Application of Supervised Machine Learning to Classification of Variable Young Stars

Philip Carr

Mentor: Lynne Hillenbrand

Abstract

Young star systems are crucial to the understanding of how planetary systems form, providing a window into the dynamics and short-lived behavior in some of the earliest stages of planetary system development. Young stars are known for the variety of distinct variability patterns in their light curves. Previous research has demonstrated the classification of young stars into eight variability types, largely based on the degree of periodicity and asymmetry in flux of the light curves, that could be used for further studies of specific variability types. As the size and scope of photometric variability surveys increase in the future, classification of these young stars will become highly favorable for computers to automate. A selection of supervised machine learning algorithms has been applied to the classification of young stars, previously-defined features have been tested to be robust for supervised learning, and new features have been developed for the classification of young stars. The random forest algorithm currently performs with the highest overall accuracy of 75 ± 5 % and balanced accuracy of 75 ± 6 %. While not an ideal performance for the direct application to unclassified young stars as of yet, the use of machine learning algorithms for the classification of these objects has been shown to be potentially viable.

Introduction

One of the fundamental questions of astronomy is the question of how planetary systems form. In particular, young star systems provide a unique and vital look into the properties and behavior of planetary system formation occurring during one of the star system's most dynamic periods of evolution. Many young stars exist with circumstellar disks containing once interstellar material that did not form the original star, providing the necessary matter for planets to form. The circumstellar disks that occasionally surround young stars may accrete to some extent onto the star, accounting for up to 10% of the star's total mass by the time the star becomes a main sequence star, adding an additional factor to the dynamics of young star systems. Young stars exist for only millions of years, compared to their total lifetimes of billions of years, meaning that young stars are short-lived but crucial to understanding of how planetary systems develop.

One important behavior that is common among young stars is their variability in brightness (flux). Many young stars show periodic variability in their light curves on the timescale of 0.5 – 20 days, which indicates the rotation of the star, and many others show aperiodic variability on the time scale of hours to years, which indicates circumstellar disk accretion activity. Factors such as a star's intrinsic properties, circumstellar disk accretion that causes flux increases, and other circumstellar disk-related activity that causes flux dimming can contribute to the aperiodic variability of young stars. Young stars also range in variable flux amplitude, from milli-magnitudes to several magnitudes.

The light curves of young stars exhibit a significant amount of variety in their variability patterns. These patterns in variability can indicate certain physical properties and dynamics of the star, as well as its circumstellar disk accretion activity. A study done by Hillenbrand and Cody 2018 developed a classification system of young stars based on their light curves,

assigning stars to one of eight distinct variability types. These variability types exist largely based on the degree of periodicity and flux asymmetry that the light curves exhibit. The degree of periodicity, quasi-periodicity (Q), is a measure of the regularity of a young star's period. This value ranges from 0 to 1, with low Q ($Q < 0.16$) indicating a highly regular period (periodic), with $0.16 < Q < 0.8$ indicating a semi-regular period (quasi-periodic), and with high Q ($Q > 0.8$) indicating a highly irregular/no period (aperiodic). Flux asymmetry (M), which usually ranges from values between -1.5 and 1.5, indicates the asymmetry between flux peaks below and above the median flux, with low M ($M < -0.25$) indicating greater-magnitude peaks above the median than below the median (bursting), with $-0.25 < M < 0.25$ indicating roughly equal-magnitude peaks above the median and below the median (symmetric), and with high M ($M > 0.25$) indicating greater-magnitude peaks below the median than above the median (dipping). By defining a classification system for variable young stars, more can be understood about the variety of ways in which young stars may develop, and future studies would be able to isolate certain variability types of young stars in order to further understand the unique physical properties of certain variability types.

Table 1

Categories of young star system light curves

Category	Quasi-periodicity	Flux Asymmetry
Periodic	Periodic	Symmetric
Multiperiodic	Quasi-periodic	Symmetric
Quasi-periodic symmetric	Quasi-periodic	Symmetric
Burster	Aperiodic, quasi-periodic	Bursting
Aperiodic dipper	Aperiodic	Dipping
Quasi-periodic dipper	Quasi-periodic	Dipping
Stochastic	Aperiodic	Symmetric
Long Timescale	Aperiodic	Symmetric, dipping

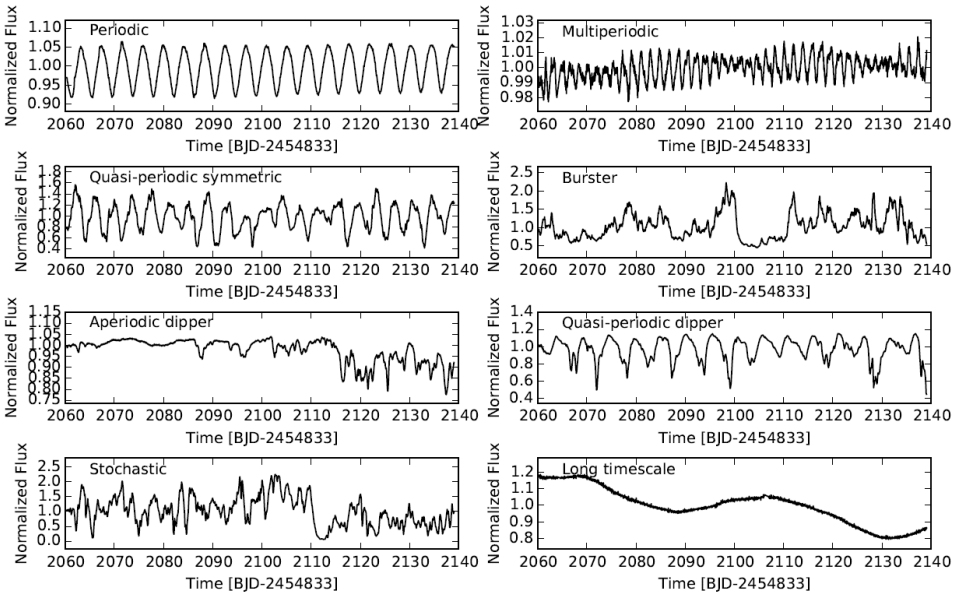

Figure 1. Examples of variable light curves (adapted from Cody, A. M. & Hillenbrand, L. A. 2018).

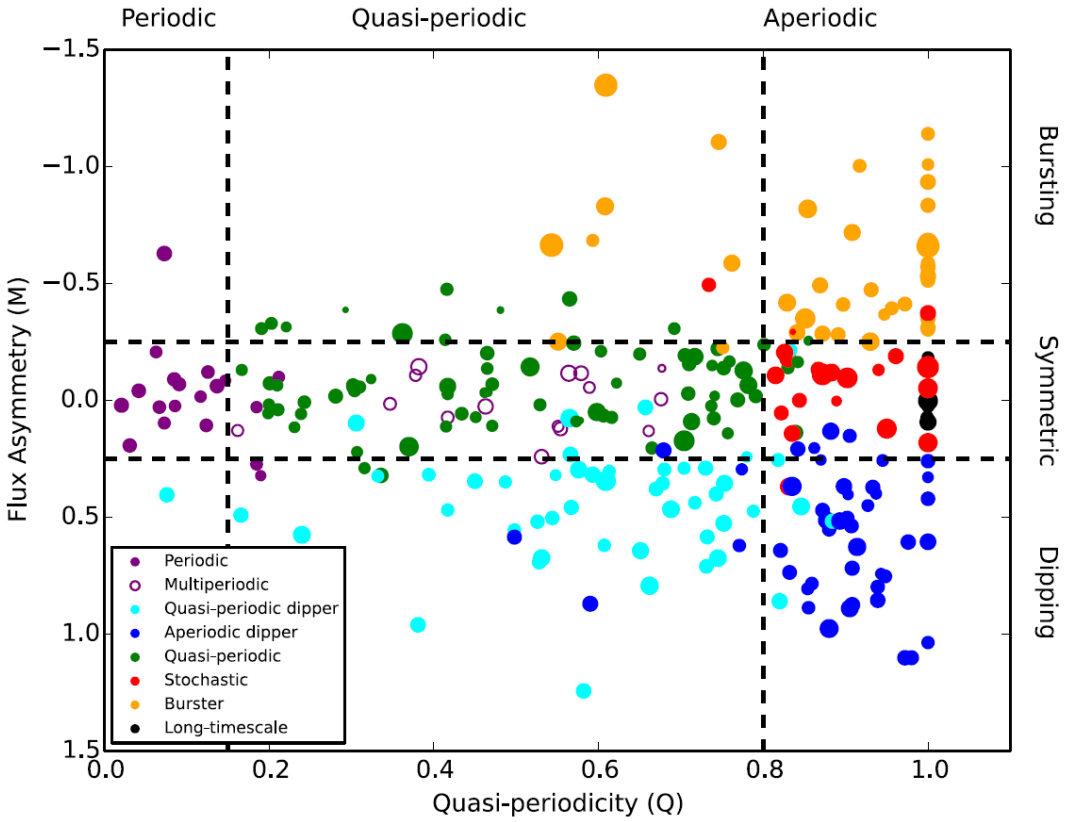

Figure 2. Plot of periodicity vs. flux asymmetry of variable young stars (adapted from Cody, A. M. & Hillenbrand, L. A. 2018).

In recent years, the field of astronomy has seen significant increases in the size of data sets that have been obtained using large-scale telescope surveys, and it is expected that the amount of data generated in future surveys will continue to increase quickly. As the amount of data used in astronomy continues to increase, the ability of humans to be able to analyze individual survey objects manually in an efficient and effective manner decreases due to the vast amounts of data available for analysis. Meanwhile, the use of computers to automate analysis routines for survey data increases, as computers could perform analysis tasks such as object classification in a fraction of the amount of time it would take a team of humans to complete manually. Thus, for the classification done for the variable young stars into the eight variability types as described above, the use of computers to automate this analysis is highly favorable.

For analysis tasks such as regression, dimensionality reduction, and in particular, classification, computers can be used to utilize machine learning methods in order to complete these tasks and more. Machine learning is a domain of artificial intelligence that uses statistical methods to learn patterns from data, and then uses these findings to make certain conclusions about data. Machine learning can be divided into two broad categories: supervised learning and unsupervised learning. Supervised learning methods learn from data that is labelled, and the labels that are assigned to data for the machine learning algorithm used are meant to directly guide the way in which the machine learning algorithm learns from the data. Unsupervised learning, on the other hand, learns from unlabeled data and makes different kinds of predictions from supervised learning methods. Classification is a supervised learning task, in which a classification algorithm (classifier) learns how to differentiate between data of different labels, and then is able to apply its classification model on unlabeled data.

The use of machine learning in astronomy has become an increasing presence over the last several years, showing the general ways that machine learning can help perform existing analysis tasks more efficiently and entirely new analysis tasks as well. Photometric variability surveys such as the Palomar Transient Factory (PTF), Zwicky Transient Facility (ZTF), and Large Synoptic Survey Telescope (LSST) have applied machine learning techniques to help process large data sets by developing predictive models to generate characteristic groupings of data that could not have been done manually (Bloom, J. S. & Richards, J. W. 2012). Additionally, in the field of young stars, some progress has been made to apply machine learning to the classification of young star systems. A machine learning algorithm developed by Richards J. W., et al. showed the potential to classify variable stars using time-series data. However, this algorithm was only able to classify T Tauri stars with an accuracy of 7%, so there exists much more improvement in the classification of young stars, especially T Tauri stars, which are known for exhibiting aperiodic features in their light curves.

Methods

Software for this project was developed in a directory containing Jupyter notebook files and separate subdirectories to contain the data used in this project as well as separate python files for various data analysis and machine learning purposes. Development for the software of this project used Anaconda running with Python 3.6.5. A variety of files were developed for this project, including Python source code files, as well as several Jupyter notebook files for interactive machine learning classifier testing and the display of machine learning classifier results, plots of star light curves, and other visualizations. For the machine learning algorithms used for classification and other machine learning related-tools, the Scikit-learn package was used.

Data for this project was a selection of light curves and their associated classification labels of variable young stars obtained by the K2 mission of the Kepler Space Telescope. Furthermore, only light curves of young stars belonging to the eight classifications were used in machine learning classification (APD, B, L, MP, P, QPD, QPS, S).

Since the ultimate goal of this project was to develop software to apply machine learning to classify variable young stars, (supervised) classification algorithms (classifiers) were used in this project. In order to develop a program to classify objects, a specific set of steps was needed to complete the classification from start to completion.

The first step was the generation of data tables usable by a classifier. Classifiers learn about objects by looking at specific quantitative data about objects called features. Features are any kind of quantity derived from the raw data. With light curves, examples of features include periodicity and flux asymmetry, as well as other quantities such as flux mean and variance.

Classifiers directly use feature data to learn the relationships between objects and their assigned labels (also known as targets). For this project, a feature library was developed specifically for the classification of these young stars. The first four features used for classification were flux amplitude, timescale, periodicity, and flux asymmetry. In total, 28 features were developed (including the four previous features mentioned), some of which were more related to determining the periodic nature of a light curve, and others more related to determining the flux asymmetry nature. This feature library was known as the default feature library used for classification.

Table 2

Default feature library feature names

Feature Name	**Periodic vs. flux asymmetry nature**
Normalized flux amplitude	Flux asymmetry
Timescale	Periodic
Periodicity (Q)	Periodic
Flux Asymmetry (M)	Flux asymmetry
Mean	Flux asymmetry
Median absolute deviation	Flux asymmetry
Reduced chi^2 to mean	Flux asymmetry
Error-weighted mean	Flux asymmetry
Number of zero-point crossings	Periodic
Stetson k index	Flux asymmetry
Half-magnitude amplitude ratio	Flux asymmetry
Half-magnitude amplitude ratio 2	Flux asymmetry
Eta parameter	Flux asymmetry
10% slope percentile	Flux asymmetry
90% slope percentile	Flux asymmetry
Cumulative sum	Flux asymmetry
2-split mean variance	Periodic
3-split mean variance	Periodic
4-split mean variance	Periodic
2-split skew variance	Periodic
3-split skew variance	Periodic

4-split skew variance	Periodic
2-split normalized flux subsection residual sum	Periodic
3-split normalized flux subsection residual sum	Periodic
4-split normalized flux subsection residual sum	Periodic
Lomb-Scargle false alarm probability	Periodic
Smoothed light curve polynomial fit rms error	Periodic
Number of frequency peaks within 1/100th power of maximum power frequency peak	Periodic

In addition to the library of default features used in classification, there were two open-source feature libraries used in this project: FATS and feets. Both these feature libraries contain a total of over 60 features (64 and 67 respectively), but neither were specifically made to focus on the aperiodic nature that many of the young star light curves had.

After the tables of feature data and corresponding labels are made for the machine learning classifier to use, this set of data must then be divided into a training set and a test set. The training set is the set of data used by the classifier to learn how to classify different objects (known as fitting the classifier to the training data), and the test set is used to determine the prediction accuracy of the classifier with data that the classifier did not use to train itself. To split the data into training and test sets, the sklearn package was used to perform this operation. It is possible for a classifier to learn from a training set too large for the classifier to be able to determine general patterns within variability types and relationships between features and variability types. This effect, known as overfitting, is mitigated when the size of the training set is limited to some proportion less than 100% of the labelled data. For testing purposes in this project, the size of training sets used was 70% of the total labelled data.

With the training and test data available for a classifier, the classifier can now train on the training data. All classifiers in sklearn are implemented with the fit method, which causes the classifier to fit its learning model to given training data. In addition, classifiers are all implemented with a certain collection of user-variable hyperparameters, parameters for a classifier that specifically adjust certain aspects of how the given classifier's learning model functions. For example, the random forest algorithm operates by generating some number of decision trees determined by the hyperparameter n_estimators, uses some number of features to determine branch splits while building the decision tree determined by the hyperparameter max_features, and forms decision trees with a maximum depth determined by the hyperparameter max_depth. Since all classifiers are implemented with a substantial number of hyperparameters (e.g. at least 10 for both the random forest and linear svc classifiers), and since the variation of hyperparameters can mean a significant difference in the classifier's accuracy, it best to optimize a classifier's hyperparameters in order to obtain a set of hyperparameters that most effectively classifies objects.

Sklearn implements two methods for hyperparameter optimization: grid search and randomized search. Grid search performs a search over a set of multiple individual sets of hyperparameters, testing the classification performance of every available combination of given hyperparameter values. Randomized search searches by randomly sampling hyperparameter values over probability distributions of quantities for quantitative hyperparameter values, and randomly samples from sets of string values for qualitative hyperparameters specified by strings. Both grid search and randomized search have been included in the implementation of the machine learning classifier testing software for the random forest, support vector classifier (svc), and linear support vector classifier (linear svc). For other classifiers, including random forest,

svc, linear svc, k-nearest neighbors, decision tree, extremely randomized trees, and adaboost, multiple versions of these classifiers with varying hyperparameters have also been implemented in the machine learning classifier testing software.

Once a classifier is trained with a training set of labelled data, the classifier can then be tested by having the classifier predict the labels of the test set. Since the test set also contains labels corresponding to the feature data in this set, the test set can be used a model for how the classifier would classify unlabeled objects. For the machine learning classifier testing software, a function to test a classifier was created to run a given number of tests, each of which would randomly split the labelled data into training and test sets, fit the classifier to the training data, predict the labels of the test data, and plot the results of the classifier test, including the weighted accuracy, balanced accuracy, and mean confusion matrix of the set of confusion matrices generated from each individual test.

Several different Python notebook files (including multiple iterations of the same Python notebook files) were created for the purposes of diagnostic testing of certain parts of the machine learning process as well as the performance testing of different machine learning algorithms.

One notebook was created as a testing platform for developing different features. This notebook contains sample object names, imports their light curve data (array of times in Julian Date units, and array of flux data in magnitude units), allowing for an interactive environment for producing functions and plots to test on the given light curve data. Several features in the default feature set, including 2-split normalized flux subsection residual sum and smoothed light curve polynomial fit rms error, were developed with this notebook. More modules were developed to demonstrate and test the reading in of light curve data, as well as plotting the light curves of the stars.

Several additional notebooks were created for running the process of testing machine learning algorithms from reading in the light curve data to producing classifier performance testing results. Notebooks for specifically testing the random forest classifier and support vector classifier individually were developed. Another set of notebooks were made to test all classifiers implemented in the machine learning testing software library, including the classifiers optimized using grid search and randomized search, using the default features set, the FATS feature set, and the feets feature set, and using full light curve data as well as light curve data with the first few days of data removed (since some of the K2 light curves had errors introduced by the Kepler Telescope calibration process). Lastly, separate machine learning testing notebooks were created specifically to test the randomized search-implemented classifiers.

The remainder to development of the software library was in the source_code directory. The programs for reading in light curve data, creating feature data tables, and performance testing classifiers are located in the various python source code files in this directory.

Results

Random forest feature importances

The sklearn random forest algorithm can output the feature importance values used in each of the estimators (decision trees of the random forest), allowing for the observation of which features had the most classification power. Below are several bar graphs displaying the feature importances of the random forest algorithm.

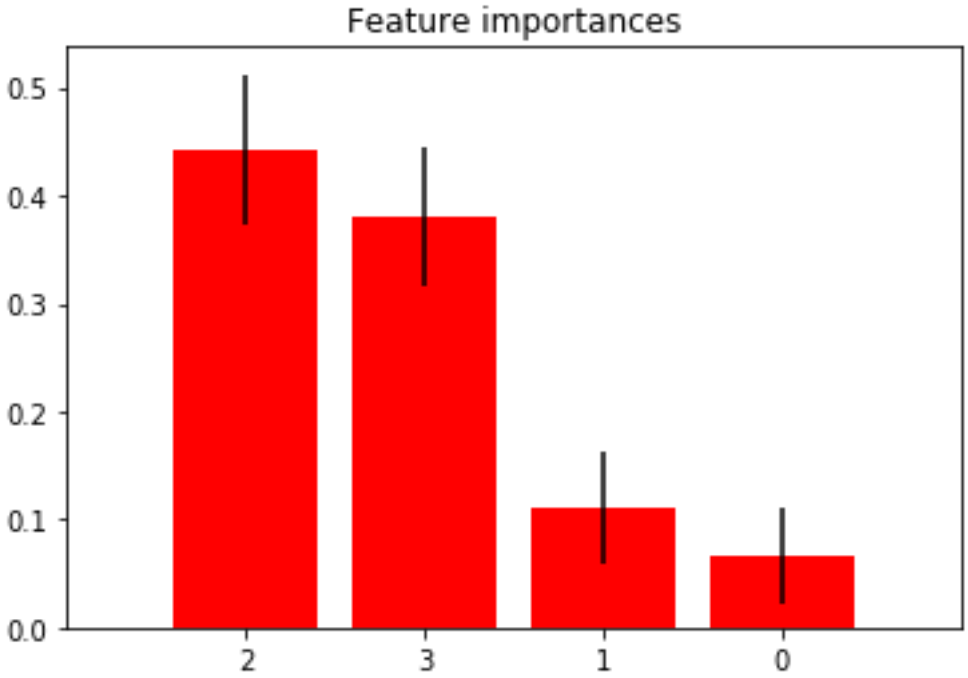

Figure 3. Feature importances from one trial of a random forest classifier using randomized hyperparameter optimization tested using the four features flux amplitude (0), timescale (1), periodicity (2), and flux amplitude (3) (total accuracy and balanced accuracy of 70.2%) (obtained from cell 13 in the notebook General Machine Learning Classification Testing (Version 5)).

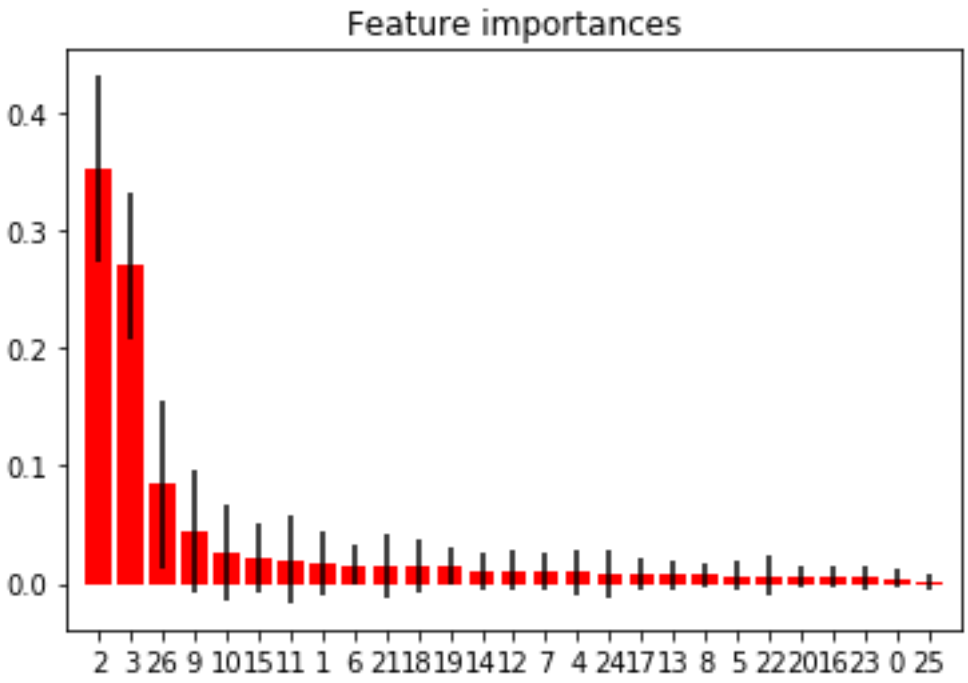

Figure 4. Feature importances from one trial of a random forest classifier using randomized hyperparameter optimization tested using the full default set of features as specified in the file feature_extraction.py (total accuracy and balanced accuracy of 71.0%) (obtained from cell 15 in the notebook General Machine Learning Classification Testing (Version 5)).

The two feature importance graphs above indicate that periodicity and flux asymmetry are the two features with the most classification power, followed by smoothed light curve polynomial fit rms error (26), Stetson k index (9), and half-magnitude amplitude ratio (10). This corresponds with Figure 2, which shows the significant separation between many of the variability types in the phase space of periodicity and flux asymmetry.

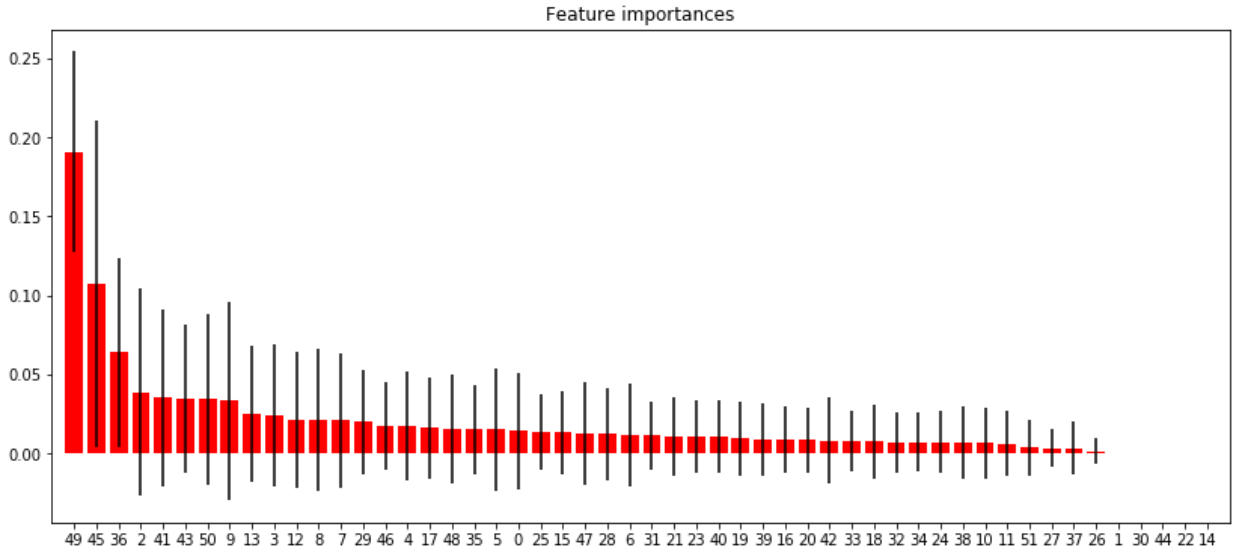

Figure 5. Feature importances from one trial of a random forest classifier using randomized hyperparameter optimization tested using the FATS features as specified in the file feature_extraction.py (total accuracy of 56.5%, balanced accuracy of 52.6%) (obtained from code cell 20 in the notebook Machine Learning Classification Testing Random Forest Default vs FATS vs feets (Version 3)).

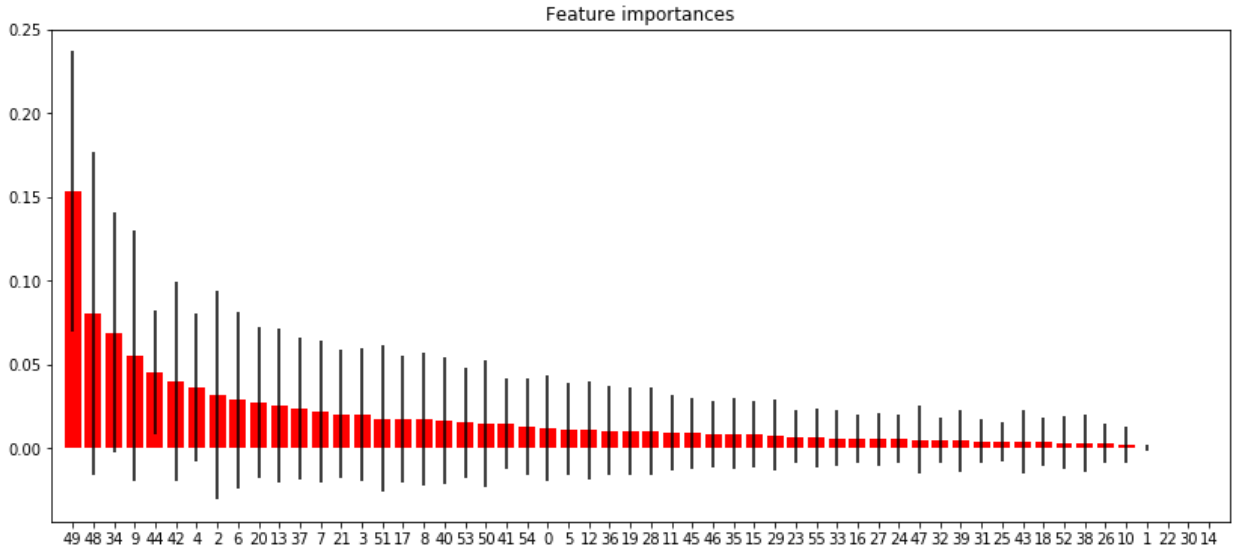

Figure 6. Feature importances from one trial of a random forest classifier tested using the feets features as specified in the file feature_extraction.py (total accuracy of 50.6%, balanced accuracy of 43.2%) (obtained from code cell in the notebook Machine Learning Classification Testing Random Forest Default vs FATS vs feets (Version 3)).

Default features vs FATS features vs feets features

Below are mean confusion matrices for the same random forest algorithm tested with selections of features of each of the three sets of features. The mean confusion matrix contains the mean values for each element in the confusion matrix averaged for 100 trials of test set prediction by the classifier. The random forest algorithm performs with the highest total accuracy and highest balanced accuracy using the set of default features used below (flux amplitude, timescale, periodicity, and flux asymmetry).

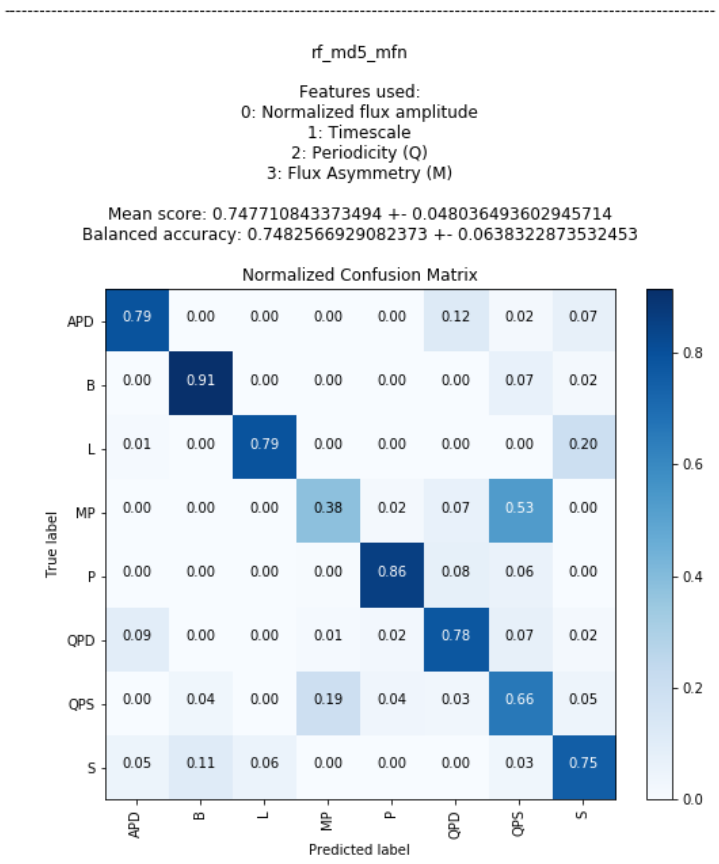

Figure 7. Mean confusion matrix of the random forest algorithm using four features from the default features set. (This is the result given in the abstract). (Obtained from code cell of the notebook Machine Learning Classification Testing Random Forest Default vs FATS vs feets (Version 3)).

For the above random forest classifier test result, the random forest algorithm produces somewhat high accuracies for each of the variability types (looking across the diagonal of the matrix) except for the multiperiodic objects. This is because the multiperiodic and quasi-periodic symmetric objects are confused for each other, with 50% of the multiperiodic objects being confused for quasi-periodic objects, and over 10% of the quasi-periodic objects being confused for multiperiodic objects. Additionally, the random forest algorithm tends to confuse aperiodic dipper objects and quasi-periodic dipper objects for each other, long timescale objects for stochastic objects, and stochastic objects for burster objects.

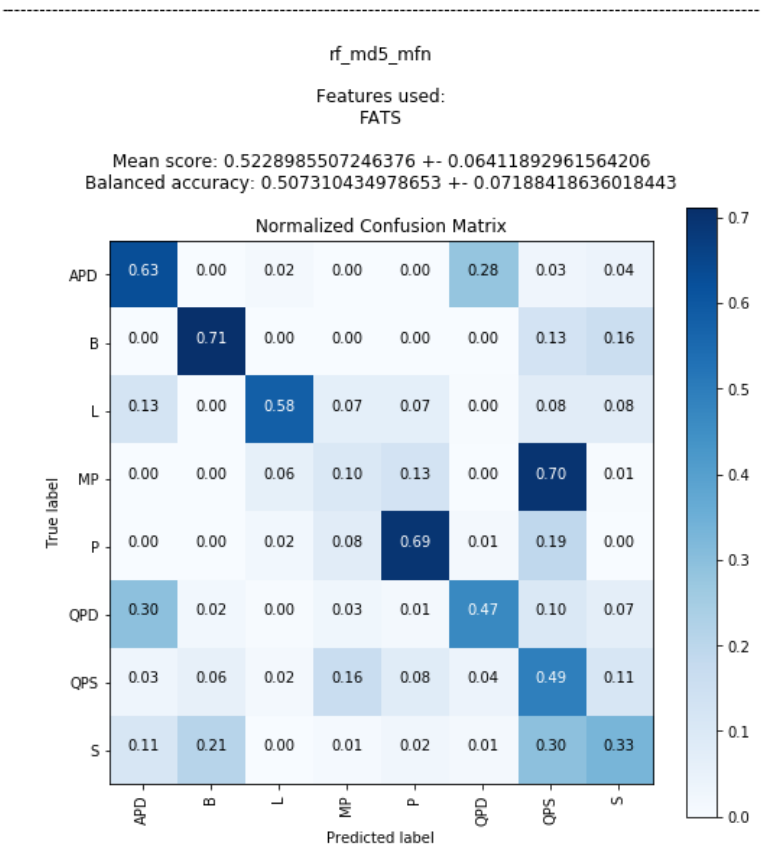

Figure 8. Mean confusion matrix of the random forest algorithm using the ten features with the greatest classification power from the FATS features set (see Figure 5). (Obtained from code cell 17 of the notebook Machine Learning Classification Testing Random Forest Default vs FATS vs feets (Version 3)).

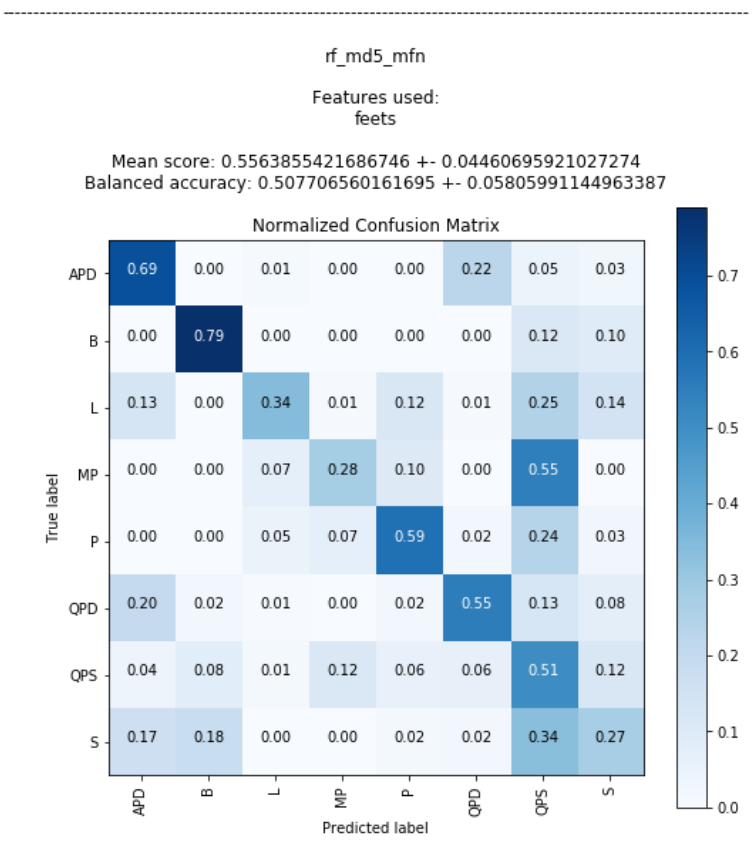

Figure 9. Mean confusion matrix of the random forest algorithm using the ten features with the greatest classification power from the feets features set (see Figure 6). (Obtained from code cell 22 of the notebook Machine Learning Classification Testing Random Forest Default vs FATS vs feets (Version 3)).

Other Notable classifier results

Below are some of the classifier testing results with the some of the highest total accuracies and balanced accuracies. All confusion matrices from the classifier testing results below were averaged for 100 trials.

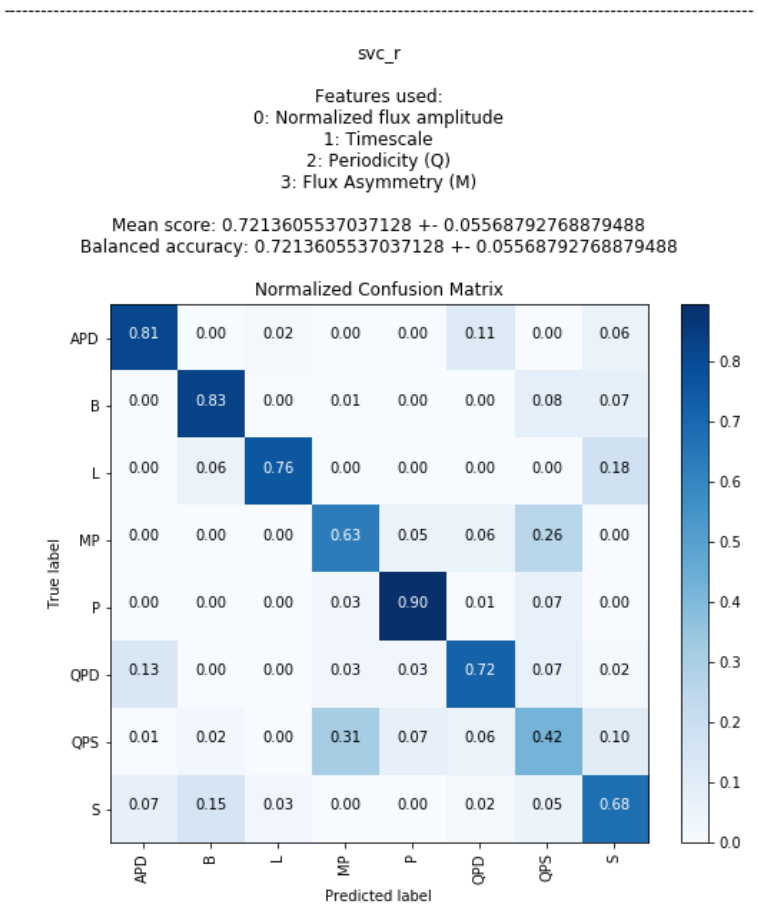

Figure 10. Mean confusion matrix of the svc algorithm (rbf kernel) using using randomized search hyperparameter optimization four of the default features. (Obtained from code cell 4 of the notebook Machine Learning Classification Testing SVC (linear kernel) Randomized Search (Version 2)).

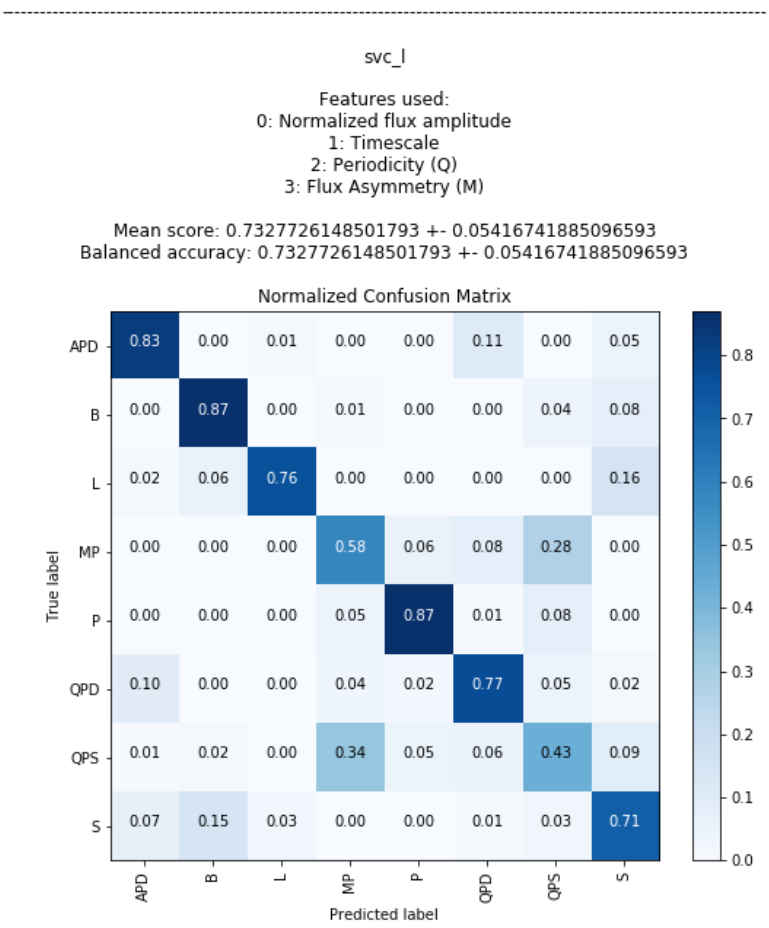

Figure 11. Mean confusion matrix of the svc algorithm (linear kernel) using using randomized search hyperparameter optimization four of the default features. (Obtained from code cell 4 of the notebook Machine Learning Classification Testing SVC (linear kernel) Randomized Search (Version 2)).

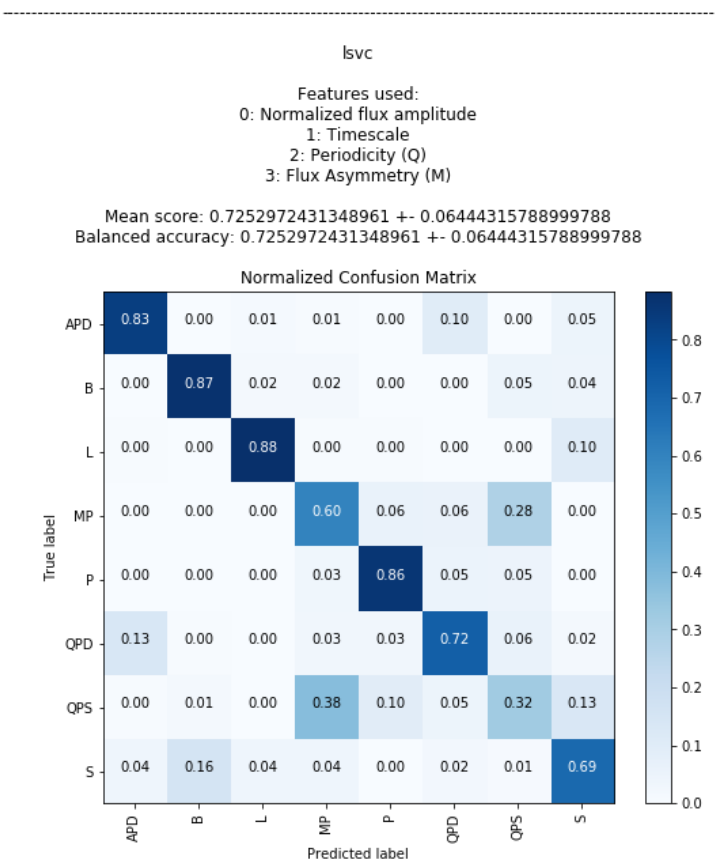

Figure 12. Mean confusion matrix of the linear svc algorithm using randomized search hyperparameter optimization using four of the default features. (Obtained from code cell 4 of the notebook Machine Learning Classification Testing Linear SVC Randomized Search (Version 5)).

The above three results for the svc (rbf kernel), svc (linear kernel) and linear svc classifiers had a similar behavior in classifying objects as the random forest algorithm. However, these three svc classifiers classified multiperiodic objects correctly at a higher rate than the random forest classifier did, but the svc classifiers classified quasi-periodic symmetric objects at a lower rate than the random forest classifier did.

Conclusions and Future Work

While less than ideal results for classification testing were found for the classification of young stars with the various machine learning classifiers tested in this project, the results found were still promising. The best classification result, total accuracy of 75 ± 5 % and balanced accuracy of 75 ± 6 %, obtained using the random forest classifier, still shows promise that with a more robust set of features, the classification of variable young stars using supervised machine learning methods could see application for unclassified astronomical data with a 90%+ balanced accuracy on test set data. Overall, supervised learning has great potential for automated variable young star classification.

Considering how accurately the random forest (using randomized hyperparameter optimization) algorithm performed for all variability types except for multiperiodic and quasi-periodic symmetric objects, one of the most significant classification performance improvements that could be made is the development of a feature (or multiple features) that can specifically identify the multiperiodic, amplitude-modulated nature of multiperiodic objects in contrast to the single-period nature of quasi-periodic symmetric objects. Additionally, new features developed could focus on differentiating between other commonly mistaken objects such as aperiodic dipper objects in contrast with quasi-periodic dipper objects, and stochastic objects in contrast to most of the other variability types. The inclusion of more light curve data from variable young stars may also improve classifier performance.

From testing classifier performance using grid search versus randomized search for hyperparameter optimization, randomized search tended to perform with greater accuracy and balanced accuracy than grid search. Since only the Random Forest, SVC, and LSVC classifiers have been implemented and tested with randomized hyperparameter optimization, more

classifiers, such as Decision Tree, Extremely Randomized Trees, and AdaBoost could be implemented in the project software library with randomized search functionality and performance tested. In particular, another type of classifier that could be explored in more depth is the neural network. Neural networks operate significantly differently from all the other classifiers previously discussed and may have the ability to perform better than other classifiers. Neural networks tend to run fastest using GPUs along with CPUs, so another machine learning package for neural networks would have to be used besides sklearn (which only uses CPUs), such as Keras or TensorFlow.

Aside from the previous examples of how supervised machine learning classification could be improved, another significantly different route to explore is unsupervised learning. The classification system that was used in this project for supervised machine learning may not be ideal for computer-automated classification, and that there may be more effective ways for classifying variable young star light curves. The equivalent of classification for unsupervised learning, clustering, is a method used by several unsupervised learning algorithms that allows the algorithm to draw its own conclusions about how to differentiate between different objects into any number of distinct groups. One possible downside to this approach is that the way in which a clustering algorithm differentiates objects may not have any relation to the visible differences seen among different light curves, but clustering algorithms could also bring to attention new and astronomically informative ways in which to classify different types of variable young stars.

Acknowledgments

I would like to thank Dr. Hillenbrand, Dr. Cody for guidance on the project.

References

Armstrong, D. J., Kirk, J., Lam, K. W. F., et al. 2016. K2 variable catalogue II: Machine learning classification of variable stars and eclipsing binaries in K2 fields 0-4. *Monthly Notices of the Royal Astronomical Society, 456*, 2260-2272. doi:10.1093/mnras/stv2836

Bloom, J. S. & Richards, J. W. (2012). *Data mining and machine learning in time-domain discovery and classification*. Boca Raton, FL: CRC Press.

Cody, A. M., Hillenbrand, L. A. 2018. The many-faceted light curves of young disk-bearing stars in Upper Sco and ρ Oph observed by K2 Campaign 2. eprint arXiv:1802.06409

Cody, A. M., Stauffer, J., Baglin, A., et al. 2014. CSI 2264: Simultaneous optical and infrared light curves of young disk-bearing stars in NGC 2264 with *Corot* and *Spitzer*– evidence for multiple origins of variability. *The Astronomical Journal, 147*, 47 pp. doi:10.1088/0004-6256/147/4/82

Hedges, C., Hodgkin, S., Kennedy, G. 2018. Discovery of new dipper stars with K2: A window into the inner disk region of T Tauri stars. *Monthly Notices of the Royal Astronomical Society, Advance Access*. doi:10.1093/mnras/sty328

Herbst, W. 2012. The variability of young stellar objects. *Journal of the American Association of Variable Star Observers, 40*, 448-455. Retrieved from https://www.aavso.org/ejaavso401448

Hinners, T. A., et al. 2018. Machine Learning Techniques for Stellar Light Curve Classification. *The Astronomical Journal, 156*, 13 pp. doi:10.3847/1538-3881/aac16d

Mackenzie, C., Pichara, K., Protopapas, P. 2016. Clustering based feature learning on variable stars. *The Astrophysical Journal, 820,* 15 pp. doi:10.3847/0004-637X/820/2/138

Richards, J. W., Starr, D. L., Butler, N. R., et al. 2011. On machine-learned classification of variable stars with space and noisy time-series data. *The Astrophysical Journal, 733,* 20 pp. doi:10.1088/0004-637X/733/1/10

Scikit-learn. (2018). Scikit-learn: Machine Learning in Python. Retrieved from http://scikit-learn.org/stable/

Strobel, N. (2010, June 8). The Basic Scheme. Retrieved from http://www.astronomynotes.com/evolutn/s3.htm

Strobel, N. (2007, June 2). Stage 3: T-Tauri. Retrieved from http://www.astronomynotes.com/evolutn/s4.htm

Valenzuela, L., Pichara, K. 2017. Unsupervised classification of variable stars. *Monthly Notices of the Royal Astronomical Society, 474,* 3259-3272. doi:10.1093/mnras/stx2913