 71 |
| 4.1 | The top 12 anomalous objects from the MANTRA data | 90 |
| 5.1 | A list of classes in the PLAsTiCC test set | 103 |
| A.1 | A List of objects with unclear labels in the MANTRA data. | 146 |



Abbreviations

| | |
|-----------------|--|
| CRTS | C atalina R ea L-T ime T ransient S urvey |
| LSST | L egacy S urvey of S pace and T ime |
| PLAsTiCC | P hotometric L SST A stronomical T ime-series C lassification C hallenge |
| LOF | L ocal O utlier F actor |
| SN | S upernova |
| KN | K ilonova |
| feets | fe ATURES e XTRACTOR for t IME s ERIES |



Physical Constants

Solar mass $M_{\odot} = 1.989 \times 10^{30}$ kg

Solar luminosity $L_{\odot} = 3.846 \times 10^{26}$ watts



UNIVERSITY *of the*
WESTERN CAPE

Chapter 1

Introduction to Time Domain Astronomy

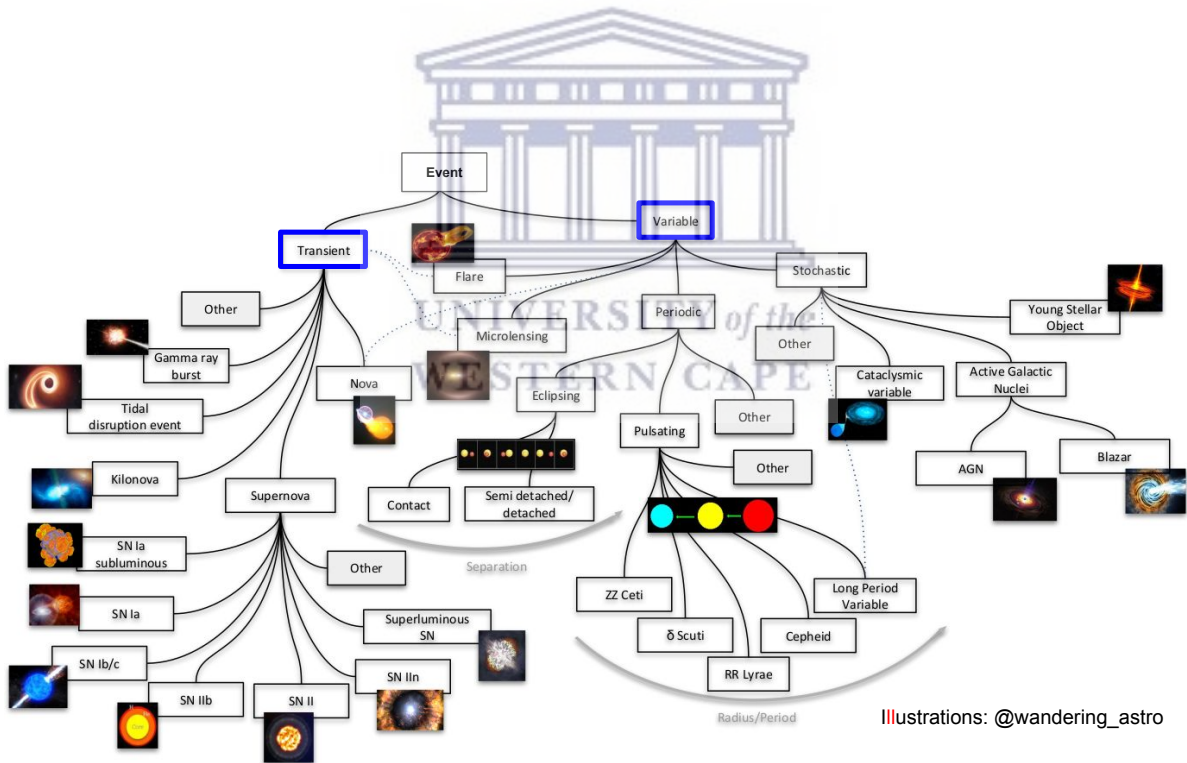


FIGURE 1.1: An example taxonomy tree for transients and variable objects (Förster et al. 2021).

Time domain astronomy is the study of short-lived astronomical phenomena with time-varying properties: how they occur, what triggered the process, what are the implications for known models, and how can they be used to better probe and understand the Universe. They are categorised into two main groups: transient events and variable events (see Figure 1.1 which outlines the two groups highlighted by blue boxes and their respective subgroups). There are many types of objects in each

group, and some of them, particularly rare events (“known unknowns” and “unknown unknowns”), do not have a well-understood underlying physics.

In this work, we investigate how well machine learning techniques detect rare variable and transient events in a given light curve dataset.

1.1 Taking astronomical observations

Scientists study the transient and variable events using photometric and spectroscopic observations. Discussions covered in this section are based on optical astronomy, however, our approaches are general and can be applied to most astronomical data.

1.1.1 Photometric Observations

Astronomers study both transient and variable events using light curves, a plot of brightness against time for a target object. To construct a light curve, a telescope is used to capture multiple images of a target event over a period of time. The images are commonly saved in a standard scientific format called FITS, an acronym for Flexible Image Transport System, which contains the raw data and additional information about the observations (e.g., exposure time, time and date of observation, and the coordinates of the target object).

Telescopes use filters (also known as bands or passbands) when capturing images to improve their contrast and sharpness. They do this by filtering other wavelengths of light except the one specified in the filter itself. The most common filters used in astronomy are the Johnson-Cousins *UBVRI* filters. However, survey telescopes such as the Sloan Digital Sky Survey (SDSS) also have their own filters, e.g., the SDSS *u'g'r'i'z'* filters. Figure 1.2 shows the wavelength ranges covered by both the Johnson-Cousins *UBVRI* and SDSS *u'g'r'i'z'* filters and their normalised efficiency (Bessell 2005).

It is important to note that each filter will have its own light curve for cases where the telescopes have multiple filters. In constructing the light curves, the multiple images captured by the telescopes can also cover other stars or events that are within the field of view of the telescope.

Astronomers often employ a technique called aperture photometry (see review in Mighell 1999) to measure the brightness of the target object for a single observation, where the brightness of the target star is obtained by fitting a circle to a target star and adding the pixel counts within the fitted circle.

The background brightness is measured by adding pixel counts from an annulus fitted around the target. The final brightness of the target is then the brightness of the target minus the background brightness. However, to measure the variability of the target, a reference star (a star that does not vary with time) has to be identified in the same image, and the same process described above is repeated to get the brightness of the reference star. The exact process of measuring the brightness for both the target star and the reference star is then repeated for the remaining images from other observing days. The target star's brightness is then compared with that of the reference star over the observing period. If there are changes detected, then the target object is either a variable or transient event. These variations in brightness will then be recorded for the observation duration and plotted as a light curve.

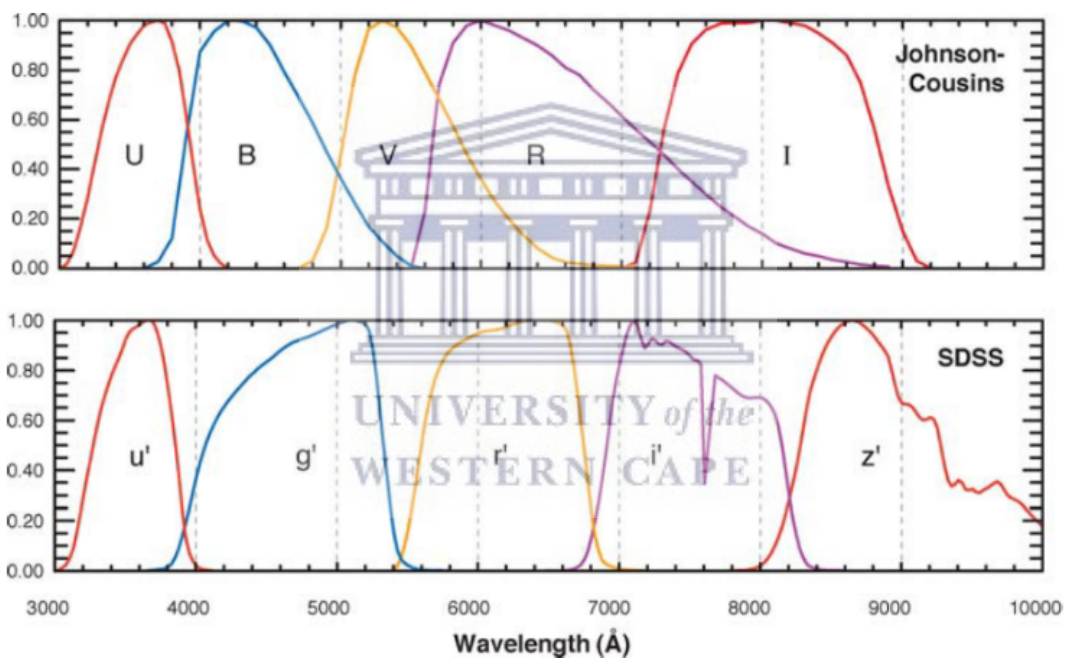


FIGURE 1.2: Example filters used by telescopes, the top plot shows the Johnson-Cousins *UBVRI* filters and the bottom plot shows the SDSS *u'g'r'i'z'* filters (note that the SDSS filters have atmosphere included while Johnson-Cousins does not). The y-axis are their normalised efficiency (normalised to unit at peak; Bessell 2005).

The technique described above is what is known as differential photometry. However, the most common technique used to find variable and transient events is difference imaging photometry (hereafter difference imaging). Difference imaging involves registering a test image (e.g., new observations) to a high signal to noise ratio (S/N), low mass, stacked reference image. A point spread function (PSF) is fitted to uncrowded stars to compute the convolution kernel required to map the reference image

to the test image. The resultant convolution kernel is applied to the reference image. The test image is then photometrically normalised to match the convolved reference image. Finally, objects are detected by subtracting the registered, normalised and convolved image from the test image (Alcock et al. 1999a).

However, the time measurements are often recorded in Julian days (JD) or in modified Julian days (MJD). The former corresponds to the number of days since 1 January 4713 BC, while the latter is the number of days since 1 January 1950 (McCarthy 1998). This time units are used to simplify calculations in astronomy.

Light curves offer insights into the astrophysics of variable and transient events. For example, the period of variable stars can be estimated directly from their light curves. The light curves can also be used to search for exoplanets through a technique called the transit method. This involves observing a target star over some time; if the star has an orbiting planet around it, its light curve will display periodic dips. The light curves, in this case, can help estimate the orbital period of the system and the size of the exoplanet (e.g., Gibson et al. 2009). Another case where light curves can be helpful is the case of categorising different variable and transient events (e.g., Belokurov, Evans, and Le Du 2004). The shape and behavior of light curves are unique for most classes, and a new transient/variable event can be assigned a class by inspecting the shape of its light curve, however, there are some uncertainties involved. Similarly, a new class of transient/variable events can be proposed if the shape of the light curve of a target object does not match any of the known shapes (support evidence, such as spectroscopy is required to declare a new class).

1.1.2 Spectroscopic Observations

The shapes of the light curves alone are often not enough to distinguish/verify a class of transient/variable events. In such cases, additional data such as spectra are required for follow up studies. In astronomy, the electromagnetic radiation of a target object is recorded with a set of instruments collectively known as a spectrograph: these include the slit, collimating mirror, dispersing device (diffraction grating are often used), a camera mirror, a detector [e.g., charged-couple device (CCD; see review of CCDs in Lesser 2015)], and computer. A small fraction of light rays from an observing telescope passes through the slit and hits the collimating mirror, which then rearranges the rays to be parallel. The parallel rays are then split into constituent wavelengths by the light dispersing device (e.g., a diffraction grating), which changes the wavelengths and position them on the focal plane of the camera mirror. The camera mirror finally focuses photons from each of the wavelengths on the

detector (e.g., the CCD) which are then converted to electrical signals, which can also be converted to flux, and the results are saved as an image on a computer. A spectrum is then constructed by recording the brightness of the target object at different wavelengths, which is then calibrated using reference stars to get the final spectrum.

The spectrum can take different characteristic shapes, depending on the type of source and the material that the light passes through between the source and the observer. There are many different examples of emission that creates a continuum spectrum such as synchrotron emission or blackbody radiation. Most sources are surrounded by gas and dust (there is always an interstellar and intergalactic material between us and any source). This material then absorbs photons at specific wavelengths, which results in a spectrum with absorption lines known as an absorption spectrum (characterised by a decrease in intensity at specific wavelengths).

Lastly, suppose there is an interactive medium (e.g., a molecular or interstellar medium) between the user and an energetic target source (e.g., supernovae). The photons from the source will excite the electrons in the cloud resulting in an emission spectrum characterised by an increase in intensity at specific wavelengths in the spectra.

A wide variety of information can be retrieved from these spectra: they can be used for the classification of different astronomical objects (e.g., Veilleux and Osterbrock 1987). We can also use them to learn about the chemical composition of ions, atoms and molecules in an object (e.g., Aoki et al. 2012). Lastly, we can use them to estimate the temperature and motion of the object by comparing its observed wavelengths to rest wavelengths to measure if the object is moving towards or away from us and most importantly, estimate its redshift (e.g., Hutchinson et al. 2016).

Some of the transient events/variable events have the same light curve shapes but different chemical compositions. This might be due to different progenitors (e.g., the progenitor stars for type Ia supernovae are white dwarfs with a companion star, while that of type II supernova are massive stars). The two progenitor stars will have a different chemical composition, and their spectra will be different (see Filippenko 1997; Branch, Baron, and Jeffery 2001). Studying both their light curves and spectra can decrease the chances of misclassification. In sections 1.2 and 1.3, we describe different classes of transients and variable events and give an example light curve for each class.

1.2 Variable Stars

Variable stars are stars that display variation in brightness in their light curves. The first detected variable star is the Omicron Ceti, which was discovered in 1638 by Johannes Holwarda, who corrected its classification as it was previously classified as a nova in 1596 by David Fabricius. He discovered that Omicron Ceti (currently known as the Mira star) was pulsating every 11 Months, which showed that the sky is not entirely invariable (Hoffleit 1997). Three decades later, Geminiano discovered an eclipsing variable star (know as Algol) whose variation mechanism was explained by John Goodricke in 1784 (Bopp 1980). Development in technology then increased the catalogue of variable stars through photographs in the late 1890s, where telescopes were able to take photographs of stars that were not visible to a human eye.

Some of the variable stars (examples and details of each of them will be explained later in this section) are periodic, and their light curves can be represented in a folded format. To fold a light curve, the period (P) of the object is first computed, and used to compute its phase as follows:


$$\text{phase} = \frac{t - t_0}{P}, \quad (1.1)$$

where t is the time observations and t_0 is the time the light curve is at a specific phase, such as maximum or minimum. The decimal points from equation 1.1 are then used as the phase, and the final light curve is thus a plot of brightness against the phase (e.g., see Figure 1.4). Variable stars are categorised into two main groups: extrinsic and intrinsic variables.

1.2.1 Extrinsic Variables

Extrinsic variables are variable stars whose brightness varies because of external characteristics such as eclipses and rotations. They are further categorised into three main groups: transiting exoplanets, eclipsing, rotating, and microlensing variables.

1.2.1.1 Eclipsing Variables and Transiting Exoplanets

Eclipsing variables are stars found when the orbital plane of two or more companion stars is aligned with the observer's line of sight. As they orbit each other, they form eclipses which are detected as dips in their light curves. A typical example is the eclipsing binary stars (EBs).

With eclipsing binaries, as the two stars orbit each other, two different dips are detected in their light curves, where the bigger deep occurs when the dimmer star passed in front of the brighter star causing a primary eclipse. The other dip occurs when the bright star passes in front of the dim star causing the secondary eclipse (see Figure 1.3). The orbital period of the system is thus the time between two successive primary eclipses (e.g., Southworth et al. 2004).

Similarly, transiting exoplanets form eclipses around stars they are orbiting, resulting in primary and secondary eclipse in their light curves. The orbital period of the system is thus computed as the period between successive primary eclipses (e.g., Gibson et al. 2009).

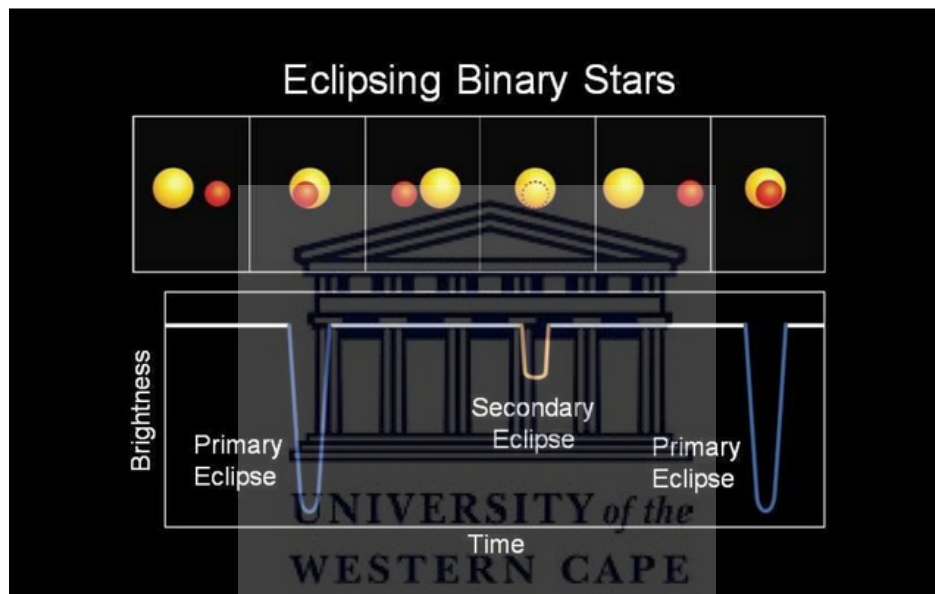


FIGURE 1.3: An example light curve of an eclipsing binary system showing how the light varies as the two stars orbit each other. The top panel shows the orbits of the stars, and the bottom panel shows the corresponding variation in brightness. Credit: NASA.

1.2.1.2 Rotating Variables

Rotating variables are variable stars whose brightness may vary because of an uneven distribution of luminosity on their surface; where there are small regions of low luminosity as opposed to the total luminosity of the star (e.g., starspots), which are caused by strong magnetic fields (Solanki 2003). Our Sun is an example as it occasionally displays sunspots on its surface. If a significant fraction of its surface was covered with sunspots, then a variation in brightness would be detected from its light curve as it rotates (see an example folded light curve in Figure 1.4). However, sunspots cover a small fraction of the Sun's surface and will not account for much brightness variability in their light curves.

Stars with bigger starspots, like the By Draconis stars, display significant variations in brightness as spots rotate in and out of sight of an observer (Bopp 1980).

Another example of rotating variable stars is young stellar objects (YSO), also known as pre-main-sequence. They are detected in the early stages of their stellar evolution before they reach the main sequence and display brightness variation in their light curves which is due to the presence of hot and cold spots on their surface (e.g., Herbst 2012). Studying the light curves of rotating variable stars can help estimate their periods (see Bopp 1980; Vogt 1975).

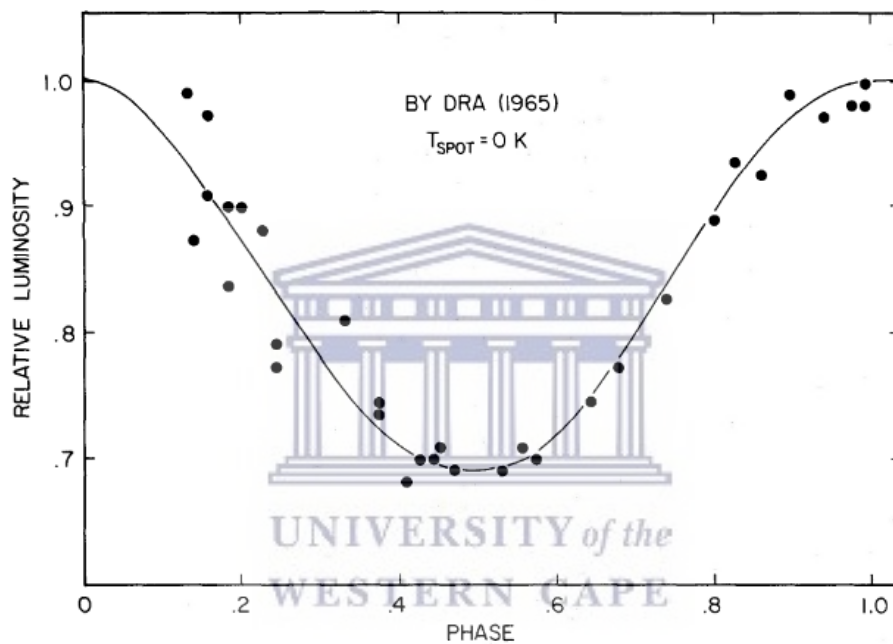


FIGURE 1.4: Phased example light curve of a rotating variable star, where the dots corresponds to the observations, and the solid line is the best fit described by Vogt 1975.

Gravitational Microlensing Variables

Gravitational microlensing occurs when the total luminosity of a distant source is magnified by a foreground stellar object (also known as a gravitational lens, there are other gravitational lenses whose details are beyond the scope of this work) as it passes between the observer's line of sight and the source. There are two main types of gravitational microlensing: microlensing due to a single lens and binary lens.

Microlensing is believed to be caused by a single gravitational lens object, while binary microlensing events are believed to be caused by multiple gravitational lens objects (multiple stars in line with each other). The light curves of the former are characterised by an increase then decrease in brightness as

the gravitational lens passes in front of the distance source, and that of the latter displays multiple peaks corresponding to each of the gravitational lens objects (see Figure 1.5; Guo et al. 2015).

For the purpose of this work, we adapted the names given to gravitational microlensing events simulated in the PLAsTiCC data, which is μ Lens-Single (for the gravitational microlensing due to a single lens) and μ Lens-Binary (for the gravitational microlensing due to a binary lens); see chapter 5 for the description of the PLAsTiCC data.

Studying the light curves of microlensing events can help astronomers search for exoplanets around stellar gravitational lenses (e.g., see Figure 1.5). They can also help study the stellar population of faint objects such as the black holes, white dwarfs, brown dwarfs, and neutron stars (e.g., Hansen 1998).

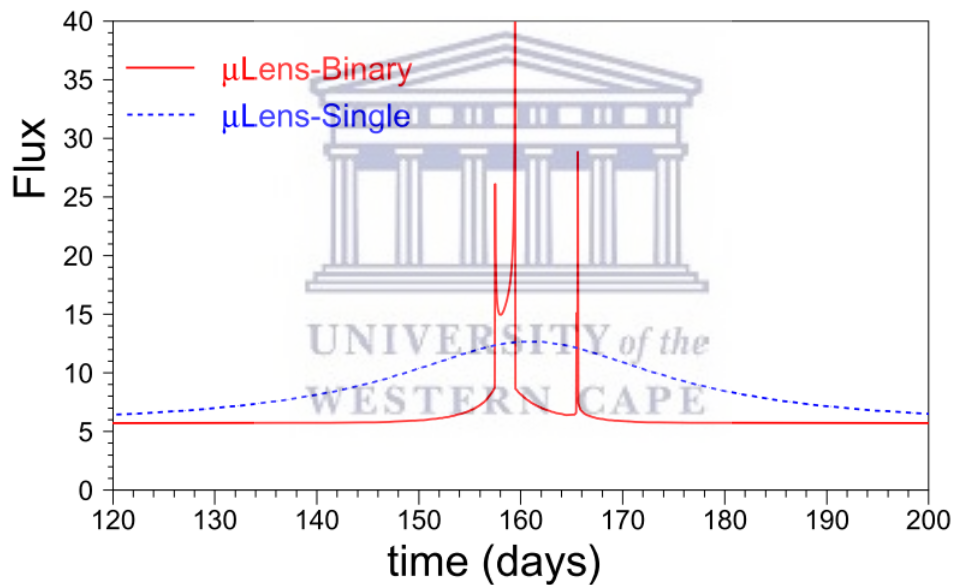


FIGURE 1.5: Example r-band light curves of microlensing events, wherein the blue dotted curve is for the single lens, and the red is for the binary lens Kessler et al. 2019.

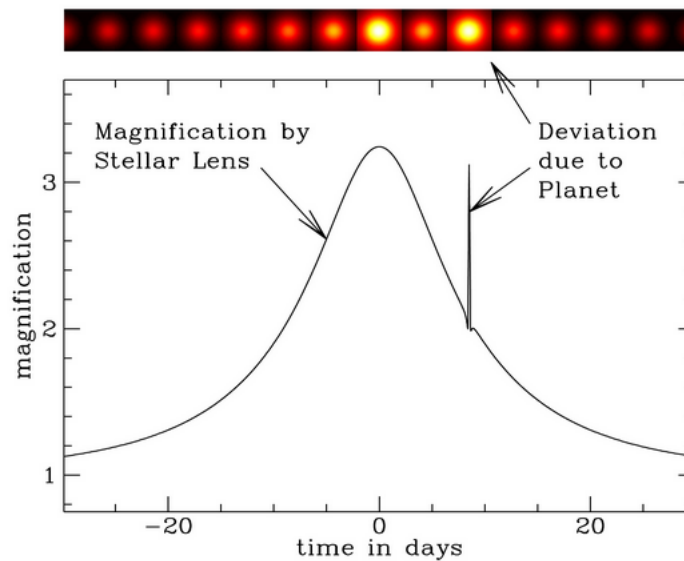


FIGURE 1.6: Similar to Figure 1.5, but here the lens is a star with an orbiting exoplanet. The top panel indicates the change in brightness of a distance as the lens passes in front of it, and the bottom panel is the corresponding light curve. Indicated with pointing arrows are the magnifications due to both the star and the exoplanet. Image credit: NASA ¹

1.2.2 Intrinsic Variables

Intrinsic variable stars are objects whose brightness varies because of their physical properties. These stars, along with other variable stars, can be found in different locations on the Hertzsprung–Russell diagram (HR diagram, e.g., see Figure 1.7), a scatter plot of luminosity against the spectral type/effective temperature. The spectral types gives information about the temperature of the stars. It takes the letters, O B A F G K M, arranged in an order from hotter to cooler. They are further subdivided into numbers from 0-9 in an order from hottest to coolest, e.g., A0 is the hottest, and A9 is the coolest (Kutner 2003).

¹<https://nexsci.caltech.edu/workshop/2011/>

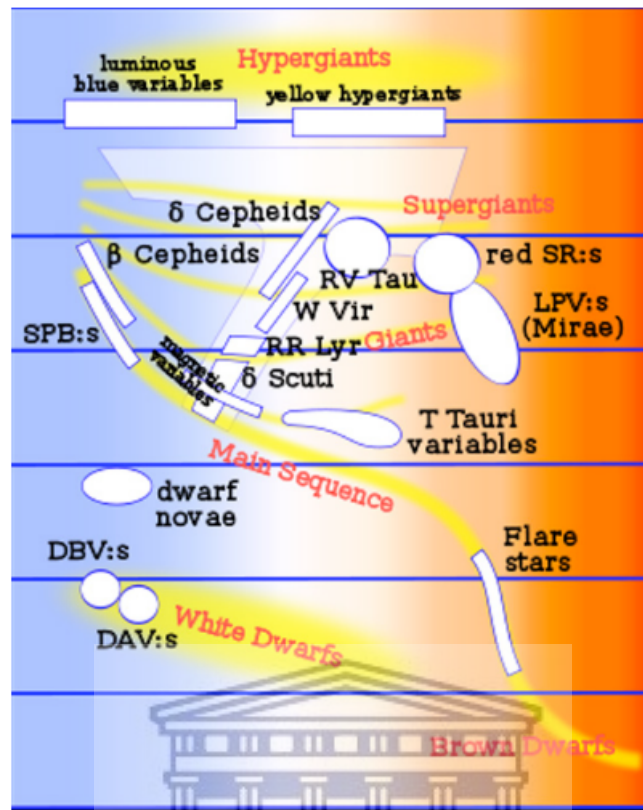


FIGURE 1.7: The HR diagram showing the locations of known variable stars. The horizontal axis is related to their spectral types/effective temperature, and the vertical axis corresponds to either their luminosity/absolute magnitudes (LSST Science Collaboration et al. 2009).

The HR diagram gives information about the stellar evolution. Most intrinsic variable stars are located at the instability regions of the HR diagram (Eyer and Mowlavi 2008). They are categorised into two main groups: pulsating and cataclysmic variables.

1.2.2.1 Pulsating Variables

Pulsating variable stars are stars whose brightness varies due to repeated expansion and contraction of their radii. Most of them follow a radial pulsation, which occurs when the sphere of the star's outer layer expands and contracts symmetrically. However, some stars also follow a non-radial pulsation which occurs when the volume of the star is kept constant while its shape changes (e.g., pulsating Be stars; Rivinius, Baade, and Štefl 2003). Their light curves are categorised into three groups: regular, semi-regular, and irregular, where regular light curves are strictly periodic, irregular light curves are non-periodic, and semi-irregular light curves are in between the regular and irregular light curves (Catelan and Smith 2014).

The variation observed of most pulsating stars is believed to be caused by a phenomenon known as the Eddington's valve (Eddington 1988). This occurs in late stellar evolution, where stars burn helium in their outer shells after depleting hydrogen in their core; as the star burns helium, the helium atom alternate between being singly and doubly ionised. The latter is more opaque than the former and absorbs radiation from the core of the star, leading to more doubly ionised helium.

As the amount of doubly ionised helium increases, more radiation is absorbed (the star is dimmer at this point). The star's outer helium layer expands due to the increased star temperature. Its temperature decreases as it expands, resulting in less ionised helium (singly ionised); the star becomes less opaque and bright as radiation escapes. The star then contracts because of its gravitational force. Its temperature increases as it contracts, leading to more doubly ionised helium ions, making the star appear dim again and the process is repeated.

TABLE 1.1: Different pulsating variable stars and their properties taken from Catelan and Smith 2015; Percy 2007; Eyer and Mowlavi 2008. Objects with regular, irregular, and semi-irregular light curves shapes are those with periodic, non-periodic, and semi-period light curves, respectively.

| Pulsating Variable | Period [days] | Amplitude [V band mags] | Spectral Type | Location on HR diagram | Light Curve Shape | Mass [M_{\odot}] |
|--------------------|---------------|-------------------------|---------------|------------------------|----------------------|----------------------|
| Type I Cepheids | 1 - 135 | 0.1 - 1.5 | F, G - K | Yellow supergiant | Regular | 2 - 20 |
| W Virginis | 0.8 - 35 | 0.3 - 1.2 | F, G - K | Supergiants | Semi-regular | 0.4 |
| RV Tauri | 30 - 150 | 1 - 3 | F - G, K - M | Yellow supergiant | Semi - regular | 0.1 - 2.2 |
| RR Lyrae | 0.1 - 1 | 0.2 - 2 | F5 - A8 | Horizontal branch | Regular | 0.6 - 0.8 |
| Delta Scuti | 0.01 - 0.2 | 0.003 - 0.9 | A0 - F5 | Giant/main sequence | Regular/Semi-regular | 2.2 |
| Mira | 100 - 1000 | 2.5 - 10 | M, C or S | Red giant | Irregular | 1 - 2 |

Most pulsating variables follow the Eddington's valve mechanism, including type I Cepheids, type II Cepheids, and RR Lyrae. See their example light curves in Figure 1.8 and their properties in table 1.1, where W Virginis and RV Tauri are sub-classes of type II Cepheids. However, Mira stars are different in that their pulsation is driven by burning hydrogen and helium shells (Whitelock 1999).

This occurs when the radius of the star expands due to a burning helium shell around the carbon/oxygen core, and as the helium shell runs out of fuel, the star shrinks. As the star shrinks, its temperature increases, and the outer hydrogen shells continues to convert hydrogen to helium. The inner helium shell starts burning helium again, resulting in what is known as the helium shell flash characterised by high luminosity, resulting in the expansion of the star radius. This process is repeated, and as the star expands and shrinks, it changes brightness (this process is also known as the thermal pulsation; Marigo et al. 2008).

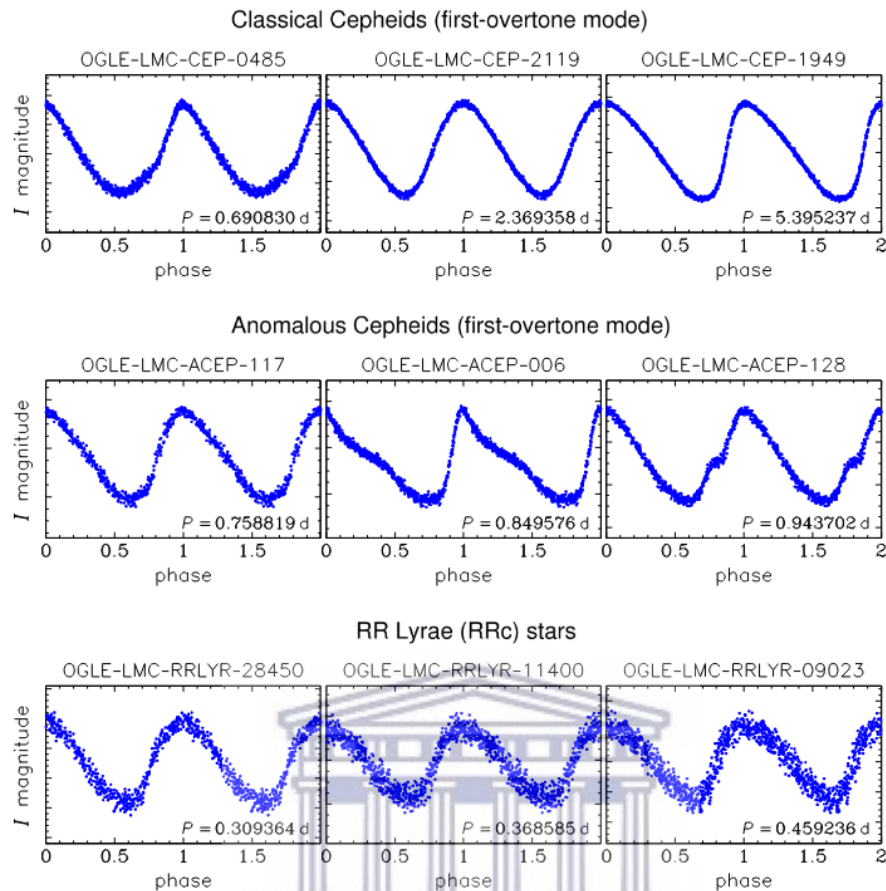


FIGURE 1.8: Phased example light curves of pulsating variable stars, top and middle panels corresponding to Cepheids and the bottom panel corresponding to the RR Lyrae. The middle panel shows the anomalous Cepheids that displays a bump close to the minimum brightness, as opposed to the smooth curves observed in the classical Cepheids in the top panel (Soszyński et al. 2019).

Mira stars are examples of a group of variable stars known as long-period variables (LPV, characterised by periods between hundred days and more than thousand days), which follow the thermal pulsation mechanism described above. During the thermal pulsation process, some of the material from the star's core (e.g., carbon) might travel to the outer shells, forming what are known as carbon stars. They are characterised by a red atmosphere dominated by carbon (Ripoche et al. 2020), and some of them are known to be variable with semi-regular light curves (e.g., Brincat, Galdies, and Hills 2020).

Table 1.1 shows properties of the most common pulsating variables. Notice the wide range in masses, temperatures and periods of these objects. This highlights the complexity in classifying the different types, however machine learning algorithms can easily address such complexities (e.g., Richards et al. 2011).

Pulsating Variables as Distance Indicators

Henrietta Leavitt discovered the period-luminosity relationship in the 1900s, where she discovered a direct proportion between the period and luminosity of Cepheids, where bright cepheids were associated with long periods and vice versa (Leavitt 1908). Right after her discovery, Ejnar Hertzsprung (Hertzsprung 1913) proposed that given the period and apparent magnitude (m) of a Cepheid star at a given distance (d), the period can be interpolated on the period-luminosity plot to estimate its absolute magnitude (M) and the distance can be calculated by substituting m and M in the distance modulus equation given by:

$$m - M = 5 \log_{10}(d) - 5. \quad (1.2)$$

Edwin Hubble also used the same principles as that of Ejnar Hertzsprung, to calculate distances to the Cepheids in the Andromeda galaxy (M31) in 1924, where he discovered that M31 was separated from our galaxy (E. P. Hubble 1926). He later measured redshifts of galaxies and discovered that galaxies at high redshifts move away from us at high speed compared to nearby galaxies, which then led to the discovery that the Universe is expanding (E. Hubble 1929). This phenomenon is also known as Hubble's law which is given by:



$$v = H_0 D, \quad (1.3)$$

where v is the recession velocity corresponding to the measured redshifts, D is the proper distance to the galaxy, and H_0 is the Hubble constant.

1.2.2.2 Cataclysmic Variables

Cataclysmic variable stars (CVs) are stars found in compact binary systems. The system is composed of a primary star (usually a white dwarf) and a secondary star (also known as a donor star) which is typically a main-sequence star (more massive stars are rarely found in binary systems). Mass transfer occurs in the binary system where the tidal forces distort and disrupt the surface of the donor stars, resulting in mass accretion from the donor star to the primary star (Warner 2003).

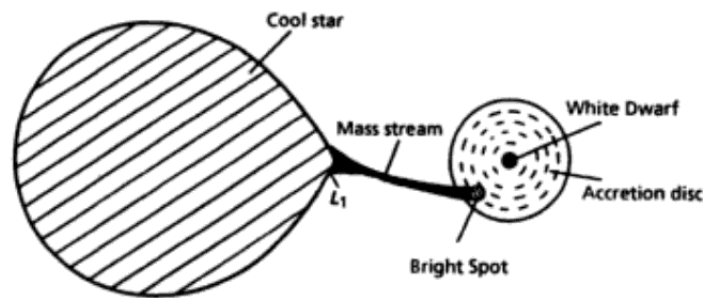


FIGURE 1.9: Schematic representation of a binary cataclysmic variable system (Warner 2003).

The mass from the donor star forms what is known as an accretion disk (a layer of matter that is orbiting a compact central object) around the white dwarf. Mass from the inner disk (mostly hydrogen) falls onto the white dwarf, increasing its temperature and density. The high temperature ignites hydrogen fusion onto the hydrogen layer around the white dwarf, which converts hydrogen to helium. This is characterised by a sudden increase in brightness in their light curves (see Figure 1.10), such phenomena are known as novae.

TABLE 1.2: Table of cataclysmic variables classes and their characteristics (Robinson 1976).

| Class | Amplitude [mag] | Energy [ergs] | Recurrence time |
|-----------------|-----------------|-----------------------------|-------------------|
| Novae | 9 – > 14 | $10^{44} - 10^{45}$ or more | Only one eruption |
| Recurrent novae | 7–9 | $10^{44} - 10^{45}$ | 10 – 100 days |
| Dwarf novae | | | |
| (a) U Gem | 2–6 | $10^{38} - 10^{39}$ | 15 – 500 days |
| (a) Z Cam | 2–5 | $10^{38} - 10^{39}$ | 10 – 50 days |
| Novalike | -- | -- | No eruptions |

Dwarf novae, on the other hand, occurs when a small portion of the outer accretion disc is not stable (see the bright spot in Figure 1.9), i.e., the disc changes from a quiescent state (associated with cool temperature and low luminosity) to an active state (associated with high temperatures and high luminosity). The characteristics of both novae and dwarf novae are outlined in Table 1.2, where recurrent novae and dwarf novae are classified as variable events, while the novae are transients since their outburst is only detected once.

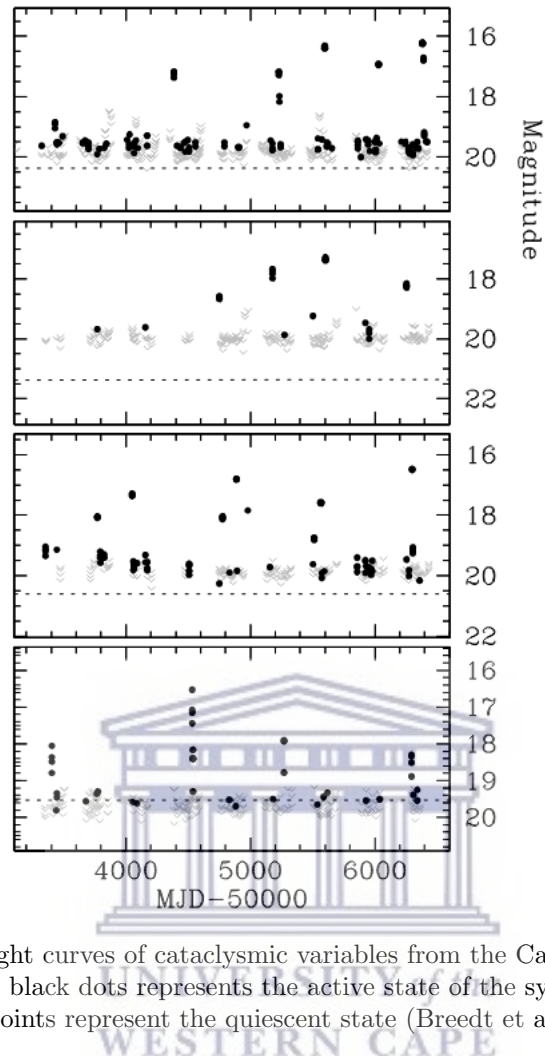


FIGURE 1.10: Example light curves of cataclysmic variables from the Catalina real-time transient survey (CRTS), where the black dots represents the active state of the system (outbursts), and the gray points represent the quiescent state (Breedt et al. 2014).

1.3 Transient Events

The study of explosive events has gained momentum in the past few decades, where astronomers extensively study the events themselves, and their environments. They are different from variable stars in that they occur once (i.e., they are non-recurrent). Astronomers have been studying these events before the development of telescopes, where a new bright star appears in the sky and disappears within short timescales, with Tycho's supernova (also known as SN 1572) being an example (Schaefer 1996).

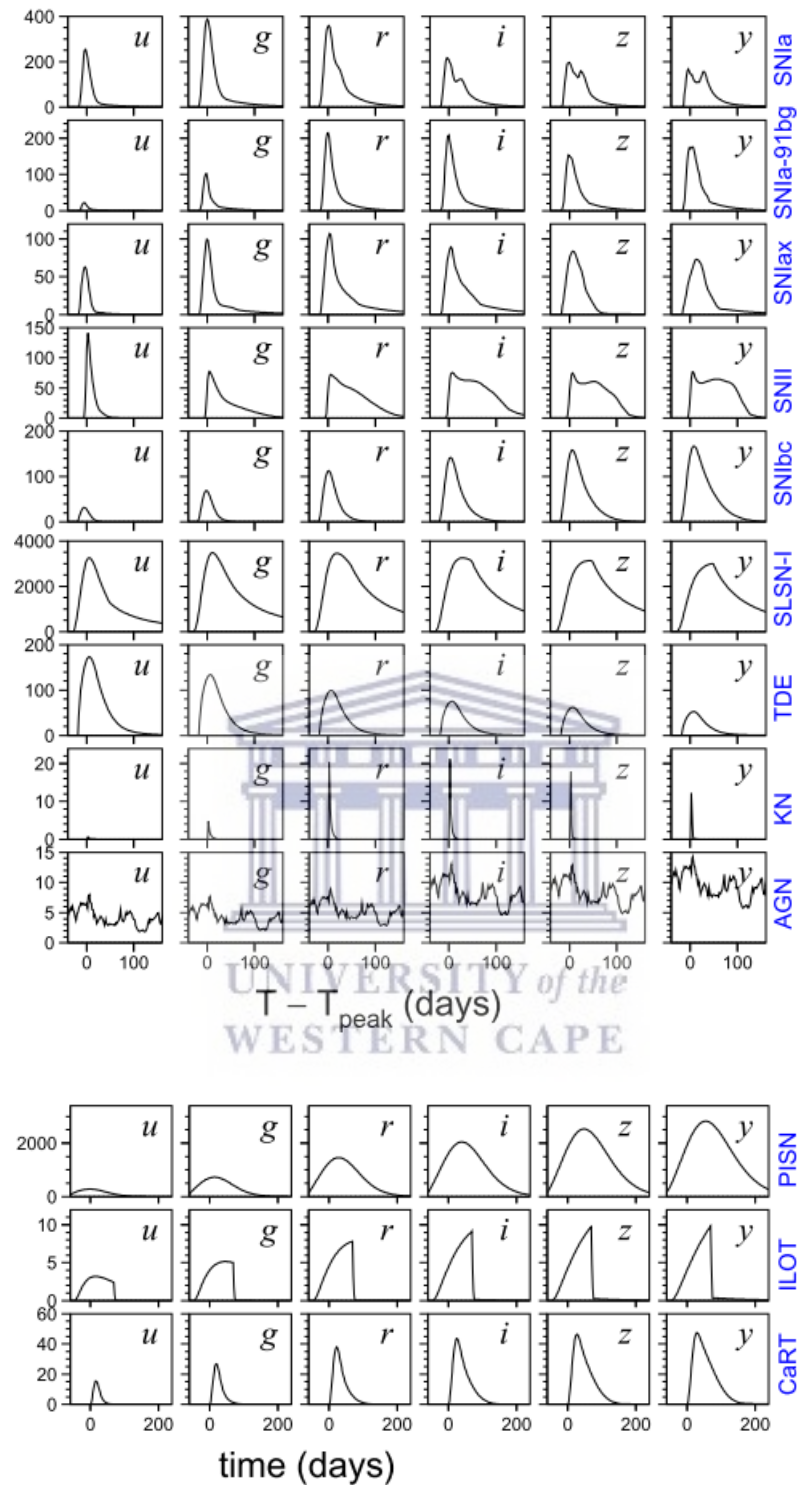


FIGURE 1.11: Example light curves of different variable and transient events in different observation bands (*ugrizy*), the y-axis are flux values in arbitrary units, the x-axis is the time in days where T_{peak} is the time at peak bolometric flux. This Figure also shows some of the rare transients including the intermediate luminosity transients (ILOT), calcium-rich transients (CaRT), and pair-instability supernova (PISN), details of which will be discussed in the next sections, (Kessler et al. 2019).

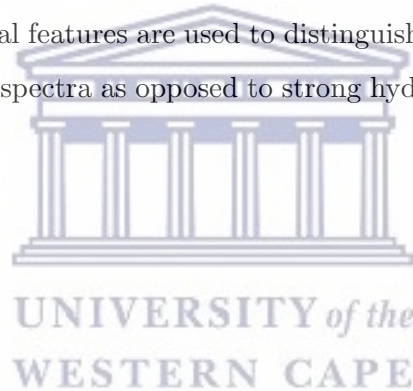
Tycho discovered a bright object in the sky in 1572, which he and other astronomers thought was

a new star (Schaefer 1996). Later observations showed that it was a supernova, called SN 1572. Supernovae are explosive events characterised by high luminosities. Type Ia supernova are used to estimate cosmic distances (Riess, Press, and Kirshner 1996; Perlmutter et al. 1998; Krause et al. 2008).

Discoveries of transient events such as SN 1572 opened a new window in astronomy where astronomers used modern technology to search for more transient events in the universe to understand their true nature. We give an overview of the most common transient events below.

1.3.1 Supernovae

Supernovae (SNe) are one of the most energetic explosive events in the Universe, having a luminosity that can be as bright as a galaxy. They are mainly triggered by the death of massive stars, and they have different classes. Their spectral features are used to distinguish between their classes where type I SNe lacks hydrogen lines in their spectra as opposed to strong hydrogen lines present in type II SNe spectra.



| Spectral Type | Ia | Ib | Ic | II |
|---------------------------|---|---|-----------|-------------|
| Spectrum | No Hydrogen | | | Hydrogen |
| | Silicon | No Silicon | | |
| | | Helium | No Helium | |
| Physical Mechanism | Nuclear explosion of low-mass star | Core collapse of evolved massive star (may have lost its hydrogen or even helium envelope during red-giant evolution) | | |
| Light Curve | Reproducible | Large variations | | |
| Neutrinos | Insignificant | ~ 100 × Visible energy | | |
| Compact Remnant | None | Neutron star (typically appears as pulsar) Sometimes black hole | | |
| Rate / h ² SNU | 0.36 ± 0.11 | 0.14 ± 0.07 | | 0.71 ± 0.34 |
| Observed | Total ~ 5600 as of 2011 (Asiago SN Catalogue) | | | |

FIGURE 1.12: The classification of SNe based on their spectral features. The rate is measured in supernova units where 1 SNU = 1 SN per century per $10^{10} L_{\odot,B}$ (Raffelt 2012), where $L_{\odot,B}$ is the B band solar luminosity. Note that the number of observed SNe has increased to 94,248 as of 5 February 2022 (see the open supernova catalog: <https://sne.space/statistics/>).

WESTERN CAPE

Type I SNe are further categorised into different groups based on their spectral features: type Ib/c does not have silicon in their spectra while SNIa do; type Ic SNe does not have helium in their spectra while type Ib do (Raffelt 2012), see the classification table in Figure 1.12, and their spectral shapes in Figure 1.14.

The progenitor stars vary between the different SNe classes: type Ib/c and type II SNe are caused by the death of a massive star through a process called core collapse explosion. This occurs in the late stellar evolution of massive stars ($8M_{\odot} \leq M_{*} \leq 40M_{\odot}$; where M_{*} is the mass of the star) whereby the stars are not stable due to the depletion of fuel in their core. This occurs when the stars have fused silicon into iron (which is why silicon is often not detected in their spectra) in their core. Because the star cannot fuse heavier elements than iron, it collapses under its own gravity. The infalling material then explodes into a shock wave, which leaves either a neutron star or a black hole remnant (Raffelt 2012).

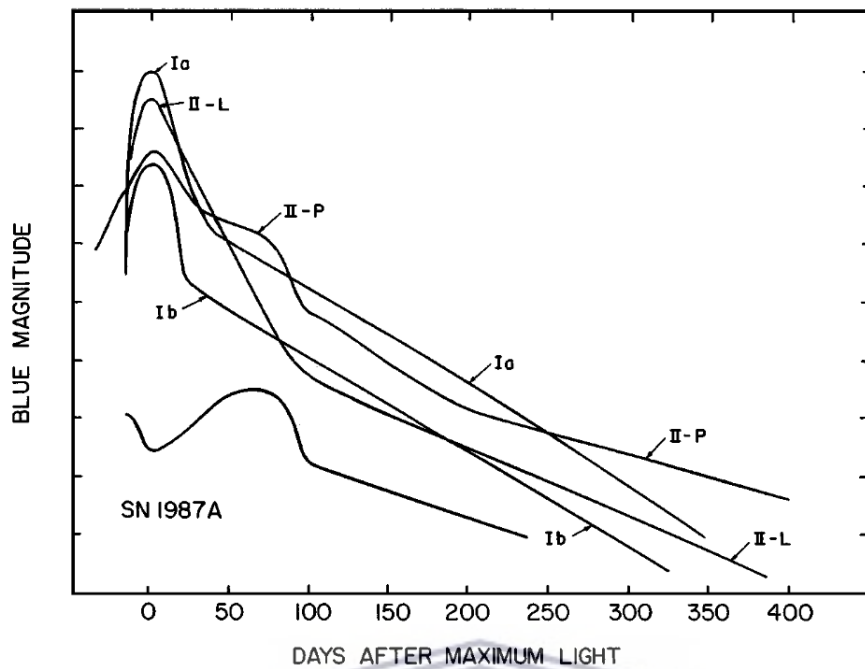


FIGURE 1.13: Example light curve shapes of different SNe. The types are indicated by the arrows pointing the curves (Filippenko 1997).

SNIa, on the other hand, are believed to be caused by thermonuclear explosions, where the white dwarf accretes mass from a companion star (a main sequence or giant star) or collides with another white dwarf. It does this until it reaches a mass limit of $1.44 M_{\odot}$ (also known as the Chandrasekhar limit, which is why the light curves of SNIa are reproducible), beyond this mass, the temperature and pressure in the core of the white dwarf rise triggering a runaway carbon fusion. The white dwarf then explodes into a type Ia supernova and becomes completely destroyed in the process (Raffelt 2012), emitting light that is mostly powered by nickel decaying into iron (Goobar and Leibundgut 2011).

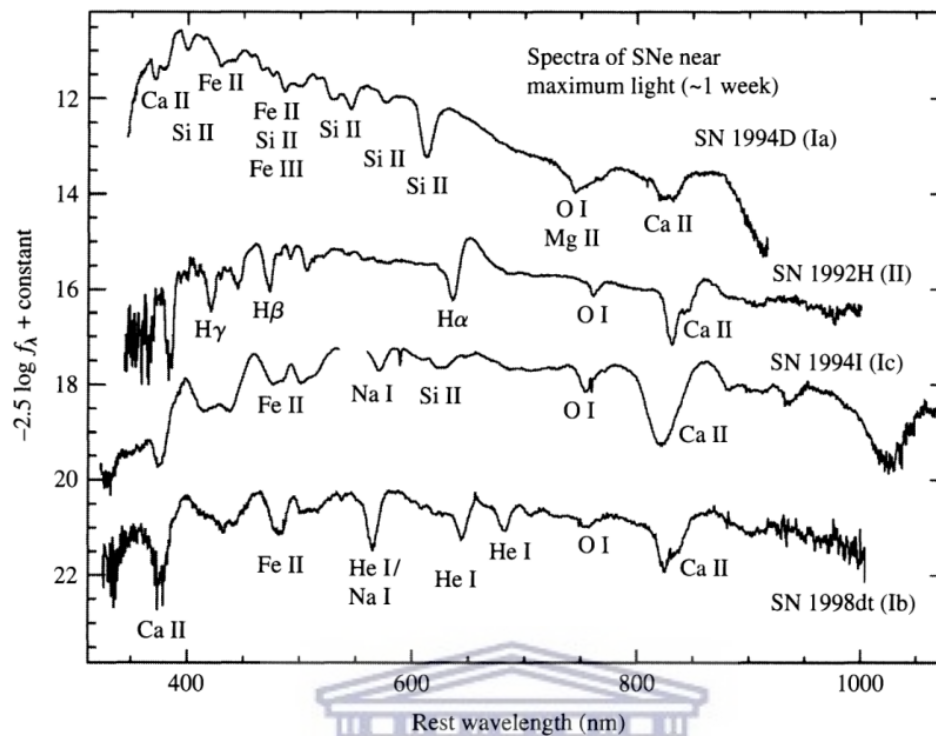


FIGURE 1.14: Spectra of different SNe, types of SNe are indicated in brackets next to the SN name (Ostlie and Carroll 2007).

The shapes of type Ib/c and type II SNe light curves vary significantly from object to another (see Figure 1.13), while those of type Ia are similar (see Figure 1.15). However, different SNIa have different luminosity and decay rate, where brighter SNe decay slower than the dimmer SNe. This phenomena is known as a stretch factor (or stretch-luminosity relation; Pskovskii 1977), and it can be corrected by estimating the amount of flux need for the observed flux to aligned with the rest frame (bottom panel of Figure 1.15; Goobar and Leibundgut 2011). Correcting the stretch factor means SNIa can be used as “standard candles” to estimate cosmological distances.

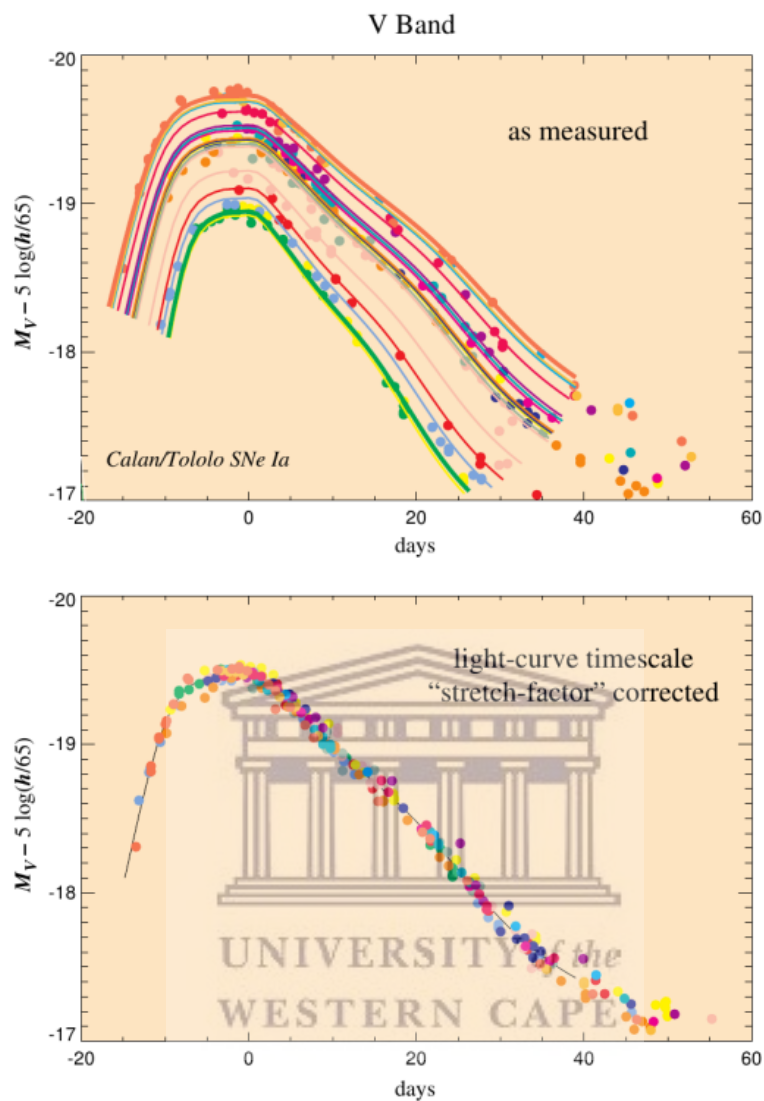


FIGURE 1.15: Example light curves of known SNIa (top panel), and their aligned light curve (bottom panel) after they were corrected for the stretch factor (Perlmutter et al. 1998).

1.3.2 Other Types of Supernovae

Modern telescopes have opened a new window of transient astronomy whereby new SNe events fail to fit any of the described classifications above (Goobar and Leibundgut 2011). Some of these events are rare [type I superluminous SNe, calcium-rich transients (CaRT), intermediate luminosity transients (ILOT), pair instability SNe (PISNe)], and some of them are common [peculiar SNIa (SNIa-91bg), and peculiar SNIax]; see their example light curves in Figure 1.11. Studying these objects is extremely useful for understanding stellar mechanisms better, and this project aims to employ machine learning algorithms to retrieve rare events from a given light curve dataset.

Type I Superluminous Supernovae (SLSN-I) are amongst the most luminous explosions with luminosity 50 times that of the normal supernova (SN Ia). They are characterised by a light curve that lasts for few months and blue hydrogen poor spectra (Chomiuk et al. 2011; Quimby et al. 2011). Their high peak brightness makes them ideal candidates as distance indicators for objects located at high redshifts (Scovaccicchi et al. 2016), if they can be standardised. They are believed to be driven by the same mechanism as that of type Ib/c and type II SNe.

Peculiar SN (SNIax) is a type of supernova explosion that is similar to SN Ia, but different in that the progenitor star (white dwarf) is not entirely disrupted after the explosion (Foley et al. 2013). The peak luminosity of SN Ia is much brighter than that of SNIax. Their spectra deviate from those of SN Ia at later times where they do not go into a “nebular” phase (a region in the spectra of SN Ia where emission lines from new elements dominate as opposed to absorption lines from elements from the progenitor stars).

SNIa-91bg (Filippenko et al. 1992) are subtypes of SNIa: they, however, differ from the normal SNIa in that they have a low luminosity, lack secondary peaks in *izy* bands, they disappear quickly after peak luminosity, and do not display any stretch-luminosity relation (Phillips 1993); hence they cannot be used as “standard candles.”

Calcium-rich transients (CaRT) are also dimmer than normal SNe, and their spectra are rich in calcium with a short-lived light curve, and their true progenitors are not known (Lunnan et al. 2017).

Pair instability SNe (PISNe) thought to follow the same mechanism as that of type Ib/c and type II SNe; they are, however, different in that their progenitor stars are low-metallicity population III stars, and they do not leave any remnant behind after the explosions (Barkat, Rakavy, and Sack 1967; Kasen, Woosley, and Heger 2011). They are characterised by high luminosity and are expected to be found at high redshifts, which makes them difficult to observe (they are only predicted theoretically).

Intermediate luminosity transients (ILOT) are a group of transients whose peak brightness lies between that of SNe and novae; they are believed to often share the same progenitor stars as that of type II SNe (Kessler et al. 2019).

1.3.3 Tidal Disruption Events

Tidal disruption events (TDEs) occur when a star is ripped apart by strong tidal forces from a super massive black hole (SMBH) having a mass between $10^6 - 10^7 M_{\odot}$. The star passes close to the SMBH,

and the strong tidal forces from the SMBH triggers a mass transfer from the star onto the black hole, wherein the blackhole accretes $\sim 50\%$ of its mass. The high relativistic speed at which the mass is accreted gives rise to a luminous emission of radiation (also known as a flare; Rees 1988) across the electromagnetic spectrum. The flare peaks in the soft X-ray, ultraviolet wavelengths (Rees 1988; Ulmer 1999; Strubbe and Quataert 2009) and the near infrared wavelength (Jiang et al. 2016), see example light curves in Figure 1.11 and 1.16.

These events can also be detected in the optical wavelength (e.g., Hung et al. 2017) with spectra characterised by a strong blue continuum with broad H_α and doubly ionised helium (HeII ; Arcavi et al. 2014). Most of their light curves follow a power-law proportional to $t^{-5/3}$ (Evans and Kochanek 1989), where t is the observing time. TDEs are mostly found in post-starburst E+A galaxies [galaxies with an elliptical galaxy spectrum (hence the E) and an A-type stellar spectral type with strong Balmer lines (hence the A)]. They are found to power outflows and jets across the electromagnetic spectrum (e.g., X-ray, radio and UV). Studying the properties of these jets and the outflows can give insights into AGN feedback and the formation of jets. (Gezari 2021).

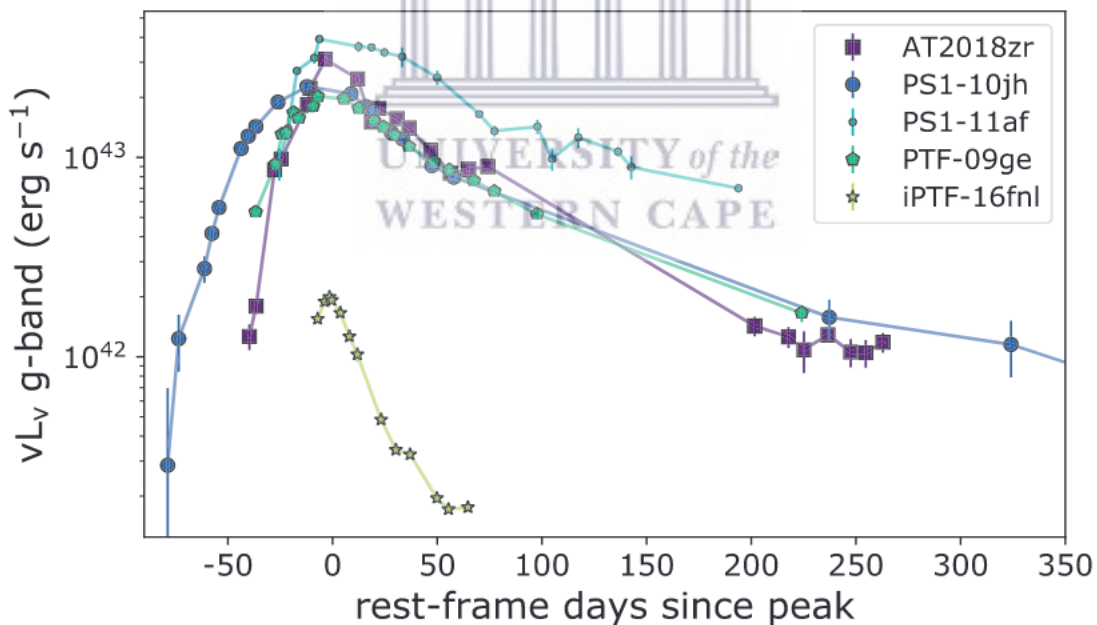


FIGURE 1.16: Examples light curves of TDEs (Velzen et al. 2019).

1.3.4 Gamma Ray Bursts

Gamma ray bursts (GRBs) are the most energetic, luminous events in the Universe detected in the gamma-ray part of the electromagnetic spectrum. They are thought to be triggered by: the death

of massive stars ($\text{Mass} > 20 M_{\odot}$), colliding neutron stars, or when a neutron star and a black hole merge or two black holes merge. They are categorised into two main groups: short-duration GRBs and long-duration GRBs. The duration of the former can last between milliseconds and 2 seconds, while the latter lasts between 2 seconds and a couple of minutes. Figure 1.17 shows the splitting point between the two objects from the Burst and Transient Source Explorer (BATSE) instrument from the Compton Gamma Ray Observatory (Gehrels, Chipman, and Kniffen 1994). Their light curves vary, i.e., they do not have a standard light curve shape (see Figure 1.18).

Short duration GRBs are likely to be triggered when a neutron star merges with another neutron star or with a black hole neutron star (Nakar 2007), producing a kilonova (e.g., Tanvir et al. 2013; Acciari et al. 2021). They are found in regions of low star formation rate or at the nuclei of galaxy clusters. In contrast, the long-duration GRBs are located in regions of high star formation rate. They are triggered by the deaths of massive stars generating a core-collapse supernova. The afterglow that remains after the burst can be detected with the ground-based telescopes in the optical, near-infrared, and radio wavelengths.

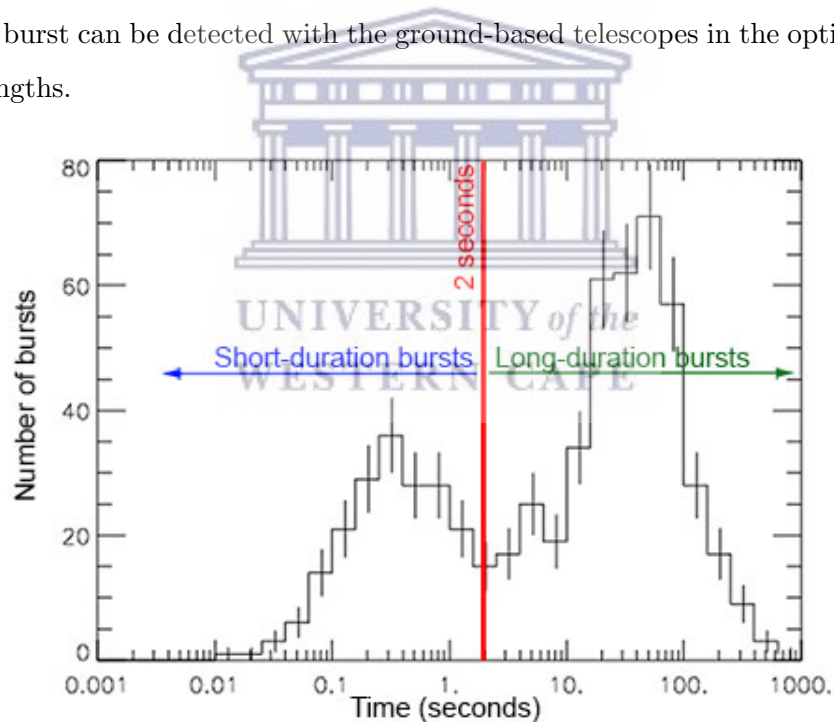


FIGURE 1.17: GRBs duration against number of detected burst. The red vertical line indicates the splitting point between the two types of GRBs, bursts on the left are short-duration bursts, while those on the right are long-duration bursts (NASA 2013).

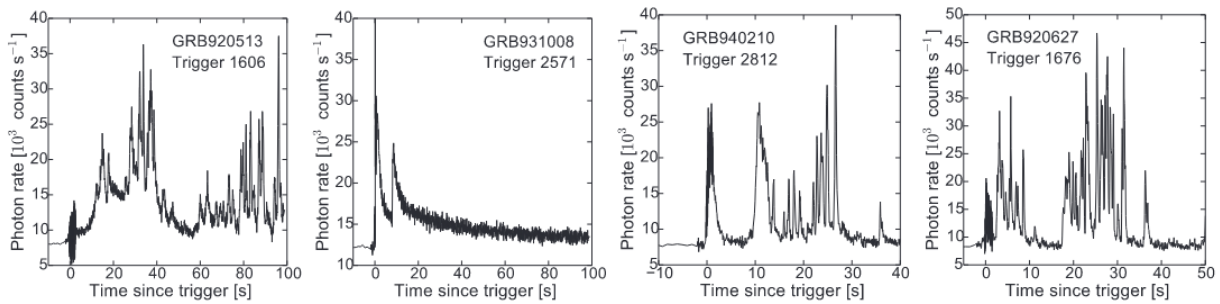


FIGURE 1.18: Examples light curves of GRBs, the x-axis is time in seconds, and y-axis is the count per seconds, which corresponds to the number of detected rays per second (Bustamante et al. 2017).

1.3.5 Kilonovae

When a neutron star is in a binary system with another neutron star (NS-NS) or a black hole (NS-BH), or when two blackholes are in binary systems (BH-BH), they can collide at high speeds, forming ripples in space and time. These ripples are known as gravitational waves. They were first predicted in 1916 by Einstein and first detected on 14 September 2015, by the advanced Laser Interferometer Gravitational-wave Observatory (aLIGO; Aasi et al. 2015) from BH-BH merger (Abbott et al. 2016).

The electromagnetic counterparts of gravitational waves are expected from either the NS-NS or NS-BH collisions. This occurs when the mass ejected from the collisions powers a transient event such as Kilonova (KN) visible in the electromagnetic spectrum. KN are strongly linked with short-duration GRBs (e.g, Tanvir et al. 2013; Ascenzi et al. 2019).

The first detected kilonova is the DLT17ck (also known as AT 2017gfo; Valenti et al. 2017), which was detected after the gravitational wave event GW170817. GW170817 was detected by the Virgo interferometer (Acernese et al. 2014) and the advanced Laser Interferometer Gravitational-wave Observatory (aLIGO; Aasi et al. 2015) on the 17 August 2017, where its progenitor stars are merging neutron stars (Soares-Santos et al. 2017). A short duration GRB, and an optical counterpart (DLT17ck) were also detected soon after the detection of GW170817 (Valenti et al. 2017).

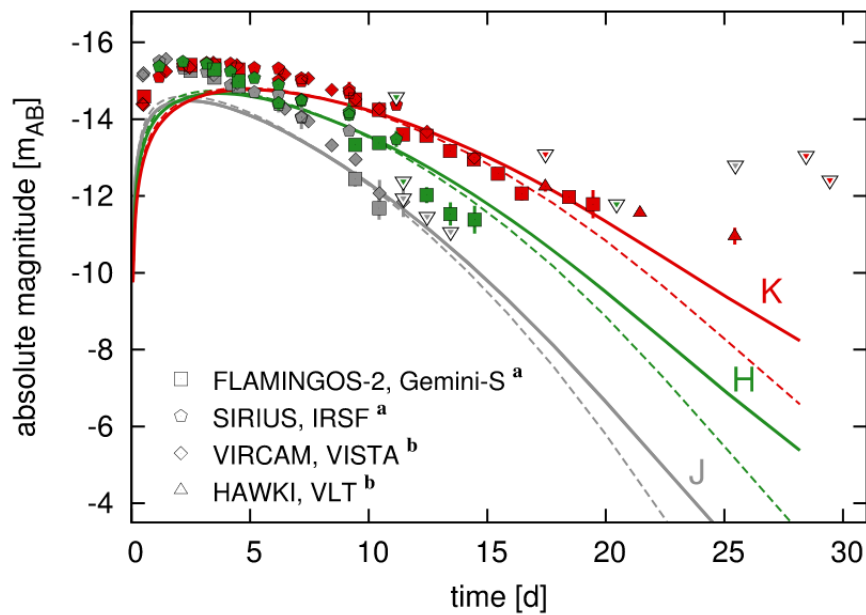


FIGURE 1.19: Example near-infrared light curves of KNe (indicated by the data points), and the solid lines are fitted theoretical models (Zhu et al. 2018).

These discoveries has proven that observations from gravitational waves and the electromagnetic spectrum can be used simultaneously to study phenomenon in the universe, which motivates the field of multimessenger astronomy, a field that studies four signals from a single astrophysical sources: gravitational waves, electromagnetic radiation, cosmic rays and neutrinos; Bartos and Kowalski 2017.

Kilonovae events are expected to peak in the red optical and near-infrared bands. Their light curves are characterised by a sudden increase in brightness that only lasts for a few weeks (see Figure 1.19).

1.3.6 Active Galactic Nuclei

Active galactic nuclei (AGN) are galaxies with nuclei that are brighter than that of normal galaxies (galaxies that does not have an active supermassive black hole in their center). Galaxies displaying such phenomena are called “active galaxies”, and they are one of the brightest objects in the universe. They are characterised by aperiodic variation in brightness, which is believed to be powered by large mass accretion by a supermassive black hole at their centers, associated with ionised matter that is released at high relativistic speed (close to the speed of light), also known as jets. This mass accretion might have been triggered by merging galaxies (Hopkins et al. 2006).

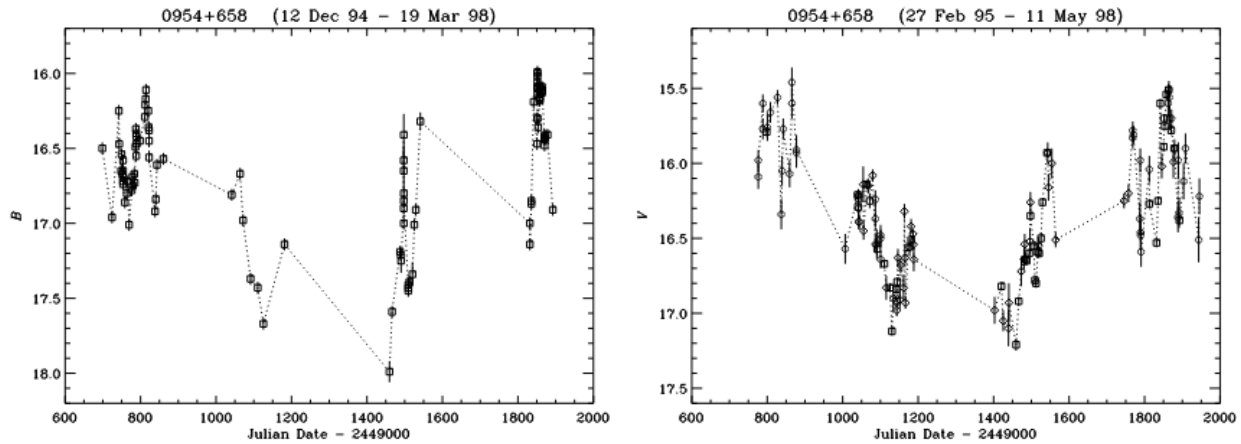


FIGURE 1.20: Example light curves of a blazar (Raiteri et al. 1999).

There are two types of AGN: type 1 and type 2; the spectra of type 1 AGN have broad balmer emission lines, and type 2 narrow balmer emission lines (Netzer 2015). Some AGN display a rare behavior where they change between types: changing from type 1 to type 2 or vice versa. These are known as “changing look” AGN (e.g., Yang et al. 2018).

A blazar is a subtype of AGN, where the jets are pointed in the direction of the observer. Their light curves are characterised by semi-periodic or aperiodic dramatic variation in brightness that can last for days to years (see Figures 1.11 and 1.20).

Studying AGN can help scientists understand the galaxy evolution as their high relativistic energy has an impact on star formation (Silk and Rees 1998). Their high luminosity makes them ideal candidates to measure the expansion rate of the universe at high redshifts (if they can be standardised), and Watson et al. 2011 attempted to standardise their luminosity so as to test this.

1.3.7 Flare Stars

Flare stars are stars that undergo a sudden dramatic luminosity outburst associated with stellar magnetic activities (Pettersen 1989). They often occur close to starspots, hence they are believed to be the results of a magnetic reconnection activity occurring around the starspots regions, which gives rise to the sudden luminosity outburst detected in their light curves (Pettersen 1989). Their light curves have both the variable and transient characteristics, where the former is due to the presence of starspots that rotate with the star [more details in section 1.2.1 and example light curves in Figure 1.21 (top and middle panels)], and the latter is due to the stellar flares (see bottom panel in Figure 1.21).

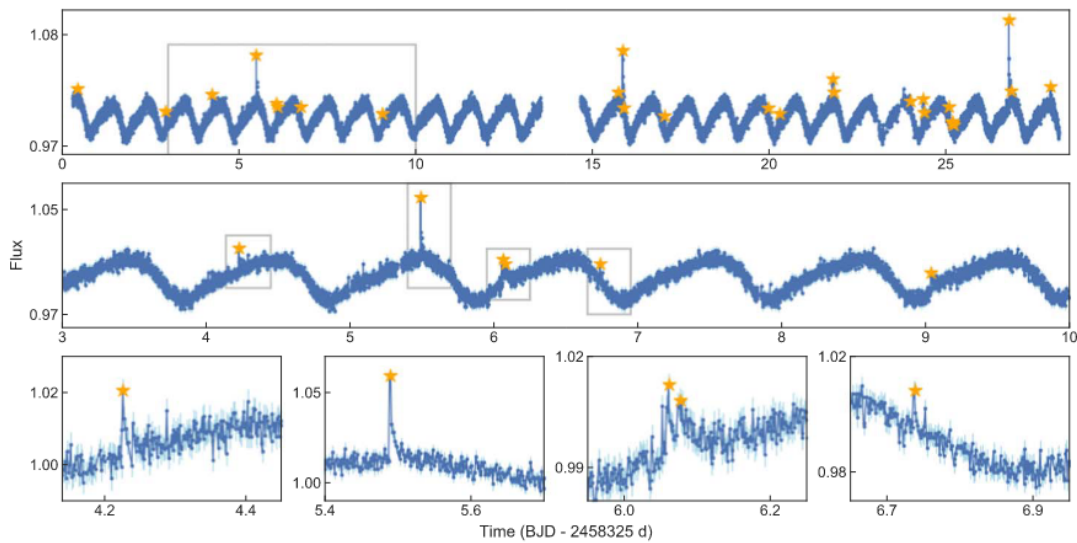


FIGURE 1.21: Example light curves of an M dwarf flare star. The star symbols indicate the flares, the top panel represent the original light curve, and the bottom panels represent zoomed-in regions indicated by gray boxes (Emmanoulopoulos, McHardy, and Uttley 2010).

Flare stars are located across the HR diagram, where the majority are found on the main sequence. The most abundant main sequence flare stars are the M spectral type dwarfs (also known as the M-dwarf). Flares from M dwarfs can last for few minutes, with amplitudes in magnitude change between 0.001 and 0.1 (Hawley et al. 2014). Their low luminosity makes them hard to detect during their quiescence periods (West et al. 2011), hence they are likely to be detected as transient stars rather than variable stars.

1.4 Optical Surveys and the Big Data Challenge

The search for known and unknown variable and transient events has recently gained momentum, making the field of transient astronomy a hot topic. Modern technology has improved, allowing us to have wide, fast, and deep surveys. The discovery rate of new transients and variable events has been exponentially increased by optical surveys such as the Panoramic Survey Telescope and Rapid Response System 1 (Chambers et al. 2016), the Dark Energy Survey (DES; Collaboration: et al. 2016), the Zwicky Transient Facility (ZTF; Bellm et al. 2018; Graham et al. 2019), the All Sky Automated Survey for Supernovae (ASAS-SN; Shappee et al. 2014), the Sloan Digital Sky Survey (SDSS; Blanton et al. 2017), and the Catalina Real-Time Transient Survey (CRTS; Drake et al. 2011). The ASAS-SN, ZTF, and DES are able to continuously survey a large fraction of the sky, releasing petabytes of data per year.

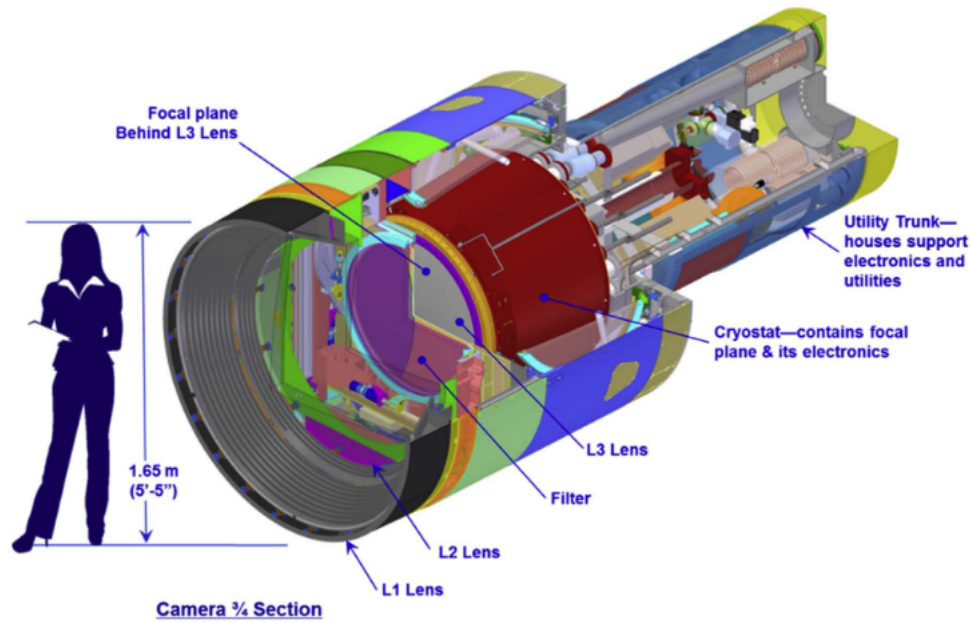


FIGURE 1.22: Schematic image of the Rubin Observatory camera, notice its size relative to an average human height (Ivezić et al. 2019).

The upcoming Vera C. Rubin observatory (Ivezić et al. 2019), through its ten-year Legacy Survey of Space and Time (LSST), is expected to increase the discovery rate to millions of new variable and transient events per year. The Rubin observatory is currently under construction in Chile, and its first light is expected in the next few years. It will operate on an 8.4-meter telescope with a gigantic 3.2 gigapixel camera with a field of view of 9.6 square degrees, and lastly, it will be observing in six passbands (*ugrizy*). Most of its observing time ($\sim 90\%$) will be on the wide fast deep survey (expected r mags ~ 24.5 , with visits separated by ~ 3 days) mode where ~ 32 trillion observations of ~ 20 million stars will be conducted to cover the primary science goal of the project. The remaining observing time will be allocated to very deep (expected r mags ~ 26) and very fast (expected short revisiting times of ~ 1 day) time-domain observations (LSST Science Collaboration et al. 2009; Ivezić et al. 2019).

The LSST is expected to detect millions of variable and transient events every night, releasing ~ 15 terabytes of data per night (Ivezić et al. 2019). There are limited resources for follow-up studies of these objects, which means that only a small fraction (roughly 0.1%) of them can be followed up with spectroscopic studies (Villar et al. 2021). The challenge is to find objects worthwhile for this studies using only their light curves. This calls for fast, automated data processing and analysis techniques to detect interesting objects from light curve datasets with minimum or no human expert guidance. This leads us to the field of machine learning.

Chapter 2

Introduction to Machine Learning

The current development in technology has resulted in systems being flooded with a massive amount of data in short timescales. Explicit programming techniques require in-depth domain knowledge and consistent system maintenance, which can be costly and time-consuming. They are often specific to the domain they are applied to, and they would fail in the current era of big data. Fast, automated, and data-driven techniques such as machine learning are required to extract information from this data.

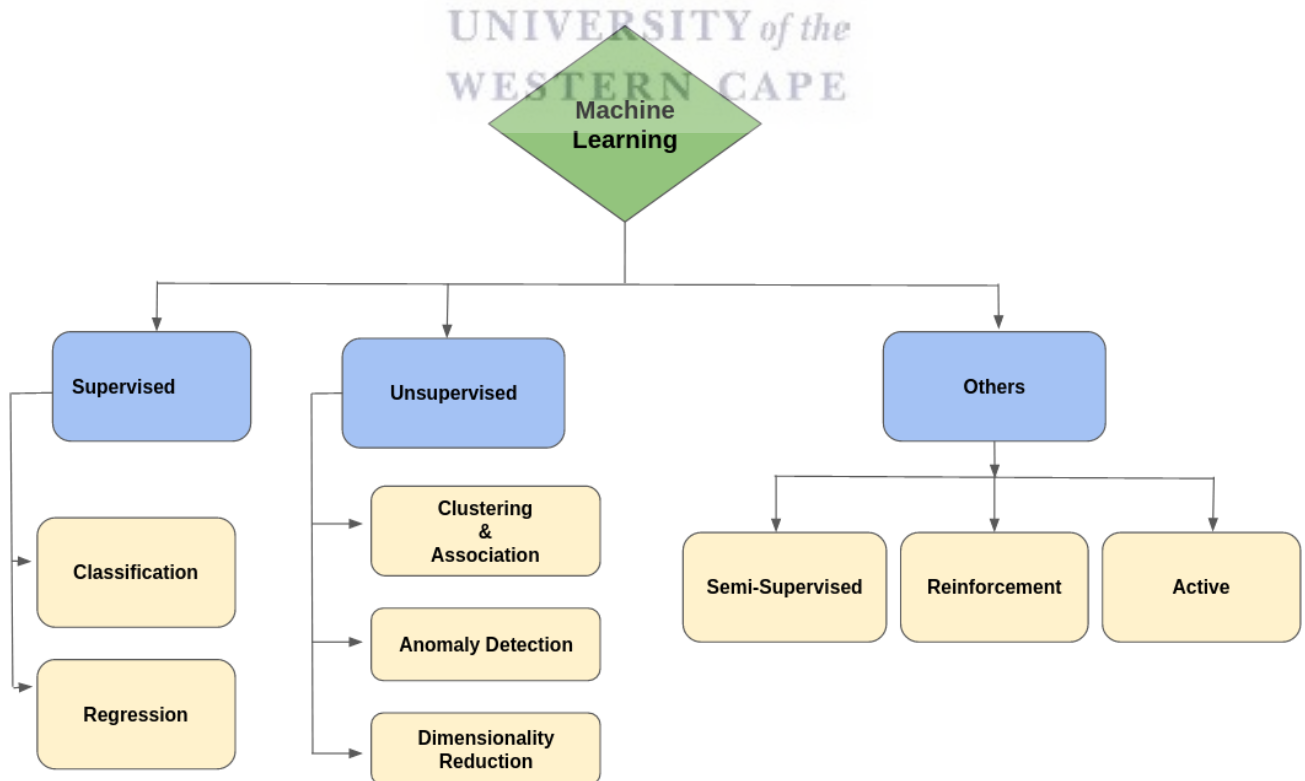


FIGURE 2.1: Types of machine learning problems.

In machine learning, a group of self-taught algorithms is employed by a computer to build computer programs that can automatically process, analyse and learn from data with minimum or no guidance (Mitchell 1997). These algorithms were first introduced in the mid-1900s, where their earliest applications were on computer gaming (Samuel 1959). For example, Samuel 1959 employed a machine to learn the game of checkers based on the rules of the game and other features. It was found that the machine outperformed a human user, and it only required ~ 8 hours (which was considered a short time, given the processing power of computers those days) to learn the game. The shortcoming of the machine learning algorithms in the early 1900s was that they required high computer processing power.

Recent development in technology has improved the processing power of computers, and they are now fast enough to run machine learning algorithms on sub-second timescales on certain datasets. Machine learning algorithms are ideal data analysis tools for the current and upcoming era of big data. They have been explored in a wide variety of fields, for example, they have been employed in banking to assess credit risks (predict if a customer will be able to pay a loan given a set of features; Shoumo et al. 2019; Aziz and Dowling 2019). They were also applied in healthcare, where machine learning algorithms are tasked to recognise and locate lung cancer in its early stages given set of lung images (Reddy, Reddy, and Reddy 2019). They were also applied in astronomy where supernovae were classified in different types based on their photometric light curves (Lochner et al. 2016). Figure 2.1 shows the main types of machine learning problems: supervised, unsupervised, and others that are outside the scope of this study except for active learning. This includes semi-supervised¹ and reinforcement learning²

2.1 Supervised Learning

In supervised machine learning, the learning algorithm is given labeled data (data with both the input, also known as features, and the desired output, also known as target labels) as input. The algorithm then maps the features to the target labels such that it can predict target labels of unseen data. The basic idea is to build a model that can generalise a mapping function f between the input data x to the target labels y such that $y = f(x)$. Here, x and y are the training data. A human expert acts

¹A group of algorithms that lie between supervised and unsupervised learning, where their input data is partially labeled.

²A group of algorithms that learn by interacting with their environment, where they are awarded when performing a right task, and punished otherwise.

as an instructor to guide the algorithm learn and generalise the mapping function such that given an unseen input test data x_1 , the model can accurately predict the target labels y_1 using the mapping function as: $y_1 = f(x_1)$. However, human labeling (labeled data) is frequently imperfect and the machine learning algorithm can only learn what is trained on. Hence any biases in a labeled training set will result in a biased machine learning algorithm.

Supervised learning is the most common used machine learning techniques in astronomy (e.g., Connolly et al. 1994; Lochner et al. 2016; Möller et al. 2016; Neira et al. 2020). Supervised learning can be subdivided into classification and regression algorithms depending on the target label available in the data.

2.1.1 Classification and Regression

Classification occurs when the target label is categorical, and the algorithm is tasked to classify between two or more target labels. The former is called binary classification, while the latter is referred to as a multi-class classification problem. Binary classification is the most commonly explored because of its simplicity: e.g., classification between transient and non-transient (see Neira et al. 2020), classification between gravitational microlensing and non-gravitational microlensing events (see Teimoorinia et al. 2020), classification between SN Ia and non-SN Ia (see Takahashi et al. 2020). Multi-class classification is also intensively explored because there are often more than two target labels in practice, e.g., classifying between different types of variable and transient stars (see Pashchenko, Sokolovsky, and Gavras 2018; Hosseinzadeh et al. 2020; Gabruseva, Zlobin, and Wang 2020).

Regression occurs when the target label is a continuous or real number. It aims to predict real numbers y_1 of a test set x_1 given a mapping function f that has been learned from the training data. These techniques have also been explored in astronomy: e.g., they were used for photometric redshift predictions (Schuldt et al. 2020; Jarvis, Hatfield, and Almosallam 2020; Dainotti et al. 2021; Curran, Moss, and Perrott 2021). The most commonly used algorithms for supervised learning are decision trees, random forest (RF; Breiman 2001), support vector machines (SVM, Pisner and Schnyer 2020), artificial neural networks (ANN; Wang 2003), and k-nearest neighbors (kNN), some of which can be adapted for both classification and regression problems, with random forest being an example.

2.1.2 Random Forest-An Example of a Classification Algorithm

Here we introduce RF because we will later use isolation forest, an anomaly detection algorithm based on the principles of RF. A RF algorithm is a collection of multiple decision tree algorithms. They fall under a group of algorithms called ensemble methods, algorithms built by combining multiple weak learners (e.g., decision trees, see an example decision tree in Figure 2.2) that work together to improve the overall performance. To understand the principles of RF, we first need to introduce bagging, which is an important algorithm adapted and improved in RF.

Given a training set T , a bagging algorithm draws multiple subsets of data samples from T with replacement (i.e., a single point in the T can be picked multiple times); this technique is known as bootstrapping, and the drawn samples are referred to as the bootstrap samples. The algorithm then builds multiple trees, one for each of the bootstrapped samples. The majority votes then give the classification of a new instance from the multiple trees (Liaw, Wiener, et al. 2002).

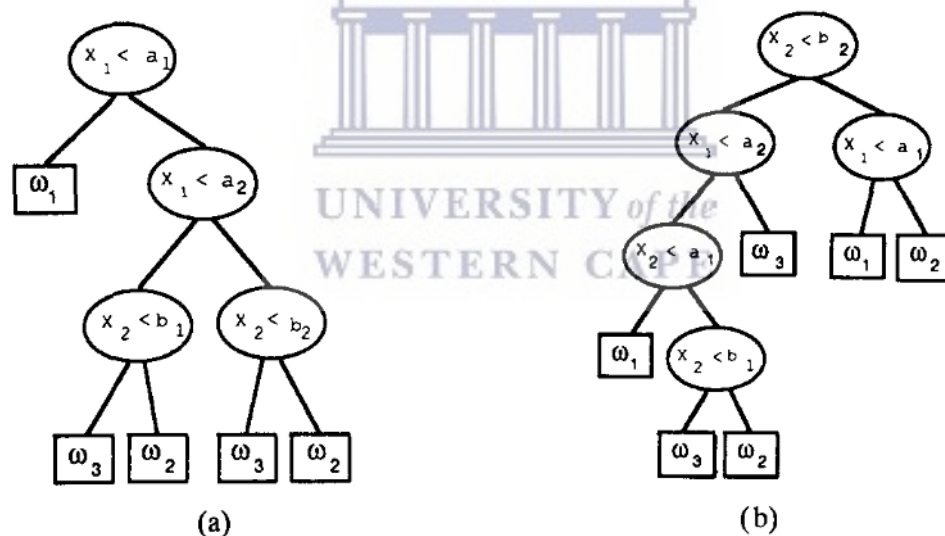


FIGURE 2.2: Schematic example of two decision trees, where x_1, x_2, x_3 are the features, and $\omega_1, \omega_2, \omega_3$ are the target labels. The nodes represent a selected feature through which the feature space is partitioned, and the branches represent the regions that satisfy the conditions specified in the nodes (left being true and right being false; Safavian and Landgrebe 1991).

RF uses an upgraded bagging technique which is different from the original one in that instead of splitting the nodes of the trees based on the best features in the entire original sample, and the splits are based on the best features that are randomly selected at each node (Breiman 2001). It takes the number of features required to split the tree nodes $[m$, which is often set to the square root of the total number of features in the original data (Breiman 2002)] and the number of trees (N) to build

as inputs. The algorithm then draws N bootstrap samples from, e.g., $\sim 67\%$ of the training data T , and the remaining $\sim 33\%$ of the T is used to test the uncertainties of the predictions, also known as “out-of-bag” data.

The algorithm then builds a tree for each bootstrapped sample, which grows by picking the best split feature among randomly selected features in the bootstrapped sample at each node. It then selects the best splitting features based mostly on two criteria: the information entropy and the Gini index.

Given a dataset with K possible classes/labels, the Gini index measures the purity of a class at each node after a split was made on a randomly selected feature x_i as:

$$G(C|x_i) = 1 - \sum_{q=1}^K P^2(C_q|x_i), \quad (2.1)$$

where $P(C_q|x_i)$ measures the likelihood that an instance belong to class C_q , given x_i . The purity can also be calculated using the information entropy as:

$$H(C|x_i) = - \sum_{q=1}^K P(C_q|x_i) \log_2 P(C_q|x_i). \quad (2.2)$$

Both the Gini index and information entropy are similar in that they measure the purity of a class at a given node given a randomly selected splitting feature. The optimum feature is the feature that gives fewer values for either the Gini index and information entropy.

The homogeneity of the classes at each node increases with decreasing Gini index and vice versa. The trees have different nodes: the parent nodes and the child nodes. A successful split occurs when the Gini index of the child node is less than that of a parent node, and the splitting continues until the Gini index is zero, leading to the terminal node (i.e., the final class). This process is repeated for N trees, and the final predictions are given by the majority votes (for classification), and the average predicted values (for regression). Random forest is famous for its high performance and will use it later in Chapter 5 to get insights into our data. We describe the general procedure for model fitting and performance analysis in the next section (section 2.1.3).

2.1.3 Building a Model and Analysing its Performance

Building a supervised machine learning model starts with data. The data is often disorganised (e.g., the data might have missing values, or the text in the data might be a mixture of lower and upper case letters).

Data Cleaning

Most machine learning algorithms require a homogeneous feature space, i.e., they do not function with missing values in the feature space metric. The feature space is composed of rows and columns, where the rows represent the instances and the columns are their features. The data with missing values are often preprocessed by different approaches: dropping the entire column with missing values or replacing the missing values mostly with the mean of values in that column. The latter is often preferred over the former because dropping a column can result in the loss of important information (Peng and Lei 2005).

The process described above is often referred to as data cleaning; data that has passed through this process is called “cleaned data.” The cleaned data can sometimes (very rarely) be used as input to a learning algorithm if its dimension is low and its features are non-redundant. However, some datasets can be high dimensional even after the data cleaning process. Many machine learning algorithms perform poorly with such datasets. High dimensional feature space can have highly correlated features, which might introduce a bias in the learning algorithm. The algorithms would then overfit the training data (see an example of an overfitted model in Figure 2.3) and not generalise the mapping function, leading to poor performance on unseen data. Dimensionality reduction techniques such as feature extraction are often employed to solve this problem (Gnana, Balamurugan, and Leavline 2016).

Feature Extraction

Feature extraction reduces the dimensions of high-dimensional data by computing a set of features that represent the original data in a non-redundant informative manner. It aims to obtain features that uniquely represent the data points and reduce the redundancy of the data. This is critical in machine learning because the algorithm will not perform well if the features are wrongly computed (Dong and Liu 2018). The feature extraction process can also result in missing values or invalid values, and the same procedure for data cleaning described above can be applied in this case. The process of data cleaning and feature extraction are often collectively called the data preprocessing process.

Model Selection and Fitting

Different algorithms perform better at some tasks than others. Each algorithm has hyperparameters that control the learning process. These parameters are flexible, and they can easily overfit data.

Overfitting occurs when the model does not learn a general mapping function between the features (X) and target (y); instead, the mapping function learns the training data too well such that it performs poorly on unseen data (see Figure 2.3). Multiple algorithms are often fitted with the training data to find the best one that performs better on the input data/features. The optimal model is then fine-tuned via a technique called model fitting to insure that it does not overfit the data.

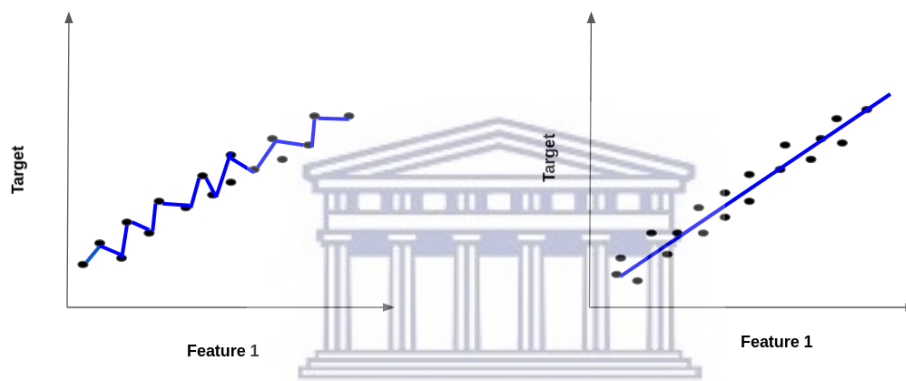


FIGURE 2.3: An illustration of an overfitted mapping function (left panel) and a generalised mapping function (right panel). The black points are the observations and the blue line is the mapping function that maps feature 1 to the target.

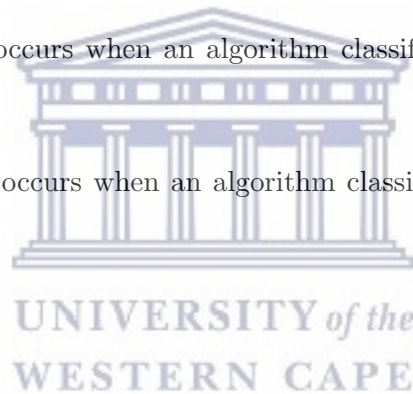
Model fitting is the process of fitting a model on the data to map a general function that can make accurate predictions on unseen data. At this point, the data is split into the training and validation set with X features and y target labels; and a test set with X_t features and their corresponding y_t target labels. This step is also critical as it helps the model from overfitting the training data, which would result in poor performance on unseen data. This is often achieved through a process known as the k-fold cross-validation (Refaeilzadeh, Tang, and Liu 2009).

In k-fold cross-validation, the training and validation data (X, y) are recursively divided into train and test sets, where an error score is computed for each iteration, and after k iterations, the average score is computed. This validation process is essential in supervised machine learning problems as it might prevent the model from being biased and overfitting the data. It is also crucial in selecting the best model (e.g., Jung 2018) and its corresponding optimum hyperparameters (an important parameter that controls the learning algorithm).

Analysing the Performance of a Model

Now that the optimum model has been fitted on the training data, we can assess its performance on unseen data (X_t, y_t) . There are a wide variety of techniques used to assess the performance of an algorithm; we will only cover the accuracy and confusion metrics in this work. To understand these metrics, we first need to introduce the following concepts, which are based on binary classification where the target labels are A (positive groups) and B (negative group):

- **True positive (TP)**: this occurs when an algorithm classifies an instance to be A while its true label is A
- **True negative (TN)**: this occurs when an algorithm classifies an instance to be B while its true label is B
- **False positive (FP)**: this occurs when an algorithm classifies an instance to be A while its true label is B
- **False negative (FN)**: this occurs when an algorithm classifies an instance to be B while its true label is A,



Accuracy is thus defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (2.3)$$

which measures how accurately an algorithm classifies instances in unseen data. It is one of the most commonly used metrics. On the other hand, the Confusion metric is a two-dimensional plot of the true target labels on one axis and the predicted target labels on the other. An algorithm is considered to be performing well if it has high percentage values along the diagonal of the confusion metrics (highlighted in dark gray in Figure 2.4).

| | | |
|---------|---------|---------|
| Class A | TP | FN |
| Class B | FP | TN |
| | Class A | Class B |

FIGURE 2.4: Confusion metrics of a binary classification problem between target label A and B.

We adopted both the accuracy and confusion metrics to assess the performance of RF, which we will discuss later in Chapter 5. We give a broad overview of unsupervised learning algorithms and introduce the anomaly detection techniques in section 2.2 below.

2.2 Unsupervised Learning

In unsupervised learning, the target labels are not available in the data (also known as unlabeled data). This means that unsupervised learning differs from supervised learning as its aim is not to build a mapping function between the input features X and target labels y ; instead, it aims to find underlying structures and patterns in the input features X . They do this independently without any human instructor. They have an advantage over supervised learning because getting labeled data can often be expensive and requires in-depth domain knowledge. They are categorised into four main groups: clustering, association, dimensionality reduction, and anomaly detection.

2.2.1 Clustering, Association and Dimensionality Reduction

With **clustering**, the algorithm is tasked to group instances in the input data (X) based on the underlying similarities found. These groups are known as clusters, and they can help identify patterns

in complex data. The most commonly used algorithms for clustering are k-means clustering and hierarchical clustering, some of which have been applied in astronomy (e.g., Hocking et al. 2018; Lee and Song 2019; Martin et al. 2020).

Unsupervised learning can also be used to find interesting associations between the input features (X) themselves. This group of algorithms is known as **association**. They can be widely applied in marketing, where the buying patterns of a customer is analysed by finding the associations in items they buy (e.g., an analyst may find that customers buy bread and juice together, and they would suggest that these items should be placed next to each other in the stores; Aggarwal 2015).

Unsupervised learning can also help reduce the dimensions of a high-dimensional features space. This is known as **dimensionality reduction**, where the algorithm is tasked to reduce the dimensions into a lower but essential dimension feature space that represents the original high dimensional feature space. They can be used to assess feature importance (a critical step for both supervised and unsupervised learning) by finding correlations between features, and correlated features are often removed as they will be redundant on the learning algorithm. The principal component analysis (PCA, Pearson 1901; Jolliffe and Cadima 2016) is often used for such tasks.

Dimensionality reduction can also be used as a visualisation tool where a high-dimensional feature space can be visualised on a two or three-dimension plots. This can be helpful to identify a group of instances that share the same properties in feature space, with t-distribution Stochastic Neighbour Embedding (t-SNE; Van der Maaten and Hinton 2008) being an example technique used for such analysis.

2.2.2 Anomaly Detection

With anomaly detection (AD), the learning algorithm is tasked with searching for outliers/anomalies in a given input sample (X). An anomaly can be an instance that deviates from a bulk of the “normal” instances in feature space (e.g. Boyajian’s star and calcium-rich transients). Traditional methods of finding outliers in data involved deep domain knowledge and manual programming that would define a thin line between the “normal” and anomalies, given training data (X). These techniques can be biased, i.e., they can only be applied to that specific domain. They also require constant maintenance because if there is a slight change in the domain, the expert would have to go and adjust the program to accommodate the new changes. These techniques would, however, fail for the current era of big data (D’Souza and Reddy 2021).

Current big data challenges have pushed the development of new robust automated AD techniques through machine learning. They can be both supervised and unsupervised. The supervised anomaly detection approach is just a classification problem with highly unbalanced classes, as such it is not useful in discovering new objects. In this work, we will only cover unsupervised AD techniques. Unsupervised learning techniques have the following advantages: they work with unlabeled data (which is the common data in practice), are adaptive, and can handle large datasets.

Unsupervised anomaly detection techniques can be applied to a wide variety of fields: e.g., fraud detection (see Monamo, Marivate, and Twala 2016; Agaskar et al. 2017), healthcare (see Baur et al. 2018; Haque, Rahman, and Aziz 2015) and astronomy (Pruzhinskaya et al. 2019; Webb et al. 2020; Storey-Fisher et al. 2020). The most commonly used algorithms for unsupervised AD problems are: isolation forest (iForest, Liu, Ting, and Zhou 2008), local outlier factor (LOF, Breunig et al. 2000), and one-class support vector machine (OC-SVM, Schölkopf et al. 2001). In this work, we used both iForest and LOF to search for anomalies in variable and transient light curve data.

2.2.2.1 Isolation Forest

Isolation Forest (iForest) is an unsupervised AD algorithm build from the principles of RF. It works by isolating points in a given training set based on their path lengths. It is built on the assumption that anomalies are “few and different,” which makes them easy to isolate (Liu, Ting, and Zhou 2008).

iForest uses ensemble binary decision trees (also known as iTrees) to isolate each object in a dataset. It works by picking a random feature and split value at each node of the iTree, and growing the iTree until the object is isolated. The number of partitions required to isolate an object is known as the path length, and the path length of an anomaly is expected to be short compared to that of a “normal” object. For example, Figure 2.5 shows that only four splits are required to isolate an anomaly (x_0) as opposed to twelve splits for the “normal” object (x_i).

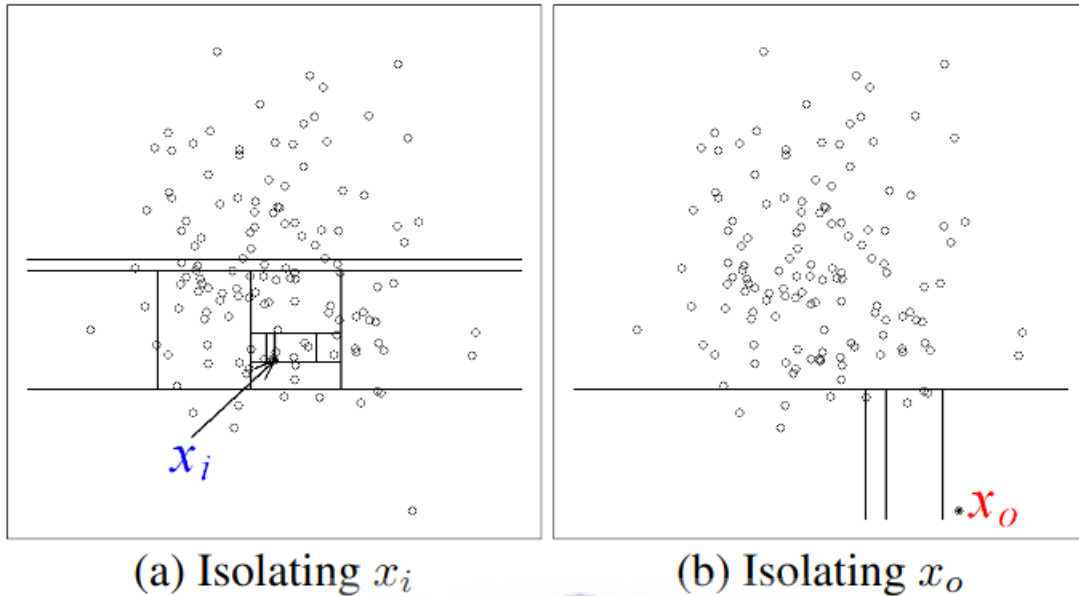


FIGURE 2.5: Demonstration that iForest requires few random splits to isolate an anomaly (x_o) as opposed to a normal instance (x_i ; Liu, Ting, and Zhou 2008).

iForest computes the anomaly score s of object x given n objects in the training set, as:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \quad (2.4)$$

where $h(x)$ is the object's path length, $E(h(x))$ is the average path length across the forest of iTrees, and $c(n)$ is a normalising factor which is derived from a harmonic number $H(i) = \ln(i) + 0.5772156649$ as follows:

$$c(n) = 2H(n-1) - 2(n-1)/n.$$

If the anomaly score $s \sim 1$, then object x is likely to be an anomaly; if $s < 0.5$, then x is likely to be normal; and if $s \sim 0.5$ for all points in the dataset, then the entire set does not have any anomalies. The advantage of iForest is that it can perform well with high dimensional datasets with redundant features (Liu, Ting, and Zhou 2008).

2.2.2.2 Local Outlier Factor

LOF is an unsupervised anomaly detection algorithm that computes an anomaly score of an object based on its local density relative to its k nearest neighbors. It is built on the assumption that anomalies are located in low-density regions instead of “normal” objects located in high-density regions (Breunig et al. 2000). The principle idea of the algorithm is to compare the local density of an object to that of its nearest neighbors, and “normal” instances will be those located in high-density regions, while anomalies will be those in low-density regions.

To compute the anomaly score, the algorithm first computes the distance $k\text{-distance}(x)$, which is the distance between object x and its k th nearest neighbour, and the complete set of k nearest neighbours $N_k(x)$, is thus the objects falling within the boundaries of $k\text{-distance}(x)$. The algorithm also computes the distance $\text{dist}(x,y)$, which is the distance between object x and y . It then calculates the maximum *Reachability distance* as:

$$\text{reach-dist}_k(x, y) = \max \{ k\text{-distance}(x), \text{dist}(x, y) \}.$$

The local reachability density of object x is thus calculated as:

$$\text{lrd}_k(x) = 1 / \left(\frac{\sum_{y \in N_k(x)} \text{reach-dist}_k(x, y)}{|N_k(x)|} \right), \quad (2.5)$$

where $|N_k(x)|$ is the sum of objects in $N_k(x)$. The local outlier factor of object x is finally calculated as:

$$\text{LOF}_k(x) = \frac{\sum_{y \in N_k(x)} \frac{\text{lrd}_k(y)}{\text{lrd}_k(x)}}{|N_k(x)|} = \frac{1}{|N_k(x)|} \sum_{y \in N_k(x)} \frac{\text{lrd}_k(y)}{\text{lrd}_k(x)}. \quad (2.6)$$

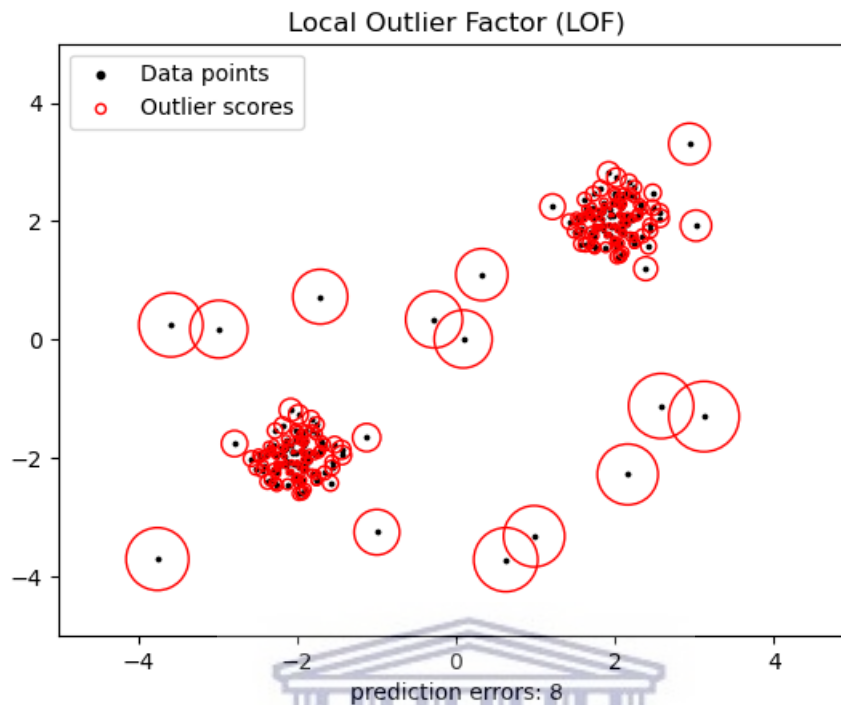


FIGURE 2.6: Demonstration of the LOF anomaly scores based on the local densities of objects. The radius of the circle indicates the anomaly score where a large radius corresponds to high anomaly scores. Points with big circles are located at low-density regions, while those with small circles are located at high-density regions (Pedregosa et al. 2011).

The anomaly score of objects in a given dataset is then calculated from equation 2.6, where objects with $\text{LOF} \leq 1$ are considered to be “normal,” while those with $\text{LOF} > 1$ are anomalies. This means that objects located in regions of low-density are anomalous and vice versa (see Figure 2.6). LOF can be helpful in cases where the outliers are relatively close to the “normal” objects in the feature space (Breunig et al. 2000).

2.3 Active Learning

Active learning algorithms are a group of algorithms that learn by interacting with an external user (also known as an oracle). They are often employed in cases where the input data is partially labeled, where they query an oracle to label some unlabelled instances in the data. Their ability to operate with partially labeled data makes them useful tools to use in practice. Most datasets are partially labeled because labeling data can be expensive and time-consuming. Active learning intelligently chooses objects to label, thus minimising user time while maximising performance (Settles 2009).

Active learning has three main query strategies: pool-based, stream-based selective sampling, and membership query synthesis. Pool-based (Lewis and Gale 1994) active learning techniques involve a learner that processes the entire input datasets, assigning a confidence score to each instance in the data. Objects with small confidence scores are presented to an oracle for labeling.

With stream-based selective sampling (Atlas, Cohn, and Ladner 1990), the learner processes each instance in the data one by one; it then decides if an instance should be labeled or not based on its query criteria. The learner query the oracle to label instances that meet the criteria.

Lastly, with membership query synthesis (Angluin 1988), the learner processed the input data and build a distribution describing the patterns found in the data. It then queries an oracle to provide a prediction to a sample generated from the distribution. These predictions are then re-introduced into the input data, meaning that the distribution of the data might change depending on the predictions provided by the oracle.

Active learning techniques have been applied in: speech recognition (Tur, Hakkani-Tür, and Schapire 2005), video classification (Hauptmann et al. 2006) and image classification (Zhang and Chen 2002) just to mention a few.

The active learning technique we use in this work is different from the above described techniques. We use a technique proposed by Lochner and Bassett 2020, where its basic principle is that we present the features of the transient and variable events to a machine learning anomaly detection algorithm. The algorithm then computes the anomaly scores of the objects. The objects are then presented to the user on the **ASTRONOMALY** frontend in a rank from the most anomalous to the least anomalous. The user can then assign a score to each of the objects (giving interesting objects a high score and vice versa). The active learning technique then computes a relevance score based on the input anomaly scores provided by the user. Lastly, the most interesting objects will then appear on the top ranks (see chapter 3.1.6 for more information).

The advantage of our technique is that it works with very few labels and allows the user to define what is interesting. It also works with any machine learning algorithm and incorporates uncertainty in the regression of the user interest score. The disadvantage is that because the user always gets presented with the most anomalous objects and not necessarily the ones that would improve the active learning the most, the algorithm won't improve as quickly as a more traditional active learning approach.

Our approach is not used to improve the performance of the anomaly detection algorithms, rather used to personalise interesting anomalies. Another interesting question that one can ask is: can this

technique be applied for real-time anomaly detection? Of course, in principle these techniques can be used for real-time anomaly detection, but in practice may require different feature extractions and algorithms (Muthukrishna et al. 2021).

2.4 Anomaly Detection Applied to Transient Astronomy

Traditional methods of detecting new objects in astronomy involved careful analysis of a specific object and comparison with known objects. If the object does not fit in any of the known classes, then further careful analysis is performed to check if the weird behavior of the object is not due to artefacts. Finally, if the unique behavior of the object is due to an astrophysical process, it is then regarded as a discovery of a new object (Ekers and Kellermann 2011). However, these techniques would require months and many experts to analyse in the current era of big data; hence automated techniques such as anomaly detection (AD) with machine learning are required.

There have been efforts to employ AD algorithms to search for anomalies in transient astronomical data. These include both supervised and unsupervised learning algorithms. Nun et al. 2014 developed a supervised AD algorithm by combining the random forest (RF) classifier and Bayesian network to search for anomalies in labeled MACHO (Alcock et al. 1999b) light curves. Each light curve was represented by 13 features described by Pichara et al. 2012 which is a combination of continuous autoregressive [CAR(1); Belcher, Hampton, and Wilson 1994] and time-series features.

Their approach starts with a RF classifier, where they train it with a sample of y_k known variable classes to get a voting distribution for each of the known variable events. For each light curve x_i in the training set, RF returns a vector $\{v_{i1}, \dots, v_{ik}\}$ which is the probability that x_i belongs to the class y_j where $v_{ij} \in [0, 1]$, $j \in [1, \dots, k]$, and $\sum_{j=1}^k v_{ij} = 1$.

Their next step involves discretisation (a process used to transform continuous variables into discrete forms) of the probabilities, after which they obtain a dataset $V = \{v_1, \dots, v_n\}$ where $v_i = \{v_{i1}, \dots, v_{ik}\}$. The V dataset gives information about how RF assigns votes among objects that belong to the known classes. They finally train a model using Bayesian network to learn the joint probability distributions over V ; this joint probability is used to learn the decision mechanism of the RF classifier. An unlabelled object is analysed by first obtaining a voting distribution $\{v_{i1}, \dots, v_{ik}\}$ from the trained RF classifier and computing a joint probability associated with its voting distribution $P(v_{i1}, \dots, v_{ik})$ using the trained Bayesian network. The outlier score is thus computed as $1 - P(v_{i1}, \dots, v_{ik})$; the lower the joint probability, the higher the outlier score, and the corresponding object is thus an outlier.

This AD algorithm has proven to be effective as it has detected anomalous objects in the MACHO light curves; however, some of the detected anomalous objects were found to be due to artefacts. To take advantage of the supervised nature of the algorithm, they repeatedly labeled these objects and reintroduced them in the data until they were no longer flagged as anomalies. The remaining anomalies detected in the data were rare known variables like the cataclysmic variables, blue variables and eclipsing Cepheids, and anomalies with unknown variability types.

Similarly, Richards et al. 2012 fitted a RF classifier algorithm to the ASAS (All-Sky Automated Survey) light curves to classify the objects into 28 known variable classes. Seventy-two features represented each object in the data, 67 of which are time-series features (e.g., Amplitude, Standard Deviation and Period) described by the author and Richards et al. 2011, and five color features. The pipeline is called MACC (machine-learned ASAS Classification Catalog), which works by training a RF classifier coupled with active learning algorithms that allow the human user to interact with the classifier. When an unlabelled object is introduced to MACC, it is assigned a class, and an anomaly score.

The anomaly score is computed based on the distance metric from the unlabeled object i to each object j in the training data. The final anomaly score of object i is thus the distance to its nearest neighbor. The calculation takes advantage of the output proximity estimate value ρ_{ij} from RF, which is a fraction of trees in the RF that has the same feature vectors (for both i and j) in the leaf nodes. Objects with an anomaly score > 10.5 are considered to be anomalies. Using these techniques, they detected anomalous objects with various characteristics: rare variable objects with a period of \sim one year, an active unknown pulsating Be star, a semi-regular pulsating Be star, and other aperiodic variables with outbursts. Both techniques described in Richards et al. 2012 and Nun et al. 2014 assumes prior knowledge of the labels of the variable stars (i.e., they are supervised learning algorithms), and their techniques are promising, as they detect anomalies. However, supervised learning techniques require labeled data, which in practice is hard and expensive to obtain (e.g., spectroscopic follow-up studies are required to verify accurate classifications of variable stars). Some of the data might be mislabelled which then results in less accurate predictions (Omar, Ngadi, and Jebur 2013).

An alternative to supervised AD is unsupervised AD algorithms. Rebbapragada et al. 2009 modified the k-means clustering algorithm to develop an unsupervised AD algorithm called PCAD (Periodic Curve AD) to search for anomalies in periodic unsynchronised light curves. This technique uses an algorithm called Phased K-means (Pk -means), which works similar to k-means in that it requires an initial value of k (the number of clusters) and assigns each of the light curves in the data a cluster

based on its closest centroid. The algorithms take the raw light curve values (time and magnitude) as features and use their periods to fold them. The light curves are then represented by their phased durations, which are adjusted to optimise the maximum correlation between the light curve and each k centroids. Lastly, the anomaly score of an object (light curve) is the distance between it and its nearest centroid, and the objects are then ranked based on the anomaly score where the most anomalous objects appear at the top of the list.

They tested the PCAD algorithm on light curves from the Optical Gravitational Lensing Experiment Survey with three variable classes, Cepheids, Eclipsing Binaries, and RR Lyrae. From this sample, they detected three main categories of anomalies: bogus light curves, misclassified light curves, and a potential new class of periodic variable stars.

Pruzhinskaya et al. 2019 also used an unsupervised AD algorithm, iForest, to search for anomalies in the Open Supernova Catalog (Guillochon et al. 2017). Their feature extraction process includes a technique called multivariate Gaussian processes (GP), which helps make the light curve data homogeneous (iForest requires homogeneous input data to operate). Given an object with light curves in different filters (e.g., *gri*), multivariate GP works by first finding the cross-correlation between all light curves and approximating a fit to all light curves using GP. This fit can then be used to extrapolate light curves with missing values given a range of values. For example, the missing peak magnitude in the *g* filter of an object can be approximated from the *r* and *i* filter using multivariate GP. Each object in the data was represented by 374 features, a combination of the normalised flux values from the multivariate GP, the maximum light curve flux value, and the fitted parameters from the GP. They further reduced the high dimensional feature space (374 features) to eight features using the t-SNE techniques and fitted iForest to the eight features. This is because they suspect that the original high dimensional data can be too sparse and reduce the performance of the iForest algorithm.

They detected anomalies with various characteristics where out of a sample of 1999 objects, only 5% of the sample was found to be anomalous objects. They also found that 16% of the detected anomalies have been misclassified as SNe while their true nature is likely to be stars or quasars. The interesting anomalies detected include Peculiar SNe Ia, Peculiar SNe II, and Superluminous SNe.

Malanchev et al. 2021 proposed a pipeline called SNAD that can be applied to both transient and variable light curves. It operates in three main steps of analysis: data processing (including minimum point cut in each light curve and feature extraction), anomaly detection using multiple algorithms, and follow-up studies with experts in the field. They extracted 42 features from light curve data from the Zwicky Transient Facility (ZTF; Bellm et al. 2018) third public data release. The feature extraction

process, in this case, is more general and involves features that describe the light curves in terms of features characterising the magnitudes of the light curves (e.g., amplitude, standard deviation, and mean) and those characterising the Fourier transform of the light curves (e.g., periodogram standard deviation and periodogram amplitude).

Multiple unsupervised AD algorithms were then employed to search for anomalies in the extracted features. This algorithms includes iForest, LOF, one-class support vector machines (O-SVM; Schölkopf et al. 1999) and Gaussian mixture model (GMM; McLachlan, Lee, and Rathnayake 2019). The top 40 anomalous objects (in terms of anomaly scores) from each algorithm were then presented to an expert for analysis. These anomalies were categorised as bogus outliers (due to artefacts) and anomaly candidates (of potential astrophysical interest). Out of the input data set with ~ 2.25 million objects, only 0.01% were detected as anomalous; and 68% of the detected anomalies were due to artefacts, while 32% were interesting anomalies. Lastly, 25.8% of the interesting anomalies were uncatalogued, and these findings prove that the AD algorithms can help with new discoveries in big data. However, the large sample of detected bogus light curves (light curves with anomalies due to artefacts) raises an alarm; this indicates that interactive learning techniques (among others) are required to customise the AD process.

Similarly, Martínez-Galarza et al. 2020 employed multiple unsupervised AD algorithms: iForest, Unsupervised Random forest (URF; Shi and Horvath 2006), t-distributed Stochastic Neighbor Embedding (t-SNE), and uniform manifold approximation and projection (UMAP; McInnes, Healy, and Melville 2018), to search for anomalies in light-curve data from the Kepler space telescope with a known anomaly, the Boyajian's star. They criticise the FATS (Nun et al. 2015a; currently updated to *feets*) feature extraction package as the package may return invalid values for instances where light curves do not meet the minimum requirements, which might lead to biased results from an AD algorithm. They thus used an alternative feature extraction technique where they considered the light curve data (time, magnitude, and magnitude errors) and the Fourier transform of the light curves [in terms of a periodogram from the Lomb-Scargle (Lomb 1976; Scargle 1982)] as features for their model. From the Lomb-Scargle, they computed periodograms on discrete frequency values and used the power values at each frequency as features. They fitted the data to the iForest and URF, then later used the t-SNE and UMAP to reduce the dimensionality of the features to two dimensions and examined the correlation among the detected anomalies. The two AD algorithms detected different objects as anomalies where the majority of the detected anomalies were associated with sharp peaks (which can be due to artefacts or could be a rare astronomical phenomenon like stellar flares), light curves with dips that can be linked to stellar eclipses (the Boyajian's star was also detected as an anomaly), and stellar

pulsations. Although their results are promising, using the light curve values as features is not ideal for most transient and variable light curve datasets as they are not translationally invariant and don't cleanly separate out classes. Hence a feature extraction technique is needed to reduce the complexity of the light curve data and extract useful information for classification.

Villar et al. 2021 explored a different feature extraction technique using neural networks through a technique called variational recurrent autoencoder neural network (VRAENN) and employed iForest to search for anomalies in extragalactic transients (SN-like) in the PLAsTiCC data (Allam et al. 2018). Their pipeline involved: interpolating the light curves using GP, encoding the interpolated light curves using VRAENN, and lastly using the encoded vectors as inputs to the iForest algorithm, which then assigned each light curve an anomaly score. They also show that given a new SN-like light curve, their technique can detect if the SN-like event is an anomaly or not before its peak luminosity. They detected interesting anomalies from the PLAsTiCC data, including the KNe, Type I Superluminous SNe, AGN, and Intermediate Luminosity Optical Transients. Their technique has the advantage of handling unevenly sampled light curves and is not biased to anomalies that are due to artefacts. Their results are presented in a ranked manner according to the anomaly scores from iForest, and some of the normal classes are ranked as anomalous. This means that other techniques are required to help the AD learn which of the detected anomalies are interesting or not. This problem can be solved by an active learning approach where the user interacts with the machine by provided labels to objects according to their interest.

How interesting an anomaly is, depends on the user; e.g., an instrumental scientist would find the anomalies due to artefacts interesting, while an astrophysicist would find astrophysical anomalies interesting. This brings us to the question; How can we help the AD algorithms learn how to flag anomalies that are not of interest to a user? Lochner and Bassett 2020 developed an anomaly detection framework called ASTRONOMALY (Lochner and Bassett 2020) that runs the full pipeline of AD techniques using a python back end and presents the results on a JavaScript front end which allows the user to explore and interact with data. To answer the question above, ASTRONOMALY presents the objects in the frontend in a rank manner according to their raw anomaly score from the algorithms. The user can assign a score between zero and five to any object where five indicate the most interesting objects and zero indicate the least interesting. The user can then train astronomaly after assigning the scores, and objects related to those assigned high scores will appear in the top ranks for further analysis (more details about ASTRONOMALY will explain in the next Chapter).

Webb et al. 2020 used the ASTRONOMALY framework to search for anomalies in 85 553 fast cadenced

transient light curves obtained through the multiwavelength Deeper, Wider, Faster (DWF) program. Their full pipeline starts outside the **ASTRONOMALY** framework where they extracted features from the light curves using FATS and adopting other features from Richards et al. 2011, and Kim et al. 2011. They then clustered the objects using Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN; McInnes, Healy, and Astels 2017). The Unclustered objects were explored by manual inspection and by running **ASTRONOMALY** on them where iForest was used to assign an anomaly score to each of them and present them in a rank according to the score. Through the manual inspection, they detected nine interesting variable anomalies, three of which were newly discovered, and they also discovered an ultrafast flare anomaly. They then investigated as to at what rank will the detected anomalies appear in **ASTRONOMALY** and found that the variable source and ultrafast flare anomalies were retrieved within the top 280 and 600 ranks.

The studies described above shows that unsupervised AD algorithms are ideal tools for searching for anomalies in big datasets. They also indicates that the algorithms detect two main types of anomalies: artefacts and interesting anomalies. Which means that they cannot distinguish between the two types. A human can help the algorithms learn which anomalies are interesting or not. Active learning techniques, as seen in from Lochner and Bassett 2020, have proven to be effective in this regard.

In this work, we updated **ASTRONOMALY** such that it can be generalised to most of the light curve data format (details described in the next Chapter). We then tested it on both the transient and variable light curve data. Lastly, we used active learning approaches to flag less interesting anomalies and retrieve more of the interesting anomalies in the top ranks.

Chapter 3

Methodology

The main goal of this work is to employ anomaly detection techniques to search for anomalies in both transient and variable light curve data. These techniques were applied to two datasets: light curves from the Catalina real-time transient survey (CRTS; Drake et al. 2009) and Photometric LSST Astronomical Time-Series Classification Challenge (PLAsTiCC; Allam et al. 2018). The former data is from real observations (see details in section 4.1) while the latter is simulated (see details in section 5.1). This means that we can investigate the anomalies detected in CRTS data (see details in section 3.2), and it also means that we can take advantage of the simulated data to assess the performance of the anomaly detection algorithms. The latter is possible because we know the anomalous sample present in the dataset, and we can use the ranking to evaluate how accurately the algorithms detect these anomalies (see section 3.1.7).

In this work, we follow the methodology covered by most machine learning problems. This includes data management, feature extraction, and anomaly detection with unsupervised machine learning algorithms. We used and extended a python package called **ASTRONOMALY**¹, to process light curve datasets, extract features from the data, employ anomaly detection (AD) algorithms to search for anomalies in the given data, and employ active learning to personalise interesting anomalies. We then investigated anomalies detected in CRTS data and assessed the performance of the algorithms on anomalies detected in the PLAsTiCC data.

¹<https://github.com/MichelleLochner/astronomy>

3.1 ASTRONOMALY

Most machine learning problems are solved using a standard procedure. This includes reading in the data, extracting features from the data, and employing an AD algorithm to search for anomalies (see section 2.1.3). For example, the same procedure of searching for anomalies in spectral data can be repeated for light curve and image data. However, the main difference is the data structure and feature extraction techniques. This means that one can automate the full anomaly detection pipeline in one setting by customising the data management and feature extraction process. The remaining processes can be standardised by choosing a suitable algorithm for the specified data and task at hand.

A suitable algorithm, in our case, is determined by the rank weighted score (RWS) and recall metric (see section 3.1.6). For example, we choose LOF over iForest in chapter 5 because it did better using both RWS and recall.

To automate the anomaly detection process, Lochner and Bassett 2020 developed a framework called **ASTRONOMALY** that couples anomaly detection algorithms with novel active learning techniques to personalise interesting anomalies according to a human expert. It is built to answer two essential questions: 1) how do we generalise the anomaly detection process (i.e., how do we automate the process from data processing to anomaly detection with minimum programming), and 2) how do we personalise interesting anomalies?

ASTRONOMALY addresses question 1 through its python backend pipeline (see Figure 3.1), where it covers the data processing, feature extraction, and anomaly detection processes. It then addresses question 2 through its JavaScript frontend web interface by using the anomaly scoring tab (see Figure 3.3). The detected anomalies are visualised in a ranked manner according to their raw anomaly scores (where the most anomalous objects appear at the top of the list). A user can then assign a relevance score to the top N anomalies where interesting anomalies are assigned high scores and less interesting a low score. The user can then retrain **ASTRONOMALY**, and it will then recommend anomalies that are similar to those assigned high relevance scores.

ASTRONOMALY can be applied to most astronomical data types (e.g., images, spectra, and light curves), and it can also be easily be extended to operate with other data types (Lochner and Bassett 2020). It currently incorporates two anomaly detection algorithms: LOF and iForest, both of which we use in this work. We have extended **ASTRONOMALY** to be more general to most light curve datasets and applied it to the CRTS and PLAsTiCC light curve data. We describe below, the full **ASTRONOMALY**

pipeline in detail and how we extended it to operate with light curve data (see the code in appendix B).

3.1.1 Data Management and Preprocessing

Data management is the first step in any machine learning problem, involving reading in the data and preprocessing it before the feature extraction process. *ASTRONOMALY* is built to generalise the process of reading in data from most astronomical data types (e.g., images). Even though it has been applied to light curve data (see Webb et al. 2020; Lochner and Bassett 2020), the data management process was not general to most light curve data. We extended it to be more general to most light curve data files with the following characteristics (see the code in appendix B):

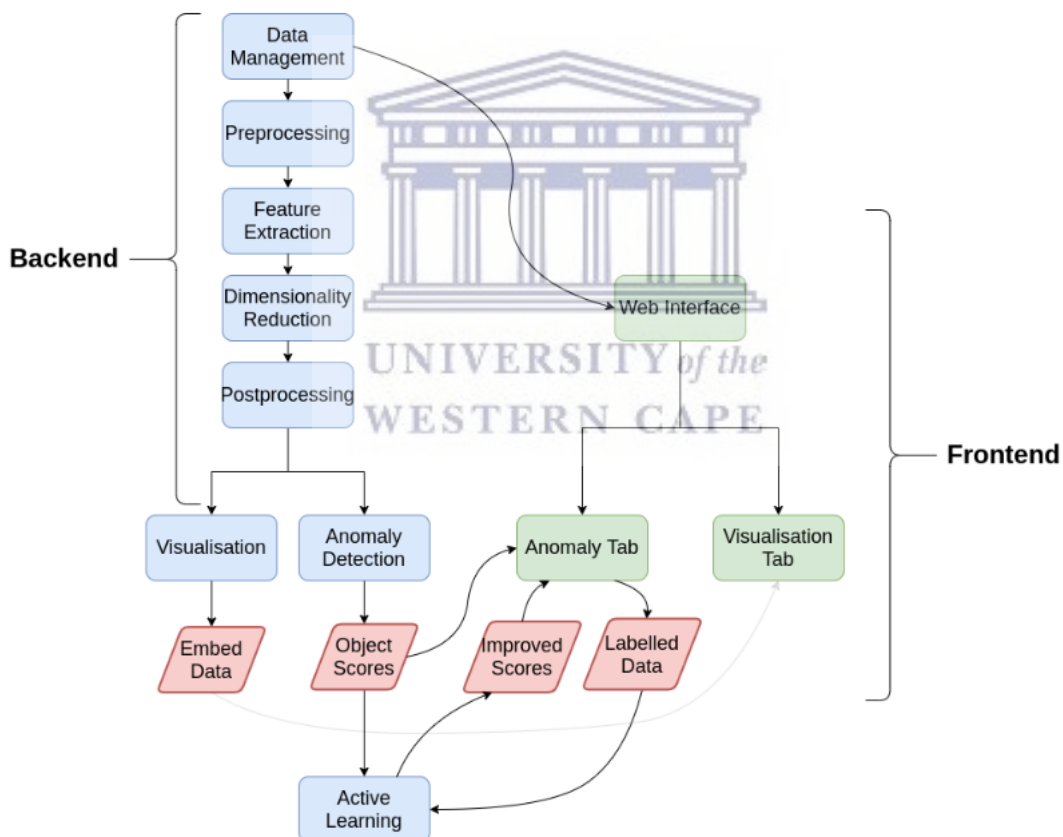


FIGURE 3.1: The *Astronomy* pipeline. The backend follows standard data processing and machine learning algorithm steps. It starts with reading in the data and preprocessing it (section 3.1.1), extracting useful features from high dimensional data (section 3.1.2), dimensionality reduction (not applicable to this work), data postprocessing (section 3.1.3), and anomaly detection (section 3.1.4). The input data (light curves in our case) and their corresponding anomaly scores (computed by the anomaly detection algorithms) are presented to the JavaScript frontend for labeling and visualisation purposes (see section 3.1.5 and subsections therein) where the objects will be presented in a ranked manner according to the anomaly scores. The anomaly tab has an option to assign objects relevance scores which are then used for active learning (see section 3.1.6 Lochner and Bassett 2020)

- Light curves of multiple objects embedded in one big file,
 - Light curves with time and magnitudes/flux observations in one broad passband (e.g., the V band for CRTS, with or without error), we refer to this as **case_1a**.
 - Light curves with time and magnitude/flux observations in multiple passbands (e.g., *ugrizy* in PLAsTiCC), we refer to this as **case_1b**.
- Light curves of individual objects that are saved in a directory where each object has its own filename; we refer to this as **case_2**.

This was done by extending an existing class called `light_curve_reader`, to generalise the data management process as follows:

1. Prompt the user to:

- Specify the file paths: filename for **case_1a** and **case_1b** and directory for **case_2**.
- Specify the number of rows covered by the header text (this is important because removing the header from the files will make it easy for us to standardise the column names) in the data files.
- Specify the column index that corresponds to object IDs (with an exception for **case_2** since the IDs are simply the filename), time, magnitude/flux, errors, passbands, and labels. Most of these are treated as boolean, because our program first checks if some of the columns are missing, and naturally adjust the data management process that follow.
- Specify if they want to use the flux values themselves or convert them to magnitude. This applies to all cases where the light curve comes with flux values instead of magnitudes.

2. Reading in the data:

Now that we have the file path to the light curve file(s), we use PANDAS (McKinney et al. 2011) to read in the file(s) and standardise their columns by renaming them as follows:

- Time column: `time`
- Magnitude column: `mag`
- Magnitude error column: `mag_error`
- Flux column: `flux`

- Flux error column: `flux_error`
- Passbands column: `filters`

3. Converting flux and flux errors to magnitudes (m) and magnitude errors (σ_m):

$$m = f_0 - 2.5 \log_{10} f, \quad (3.1)$$

where f_0 is the flux of a standard source and f is the observed flux. The magnitude errors were estimated using the propagation of errors relation given by:

$$\sigma_m^2 = \left(\frac{\partial m}{\partial f} \right)^2 \sigma_f^2, \quad (3.2)$$

where m is the magnitude defined by equation 3.1 and σ_f is the recorded flux errors. Differentiating equation 3.1 with respect to f we get:

$$\frac{\partial m}{\partial f} = -2.5 \frac{\partial}{\partial f} \log_{10}(f), \quad (3.3)$$

but we can substitute $\log_{10}(f) = \frac{\ln f}{\ln 10}$ in 3.3 to get:

$$\frac{\partial m}{\partial f} = \frac{-2.5}{\ln 10} \frac{\partial \ln(f)}{\partial f} \approx \frac{-1.09}{f}. \quad (3.4)$$

Substituting equation 3.4 in 3.2, and taking the square root both sides we get:

$$\sigma_m \approx 1.09 \frac{\sigma_f}{f}. \quad (3.5)$$

Note that this step is optional, and can be neglected if the feature extractor tool can operate with flux values themselves (which is true for this work).

4. Returning a PANDAS dataframe with standard column names: changes described above were then applied to `case_1a` and `case_1b` data, we then returned a dataframe with standard column names.

The `case_2` data is, however, treated differently: we read in and standardise the columns in each individual file and concatenate all the light curves into one PANDAS dataframe; which then takes the same properties as `case_1a` and `case_1b`. We then return the dataframe.

The standardised light curve datasets were then used as an input to one or more feature extraction techniques described below.

3.1.2 Feature Extraction

The next critical step in machine learning is feature extraction. Currently, **ASTRONOMALY** has feature extraction tools for image data: ellipse-fitting, wavelet decomposition, and power spectral density (Lochner and Bassett 2020). It, however, does not have a feature extractor for light curve data. This means that we need to extend it to be able to extract features from data returned in section 3.1.1.

To do this, we added a feature extractor class called `feets.features` which makes use of a python package called the feATURES eXTRACTOR for tIME sERIES (`feets`; Cabral et al. 2018). We also adopted features from another python package called `avocado`² (Boone 2019), however, the features were extracted outside the **ASTRONOMALY** environment. It will be incorporated in **ASTRONOMALY** in the near future.

3.1.2.1 Feature Extraction With `feets`

`Feets` is an open-source python package that is designed to extract features from any time-series data, including light curves. It is an updated version of a package known as `FATS`, which was also designed for the same purpose; it, however, had some restrictions (e.g., it operates on python 2.7; Cabral et al. 2018). Cabral et al. 2018 adopted the same features computed by `FATS` and updated it to operate with python 3 and to be an open-source where the community can add more features to it. The new updated version was then renamed `feets`. `Feets`, which computes a comprehensive set of features, was successfully applied in astronomy, particularly for classification of variable stars (Khalil, Fantino, and Liatsis 2019; Gabruseva, Zlobin, and Wang 2020; Hosenie et al. 2019; Sirigiri 2019). Hence, it is chosen in this work as a feature extractor. We describe below, its feature extraction procedure.

Input Data and Data Preprocessing

The data returned in section 3.1.1 is in a form a **PANDAS** dataframe with standardised light curve columns. We designed **ASTRONOMALY** such that it can incorporate the basic procedure `feets` follows

²<https://github.com/kboone/avocado>

when extracting features. This includes taking light curve data as an input and computing features from it. This was done by adding a new class called `feets_features` which takes the output light curve data from the `light_curve_reader` and computes features from it. `feets` takes the following as inputs:

- **Time**: this corresponds to the observation time column (`time` column in our case),
- **Magnitude**: this corresponds to the brightness column (`mag` or `flux` column in our case),
- **Error**: this corresponds to the brightness errors (`mag_error` or the `flux_error` column in our case).
- **Time2, Magnitude2 and Error2**: this corresponds to the observations in a different band.

It is important to note that some of the inputs specified above might not be available in other light curve datasets. However, `feets` is designed in such a way that a user can specify the columns available in their light curves, and it can then compute features that are only supported by these columns. We automated this process in `ASTRONOMALY` such that it scans through the columns from the output of the `light_curve_reader` and computes features based on them.

We also added a preprocessing step in the `feets_features` class, whereby we prompt the user to give the light curve point cut number N . Objects with light curves points $< N$ are then discarded from the dataset, and we then extracted features for light curves with points $\geq N$. This is because some features in `feets` are sensitive to the number of points in a light curve.

Given a light curve of an object, `feets` computes features based on the columns available in the light curve and returns a one-dimensional array with values corresponding to the computed features. If a dataset has N objects, and `feets` computes M features, then an $N \times M$ matrix is returned.

Extracting Features

After the light curve data is preprocessed and the available columns detected, `feets` can be employed to extract features.. We designed `feets_features` such that it only computes features based on the available columns. It is also designed to prompt the user to specify features to exclude when computing features if required. Table 3.1 shows features that can be extracted with the light curve columns available in our data.

It is important to note that other features can be computed by **feets**; most of these are based on the colour characteristics of the light curves. Even though the PLAsTiCC data has observations in different bands, we chose to neglect the colour features, because some objects has missing values in some bands, this will result in a heterogeneous feature space. Instead, we computed features for each passband and concatenated them to form a vector with all computed features. For example, if we compute the **amplitude** feature for all six bands in PLAsTiCC (*ugrizy*), then the final vector will have the following as outputs: **amplitude_u**, **amplitude_g**, **amplitude_r**, **amplitude_i**, **amplitude_z** and **amplitude_y**, where **u**, **g**, **r**, **i**, **z** and **y** are features computed in different bands. However, this can be improved in future by interpolating the missing values using 2D Gaussian processes as discussed in section 3.1.2.2.

Some of the features computed by **feets** are derived from the Lomb-Scargle periodogram, Fourier components (Richards et al. 2011), CAR (Brockwell and Davis 2002), and Stetson (Stetson 1996) statistics.

- Lomb-Scargle Periodogram Features

Lomb-Scargle periodogram is a period finding algorithm that is used to find periods in both evenly and unevenly sampled data. It is often preferred over the discrete Fourier transform (DFT) algorithm because the DFT algorithm is built on the assumption that the data points are evenly sampled, which is not always the case in astronomy. Lomb-Scargle algorithm works by decomposing the time-series data into a linear combination of sinusoidal functions with the following characteristics: $y = a \cos \omega t + b \sin \omega t$. It does this by first transforming the observations from time-domain to frequency-domain and computing a periodogram as follows:

$$P(\omega) = \frac{1}{2\sigma^2} \left(\frac{\left[\sum_{n=1}^N (m_n - \bar{m}) \cos [\omega(t_n - \tau)] \right]^2}{\sum_{n=1}^N \cos^2 [\omega(t_n - \tau)]} + \frac{\left[\sum_{n=1}^N (m_n - \bar{m}) \sin [\omega(t_n - \tau)] \right]^2}{\sum_{n=1}^N \sin^2 [\omega(t_n - \tau)]} \right), \quad (3.6)$$

where $\omega = 2\pi/T$, T is the period and τ can be calculated from:

$$\tan(2\omega\tau) = \frac{\sum_{n=1}^N \sin(2\omega t_n)}{\sum_{n=1}^N \cos(2\omega t_n)}. \quad (3.7)$$

Periodic light curves can be folded using equation 1.1 to form what is known as phase folded light curves. **feets** computes the Lomb-Scargle periodogram for each object in the data. It then estimates

the object's period by converting its frequency at peak periodogram value to period. The period is then used to fold the light curve of the object (Kim et al. 2011; Kim et al. 2014). Features are then computed from both the periodogram itself and the phase folded light curves. These features, along with others, are described in table 3.1 below.

- *Fourier Components Features*

Features extracted here are computed by first representing the light curves of the objects using the superposition of sinusoidal functions of the form:

$$y_i(t|f_i) = a_i \sin(2\pi f_i t) + b_i \cos(2\pi f_i t) + b_{i,o}, \quad (3.8)$$

where f_i are the sinusoidal frequencies normalised by the constants a and b ; $b_{i,o}$ is the magnitude offset. χ^2 minimisation is then applied to equation 3.8 to find the periodic variation in the data as follows:

$$\chi^2 = \sum_k \frac{(d_k - y_i(t_k))^2}{\sigma^2} \quad (3.9)$$

where σ is the error in the measurement of data point d_k . The generalised Lomb-Scargle periodogram is then defined as:

$$P_f(f) = \frac{(N-1) \chi_o^2 - \chi_m^2(f)}{2 \chi_o^2} \quad (3.10)$$

where $\chi_m^2(f)$ is χ^2 minimised over a, b and b_0 .

$$\chi_o^2 = \sum_k \frac{(d_k - \mu)^2}{\sigma_k^2}, \mu = \frac{\sum_k d_k / \sigma_k^2}{\sum_k 1 / \sigma_k^2}. \quad (3.11)$$

Each light curve is then fitted with the harmonic sum of the sinusoidal function y_i plus a linear term as follows:

$$y(t) = ct + \sum_{i=1}^3 \sum_{j=1}^4 y_i(t|j f_i), \quad (3.12)$$

where f_i is the frequency that can only have four harmonics at $f_{i,j} = jf_i$ frequencies. The frequencies f_i are found by recursively searching for periodic signals in $P_f(f)$ and removing them from the data.

The algorithm works by finding a peak in $P_f(f)$, fitting the model $y(t)$ with a frequency corresponding to the peak $P_f(f)$ and the 2, 3 and 4 frequency harmonics. The algorithm then subtracts the fitted model from the data and updates χ_o^2 . The process is then repeated to find more periodic components.

The final features are that given as the phase and amplitude (see table 3.1 for a full list of these features):

$$A_{ij} = \sqrt{a_{i,j}^2 + b_{i,j}^2} \quad (3.13)$$

$$PH_{i,j} = \arctan\left(\frac{b_{i,j}}{a_{i,j}}\right) \quad (3.14)$$

where $PH_{i,j}$ is the phase component of the i th frequency with the j th harmonic and A_{ij} is the amplitude. The final phase is then computed as:

$$PH'_{i,j} = PH_{i,j} - PH_{0,0}, \quad (3.15)$$

where $PH_{0,0}$ is the phase of the first component (Richards et al. 2011).

- CAR Features

CAR is used to model the irregular sampled time series data. It has three parameters: the mean (CAR_mean), variance (CAR_sigma) and relaxation time (CAR_tau). It is defined by the equation:

$$dX(t) = -\frac{1}{\tau}dt + \sigma_C\sqrt{dt}\epsilon(t) + bdt, \quad (3.16)$$

for $\sigma_C, t, \tau \geq 0$, where τ is the relaxation time of $X(t)$, $\frac{\tau\sigma_C^2}{2}$ is the variance and $b\tau$ is the mean value of $X(t)$. σ_C is parameter describing the variability of the light curve with time observations less than τ , and $\epsilon(t)$ describes the “white noise” process with a variance equal to one and mean of zero.

The likelihood function of equation 3.16 can be computed as follows:

$$p(x|b, \sigma_C, \tau) = \prod_{i=1}^n \frac{1}{[2\pi(\Omega_i + \delta_i^2)]^{1/2}} \exp\left(-\frac{1}{2} \frac{(\hat{x}_i - x_i^*)^2}{\Omega_i + \delta_i^2}\right) \quad (3.17)$$

where $x = \{x_1, \dots, x_n\}$ are the observations, $\{t_1, \dots, t_n\}$ are the observed times, and $\{\delta^2, \dots, \delta_n^2\}$ are the errors in the measurements.

$$x_i^* = x_i - b\tau, \hat{x}_0 = 0,$$

$$\Omega_0 = \frac{\tau\sigma_C^2}{2}, \hat{x}_i = a_i\hat{x}_{i-1} + \frac{a_i\Omega_{i-1}}{\Omega_{i-1} + \delta_{i-1}^2}(x_{i-1}^* + \hat{x}_{i-1}),$$

$$\Omega_i = \Omega_0(1 - a_i^2) + a_i^2\Omega_{i-1} \left(1 - \frac{\Omega_{i-1}}{\Omega_{i-1} + \delta_{i-1}^2}\right),$$

and

$$a_i = e^{-(t_i - t_{i-1})/\tau}.$$

The final features are computed by maximising the likelihood with respect to τ and σ_C . b is calculated by taking the average light curve magnitudes divided by τ (Pichara et al. 2012). See table 3.1 for a list of features extracted from CAR.

- Stetson Features

The Stetson features are extracted from the Welch/Stetson variability index (Stetson 1996) defined as:

$$I = \sqrt{\frac{1}{n(n-1)}} \sum_{i=1}^n \left(\frac{b_i - \hat{b}}{\sigma_{b,i}} \right) \left(\frac{v_i - \hat{v}}{\sigma_{v,i}} \right), \quad (3.18)$$

where v_i and b are the magnitudes for an object observed in two different bands, and $\sigma_{v,i}$, and $\sigma_{b,i}$ are errors in the magnitude measurements. n is the number of observation pairs, \hat{v} , and \hat{b} are the mean in magnitude from each of the light curves in the two bands.

The number of observations from the two bands might be different, and their “relative error” can be computed as:

$$\delta = \sqrt{\frac{n}{n-1}} \frac{v - \hat{v}}{\sigma_v}. \quad (3.19)$$

StetsonK measure the kurtosis of the light curve. It calculated as:

$$K = \frac{1/N \sum_{i=1}^N |\delta_i|}{\sqrt{1/N \sum_{i=1}^N \delta_i^2}}, \quad (3.20)$$

and it is computed on a single light curve, i.e., it is computed on a single band. This means that it is not limited to objects with multiple bands (Richards et al. 2011).

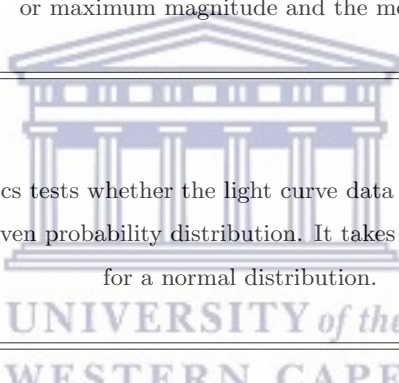
TABLE 3.1: Descriptions of features covered in this work. Note that **feets** has more features which are not covered here. We only cover those that we computed based on the input data we have. Their references can be found in the footnotes at the end of the table.

| Features | Description | Input Data |
|--|---|------------|
| Mean ⁶ | The average magnitude is calculated as: $\bar{m} = \frac{\sum_{i=1}^N m_i}{N}$, where m_i are the observed magnitudes and N is the number of observations. | Magnitude |
| Standard Deviation (std) ² | The standard deviation of the magnitudes is given by: $\sigma = \sqrt{\frac{1}{N} \sum_i (m_i - \bar{m})^2}$ | Magnitude |
| Mean Variance (Meanvariance) ⁴ | The mean variance is given by the ratio: std (σ)/ mean (\bar{m}). It is useful to measure the variability of an object (also known as the variability index) where it takes a large value for highly variable objects. | Magnitude |

| Continuation of Table 3.1 | | |
|--|--|--------------------|
| Feature | Description | Input Data |
| Amplitude ² | <p>The amplitude of magnitude observations, m, is calculated as follows:</p> $\text{amplitude} = 1/2 \{ \text{median}[\text{max}_{5\%}(m)] - \text{median}[\text{min}_{5\%}(m)] \}$ <p>where $\text{max}_{5\%}$ is the maximum 5% of the sample and $\text{min}_{5\%}$ is the minimum 5%. E.g., a sample with magnitude values from 0 to 500 will have an amplitude of 48.</p> | Magnitude |
| Median buffer range percentage (MedianBRP) ² | <p>Given an object with magnitude points, m, with a median, n, then MedianBRP is the percentage (≤ 1) of $m \leq \text{amplitude}/10$ from n.</p> | Magnitude |
| Range of cumulative sum (R_{cs}) ⁴ | <p>R_{cs} is defined by: $R_{cs} = \text{max}S - \text{min}S$,</p> <p>where $S = \frac{1}{N\sigma} \sum_{i=1}^l (m_i - \bar{m})$, and $l = 1, 2, 3, \dots, N$. It takes values ~ 0 for symmetric distributions.</p> | Magnitude |
| Eta_e (η^e) ⁶ | <p>This is derived from a variability index:</p> $\eta = \frac{1}{(N-1)\sigma^2} \sum_{i=1}^{N-1} (m_{i+1} - m_i)^2,$ <p>which is used to check if successive data points are independent or not. It, however, assumes an evenly sampled data which is rarely the case for astrophysical light curves. η^e accounts for this by modifying η as follows:</p> $\eta^e = \bar{\omega} (t_{N-1} - t_1)^2 \frac{\sum_{i=1}^{N-1} \omega_i (m_{i+1} - m_i)^2}{\sigma^2 \sum_{i=1}^{N-1} \omega_i},$ <p>where $\omega_i = \frac{1}{(t_{i+1} - t_i)^2}$, $\bar{\omega}$ is the mean values of ω_i and t is the measurement epoch.</p> | Time and magnitude |

| Continuation of Table 3.1 | | |
|---|--|--------------------|
| Feature | Description | Input Data |
| Lomb-Scargle features ^{4,6} | <p>The following features are derived from the Lomb Scargle periodogram algorithm:</p> <p>PeriodLS: Period estimated from the Lomb-Scargle periodogram</p> <p>Period_fit: the probability of the false-alarm maximum periodogram values. It is expected to take values close to zero for periodic light curves.</p> <p>Psi_CS (Ψ_{cs}): R_{cs} computed from the folded light curves.</p> <p>Psi_eta (Ψ_{η}): η^e applied to the folded light curves.</p> | Time and magnitude |
| Autocorrelation function length (Autocor_length) ⁴ | <p>Autocor_length⁴ is used to measure the similarities in observations as a function of lag time between them. It is used to find repeated patterns in a light curve. It is defined by:</p> $\hat{\rho}_h = \frac{\sum_{t=h+1}^T (m_t - \bar{m})(m_{t-h} - \bar{m})}{\sum_{t=1}^T (m_t - \bar{m})^2}$ <p>where \bar{m} is the mean of a sample with observations m_1, m_2, \dots, m_T and h is the lag time. When $\hat{\rho}_h$ is applied to a light curve, it returns a vector. However, only one value can be used as a feature. feets thus returns the lag value where $\hat{\rho}_h$ becomes $< e^{-1}$.</p> | Magnitude |

| Continuation of Table 3.1 | | |
|--|--|---------------------|
| Feature | Description | Input Data |
| Slotted Autocorrelation function length (SlottedA_length) ⁸ | <p>Unlike with Autocor_length where the time lag is a single value, the time lags are given as intervals in SlottedA_length. It is computed by taking the average cross product between observations that have a time difference that falls within a given interval.</p> <p>SlottedA_length is defined by:</p> $\hat{\rho}(\tau = kh) = \frac{1}{\hat{\rho}(0)N_\tau} \sum_{t_i} \sum_{t_j=t_i+(k-1/2)h}^{t_i+(k+1/2)/2} \bar{m}_i(t_i)\bar{m}_j(t_j),$ <p>where \bar{m} is the normalised magnitude, h is the interval size, $\hat{\rho}(0)$ is the SlottedA_length for the first lag, and N_τ is the number of pairs that fall in a given interval. Again feets returns the lag value where $\hat{\rho}(\tau = kh)$ becomes $< e^{-1}$.</p> | Time and Magnitude |
| Stetson features ² | <p>StetsonK is computed from equation 3.20 and it takes values close to 0.2 for a Gaussian distribution.</p> <p>StetsonK_AC: This is StetsonK applied to the slotted autocorrelation function of the time series.</p> | Magnitude and error |
| SmallKurtosis ² | <p>The small kurtosis is computed as:</p> $\kappa = \frac{N(N+1)}{(N-1)(N-2)(N-3)} \sum_{i=1}^N \left(\frac{m_i - \bar{m}}{\sigma} \right)^4 - \frac{3(N-1)^2}{(N-2)(N-3)}$ | Magnitude |
| Skewness (γ_1) ² | <p>The measures the skewness of the light curve and it is defined as:</p> $\gamma_1 = \frac{N}{(N-1)(N-2)} \sum_{i=1}^N \left(\frac{m_i - \bar{m}}{\sigma} \right)^3$ <p>The SmallKurtosis and Skewness should be zero for a normal distribution.</p> | Magnitude |

| Continuation of Table 3.1 | | |
|---|---|---------------------|
| Feature | Description | Input Data |
| Median absolute deviation (MeadianAbsDev) ² | <p>This is defined as:</p> $\text{MeadianAbsDev} = \text{median}(m - \text{median}(m)),$ <p>where m is the magnitude observations. For a normal distribution, it takes a value $\sim 0,675$.</p> | Magnitude |
| Percentage Amplitude PercentageAmplitude ² | The maximum percentage difference between either the minimum or maximum magnitude and the median. | Magnitude |
| Anderson-Darling test (AndersonDarling) ³ | <p>This statistics tests whether the light curve data has a characteristic of a given probability distribution. It takes values ~ 0.25 for a normal distribution.</p>  | Magnitude |
| Linear trend (Lineartrend) ² | This is slope of a linear function that is fitted on the light curve. | Time and magnitude |
| Maximum slope (MaxSlope) ² | This is the maximum slope computed from two successive observations, it calculated from the absolute magnitudes. | Time and magnitude |
| Beyond 1 std (Beyond1Std) ² | This is a fraction of magnitude observations that are beyond one std from the weighted mean , where std is the standard deviation. | Magnitude and error |

| Continuation of Table 3.1 | | |
|---|--|---------------------------|
| Feature | Description | Input Data |
| Pair slope trend (PairSlopeTrend) ² | This is computed by taking the ratio of the rising first differences to the declining first differences from the last 30 magnitude measurements (sorted by the time) | Magnitude |
| Q_{3-1} (Q31) ⁶ | The is computed as the third quartile (Q_3) minus the first quartile (Q_1) of the magnitude observations. | Magnitude |
| CAR ⁵ | Three features are extracted from CAR (see equation 3.16 and the descriptions that follow it): CAR.tau (τ), CAR.mean (mean), and CAR.sigma (variance). | Time, magnitude and error |
| Fourier Components ² | <p>Below is a list of features computed from the equations 3.13 and 3.15</p> <p>Freq3_harmonics_amplitude_0, Freq3_harmonics_amplitude_1, Freq3_harmonics_amplitude_2, Freq3_harmonics_amplitude_3, Freq2_harmonics_rel_phase_3, Freq2_harmonics_rel_phase_2, Freq2_harmonics_rel_phase_1, Freq2_harmonics_rel_phase_0, Freq1_harmonics_amplitude_2, Freq1_harmonics_amplitude_3, Freq1_harmonics_amplitude_0, Freq1_harmonics_amplitude_1, Freq3_harmonics_rel_phase_2, Freq3_harmonics_rel_phase_3, Freq3_harmonics_rel_phase_0, Freq3_harmonics_rel_phase_1, Freq2_harmonics_amplitude_1, Freq2_harmonics_amplitude_0, Freq2_harmonics_amplitude_3, Freq2_harmonics_amplitude_2, Freq1_harmonics_rel_phase_0, Freq1_harmonics_rel_phase_1, Freq1_harmonics_rel_phase_2, Freq1_harmonics_rel_phase_3</p> | Time and magnitude |

| Continuation of Table 3.1 | | |
|---------------------------|--|------------|
| Feature | Description | Input Data |
| Con ⁴ | This is computed by counting 3 consecutive magnitude observations that are fainter or brighter than two times the standard deviation of the observations. The number is then divided by N-2 to get the Con feature, where N is the total number of observations in the data. | Magnitude |

References: ¹Nun et al. 2015a; ²Richards et al. 2011; ³Kim et al. 2009; ⁴Kim et al. 2011; ⁵Pichara et al. 2012; ⁶Kim et al. 2014; ⁷Stetson 1996; ⁸Huijse et al. 2012

3.1.2.2 Feature extraction with avocado

Avocado is a python package aimed to classify different transient and variable events in the PLAsTiCC data. It was developed by Boone 2019 who applied it to the data and won the PLAsTiCC astronomical classification kaggle challenge³ (see details about the challenge in Chapter 5). Their approach of solving the classification challenge involved the basic machine learning approaches described in Chapter 2. We are, however, interested in their data pre-processing and features extraction techniques.

They pre-processes the light curves using Gaussian processes (GP; Rasmussen and Williams 2006). A GP is built on the assumption that the data is drawn from a random sample with a probability function characterised by Gaussian noise. The joint probability distribution (Rasmussen and Williams 2006):

$$\begin{bmatrix} y \\ f^* \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \mu \\ \mu^* \end{bmatrix}, \begin{bmatrix} K(X, X) + C & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix} \right), \quad (3.21)$$

characterises the reconstruction of function $f^*(X^*)$ from an input data X defined by a probability distribution $y(X)$, with a covariance matrix C . The symbols μ and μ^* are the mean of the of f^* and y , given as initial guesses when fitting the GP, and K is the GP kernel which is also chosen beforehand.

³<https://www.kaggle.com/c/PLAsTiCC-2018>

The reconstructed function f^* is defined by the mean, covariance (cov), and marginal log likelihood ($\ln \mathcal{L}$) computed as:

$$\text{mean}(f^*) = \mu^* + K(X^*, X)[K(X, X) + C]^{-1}(y - \mu), \quad (3.22)$$

$$\text{cov}(f^*) = K(X^*, X^*) - K(X^*, X)[K(X, X) + C]^{-1}K(X, X^*), \quad (3.23)$$

$$\ln \mathcal{L} = -\frac{1}{2}(y - \mu)^T [K(X, X) + C]^{-1}(y - \mu) - \frac{1}{2} \ln |K(X, X) + C| - \frac{n}{2} \ln 2\pi. \quad (3.24)$$

where n is total number of observations in the data. **Avocado** makes use of the kernel function $K_{3/2}$ defined by:

$$K_{3/2}(x_1, x_2; \alpha, l) = \alpha^2 \left(1 + \sqrt{3 \frac{(x_1 - x_2)^2}{l^2}} \right) \exp \left(-\sqrt{3 \frac{(x_1 - x_2)^2}{l^2}} \right), \quad (3.25)$$

where α , l , x_1 , and x_2 are the amplitude scale, length scale, and a set of points in the observations, respectively. Now, to model the light curves in both time and wavelengths, they used the two-dimensional kernel by taking the product of the equation 3.25 in both time and wavelength space:

$$K_{2D}(t_1, t_2, \lambda_1, \lambda_2; \alpha, l_t, l_\lambda) = \alpha^2 K_{3/2}(t_1, t_2; 1, l_t) K_{3/2}(\lambda_1, \lambda_2; 1, l_\lambda), \quad (3.26)$$

which takes the amplitude (α), length scale in both wavelength (l_λ) and time (l_t) as hyperparameters. The length scale in wavelength was kept fixed ($l_\lambda = 6000\text{\AA}$), the time scale length (l_t) and amplitude (α) were fitted using the **George** package (Ambikasaran et al. 2016)). Table 3.2 shows features computed by **avocado**, some of which are derived from the GP described above.

We extracted 41 features from the PLAsTiCC data using **avocado**. We extracted the features outside the framework and saved them as a **csv** file with an $N \times M$ matrix, where N is the objects in the data and M is their corresponding features. We then used **PANDAS** to read in the **csv** file during the anomaly detection step (see section 3.1.4) in **ASTRONOMALY**.

Even though table 3.2 shows the list of all features, some of the features were neglected in this work because `avocado` returns invalid values for them. A list of the neglected features is outlined in the footnote at the bottom of table 3.2.

TABLE 3.2: List of features computed from the `avocado` code.

| Features | Description |
|---|---|
| <code>host_photoz</code> | This is the photometric redshift of the galaxy at which the event occurs, it is given in the metadata. |
| <code>host_photoz_err</code> | This is the error in the measurement of the <code>host_photoz</code> , it is also given in the metadata. |
| <code>length_scale</code> | This is the fitted l_t from the GP described above, where l_t is the time length scale, it takes units of days. |
| <code>max_mag</code> | This is the peak magnitude of the predicted GP flux. It measured in the i passband of the data. |
| <code>pos_flux_ratio</code> | The ratio of the positive maximum flux to the difference between the maximum flux and minimum flux. It is computed on the i band observations. |
| <code>[max,min]_flux_ratio</code> <code>_[blue,red]</code> | This is the normalised difference of an object's colour, calculated from its maximum/minimum light curve flux. The red measurements is computed by taking the difference between the y and i bands; and the blue measurements are computed as the difference between i and g bands. The final normalised difference is thus computed by subtracting the fluxes in the two bands and dividing by their sum |
| <code>max_dt</code> | This is calculated by taking the difference between the peak time in y and g bands, where peak time refers to the time at maximum flux. |
| <code>[positive,negative]</code> <code>_width</code> | This is computed by taking the integral of the positive/negative parts of the predicted GP fluxes and dividing by the maximum positive/negative fluxes. It is used to estimate "width" of the light curves. |
| <code>time-[fwd,bwd]_max</code> <code>_[0.2,0.5]</code> | This is the time it takes for the light curve to decline (<code>bwd</code>) or rise (<code>fwd</code>) to a specified percentage (either 20% or 50%) of the maximum flux in i band. |

| Continuation of Table 3.2 | |
|--|---|
| Feature | Description |
| <code>time_[fwd,bwd]_max</code> <code>_[0.2,0.5]_ratio</code> <code>_[blue,red]</code> | This is computed as the ratio of the rise time to the decline time, similar to those described above, but computed in different bands. The red and blue colours are also similar to those described above. |
| <code>frac_s2n_[5,-5]</code> | This is the fraction of observations with a signal that is less than -5 or greater than 5 times the level of noise in the observations. |
| <code>frac_background</code> | Fraction of observations with an absolute signal-to-noise that is less than 3. |
| <code>time_width_s2n_5</code> | This is computed based on observations (in any band) with signal-to-noise ratio that is greater five. It is calculated by taking the time difference between the first observation and the last observation. |
| <code>count_max_center</code> | This counts observations falling within 5 days of the maximum flux (the observations can be in any band). |
| <code>count_max_rise</code> <code>_[20,50,100]</code> | This counts observations between twenty, fifty, or hundred days before maximum flux and five days after maximum flux (observations also measured in any band). |
| <code>count_max_fall</code> <code>_[20,50,100]</code> | This counts observations between five days before maximum flux and twenty, fifty, or hundred days after maximum flux (observations can be in any band). |
| <code>peak_frac_2</code> | This is the ratio of the second maximum peak flux to the original maximum peak flux in the observations per band, averaged over all bands. It used to separate SN-like light curves from others since they mostly have a single peak |
| <code>total_s2n</code> | Total signal-to-noise taken from all light curves (in all bands) of the object. |
| <code>percentile_diff</code> <code>_[10,30,70,90]_50</code> | This measures the flux distribution of observations. It is computed by taking the flux level in each band at a specified percentile and normalising it with the maximum flux minus the minimum flux from the flux predicted by the GP. The final value is calculated by taking the median (in all bands) of the difference between the normalised flux at any given percentile and the flux at the 50th percentile. |

Note: The following features were neglected in this work because of reasons described above: `peak_frac_2`,

d_max_0.2_ratio_blue, time_bwd_max_0.2_ratio_red, time_bwd_max_0.2, time_bwd_max_0.5_ratio_blue, time_fwd_max_0.5_ratio_red, time_fwd_max_0.5, time_fwd_max_0.2_ratio_red, time_fwd_max_0.2, time_bwd_max_0.5_ratio_red, time_bwd_max_0.5, time_fwd_max_0.5_ratio_blue, and time_fwd_max_0.2_ratio_blue.

3.1.3 Data post-processing

ASTRONOMALY has additional post-processing techniques after the feature extraction process. This includes employing techniques such as feature scaling and the principal component analysis (Pearson 1901; Hotelling 1933) for dimensionality reduction. We only implemented the feature scaling technique in this work.

The feature scaling technique incorporated in ASTRONOMALY works by standardising each column in feature space such that they have a mean of zero and standard deviation of one:

$$Y = \frac{x - \text{mean}(X)}{\text{std}(X)}, \quad (3.27)$$

where Y is the new column with standardised features, std is the standard deviation, x is the feature value for a single object, and X is the total sample for a given feature column.

This is crucial for most machine learning algorithms (LOF for our case) as they tend to be biased to features with large values. For example, the `percentile_diff` feature returns values between 0 and 1, while the `mean` returns values close to 1000, the algorithm will then assume that the `mean` is greater than `percentile_diff`. This means that the algorithms assume that features with large values are more important than those with small values (Surbhi 2019).

3.1.4 Anomaly Detection

The anomaly detection algorithms are employed to assign anomaly scores to individual objects in a given dataset based on their extracted features. A wide variety of AD algorithms can be used, both in supervised and unsupervised learning problems. The challenge with the former is that it operates with labeled data which can be expensive to acquire in practice. Currently, two unsupervised AD algorithms are incorporated in ASTRONOMALY: iForest and LOF, both of which were used in this work.

We have described in section 2.2.2 the iForest and LOF algorithms, both of which return anomaly scores of objects in a given dataset. However, the range of the anomaly scores differ for each algorithm, e.g., iForest does not take scores greater than one while LOF does. **ASTRONOMALY** normalises the raw anomaly scores from the algorithms such that they both have a standard scale. The anomaly scores from the algorithms are standardised such that they both have anomaly scores in the range from zero to five. The former and latter scores correspond to the sample's least anomalous and most anomalous objects, respectively.

3.1.5 The Frontend Web Interface

ASTRONOMALY has a frontend web interface that has two tabs: the `anomaly scoring` and `visualisation` tab.

Visualisation Tab

The visualisation tab incorporates the t-SNE technique to reduce the dimensions of a given high dimensional features space to two-dimensional feature space for visualisation purposes. t-SNE works by computing the probability of the similarities between point pairs in a high dimensional features space using Euclidean distances. The computed probability distribution is then used to represent points that are similar in the high dimensional space to lie close to each other in the lower dimensional space, two or three dimensions (Van der Maaten and Hinton 2008).

t-SNE plots are useful in finding underlying structures that may be present in high dimensional feature space. This includes finding clusters of objects and outliers in the given feature space. Figure 3.2 shows an example t-SNE plot of features computed for the CRTS data (we set `perplexity = 100` `max_sample = 2000` and `shuffle = False`). We do not focus on t-SNE plot analysis in this thesis; rather, we focus only on the anomaly detection algorithms and active learning techniques incorporated in **ASTRONOMALY**.

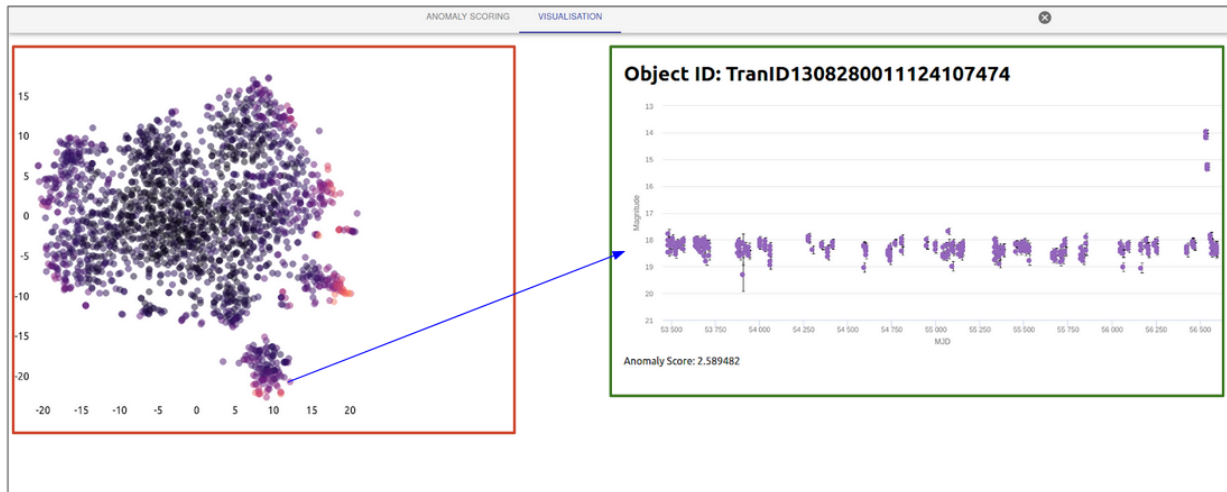


FIGURE 3.2: A screenshot of the **ASTRONOMALY** frontend visualisation tab. Highlighted in red is the t-SNE plot computed on CRTS data. Note that both the x- and y-axis are in arbitrary units. The points on the t-SNE plot are colour coded with the raw anomaly scores (darker points indicate the least anomalous objects and vice versa). The visualisation tab is interactive, one can click on any point on the t-SNE plot, and the original data (a light curve for our case) will be plotted on the right (see the plot highlighted in green). The blue arrow indicates the point clicked on the t-SNE plot and its corresponding light curve on the right.

Anomaly Scoring Tab

By default, the objects are presented on the anomaly scoring tab in a ranked manner according to the standardised anomaly score described in section 3.1.4. The most anomalous objects appear at the top ranks and vice versa. **ASTRONOMALY** has other options at which the objects can be sorted in the anomaly scoring tab. This includes sorting the object randomly or by the human retrained scores.

The anomaly scoring tab is interactive such that a user can assign scores to objects according to how interesting they are to them. We refer to these scores as the “relevance” scores. They also take values from zero to five, where five indicate that the object at hand is interesting to the user, three indicate moderate, and zero indicate boring objects (see Figure 3.3). The user can then retrain **ASTRONOMALY** to get the human retrained scores. These scores are obtained from active learning, which will be described in the next subsection.

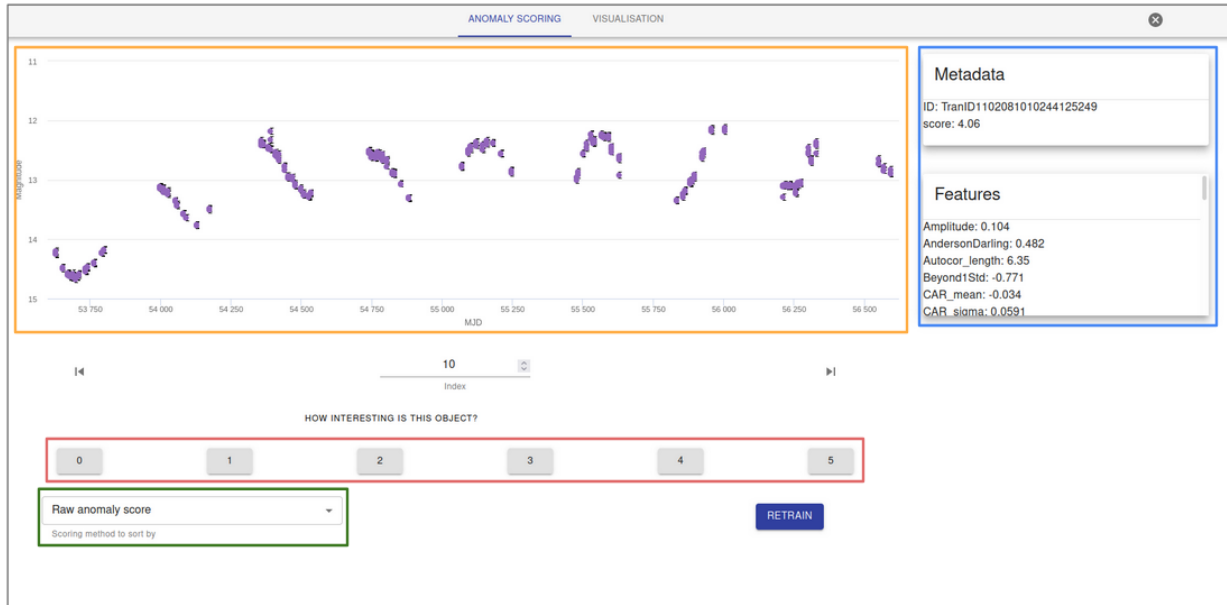


FIGURE 3.3: A screenshot of the ASTRONOMY frontend anomaly scoring tab. Highlighted in orange is the example light curve from the CRTS data. To the right (highlighted in blue) is additional information about the object (e.g., features computed and metadata). The numbers from 0 to 5 in gray boxes (highlighted in red) are used to assign the relevance scores. The drop box at the bottom left (highlighted in green) prompts the user to choose how they want the objects to be sorted.

3.1.6 Active Learning

The raw anomaly scores S from the machine learning algorithm and the relevance scores U can be used to help ASTRONOMY to personalise interesting anomalies. This is done by ranking objects on the anomaly scoring tab (Figure 3.3) according to score calculated as:

$$\hat{S} = S \tanh(\delta - 1 + \operatorname{arctanh}(\tilde{U})), \quad (3.28)$$

where \hat{S} is the new active learning score, δ is a distance penalty term, which takes a large value for cases where S is not well defined, and \tilde{U} is the normalised relevance score calculated as:

$$\tilde{U} = \epsilon_1 + \epsilon_2 \left(\frac{U}{U_{max}} \right) \quad (3.29)$$

where U_{max} is the maximum possible user score, which is five for ASTRONOMY. The normalisation constants are set to: $\epsilon_1 = 0.1$ and $\epsilon_2 = 0.85$. We use arctan because it is a smoothly varying function with the desired asymptotic properties and other similar functions will likely perform equally well.

The constant terms ϵ_1 and ϵ_2 were chosen to give stable numerics for the full range of possible user scores, since $\operatorname{arctanh}$ diverges for arguments of unity.

Predicting the Relevance Score

The user can provide a relevance score U for a limited number of objects in the sample. However, for **ASTRONOMALY** to successfully run, all objects in the sample need to have a relevance score. This means that equations 3.28 and 3.29 have two unknown parameters that needs to be predicted and calculated: the remaining relevance scores U , and the distance penalty term δ .

ASTRONOMALY incorporates the random forest regression algorithm to estimate U , where the hyperparameter `n_estimators` is set to 100. The uncertainties in the regression estimates are quantified with the distance penalty term δ defined by:

$$\delta = \exp\left(\alpha \frac{d}{d_0}\right), \quad (3.30)$$

where d is the Euclidean distance between the object at hand in feature space and its nearest labeled relevance score provided by a human user. α and d_0 are the tuning parameter and mean distance to a human-labeled neighbor, respectively. In this work, we adopted the default setting of **ASTRONOMALY**, where d was computed using `KDTree` (Maneewongvatana and Mount 2002) and $\alpha = 1$ (Lochner and Bassett 2020).

3.1.7 Analysing the Performance of Unsupervised Anomaly Detection Algorithms

Just like with supervised learning algorithms, we can analyze the performance of anomaly detection algorithms. We described in section 2.1.3 some of the metrics used to assess the performance of supervised learning algorithms. However, these metrics assume a known target label. The case is different with unsupervised anomaly detection, particularly for **ASTRONOMALY**, since it is built on the assumption that anomalous classes are unknown in the sample (Lochner and Bassett 2020). However, for the case of simulated data, e.g., **PLAsTiCC** for our case, we can evaluate the performance of **ASTRONOMALY** since we know a sample of anomalies present in the dataset.

This is done by taking advantage of how **ASTRONOMALY** arranges the objects in the datasets from the most anomalous to the least anomalous as by the raw anomaly score. Of course, the objects can also

be rearranged according to the computed active learning scores (see equation 3.28). This means that we can use different metrics to evaluate the ranks at which the known anomalies in the dataset appear in `ASTRONOMALY anomaly scoring` tab. In this work, we adopted two metrics used by Lochner and Bassett 2020:

- **Recall:** Given a set of N objects a user views, recall counts the number of anomalies retrieved after viewing these objects. For example, if the user views one hundred objects and only two anomalies are retrieved, then the recall is two. This is an important metric because, ideally, we want anomalies to be at the top of the ranked list so that a human expert can quickly analyse them without going through the entire dataset.
- **Rank Weighted Score:** Given a dataset with a known number of anomalies, N , the Rank Weighted Score (RWS; Roberts, Bassett, and Lochner 2019) is computed as:

$$S_{\text{RWS}} = \frac{1}{S_0} \sum_{i=1}^N w_i I_i, \quad (3.31)$$

where $w_i = (N + 1 - i)$ are the weights, $S_0 = N(N + 1)/2$ and I_i is variable used to indicate if object i is an anomaly or not. $I_i = 0$ if object i is not an anomaly, and one otherwise. S_{RWS} takes values between zero and one. It returns zero when zero anomalies are retrieved after viewing N objects and one when all objects in the top N are anomalies.

Even though the description above is based on N corresponding to the number of anomalies in the dataset, it can also be set to a reasonable number that a human expert would want to look at. In this work, we set the limit of N to be 1000.

3.2 Follow-up Studies on Detected Anomalies

Anomaly detection techniques are ideal tools for making new discoveries in the current and upcoming era of big data. However, most of the anomalies that will be detected might be known objects, and only a small fraction of them might be new discoveries. Follow-up studies are required to check if the detected anomalies are already cataloged in the literature or not. They are also needed to check whether the anomalous behavior of the objects is not due to artefacts.

In this work, we conducted follow-up studies on the top twelve anomalies detected in the CRTS data using the Caltech⁴ and SIMBAD⁵ platforms. SIMBAD is an online astronomical database platform that has basic information about astronomical objects. This information includes the magnitude, coordinates, proper motions, and identifiers. The latter is important as it provides other names that identify the objects in other surveys.

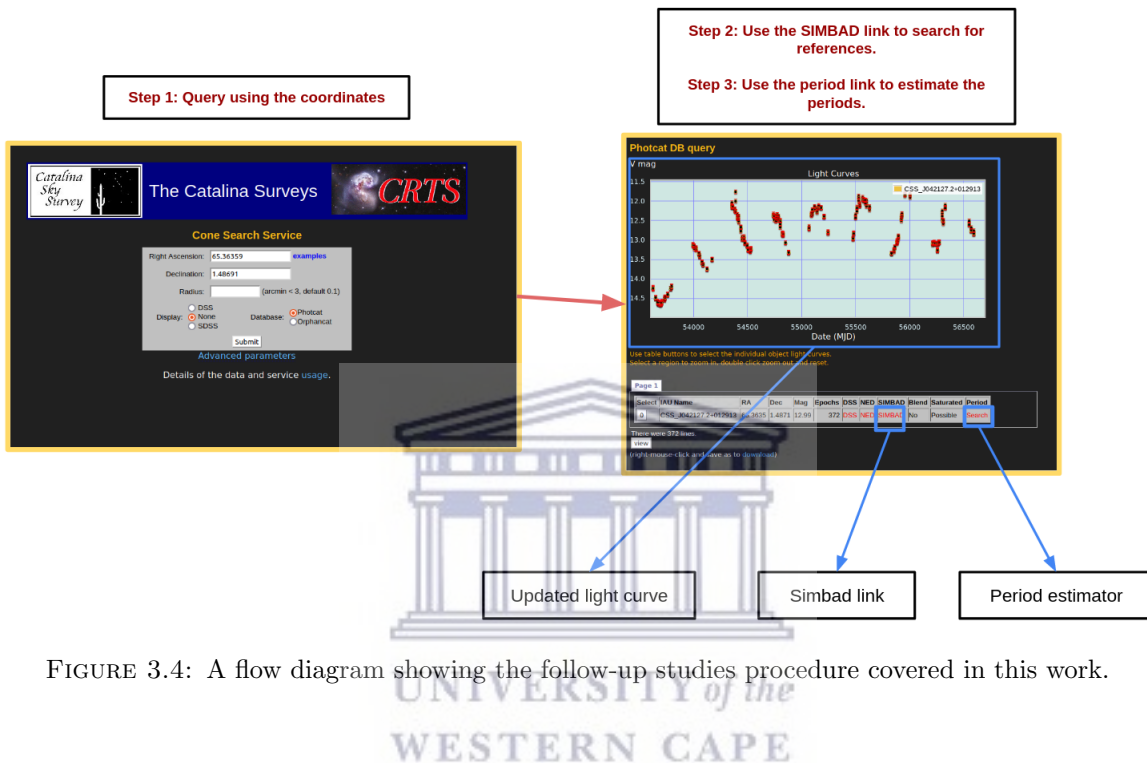


FIGURE 3.4: A flow diagram showing the follow-up studies procedure covered in this work.

Caltech is an online database platform that currently has updated light curves for objects in the CRTS. It also has links to other database platforms such as SIMBAD and a link that is used for period estimation. Figure 3.4 shows the steps we follow in making follow-up studies on anomalies detected from the CRTS data. Step 1 includes querying the Caltech database using the coordinates of the objects. We then compare the updated light curves with those detected from MANTRA data. We finally click on the SIMBAD link to get additional information about the objects, including the classification of the object and other identifiers.

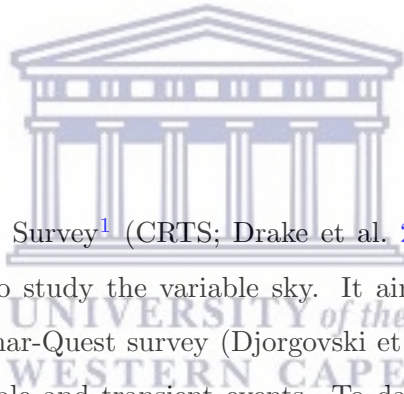
We describe in Chapters 4 and 5 the application of ASTRONOMALY to CRTS and PLAsTiCC data respectively. We also discuss the properties of some of the detected anomalies from the CRTS data after we conducted follow-up studies. Lastly, we assess the performance of ASTRONOMALY using anomalies detected in the PLAsTiCC data.

⁴http://nunuku.caltech.edu/cgi-bin/getcssconedb_release_img.cgi

⁵<http://simbad.u-strasbg.fr/simbad/>

Chapter 4

Application of Astronomy on the Catalina Real-Time Transient Survey data



The Catalina Real-Time Transient Survey¹ (CRTS; Drake et al. 2008; Djorgovski et al. 2011) is a synoptic survey that is designed to study the variable sky. It aims to continue studies conducted by early surveys such as the Palomar-Quest survey (Djorgovski et al. 2008), and its main goal is to discover interesting and rare variable and transient events. To date, it has discovered ~ 4190 SNe events, and ~ 1513 cataclysmic variable stars to date. It operates on three surveys from the Catalina sky survey, the: 1.5-meter Mount Lemmon survey (MLS), 0.7-meter Catalina sky survey (CSS), and 0.5-meter siding springs survey (SSS).

The MLS and CSS are both located in Tucson, Arizona, and SSS is located in Australia. They all operate on a 4×4 CCD camera, with a coverage of 4, 1.1, and 8.2 square degrees for the SSS, MLS, and CSS, respectively. In total, they can cover an area of ~ 33000 square degrees. They, however, exclude the Galactic latitude $b < 10 - 15$ degrees to avoid the overcrowded stellar region on the Galactic plane.

The three telescopes make observations for 23 nights per lunation, with an exposure time of 30 seconds. They take four images of the same region during the observing period, each taken after ten minutes. They observe in the V band with a magnitude limit of $V \sim 19 - 20$ mag. The light curves are

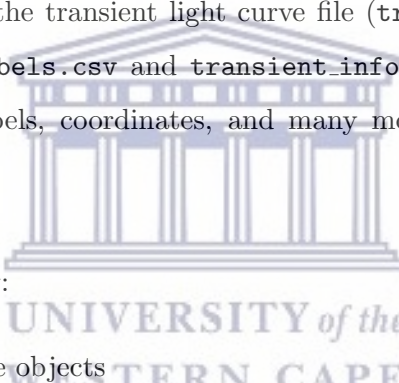
¹<http://crts.caltech.edu/>

constructed using aperture photometry through a package called **SEXTRACTOR** (Bertin and Arnouts 1996).

4.1 The Data

The data used in this work is from one of the CRTS surveys, the CSS. The survey covers $\sim 4000 \text{ deg}^2$ of the sky per night, with a magnitude limit of $V \sim 19.5 \text{ mag}$. We used the MANTRA (Neira et al. 2020) data that is composed of 4869 transient light curves (see example light curves in Figure 4.1) and 71207 non-transient light curves. Both the transient light curves and non-transient light were stored in a repository that is found at: <https://github.com/MachineLearningUniandes/MANTRA>, under the `data/lightcurves` folder.

In this work, we are interested in the transient light curve file (`transient_lightcurves.csv`), and its supporting files (`transient_labels.csv` and `transient_info.txt`). The supporting files have information about the object's labels, coordinates, and many more. The files have the following columns:

- 
- `transient_lightcurves.csv`:
 - ID: The unique ID of the objects
 - `observation_id`: This is the order at which the objects were observed
 - ID: The unique ID of the objects
 - Mag: The magnitude measurement
 - `Magerr`: The error in the magnitude measurements
 - MJD: Observing time in modified Julian dates
 - `transient_labels.csv`:
 - `TransientID`: The unique ID of the objects
 - `Classification`: Their labels
 - `transient_info.txt`:
 - `#CRTS ID`: The object ID as by CSS catalog
 - RA (J2000): Right Ascension in degrees

– Dec (J2000): Declination in degrees

Note: The `transient_info.txt` has more columns, however, in this, we only utilized those described above. The three files can be linked to each other through the object’s unique IDs.

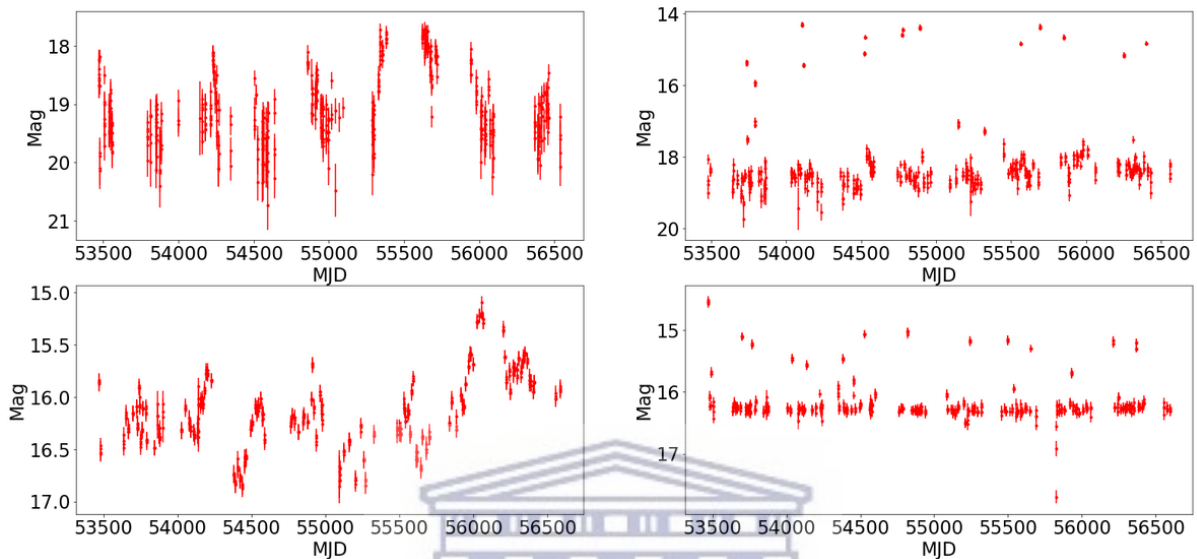


FIGURE 4.1: Example light curve from the MANTRA dataset.

Even though the objects in the data are labeled, some of them do not have a clear class. These are identified with the special characters: ‘/’, ‘?’ and ‘Unclear’. The ‘?’ character indicates events with unclear classes (e.g., CV?) and ‘/’ indicates cases where a single event has multiple classes (e.g., Var/Nova), see table A.1 for a complete list of these objects. Objects with clear labels are those whose labels were confirmed with spectroscopic and photometric follow-up studies².

Figure 4.2 shows a list of clear labels present in the dataset and the corresponding number of objects in each of the labels. The dataset is dominated by supernovae, cataclysmic variables, high proper motion stars, and active galactic nuclear events.

²<http://crts.caltech.edu/Followup.html>

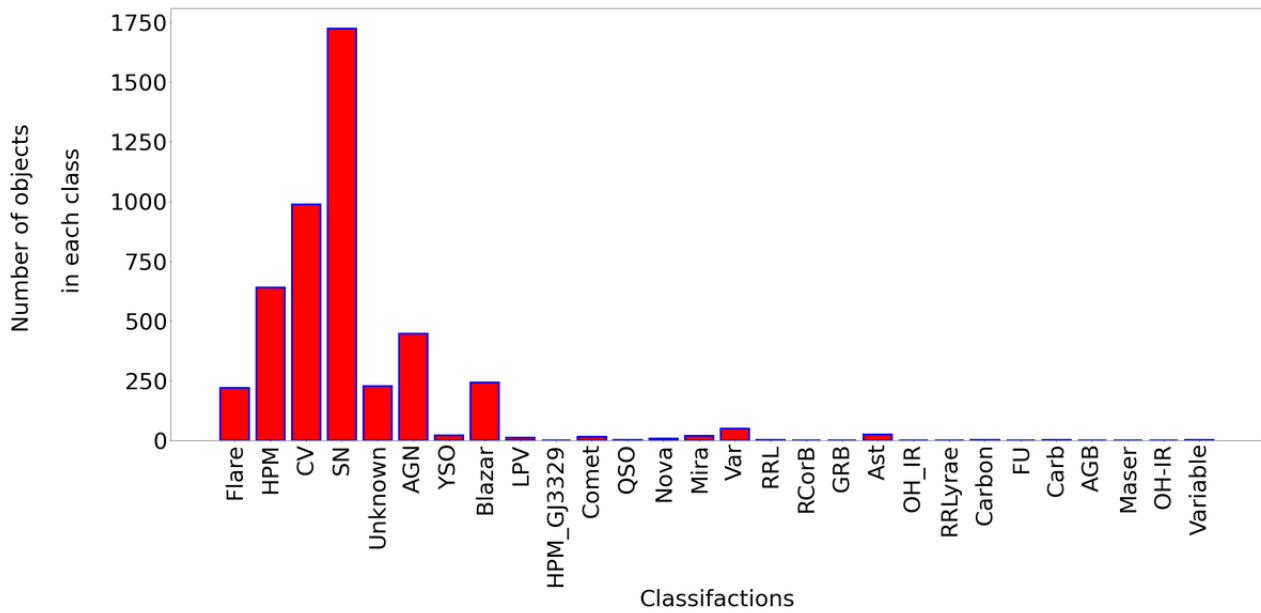


FIGURE 4.2: A bar graph showing a list of labels present in the MANTRA data before applying the cuts described in section 4.2 but only including objects in the dataset with clear labels. It is important to note that the list is not complete, as some of the objects do not have clear labels (see table A.1).

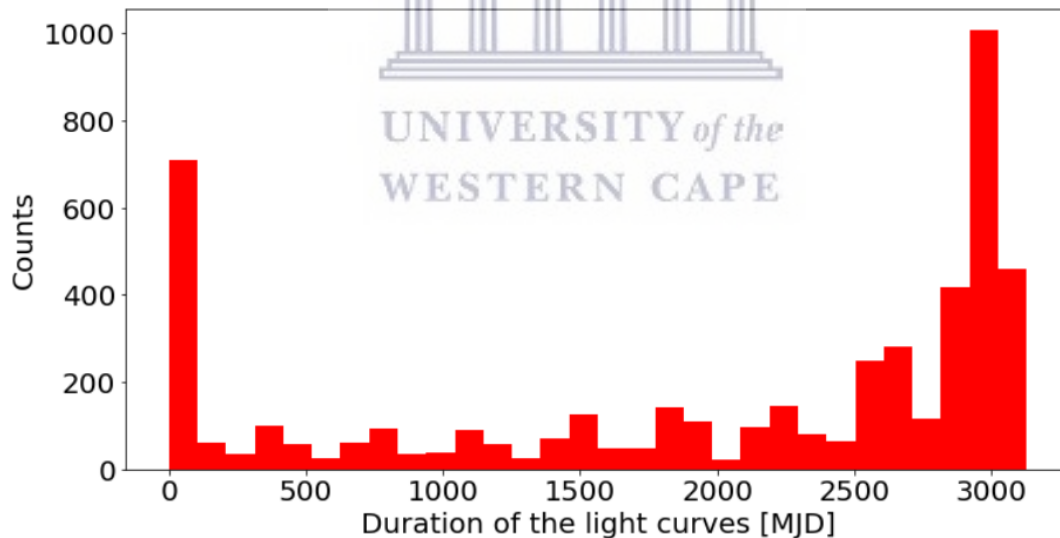


FIGURE 4.3: Histogram showing the light curve duration of objects in the MANTRA data in units of MJD.

Figure 4.3 shows the light curve duration of objects in the dataset. 176 objects have a light curve duration of zero, which means they only have a single point in their light curves. These objects were removed from the dataset during the data pre-processing step discussed in the next section. The longest light curve duration in the dataset is 3130.29 MJD.

In this work, we used all the 4869 light curves from the light curve data as an input to **ASTRONOMALY**. We then pre-processed the light curves, extracted features, ran an anomaly detection algorithm, and made follow-up studies on the detected anomalies.

4.2 An Overview of the Methodology

The MANTRA data have light curves of multiple objects embedded in one file with one V band. This means that this data is treated as **case_1a** as described in section 3.1.1. We used **ASTRONOMALY** to read in the data, extract features from the light curves and assign an anomaly score to each object using iForest algorithm. 4,869 light curves were used as input to **ASTRONOMALY**, and we only considered objects with ≥ 20 points in their light curves because we found that some features in **feets** require a minimum of 20 points per light curve to be successfully computed. Only 2506 objects remained after this pre-processing step, and features were extracted only on these objects.

4.2.1 Feature Extraction

We extracted 56 features using **feets**, see a full list of feature in table 3.1. We, however, excluded the **period_fit** feature as it fails to return a value for non-periodic light curves. Figure 4.4 and 4.5 shows the distribution of features extracted from the most abundant classes in the data. These are used to get an insight into how the features would succeed in separating the different classes.

We can see from the top left plot of Figure 4.4, that the **Amplitude** values differ for each of the classes. This can also be seen on the **Beyond1Std** (right plot on the middle panel), and **Mean** (bottom right plot) features. However, not much difference is observed with the **AndersonDarling** and **CAR_sigma** features.

Figure 4.5 shows scatter plots of the features. These plots give insights into the relationship between any random two features. The majority of the objects occupy the same region (see plots **b**, **e**, and **f**) with few outliers. There is also a clear, distinct difference observed between the objects using plot **d**. This means that the **Amplitude** and **Mean** features are the best features to separate the different classes. Also notice the correlation metrics on Figure 4.6, which shows that on average, the computed features are not highly correlated.

Traditionally, less important features are normally discarded from the data to improve the performance of machine learning algorithms. This is done mainly in classification problems. However, we are

interested in anomaly detection algorithms, and we opt to keep all the computed features and neglect only those that do not return values. Our final sample has 55 features.

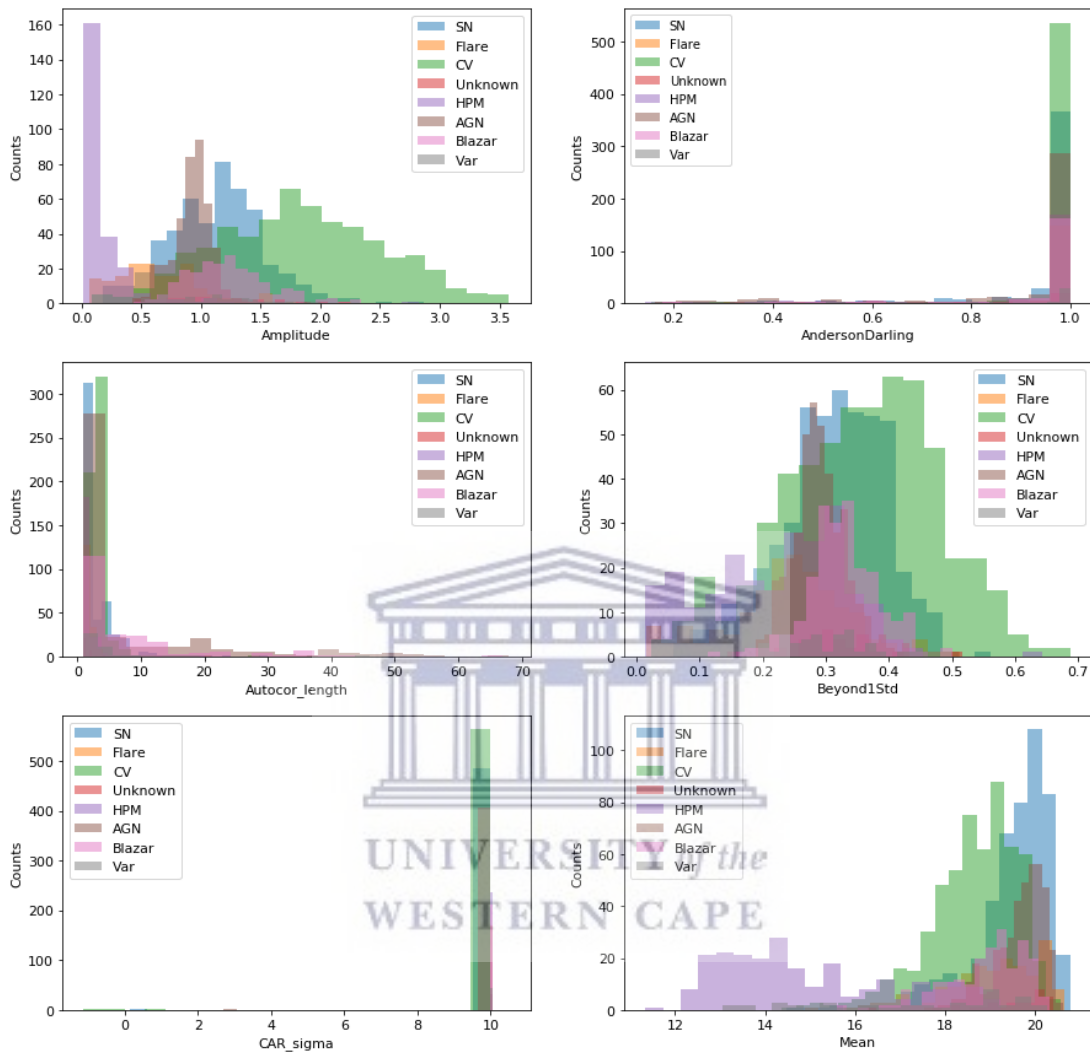


FIGURE 4.4: Histogram showing the distribution of some of the computed features for different classes. The “unknown” class corresponds to objects that are not catalogued in the literature, i.e., possible new discoveries.

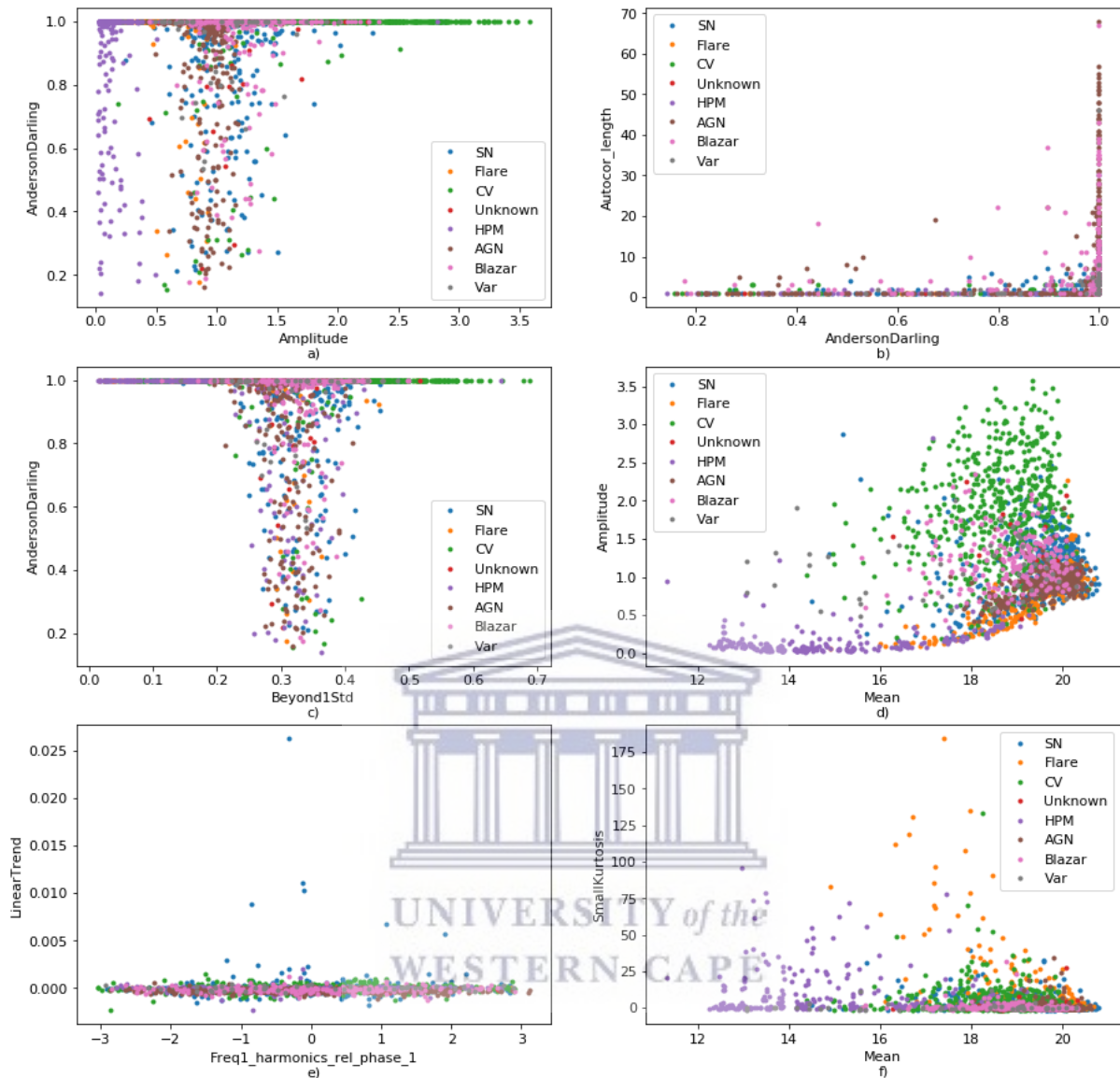


FIGURE 4.5: Scatter plot of some of the calculated features. Note that all the plots have the same legend.

4.2.2 Anomaly Detection and Active Learning

We employed iForest to compute anomaly scores for each of the 2506 objects, each represented by 55 features described above. We use the default hyperparameters from the `scikit-learn` implementation³. ASTRONOMALY then displayed the objects in a ranked manner from the most anomalous to the least anomalous on the anomaly scoring tab of the web interface. We assign three new classes to each of the top 100 light curves based on the following characteristics:

³<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html>

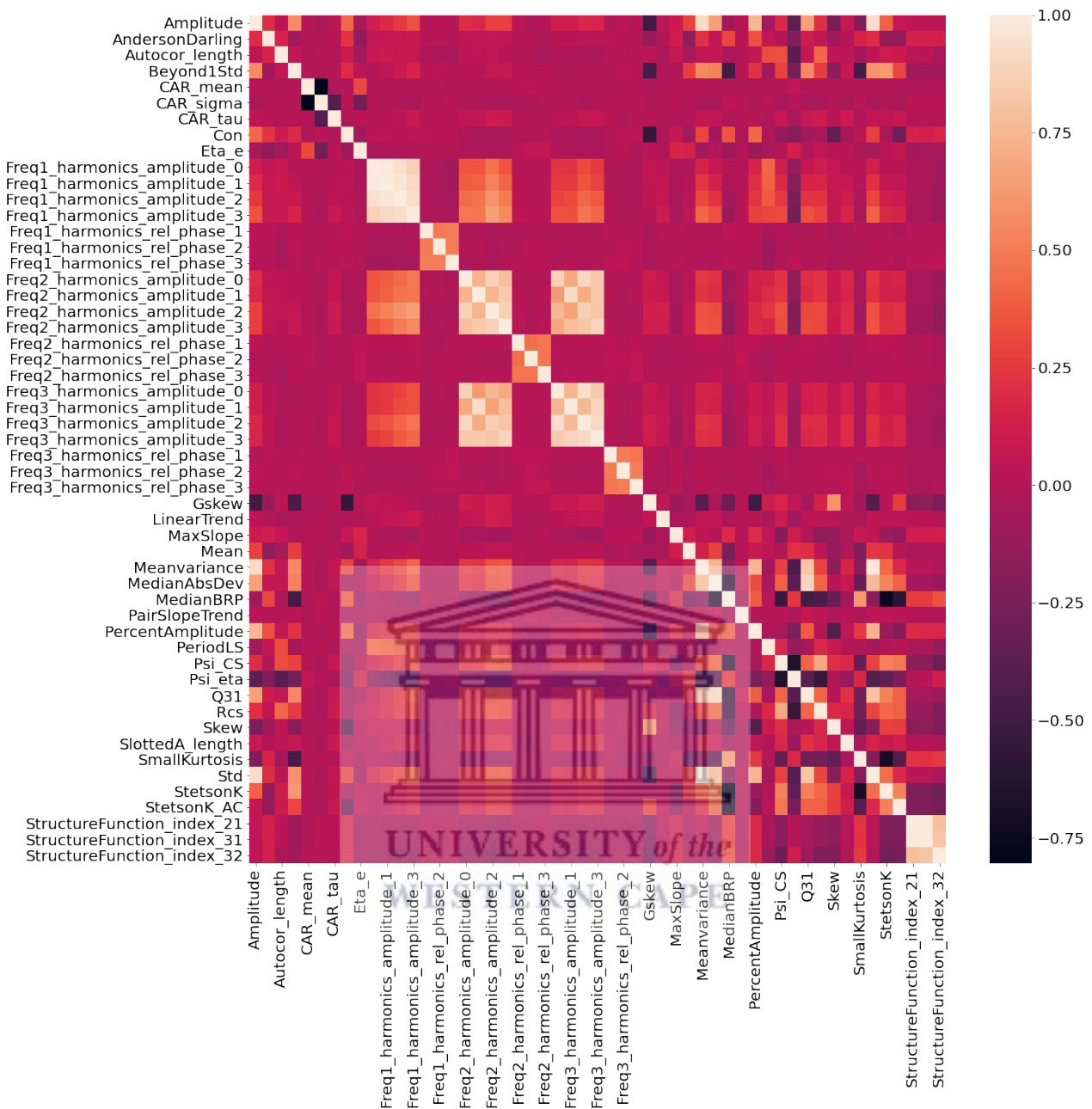


FIGURE 4.6: Correlation metric of the computed features from `feets`. Notice that there are few features that are highly correlated (mostly the frequency features). However, on average, the features are not highly correlated.

- **Bogus:** These are light curves with few data points or light curves with large gaps between the observations (gaps > 600 MJD).
- **Normal:** These are light curves that display normal shapes.
- **Anomaly:** These are light curves that display unusual interesting shapes.

The bogus, normal and anomaly light curves were assigned a relevance score of zero, two and five respectively. We then retrained `ASTRONOMALY` through active learning to predict the relevance scores of the remaining light curves. Lastly, we make follow-up studies on the top 12 anomalies that are returned after active learning.

4.3 Results

Figure 4.7 shows a violin plot of the raw anomaly scores obtained from iForest. Even though iForest is expected to return an anomaly score ~ 1 for the most anomalous objects as described in subsection 2.2.2.1, the case is different for the `scikit-learn` implementation. The most anomalous objects are assigned a low anomaly score (negative scores) and the least anomalous a high score (positive scores). The violin plot shows the anomaly score distribution in each input class plotted in a ranked manner from the least abundant class to the most abundant class in the dataset.

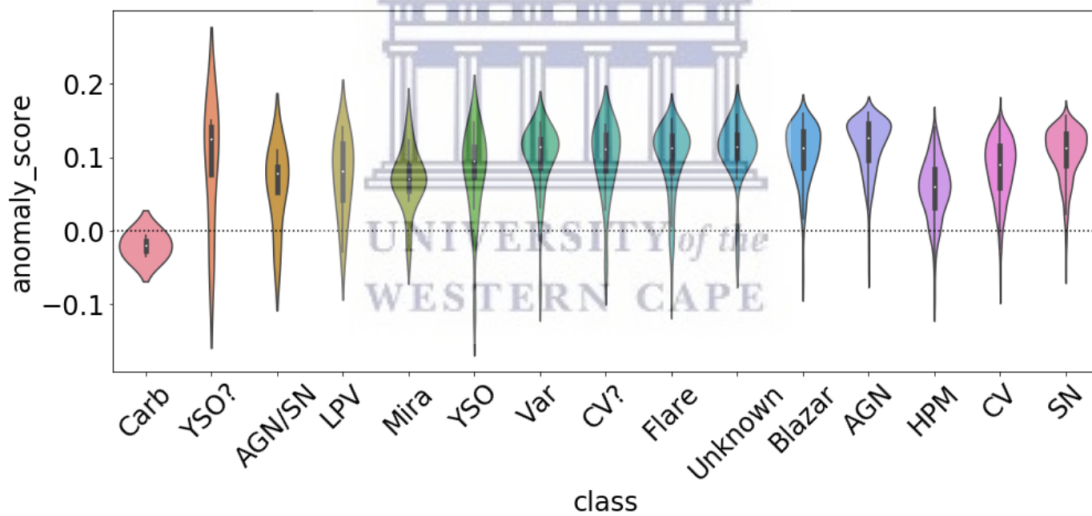


FIGURE 4.7: A violin plot of the computed anomaly scores from iForest for every input class in the MANTRA data. On the y-axis is the anomaly score, and on the x-axis are the classes. The black dotted horizontal line divides the y-axis such that the negative scores are below the line and the positive scores are above the line. Note that the classes are arranged on the x-axis from the least abundant class in the data to the most abundant class. One can easily spot outliers from the elongated tails of the violin.

It is expected that the least abundant classes in the dataset should have negative scores and vice versa. We, however, find from Figure 4.7 that each class in the dataset has both positive and negative anomaly scores. This implies that iForest detects outliers in every class independent of the number of objects present in each class. Note that this analysis was conducted on the raw anomaly scores before they were standardised by `ASTRONOMALY`.

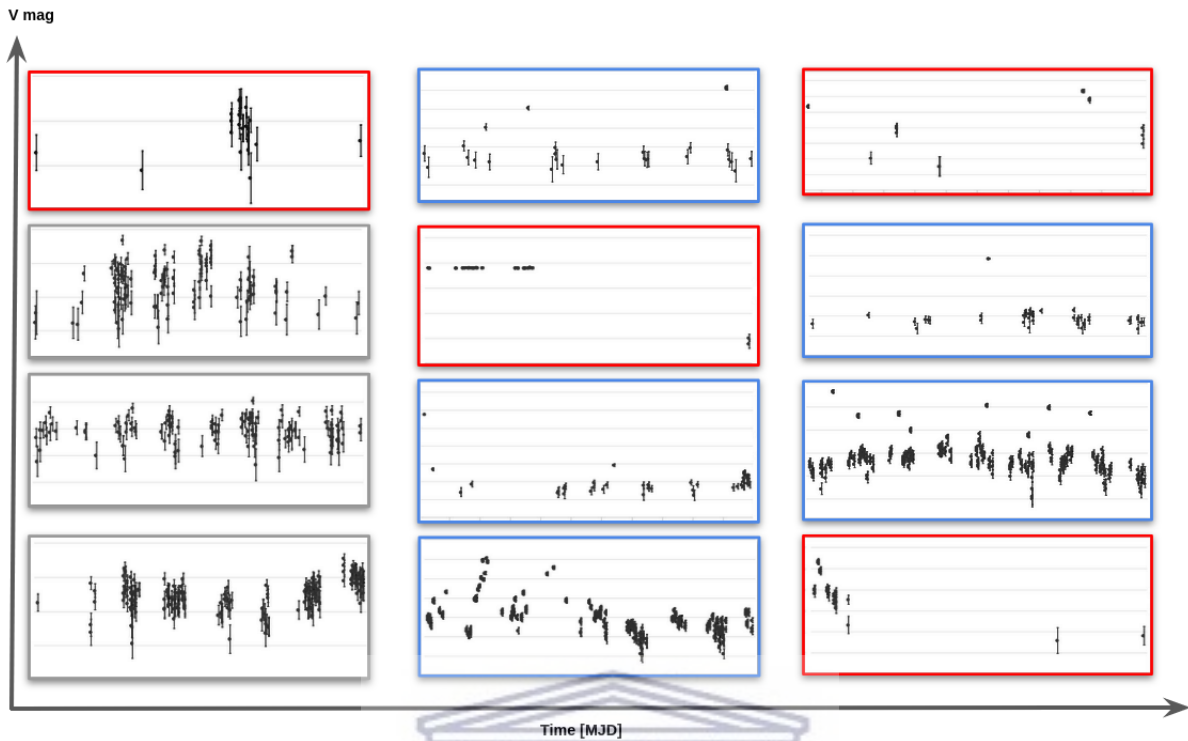


FIGURE 4.8: Light curves of random objects in the CRTS data. The black points are the light curve points, and this plots are only intended to show different light curve shapes. Highlighted in grey, blue and red boxes are the normal, anomaly, and bogus classes respectively.

We plot in Figures 4.8, 4.9 and 4.10 the top twelve light curves sorted randomly, by raw anomaly scores and human retrained scores, respectively. Notice that the light curves plotted randomly display all the three classes (bogus, normal, and anomaly). The normal classes disappear after running iForest and the top twelve anomalies are dominated by the bogus class light curves and a few anomaly class light curves.

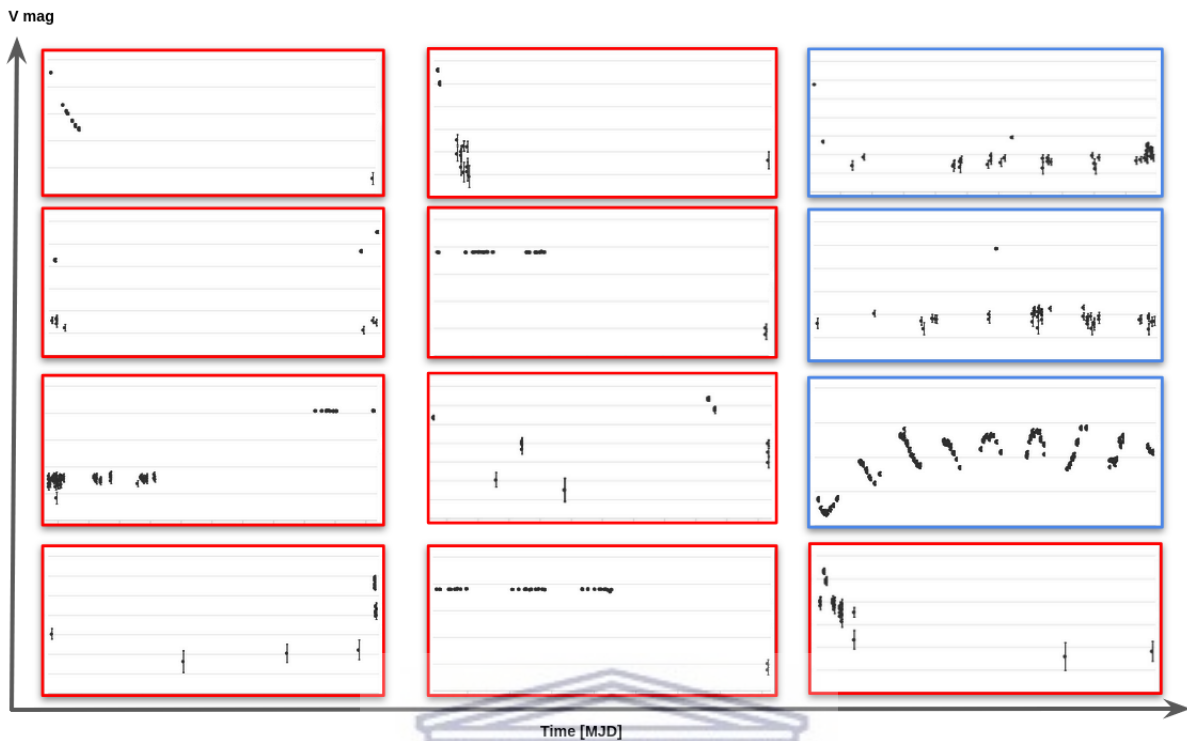


FIGURE 4.9: The top 12 anomalies as by the raw anomaly scores. Notice that only the **bogus** (red boxes), and **anomaly** (blue boxes) classes are detected in the top 12 anomalies.

4.3.0.1 Analysis of the Top 12 anomalies after Active learning

Figure 4.10 shows the top twelve anomalies after assigning a relevance score to the top 100 anomalies as by the raw anomaly scores and running active learning to predict the human retrained scores (see section 3.1.6). We find from these plots that the **bogus** class light curves are successfully flagged from the top twelve anomalies, and only the **anomaly** class is retrieved. We outline in table 4.1 the names, coordinates and labels of the top 12 anomalies. We also give a detailed description of the anomalous properties of *IRAS 04188+0122*, *SN 2013cv*, *1FGL J0050.0-0446* and *CSS130509:121714+121504*.

TABLE 4.1: The top 12 anomalous objects from the MANTRA data. We give their CSS names, labels, coordinates, other identifier names and periods. Information outlined here is taken from the Caltech database¹ and SIMBAD database.

| CSS Name ^a | Ra [Deg] ^b | Dec [Deg] ^c | Label ^d | Other identifiers ^e | Period [MJD] |
|-----------------------|-----------------------|------------------------|--------------------|--------------------------------|--------------|
| CSS_J042127.2+012913 | 65.3635 | 1.4871 | Carbon Star | IRAS 04188+0122 | 408.95 |
| CSS_J051750.2+005415 | 79.4590 | 0.9042 | Flare Star | – | – |

| Continuation of Table 4.1 | | | | | |
|-----------------------------|-----------------------|------------------------|------------------------------|--------------------------------|------------------------------|
| CSS Name ^a | Ra [Deg] ^c | Dec [Deg] ^d | Label ^b | Other identifiers ^e | Period ^f [MJD] |
| CSS_J005021.4-045221 | 12.5893 | -4.8726 | Blazar | 1FGL J0050.0-0446 | – |
| CSS_J121713.9+121505 | 184.3080 | 12.2516 | type Ia-91T SN | – | – |
| CSS_J052650.4+244516 | 81.7101 | 24.7545 | Carbon Star | IRAS 05237+2442 | 0.73 |
| CSS_J123014.1+251806 | 187.5587 | 25.3019 | Blazar | 1FGL J1230.4+2520 | – |
| CSS_J082433.0+243843 | 126.1375 | 24.6455 | Blazar | ICRF J082433.0+243843 | – |
| CSS_J040338.3-104945 | 60.9097 | -10.8292 | High proper motion star | DR2 3190603665145353984 | – |
| CSS_J131012.3+474515 | 197.5513 | 47.7543 | High proper motion star | – | – |
| CSS_J104031.6+061722 | 160.1317 | 6.2895 | Blazar | SDSS J104031.62+061721.7 | – |
| CSS_J140453.9-102701 | 211.2246 | -10.4505 | Cataclysmic Variable Star | SDSS J140453.98-102702.1 | – |
| CSS_J162243.0+185734 | 245.6790 | 18.9596 | Type Ia-91T SN | PTF 13asv | – |

¹We query the Caltech database: http://numuku.caltech.edu/cgi-bin/getcssconedb_release_img.cgi, using the coordinates of the objects. The caltech database has an updated light curve data, wherein most of the noise is removed from the light curves. Highlighted in bold are objects that were detected as anomalies by **ASTRONOMALY** because of the noise in the MANTRA data, and this noise is removed in the updated data from caltech. The light curves of this objects can be seen on the top right plot and bottom left plots in Figure 4.10.

^aThis is the object name as the CSS

^bThis is the right ascension in degrees

^cThis is the declination ascension in degrees

^dThis is classification of the objects taken from the MANTRA data and SIMBAD

^eThis are object IDs that can be used to identify object in other surveys.

^fThis the period estimate of the periodic objects, taken from the Caltech.

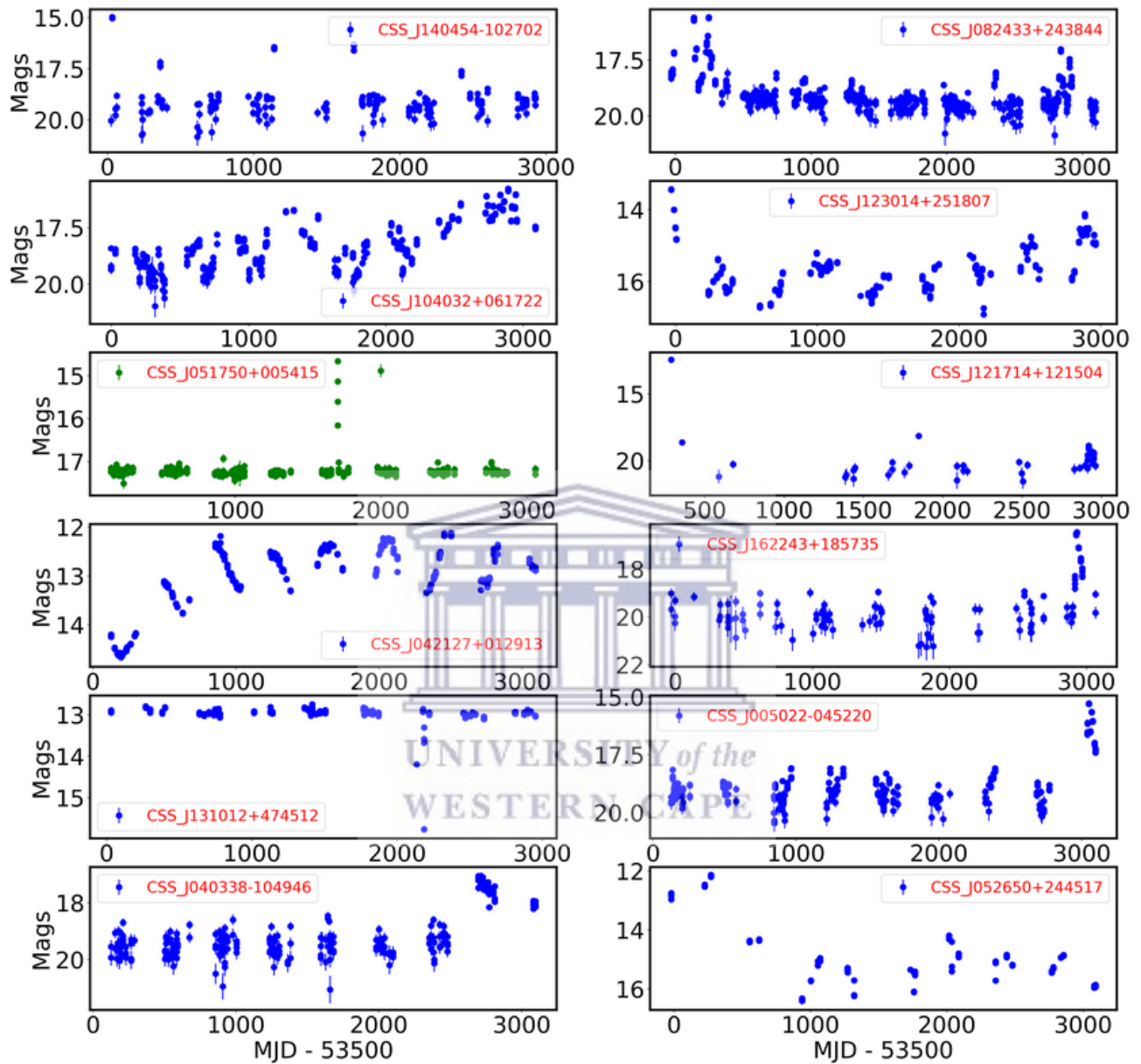


FIGURE 4.10: The top 12 anomalies after active learning. Notice that only the **anomaly** class is detected, and all objects except the one highlighted in green, were assigned a relevance score.

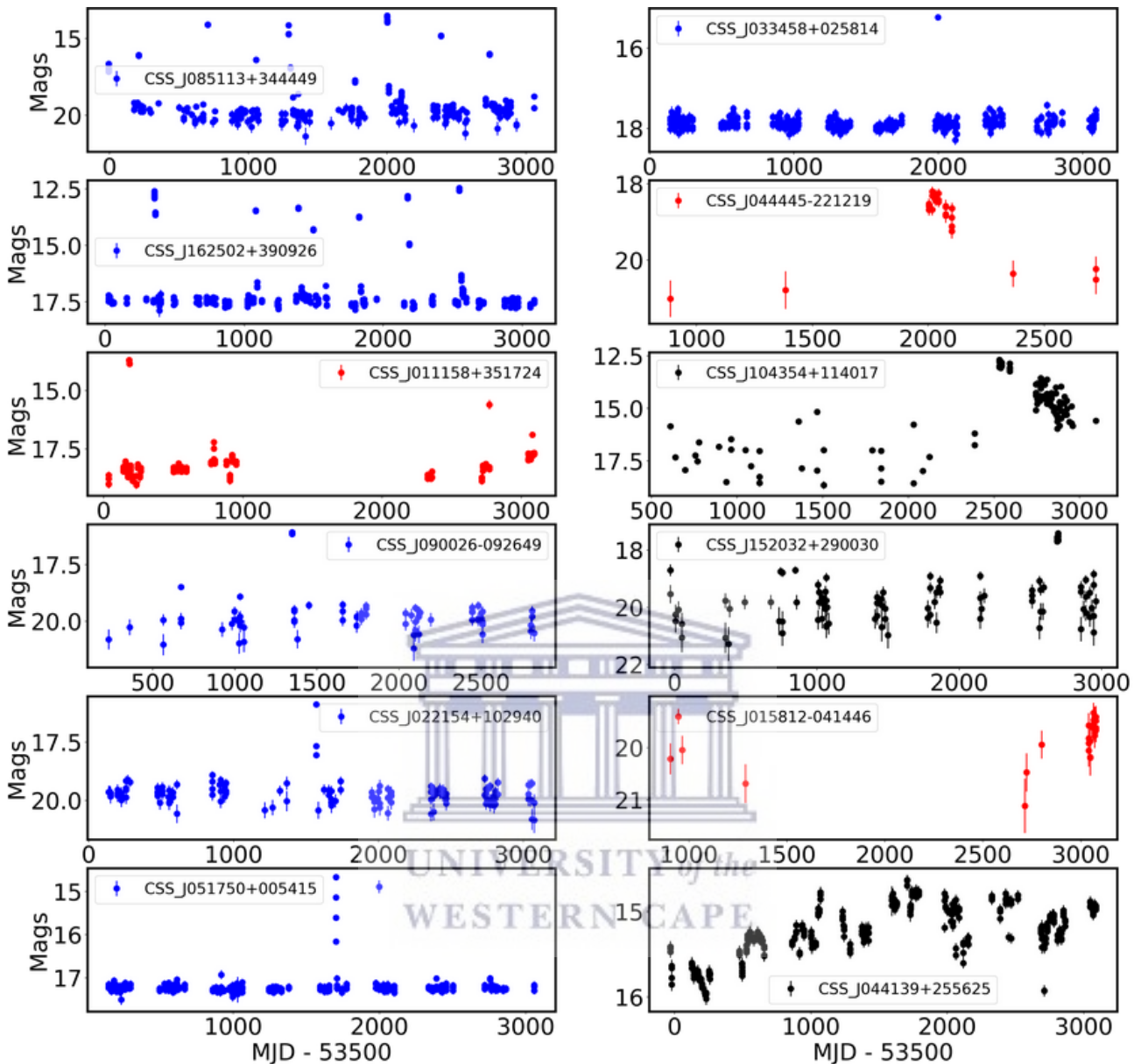


FIGURE 4.11: Top 12 anomalies flagged by active learning (i.e., they were not assigned a relevance score, rather predicted by active learning). Notice that the `anomaly` class (highlighted in blue) dominates, and few `bogus` (red) and `normal` (black) classes are detected.

IRAS 04188+0122

This is a red N-type carbon star that was first discovered as APM 0418+0122 by Totten and Irwin 1998. It is characterised by an anomalous red spectroscopic behaviour, where there are no flux measurements detected before wavelengths $\sim 5000 \text{ \AA}$. Figure 4.13 shows the spectra of APM 0418+0122 compared to other carbon stars in the sample described by Totten and Irwin 1998. APM 0418+0122 is located at an approximate distance of 6 kpc, with galactic coordinates of $RA = 65.36359^\circ$ and $Dec = 1.48691^\circ$ (Mauron, Gigoyan, and Kendall 2007).

The light curve displays a periodic variation with an approximate period of 408.95 MJD. It also displays an increase in the magnitude of the entire periodic variation of the object, where the variation starts at a low magnitude ($V \sim 14.5$ mag) and then increases to higher magnitudes ($V \sim 12.5$ mag, see Figure 4.12).

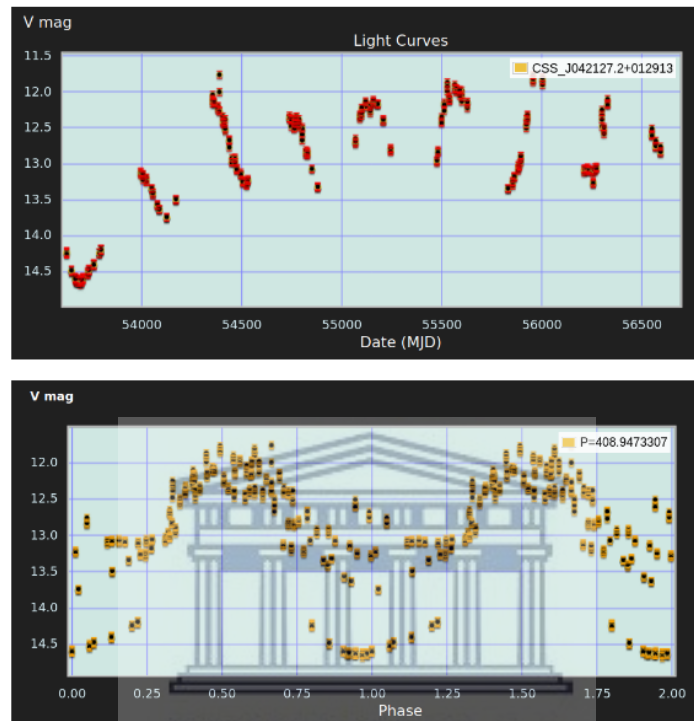


FIGURE 4.12: Light curve of IRAS 04188+0122. Notice that it is identified as CSS_J042127.2+012913 in the CRTS database. The top plot is the original light curve of IRAS 04188+0122 in the MANTRA data, and the bottom plot is the phase folded light curve with period of 408.95 MJD. Credits: The Catalina survey data release 1 (CSDR1).

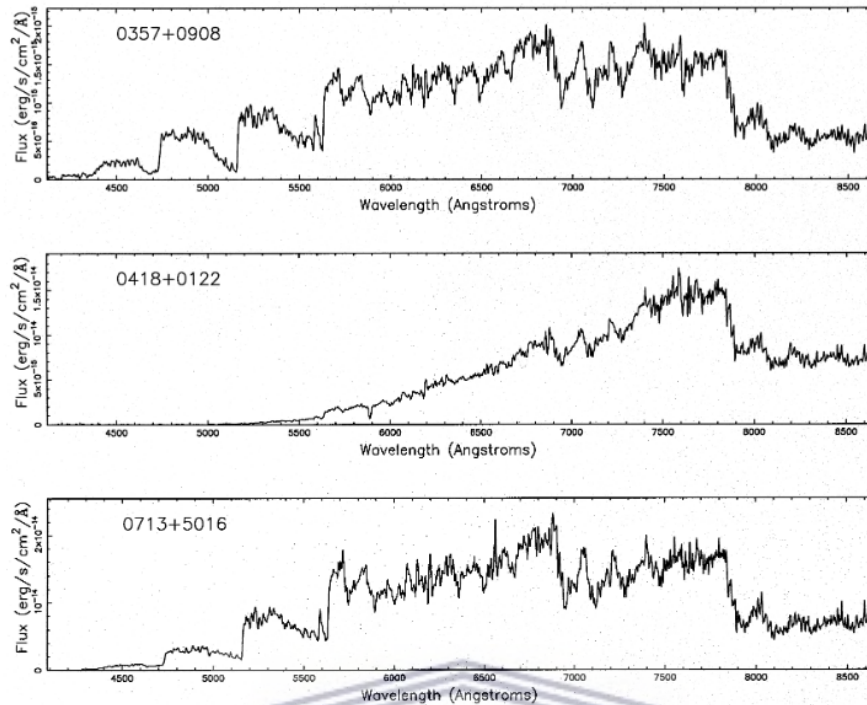


FIGURE 4.13: Anomalous spectra of IRAS 04188+0122. Notice the anomalous characteristic of the IRAS 04188+0122 in the middle spectrum compared to the other two spectra (Totten and Irwin 1998).

SN 2013cv

SN 2013cv (also known as iPTF 13asv or PTF 13asv) is a peculiar type Ia supernova which was discovered by Zhou et al. 2013 in the galaxy SDSS J162243.02+185733.8. Its classification lies between the super-Chandrasekhar and the normal type Ia SN events (Cao et al. 2016). The super-Chandrasekhar events occur when the progenitor white dwarf of type Ia SNe exceeds the Chandrasekhar mass limit of $1.44M_{\odot}$ (Tomaschitz 2018).

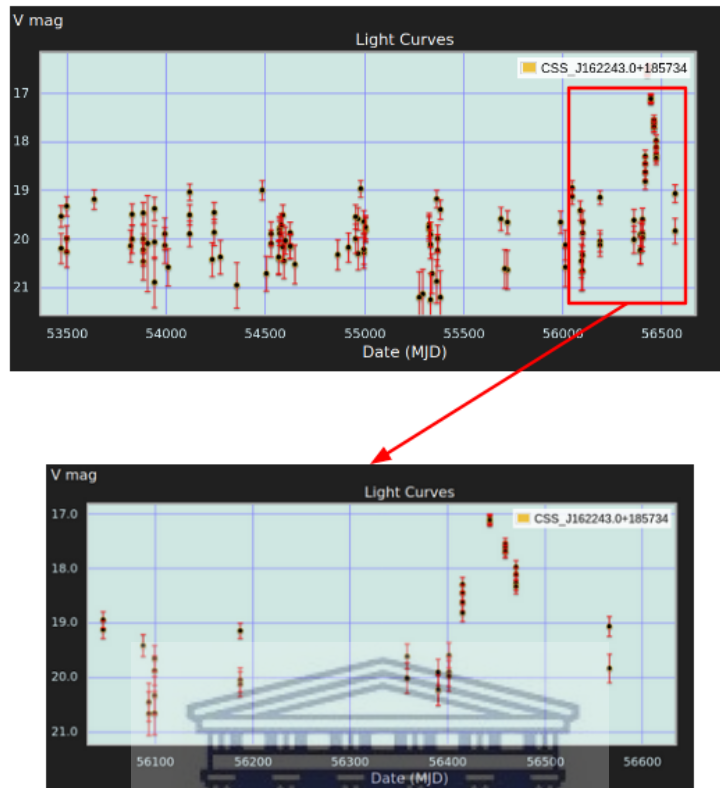


FIGURE 4.14: Light curve of SN 2013cv. The top plot is the original light curve of SN 2013cv and the bottom plot is zoomed in light curve where the SN occurs. Credits: The Catalina survey data release 1 (CSDR1).

SN 2013cv was also detected as anomalous by Pruzhinskaya et al. 2019, where they employed iForest to search for anomalies in the open supernova catalogue (see section 2.4).

1FGL J0050.0-0446

This is blazar located at a redshift of 0.922 associated with the AGN *PKS 0047-051*. It is classified as the flat spectrum radio quasar (FSRQ), which is a class of blazar characterised by rapid and strong variability. They are thought to be the most extreme classes of the AGN. The host galaxy of *1FGL J0050.0-0446* has a logarithmic SMBH mass of $8.2 M_{\odot}$ (Chen et al. 2021).

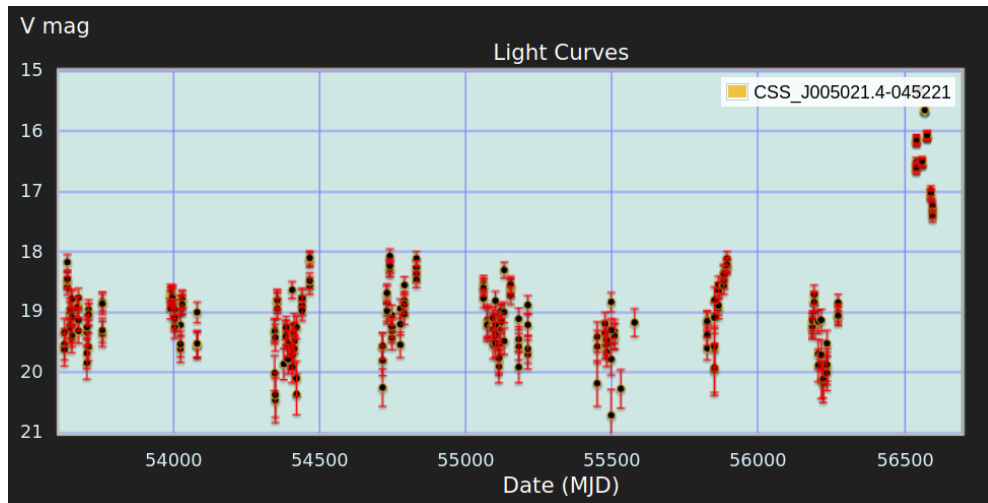


FIGURE 4.15: Light curve of 1FGL J0050.0-0446. Credits: The Catalina survey data release 1 (CSDR1).

ASTRONOMY detected *1FGL J0050.0-0446* an anomaly after we ran active learning. It was found from Figure 4.15 that *1FGL J0050.0-0446*, identified as *CSS_J005021.4-045221*, displays a rapid variation in brightness. It, however, displays a sudden increase in brightness at ~ 56500 MJD, which might be the reason why our algorithm detected it as an outlier.

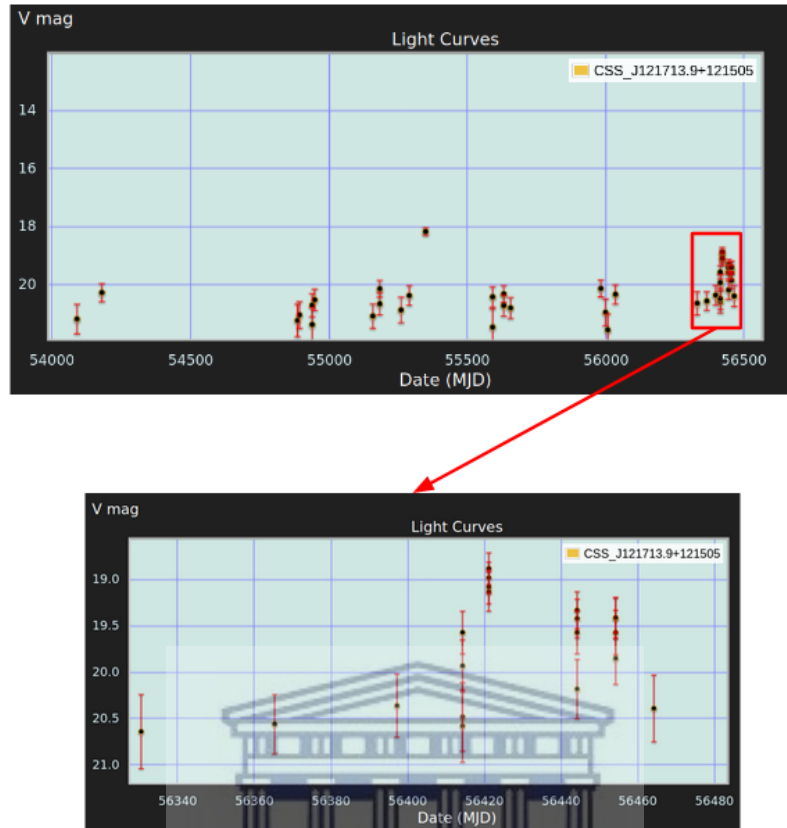
CSS130509:121714+121504

UNIVERSITY of the
WESTERN CAPE

CSS130509:121714+121504, herein referred as *CSS130509*, is a peculiar type Ia-91T supernova located at redshift 0.10 (Yang et al. 2013). It was discovered in 2013 by the CRTS, and no additional information is available for it in the literature. Type Ia-91T supernovae differ from the normal type Ia supernovae in that they have broad light curves with higher peak luminosity (Taubenberger 2017).

The original light of *CSS130509*, we refer from the MANTRA data can be seen on the bottom left plot in Figure 4.10. Notice that the light curve has two bright points at the beginning; these points are flagged out in the new Catalina survey data release 1 (CSDR1; see Figure 4.16)⁴. This implies that the two points are triggered by artefacts, which might be the reason why our algorithm detects *CSS130509* as an anomaly in the data. Even though that might be the case, *CSS130509* is an interesting transient object.

⁴http://nunuku.caltech.edu/cgi-bin/getcssconedb_release_img.cgi#simtable

FIGURE 4.16: Same as Figure 4.14 but for *CSS130509*.

4.4 Discussion

We employed iForest to assign anomaly scores to objects in the MANTRA data, based on features extracted using `feets`. The data has a significant class imbalance (see Figure 4.2) and given the characteristics of iForest: “anomalies are few and different” (Liu, Ting, and Zhou 2008), we expect classes with few objects to be detected as anomalies. However, we found that iForest detected anomalies in each of the MANTRA classes (see Figure 4.7). This is because the `feets` feature is biased towards bogus light curves, as a result, iForest detects bogus light curves as anomalies (see Figure 4.9). We flagged these bogus light curves from the top ranks by running active learning as described above.

After active learning, no bogus light curve was detected in the top 12 anomalies as by the human retrained score. However, only the interesting anomalies are retrieved. An example is a long-period red carbon star, *IRAS 04188+0122*. This object has an anomalous light curve, where the entire brightness of the object increases from ~ 14.5 mags to ~ 12.5 mags (see Figure 4.12), it also has an anomalous spectra (see Figure 4.13). These results show that active learning techniques, coupled with

anomaly detection algorithms, are promising tools to use in searching for anomalies in light curve data and personalising interesting anomalies.



Chapter 5

Application of **ASTRONOMALY** on the Photometric LSST Astronomical Time-Series Classification Challenge data

We have seen in the previous Chapter that **ASTRONOMALY** is a promising tool to be employed for anomaly detection in any given light curve data. Even though it has detected interesting objects in the MANTRA data, its overall performance is not fully analysed. This is because the anomalous classes are not known in the MANTRA data; hence we cannot assess the performance of **ASTRONOMALY** based on ranking as discussed in section 3.1.7.

In this Chapter, we apply **ASTRONOMALY** to simulated data from PLAsTiCC with a known anomalous sample. We then assess its performance based on the ranks at which the anomalous objects appear in the results.

5.1 The Photometric LSST Astronomical Time-Series Classification Challenge

PLAsTiCC is a `kaggle` classification challenge developed by two collaboration teams: the LSST dark energy science collaboration (DESC¹); and the transient and variable stars collaboration (TVS²). The

¹<http://lsstdesc.org/>

²<https://lsst-tvssc.github.io/>

participants are tasked to use machine learning algorithms to classify simulated large light curve data. This data represents what is expected from the Vera C. Rubin observatory during its 10-year LSST. The classification challenge is designed to prepare the scientific community for the big data challenges that will come with the LSST (Allam et al. 2018).

The PLAsTiCC data have light curves for both transient and variable events. Each object in the data is represented by six light curves from six passbands: u , g , r , i , z , y (see Figure 5.2). Figure 5.1 shows the wavelength ranges covered by each of the six passbands and their corresponding filter efficiency. The advantage of observing objects in different passbands is that they can resolve fainter objects that are located at higher redshifts (Allam et al. 2018). However, objects located at high redshifts are often affected by observational challenges such as Galactic extinction, which needs to be corrected.

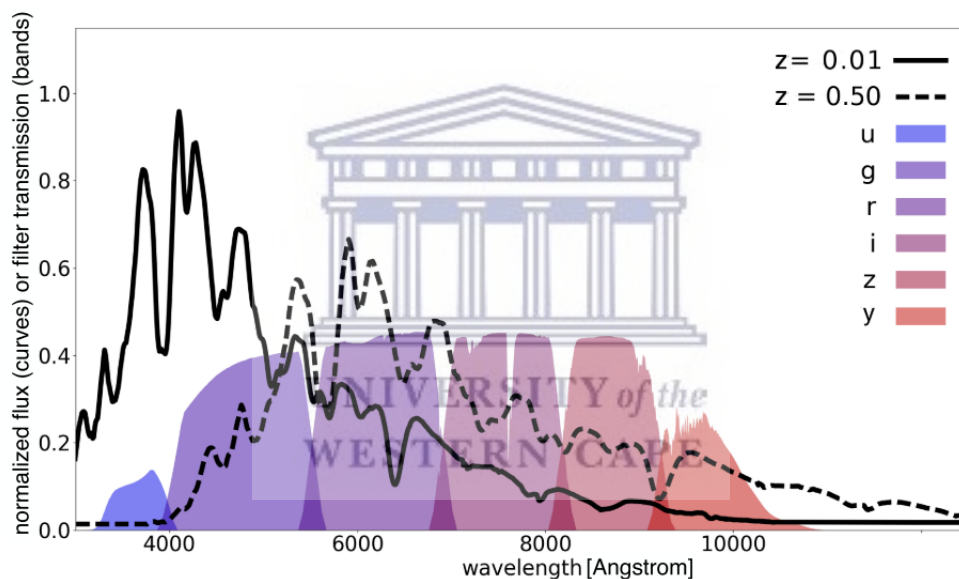


FIGURE 5.1: A plot showing the distribution of the six passbands in the LSST data and the spectra of a SN Ia located at redshift 0.01 (solid black curve) and 0.50 (dashed black curve). The x-axis shows the wavelength covered by each passbands and the y-axis is their normalised flux (for the curves) and filter transmission (for the bands). Notice that shapes of both curves are similar, however, SN Ia at high redshift appears to be redder than that of a lower redshift (Allam et al. 2018).

The light curves data in PLAsTiCC are corrected for the atmosphere transmission and Galactic extinction. The latter is important as it accounts for the amount of light absorbed by the Milky Way dust as the light passes through it to an observer on Earth. This is known as the Milky Way extinction, and its value for each object is given in the metadata file (described below) with column a name `MWEBV`.

The LSST will have two survey fields, both of which were incorporated in simulating the PLAsTiCC data. This includes the wide-fast-deep (WFD) and the deep drilling fields (DDF). The former will cover a small region of the sky that will be sampled on a regular basis to obtain the optimal observational depth, while the latter will cover a wider region of the sky that will be sampled less frequently. The DDF will be able to resolve fainter objects, and their light curves will have small flux errors and most frequent observations as opposed to those from WFD. However, the WFD will make more discoveries because of its wider field (Ivezić et al. 2019; Allam et al. 2018). The PLAsTiCC metadata file (described below) has a boolean column that indicates whether a given object is from the DDF or WFD.

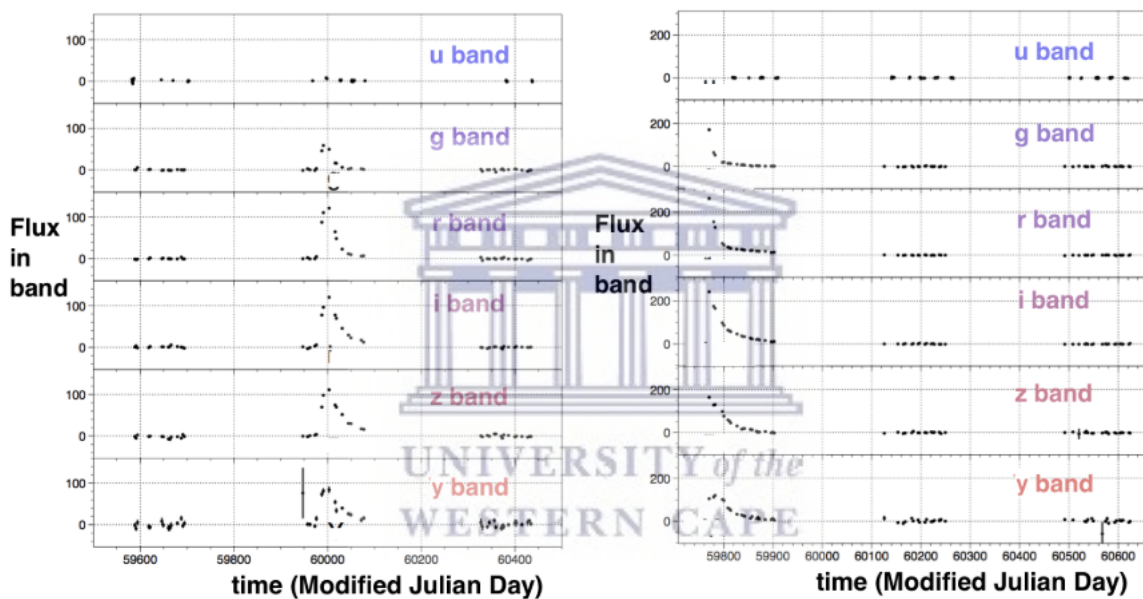


FIGURE 5.2: Example light curves of two randomly selected objects from the PLAsTiCC data. Each panel represents one object plotted in the six LSST bands. These plots also shows the gaps between the observation, which is what is expected from the LSST (Allam et al. 2018).

5.2 The Data

Since PLAsTiCC is a machine learning classification challenge, its data is split into the training set and test set. The training set is composed of 8000 labeled objects, and the test set has ~ 3.5 million unlabelled objects that need to be classified. The data is stored in multiple `csv` files that can be found on the kaggle website³.

³<https://www.kaggle.com/c/PLAsTiCC-2018/data>

In this work, we are interested in the test data files. However, due to the computational cost of our feature extraction techniques, we apply our machine learning algorithms on two samples drawn from the original data which has ~ 3.5 million object (see table 5.1 and section 5.2.1 for more details).

PLAsTiCC has 18 classes of different transient and variable events. Some of the more common events (“normal” events) in the simulated data includes type Ia and core collapse supernovae, variable stars (e.g., RR Lyrae, and eclipsing binaries), and AGN. The data also contains $\sim 0.37\%$ of anomalous objects, including calcium rich transients (CaRTs), kilonovae, and pair instability supernovae (PISNe). Table 5.1 shows a full list of transient and variable events in the PLAsTiCC data.

The data is simulated from different models and real observation data (see Kessler et al. 2019 for a full description). This models includes: SALT-II (Guy et al. 2007), SED (Kessler et al. 2010; Anderson et al. 2014; Galbany et al. 2016), MOSFiT (Villar et al. 2017; Guillochon et al. 2017; Sako et al. 2018), GalFast (Villar et al. 2017; Guillochon et al. 2018; Jiang, Jiang, and Villar 2020), Galaxia (Sharma et al. 2011), CODEX (Ireland, Scholz, and Wood 2008; Ireland, Scholz, and Wood 2011) and PyLIMA (Bachelet et al. 2017).

The real observations are from the: joint light curve analysis (JLA; Betoule et al. 2014), open supernova catalog (OSC; Guillochon et al. 2017) and GenLens which is microlensing data drawn from multiple surveys (Udalski et al. 1992; Alcock et al. 1993; Bond et al. 2001). Some light curves are modeled from theoretical calculations (TC; Kasen et al. 2017). We highlighted in table 5.1, the models used in simulating each event in the data.

TABLE 5.1: A list of classes in the PLAsTiCC test set, their true labels, model used in the simulation, number of events in the original test data, number of events in the sample used in this work and the redshift range of the simulated data.

| Target ^a | True Label ^b | Model ^c | N _{event_original} ^d | N _{event_new} ^e | Redshift Range ^f |
|---------------------|-------------------------|--------------------|--|-------------------------------------|-----------------------------|
| 90 | SNIa | SALT-II+JLA | 1,659,831 | 1,538 | < 1.6 |
| 67 | SNIa-91bg | SED | 40,193 | 1,538 | < 0.9 |
| 52 | SNIax | SED+OSC | 63,664 | 1,538 | < 1.3 |
| 42 | SNIi | SED | 1,000,150 | 1,538 | < 2.0 |
| 62 | SNIbc | SED+MOSFiT | 175,094 | 1,538 | < 1.3 |
| 95 | SLSN-I | MOSFiT | 35,782 | 1,538 | < 3.4 |
| 15 | TDE | MOSFiT | 13,555 | 1,538 | < 2.6 |
| 64 | KN | TC | 133 | 133 | < 0.3 |

| Continuation of Table 5.1 | | | | | |
|---------------------------|--------------------------------|--------------------|--|-------------------------------------|-----------------------------|
| Target ^a | Label Description ^b | Model ^c | N _{event_original} ^d | N _{event_new} ^e | Redshift Range ^f |
| 88 | AGN | DRW | 101,424 | 1,538 | < 3.4 |
| 92 | RRL | GalFast | 197,155 | 1,538 | 0 |
| 65 | M-dwarf flare stars | GalFast | 93,494 | 1,538 | 0 |
| 16 | Eclipsing binaries (EB) | Galaxia | 96,572 | 1,538 | 0 |
| 53 | Mira stars | CODEX | 1,453 | 1,453 | 0 |
| 6 | μ Lens-Single | PyLIMA+GenLens | 1,303 | 1,303 | 0 |
| 991 | μ Lens-Binary | GenLens | 533 | 50 | 0 |
| 992 | ILOT | MOSFiT | 1,702 | 50 | < 0.4 |
| 993 | CaRT | MOSFiT | 9,680 | 50 | < 0.9 |
| 994 | PISN | MOSFiT | 1,172 | 50 | < 1.9 |
| Total | the total number of objects | – | 3,492,890 | 20,007 | – |

^aThis is the target integers from the PLAsTiCC metadata files. Note that the targets 991-994 were treated as anomalies in this work as they were categorised as the unknown classes in the original classification challenge.

^bThis is the corresponding true label of the object as by the literature (see Chapter 1)

^cThis is the model through which the data was simulated from. The “+” sign indicates cases where the data is simulated from multiple models.

^dThis is the number of object in the original in the original test set.

^eThis is the number of object in the new sample used in this work.

^fThis is the the redshift range covered by objects in a given target label. A redshift of zero indicates that the objects are within the Milky Way galaxy and those with redshift range > 0 are extra galactic.

The PLAsTiCC data is composed of two main types of files: the light curve files, and the metadata files. The light curve files have light curves of multiple objects in the six bands, each object identified by a unique ID. The metadata files list the properties of each object in the data, each identified by a unique ID. This means that the metadata files and the light curves can be linked by the unique ID of a given object. We describe the columns found in both the light curve files and the metadata files below:

The metadata columns

- **object_id**: unique ID of the object.

- **ra**: right ascension in degrees.
- **decl**: declination in degrees.
- **gal_l**: Galactic longitude in degrees.
- **gal_b**: Galactic latitude in degrees.
- **ddf**: A boolean used to identify if a given object is sampled from the DDF or WFD, it takes a value of one for DDF and zero otherwise.
- **hostgal_specz**: the spectroscopic redshift of the target object.
- **hostgal_photoz**: The photometric redshift of the galaxy at which the event occurs.
- **hostgal_photoz_err**: The errors in the measurements of the **hostgal_photoz** based on the projections of the LSST survey.
- **distmod**: The distance modulus to the target object, calculated from the **hostgal_photoz**.
- **MWEBV = MW E(B-V)**: this is the Galactic extinction value as described above. It is a function of **ra** and **decl** and it provides information about the reddening and dimming of target objects as they pass through the Milky Way dust to an observer on Earth.
- **target** and **true_target**: this gives the classification or label of an object at hand. The target labels for the training set are found in `training_set_metadata.csv` file (in the **target** column) on the original `kaggle` website and those of a the test set in the `plastic_test_metadata.csv` (in the **true_target** column) file on the unblinded data release on the `zenodo` website⁴.

The Light Curve columns

- **object_id**: unique id of the objects, similar to that of the metadata.
- **mjd**: observation time in MJD.
- **passband**: an integer representing the LSST passbands, where 0,1,2,3,4, and 5 represents the *u*, *g*, *r*, *i*, *z*, and *y* passbands, respectively.
- **flux**: the measurement of the brightness of the objects in each of the six passbands. The **flux** has arbitrary units.

⁴<https://zenodo.org/record/2539456#.YXvkPpuxXs0>

- **flux_err**: the uncertainty in the **flux** measurements.
- **detected**: a boolean used to indicate if the object is detected after subtracting the reference template image from the image containing the target object. This step is done during aperture photometry (see Chapter 1). **detected** = 1 if the object is detected and zero otherwise.

Note that additional metadata (such as redshift and sky location) can be added to the column list above.

5.2.1 The Samples

We sampled two different data from the original PLAsTiCC test set for our analysis. The first sample is motivated by the assumptions through which iForest is built (anomalies are few and different; Liu, Ting, and Zhou 2008), and we aim to use this sample to ensure that our algorithms work on a very simple test case. We sample a small dataset that is composed of two distinct labels (KN and RRL) that should be trivially distinguishable. This sample is composed of 133 KNe and 10,000 RRL that were randomly picked. The final sample is composed of 10,133 objects, and we refer to this as the `KN_RRL_sample`. Since KNe are few and different from the RRL, they are treated as anomalous objects.

The second sample is composed of all labels in the PLAsTiCC test set, where we randomly selected all objects in the KN, Mira, and μ Lens-Single labels and picked 50 objects from each of the anomalous labels: μ Lens-Binary, ILOT, CaRT, and PISN. Lastly, we selected 1,538 objects from the remaining labels in the data. The final sample is composed of 20,007 objects, and we refer to this as the `all_labels_sample`. The μ Lens-Binary, ILOT, CaRT, and PISN are treated as the anomalous objects in this sample, as they were labeled the unknown classes in the original PLAsTiCC challenge.

It is important to note that both samples (`KN_RRL_sample` and `all_labels_sample`) were used for testing purposes before scaling to the larger sample and `feets` failed our test. This is why we did not apply our techniques to a larger PLAsTiCC sample.

5.3 Astronomy applied to a small PLAsTiCC sample

We applied *ASTRONOMALY* on the `KN_RRL_sample` data. Figure 5.3 shows example light curves of the RR Lyrae and KNe objects in the `KN_RRL_sample`. We see from these plots that the two classes are

significantly different from each other. We expect iForest to detect the objects from KNe as anomalous since they are few in the sample.

5.3.1 The Methodology

We follow the same procedure as that described in section 3.1. Since the PLAsTiCC data is observed in multiple bands and the light curves of multiple objects are embedded in one csv file, then it is treated as **case_1b** dataset as described in section 3.1.1.

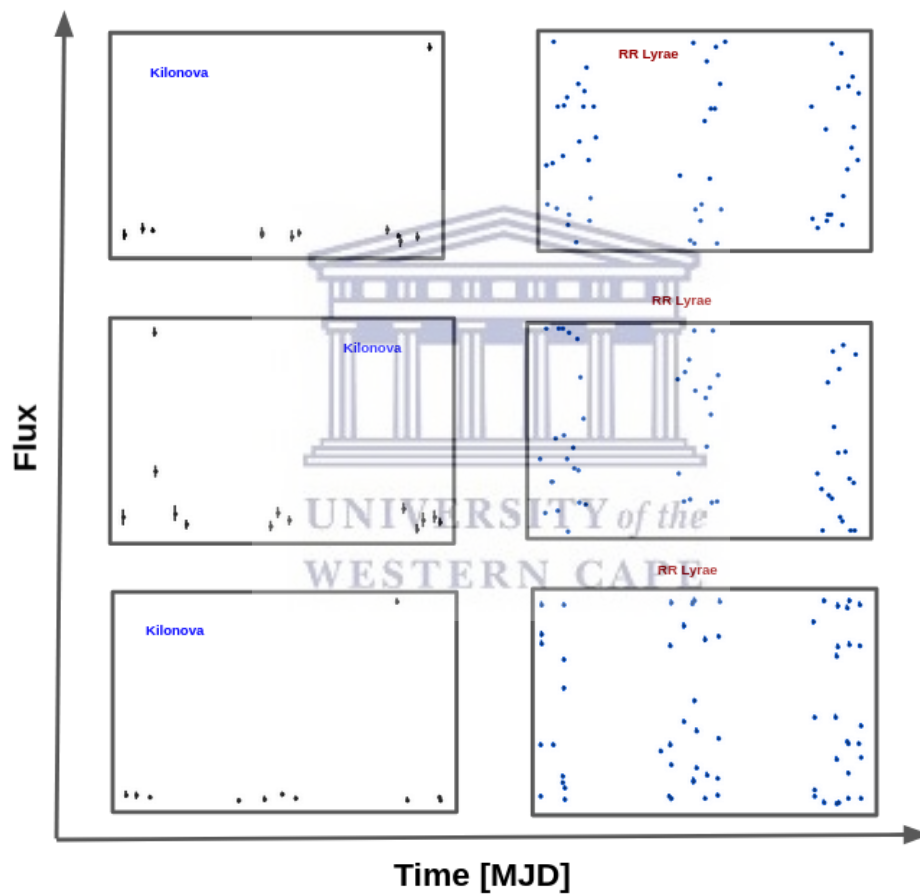


FIGURE 5.3: Example light curves from the `KN_RRL_sample`. These plots are from random bands in the sample, and we only intend to show the difference in the light curve shapes between the kilonova and RR Lyrae classes.

5.3.1.1 Data Pre-processing and Feature Extraction

The PLAsTiCC light curves are recorded in flux measurements in six passbands instead of magnitude like those from the CRTS. Of course we can convert the flux to magnitudes using equations 3.1 and

3.5, however, our feature extractor is not restricted to magnitude units, i.e., it can be applied to any time series data (Nun et al. 2015b; Cabral et al. 2018). We thus use the flux measurements themselves.

The light curves also have large gaps between observations (see Figure 5.3). We split the light curves in chunks based on the gaps between the observations to account for these gaps. However, the process of splitting the light curves into chunks is computationally expensive. Because of this, we choose to neglect it in this work. However, we describe the two possible methods in which the complexity of the PLAsTiCC can be solved, where in method 1, we consider the gaps between the observations, and in method 2 we neglect the gaps (see Figure 5.4).

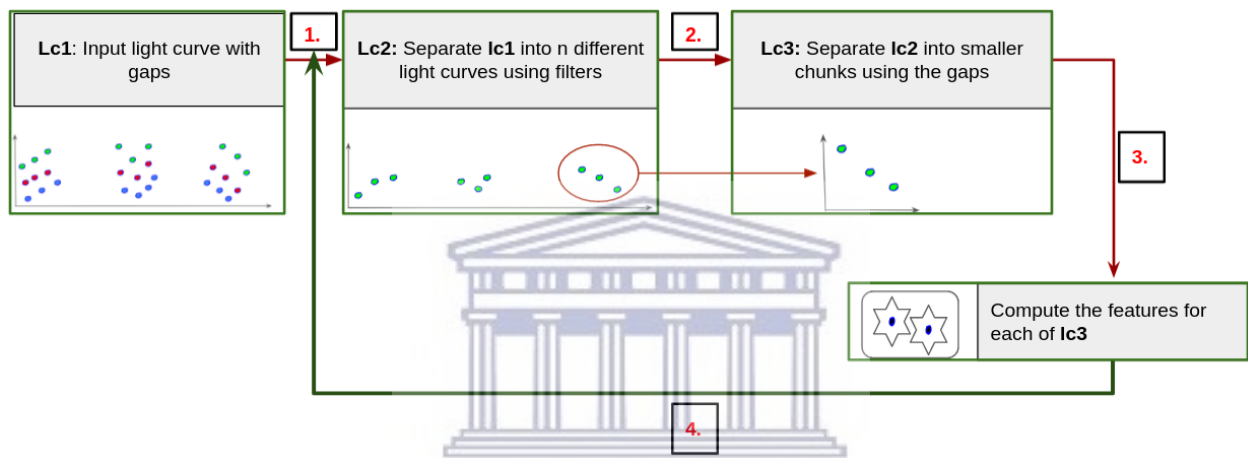


FIGURE 5.4: The flow diagram of the pipeline used for data pre-processing in the PLAsTiCC data. The numbers in red are the steps that we followed in reducing the complexity of the input light curves (steps 1 and 2) and finally, in step 3, we extracted features from the light curve from step 2. The whole process was repeated for all passbands.

Method 1: Considering the gaps between observations

Even though this method was not considered in this work, we have experimented on it and found that it is computationally expensive. Our algorithms perform poorly with light curves pre-processed with it. This is because separating the light into chunks results in very few light curves passing the number of point cut described below. This is also because most light curve chunks that pass the pointcut do not have the actual transient activity; hence all objects from both the “normal” and anomaly classes will have similar light curve characteristics. Below we describe the steps covered by this method:

1. **Passbands:** Separate the input light curves of a single object into n passbands (see step 1 from fig 5.4), where n is the number of passbands. We chose $n = 4$ corresponding to *girz* passbands.

2. **Gaps:** Separate the light curve from step 1 into k chunks, where k is the number of light curves. $k = \text{gaps} + 1$, where `gaps` is the number of gaps in the light curve. We computed the difference between consecutive observations (`time_difference`), we then defined a gap as `time_difference` ≥ 100 days. For example, the bottom left light curve on fig 5.3 has 2 gaps, it will be separated into 3 smaller light curves chunks. If the ID of the object is 13, we renamed it to 13_0, 13_1, and 13_2, where 0, 1, and 2 correspond to light curves 1, 2, and 3 respectively.
3. **Point Cut:** The `feets` feature extractor needs atleast 20 point in a light curve to compute all features described in table 3.1. We found that the following features:

- `Freq1_harmonics_rel_phase_0`, `PercentDifferenceFluxPercentile`,
- `FluxPercentileRatioMid20`, `FluxPercentileRatioMid35`,
- `FluxPercentileRatioMid50`, `FluxPercentileRatioMid65`,
- `FluxPercentileRatioMid80`, `Freq1_harmonics_rel_phase_0`,
- `Freq2_harmonics_rel_phase_0`, and `Freq3_harmonics_rel_phase_0`,

are the only features that can't be calculated by `feets` with less points (< 20 points). They were neglected in the feature extraction process. This means we can set the point cut to a lower number, this was set to 5.

4. **Feature Extraction:** We extracted 50 features from the light curves that pass the point cut. We consider each light curve from step 2 as a separate object (i.e., we compute features for each of objects 13_0, 13_1, and 13_2). Steps 1 to 4 were repeated for all passbands, and the features were concatenated to form a matrix with 200 columns. For example, the `mean` feature for object 13_0 will be `mean_g`, `mean_i`, `mean_r` and `mean_z` for the 4 passbands.

Method 2: Neglecting the gaps between observations

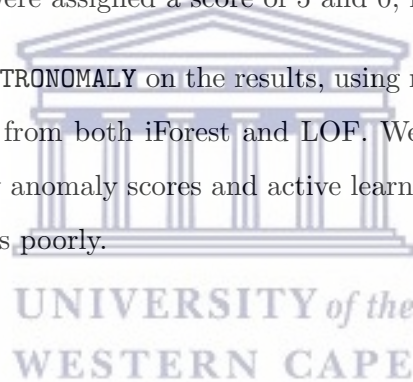
In this method, we considered the entire light curves with the gaps between the observations. This is because our feature extractor `feets` is not sensitive to unevenly sampled light curves (Nun et al. 2015b). Splitting the light curves into chunks as in **method 1** led to fewer points in the light curve. This means that a large fraction of the data will not pass the pointcut, resulting in losing information about the observations. We thus consider steps 1, 3, and 4 from **method 1**; the only difference is that we are extracting features from the entire light curve.

We computed 50 features for each light curve band with the `feets_features` extractor. A full list of all the computed features can be seen in table 3.1 with the following neglected: `con`, `Autocor_length` and features outlined in **method 1**. As with **method 1**, we compute the features on each passband and concatenate them. This means that each object is represented by 200 features from the *girz* bands.

5.3.1.2 Anomaly Detection, Active Learning, and Classification

iForest and LOF were employed to assign anomaly scores to each object in the `KN_RRL_sample`, each represented by features described in **method 2**. We use the default hyperparameters as those described in Chapter 4 for iForest and set `n_neighbours = 20` for LOF. We then ran active learning on the results by assigning relevance scores to the top 500 anomalies as by the raw anomaly scores from iForest. The KNe and RR Lyrae were assigned a score of 5 and 0, respectively.

We assessed the performance of *ASTRONOMY* on the results, using recall and the rank weighted score (RWS), and compared the results from both iForest and LOF. We describe in the next section the results obtained from both the raw anomaly scores and active learning. We then discuss insights into why the iForest algorithm performs poorly.



5.3.2 Results

Figure 5.5 shows the recall and RWS curves for both LOF and iForest, where **before active learning** represents the results ranked by the raw anomaly scores and **after active learning** are results ranked by the human retrained score (see section 3.1.6 for details).

Before active learning, a recall of 0% and 8.27% was retrieved after viewing 133 objects (this is the number of anomalies present in the data) from iForest and LOF, respectively. The recall improved to 6.76% and 21.05% after active learning for iForest and LOF, respectively.

Similarly, the RWS curves indicate a significant improvement after active learning for LOF and a slight improvement for iForest. The recall curves on figure 5.5 shows that LOF outperforms iForest before active learning. It also shows that after active learning, LOF has a good performance “out of the box”, i.e., it successfully retrieves more anomalies in the top ranks compared to iForest. However, it levels off after ~ 500 index, and iForest shows an excellent improvement after the same index. The rank weighted score curves shows that iForest performs poorly on average.

We investigate if our feature extractor contributes to the poor performance of iForest by fitting a random forest classification algorithm to the same features used in iForest. Since this is a supervised problem, we split the features into a training and test set. The test set was composed of 20% of the `KN_RRL_sample` and the remaining 80% was used as the training set.

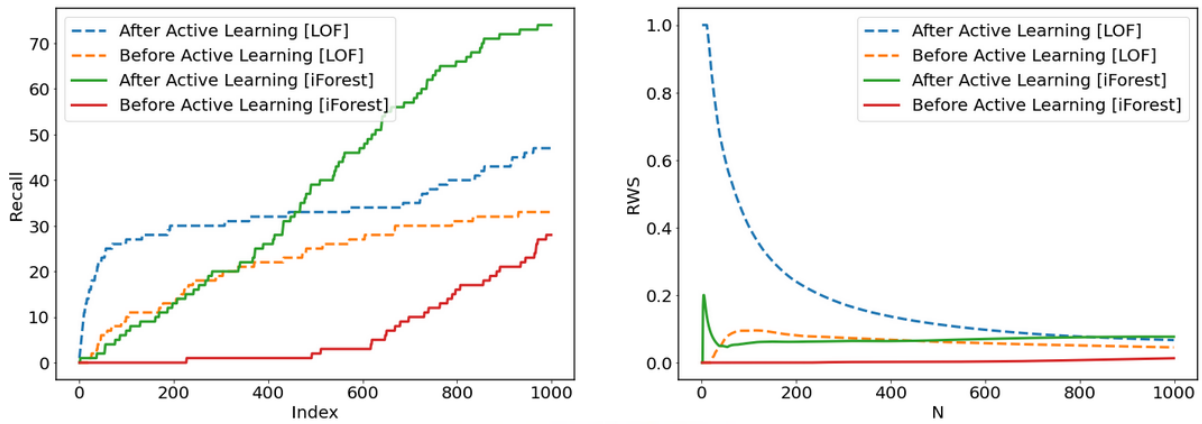


FIGURE 5.5: Recall and RWS for anomalies in the `KN_RRL_sample` before active learning and after active learning. The solid lines indicates results from iForest, while the dashed line are those from LOF.

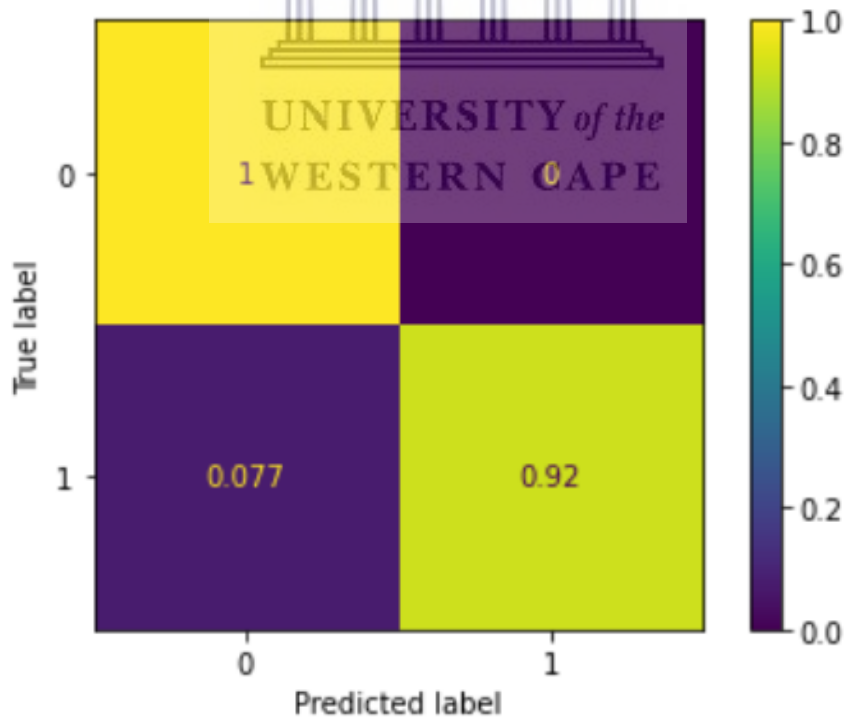
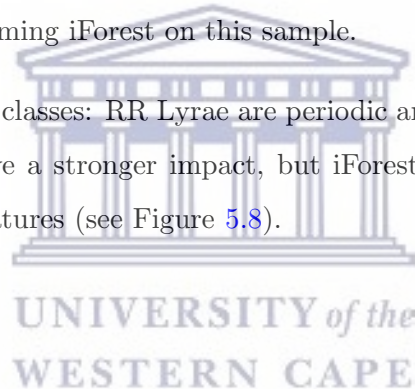


FIGURE 5.6: Confusion matrix from the random forest classifier, class 1 is for the KNe and class 0 is for the RRL.

Surprisingly, the random forest classifier performs well with an accuracy of $\sim 99.9\%$ (see the confusion matrix in Figure 5.6). We further investigate the feature importance from the random forest results. Figure 5.8 shows the top 30 important features. We see from this that the mean in flux measurements from all *griz* bands are the most important features; which is worrying because this implies that the algorithm is learning rates (i.e., it learns that certain objects are brighter than others), and as a result, we might miss interesting objects that are far away. We plot these features and others in figure 5.7 to get insights into the distribution of the features space for the two labels (KN and RR Lyrae).

We found that there is a distinct separation between the two labels, which is why random forest performs well. However, we also found that majority of the anomalous objects (KNe) lie close to normal objects (RR Lyrae) in the features space, i.e., they are not isolated from the “normal” sample. This might be why iForest is performing poorly on this sample. LOF, on the other hand, is built to detect anomalies that lie relatively close to the “normal” sample in feature space (Breunig et al. 2000), which is why it is outperforming iForest on this sample.

Again, given the nature of the two classes: RR Lyrae are periodic and KNe are not, we expected that the periodicity features would have a stronger impact, but iForest sees more anomalies in the flux features than in the periodicity features (see Figure 5.8).



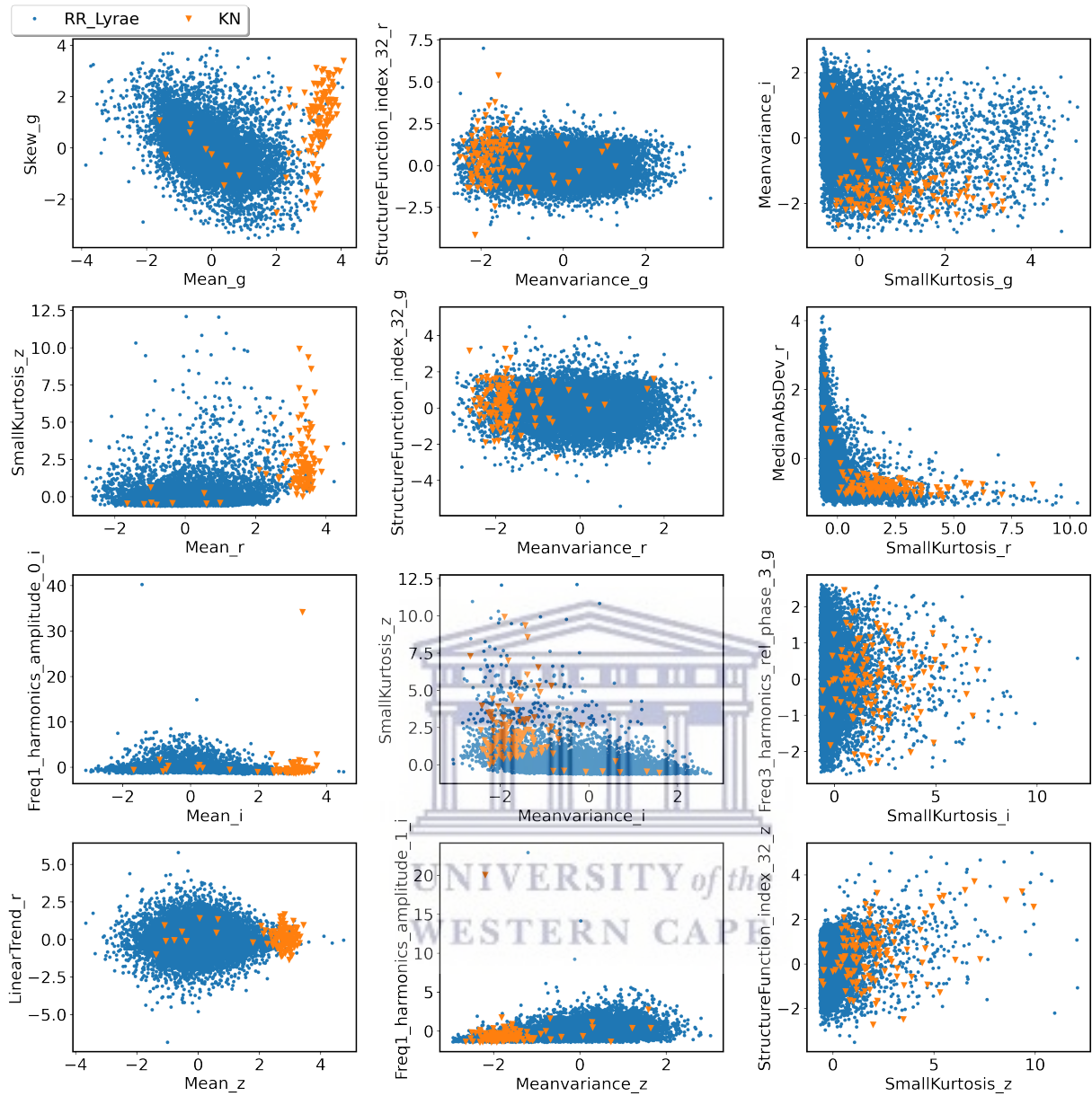


FIGURE 5.7: The scatter plots showing the relationship between some of the important features as by random forest (see figure 5.8) and random features in the data. We can see from left panel and middle panel plots, that there is a distinct difference between the KNe and RR Lyrae labels.

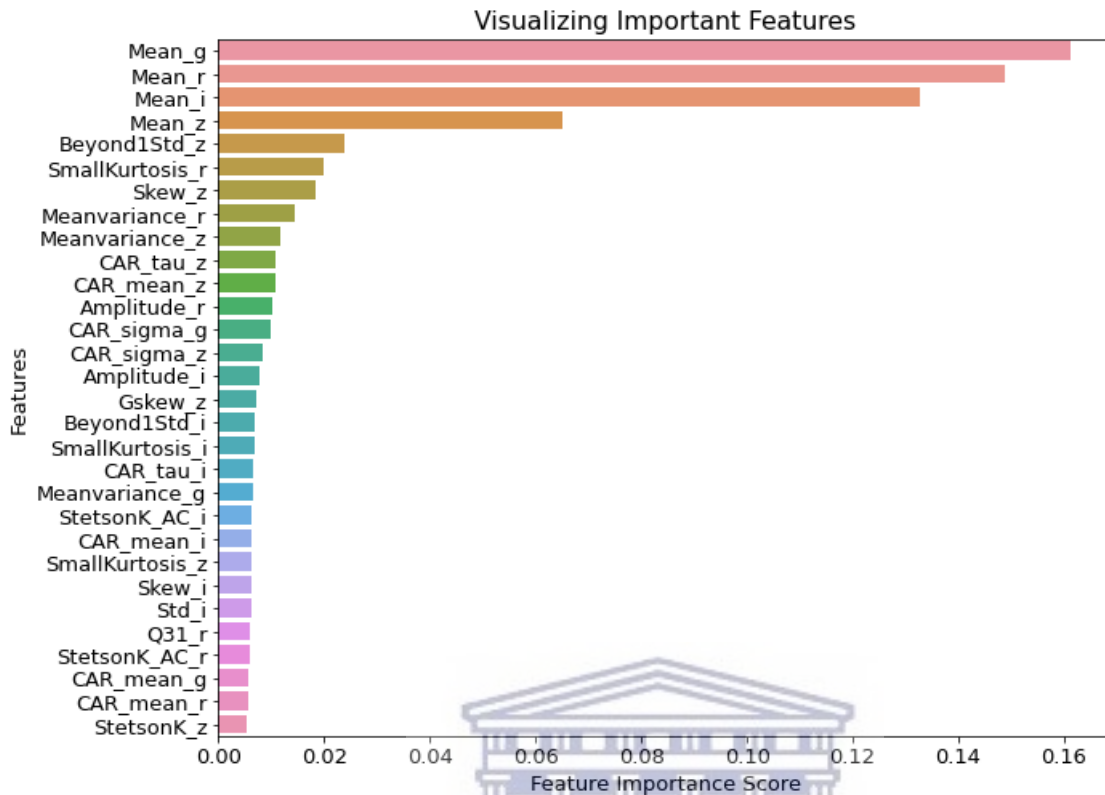


FIGURE 5.8: The top 30 important features as by the random forest classifier. The most important features are the mean in flux measurements in all bands.

5.4 Astronomy applied to a large sample with all classes in the PLAsTiCC Data

We have found in the previous section that the LOF algorithm is the best algorithm to search for anomalies in the `KN_RRL_sample`. In this section, we adopt the same procedure (**method 2**) as that described in section 5.3.1 and apply *ASTRONOMALY* to the `all_labels_sample`. The `all_labels_sample` is composed of 200 anomalies and 19970 normal objects.

We found from our analysis that given the `feets` features, where the anomalies are not particularly distinct from the “normal” objects, LOF outperforms iForest on the `all_labels_sample`; hence we neglect analysis from iForest and focus on those from LOF. The main goal of this section is to assess the performance of *ASTRONOMALY* on a large sample with all the PLAsTiCC labels. We also aim to assess its performance on two feature extractors: `feets` and `avocado`.

5.4.1 The methodology

We extracted 200 features from `feets` (see **method 2** in section 5.3.1) and 28 features from `avocado` (see a full list of features in table 3.2). We refer to features extracted from `feets` and `avocado` as the `feets_sample` and `avocado_sample`, respectively. We employed LOF to assign anomaly scores to each object in the `feets_sample` and `avocado_sample`. We set `n_neighbours = 20`. We then assigned relevance scores to the top 1000 objects sorted by the raw LOF anomaly scores, and ran active learning to get the human retrained scores. We assigned a relevance score of 5 to labels: 991, 992, 993 and 994; and zero to the remaining labels (see table 5.1 for list of the labels).

We then assessed the performance of *ASTRONOMALY* on anomalies detected from both the `feets_sample` and `avocado_sample`.

5.4.2 Results

Figure 5.9 shows the recall and rank weighted scores from the `feets_sample` and `avocado_sample` before and after active learning. A recall of 0.5% and 3.0% were retrieved from the `feets_sample` after viewing 200 objects (number of anomalies in the data) before and after active learning, respectively. The recall has improved slightly for the `avocado_sample`, where the recall was found to be 1.5% and 7.0% before and after active learning, respectively. However, this is poor performance.

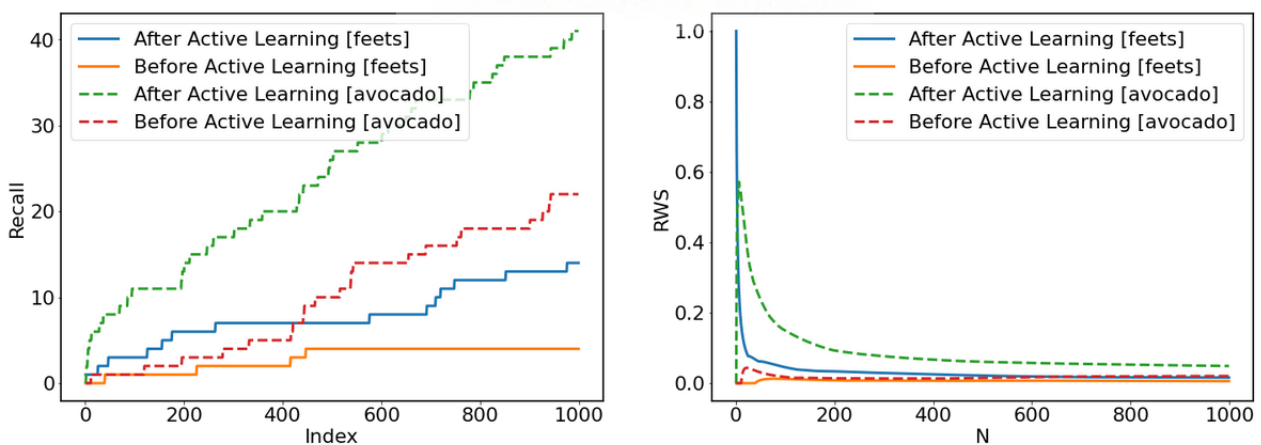
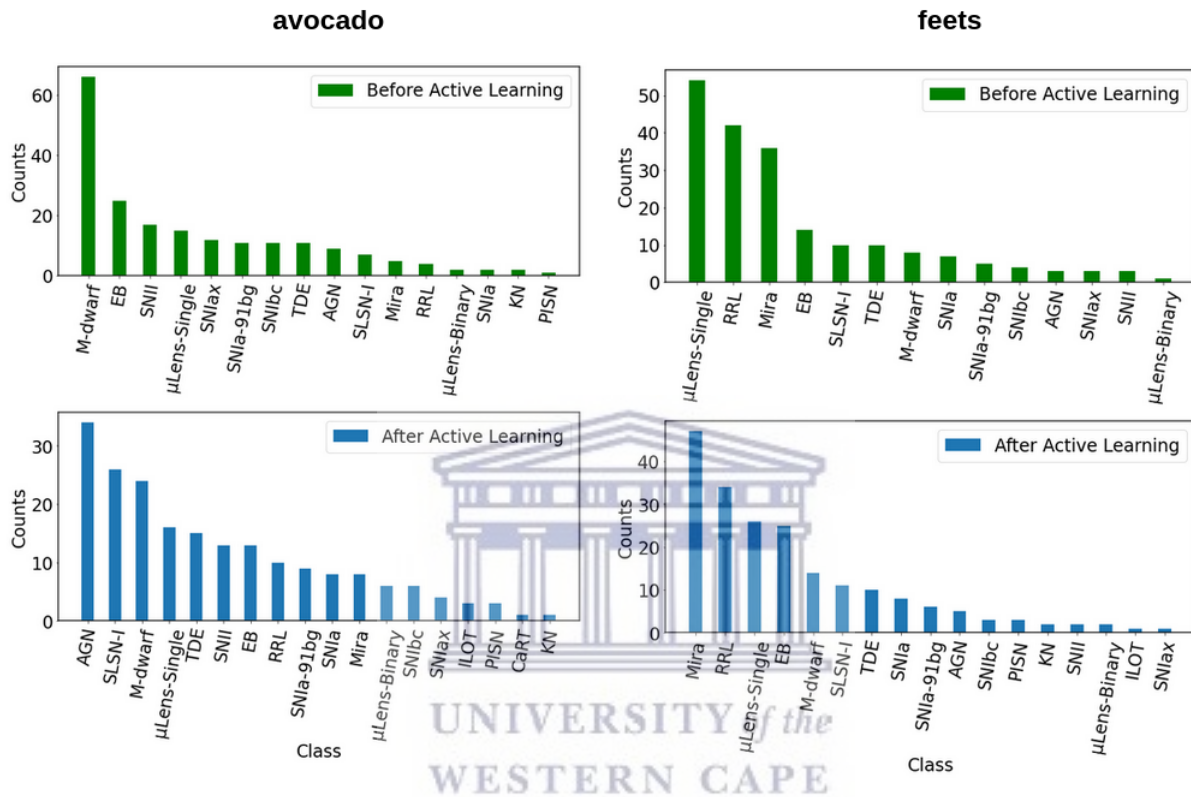


FIGURE 5.9: Same as figure 5.5, but for the `all_labels_sample`. The solid curves indicates results from the `feets_sample`, while the dashed curves are those from the `avocado_sample`

To get insights into why the performance of LOF is poor from both feature samples (`feets_sample` and `avocado_sample`), we plotted a bar graph showing the labels retrieved in the top 200 anomalies before and after active learning in figure 4.9. It is found that $\sim 53.5\%$ of the top anomalies are from labels:

M-dwarf flare star, EB, and SNII, for the `avocado_sample` before active learning; and $\sim 68\%$ are from labels: μ Lens-Single, RRL, and Mira for the `feets_sample`. Two anomalous labels (μ Lens-Binary and PISN) were retrieved from the `avocado_sample` and only one anomalous label (μ Lens-Binary) from `feets_sample`.



Lastly, we find from figure 5.13, that the anomalous objects detected in the top 200 have maximum photometric redshift of 2.07 and 2.87 for the `feets_sample` and `avocado_sample`, respectively. The majority of the anomalies are galactic: with 58.5% and 77.5% detected from the `avocado_sample` and `feets_sample`, respectively.

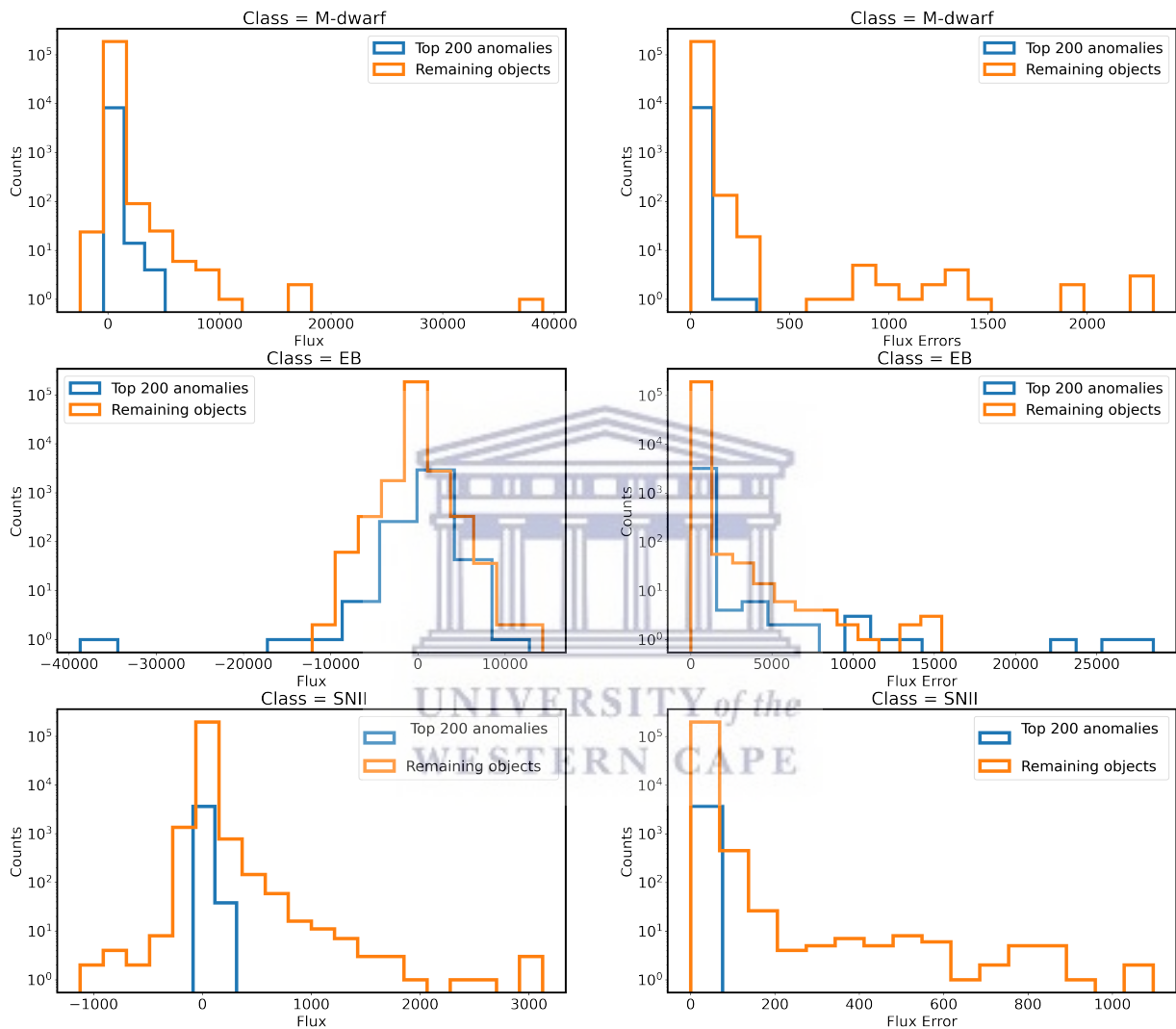


FIGURE 5.11: The flux and flux error distribution of the top 3 anomalous labels from the `avocado_sample`. The left and right panels are the flux and flux error measurements taken directly from the light curve data. The blue lines indicates observations from the top 200 most anomalous objects (as by the raw LOF anomaly scores) within the given label and the orange is for the remaining objects. On average, anomalies detected in avocado are dimmer than those considered normal and they also have small flux errors.

5.5 Discussion

We have applied `ASTRONOMY` on the `KN_RRL_sample` and `all_labels_sample` and outlined the results in sections 5.3 and 5.4. The main goal of the former sample is to find the best algorithm that can be employed to detect anomalies in the `PLAsTiCC` data. We employed two anomaly detection algorithms (iForest and LOF) to assign anomaly scores to object in the `KN_RRL_sample` based on features extracted from `feets`. It was found that the LOF outperforms iForest with a recall of 8.27% and 21.05%. This is because the anomaly objects are not isolated in feature space; rather, they lie close to the “normal” objects (see figure 5.7).

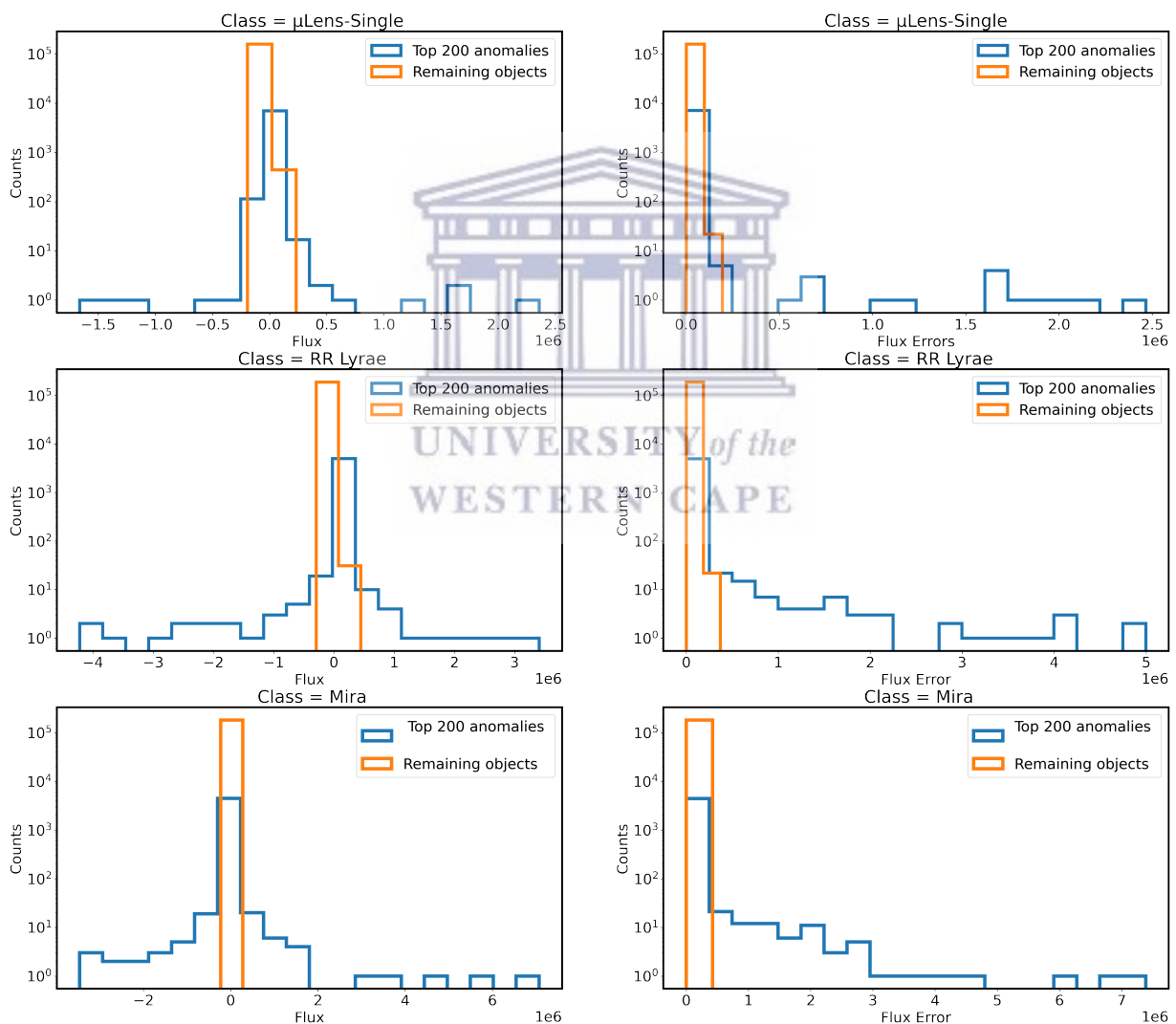


FIGURE 5.12: Same as figure 5.11 but from the `feets_sample`. On average, anomalies detected from the `feets_sample` are bright and they have high flux error values; they also have negative flux values which is a result of difference imaging in `PLAsTiCC`.

Since iForest is built to find objects that are isolated from the “normal” sample, it is expected to perform poorly on the `KN_RRL_sample` because of the reason outlined above. LOF, on the other hand, is built to find anomalies that lie relatively close to the “normal” sample; hence it is performing well on the `KN_RRL_sample`. We also found that active learning improves iForest more than LOF, which indicates that iForest is also doing a useful job, and with a better feature extraction technique, we might find excellent results from it.

Since LOF is found to be the best algorithm to search for anomalies in the PLAsTiCC data, we applied it to a larger sample (`all_labels_sample`). We however, assessed its performance based on two features extraction techniques (`feets_sample` and `avocado_sample`). It was found that LOF performs poorly on both the `avocado_sample` and `feets_sample`, with a recall of 0.5% and 3.0% respectively.

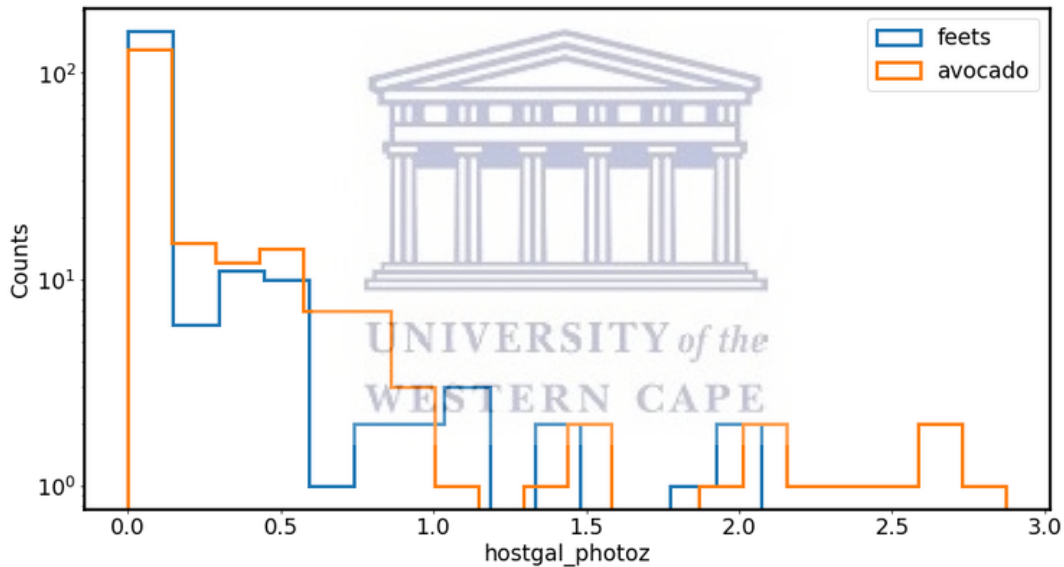


FIGURE 5.13: A histogram showing the host galaxy photometric redshift distribution for the top 200 anomalies detected from both the `feets_sample` (blue lines) and `avocado_sample` (orange lines). The x-axis shows the photometric redshift of the host galaxy, and the y-axis corresponds to the counts. The majority of anomalies detected from both `avocado_sample` and `feets_sample` are galactic (redshift = 0). However, anomalies in `avocado_sample` can be found at high redshifts ~ 2.9 , while those in the `feets_sample` are restricted to a redshift of ~ 2 .

We also found that the the results from the `avocado_sample` outperforms those from `feets_sample`. This is because anomalies detected from the `feets_sample` are biased towards bright Galactic (low redshift) transient and variable objects (see figures 5.8, 5.11, 5.12). However, majority of the anomalous samples (ILOT, CaRT, and PISN) from PLAsTiCC are extra-galactic (with an exception for the μ Lens-Binary label).

The negative flux values and high flux error values detected in the anomalies from the `feets_sample` (see figure 5.12) shows that the `feets` feature extractor is biased towards light curves with low S/N. Hence, it is not an ideal feature extractor tool for anomaly detection problems. Even though the results from the `avocado_sample` are not the best, we found that `avocado` is not biased to either in the redshift or brightness of the object.

5.5.1 Comparing Anomalies Detected from the `feets` and `Avocado` samples

We find from the top panel of figure 5.10, that different objects are detected as anomalies from both the `feets_sample` and `avocado_sample`. For example, the top 3 most abundant labels from the `feets_sample` are the microlensing (μ Lens-Single), RR Lyrae, and Mira events. We have seen from chapter 1 that these events are one of the most commonly studied events. However, due to the Lomb-Scargle features in `feets` that are related to periodic events (see table 3.1), this might be the reason why the RR Lyrae and Mira variables were detected as anomalous because they and eclipsing binaries (EBs), are the only periodic events in the PLAsTiCC. Take note that the EBs class is the fourth most abundant label as seen from figure 5.10.

The most abundant labels of anomalies detected in the `avocado_sample` are M-dwarf, EBs, and type II supernovae (SNII). These events also fall within the most commonly studied transient and variable events in astronomy (see chapter 1). However, M-dwarfs have interesting activities in their light curves, where they display a sudden, short-lived outburst (see figure 1.21 of chapter 1). This might result in a unique Gaussian process fit with a small width compared to the rest of the population. Hence features from `avocado` such as `[positive,negative]_width`, might flag such objects as unique and place them in low-density regions in the feature space. This might be why LOF is detecting most of them as anomalous.

Further investigations are required to get more insights into why LOF is detecting common transient and variable events as anomalies other than the actual anomalies in the data. The bias characteristics observed from the `feets_sample` can be further investigated by normalising the flux values before the feature extraction process.

5.5.2 Scaling Astronomy

The `feets` feature extractor incorporated in `Astronomy`, is fairly computationally expensive and this can be a problem for huge datasets coming from the Rubin observatory. However, the feature

extraction process can be easily parallelised, and it can help speed up the feature extraction process. An alternative approach might be to only do feature extraction on a subset of objects, because some will have poor data quality, and some will be fit very well by other classifiers (which makes anomaly detection on them redundant).



Chapter 6

Conclusions

We have extended an existing public framework, **ASTRONOMALY**, to be able to load, extract and run anomaly detection algorithms on time-series data. **ASTRONOMALY** is a flexible framework designed to search for anomalies in the most common astronomical data type, including images, spectra, and time-series data (with the updates done in this work).

It is coupled with an active learning technique that is used to personalise interesting anomalies detected by the machine learning algorithms. This is because anomalies are mostly triggered by either artefacts or astrophysical processes. An instrumental scientist will find those triggered by artefacts interesting, while an astrophysicist would find the latter interesting.

ASTRONOMALY has a python backend where the data management, feature extraction, data pre-processing, data post-processing, and machine learning [currently operating with the anomaly detection algorithms isolation forest (iForest) and local outlier factor (LOF)] processes occurs. The backend returns anomaly scores for objects in a given data set, and these objects are then visualised on the JavaScript frontend web interface. The frontend is also used for labeling the objects according to how relevant they are to a user. This follows after the objects are ranked according to the raw anomaly scores returned from the frontend. **ASTRONOMALY** can then be retrained on the human labels, and the object can be sorted by the human retrained label. This technique is referred to as active learning.

ASTRONOMALY has been applied to time series data (Webb et al. 2020); however, the implementation was not generalised. In this work, we have extended **ASTRONOMALY** to operate with time-series data (light curves) with two main characteristics: light curves observed in one band and those observed in multiple bands. We designed the light curve reader to be able to load two types of light curve data files: a light curve file with light curves of multiple objects embedded in one large file and multiple

light curve files, each having a light curve of a single object. We also incorporated the `feets` feature extractor in `ASTRONOMALY` to extract features from any given time series data. `feets` is an open feature extractor package designed to extract features from any time series data.

We also adopted features from `avocado`, a feature extractor package designed to extract features from the PLAsTiCC data (a simulated data designed to mimic what is expected from the LSST). However, the features were extracted outside `ASTRONOMALY` (it will be incorporated in the near future).

We applied the updated `ASTRONOMALY` (using `iForest` and `feets` features) on the MANTRA datasets, a data composed of light curve data with real observations from the Catalina real-time transient survey. The goal is to test how effective `ASTRONOMALY` is in detecting anomalies from real observations. We also aim to test if the anomalies detected from the real observations are from the anomalous objects in the literature or are due to artefacts.

It was found that `ASTRONOMALY` detects anomalies triggered by both artefacts and interesting objects. However, most of the detected anomalies (appearing at the top ranks) are those triggered by artefacts. We further tested the active learning technique by assigning a relevance score to the top 100 anomalies sorted by the raw anomaly scores, where bogus light curves were given a low score and interesting objects a high score. We found that active learning successfully flags most of the artefacts from top ranks and retrieves interesting anomalies instead.

This indicates that `ASTRONOMALY` is an ideal tool for personalising interesting anomalies in a given dataset. It can be useful in searching for anomalies in big datasets from upcoming surveys such as the LSST, which will detect ~ 10 million alerts per night.

Even though the results from the MANTRA data are promising, the observing strategies of big surveys such as the Vera C. Rubin Observatory are different from those of the CRTS. This means that we need to test `ASTRONOMALY` on a dataset that closely mimics what is expected from the Rubin Observatory.

To get insights into how `ASTRONOMALY` would perform when applied to data from the Rubin Observatory, we tested it on the PLAsTiCC data. Because we know the ground truth of the anomalous samples in the PLAsTiCC data, we can assess the performance of our algorithms using recall and the rank-weighted score metrics. We tested this on a small sample of the PLAsTiCC data composed of the anomalous KNe labels and “normal” RR Lyrae labels. It was found that the LOF algorithm is the best in detecting anomalies in the small sample, whose features are extracted from `feets`.

Similarly, we tested `ASTRONOMALY` on a large PLAsTiCC sample composed of 18 labels, four of which are anomalous labels. Again, it was found that the LOF algorithm outperforms `iForest`, and returns

the best results when applied to features from `avocado`. It was also found that the anomalies detected from the `feets` feature extractor is biased towards bright and nearby objects with bogus light curves. This is why our algorithms are not performing well with features extracted from them. This indicates the importance of a feature extractor technique: a poor, biased feature extractor is responsible for the poor performance of a machine learning algorithm.

We have shown in this work that anomaly detection techniques coupled with active learning are ideal tools to optimise the anomaly detection process. Although the features used in this work result in poor performance, machine learning still has a great deal to offer to speed up the process of finding anomalies. This will be critical in the data deluge expected from Rubin. Our work also shows that more needs to be done to find an appropriate feature extractor for anomaly detection in time series data from the Rubin Observatory.

Future directions for applying `ASTRONOMALY` to time series data include incorporating other feature extraction techniques such as the Lomb-Scargle periodogram, taking the colour features from `feets` into account when extracting features, and adding a light curve pre-processing step for sigma clipping to remove noise from the light curves.

Anomaly detection techniques with machine learning are unavoidable in the current and upcoming era of big data, as traditional techniques of making new discoveries will be prohibitively slow to keep up with incoming data. As we have seen in this work, anomaly detection algorithms are powerful and successful in detecting anomalies in transient and variable data. However, they cannot distinguish between bogus and interesting anomalies. Employing a human expert to label the anomalies and personalise interesting anomalies using active learning techniques, has proven to be effective (Lochner and Bassett 2020) and promising (through this work). However, more work still need to be done, particularly on feature extraction techniques, to optimise the results before such techniques will be applicable to the high data rates expected from the Rubin Observatory and the SKA.

Bibliography

- Aasi, Junaid, BP Abbott, Richard Abbott, Thomas Abbott, MR Abernathy, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addresso, RX Adhikari, et al. (2015). “Advanced ligo”. In: *Classical and quantum gravity* 32.7, p. 074001.
- Abbott, B. P. et al. (Feb. 2016). “Observation of Gravitational Waves from a Binary Black Hole Merger”. In: 116.6, 061102, p. 061102. DOI: [10.1103/PhysRevLett.116.061102](https://doi.org/10.1103/PhysRevLett.116.061102). arXiv: [1602.03837](https://arxiv.org/abs/1602.03837) [gr-qc].
- Acciari, VA, S Ansoldi, LA Antonelli, A Arbet Engels, K Asano, D Baack, A Babić, A Baquero, U Barres de Almeida, JA Barrio, et al. (2021). “MAGIC observations of the nearby short gamma-ray burst GRB 160821B”. In: *The Astrophysical Journal* 908.1, p. 90.
- Acernese, Fet al, M Agathos, K Agatsuma, D Aisa, N Allemandou, A Allocca, J Amarni, P Astone, G Balestri, G Ballardín, et al. (2014). “Advanced Virgo: a second-generation interferometric gravitational wave detector”. In: *Classical and Quantum Gravity* 32.2, p. 024001.
- Agaskar, Vikrant, Megha Babariya, Shruthi Chandran, and Namrata Giri (2017). “Unsupervised learning for credit card fraud detection”. In: *International Research Journal of Engineering and Technology (IRJET)* 4.3, pp. 2343–2346.
- Aggarwal, Charu C (2015). *Data mining: the textbook*. Springer.
- Alcock, C., R. A. Allsman, D. Alves, T. S. Axelrod, A. C. Becker, D. P. Bennett, K. H. Cook, A. J. Drake, K. C. Freeman, K. Griest, and et al. (Aug. 1999a). “Difference Image Analysis of Galactic Microlensing. I. Data Analysis”. In: *The Astrophysical Journal* 521.2, pp. 602–612. ISSN: 1538-4357. DOI: [10.1086/307567](https://doi.org/10.1086/307567). URL: <http://dx.doi.org/10.1086/307567>.
- Alcock, Charles, Carl W Akerlof, RA Allsman, TS Axelrod, DP Bennett, S Chan, KH Cook, KC Freeman, K Griest, SL Marshall, et al. (1993). “Possible gravitational microlensing of a star in the Large Magellanic Cloud”. In: *nature* 365.6447, pp. 621–623.
- Alcock, Charles, RA Allsman, David R Alves, TS Axelrod, Andrew C Becker, DP Bennett, DF Bersier, Kem H Cook, KC Freeman, K Griest, et al. (1999b). “The MACHO project LMC variable

- star inventory. VIII. The recent star formation history of the Large Magellanic Cloud from the Cepheid period distribution”. In: *The Astronomical Journal* 117.2, p. 920.
- Allam, Tarek, Anita Bahmanyar, Rahul Biswas, Mi Dai, Lluís Galbany, Renée Hložek, Emille E.O. Ishida, Saurabh W Jha, David O Jones, Richard Kessler, Michelle Lochner, Ashish A Mahabal, Alex I Malz, Kaisey S Mandel, Juan Rafael Martínez-Galarza, Jason D. McEwen, Daniel Muthukrishna, Gautham Narayan, Hiranya Peiris, Christina M Peters, Kara Ponder, and Christian N Setzer (2018). *The photometric LSST Astronomical time-series classification challenge (PLAsTiCC): Data set*. arXiv: 1810.00001. URL: <https://lsst-tvssc.github.io/>.
- Ambikasaran, Sivaram, Daniel Foreman-Mackey, Leslie Greengard, David W. Hogg, and Michael O’Neil (2016). “Fast Direct Methods for Gaussian Processes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.2, pp. 252–265. DOI: [10.1109/TPAMI.2015.2448083](https://doi.org/10.1109/TPAMI.2015.2448083).
- Anderson, Joseph P, Santiago González-Gaitán, Mario Hamuy, Claudia P Gutiérrez, Maximilian D Stritzinger, Felipe Olivares, Mark M Phillips, Steve Schulze, Roberto Antezana, Luis Bolt, et al. (2014). “Characterizing the V-band light-curves of hydrogen-rich type II supernovae”. In: *The Astrophysical Journal* 786.1, p. 67.
- Angluin, Dana (1988). “Queries and concept learning”. In: *Machine learning* 2.4, pp. 319–342.
- Aoki, Wako, Timothy C Beers, Young Sun Lee, Satoshi Honda, Hiroko Ito, Masahide Takada-Hidai, Anna Frebel, Takuma Suda, Masayuki Y Fujimoto, Daniela Carollo, et al. (2012). “High-resolution spectroscopy of extremely metal-poor stars from SDSS/SEGUE. I. Atmospheric parameters and chemical compositions”. In: *The Astronomical Journal* 145.1, p. 13.
- Arcavi, Iair, Avishay Gal-Yam, Mark Sullivan, Yen-Chen Pan, S. Bradley Cenko, Assaf Horesh, Eran O. Ofek, Annalisa De Cia, Lin Yan, Chen-Wei Yang, D. A. Howell, David Tal, Shrinivas R. Kulkarni, Shriharsh P. Tendulkar, Sumin Tang, Dong Xu, Assaf Sternberg, Judith G. Cohen, Joshua S. Bloom, Peter E. Nugent, Mansi M. Kasliwal, Daniel A. Perley, Robert M. Quimby, Adam A. Miller, Christopher A. Theissen, and Russ R. Laher (Sept. 2014). “A Continuum of H- to He-rich Tidal Disruption Candidates With a Preference for E+A Galaxies”. In: 793.1, 38, p. 38. DOI: [10.1088/0004-637X/793/1/38](https://doi.org/10.1088/0004-637X/793/1/38). arXiv: 1405.1415 [astro-ph.HE].
- Ascenzi, Stefano, Michael W Coughlin, Tim Dietrich, Ryan J Foley, Enrico Ramirez-Ruiz, Silvia Piranomonte, Brenna Mockler, Ariadna Murguia-Berthier, Chris L Fryer, Nicole M Lloyd-Ronning, et al. (2019). “A luminosity distribution for kilonovae based on short gamma-ray burst afterglows”. In: *Monthly Notices of the Royal Astronomical Society* 486.1, pp. 672–690.
- Atlas, Les E, David A Cohn, and Richard E Ladner (1990). “Training connectionist networks with queries and selective sampling”. In: *Advances in neural information processing systems*. Citeseer, pp. 566–573.

- Aziz, Saqib and Michael Dowling (2019). “Machine learning and AI for risk management”. In: *Disrupting finance*. Palgrave Pivot, Cham, pp. 33–50.
- Bachelet, E, M Norbury, V Bozza, and R Street (2017). “pyLIMA: An Open-source Package for Microlensing Modeling. I. Presentation of the Software and Analysis of Single-lens Models”. In: *The Astronomical Journal* 154.5, p. 203.
- Barkat, Z, G Rakavy, and N Sack (1967). “Dynamics of supernova explosion resulting from pair formation”. In: *Physical Review Letters* 18.10, p. 379.
- Bartos, Imre and Marek Kowalski (2017). *Multimessenger astronomy*. IOP Publishing Bristol.
- Baur, Christoph, Benedikt Wiestler, Shadi Albarqouni, and Nassir Navab (2018). “Deep autoencoding models for unsupervised anomaly segmentation in brain MR images”. In: *International MICCAI Brainlesion Workshop*. Springer, pp. 161–169.
- Belcher, J, JS Hampton, and G Tunnicliffe Wilson (1994). “Parameterization of continuous time autoregressive models for irregularly sampled time series data”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 56.1, pp. 141–155.
- Bellm, Eric C, Shrinivas R Kulkarni, Matthew J Graham, Richard Dekany, Roger M Smith, Reed Riddle, Frank J Masci, George Helou, Thomas A Prince, Scott M Adams, et al. (2018). “The zwicky transient facility: System overview, performance, and first results”. In: *Publications of the Astronomical Society of the Pacific* 131.995, p. 018002.
- Belokurov, Vasily, N Wyn Evans, and Yann Le Du (2004). “Light-curve classification in massive variability surveys—II. Transients towards the Large Magellanic Cloud”. In: *Monthly Notices of the Royal Astronomical Society* 352.1, pp. 233–242.
- Bertin, Emmanuel and Stephane Arnouts (1996). “SExtractor: Software for source extraction”. In: *Astronomy and astrophysics supplement series* 117.2, pp. 393–404.
- Bessell, Michael S (2005). “Standard photometric systems”. In: *Annu. Rev. Astron. Astrophys.* 43, pp. 293–336.
- Betoule, MEA, R Kessler, J Guy, J Mosher, D Hardin, R Biswas, P Astier, P El-Hage, M Konig, S Kuhlmann, et al. (2014). “Improved cosmological constraints from a joint analysis of the SDSS-II and SNLS supernova samples”. In: *Astronomy & Astrophysics* 568, A22.
- Blanton, Michael R, Matthew A Bershady, Bela Abolfathi, Franco D Albareti, Carlos Allende Prieto, Andres Almeida, Javier Alonso-Garcia, Friedrich Anders, Scott F Anderson, Brett Andrews, et al. (2017). “Sloan digital sky survey IV: Mapping the Milky Way, nearby galaxies, and the distant universe”. In: *The Astronomical Journal* 154.1, p. 28.
- Bond, IA, F Abe, RJ Dodd, JB Hearnshaw, M Honda, J Jugaku, PM Kilmartin, A Marles, K Masuda, Y Matsubara, et al. (2001). “Real-time difference imaging analysis of MOA Galactic bulge

- observations during 2000”. In: *Monthly Notices of the Royal Astronomical Society* 327.3, pp. 868–880.
- Boone, Kyle (2019). “Avocado: Photometric classification of astronomical transients with gaussian process augmentation”. In: *The Astronomical Journal* 158.6, p. 257.
- Bopp, Bernard W (1980). “By Draconis and RS Canum Venaticorum Stars: The Discoveries of Classical Photometry and Spectroscopy”. In: *Highlights of Astronomy* 5, pp. 847–848.
- Branch, David, E Baron, and David J Jeffery (2001). “Optical spectra of supernovae”. In: *arXiv preprint astro-ph/0111573*.
- Breedt, E., B. T. Gänsicke, A. J. Drake, P. Rodríguez-Gil, S. G. Parsons, T. R. Marsh, P. Szkody, M. R. Schreiber, and S. G. Djorgovski (Aug. 2014). “1000 cataclysmic variables from the Catalina Real-time Transient Survey”. In: *Monthly Notices of the Royal Astronomical Society* 443.4, pp. 3174–3207. ISSN: 0035-8711. DOI: [10.1093/mnras/stu1377](https://doi.org/10.1093/mnras/stu1377). eprint: <https://academic.oup.com/mnras/article-pdf/443/4/3174/13765784/stu1377.pdf>. URL: <https://doi.org/10.1093/mnras/stu1377>.
- Breiman, Leo (2001). “Random forests”. In: *Machine learning* 45.1, pp. 5–32.
- (2002). “Manual on setting up, using, and understanding random forests v3. 1”. In: *Statistics Department University of California Berkeley, CA, USA* 1.58.
- Breunig, Markus M, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander (2000). “LOF: identifying density-based local outliers”. In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 93–104.
- Brincat, Stephen M., Charles Galdies, and Kevin Hills (Nov. 2020). “Carbon star CGCS 673 identified as a semi-regular variable star”. In: *Research in Astronomy and Astrophysics* 20.11, p. 177. ISSN: 2397-6209. DOI: [10.1088/1674-4527/20/11/177](https://doi.org/10.1088/1674-4527/20/11/177). URL: <http://dx.doi.org/10.1088/1674-4527/20/11/177>.
- Brockwell, PJ and RA Davis (2002). *Introduction to time series and forecasting*. New York: Springer-Verlag.
- Bustamante, Mauricio, Jonas Heinze, Kohta Murase, and Walter Winter (2017). “Multi-messenger light curves from gamma-ray bursts in the internal shock model”. In: *The Astrophysical Journal* 837.1, p. 33.
- Cabral, J B, B Sánchez, F Ramos, S Gurovich, P Granitto, and J Vanderplas (Sept. 2018). “From FATS to feets: Further improvements to an astronomical feature extraction tool based on machine learning”. In: arXiv: [1809.02154](https://arxiv.org/abs/1809.02154). URL: <https://www.r-project.org/%20http://arxiv.org/abs/1809.02154>.

- Cao, Yi, J Johansson, Peter E Nugent, Ariel Goobar, Jakob Nordin, SR Kulkarni, S Bradley Cenko, Ori D Fox, Mansi M Kasliwal, Christoffer Fremling, et al. (2016). “Absence of fast-moving iron in an intermediate type ia supernova between normal and super-chandrasekhar”. In: *The Astrophysical Journal* 823.2, p. 147.
- Catelan, Márcio and Horace A Smith (2014). *Pulsating Stars*. John Wiley & Sons.
- (2015). *Pulsating stars*. John Wiley & Sons.
- Chambers, Kenneth C, EA Magnier, N Metcalfe, HA Flewelling, ME Huber, CZ Waters, L Denneau, PW Draper, D Farrow, DP Finkbeiner, et al. (2016). “The pan-starrs1 surveys”. In: *arXiv preprint arXiv:1612.05560*.
- Chen, Yongyun, Qiusheng Gu, Junhui Fan, Hongyan Zhou, Yefei Yuan, Weimin Gu, Qinwen Wu, Dingrong Xiong, Xiaotong Guo, Nan Ding, et al. (2021). “The Powers of Relativistic Jets Depend on the Spin of Accreting Supermassive Black Holes”. In: *The Astrophysical Journal* 913.2, p. 93.
- Chomiuk, L, R Chornock, Alicia M Soderberg, Edo Berger, RA Chevalier, RJ Foley, ME Huber, G Narayan, A Rest, S Gezari, et al. (2011). “Pan-starrs1 discovery of two ultraluminous supernovae at $z \approx 0.9$ ”. In: *The Astrophysical Journal* 743.2, p. 114.
- Collaboration: Dark Energy Survey, T. Abbott, F. B. Abdalla, J. Aleksić, S. Allam, A. Amara, D. Bacon, E. Balbinot, M. Banerji, K. Bechtol, A. Benoit-Lévy, G. M. Bernstein, E. Bertin, J. Blazek, C. Bonnett, S. Bridle, D. Brooks, R. J. Brunner, E. Buckley-Geer, D. L. Burke, G. B. Caminha, D. Capozzi, J. Carlsen, A. Carnero-Rosell, M. Carollo, M. Carrasco-Kind, J. Carretero, F. J. Castander, L. Clerkin, T. Collett, C. Conselice, M. Croce, C. E. Cunha, C. B. D’Andrea, L. N. da Costa, T. M. Davis, S. Desai, H. T. Diehl, J. P. Dietrich, S. Dodelson, P. Doel, A. Drlica-Wagner, J. Estrada, J. Etherington, A. E. Evrard, J. Fabbri, D. A. Finley, B. Flaugher, R. J. Foley, P. Fosalba, J. Frieman, J. García-Bellido, E. Gaztanaga, D. W. Gerdes, T. Giannantonio, D. A. Goldstein, D. Gruen, R. A. Gruendl, P. Guarnieri, G. Gutierrez, W. Hartley, K. Honscheid, B. Jain, D. J. James, T. Jeltema, S. Jouvel, R. Kessler, A. King, D. Kirk, R. Kron, K. Kuehn, N. Kuropatkin, O. Lahav, T. S. Li, M. Lima, H. Lin, M. A. G. Maia, M. Makler, M. Manera, C. Maraston, J. L. Marshall, P. Martini, R. G. McMahon, P. Melchior, A. Merson, C. J. Miller, R. Miquel, J. J. Mohr, X. Morice-Atkinson, K. Naidoo, E. Neilsen, R. C. Nichol, B. Nord, R. Ogando, F. Ostrovski, A. Palmese, A. Papadopoulos, H. V. Peiris, J. Peoples, W. J. Percival, A. A. Plazas, S. L. Reed, A. Refregier, A. K. Romer, A. Roodman, A. Ross, E. Roza, E. S. Rykoff, I. Sadeh, M. Sako, C. Sánchez, E. Sanchez, B. Santiago, V. Scarpine, M. Schubnell, I. Sevilla-Noarbe, E. Sheldon, M. Smith, R. C. Smith, M. Soares-Santos, F. Sobreira, M. Soumagnac, E. Suchyta, M. Sullivan, M. Swanson, G. Tarle, J. Thaler, D. Thomas, R. C. Thomas, D. Tucker, J. D. Vieira, V. Vikram, A. R. Walker, R. H. Wechsler, J. Weller, W. Wester, L. Whiteway, H. Wilcox, B. Yanny, Y. Zhang, and J. Zuntz (Mar.

- 2016). “The Dark Energy Survey: more than dark energy – an overview”. In: *Monthly Notices of the Royal Astronomical Society* 460.2, pp. 1270–1299. ISSN: 0035-8711. DOI: [10.1093/mnras/stw641](https://doi.org/10.1093/mnras/stw641). eprint: <https://academic.oup.com/mnras/article-pdf/460/2/1270/8117541/stw641.pdf>. URL: <https://doi.org/10.1093/mnras/stw641>.
- Connolly, AJ, AS Szalay, MA Bershady, AL Kinney, and D Calzetti (1994). “Spectral classification of galaxies: an orthogonal approach”. In: *arXiv preprint astro-ph/9411044*.
- Curran, SJ, JP Moss, and YC Perrott (2021). “QSO photometric redshifts using machine learning and neural networks”. In: *Monthly Notices of the Royal Astronomical Society* 503.2, pp. 2639–2650.
- D’Souza, Divya Jennifer and KR Uday Kumar Reddy (2021). “Anomaly Detection for Big Data Using Efficient Techniques: A Review”. In: *Advances in Artificial Intelligence and Data Engineering*, pp. 1067–1080.
- Dainotti, Maria Giovanna, Malgorzata Bogdan, Aditya Narendra, Spencer James Gibson, Blazej Misojedow, Ioannis Liodakis, Agnieszka Pollo, Trevor Nelson, Kamil Wozniak, Zooey Nguyen, et al. (2021). “Predicting the redshift of gamma-ray loud AGNs using supervised machine learning”. In: *arXiv preprint arXiv:2107.10952*.
- Djorgovski, S. G., A. J. Drake, A. A. Mahabal, M. J. Graham, C. Donalek, R. Williams, E. C. Beshore, S. M. Larson, J. Prieto, M. Catelan, E. Christensen, and R. H. McNaught (2011). *The Catalina Real-Time Transient Survey (CRTS)*. arXiv: [1102.5004](https://arxiv.org/abs/1102.5004) [astro-ph.IM].
- Djorgovski, SG, C Baltay, AA Mahabal, AJ Drake, Roy Williams, David Rabinowitz, MJ Graham, Ciro Donalek, Eilat Glikman, Anne Bauer, et al. (2008). “The Palomar-Quest digital synoptic sky survey”. In: *Astronomische Nachrichten: Astronomical Notes* 329.3, pp. 263–265.
- Dong, Guozhu and Huan Liu (2018). *Feature engineering for machine learning and data analytics*. CRC Press.
- Drake, A. J., S. G. Djorgovski, A. Mahabal, E. Beshore, S. Larson, M. J. Graham, R. Williams, E. Christensen, M. Catelan, A. Boattini, A. Gibbs, R. Hill, and R. Kowalski (Sept. 2008). “First Results from the Catalina Real-time Transient Survey”. In: *Astrophysical Journal* 696.1, pp. 870–884. ISSN: 15384357. DOI: [10.1088/0004-637X/696/1/870](https://doi.org/10.1088/0004-637X/696/1/870). arXiv: [0809.1394](https://arxiv.org/abs/0809.1394). URL: <http://arxiv.org/abs/0809.1394%20http://dx.doi.org/10.1088/0004-637X/696/1/870>.
- Drake, AJ, SG Djorgovski, A Mahabal, E Beshore, S Larson, MJ Graham, R Williams, E Christensen, M Catelan, A Boattini, et al. (2009). “First results from the catalina real-time transient survey”. In: *The Astrophysical Journal* 696.1, p. 870.
- Drake, AJ, SG Djorgovski, A Mahabal, JL Prieto, E Beshore, MJ Graham, M Catalan, S Larson, E Christensen, C Donalek, et al. (2011). “The catalina real-time transient survey”. In: *Proceedings of the International Astronomical Union* 7.S285, pp. 306–308.

- Eddington, Arthur S (1988). *The internal constitution of the stars*. Cambridge University Press.
- Ekers, RD and KI Kellermann (2011). “Introduction: Discoveries in astronomy”. In: *Proceedings of the American Philosophical Society*, pp. 129–133.
- Emmanoulopoulos, Dimitrios, Ian M McHardy, and Phil Uttley (2010). “On the use of structure functions to study blazar variability: caveats and problems”. In: *Monthly Notices of the Royal Astronomical Society* 404.2, pp. 931–946.
- Evans, Charles R and Christopher S Kochanek (1989). “The tidal disruption of a star by a massive black hole”. In: *The Astrophysical Journal* 346, pp. L13–L16.
- Eyer, Laurent and Nami Mowlavi (2008). “Variable stars across the observational HR diagram”. In: *Journal of Physics: Conference Series*. Vol. 118. 1. IOP Publishing, p. 012010.
- Filippenko, Alexei V (1997). “Optical spectra of supernovae”. In: *Annual Review of Astronomy and Astrophysics* 35.1, pp. 309–355.
- Filippenko, Alexei V, Michael W Richmond, David Branch, Martin Gaskell, William Herbst, Charles H Ford, Richard R Treffers, Thomas Matheson, Luis C Ho, Arjun Dey, et al. (1992). “The subluminal, spectroscopically peculiar type IA supernova 1991bg in the elliptical galaxy NGC 4374”. In: *The Astronomical Journal* 104, pp. 1543–1556.
- Foley, Ryan J, Peter J Challis, Ryan Chornock, Mohan Ganeshalingam, W Li, GH Marion, Nidia I Morrell, G Pignata, Maximillian D Stritzinger, Jeffrey M Silverman, et al. (2013). “Type Iax supernovae: a new class of stellar explosion”. In: *The Astrophysical Journal* 767.1, p. 57.
- Förster, F, G Cabrera-Vives, E Castillo-Navarrete, PA Estévez, P Sánchez-Sáez, J Arredondo, FE Bauer, R Carrasco-Davis, M Catelan, F Elorrieta, et al. (2021). “The Automatic Learning for the Rapid Classification of Events (ALeRCE) Alert Broker”. In: *The Astronomical Journal* 161.5, p. 242.
- Gabruseva, Tatiana, Sergey Zlobin, and Peter Wang (2020). “Photometric light curves classification with machine learning”. In: *Journal of Astronomical Instrumentation* 9.01, p. 2050005.
- Galbany, Lluís, Mario Hamuy, Mark M Phillips, Nicholas B Suntzeff, José Maza, Thomas De Jaeger, Tania Moraga, Santiago González-Gaitán, Kevin Krisciunas, Nidia I Morrell, et al. (2016). “UBVR_{Iz} light curves of 51 type II supernovae”. In: *The Astronomical Journal* 151.2, p. 33.
- Gehrels, N, E Chipman, and D Kniffen (1994). “The compton gamma ray observatory”. In: *The Astrophysical Journal Supplement Series* 92, pp. 351–362.
- Gezari, Suvi (Sept. 2021). “Tidal Disruption Events”. In: *Annual Review of Astronomy and Astrophysics* 59.1, pp. 21–58. ISSN: 1545-4282. DOI: [10.1146/annurev-astro-111720-030029](https://doi.org/10.1146/annurev-astro-111720-030029). URL: <http://dx.doi.org/10.1146/annurev-astro-111720-030029>.

- Gibson, NP, D Pollacco, EK Simpson, S Barros, YC Joshi, I Todd, FP Keenan, I Skillen, C Benn, D Christian, et al. (2009). “A transit timing analysis of nine rise light curves of the exoplanet system TrES-3”. In: *The Astrophysical Journal* 700.2, p. 1078.
- Gnana, D Asir Antony, S Appavu Alias Balamurugan, and E Jebamalar Leavline (2016). “Literature review on feature selection methods for high-dimensional data”. In: *International Journal of Computer Applications* 136.1, pp. 9–17.
- Goobar, Ariel and Bruno Leibundgut (2011). “Supernova cosmology: legacy and future”. In: *Annual Review of Nuclear and Particle Science* 61, pp. 251–279.
- Graham, Matthew J, SR Kulkarni, Eric C Bellm, Scott M Adams, Cristina Barbarino, Nadejda Blagorodnova, Dennis Bodewits, Bryce Bolin, Patrick R Brady, S Bradley Cenko, et al. (2019). “The zwicky transient facility: Science objectives”. In: *Publications of the Astronomical Society of the Pacific* 131.1001, p. 078001.
- Guillochon, James, Matt Nicholl, V Ashley Villar, Brenna Mockler, Gautham Narayan, Kaisey S Mandel, Edo Berger, and Peter KG Williams (2018). “MOSFiT: modular open source fitter for transients”. In: *The Astrophysical Journal Supplement Series* 236.1, p. 6.
- Guillochon, James, Jerod Parrent, Luke Zoltan Kelley, and Raffaella Margutti (2017). “An open catalog for supernova data”. In: *The Astrophysical Journal* 835.1, p. 64.
- Guo, Xinyi, Ann Esin, Rosanne Di Stefano, and Jeffrey Taylor (2015). “Periodic Signals in Binary Microlensing Events”. In: *The Astrophysical Journal* 809.2, p. 182.
- Guy, Julien, P Astier, S Baumont, D Hardin, R Pain, N Regnault, S Basa, RG Carlberg, A Conley, S Fabbro, et al. (2007). “SALT2: using distant supernovae to improve the use of type Ia supernovae as distance indicators”. In: *Astronomy & Astrophysics* 466.1, pp. 11–21.
- Hansen, Brad MS (1998). “Old and blue white-dwarf stars as a detectable source of microlensing events”. In: *Nature* 394.6696, pp. 860–862.
- Haque, Shah Ahsanul, Mustafizur Rahman, and Syed Mahfuzul Aziz (2015). “Sensor anomaly detection in wireless sensor networks for healthcare”. In: *Sensors* 15.4, pp. 8764–8786.
- Hauptmann, Alexander G, Wei-Hao Lin, Rong Yan, Jun Yang, and Ming-Yu Chen (2006). “Extreme video retrieval: joint maximization of human and computer performance”. In: *Proceedings of the 14th ACM international conference on Multimedia*, pp. 385–394.
- Hawley, Suzanne L, James RA Davenport, Adam F Kowalski, John P Wisniewski, Leslie Hebb, Russell Deitrick, and Eric J Hilton (2014). “Kepler flares. I. Active and inactive M dwarfs”. In: *The Astrophysical Journal* 797.2, p. 121.
- Herbst, William (2012). “The Variability of Young Stellar Objects”. In: *Journal of the American Association of Variable Star Observers (JAAVSO)* 40.1, p. 448.

- Hertzsprung, Ejnar (Nov. 1913). “Über die räumliche Verteilung der Veränderlichen vom δ Cephei-Typus”. In: *Astronomische Nachrichten* 196, p. 201.
- Hocking, Alex, James E Geach, Yi Sun, and Neil Davey (2018). “An automatic taxonomy of galaxy morphology using unsupervised machine learning”. In: *Monthly Notices of the Royal Astronomical Society* 473.1, pp. 1108–1129.
- Hoffleit, Dorrit (Jan. 1997). “History of the Discovery of Mira Stars”. In: 25.2, pp. 115–136.
- Hopkins, Philip F, Lars Hernquist, Thomas J Cox, Tiziana Di Matteo, Brant Robertson, and Volker Springel (2006). “A unified, merger-driven model of the origin of starbursts, quasars, the cosmic X-ray background, supermassive black holes, and galaxy spheroids”. In: *The Astrophysical Journal Supplement Series* 163.1, p. 1.
- Hosenie, Zafirah, Robert J Lyon, Benjamin W Stappers, and Arrykrishna Mootoovaloo (2019). “Comparing multiclass, binary, and hierarchical machine learning classification schemes for variable stars”. In: *Monthly Notices of the Royal Astronomical Society* 488.4, pp. 4858–4872.
- Hosseinzadeh, Griffin, Frederick Dauphin, V Ashley Villar, Edo Berger, David O Jones, Peter Challis, Ryan Chornock, Maria R Drout, Ryan J Foley, Robert P Kirshner, et al. (2020). “Photometric Classification of 2315 Pan-STARRS1 Supernovae with Superphot”. In: *The Astrophysical Journal* 905.2, p. 93.
- Hotelling, Harold (1933). “Analysis of a complex of statistical variables into principal components.” In: *Journal of educational psychology* 24.6, p. 417.
- Hubble, E. P. (Dec. 1926). “Extragalactic nebulae.” In: 64, pp. 321–369. DOI: [10.1086/143018](https://doi.org/10.1086/143018).
- Hubble, Edwin (Mar. 1929). “A Relation between Distance and Radial Velocity among Extra-Galactic Nebulae”. In: *Proceedings of the National Academy of Science* 15.3, pp. 168–173. DOI: [10.1073/pnas.15.3.168](https://doi.org/10.1073/pnas.15.3.168).
- Huijse, Pablo, Pablo A Estevez, Pavlos Protopapas, Pablo Zegers, and Jose C Principe (2012). “An information theoretic algorithm for finding periodicities in stellar light curves”. In: *IEEE Transactions on Signal Processing* 60.10, pp. 5135–5145.
- Hung, T, S Gezari, N Blagorodnova, N Roth, SB Cenko, SR Kulkarni, A Horesh, I Arcavi, C McCully, Lin Yan, et al. (2017). “Revisiting optical tidal disruption events with iPTF16axa”. In: *The Astrophysical Journal* 842.1, p. 29.
- Hutchinson, Timothy A, Adam S Bolton, Kyle S Dawson, Carlos Allende Prieto, Stephen Bailey, Julian E Bautista, Joel R Brownstein, Charlie Conroy, Julien Guy, Adam D Myers, et al. (2016). “Redshift measurement and spectral classification for eboss galaxies with the redmonster software”. In: *The Astronomical Journal* 152.6, p. 205.

- Ireland, Michael J, Michael Scholz, and Peter R Wood (2008). “Dynamical opacity-sampling models of Mira variables–I. Modelling description and analysis of approximations”. In: *Monthly Notices of the Royal Astronomical Society* 391.4, pp. 1994–2002.
- (2011). “Dynamical opacity-sampling models of Mira variables–II. Time-dependent atmospheric structure and observable properties of four M-type model series”. In: *Monthly Notices of the Royal Astronomical Society* 418.1, pp. 114–128.
- Ivezić, Željko, Steven M Kahn, J Anthony Tyson, Bob Abel, Emily Acosta, Robyn Allsman, David Alonso, Yusra AlSayyad, Scott F Anderson, John Andrew, et al. (2019). “LSST: from science drivers to reference design and anticipated data products”. In: *The Astrophysical Journal* 873.2, p. 111.
- Jarvis, M, P Hatfield, and I Almosallam (2020). “Augmenting machine learning photometric redshifts with Gaussian mixture models”. In.
- Jiang, Brighten, Shuai Jiang, and V Ashley Villar (2020). “Extended Self-similar Solution for Circumstellar Material-Supernova Ejecta Interaction”. In: *arXiv preprint arXiv:2008.10397*.
- Jiang, Ning, Liming Dou, Tinggui Wang, Chenwei Yang, Jianwei Lyu, and Hongyan Zhou (2016). “The WISE detection of an infrared echo in tidal disruption event ASASSN-14li”. In: *The Astrophysical Journal Letters* 828.1, p. L14.
- Jolliffe, Ian T and Jorge Cadima (2016). “Principal component analysis: a review and recent developments”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065, p. 20150202.
- Jung, Yoonsuh (2018). “Multiple predicting K-fold cross-validation for model selection”. In: *Journal of Nonparametric Statistics* 30.1, pp. 197–215.
- Kasen, Daniel, Brian Metzger, Jennifer Barnes, Eliot Quataert, and Enrico Ramirez-Ruiz (2017). “Origin of the heavy elements in binary neutron-star mergers from a gravitational-wave event”. In: *Nature* 551.7678, pp. 80–84.
- Kasen, Daniel, SE Woosley, and Alexander Heger (2011). “Pair instability supernovae: light curves, spectra, and shock breakout”. In: *The Astrophysical Journal* 734.2, p. 102.
- Kessler, R., G. Narayan, A. Avelino, E. Bachelet, R. Biswas, P. J. Brown, D. F. Chernoff, A. J. Connolly, M. Dai, S. Daniel, R. Di Stefano, M. R. Drout, L. Galbany, S. González-Gaitán, M. L. Graham, R. Hložek, E. E.O. Ishida, J. Guillochon, S. W. Jha, D. O. Jones, K. S. Mandel, D. Muthukrishna, A. O’grady, C. M. Peters, J. R. Pierel, K. A. Ponder, A. Prša, S. Rodney, and V. A. Villar (2019). “Models and simulations for the photometric lsst astronomical time series classification challenge (Plasticc)”. In: *Publications of the Astronomical Society of the Pacific* 131.1003. ISSN: 00046280. DOI: [10.1088/1538-3873/ab26f1](https://doi.org/10.1088/1538-3873/ab26f1). arXiv: [1903.11756](https://arxiv.org/abs/1903.11756).

- Kessler, Richard, Alex Conley, Saurabh Jha, and Stephen Kuhlmann (2010). “Supernova photometric classification challenge”. In: *arXiv preprint arXiv:1001.5210*.
- Khalil, Mahmoud, Elena Fantino, and Panos Liatsis (2019). “Classification of Space Objects Using Machine Learning Methods”. In: *2019 IEEE First International Conference on Cognitive Machine Intelligence (CogMI)*. IEEE, pp. 93–96.
- Kim, Dae Won, Pavlos Protopapas, Charles Alcock, Yong Ik Byun, and Federica B Bianco (2009). “Detrending time series for astronomical variability surveys”. In: *Monthly Notices of the Royal Astronomical Society* 397.1, pp. 558–568.
- Kim, Dae Won, Pavlos Protopapas, Coryn AL Bailer-Jones, Yong-Ik Byun, Seo-Won Chang, Jean-Baptiste Marquette, and Min-Su Shin (2014). “The EPOCH Project-I. Periodic variable stars in the EROS-2 LMC database”. In: *Astronomy & Astrophysics* 566, A43.
- Kim, Dae Won, Pavlos Protopapas, Yong Ik Byun, Charles Alcock, Roni Khardon, and Markos Trichas (2011). “Quasi-stellar object selection algorithm using time variability and machine learning: Selection of 1620 quasi-stellar object candidates from MACHO Large Magellanic Cloud database”. In: *The Astrophysical Journal* 735.2, p. 68.
- Krause, Oliver, Masaomi Tanaka, Tomonori Usuda, Takashi Hattori, Miwa Goto, Stephan Birkmann, and Ken’ichi Nomoto (Dec. 2008). “Tycho Brahe’s 1572 supernova as a standard typeIa as revealed by its light-echo spectrum”. In: *Nature* 456.7222, pp. 617–619. ISSN: 1476-4687. DOI: [10.1038/nature07608](https://doi.org/10.1038/nature07608). URL: <http://dx.doi.org/10.1038/nature07608>.
- Kutner, Marc L (2003). *Astronomy: A physical perspective*. cambridge university press.
- Leavitt, Henrietta S (1908). “variables in the Magellanic Clouds”. In: *Annals of Harvard College Observatory* 60.1777, p. 87.
- Lee, Jinhee and Inseok Song (2019). “Evaluation of nearby young moving groups based on unsupervised machine learning”. In: *Monthly Notices of the Royal Astronomical Society* 489.2, pp. 2189–2194.
- Lesser, Michael (Nov. 2015). “A Summary of Charge-Coupled Devices for Astronomy”. In: *Publications of the Astronomical Society of the Pacific* 127.957, pp. 1097–1104. DOI: [10.1086/684054](https://doi.org/10.1086/684054). URL: <https://doi.org/10.1086/684054>.
- Lewis, David D and William A Gale (1994). “A sequential algorithm for training text classifiers”. In: *SIGIR’94*. Springer, pp. 3–12.
- Liaw, Andy, Matthew Wiener, et al. (2002). “Classification and regression by randomForest”. In: *R news* 2.3, pp. 18–22.
- Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou (2008). “Isolation forest”. In: *2008 eighth ieee international conference on data mining*. IEEE, pp. 413–422.

- Lochner, Michelle and Bruce A Bassett (2020). “Astronomy: Personalised Active Anomaly Detection in Astronomical Data”. In: arXiv: 2010.11202. URL: <https://github.com/MichelleLochner/astronomy.%20http://arxiv.org/abs/2010.11202>.
- Lochner, Michelle, Jason D McEwen, Hiranya V Peiris, Ofer Lahav, and Max K Winter (2016). “Photometric supernova classification with machine learning”. In: *The Astrophysical Journal Supplement Series* 225.2, p. 31.
- Lomb, Nicholas R (1976). “Least-squares frequency analysis of unequally spaced data”. In: *Astrophysics and space science* 39.2, pp. 447–462.
- LSST Science Collaboration, LSST Science, Paul A. Abell, Julius Allison, Scott F. Anderson, John R. Andrew, J. Roger P. Angel, Lee Armus, David Arnett, S. J. Asztalos, Tim S. Axelrod, Stephen Bailey, D. R. Ballantyne, Justin R. Bankert, Wayne A. Barkhouse, Jeffrey D. Barr, L. Felipe Barrientos, Aaron J. Barth, James G. Bartlett, Andrew C. Becker, Jacek Becla, Timothy C. Beers, Joseph P. Bernstein, Rahul Biswas, Michael R. Blanton, Joshua S. Bloom, John J. Bochanski, Pat Boeshaar, Kirk D. Borne, Marusa Bradac, W. N. Brandt, Carrie R. Bridge, Michael E. Brown, Robert J. Brunner, James S. Bullock, Adam J. Burgasser, James H. Burge, David L. Burke, Phillip A. Cargile, Srinivasan Chandrasekharan, George Chartas, Steven R. Chesley, You-Hua Chu, David Cinabro, Mark W. Claire, Charles F. Claver, Douglas Clowe, A. J. Connolly, Kem H. Cook, Jeff Cooke, Asantha Cooray, Kevin R. Covey, Christopher S. Culliton, Roelof de Jong, Willem H. de Vries, Victor P. Debattista, Francisco Delgado, Ian P. Dell’Antonio, Saurav Dhital, Rosanne Di Stefano, Mark Dickinson, Benjamin Dilday, S. G. Djorgovski, Gregory Dobler, Ciro Donalek, Gregory Dubois-Felsmann, Josef Durech, Ardis Eliasdottir, Michael Eracleous, Laurent Eyer, Emilio E. Falco, Xiaohui Fan, Christopher D. Fassnacht, Harry C. Ferguson, Yanga R. Fernandez, Brian D. Fields, Douglas Finkbeiner, Eduardo E. Figuerao, Derek B. Fox, Harold Francke, James S. Frank, Josh Frieman, Sebastien Fromenteau, Muhammad Furqan, Gaspar Galaz, A. Gal-Yam, Peter Garnavich, Eric Gawiser, John Geary, Perry Gee, Robert R. Gibson, Kirk Gilmore, Emily A. Grace, Richard F. Green, William J. Gressler, Carl J. Grillmair, Salman Habib, J. S. Haggerty, Mario Hamuy, Alan W. Harris, Suzanne L. Hawley, Alan F. Heavens, Leslie Hebb, Todd J. Henry, Edward Hileman, Eric J. Hilton, Keri Hoadley, J. B. Holberg, Matt J. Holman, Steve B. Howell, Leopoldo Infante, Zeljko Ivezic, Suzanne H. Jacoby, Bhuvnesh Jain, R. Jedicke, M. James Jee, J. Garrett Jernigan, Saurabh W. Jha, Kathryn V. Johnston, R. Lynne Jones, Mario Juric, Mikko Kaasalainen, Styliani, Kafka, Steven M. Kahn, Nathan A. Kaib, Jason Kalirai, Jeff Kantor, Mansi M. Kasliwal, Charles R. Keeton, Richard Kessler, Zoran Knezevic, Adam Kowalski, Victor L. Krabbendam, K. Simon Krughoff, Shrinivas Kulkarni, Stephen Kuhlman, Mark Lacy, Sebastien Lepine, Ming Liang, Amy Lien, Paulina Lira, Knox S. Long, Suzanne Lorenz, Jennifer

- M. Lotz, R. H. Lupton, Julie Lutz, Lucas M. Macri, Ashish A. Mahabal, Rachel Mandelbaum, Phil Marshall, Morgan May, Peregrine M. McGehee, Brian T. Meadows, Alan Meert, Andrea Milani, Christopher J. Miller, Michelle Miller, David Mills, Dante Minniti, David Monet, Anjum S. Mukadam, Ehud Nakar, Douglas R. Neill, Jeffrey A. Newman, Sergei Nikolaev, Martin Nordby, Paul O'Connor, Masamune Oguri, John Oliver, Scot S. Olivier, Julia K. Olsen, Knut Olsen, Edward W. Olszewski, Hakeem Oluseyi, Nelson D. Padilla, Alex Parker, Joshua Pepper, John R. Peterson, Catherine Petry, Philip A. Pinto, James L. Pizagno, Bogdan Popescu, Andrej Prsa, Veljko Radcka, M. Jordan Raddick, Andrew Rasmussen, Arne Rau, Jeonghee Rho, James E. Rhoads, Gordon T. Richards, Stephen T. Ridgway, Brant E. Robertson, Rok Roskar, Abhijit Saha, Ata Sarajedini, Evan Scannapieco, Terry Schalk, Rafe Schindler, Samuel Schmidt, Sarah Schmidt, Donald P. Schneider, German Schumacher, Ryan Scranton, Jacques Sebag, Lynn G. Seppala, Ohad Shemmer, Joshua D. Simon, M. Sivertz, Howard A. Smith, J. Allyn Smith, Nathan Smith, Anna H. Spitz, Adam Stanford, Keivan G. Stassun, Jay Strader, Michael A. Strauss, Christopher W. Stubbs, Donald W. Sweeney, Alex Szalay, Paula Szkody, Masahiro Takada, Paul Thorman, David E. Trilling, Virginia Trimble, Anthony Tyson, Richard Van Berg, Daniel Vanden Berk, Jake VanderPlas, Licia Verde, Bojan Vrsnak, Lucianne M. Walkowicz, Benjamin D. Wandelt, Sheng Wang, Yun Wang, Michael Warner, Risa H. Wechsler, Andrew A. West, Oliver Wiecha, Benjamin F. Williams, Beth Willman, David Wittman, Sidney C. Wolff, W. Michael Wood-Vasey, Przemek Wozniak, Patrick Young, Andrew Zentner, and Hu Zhan (Dec. 2009). "LSST Science Book, Version 2.0". In: arXiv: [0912.0201](https://arxiv.org/abs/0912.0201). URL: <http://arxiv.org/abs/0912.0201>.
- Lunnan, Ragnhild, Mansi M Kasliwal, Yea Cao, Laura Hangard, Ofer Yaron, JT Parrent, C McCully, Avishay Gal-Yam, JS Mulchaey, Sagi Ben-Ami, et al. (2017). "Two new calcium-rich gap transients in group and cluster environments". In: *The Astrophysical Journal* 836.1, p. 60.
- Malanchev, KL, MV Pruzhinskaya, VS Korolev, PD Aleo, MV Kornilov, EEO Ishida, VV Krushinsky, F Mondon, S Sreejith, AA Volnova, et al. (2021). "Anomaly detection in the Zwicky Transient Facility DR3". In: *Monthly Notices of the Royal Astronomical Society* 502.4, pp. 5147–5175.
- Maneewongvatana, Songrit and David M Mount (2002). "Analysis of approximate nearest neighbor searching with clustered point sets". In: *Data Structures, Near Neighbor Searches, and Methodology* 59, pp. 105–123.
- Marigo, P., L. Girardi, A. Bressan, M. A. T. Groenewegen, L. Silva, and G. L. Granato (Mar. 2008). "Evolution of asymptotic giant branch stars". In: *Astronomy Astrophysics* 482.3, pp. 883–905. ISSN: 1432-0746. DOI: [10.1051/0004-6361:20078467](https://doi.org/10.1051/0004-6361:20078467). URL: <http://dx.doi.org/10.1051/0004-6361:20078467>.

- Martin, Garreth, Sugata Kaviraj, Alex Hocking, Shaun C Read, and James E Geach (2020). “Galaxy morphological classification in deep-wide surveys via unsupervised machine learning”. In: *Monthly Notices of the Royal Astronomical Society* 491.1, pp. 1408–1426.
- Martínez-Galarza, Juan Rafael, Federica Bianco, Dennis Crake, Kushal Tirumala, Ashish A Mahabal, Matthew J Graham, and Daniel Giles (2020). “Where is Waldo (and his friends)? A comparison of anomaly detection algorithms for time-domain astronomy”. In: *MNRAS* 000, pp. 1–21. arXiv: 2009.06760. URL: <https://github.com/fedhere/DtUhackOutliers%20http://arxiv.org/abs/2009.06760>.
- Mauron, N, KS Gigoyan, and TR Kendall (2007). “Cool carbon stars in the halo: new very red or distant objects”. In: *Astronomy & Astrophysics* 475.3, pp. 843–849.
- McCarthy, Dennis D (1998). “The julian and modified julian dates”. In: *Journal for the History of Astronomy* 29.4, pp. 327–330.
- McInnes, L, J Healy, and S Astels (2017). *hdbscan: Hierarchical density based clustering*. *J Open Source Softw* 2: 205.
- McInnes, Leland, John Healy, and James Melville (2018). “Umap: Uniform manifold approximation and projection for dimension reduction”. In: *arXiv preprint arXiv:1802.03426*.
- McKinney, Wes et al. (2011). “pandas: a foundational Python library for data analysis and statistics”. In: *Python for high performance and scientific computing* 14.9, pp. 1–9.
- McLachlan, Geoffrey J, Sharon X Lee, and Suren I Rathnayake (2019). “Finite mixture models”. In: *Annual review of statistics and its application* 6, pp. 355–378.
- Mighell, Kenneth J (1999). “Algorithms for CCD stellar photometry”. In: *Astronomical Data Analysis Software and Systems VIII*. Vol. 172, p. 317.
- Mitchell, T. M. (1997). *Machine Learning*. New York, NY, USA: McGraw-Hill.
- Möller, A, V Ruhlmann-Kleider, C Leloup, J Neveu, N Palanque-Delabrouille, J Rich, R Carlberg, C Lidman, and C Pritchard (2016). “Photometric classification of type Ia supernovae in the Supernova Legacy Survey with supervised learning”. In: *Journal of Cosmology and Astroparticle Physics* 2016.12, p. 008.
- Monamo, Patrick, Vukosi Marivate, and Bheki Twala (2016). “Unsupervised learning for robust Bitcoin fraud detection”. In: *2016 Information Security for South Africa (ISSA)*. IEEE, pp. 129–134.
- Muthukrishna, Daniel, Kaisey S. Mandel, Michelle Lochner, Sara Webb, and Gautham Narayan (2021). *Real-time detection of anomalies in large-scale transient surveys*. arXiv: 2111.00036 [astro-ph.IM].
- Nakar, Ehud (2007). “Short-hard gamma-ray bursts”. In: *Physics Reports* 442.1-6, pp. 166–236.
- NASA (2013). *Gamma-ray Burst*. Last accessed 28 September 2021. URL: <https://imagine.gsfc.nasa.gov/science/objects/bursts1.html>.

- Neira, Mauricio, Catalina Gómez, John F Suárez-Pérez, Diego A Gómez, Juan Pablo Reyes, Marcela Hernández Hoyos, Pablo Arbeláez, and Jaime E Forero-Romero (June 2020). “MANTRA: A Machine Learning reference lightcurve dataset for astronomical transient event recognition”. In: ML arXiv: 2006.13163. URL: <http://arxiv.org/abs/2006.13163>.
- Netzer, Hagai (2015). “Revisiting the unified model of active galactic nuclei”. In: *Annual Review of Astronomy and Astrophysics* 53, pp. 365–408.
- Nun, Isadora, Karim Pichara, Pavlos Protopapas, and Dae-Won Kim (2014). “Supervised detection of anomalous light curves in massive astronomical catalogs”. In: *The Astrophysical Journal* 793.1, p. 23.
- Nun, Isadora, Pavlos Protopapas, Brandon Sim, Ming Zhu, Rahul Dave, Nicolas Castro, and Karim Pichara (2015a). “FATS: Feature Analysis for Time Series”. In: arXiv: 1506.00010. URL: <http://isadoranun.github.io%20http://arxiv.org/abs/1506.00010>.
- (2015b). “Fats: Feature analysis for time series”. In: *arXiv preprint arXiv:1506.00010*.
- Omar, Salima, Asri Ngadi, and Hamid H Jebur (2013). “Machine learning techniques for anomaly detection: an overview”. In: *International Journal of Computer Applications* 79.2.
- Ostlie, Dale A and Bradley W Carroll (2007). *An introduction to modern astrophysics*. Addison-Wesley.
- Pashchenko, Ilya N, Kirill V Sokolovsky, and Panagiotis Gavras (2018). “Machine learning search for variable stars”. In: *Monthly Notices of the Royal Astronomical Society* 475.2, pp. 2326–2343.
- Pearson, Karl (1901). “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin philosophical magazine and journal of science* 2.11, pp. 559–572.
- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.
- Peng, Liu and Lei Lei (2005). “A review of missing data treatment methods”. In: *Intell. Inf. Manag. Syst. Technol* 1, pp. 412–419.
- Percy, John R (2007). *Understanding variable stars*. Cambridge University Press.
- Perlmutter, Saul, G Aldering, S Deustua, S Fabbro, G Goldhaber, DE Groom, AG Kim, MY Kim, RA Knop, P Nugent, et al. (1998). “Cosmology from type Ia supernovae”. In: *arXiv preprint astro-ph/9812473*.
- Pettersen, BR (1989). “A review of stellar flares and their characteristics”. In: *International Astronomical Union Colloquium*. Vol. 104. 1. Cambridge University Press, pp. 299–312.

- Phillips, Mark M (1993). “The absolute magnitudes of Type IA supernovae”. In: *The Astrophysical Journal* 413, pp. L105–L108.
- Pichara, Karim, Pavlos Protopapas, D-W Kim, J-B Marquette, and Patrick Tisserand (2012). “An improved quasar detection method in EROS-2 and MACHO LMC data sets”. In: *Monthly Notices of the Royal Astronomical Society* 427.2, pp. 1284–1297.
- Pisner, Derek A and David M Schnyer (2020). “Support vector machine”. In: *Machine Learning*. Elsevier, pp. 101–121.
- Pruzhinskaya, Maria V, Konstantin L Malanchev, Matvey V Kornilov, Emille EO Ishida, Florian Mondon, Alina A Volnova, and Vladimir S Korolev (2019). “Anomaly detection in the open supernova catalog”. In: *Monthly Notices of the Royal Astronomical Society* 489.3, pp. 3591–3608.
- Pskovskii, Iu P (1977). “Light curves, color curves, and expansion velocity of type I supernovae as functions of the rate of brightness decline”. In: *Soviet Astronomy* 21, pp. 675–682.
- Quimby, RM, SR Kulkarni, MM Kasliwal, A Gal-Yam, I Arcavi, M Sullivan, P Nugent, R Thomas, DA Howell, E Nakar, et al. (2011). “Hydrogen-poor superluminous stellar explosions”. In: *Nature* 474.7352, pp. 487–489.
- Raffelt, Georg (2012). *Neutrinos and the stars*. arXiv: [1201.1637](https://arxiv.org/abs/1201.1637) [astro-ph.SR].
- Raiteri, CM, M Villata, G Tosti, M Fiorucci, G Ghisellini, LO Takalo, A Sillanpää, E Valtaoja, H Teräsranata, M Tornikoski, et al. (1999). “Optical and radio behaviour of the blazar S4 0954+ 65”. In: *Astronomy and Astrophysics* 352, pp. 19–31.
- Rasmussen, CE and CKI Williams (2006). *Gaussian Processes for Machine Learning* (; Cambridge, MA.
- Rebbapragada, Umaa, Pavlos Protopapas, Carla E Brodley, and Charles Alcock (2009). “Finding anomalous periodic time series”. In: *Machine learning* 74.3, pp. 281–313.
- Reddy, Ummadi, Busi Reddy, and Bodi Reddy (2019). “Recognition of Lung Cancer Using Machine Learning Mechanisms with Fuzzy Neural Networks.” In: *Traitement du Signal* 36.1, pp. 87–91.
- Rees, Martin J (1988). “Tidal disruption of stars by black holes of 10^6 – 10^8 solar masses in nearby galaxies”. In: *Nature* 333.6173, pp. 523–528.
- Refaeilzadeh, Payam, Lei Tang, and Huan Liu (2009). “Cross-validation.” In: *Encyclopedia of database systems* 5, pp. 532–538.
- Richards, Joseph W, Dan L Starr, Nathaniel R Butler, Joshua S Bloom, John M Brewer, Arien Crellin-Quick, Justin Higgins, Rachel Kennedy, and Maxime Rischard (2011). “On machine-learned classification of variable stars with sparse and noisy time-series data”. In: *The Astrophysical Journal* 733.1, p. 10.

- Richards, Joseph W, Dan L Starr, Adam A Miller, Joshua S Bloom, Nathaniel R Butler, Henrik Brink, and Arien Crellin-Quick (2012). “Construction of a calibrated probabilistic classification catalog: application to 50k variable sources in the all-sky automated survey”. In: *The Astrophysical Journal Supplement Series* 203.2, p. 32.
- Riess, Adam G, William H Press, and Robert P Kirshner (1996). “A precise distance indicator: Type Ia supernova multicolor light-curve shapes”. In: *The Astrophysical Journal* 473.1, p. 88.
- Ripoche, Paul, Jeremy Heyl, Javiera Parada, and Harvey Richer (May 2020). “Carbon stars as standard candles: I. The luminosity function of carbon stars in the Magellanic Clouds”. In: *Monthly Notices of the Royal Astronomical Society* 495.3, pp. 2858–2866. ISSN: 13652966. DOI: [10.1093/mnras/staa1346](https://doi.org/10.1093/mnras/staa1346). arXiv: [2005.05539](https://arxiv.org/abs/2005.05539). URL: <https://arxiv.org/abs/2005.05539>.
- Rivinius, Th, D Baade, and S Štefl (2003). “Non-radially pulsating Be stars”. In: *Astronomy & Astrophysics* 411.2, pp. 229–247.
- Roberts, Ethan, Bruce A. Bassett, and Michelle Lochner (2019). *Bayesian Anomaly Detection and Classification*. arXiv: [1902.08627](https://arxiv.org/abs/1902.08627) [stat.ML].
- Robinson, Edward L (1976). “The structure of cataclysmic variables”. In: *Annual review of astronomy and astrophysics* 14.1, pp. 119–142.
- Safavian, S Rasoul and David Landgrebe (1991). “A survey of decision tree classifier methodology”. In: *IEEE transactions on systems, man, and cybernetics* 21.3, pp. 660–674.
- Sako, Masao, Bruce Bassett, Andrew C Becker, Peter J Brown, Heather Campbell, Rachel Wolf, David Cinabro, Chris B D’andrea, Kyle S Dawson, Fritz DeJongh, et al. (2018). “The data release of the Sloan Digital Sky Survey-II supernova survey”. In: *Publications of the Astronomical Society of the Pacific* 130.988, p. 064002.
- Samuel, Arthur L. (1959). “Some studies in machine learning using the game of Checkers”. In: *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, pp. 71–105.
- Scargle, Jeffrey D (1982). “Studies in astronomical time series analysis. II-Statistical aspects of spectral analysis of unevenly spaced data”. In: *The Astrophysical Journal* 263, pp. 835–853.
- Schaefer, Bradley E (1996). “Peak brightnesses of historical supernovae and the Hubble constant”. In: *The Astrophysical Journal* 459, p. 438.
- Schölkopf, Bernhard, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson (2001). “Estimating the support of a high-dimensional distribution”. In: *Neural computation* 13.7, pp. 1443–1471.
- Schölkopf, Bernhard, Robert C Williamson, Alexander J Smola, John Shawe-Taylor, John C Platt, et al. (1999). “Support vector method for novelty detection.” In: *NIPS*. Vol. 12. Citeseer, pp. 582–588.

- Schuldt, S, SH Suyu, R Cañameras, S Taubenberger, T Meinhard, L Leal-Taixé, and BC Hsieh (2020). “Photometric Redshift Estimation with a Convolutional Neural Network: NetZ”. In: *arXiv preprint arXiv:2011.12312*.
- Scovaccicchi, Dario, Robert C Nichol, David Bacon, Mark Sullivan, and Szymon Prajs (2016). “Cosmology with superluminous supernovae”. In: *Monthly Notices of the Royal Astronomical Society* 456.2, pp. 1700–1707.
- Settles, Burr (2009). *Active Learning Literature Survey*. Computer Sciences Technical Report 1648. University of Wisconsin–Madison.
- Shappee, Benjamin J, JL Prieto, D Grupe, CS Kochanek, KZ Stanek, G De Rosa, S Mathur, Y Zu, BM Peterson, RW Pogge, et al. (2014). “The man behind the curtain: X-rays drive the UV through NIR variability in the 2013 active galactic nucleus outburst in NGC 2617”. In: *The Astrophysical Journal* 788.1, p. 48.
- Sharma, Sanjib, Joss Bland-Hawthorn, Kathryn V Johnston, and James Binney (2011). “Galaxia: A code to generate a synthetic survey of the Milky Way”. In: *The Astrophysical Journal* 730.1, p. 3.
- Shi, Tao and Steve Horvath (2006). “Unsupervised learning with random forest predictors”. In: *Journal of Computational and Graphical Statistics* 15.1, pp. 118–138.
- Shoumo, Syed Zamil Hasan, Mir Ishrak Maheer Dhruba, Sazzad Hossain, Nawab Haider Ghani, Hossain Arif, and Samiul Islam (2019). “Application of machine learning in credit risk assessment: a prelude to smart banking”. In: *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*. IEEE, pp. 2023–2028.
- Silk, Joseph and Martin J Rees (1998). “Quasars and galaxy formation”. In: *arXiv preprint astro-ph/9801013*.
- Sirigiri, Sai Kiran Varma (2019). “Hierarchical Classification of Variable Stars Using Neural Networks”. PhD thesis.
- Soares-Santos, Marcelle, DE Holz, J Annis, R Chornock, K Herner, Eric Berger, D Brout, H-Y Chen, R Kessler, M Sako, et al. (2017). “The electromagnetic counterpart of the binary neutron star merger LIGO/Virgo GW170817. I. Discovery of the optical counterpart using the dark energy camera”. In: *The Astrophysical Journal Letters* 848.2, p. L16.
- Solanki, Sami K (2003). “Sunspots: an overview”. In: *The Astronomy and Astrophysics Review* 11.2, pp. 153–286.
- Soszyński, I, R Smolec, A Udalski, and P Pietrukowicz (2019). “Type II Cepheids Pulsating in the First Overtone from the OGLE Survey”. In: *The Astrophysical Journal* 873.1, p. 43.
- Southworth, J, S Zucker, PFL Maxted, and B Smalley (2004). “Eclipsing binaries in open clusters—III. V621 Per in χ Persei”. In: *Monthly Notices of the Royal Astronomical Society* 355.3, pp. 986–994.

- Stetson, Peter B (1996). “On the automatic determination of light-curve parameters for cepheid variables”. In: *Publications of the Astronomical Society of the Pacific* 108.728, p. 851.
- Storey-Fisher, Kate, Marc Huertas-Company, Nesar Ramachandra, Francois Lanusse, Alexie Leauthaud, Yifei Luo, and Song Huang (2020). “Anomaly Detection in Astronomical Images with Generative Adversarial Networks”. In: *arXiv preprint arXiv:2012.08082*.
- Strubbe, Linda E and Eliot Quataert (2009). “Optical flares from the tidal disruption of stars by massive black holes”. In: *Monthly Notices of the Royal Astronomical Society* 400.4, pp. 2070–2084.
- Surbhi, Sultania (2019). *Gamma-ray Burst*. Last accessed 19 October 2021. URL: <https://medium.com/codex/feature-scaling-in-machine-learning-e86b360d1c31>.
- Takahashi, Ichiro, Nao Suzuki, Naoki Yasuda, Akisato Kimura, Naonori Ueda, Masaomi Tanaka, Nozomu Tominaga, and Naoki Yoshida (2020). “Photometric classification of HSC transients using machine learning”. In: *arXiv preprint arXiv:2008.06726*.
- Tanvir, NR, AJ Levan, ASe Fruchter, J Hjorth, RA Hounsell, K Wiersema, and RL Tunnicliffe (2013). “A ‘kilonova’ associated with the short-duration γ -ray burst GRB 130603B”. In: *Nature* 500.7464, pp. 547–549.
- Taubenberger, Stefan (2017). “The Extremes of Thermonuclear Supernovae”. In: *Handbook of Supernovae*, pp. 317–373. DOI: [10.1007/978-3-319-21846-5_37](https://doi.org/10.1007/978-3-319-21846-5_37). URL: http://dx.doi.org/10.1007/978-3-319-21846-5_37.
- Teimoorinia, Hossen, Robert D Toyonaga, Sebastien Fabbro, and Connor Bottrell (2020). “Comparison of Multi-class and Binary Classification Machine Learning Models in Identifying Strong Gravitational Lenses”. In: *Publications of the Astronomical Society of the Pacific* 132.1010, p. 044501.
- Tomaschitz, Roman (2018). “White dwarf stars exceeding the Chandrasekhar mass limit”. In: *Physica A: Statistical Mechanics and its Applications* 489, pp. 128–140.
- Totten, E J and MJ Irwin (1998). “The APM survey for cool carbon stars in the Galactic halo—I”. In: *Monthly Notices of the Royal Astronomical Society* 294.1, pp. 1–27.
- Tur, Gokhan, Dilek Hakkani-Tür, and Robert E Schapire (2005). “Combining active and semi-supervised learning for spoken language understanding”. In: *Speech Communication* 45.2, pp. 171–186.
- Udalski, Andrzej, M Szymanski, J Kaluzny, M Kubiak, and Mario Mateo (1992). “The optical gravitational lensing experiment”. In: *Acta Astronomica* 42, pp. 253–284.
- Ulmer, Andrew (1999). “Flares from the tidal disruption of stars by massive black holes”. In: *The Astrophysical Journal* 514.1, p. 180.

- Valenti, Stefano, J David, Sheng Yang, Enrico Cappellaro, Leonardo Tartaglia, Alessandra Corsi, Saurabh W Jha, Daniel E Reichart, Joshua Haislip, and Vladimir Kouprianov (2017). “The discovery of the electromagnetic counterpart of GW170817: kilonova AT 2017gfo/DLT17ck”. In: *The Astrophysical Journal Letters* 848.2, p. L24.
- Van der Maaten, Laurens and Geoffrey Hinton (2008). “Visualizing data using t-SNE.” In: *Journal of machine learning research* 9.11.
- Veilleux, Sylvain and Donald E Osterbrock (1987). “Spectral classification of emission-line galaxies”. In: *The Astrophysical Journal Supplement Series* 63, pp. 295–310.
- Velzen, Sjoert van, Suvi Gezari, S Bradley Cenko, Erin Kara, James CA Miller-Jones, Tiara Hung, Joe Bright, Nathaniel Roth, Nadejda Blagorodnova, Daniela Huppenkothen, et al. (2019). “The first tidal disruption flare in ZTF: from photometric selection to multi-wavelength characterization”. In: *The Astrophysical Journal* 872.2, p. 198.
- Villar, V Ashley, Edo Berger, Brian D Metzger, and James Guillochon (2017). “Theoretical Models of Optical Transients. I. A Broad Exploration of the Duration–Luminosity Phase Space”. In: *The Astrophysical Journal* 849.1, p. 70.
- Villar, V. Ashley, Miles Cranmer, Edo Berger, Gabriella Contardo, Shirley Ho, Griffin Hosseinzadeh, and Joshua Yao-Yu Lin (2021). “A Deep Learning Approach for Active Anomaly Detection of Extragalactic Transients”. In: arXiv: [2103.12102](https://arxiv.org/abs/2103.12102). URL: <http://arxiv.org/abs/2103.12102>.
- Vogt, SS (1975). “Light and color variations of the flare star by Draconis A critique of starspot properties”. In: *The Astrophysical Journal* 199, pp. 418–426.
- Wang, Sun-Chong (2003). “Artificial neural network”. In: *Interdisciplinary computing in java programming*. Springer, pp. 81–100.
- Warner, Brian (2003). *Cataclysmic variable stars*. Vol. 28. Cambridge University Press.
- Watson, D, KD Denney, Marianne Vestergaard, and Tamara Maree Davis (2011). “A new cosmological distance measure using active galactic nuclei”. In: *The Astrophysical Journal Letters* 740.2, p. L49.
- Webb, Sara, Michelle Lochner, Daniel Muthukrishna, Jeff Cooke, Chris Flynn, Ashish Mahabal, Simon Goode, Igor Andreoni, Tyler Pritchard, and Timothy MC Abbott (2020). “Unsupervised machine learning for transient discovery in deeper, wider, faster light curves”. In: *Monthly Notices of the Royal Astronomical Society* 498.3, pp. 3077–3094.
- West, Andrew A, Dylan P Morgan, John J Bochanski, Jan Marie Andersen, Keaton J Bell, Adam F Kowalski, James RA Davenport, Suzanne L Hawley, Sarah J Schmidt, David Bernat, et al. (2011). “The sloan digital sky survey data release 7 spectroscopic M dwarf catalog. I. Data”. In: *The Astronomical Journal* 141.3, p. 97.

- Whitelock, Patricia A (1999). ““Real-time” evolution in Mira variables”. In: *New Astronomy Reviews* 43.6-7, pp. 437–440.
- Yang, Qian, Xue-Bing Wu, Xiaohui Fan, Linhua Jiang, Ian McGreer, Jinyi Shangguan, Su Yao, Bingquan Wang, Ravi Joshi, Richard Green, et al. (2018). “Discovery of 21 New Changing-look AGNs in the Northern Sky”. In: *The Astrophysical Journal* 862.2, p. 109.
- Yang, T. -C., A. J. Drake, A. A. Mahabal, S. G. Djorgovski, M. J. Graham, R. Williams, J. L. Prieto, M. Catelan, E. Christensen, and S. M. Larson (May 2013). “Classification of CRTS Supernova Discoveries”. In: *The Astronomer’s Telegram* 5077, p. 1.
- Zhang, Cha and Tsuhan Chen (2002). “An active learning framework for content-based information retrieval”. In: *IEEE transactions on multimedia* 4.2, pp. 260–268.
- Zhou, Li, Xiaofeng Wang, Kaicheng Zhang, Juncheng Chen, Jide Liang, Tianmeng Zhang, Xu Zhou, Fang Huang, Xulin Zhao, Xiaofeng Wang, Tianmeng Zhang, D. D. Balam, M. L. Graham, and E. Y. Hsiao (June 2013). “Supernova 2013cv = Psn J16224316+1857356”. In: *Central Bureau Electronic Telegrams* 3543, p. 1.
- Zhu, Yonglin, Ryan Thomas Wollaeger, Nicole Vassh, Rebecca Surman, TM Sprouse, Matthew Ryan Mumpower, P Möller, GC McLaughlin, Oleg Korobkin, Toshihiko Kawano, et al. (2018). “Californium-254 and kilonova light curves”. In: *The Astrophysical Journal Letters* 863.2, p. L23.



Appendix A

Objects with Unclear Labels in the MANTRA Data

TABLE A.1: A List of objects with unclear labels in the MANTRA data.

| Classification | Number of objects | Classification | Number of objects |
|----------------|-------------------|----------------|-------------------|
| AGN/Blazar | 3 | Var/Ast? | 2 |
| AGN/CV | 2 | Var/Flare? | 1 |
| AGN/Flare? | 1 | Var/Nothing | 1 |
| AGN/SN | 6 | Var/Nova | 1 |
| AGN/SN? | 4 | Var/SN | 2 |
| AGN/Var | 4 | Var/SN? | 2 |
| AGN/Var? | 1 | Var/nothing? | 1 |
| AGN/nothing? | 1 | Var? | 15 |
| AGN? | 138 | YSO? | 5 |
| AMCVn? | 1 | Unclear | 1 |
| Ast/CV? | 2 | Var/AGN | 1 |
| Ast/Flare | 1 | Var/Artifact | 1 |
| Ast/Flare? | 1 | Var/Ast | 5 |
| Ast/SN | 4 | SN/AGN? | 11 |
| Ast/SN? | 1 | SN/Ast | 10 |
| Ast/Var | 1 | SN/Ast? | 2 |
| Ast/Var? | 2 | SN/CV | 36 |
| Ast/Var?? | 1 | SN/CV? | 6 |
| Ast? | 68 | SN/TDE | 6 |

| Continuation of Table 5.1 | | | |
|---------------------------|-------------------|------------------|-------------------|
| Classification | Number of objects | Classification | Number of objects |
| Blazar/AGN | 4 | SN/TDE? | 1 |
| Blazar/SN | 1 | SN/Var | 3 |
| Blazar? | 19 | SN/Var? | 1 |
| CV/AGN | 3 | SN/nothing? | 1 |
| CV/AGN? | 3 | SN? | 319 |
| CV/Ast | 4 | TDE? | 3 |
| CV/Ast? | 1 | SN/AGN | 23 |
| CV/Blazar | 1 | O/Ne | 1 |
| CV/Flare | 2 | Nova? | 1 |
| CV/SN | 19 | Nova/CV | 1 |
| CV/SN? | 1 | Nothing/Lensing | 1 |
| CV/Var | 5 | Merger/CV? | 1 |
| CV/Var/Ast | 1 | Lensing/nothing? | 1 |
| CV/Var? | 1 | HPM? | 3 |
| CV? | 77 | HPM/Var? | 1 |
| Comet/Ast? | 1 | Flare? | 20 |
| Flare/CV | 1 | Flare/SN? | 1 |
| Flare/SN | 2 | | |

UNIVERSITY of the
WESTERN CAPE

Appendix B

The code developed when extending Astronomy to Operate with Light Curve Data



UNIVERSITY of the
WESTERN CAPE

B.1 The Light Curve Reader Class

This code is used to read in different kinds of light curve datasets. The code first prompts the user to specify: file path(s), time columns, brightness columns, brightness error columns, the number of lines covered by the header, if the user wants to convert flux values to magnitudes, and if the user wants to split the light curves into chunks based on their gaps. It then loads the data, discards the header, converts flux to magnitudes (optional), splits the light curves (optional), and standardises the column names as described in Chapter 3. Lastly, it returns a PANDAS dataframe with standard column names.

```
import pandas as pd
import numpy as np
from astronomy.base.base_dataset import Dataset

# ignores the false positive pandas warning
# for the following kind of code
# df['key'] == item, for an existing key in a df
pd.options.mode.chained_assignment = None
```

```

def split_lc(lc_data, max_gap):
    '''Splits the light curves into smaller chunks based on their gaps

    Parameters
    -----
        lc_data: Dataframe with the light curves
        max_gap: Maximum gap between observations'''

    unq_ids = np.unique(lc_data.ID)
    unq_ids = unq_ids
    splitted_dict = {}

    id_n = 0
    for ids in unq_ids:
        id_n += 1
        progress = id_n/len(unq_ids)
        progress = progress*100

        print('Concatinating {}%'.format(progress))
        lc = lc_data[lc_data['ID'] == ids]
        if 'filters' in lc.columns:
            unq_filters = np.unique(lc.filters)

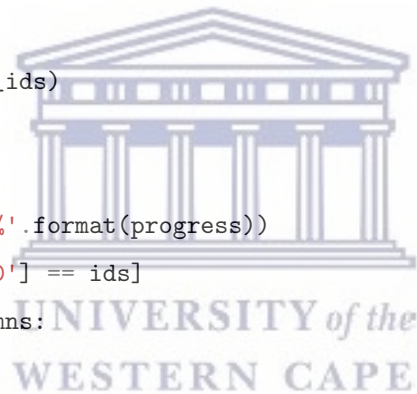
            for filtr in unq_filters:

                lc1 = lc[lc['filters'] == filtr]

                time = lc1.time
                time_diff = [time.iloc[i] - time.iloc[i-1]
                             for i in range(1, len(time))]
                time_diff.insert(0, 0)
                lc1['time_diff'] = time_diff
                gap_idx = np.where(lc1.time_diff > max_gap)[0]

                # Separating the lc as by the gap index
                try:

```




```

lc0 = lc1.iloc[:gap_idx[0]]
lc0['ID'] = [ids+'_0' for i in range(len(lc0.time))]

splitted_dict.update({'lc'+ids+'_'+str(filtrl)+str(0): lc0})

for k in range(1, len(gap_idx)):

    lcn = lc1.iloc[gap_idx[k-1]:gap_idx[k]]
    lcn['ID'] = [ids+'_'+str(k)
                 for i in range(len(lcn.time))]

    splitted_dict.update({'lc'+ids+'_'+str(filtrl)+str(k):
                          lcn})

    lc2 = lc1.iloc[gap_idx[k]:]
    lc2['ID'] = [ids+'_'+str(k+1)
                 for i in range(len(lc2.time))]

    splitted_dict.update({'lc'+ids+'_'+str(filtrl)+str(k+1):
                          lc2})

except (IndexError, UnboundLocalError):
    pass

final_data = pd.concat(splitted_dict.values(), ignore_index=False)
return final_data

def convert_flux_to_mag(lcs, f_zero):
    '''Converts flux to mags for a given light curve data

    Parameters
    -----
        lcs: DataFrame with the light curve values
        zeropoint: Zeropoint magnitude
    '''

```

```

# Replacing the negative flux values with their respective errors
neg_flux_indx = np.where(lcs['flux'].values < 0)
lcs.loc[lcs['flux'] < 0, ['flux']] = lcs['flux_error'].iloc[neg_flux_indx]

# Only consider ugiz bands
lc = lcs[lcs['filters'].isin([1, 2, 3, 4])]

# Flux and flux error
f_obs = lc.flux.values
f_obs_err = lc.flux_error.values
constants = (2.5/np.log(10))
# converting
flux_convs = - 2.5*np.log10(f_obs/f_zero)
err_convs = constants*(f_obs_err/f_obs)
# Adding the new mag and mag_error column
lc['mag'] = flux_convs
lc['mag_error'] = err_convs

return lc

```



```

class LightCurveDataset(Dataset):
    def __init__(self, data_dict, f_zero=22, header_nrows=1,
                 delim_whitespace=False, max_gap=50, plot_errors=True,
                 convert_flux=True, split_lc=False,
                 filter_colors=['#9467bd', '#1f77b4', '#2ca02c', '#d62728',
                               '#ff7f0e', '#8c564b'],
                 filter_labels=[],
                 **kwargs):
        """
        Reads in light curve data from file(s).

        Parameters
        -----
        filename : str
            If a single file (of any time) is to be read from, the path can be
            given using this kwarg.

```

directory : str

A directory can be given instead of an explicit list of files. The child class will load all appropriate files in this directory.

list_of_files : list

Instead of the above, a list of files to be loaded can be explicitly given.

output_dir : str

The directory to save the log file and all outputs to. Defaults to './'

data_dict: Dictionary

It a dictionary with index of the column names corresponding to the following specific keys:

('id', 'time', 'mag', 'mag_err', 'flux', 'flux_err', 'filters', 'labels')

e.g {'time':1, 'mag':2}, where 1 and 2 are column index correponding to 'time' and 'mag' in the input data.

If the data does not have unique ids, the user can neglect the 'id' key, and the ids will be the file path by default.

The user can also provide a list of indices for the 'mag' and 'flux' columns.

This is the case where the brightness is recorded in more than one column. e.g {'time':1, 'mag':[2,3]} 2 and 3 corresponds to columns with brightness records

header_nrows: int

The number of rows the header covers in the dataset, by default 1

f_zero : float/int

The zero flux magnitude values, by default 22

max_gap: int

Maximum gap between consecute observations, default 50

split_lc: bool

Should be true if one wants to split the ligh curves based on the gaps. Default False.

delim_whitespace: bool

Should be True if the data is not separated by a comma, by default False

plot_errors: bool

If errors are available for the data, this boolean allows them

```

        to be plotted
filter_colors: list
    Allows the user to define their own colours (using hex codes)
    for the different filter bands. Will revert to default
    behaviour of the JavaScript chart if the list of colors
    provided is shorter than the number of unique filters.
filter_labels: list
    For multiband data, labels will be passed to the frontend
    allowing easy identification of different bands in the light
    curve. Assumes the filters are identified by an integer in the
    data such that the first filter (e.g. filter 0) will correspond
    to the first label provided. For example, to plot PLASTiCC
    data, provide filter_labels=['u','g','r','i','z','y']
"""

super().__init__(data_dict=data_dict, header_nrows=header_nrows,
                 delim_whitespace=delim_whitespace, f_zero=f_zero,
                 max_gap=max_gap, plot_errors=plot_errors,
                 filter_labels=filter_labels, split_lc=split_lc,
                 convert_flux=convert_flux,
                 filter_colors=filter_colors,**kwargs)

self.data_type = 'light_curve'
self.metadata = pd.DataFrame(data=[])
self.data_dict = data_dict
self.header_nrows = header_nrows
self.delim_whitespace = delim_whitespace
self.f_zero = f_zero
self.max_gap = max_gap
self.plot_errors = plot_errors
self.filter_labels = filter_labels
self.filter_colors = filter_colors
self.convert_flux = convert_flux
self.split_lc = split_lc

# =====
#           Reading the light curve data
# =====

```

```

# The case where there is one file
data = pd.read_csv(self.files[0], skiprows=self.header_nrows,
                  delim_whitespace=self.delim_whitespace, header=None)

# Splitting the light curve data using the gaps
if self.split_lc is True:
    data = split_lc(data, self.max_gap)

# The case for multiple files of light curve data
file_len = [len(data)]
if len(self.files) > 1:

    file_paths = [self.files[0]]
    for fl in range(1, len(self.files)):
        data = pd.concat([data, pd.read_csv(self.files[fl],
                                           skiprows=self.header_nrows,
                                           delim_whitespace=self.delim_whitespace,
                                           header=None)])

    file_paths.append(self.files[fl])
    file_len.append(len(data))

IDs = [file_paths[0] for i in range(file_len[0])]
for fl in range(1, len(file_len)):

    for f in range(file_len[fl] - file_len[fl-1]):

        IDs.append(file_paths[fl])

# =====
#           Renaming the columns into standard columns for astronomy
# =====
time = data.iloc[:, self.data_dict['time']]
standard_data = {'time': time}

if 'id' in data_dict.keys():
    idx = data.iloc[:, self.data_dict['id']]

```

```

        ids = np.unique(idx)
        ids = np.array(ids, dtype='str')
#         self.index = ids[:5] # Testing for 100 objects
        standard_data.update({'ID': np.array(ids, dtype='str')})

    else:
        idx = self.files
        self.index = idx
        self.metadata = pd.DataFrame({'ID': idx}, index=idx)
        standard_data.update({'ID': IDs})

    if 'labels' in data_dict.keys():

        labels = data.iloc[:, self.data_dict['labels']]

        standard_data.update({'labels': labels})

# Possible brightness columns
    brightness_cols = ['mag', 'flux']

# WE NEED TO CONVERT FLUX TO MAG FOR FEETS FEATURE EXTRACTOR
# Looping through the brightness columns
    for col in range(len(brightness_cols)):
        data_col = brightness_cols[col]
        if data_col in self.data_dict.keys():

            # =====Multiple brightness columns=====
            try:

                for i in range(len(self.data_dict[data_col])):

                    # The case where there are no error columns
                    standard_data.update({data_col+str(i+1):
                                         data.iloc[:, self.data_dict[
                                             data_col][i]]})

                    # The case where there are brightness error columns
                    if data_col+'_err' in self.data_dict.keys():

```

```

        # Updating the standard dictionary to include the
        # brightness_errors
        standard_data.update({data_col+'_error'+str(i+1):
                             data.iloc[:, self.data_dict[
                                 data_col+'_err'][i]]})

# =====Single brightness Column=====
# =====
except TypeError:

    # The case for single brightness column and no errors
    standard_data.update({data_col:
                          data.iloc[:, self.data_dict[
                              data_col]]})

    if data_col+'_err' in self.data_dict.keys():

        standard_data.update({data_col+'_error':
                              data.iloc[:, self.data_dict[
                                  data_col+'_err']]})

# =====The case where there are filters in the data=====
if 'filters' in self.data_dict.keys():

    standard_data.update({'filters': data.iloc[
                          :, self.data_dict['filters']]})

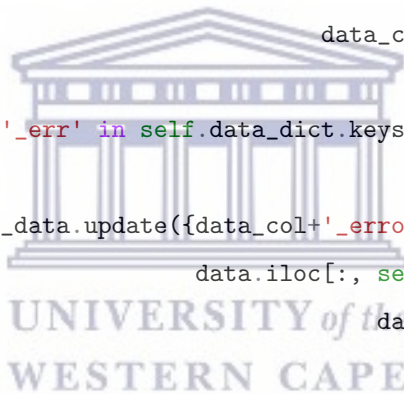
lc = pd.DataFrame.from_dict(standard_data)

if 'flux' in lc.columns:

    # Convert flux to mag
    if convert_flux is True:
        lc = convert_flux_to_mag(lc, self.f_zero)

    # THIS IS TEMPORARY, ESSENTIAL FOR PLOTTING
    # lc1 = lc.copy()

```




```

    lc['mag'] = lc.flux
    lc['mag_error'] = lc.flux_error
    # =====UnComment To Split Lcs=====
    # Split the light curve into chunks
    self.light_curves_data = lc

else:
    self.light_curves_data = lc

ids = np.unique(lc.ID)
self.index = ids

# Add the classes to the metadata
if 'labels' in lc.columns:

    lc1 = lc.copy()
    lc1 = lc.drop_duplicates(subset='ID')
    labels = [lc1[lc1['ID'] == i]['labels'].values[0] for i in ids]
    self.metadata = pd.DataFrame({'label': labels, 'ID': ids},
                                index=ids)

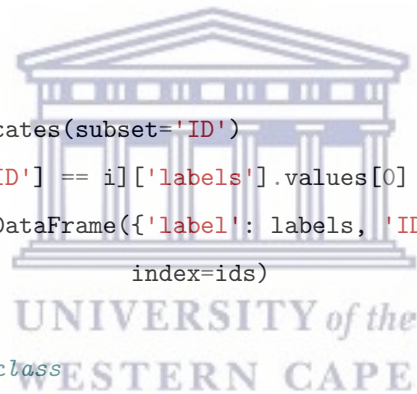
# Metadata without the class
else:
    self.metadata = pd.DataFrame({'ID': ids}, index=ids)

def get_display_data(self, idx):
    """
    Returns a single instance of the dataset in a form that is ready to be
    displayed by the web front end.

    Parameters
    -----
    idx : str
        Index (should be a string to avoid ambiguity)

    Returns
    -----
    dict

```



```

        json-compatible dictionary of the light curve data
        """

        # WE NEED TO EXPAND THIS TO BE MORE GENERAL

        # All the standard columns are included here
        data_col = ['mag']
        err_col = ['mag_error']
        out_dict = {'data': [], 'errors': [], 'filter_labels': [],
                   'filter_colors': []}

        # Reading in the light curve data
        light_curve_original = self.light_curves_data[
            self.light_curves_data['ID'] == idx]

        lc_cols = light_curve_original.columns.values.tolist()
        if err_col[0] in lc_cols and self.plot_errors:
            plot_errors = True
        else:
            plot_errors = False

        if 'filters' in lc_cols:
            multiband = True
            unique_filters = np.unique(light_curve_original['filters'])
        else:
            multiband = False
            unique_filters = [0]

        k = 0
        for filt in unique_filters:
            if multiband:
                msk = light_curve_original['filters'] == filt
                light_curve = light_curve_original[msk]
            else:
                light_curve = light_curve_original

        mag_indx = [cl for cl in data_col if cl in lc_cols]
        err_indx = [cl for cl in err_col if cl in lc_cols]

```



```

if plot_errors:
    light_curve['err_lower'] = light_curve[mag_indx].values - \
        light_curve[err_indx].values

    light_curve['err_upper'] = light_curve[mag_indx].values + \
        light_curve[err_indx].values

    lc_errs = light_curve[['time', 'err_lower', 'err_upper']]

    err = lc_errs.values.tolist()

# inserting the time column to data and adding 'data'
# and 'errors' to out_dict
mag_indx.insert(0, 'time')
dat = light_curve[mag_indx].values.tolist()

out_dict['data'].append(dat)

if plot_errors:
    out_dict['errors'].append(err)
else:
    out_dict['errors'].append([])

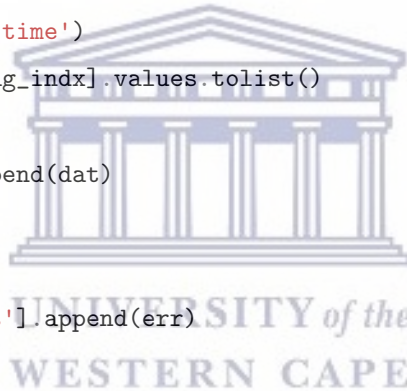
if len(self.filter_labels) >= len(unique_filters):
    out_dict['filter_labels'].append(self.filter_labels[k])
else:
    out_dict['filter_labels'].append((str)(filt))

if len(self.filter_colors) >= len(unique_filters):
    out_dict['filter_colors'].append(self.filter_colors[k])
else:
    out_dict['filter_colors'].append('')

k += 1

return out_dict

```



```

def get_sample(self, idx):

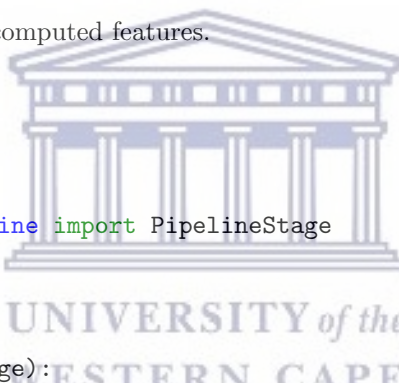
    # Choosing light curve values for a specific ID
    light_curve_sample = self.light_curves_data[
        self.light_curves_data['ID'] == idx]

    return light_curve_sample

```

B.2 The feets Feature Extractor Class

This code incorporates the `feets` package to extract features from the standardised light curves returned by the light curve reader. It prompts the user to: give a list of features to exclude, specify if they want the features to be computed on the magnitude or flux, and specify the number of point cut. It then extracts features and returns a PANDAS dataframe with the computed features.



```

import numpy as np
import feets
from astronomaly.base.base_pipeline import PipelineStage

class Feets_Features(PipelineStage):

    '''Computes the features using feets package

    Parameters:
        exclude_features: Features to be excluded when calculating the features

    Output:
        A 1D array with the extracted feature'''

    def __init__(self, exclude_features, compute_on_mags=True, n_points_plc=5,
                 **kwargs):

        super().__init__(exclude_features=exclude_features,
                         compute_on_mags=compute_on_mags,
                         n_points_plc=n_points_plc, **kwargs)

```

```
self.exclude_features = exclude_features
self.labels = None
self.compute_on_mags = compute_on_mags
self.n_points_plc = n_points_plc

def _set_labels(self, feature_labels):

    # All available features
    self.labels = feature_labels

def _execute_function(self, lc_data):

    '''Takes light curve data for a single object and computes the features
    based on
    the available columns.

    Input:
        lc_data: Light curve of a single object

    Output:
        An array of the calculated features or an array of nan values
        incase there is an error during the feature extraction process'''

    # Sorting the columns for the feature extractor
    # This needs to be extended to be more general

    if self.compute_on_mags is True:

        standard_lc_columns = ['time', 'mag', 'mag_error']

    else:

        standard_lc_columns = ['time', 'flux', 'flux_error']

    current_lc_columns = [cl for cl in standard_lc_columns
                          if cl in lc_data.columns]

    # list to store column names supported by feets
    available_columns = ['time']
```



```
# Renaming the columns for feets
for cl in current_lc_columns:

    if cl == 'mag' or cl == 'flux':

        available_columns.append('magnitude')

    if cl == 'mag_error' or cl == 'flux_error':

        available_columns.append('error')

# Getting the length of features to be calculated
fs = feets.FeatureSpace(data=available_columns,
                        exclude=self.exclude_features)

len_labels = len(fs.features_)

# The case where we have filters
if 'filters' in lc_data.columns:

    ft_values = []
    ft_labels = []

    for i in range(0, 6):

        passbands = ['u', 'g', 'r', 'i', 'z', 'y']
        # passbands = ['g', 'r', 'i', 'z']
        filter_lc = lc_data[lc_data['filters'] == i]

        lc_columns = []
        for col in current_lc_columns:
            lc_columns.append(filter_lc[col])

# Accounts for light curves that do not have some filters
if len(filter_lc.ID) != 0:

    # Checking the number of points in the light curve
    if len(filter_lc.ID) >= self.n_points_plc:
```



```
        features, values = fs.extract(*lc_columns)

        # print(features)

        new_labels = [f + '_' + passbands[i-1]
                       for f in features]

        for j in range(len(features)):
            ft_labels.append(new_labels[j])
            ft_values.append(values[j])

    else:
        for ft in fs.features_:
            ft_labels.append(ft+'_'+passbands[i-1])
            ft_values.append(np.nan)

    else:
        for vl in fs.features_:
            ft_values.append(np.nan)
            ft_labels.append(vl+'_'+passbands[i-1])

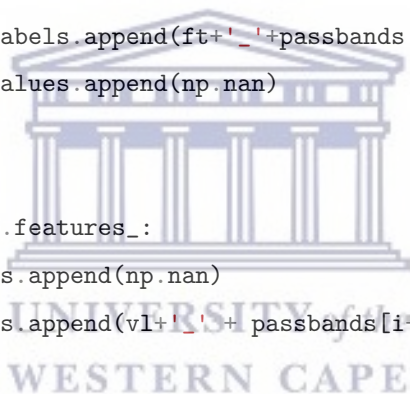
    # Updating the labels
    if self.labels is None:

        self._set_labels(list(ft_labels))

    return ft_values

# The case with no filters
else:

    if len(lc_data.ID) >= self.n_points_plc:
        # print('passed')
        lc_columns = []
        for col in current_lc_columns:
            lc_columns.append(lc_data[col])
```




```

ft_labels, ft_values = fs.extract(*lc_columns)

# # Updating the labels
if self.labels is None:

    self._set_labels(list(ft_labels))
return ft_values

# Returns an array of nan values
else:
    print('Not satisfied')
    return np.array([np.nan for i in range(len_labels)])

```

B.3 An Example script showing how to implement Astronomy to search for anomalies in the MANTRA data.

This is an example script that is passed to the ASTRONOMY frontend, specifically for the MANTRA data (i.e., `case_1a` light curve data type as described in Chapter 3).

```

from astronomy.data_management import light_curve_reader
from astronomy.feature_extraction import feets_features
from astronomy.postprocessing import scaling
from astronomy.anomaly_detection import isolation_forest, human_loop_learning
from astronomy.visualisation import tsne
import os
import pandas as pd

# Root directory for data
data_dir = os.path.join(os.getcwd(), 'example_data')
lc_path = os.path.join(data_dir, 'transient_lightcurves.csv')

# Where output should be stored
output_dir = os.path.join(
    data_dir, 'astronomy_output', '')

```

```
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

display_transform_function = []

def run_pipeline():
    """
    Any script passed to the Astronomy server must implement this function.
    run_pipeline must return a dictionary that contains the keys listed below.

    Parameters
    -----

    Returns
    -----
    pipeline_dict : dictionary
        Dictionary containing all relevant data. Keys must include:
        'dataset' - an astronomy Dataset object
        'features' - pd.DataFrame containing the features
        'anomaly_scores' - pd.DataFrame with a column 'score' with the anomaly
        scores
        'visualisation' - pd.DataFrame with two columns for visualisation
        (e.g. TSNE or UMAP)
        'active_learning' - an object that inherits from BasePipeline and will
        run the human-in-the-loop learning when requested

    """

    # This creates the object that manages the data
    lc_dataset = light_curve_reader.LightCurveDataset(
        filename=lc_path,
        data_dict={'id': 0, 'time': 4, 'mag': 2, 'mag_err': 3}
    )

    # Creates a pipeline object for feature extraction
    pipeline_feats = feets_features.Feets_Features(
        exclude_features=['Period_fit'])
```



```
# Actually runs the feature extraction
features = pipeline_feats.run_on_dataset(lc_dataset)

# Now we rescale the features using the same procedure of first creating
# the pipeline object, then running it on the feature set
pipeline_scaler = scaling.FeatureScaler(force_rerun=False,
                                       output_dir=output_dir)
features = pipeline_scaler.run(features)

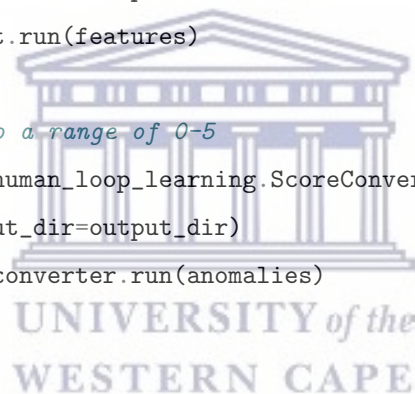
# The actual anomaly detection is called in the same way by creating an
# Iforest pipeline object then running it
pipeline_iforest = isolation_forest.IforestAlgorithm(
    force_rerun=False, output_dir=output_dir)
anomalies = pipeline_iforest.run(features)

# We convert the scores onto a range of 0-5
pipeline_score_converter = human_loop_learning.ScoreConverter(
    force_rerun=False, output_dir=output_dir)
anomalies = pipeline_score_converter.run(anomalies)

try:
    # This is used by the frontend to store labels as they are applied so
# that labels are not forgotten between sessions of using Astronomy
    if 'human_label' not in anomalies.columns:
        df = pd.read_csv(
            os.path.join(output_dir, 'ml_scores.csv'),
            index_col=0,
            dtype={'human_label': 'int'})
        df.index = df.index.astype('str')

        if len(anomalies) == len(df):
            anomalies = pd.concat(
                (anomalies, df['human_label']), axis=1, join='inner')
except FileNotFoundError:
    pass

# This is the active learning object that will be run on demand by the
```



```

# frontend
pipeline_active_learning = human_loop_learning.NeighbourScore(
    alpha=1, output_dir=output_dir)

# We use TSNE for visualisation which is run in the same way as other parts
# of the pipeline.
pipeline_tsne = tsne.TSNE_Plot(
    force_rerun=False,
    output_dir=output_dir,
    perplexity=100)
t_plot = pipeline_tsne.run(features)

# The run_pipeline function must return a dictionary with these keywords
return {'dataset': lc_dataset,
        'features': features,
        'anomaly_scores': anomalies,
        'visualisation': t_plot,
        'active_learning': pipeline_active_learning}

```

B.4 An Example script showing how to implement Astronomy to search for anomalies in the PLAsTiCC data.

Similar to B.4 but for the PLAsTiCC data (i.e., `case_1b` light curve data type).

```

from astronomy.data_management import light_curve_reader
from astronomy.feature_extraction import feats_features
from astronomy.postprocessing import scaling
from astronomy.anomaly_detection import isolation_forest, human_loop_learning
from astronomy.visualisation import tsne
import os
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer

# Replace missing values with the median of the data
imp = SimpleImputer(missing_values=np.nan, strategy='median')

```

```
# Root directory for data
data_dir = os.path.join(os.getcwd(), 'example_data')
lc_path = os.path.join(data_dir, '10k_KN_RRL_test_sample.csv')

# Where output should be stored
output_dir = os.path.join(
    data_dir, 'astronomy_output', 'plasticc_norm_flux', '')

if not os.path.exists(output_dir):
    os.makedirs(output_dir)

display_transform_function = []

def artificial_human_labelling(anomalies=None, metadata=None, N=200,
                               human_labels={0: 0, 1: 5}):

    print('Artificially adding human labels...')
    if anomalies is None:
        raise ValueError('Anomaly score dataframe not provided')
    if metadata is None:
        raise ValueError('True labels not given')

    anomalies['human_label'] = [-1] * len(anomalies)

    labels = metadata.loc[anomalies.index]
    for k in list(human_labels.keys()):
        inds = labels.index[:N][(np.where(labels.label[:N] == k))[0]]
        anomalies.loc[inds, 'human_label'] = human_labels[k]

    print('Done!')

    return anomalies

def run_pipeline():
    """
    Any script passed to the Astronomy server must implement this function.

```

run_pipeline must return a dictionary that contains the keys listed below.

Parameters

Returns

pipeline_dict : dictionary

Dictionary containing all relevant data. Keys must include:

'dataset' - an astronomy Dataset object

'features' - pd.DataFrame containing the features

'anomaly_scores' - pd.DataFrame with a column 'score' with the anomaly scores

'visualisation' - pd.DataFrame with two columns for visualisation (e.g. TSNE or UMAP)

'active_learning' - an object that inherits from BasePipeline and will run the human-in-the-loop learning when requested

"""

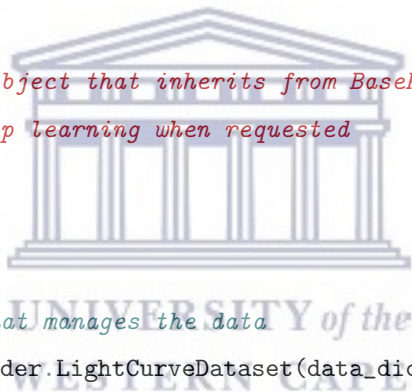
This creates the object that manages the data

```
lc_dataset = light_curve_reader.LightCurveDataset(data_dict={'id': 0,
                                                            'time': 1,
                                                            'flux_err': 4,
                                                            'flux': 3,
                                                            'filters': 2,
                                                            'labels': 6},
                                                  filename=lc_path,
                                                  plot_errors=False,
                                                  delim_whitespace=False,
                                                  header_nrows=1,
                                                  max_gap=100,
                                                  filter_labels=['g', 'r',
                                                                'i', 'z'],
                                                  convert_flux=False)
```

print(lc_dataset.index)

Creates a pipeline object for feature extraction

```
pipeline_feats = feets_features.Feets_Features(
```



```
compute_on_mags=False,
exclude_features=['Period_fit', 'PercentDifferenceFluxPercentile',
                 'FluxPercentileRatioMid20',
                 'FluxPercentileRatioMid35',
                 'FluxPercentileRatioMid50',
                 'FluxPercentileRatioMid65',
                 'FluxPercentileRatioMid80',
                 'Freq1_harmonics_rel_phase_0',
                 'Freq2_harmonics_rel_phase_0',
                 'Freq3_harmonics_rel_phase_0',
                 'Autocor_length', 'Con']
)

# Actually runs the feature extraction
features = pipeline_feats.run_on_dataset(lc_dataset)
columns = features.columns
indx = features.index.values.tolist()
features = imp.fit_transform(features,)
features = pd.DataFrame(features, columns=columns, index=indx)
# print(features)

# Now we rescale the features using the same procedure of first creating
# the pipeline object, then running it on the feature set
pipeline_scaler = scaling.FeatureScaler(force_rerun=False,
                                       output_dir=output_dir)
features = pipeline_scaler.run(features)

# The actual anomaly detection is called in the same way by creating an
# Iforest pipeline object then running it
pipeline_iforest = isolation_forest.IforestAlgorithm(
    force_rerun=False, output_dir=output_dir)
anomalies = pipeline_iforest.run(features)

# We convert the scores onto a range of 0-5
pipeline_score_converter = human_loop_learning.ScoreConverter(
    force_rerun=False, output_dir=output_dir)
anomalies = pipeline_score_converter.run(anomalies)
```



```

# Human labels
anomalies = anomalies.sort_values('score', ascending=False)
anomalies = artificial_human_labelling(
    anomalies=anomalies, metadata=lc_dataset.metadata, N=500,
    human_labels={0: 0, 1: 5})

try:
    # This is used by the frontend to store labels as they are applied so
    # that labels are not forgotten between sessions of using Astronomy
    if 'human_label' not in anomalies.columns:
        df = pd.read_csv(
            os.path.join(output_dir, 'ml_scores.csv'),
            index_col=0,
            dtype={'human_label': 'int'})
        df.index = df.index.astype('str')

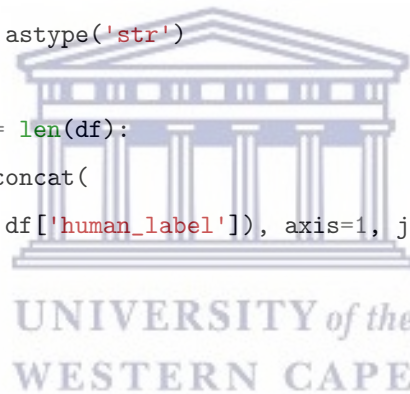
        if len(anomalies) == len(df):
            anomalies = pd.concat(
                (anomalies, df['human_label']), axis=1, join='inner')
except FileNotFoundError:
    pass

# This is the active learning object that will be run on demand by the
# frontend
pipeline_active_learning = human_loop_learning.NeighbourScore(
    alpha=1, output_dir=output_dir)

# We use TSNE for visualisation which is run in the same way as other parts
# of the pipeline.
pipeline_tsne = tsne.TSNE_Plot(
    force_rerun=False,
    output_dir=output_dir,
    perplexity=100)
t_plot = pipeline_tsne.run(features)

# The run_pipeline function must return a dictionary with these keywords
return {'dataset': lc_dataset,
        'features': features,

```



```
'anomaly_scores': anomalies,  
'visualisation': t_plot,  
'active_learning': pipeline_active_learning}
```

