UNSUPERVISED CLUSTERING OF TRANSIENT EVENTS IN THE VARIABLE SKY

AUTHOR: NISHANK JAIN

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GANDHINAGAR

SUMMER INTERNSHIP PROJECT REPORT

MAY 20 – JULY 26, 2013

MENTOR: ASHISH MAHABAL

CO-MENTOR: CIRO DONALEK

Contents

Abstract

Discovering transients and classifying them is a complex process. The Catalina Real Time Transient Survey detects and publishes all transients that it discovers within a span of few minutes of observation to VOEventNet and SkyAlert. Using the Caltech Time Series Characterization Service, many periodic and non-periodic features are extracted from each lightcurve thereby creating a new dataset to be used for categorization and classification. Lightcurves can show tremendous variation in various characteristics which makes comparisons between them difficult and training classifiers even harder. A way to approach this problem is to characterize a set of lightcurves via a set of common features and then use this representation of lightcurves for analysis and training of classifiers. From the lightcurves of known classes, mathematical representations of the best of these features can be derived and then used as training sets for unsupervised clustering of transients. One such approach using Self-Organizing Maps with the CRTS dataset through application of the SOM Toolbox in Matlab is demonstrated.

Acknowledgements

I wish to acknowledge and express my sincere gratitude towards the efforts put in by Prof. Arup Lal Chakraborty and Prof. Anand Sengupta for writing my recommendation letters and helping me get selected for the internship. I would like to thank Ashish Mahabal and Ciro Donalek for being a great mentor and co-mentor respectively and helping me out and guiding me through the ten weeks that I spent with them. I learnt a lot from these people and not just about the project at hand but also about the research going all around the world, the magnitude of people involved in scientific discoveries, the intricacies of getting a Masters or a PhD degree when there is so much competition around and the scope of improvement is so little yet so much.

I would also like to thank George Djorgovski, Andrew Drake, Matthew Graham, Arun Kumar for their dedicated help and constant inputs throughout the project duration.

I would like to thank my financial sponsors for helping in materializing this project

I would also like to acknowledge the extended help offered by my fellow intern Kartik Saxena time and again.

Thank You for what you have given to me and I seek to continue this relationship for years to come.

Nishank Jain

Background

Time-domain astronomy is a rapidly advancing field wherein investigation and classification of different kinds of objects is done on the basis of varying brightness over a specific period of time which may range from minutes to years. These objects can be Supernovae, Quasars, Blazars, Flares, Cataclysmic Variables or some undetected or unidentified objects. Such objects are termed Variables or Transients. The Catalina Real-time Transient Sky Survey (CRTS) uses data from the Catalina Sky Survey which comprises of observations from three telescopes – CSS, MLS and SSS to detect these transients and publishes all transients within a span of few minutes of observation to VOEventNet and SkyAlert.

Lightcurves are published for different kinds of objects such as Cataclysmic Variables (CVs), Blazars, RR Lyrae, Supernovae (SN) etc. Using the Caltech Time Series Characterization Service, many periodic and non-periodic features are extracted from each lightcurve thereby creating a new dataset. This dataset is used for categorization and classification of astronomical lightcurves.

Algorithms are being written to classify the objects from the CRTS data stream into various categories as mentioned above and many more. Each night, $10\text{-}10^2$ transients are detected currently producing nearly 0.1 TB of data. Once the algorithms for each of the objects are in place, the number of transients detected per night are expected to increase to $10^5\text{-}10^6$ producing nearly 30 TB of data with the Large Synoptic Sky Survey (LSST).

Due to the constantly increasing amount and complexity of the data collected from the Synoptic Sky surveys every night, which is on the order of Gigabytes, there is an increasing need for knowledge extraction as rapidly and efficiently as possible from these surveys. Discovering transients and classifying them is a complex process. Initially all transients appear same but in reality they could be entirely different physical phenomena.

Lightcurves can show tremendous variation in various characteristics such as temporal coverage, sampling rates, errors, missing values, etc. This makes comparisons between them difficult and training classifiers even harder. A way to approach this problem is to characterize a set of lightcurves via a set of common features and then use this representation of lightcurves for analysis and training of classifiers.

Many different types of features are used to extract information from the lightcurves such as flux and shape ratios, moments, variability indices, etc. From the lightcurves of known classes, mathematical representations of the best of these features can be derived using the SOM Toolbox in Matlab and then used as training sets for unsupervised classification of transients.

Work Done in the first month

Once the data from the survey is published for example at Skyalert.org, if some objects are found to be of particular interest and need another observation, finding charts prove to be a useful and crucial tool for the observers. As practice a workflow was designed for producing finding charts from datasets from observations of different areas of the sky. The computer code produces finding charts and links them to a web page. Finding charts can directly be generated from more such datasets customized for a specified magnitude range or for a specified field of view in the sky around the desired object.

Various papers were read related to the method of classification of transients and machine learning as mentioned in the references below.

The coding is a part of the learning process which includes getting acquainted with scripting using Python, MySQL, UNIX. Using Python to handle MySQL databases and working in a UNIX environment on Windows have been learnt.

- The first task undertaken was to learn to work on a UNIX environment inside Windows. Xming which provides the X Window System display server for graphical output was used for dynamic graphical output.

- Next task was to learn how to use MySQL to create databases, tables and implementing various techniques to extract and store data into/from them.

- The next task was to learn how to code using Python as the programming language. Some simple programs were created first to get used to the language.

- The last task was to use Python to place queries in MySQL and to reproduce selected and relevant information from the query in the form of a figure using the matplotlib plotting library from Python and a table on an HTML page.

- These skills proved to be useful for work in the next month.

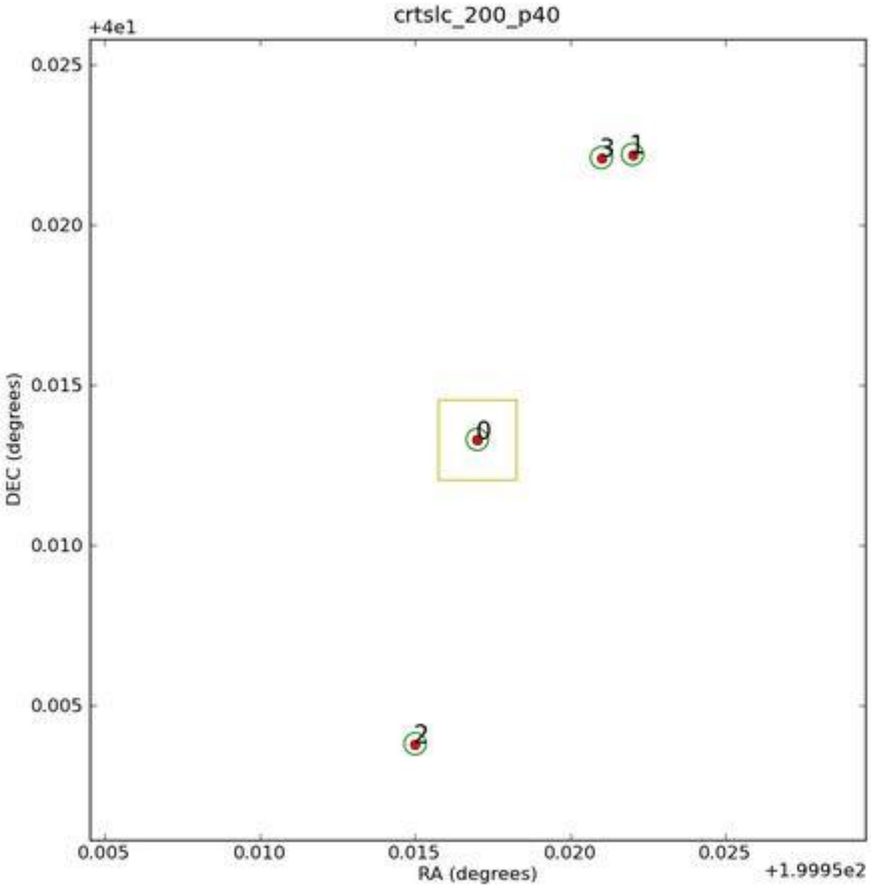
The process for producing the charts from the datasets is as follows:

→ From the .csv file of the dataset, import data into a MySQL table.
→ From the table select objects with unique MasterIDs and export those to another .csv file.
→ From this .csv file, import data into the final MySQL table.
→ Take user input for various parameters such as the dataset name, axes range, RA, Dec, number of objects of interest.
→ From the final table select objects according to the above specifications and store them in an array for further analysis.
→ The steps above are required because the current datasets do not exist in the working directory.
→ Calculate delra, deldec and position angle (PA) of the objects of interest relative to the object in consideration.
→ Plot the object in consideration at the center marked by a square and plot the other objects around it satisfying the above criteria marked by circles.

→ Create a table showing MasterID, Mag, Magerr, RA, Dec, delra, deldec and PA of the objects in the plot.

→ Export the generated figure and table with proper identification tags to an HTML web page with a link from www.astro.caltech.edu.

The finding charts were made for 10 datasets. The tables and figures were generated automatically through the code with just the required input for dataset name, RA and Dec. One of them is shown here:



Problems Encountered

Various issues arose while developing the code. Some of the commands are not supported anymore and have become obsolete, so after searching for them and implementing them, the code generated errors which had to be rectified by finding a way around them or by substitution with a different command.

While plotting with plt.show() command and saving the figure simultaneously, the figure did not display on the HTML page.

The axes of the plot were getting auto scaled if the limits were not set. If the limits were set as variance around a certain point for example, (ra – 0.0125, ra + 0.0125, dec – 0.0125, dec + 0.0125), the axes couldn’t get set, but if the exact number was specified as a variable for example, (ra1, ra2, dec1, dec2) the axes got scaled properly.

After making a table to export the data into an HTML page, the first entry is meant to be the object in consideration even though the table is arranged in descending order of magnitude i.e. the fainter ones first, so some loops are needed to check and assign the right values according to the need.

The SOM Toolbox

Self-Organizing Map (SOM) is an unsupervised neural network method which has properties of both vector quantization and vector projection algorithms. The prototype vectors are positioned on a low-dimensioned grid in an ordered fashion, making SOM a powerful visualization tool. The toolbox can be used to preprocess data, initialize and train SOMs using a range of different kinds of topologies, visualize SOMs in various ways and analyze the properties of the SOMs and the data, e.g., SOM quality, clusters on the map, and correlations between variables (parameters) etc.

A SOM consists of neurons organized on a regular low-dimensional grid represented by a d-dimensional weight vector (codebook vector) where d is equal to the dimension of the input vectors. The neurons are connected to adjacent neurons by a neighborhood relation, which dictates the topology of the map. The SOM training algorithm resembles vector quantization algorithms such as k-means with the distinction that in addition to the best-matching weight vector, it's topological neighbors on the map are also updated, i.e., the region around the best-matching vector is stretched towards the present training sample with the result that the neurons on the grid become ordered and which leads to clustering.

There are two training algorithms – Sequential and Batch. For the purpose of this project, the batch training algorithm is used because of the large number of data points in the dataset which reduces computational time significantly.

Implementation of SOM Toolbox in this Project

The main aim of this project is to train the classifier with a subset of parameters coming out from feature selection algorithms (Donalek et al) so that the best classification of transients is achieved. For this purpose a dataset (set of features extracted from the Caltech Time Series Charaterization Service, Graham et al) with known classes of objects is used. Different combinations of parameters are used to train the SOM with selected classes of objects to produce various visualizations which can then be compared for correlation and analyzed further for acceptance or rejection.

In unsupervised learning, the model is not provided with the correct results during the training. The model can be used to cluster the input data in classes on the basis of their statistical properties only. After the clustering, known labels are assigned to an object which helps to identify clusters. Once the map is trained and labelled, it learns to recognize groups of similar input vectors in such a way that neurons physically near each other in the neuron layer respond to similar input vectors.

Work Done in the second month

Initially, I read the manual of the toolbox thoroughly and undertook different demos and tutorials to get used to the working of the toolbox. I read different articles on Artificial Neural Networks and SOMs to get an idea of how they are implemented.

Next, we fed the SOM toolbox a dataset containing about 20 parameters and 6 classes of objects. Initially, all the parameters were plotted against each other for all classes as well as SN vs All Classes (combined as one class) and plotted as in Figure 1 and 2 respectively. These plots show only the graphs between the first 8 parameters, it is actually a matrix of graphs where each parameter is plotted against the other to search for correlation among them.

All the other classes are combined together and SN are separately plotted against them so that it is easier to work on the first node and identify relevant parameters for selecting and classifying SN from the dataset first.

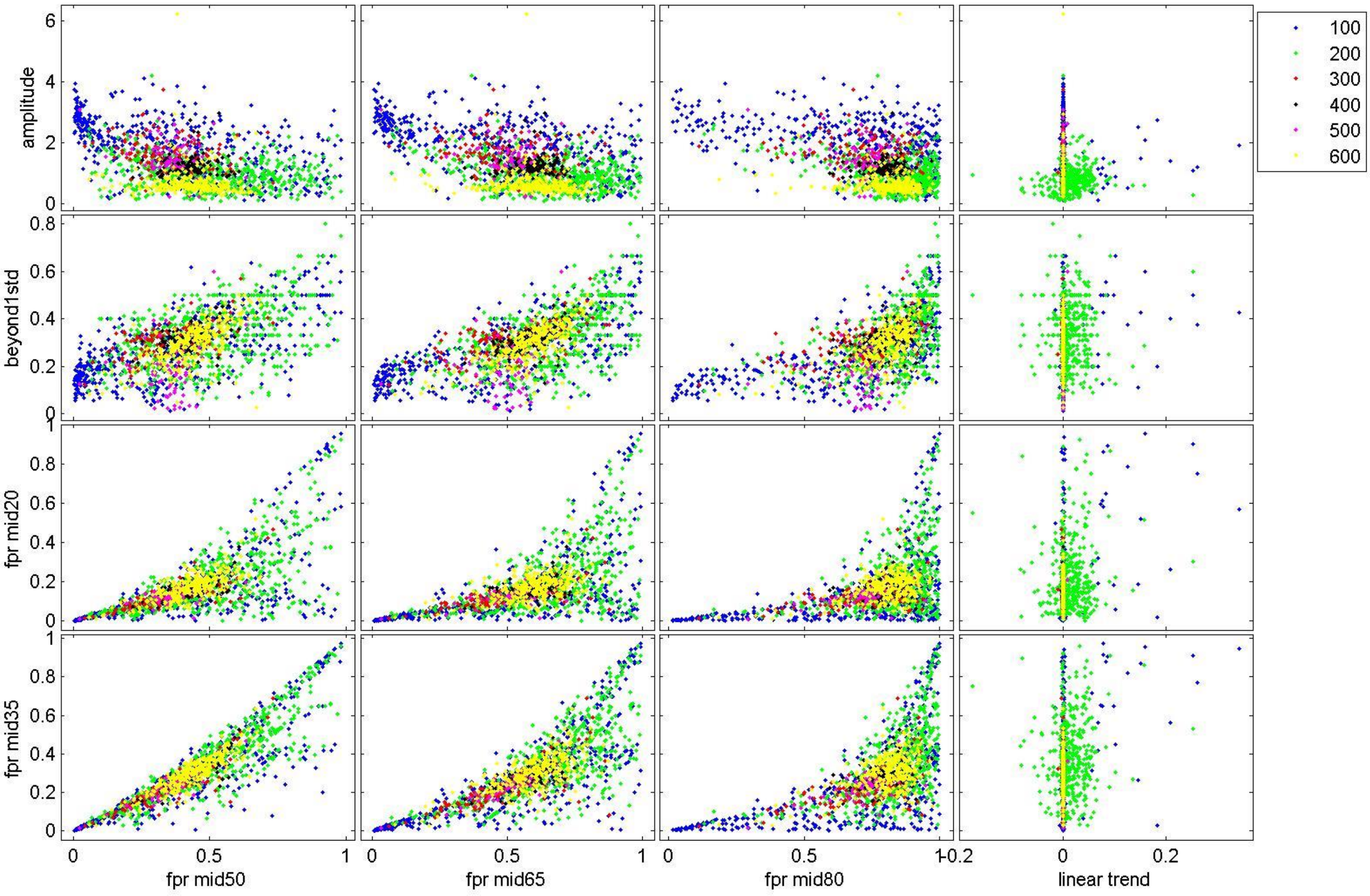


Figure 1: A plotmatrix showing the scatter plots when different parameters are plotted against each other for all the 6 classes of objects in the dataset as mentioned below, used to find those parameters which provide good clustering of the data points. (100 = CV, 200 = SN, 300 = BI, 400 = AGN, 500 = Flare, 600 = RR Lyrae)

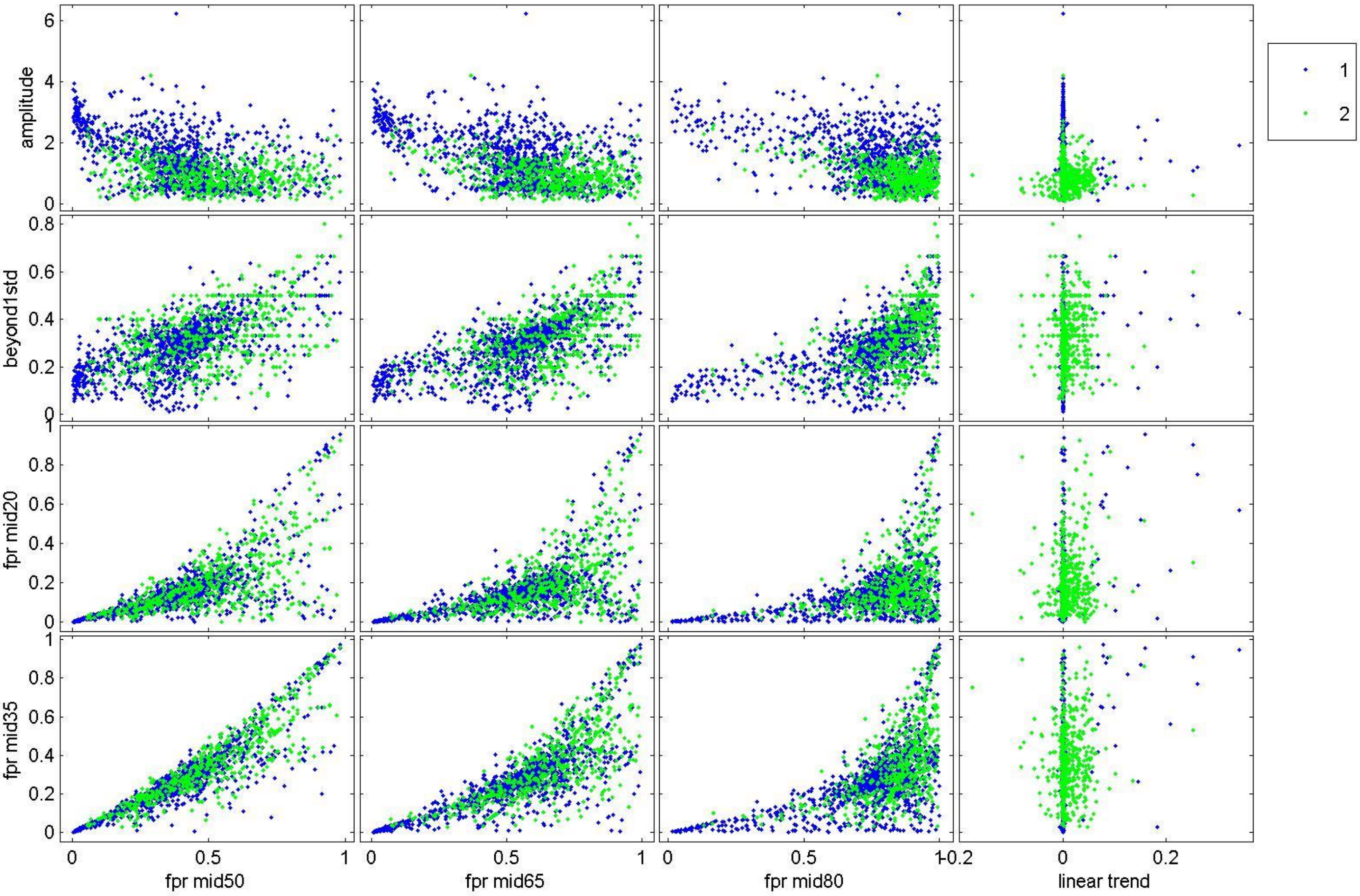


Figure 2: A plotmatrix showing the scatter plots when different parameters are plotted against each other for SN vs all other classes of objects in the dataset as mentioned below, used to find those parameters which provide good clustering of the data points.
(1 = Other Classes, 2 = SN)

Then we trained the map with all the parameters for SN vs All Other Classes in the dataset and visualized to select the best correlated parameters which could be used for further analysis. The visualization looks as in Figure 3.

The U-matrix (unified distance matrix) is a representation of SOM where the Euclidean distance between the codebook vectors of neighboring neurons is depicted. It is used to visualize high-dimensional data.

By component plane representation we can visualize the relative component distributions of the input data. Component plane representation can be thought as a sliced version of the Self-Organizing Map. Each component plane has the relative distribution of one data vector component. By comparing component planes we can see if two components correlate. If the outlook is similar, the components strongly correlate.

From the visualization in Figure 3 and the graphs similar to Figure 2, we selected certain parameters which were best correlated for classification of SN vs All Other Classes. We then cross-matched these parameters with parameters from another study and then visualized them separately for performance check.

Description of all the parameters can be found on this link[7].

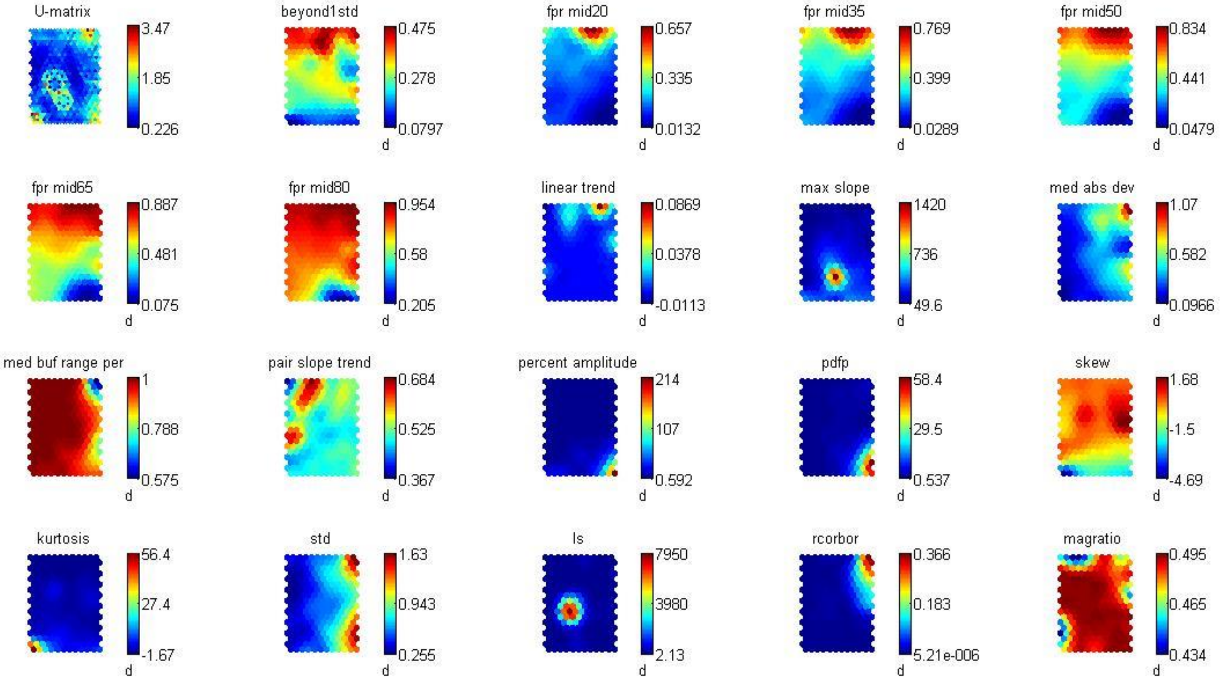


Figure 3: A SOM visualization showing the U-matrix and the different component planes representing the various parameters used to train the map and the clustering obtained for all classes of objects.

The parameters chosen and analyzed for correlation are explained through the following plots:

1. Parameters: beyond 1 std, linear trend and percent amplitude

The first set consisted three parameters i.e. beyond 1 std (Percentage of points beyond one standard deviation from the weighted mean), linear trend (Slope of a linear fit to the light curve) and percent amplitude (Largest percentage difference between either the maximum or minimum flux and the median) obtained from the feature selection algorithm (Donalek et al). The U-matrix and component planes were plotted and it was found that there was no correlation among these parameters as can be seen from Figure 4.

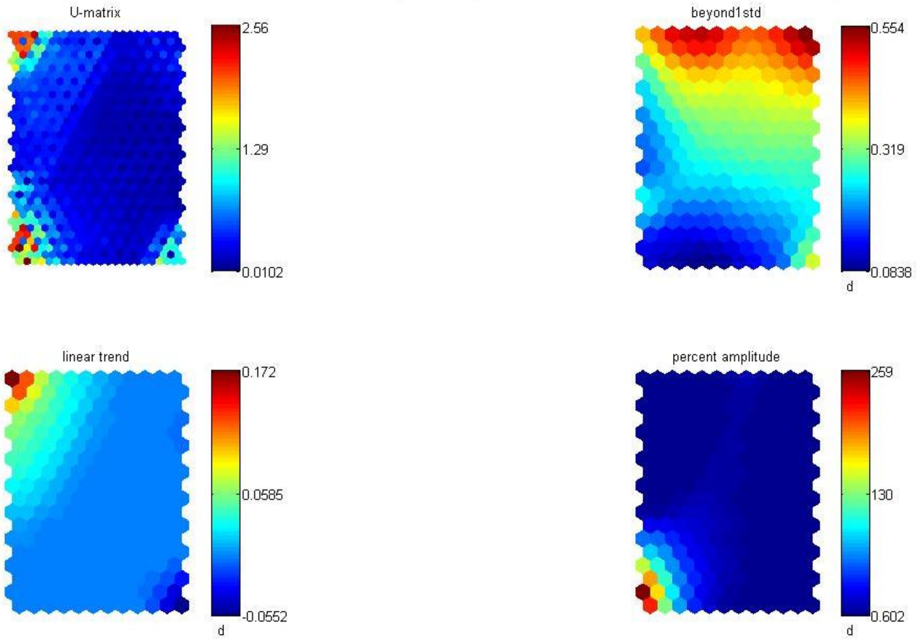


Figure 4: A SOM visualization showing the U-matrix and component planes for the three parameters in consideration.

The following figure shows the distribution and clustering of SN as compared to objects in other classes taken as one single class for the parameters in consideration.

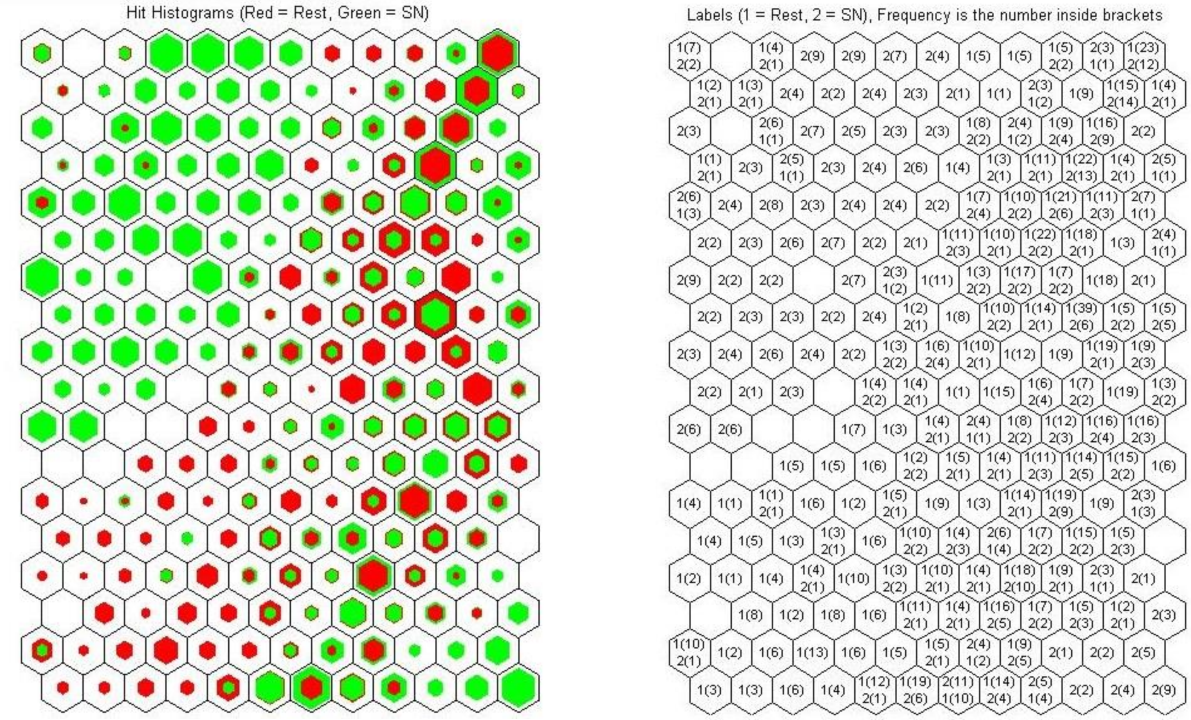


Figure 5: A SOM visualization showing the number of distinct objects in each map unit when there are two classes of objects i.e. SN and All Other Classes (combined) through Hit Histograms and Frequency Plots. We can see that there is clustering in the upper left portion and the bottom right portion of the map.

2. Parameters: amplitude, fpr mid20, linear trend and pdfp

The second set of four parameters consisted of amplitude (Half the difference between the maximum and minimum magnitudes), fpr mid20 (Ratio of flux percentiles (60th - 40th) over (95th - 5th)), linear trend (Slope of a linear fit to the light curve) and pdfp (Ratio of (95th - 5th) flux percentile over the median flux) obtained from the feature selection algorithm (Donalek et al). From the component planes, it can be seen that amplitude and pdfp are slightly correlated. Also, it is seen that there is a moderate correlation between fpr mid20 and linear trend.

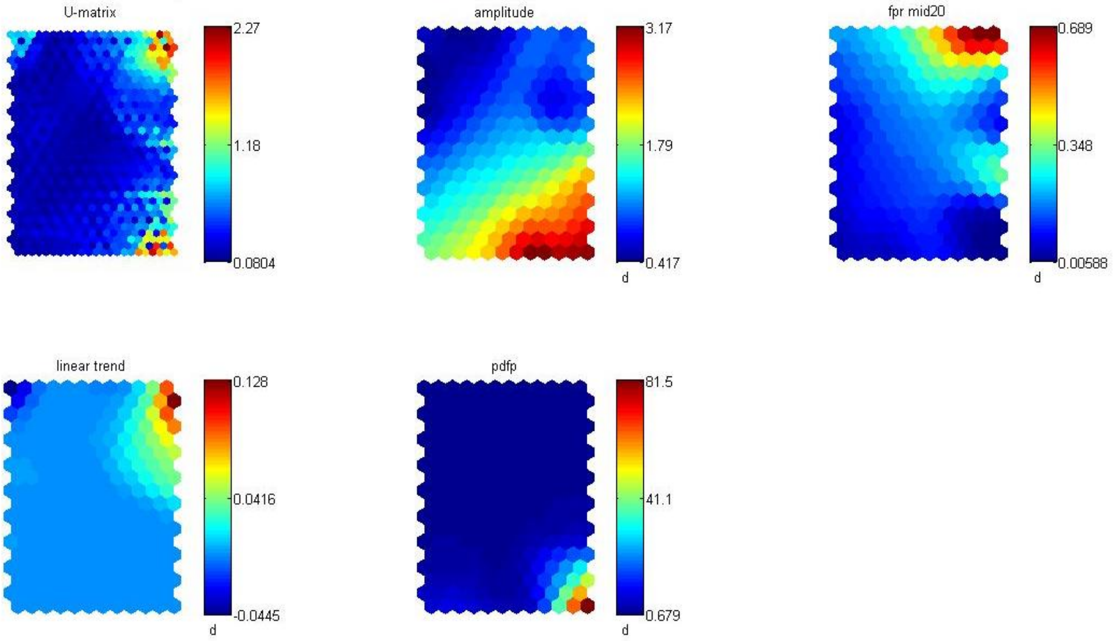


Figure 6: A SOM visualization showing the U-matrix and component planes for the four parameters in consideration.

The hit histograms and the frequency of occurrence plot are shown in Figure 7 for the four parameters used for training the map.

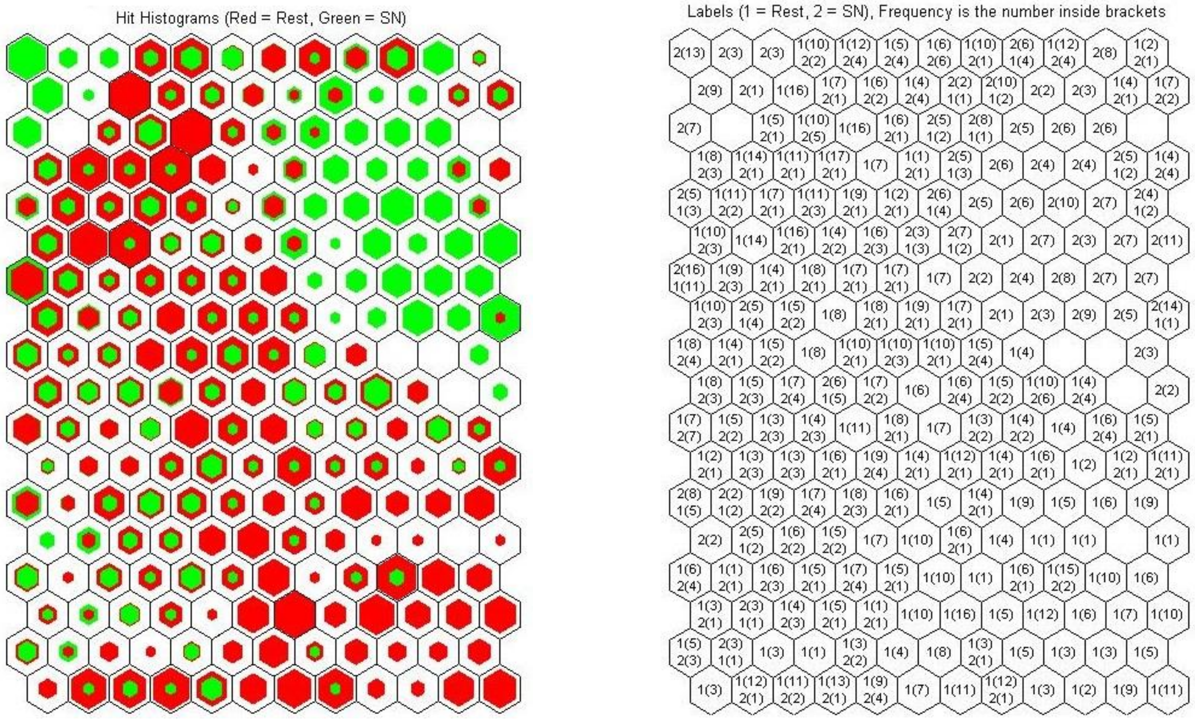


Figure 7: A SOM visualization showing the number of distinct objects in each map unit when there are two classes of objects i.e. SN and All Other Classes (combined) through Hit Histograms and Frequency Plots. We can see that there is a clustering of SN in the upper right portion of the map.

3. Parameters: kurtosis, pdfp, percent amplitude and max slope

The last set consisted of four parameters namely: kurtosis (measure of the "peakedness" of the probability distribution of a real-valued random variable), pdfp, percent amplitude and max slope (Maximum absolute flux slope between two consecutive observations). It is seen from the component planes that there is a correlation between Kurtosis and max slope. Also a correlation can be seen between pdfp and percent amplitude.

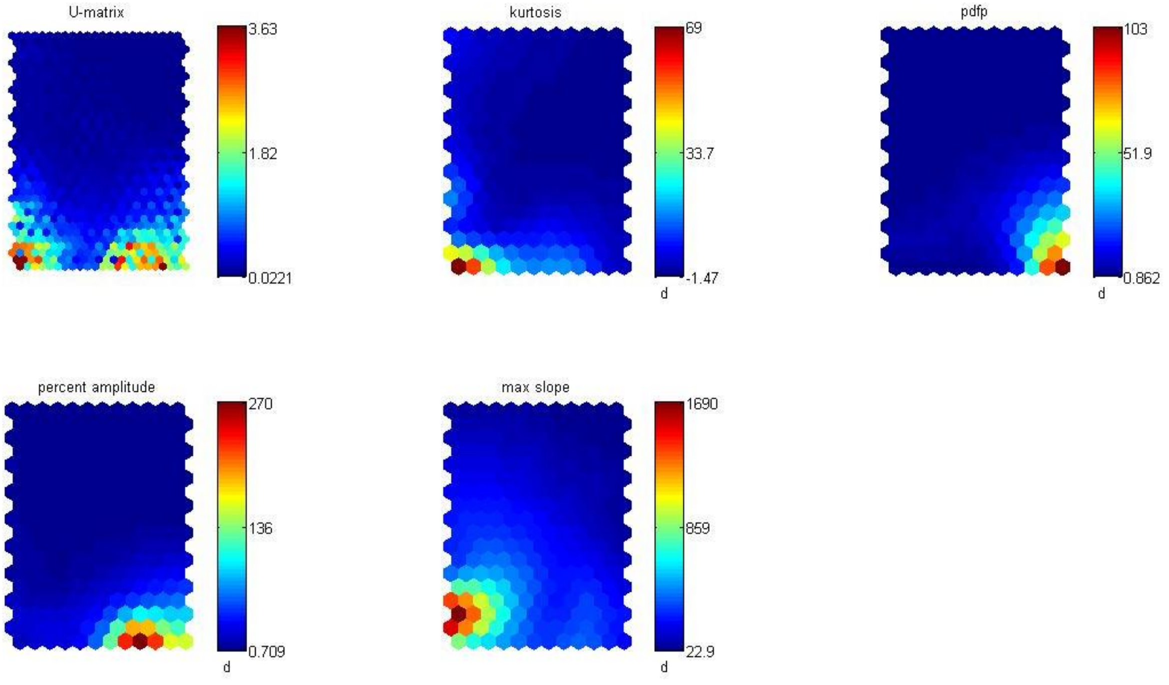


Figure 8: A SOM visualization showing the U-matrix and component planes for the four parameters in consideration.

The hit histograms and the frequency of occurrence plot are shown in Figure 9 for the last four parameters used for training the map.

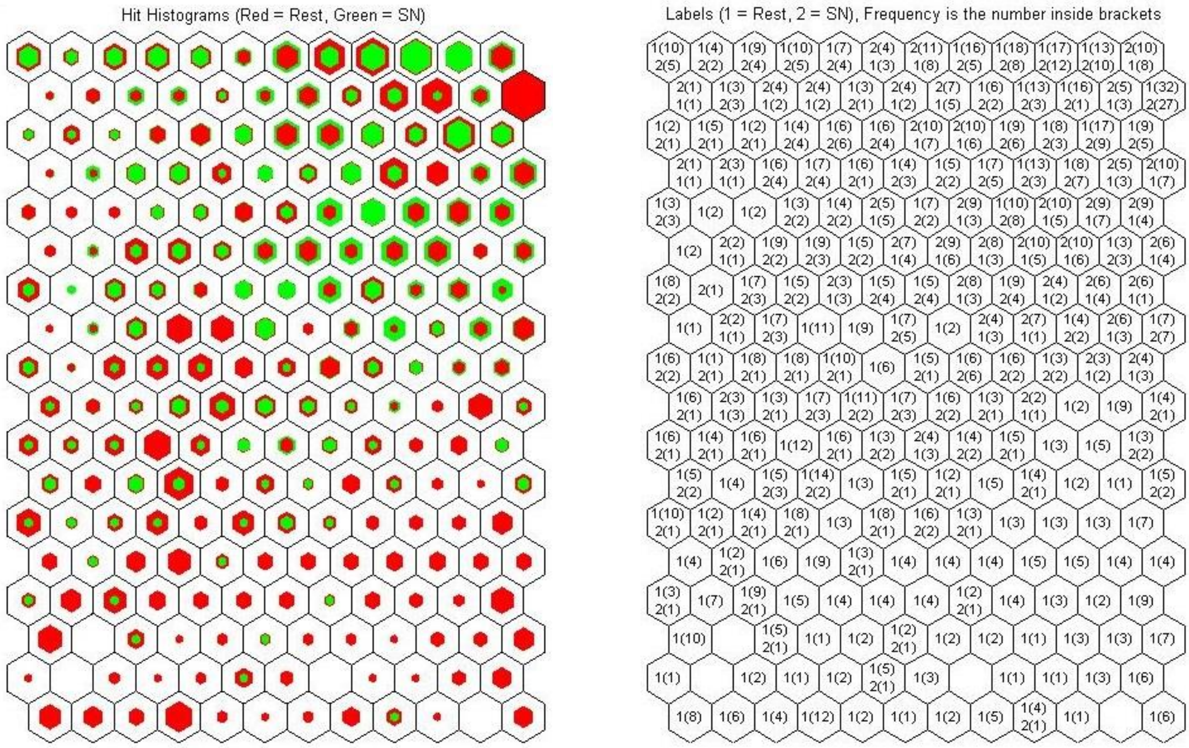


Figure 9: A SOM visualization showing the number of distinct objects in each map unit when there are two classes of objects i.e. SN and All Other Classes (combined) through Hit Histograms and Frequency Plots. We can see that there is no clustering of SN as compared to all other classes.

Observations and Conclusions

We can observe that the first set of parameters show no correlation which means they are good for clustering. The parameters in the second case also show very less correlation in specific cases of comparison. The parameters in the third case are highly correlated which means they will produce almost the same results when used for classification. We can see two separate clusters in Figure 5 and some mixing on the diagonal from top right to bottom left. This is good clustering. In Figure 7, there is a single distinguishable cluster and the rest of the objects seem to be mixed up. In Figure 9, it is difficult to identify a single cluster from the map which shows that there is no clustering at all when the third set of parameters is used.

Problems Encountered

Most of the problems that I encountered were almost always scripting related which were solved after some deliberations. Sometimes the program produced unexpected results which were finally rectified after a better understanding of the working of the toolbox.

Future Work

The project was a learning phase for me in the ten weeks that I worked and most of the work done was qualitative in nature. The results and interpretations obtained from this work need to be quantified which can be taken up as future work for which I will continue receiving different sets of parameters from feature selection algorithms and use them to train SOMs to find out the correlation between them and the clustering that they perform.

References

[1] arXiv:1211.3607 [astro-ph.IM] Classification by Boosting Differences in Input Vectors

[2] arXiv:0712.3797 [astro-ph] Variable stars across the observational HR diagram

[3] arXiv:1111.0313 [astro-ph.IM] Discovery, classification, and scientific exploration of transient events from the Catalina Real-time Transient Survey

[4] http://www.lsst.org/lsst/scibook LSST Science Book

[5] http://www.astro.caltech.edu/~donalek/bi199/bi199_2013.pdf Matlab Tutorial by Ciro Donalek

[6] http://www.cis.hut.fi/somtoolbox/package/papers/techrep.pdf SOM Toolbox manual

[7] http://nirgun.caltech.edu:8000/scripts/description.html Caltech Time Series Characteriza -tion Service