Initial Improvements:

Following the analysis performed in the project proposal, the following improvements were made to the feature space in an attempt to improve classification performance:

- All timeseries were properly concatenated to allow for improved fitting of the transit curve.
- Instances which contained an outlier in any numeric feature were removed. Since many of these features are like to not be normally distributed and are likely to contain a large number of points, Median Absolute Deviation (MAD) outlier detection was used where an outlier is defined as any point *x* in the dataset *X* where:

$$|x - \text{median}(X)| > 3 * \text{MAD}, \quad \text{MAD} = \frac{-1}{\sqrt{2}\,\text{erfc}\left(\frac{3}{2}\right)}\text{median}(|X - \text{median}(X)|)$$

- All numeric features were normalized.

With the LibSVM model created for the project proposal, the baseline model performance correctly classified 75.16% of instances *but*, given the structure of the training data, this is not impressive and that was confirmed proposal baseline's abysmal kappa of 0.0574.

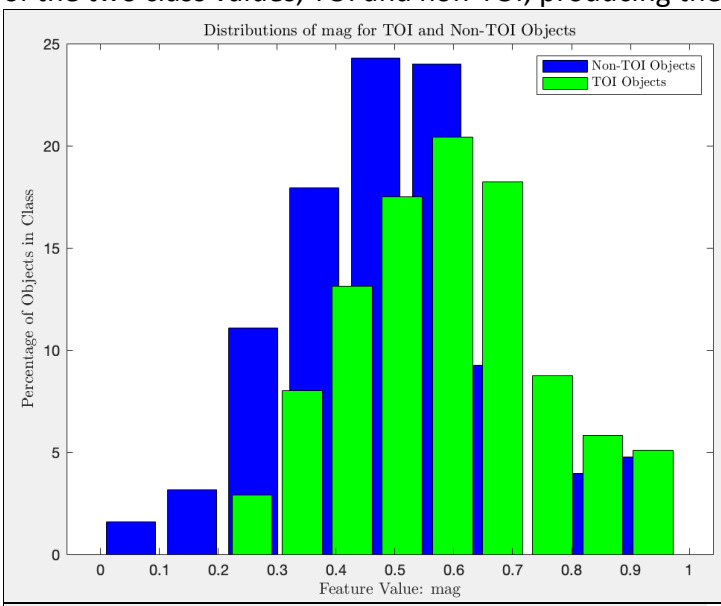
New Baseline:

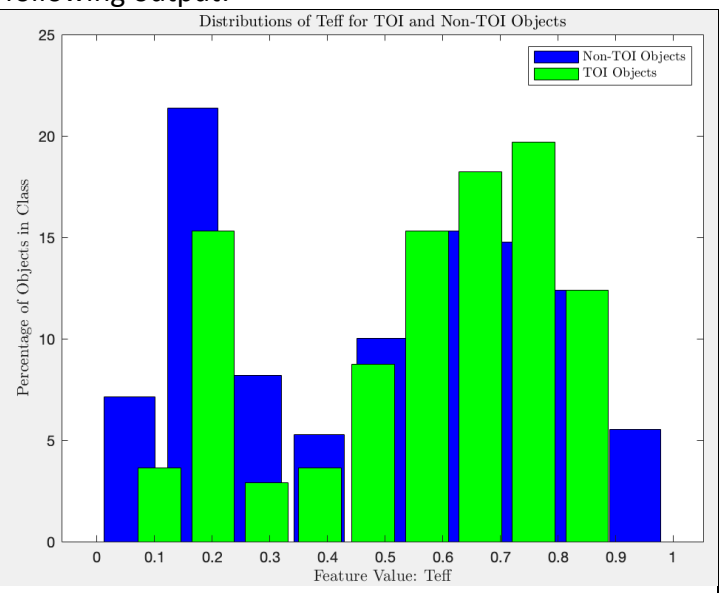
Having made these improvements following the proposal, several baseline models were tried out using the default settings, the best performing of which will be developed further.

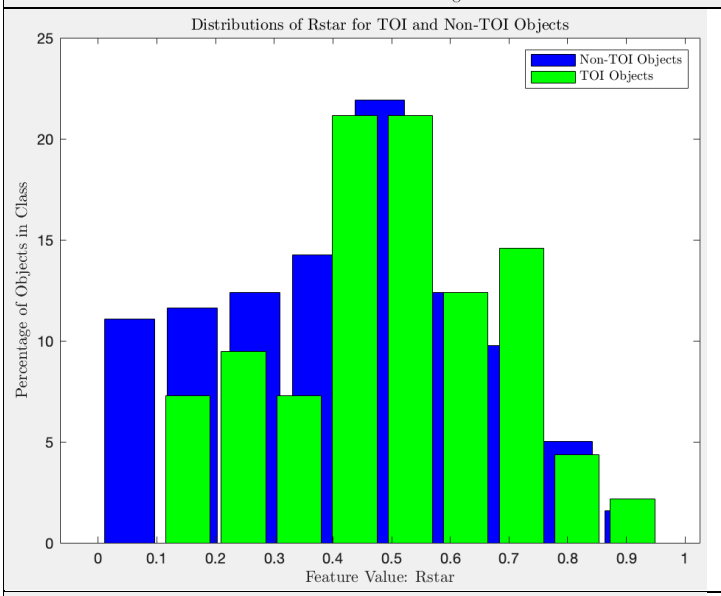
	Percent Correct	Kappa
LibSVM with PolyKernel	79.07%	0.00
Logistic Regression	83.52%	0.30
Naïve Bayes	82.17%	0.33
J48 Decision Tree	83.52%	0.37
Multilayer Perceptron	**87.98%**	**0.59**

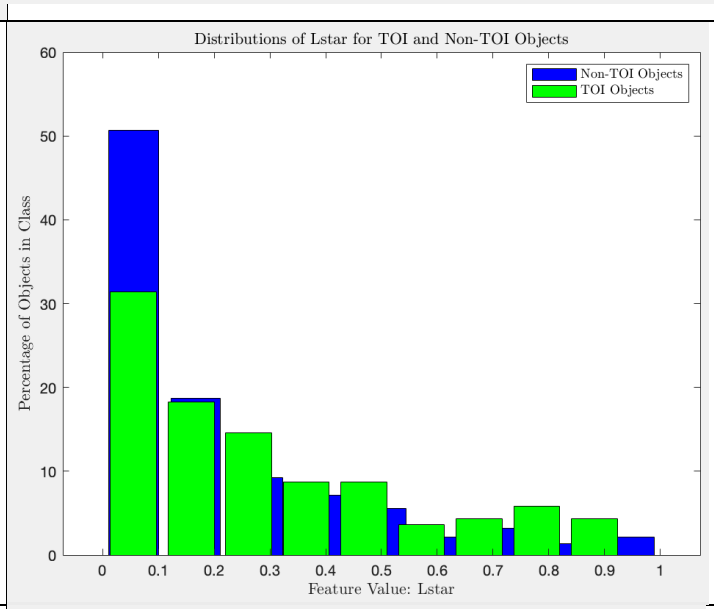
Initial Observations:

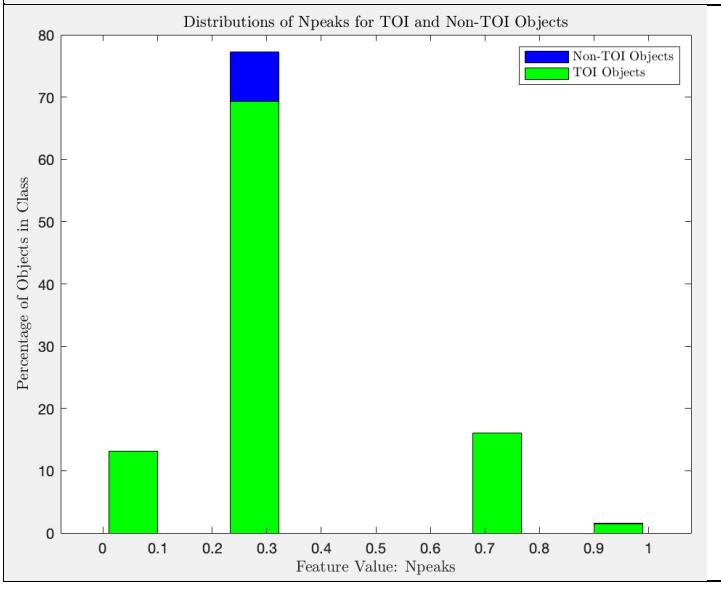
A MATLAB script was written to help with initial observations by plotting the distributions within each feature of the two class values, TOI and non-TOI, producing the following output:

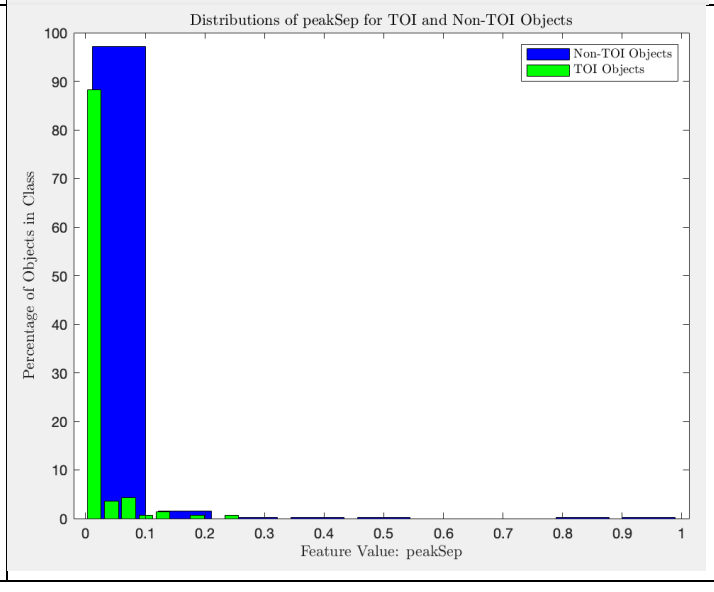








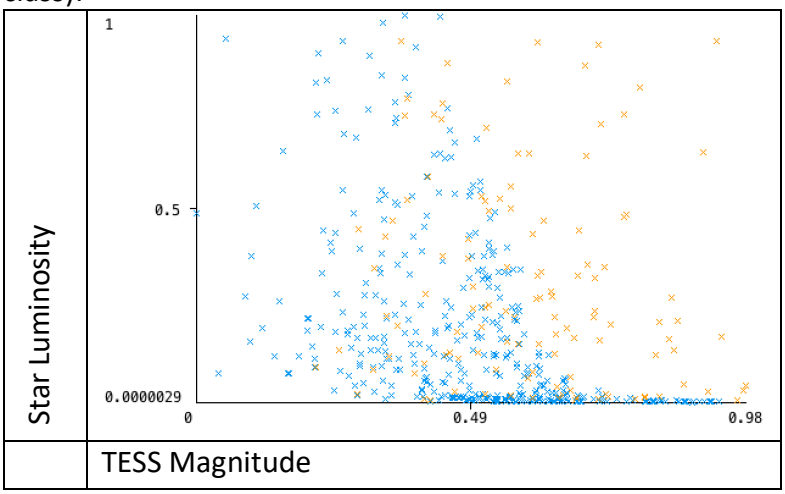


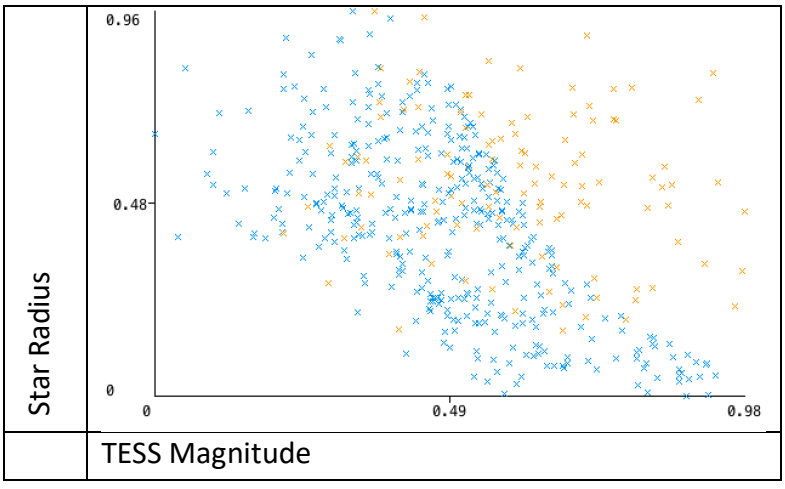


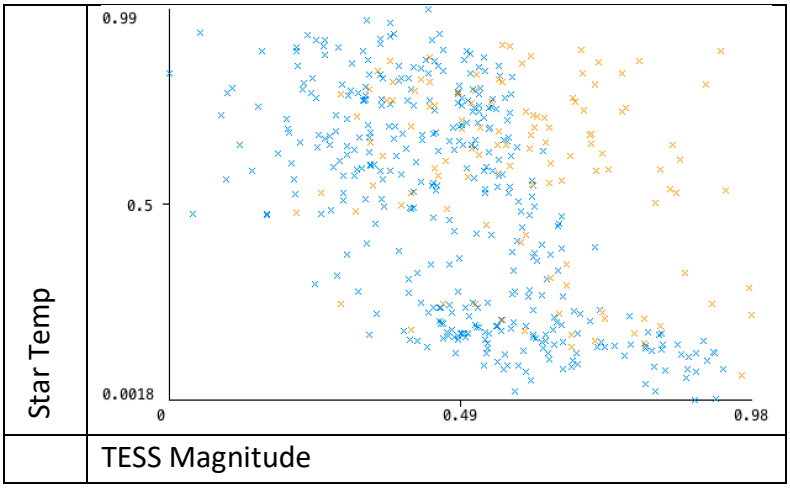
```
The avg. mag for Non-TOI objects was 0.485 but 0.593 for TOI objects (percent difference: 20.0%)
The avg. Teff for Non-TOI objects was 0.482 but 0.565 for TOI objects (percent difference: 15.8%)
The avg. Rstar for Non-TOI objects was 0.410 but 0.503 for TOI objects (percent difference: 20.2%)
The avg. Lstar for Non-TOI objects was 0.190 but 0.281 for TOI objects (percent difference: 38.3%)
The avg. Npeaks for Non-TOI objects was 0.344 but 0.353 for TOI objects (percent difference: 2.6%)
The avg. peakSep for Non-TOI objects was 0.015 but 0.012 for TOI objects (percent difference: 25.9%)
```

The most clear observation that comes from these data is that there is no feature which contains a partition between the two classes within itself. If possible, it would be great to find or add features which have such a distinction. That said, from this early analysis, the somewhat distinguished difference in the means for the Lstar feature indicates that it may serve as a useful feature for later partitioning (perhaps through a clustering algorithm).

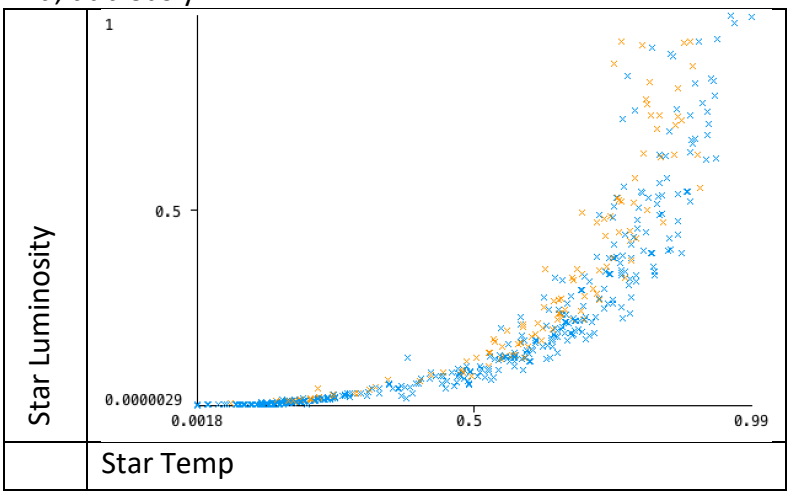
To gain further insight into whether there exist obvious partitions between the two classes in the feature space, the Weka Visualize workspace was used. Distinct partitions between the TOI and Non-TOI classes were observed in the following 2D slices of the feature space (where blue is the Non-TOI class and orange is the TOI class):





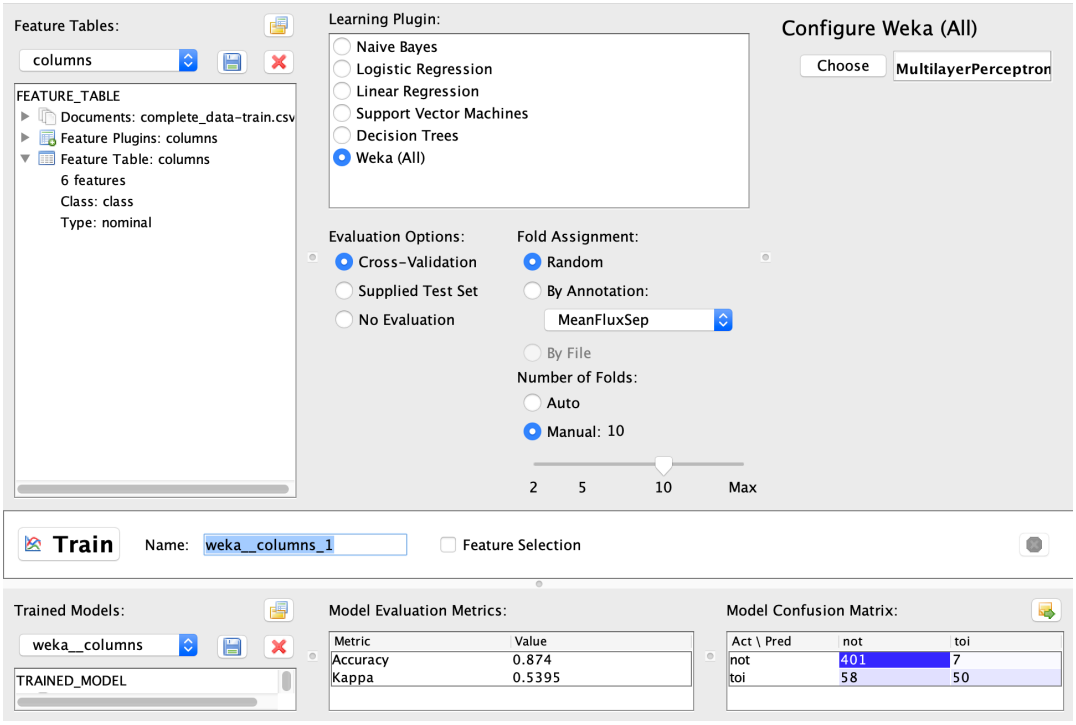


And, dubiously:

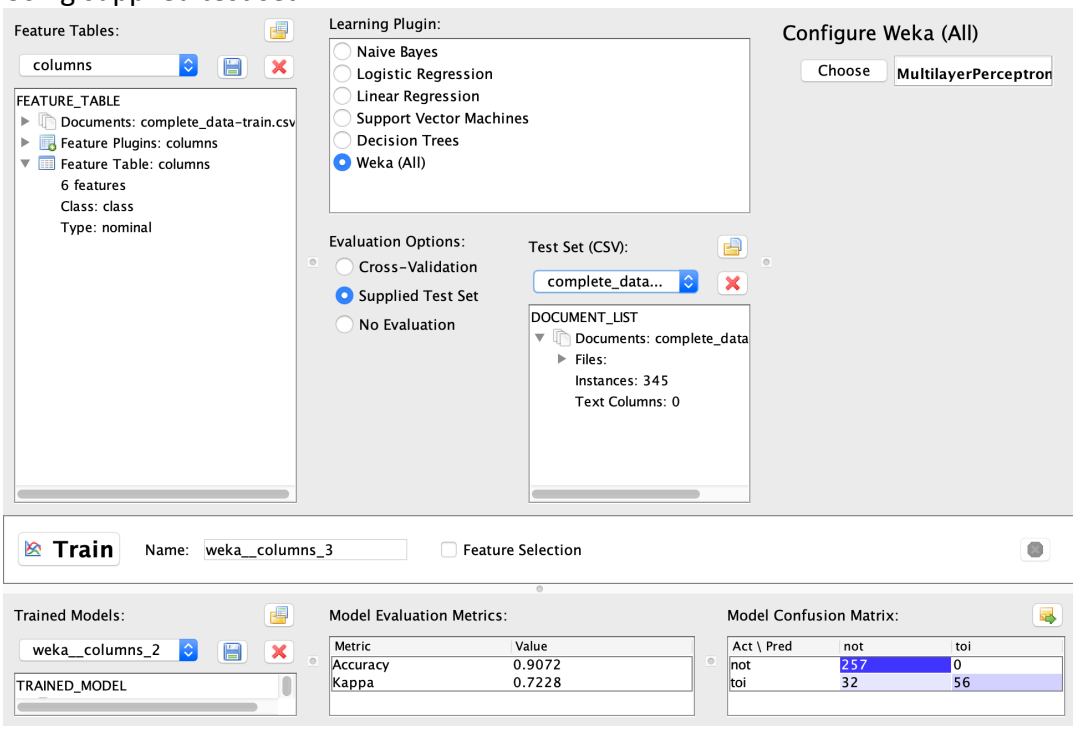


Error Analysis:

Model Setup and Baseline Training Outcome:

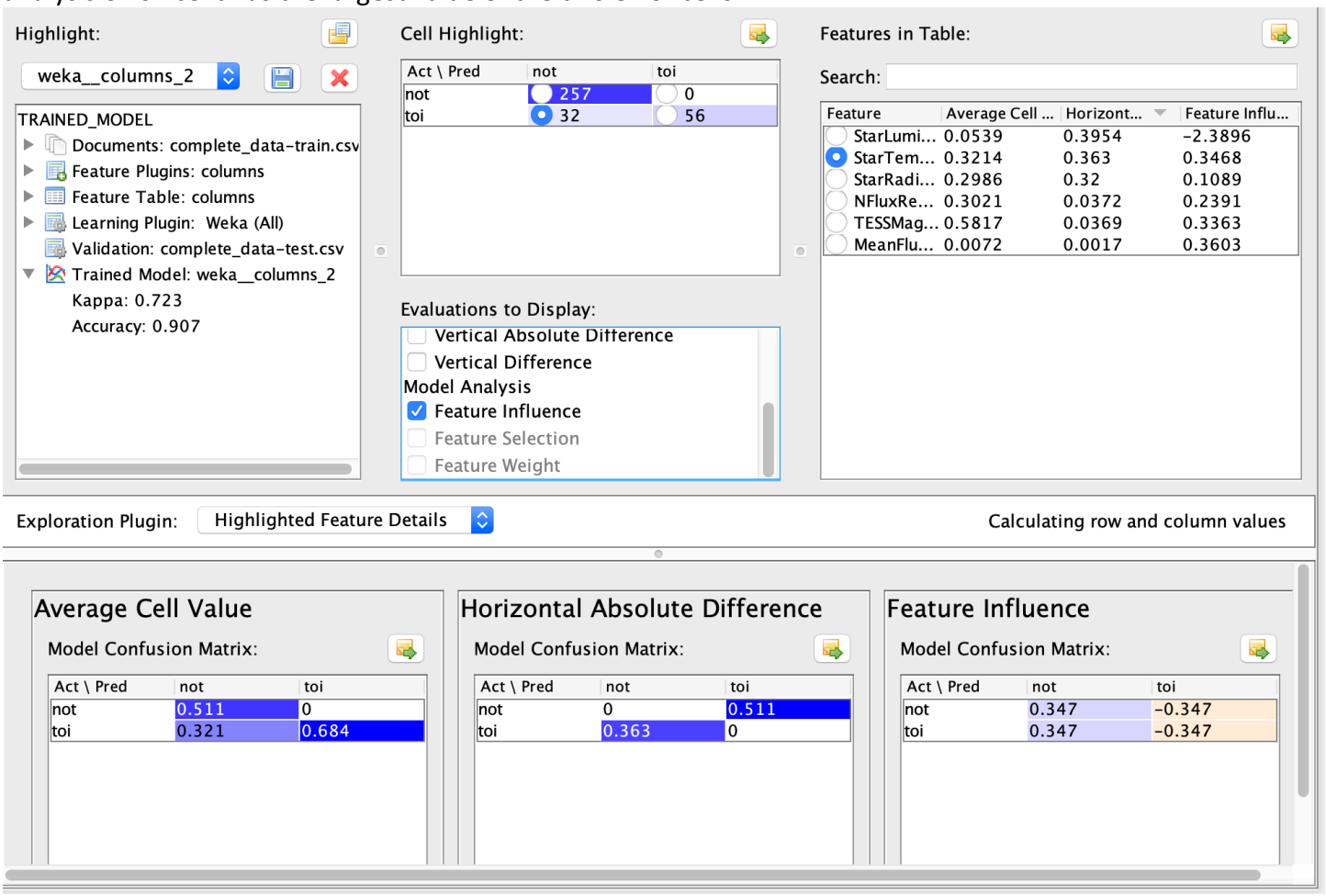


Using supplied test set:



Horizontal Comparisons:

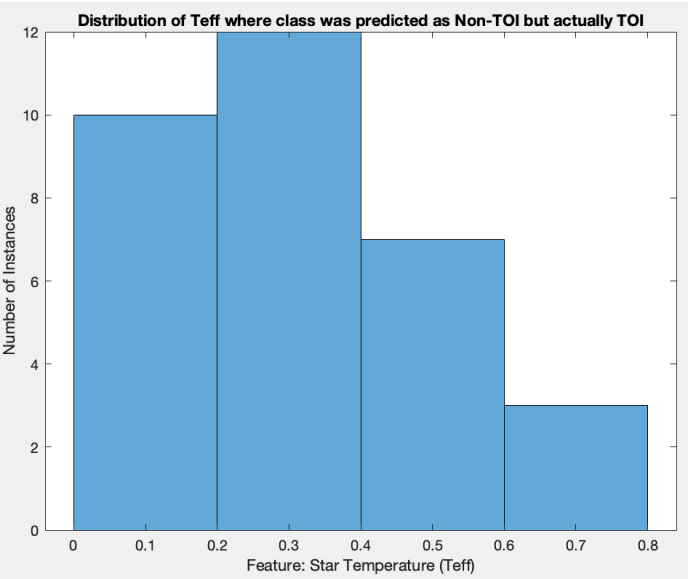
The error cell in which class was predicted to be Non-TOI but was actually TOI was chosen to perform error analysis on since it has the largest value of the two error cells.

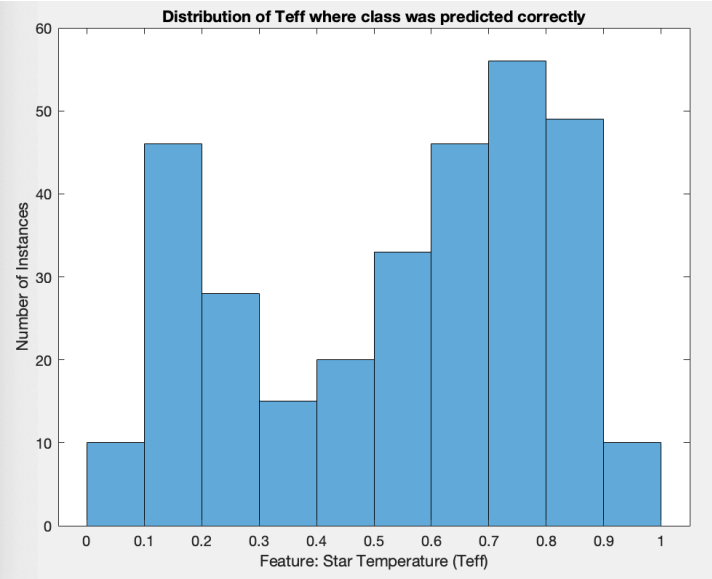


The first problematic feature chosen was the "StarTemperature" column since it had the highest horizontal absolute difference and also had a large feature influence magnitude (feature weight wasn't an option), and non-zero average cell value.

The selected data shows that the model decreases the likelihood of TOI as StarTemp increases and that lots of the instances which are actually classified as TOI, get classified as Not-TOI.

To gain further insight on this, the dataset including the predicted labels were exported from the Predict Labels pane. Feeding this data into a basic MATLAB helper script produces the attached histograms which show that for all instances where class was predicted to be Not-TOI but actually was TOI, "Teff" was always less than or equal to 0.7 and most often 0.3. Notably, this appears to be almost an inverse of the distribution for where Teff was predicted correctly would be relatively flat if not for a dip centered at Teff=0.3, extending to about Teff=0.7.



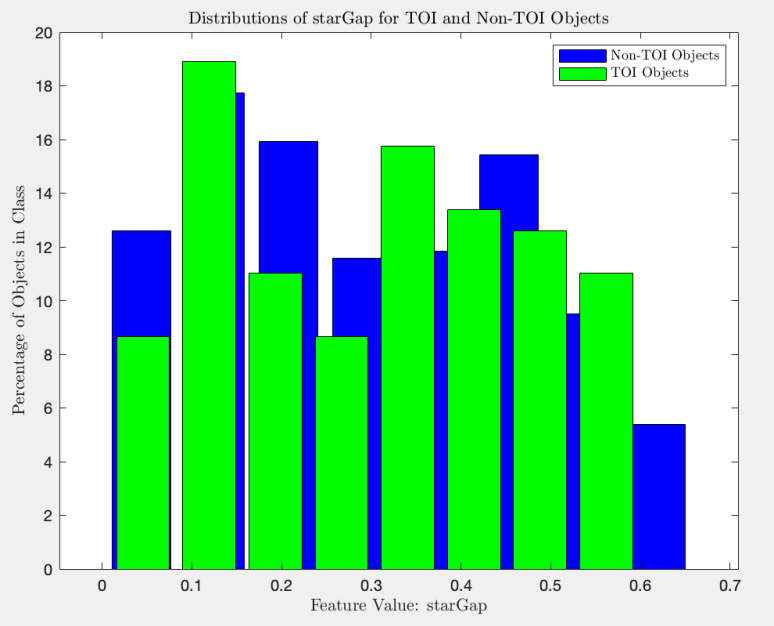


Notably, looking at the raw data, this Teff=0.3 corresponds to the gap between main sequence and red dwarf stars around Teff=4000K. Any stars within this range are likely to be anomalous. A potential addition to the feature-space which could resolve this would be either a Boolean **Teff>0.2 & Teff<0.5** or a distance metric to the center of the gap of the form **abs(Teff-0.3)**. Since the borders between these types of stars are not particularly hard, the second type makes the most sense.

Model Improvement:

The **abs(Teff-0.3)** feature, called StarGap, proposed in the horizontal comparison error analysis was added to the feature space and the model was retrained.

In Weka Visualize, no 2D subspaces involving StarGap show clear partitioning between the classes and, as shown below, there are no salient differences in the distributions of StarGap for TOI and Non-TOI Objects.



In comparison to the baseline, this model offers a significant improvement in both correctness and kappa as shown below. Notably, there is marked improvement in the error cell studied.

Baseline Model: weka__columns

TRAINED_MODEL
- Documents: complete_data-train.csv
- Feature Plugins: columns
- Feature Table: columns
- Learning Plugin: Weka (All)
- Validation: complete_data-test.csv

Competing Model: weka__columns_1

TRAINED_MODEL
- Documents: complete_data-train-stargap.csv
- Feature Plugins: columns
- Feature Table: columns_1
- Learning Plugin: Weka (All)
- Validation: complete_data-test-stargap.csv

Comparison Plugin: Basic Model Comparison

Baseline Model Metrics:

Metric	Value
Accuracy	0.8928
Kappa	0.6428

Competing Model Metrics:

Metric	Value
Accuracy	0.9159
Kappa	0.7184

Baseline Confusion Matrix:

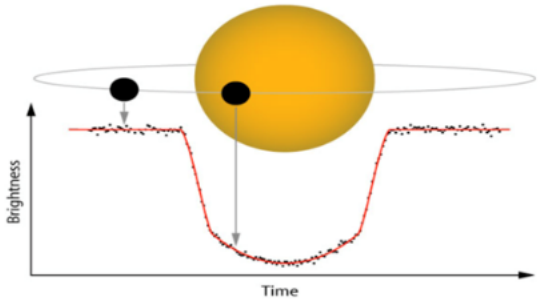
Act \ Pred	not	toi
not	264	4
toi	33	44

Competing Confusion Matrix:

Act \ Pred	not	toi
not	268	4
toi	25	48

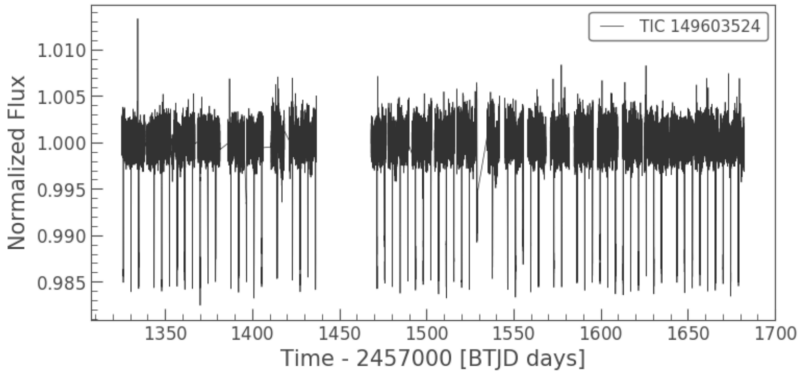
Feature Space Revisions

One of the most important potential improvements identified during the initial observations phase is the need for more distinguishing features. The most underutilized and, simultaneously, most important aspect of the dataset is the light curves. The problem is that useful information about them is aggregate metadata about all observed transit events for a given object, namely depth, rise time, fall time, width, and period. While an ideal transiting event in a light curve would look like the following:

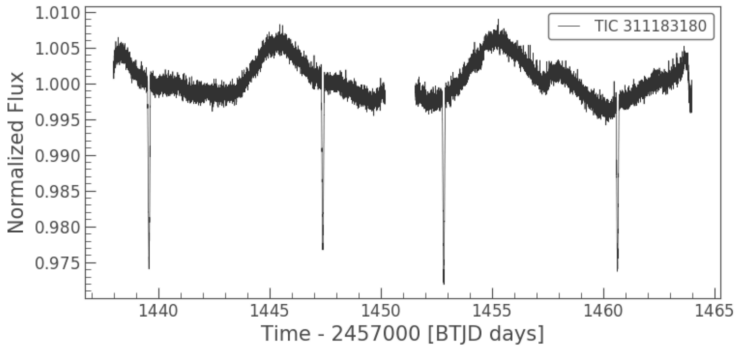


[1]

Thus, a full ideal light curve would look a series of these dips occurring at a regular interval where the photometry at all other points in time is flat. However, the actual data from TESS looks like:



wherein some the sudden downward spikes are transit events but some are noise, or, even, worse:

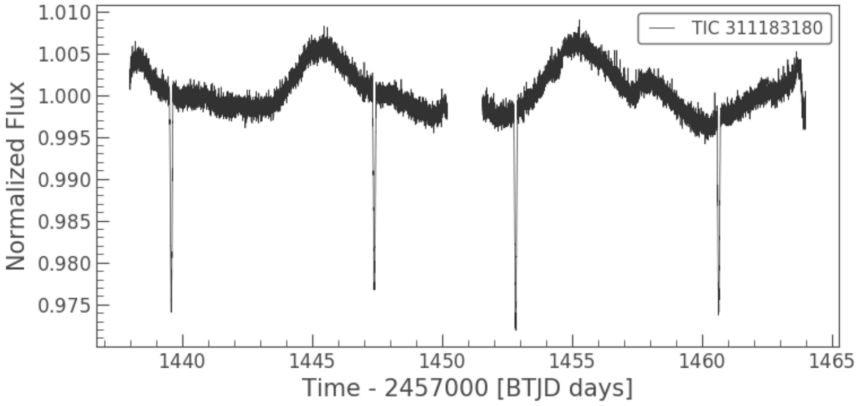


wherein the transit events are not the large sinusoidal oscillation but rather the 4 sudden downward spikes. Clearly, there needs to be an automated process which can denoise the data, remove long term trends, determine the frequency of oscillations, and aggregate the light curve sections of each transit event to create a single transit curve from which the important metadata can be extracted. To accomplish this in fairly robust way, the following process was developed using python and the Lightkurve Collaboration's lightkurve package. To illustrate this process, the light curve of TIC 311183180, which is a TOI, will be used for examining flattening since it has a long term oscillatory trend and TIC 149603524 will be used for examining curve aggregation since it has a known exoplanet whose period the recovered period can be compared against.

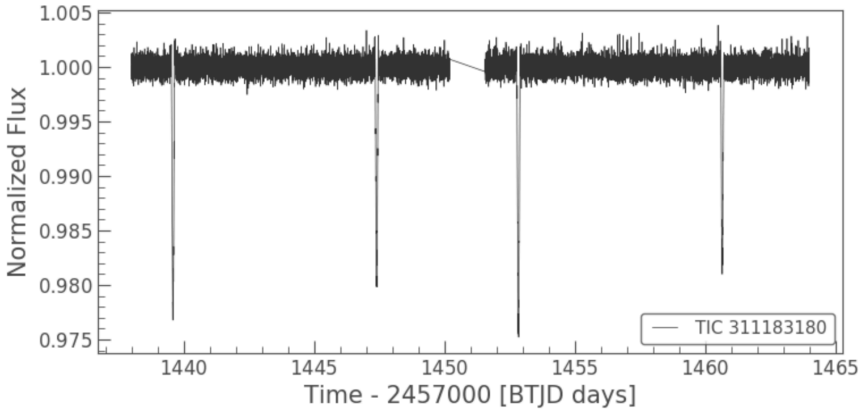
[1] https://www.apus.edu/academic-community/space-studies/exoplanet-transit-photometry

Removing long term (low-frequency) trends

Trends in the data with the lowest frequencies were removed by applying a Savitzky-Golay filter using *scipy* with a second order polynomial with 101 coefficients, transforming this:

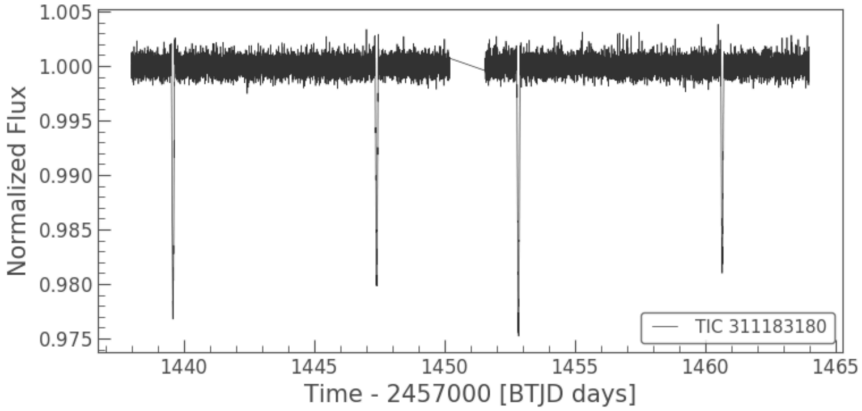


into this:

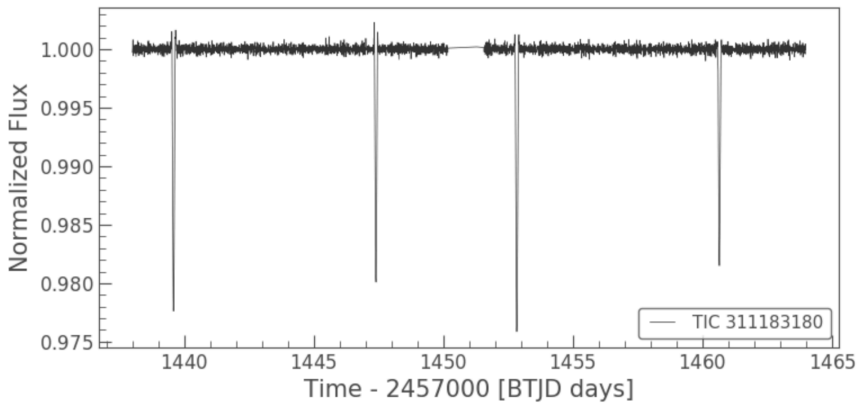


Remove noise

Noise was removed by binning the data (reducing its time frequency by applying a moving average filter over it). The size of the averaging window used in this stage was varied a convenient width of 5 data points was chosen. The result of this process reduces the high frequency noise while still preserving the sustained dips caused by transiting events, transforming this:

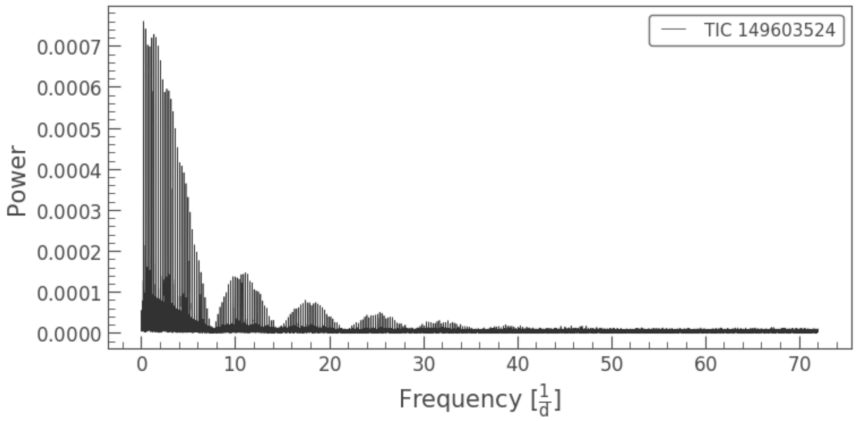


into this:

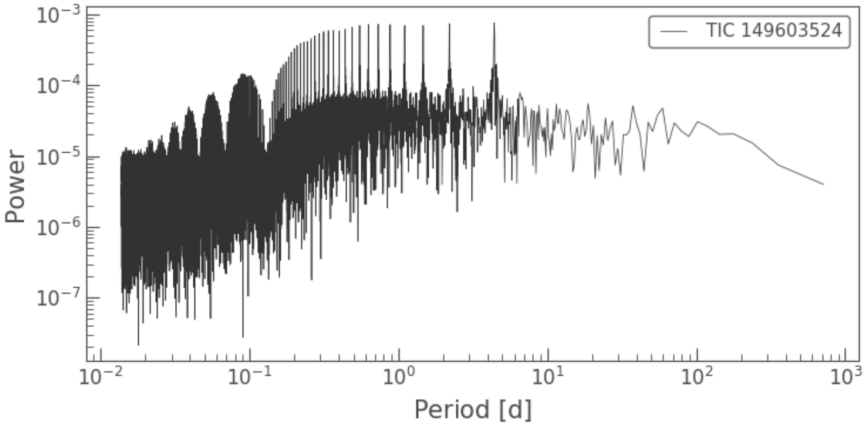


Determine Period

Since by definition transit events consist of many points with a similar sustained drop in flux occurring at a regular interval, the period of these transiting events was determined by computing a Fourier-like power spectrum using *lightkurve*'s Periodogram object with double oversampling to smooth out the result. The following plot produced using this method shows the spectral density of the signal vs the frequency at which datapoints recur.



By transforming this plot from frequency parameterization to period parameterization, the maximum will be the period at which the largest change in signal value occurs (id est, a transit event).

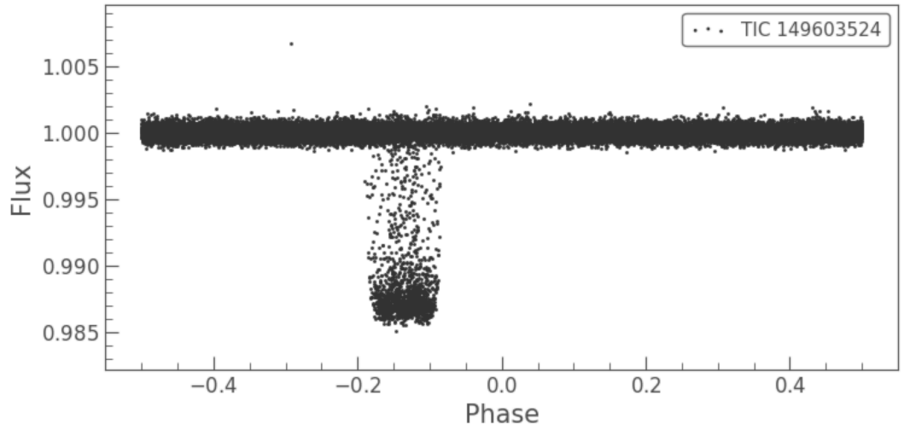


For the above signal, the identified period for TIC 149603524 is 4.408 days. This is noteworthy since this star has a known exoplanet, WASP-62b, whose period has been officially determined to be 4.41195 days[2].

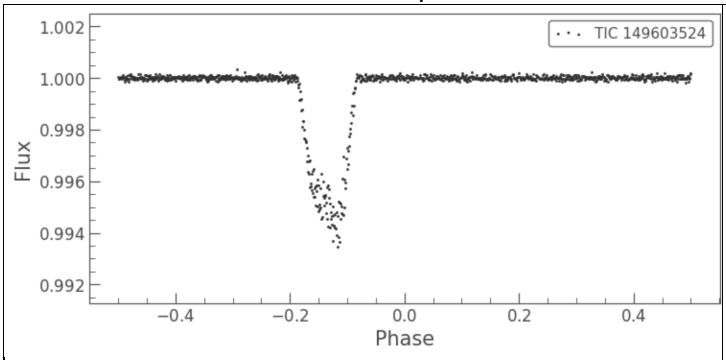
[2] https://exofop.ipac.caltech.edu/tess/target.php?id=149603524

Aggregate Transit Events

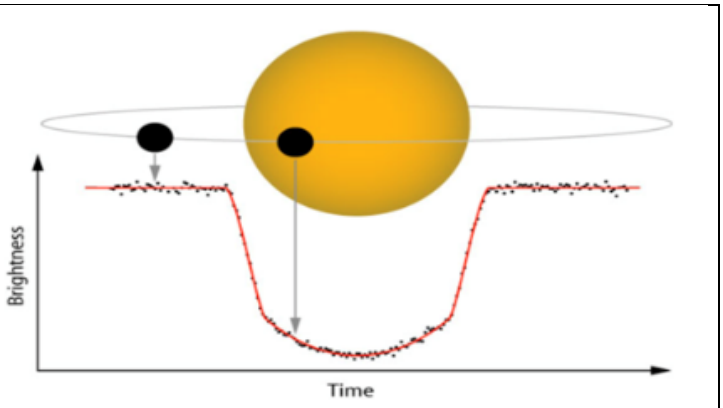
With *lightkurve*, the process of combining data which occurs with constant frequency into a single aggregate plot is called “folding” and done with a simple call to the *fold* function. So, folding the light curve of TIC 149603524 with the determined period gives the following plot of its exoplanet’s transit.



Since there is no distinct fall or rise which will be needed to extract the significant meta data, this curve must be smoothed, in this case by binning the plot. After trying various values on the curves of many stars, a bin width of 35 entries was chosen, producing the following completed aggregate light curve which looks strikingly similar to that of a standard exoplanet transit event.



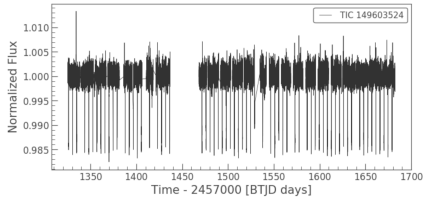
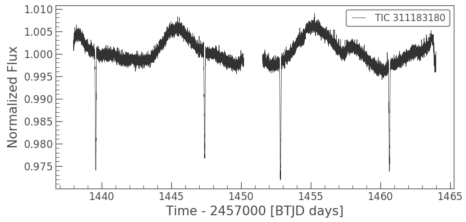
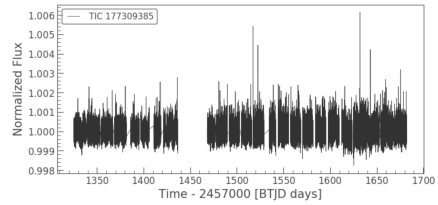
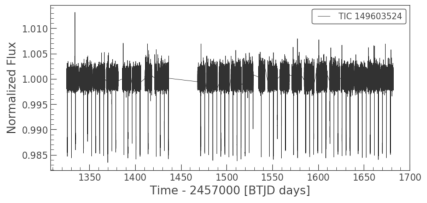
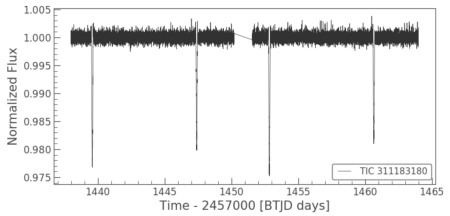
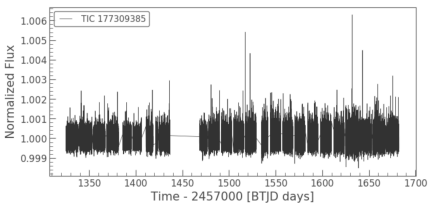
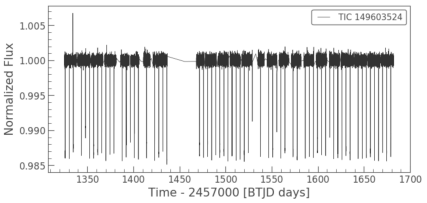
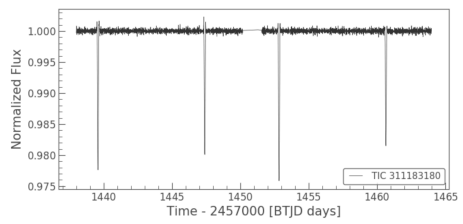
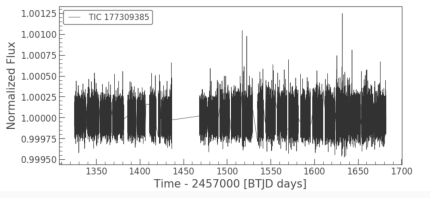
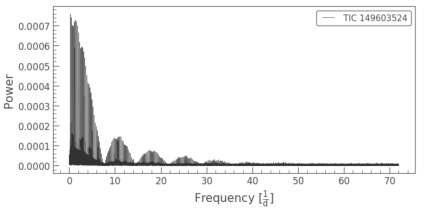
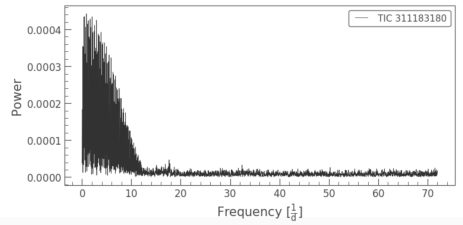
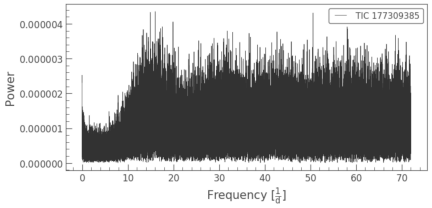
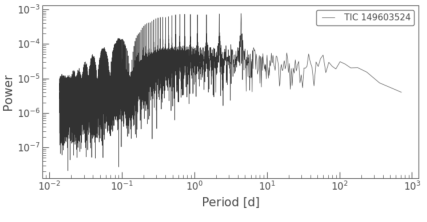
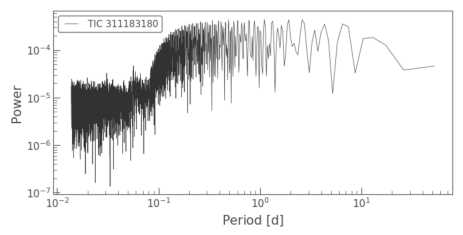
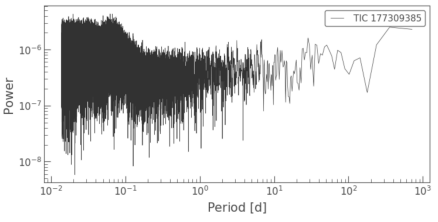
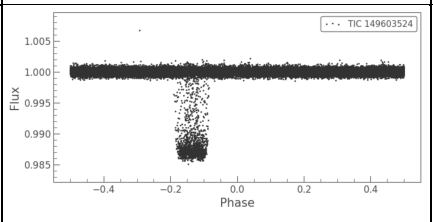
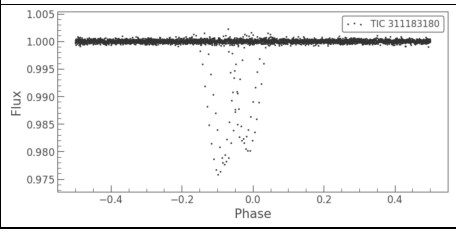
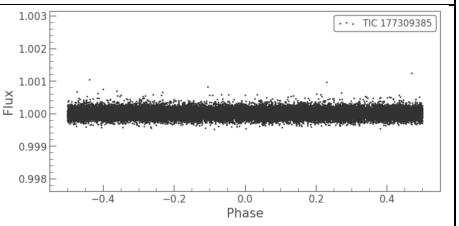
Recovered Transit Curve for TIC 149603524.

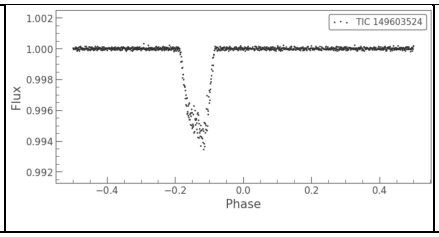
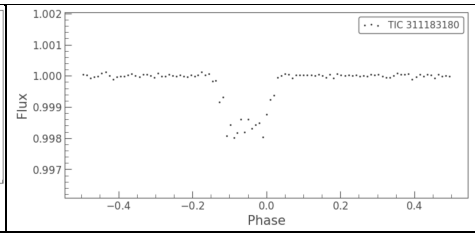
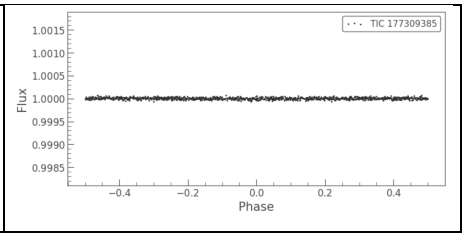


Prototypical Exoplanetary Transit Curve.

Extended Application

Each step of this process was repeated for a model TOI object (TIC 149603524: single star, single exoplanet), a "tricky" TOI object (TIC 311183180: multiple star, multiple planet), and a model Non-TOI object (TIC 311183180: just a single star with no noticeable anomalies). Plots for each of the above stages are shown below.

Step		Distinctive TOI Object Example 1 (TIC 149603524)	Tricky TOI Object Example 2 (TIC 311183180)	Distinctive Non-TOI Object Example (TIC 311183180)
0.	Raw Concatenated			
1.	Flattened Data			
2.	Binned Data			
3.	Frequency Power			
	Period Power			
4.	Folded at Peak Period			

5.	**Binned Folded Data**			

- Note: not all Non-TOI objects are as distinctive as TIC 177309385, many also have transit events.

Currently, I have a script combining through ~25GB of light curve data for the 3545 objects in my unfiltered dataset and applying the above process to extract aggregate light curves at the highest power period. Once this completes in about 2.5 days, I will be able begin extracting the critical metadata from each curve (transit depth, rise time, fall time, width, period, predicted planet mass, predicted planet radius).

Identified Flaws for Next Revision:

- Need much larger dataset (finishing processing light curves).
- Need more features which have inherent separation between classes (ie. transit curve metadata mentioned above).
- Perform phased parameter tuning on multilayer perceptron (has lots of parameters).
- More formal addition of a feature corresponding to sub-mainsequence stars, pre-red dwarf stars with Teff near 4000K.
  Compute **abs(Teff[K]-4000K)** then normalize.