Applying Machine Learning to Exoplanet Detection

Charles Zhao
Princeton University
hczhao@princeton.edu

Bilal Mukadam
Princeton University
bmukadam@princeton.edu

Abstract

Machine learning has shown great promise in many scientific disciplines, including astrophysics. The goal of our work is to explore several machine learning methods for automatically detecting stars with orbiting exoplanets. To do this, we use data from the Kepler space telescope, which includes flux (light intensity) measurements from several thousand stars over time, as well as a binary variable indicating whether each of them has one or more exoplanets. Our first analysis uses a standard exoplanet detection technique called box least squares. In our second analysis, we construct several features from each time series and run these features through several "classic" (i.e. not deep learning) machine learning classifiers. In our last analysis, we explore two deep learning methods. Overall, both deep learning methods appeared to perform the best out of our analyses.

1 Introduction

Accurate automatic detection of exoplanets could be extremely time-saving. Even if a classifier's accuracy is not perfect, providing likely candidates could also be immensely valuable for closer examination by astrophysicists.

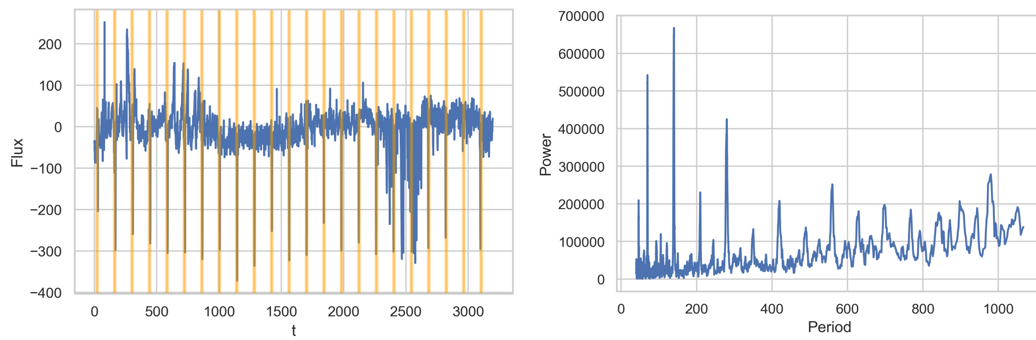
Since an exoplanet should periodically obscure part of the star it orbits, theoretically, one should be able to detect exoplanets using flux time series data. We use data collected by the Kepler space telescope and cleaned by NASA [9]. Over 99% of this data comes from Campaign 3 of the mission, which lasted from November 14, 2014 to February 6, 2015 [12]. The training set consists of 3,197 flux measurements for 5,087 stars, with 5,050 without-exoplanet stars and 37 with-exoplanet stars. The test set consists of the same number of measurements for 570 stars, with 565 without-exoplanet stars and 5 with-exoplanet stars.

This paper starts with a survey of related work that guided our model decisions, followed by some preliminary exploration of the data. We then describe our models and their results from our three analyses. Finally, we discuss our results as well as possible future steps.

1.1 Related Work

Kovács, Zucker, and Mazeh [13] presented an algorithm called Box Least Squares (BLS) for finding periodic transits. This algorithm is now quite standard in exoplanet detection and analysis. BLS models the transit light curve as a periodic box (see Figure 11 in the appendix), where the box represents the decrease in flux due to the transiting exoplanet. The authors found that this method works particularly well when the signal-to-noise ratio (SNR) is small and thus the periodic signal can only be detected after measuring the unknown transit many times. They concluded that when the effective SNR exceeds 6, this indicates a significant detection.

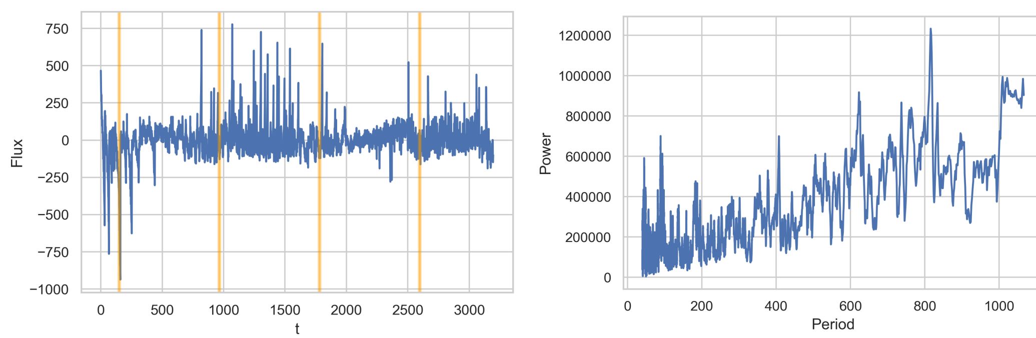
Graves and Schmidhuber [10] presented bidirectional long short-term memory (BLSTM) networks, which they evaluated on framewise phoneme classification. LSTMs are a variant of recurrent neural networks (RNNs) that give them a kind of long-term memory. BLSTMs are a variant of standard



(a) Light curve of a with-exoplanet star, with the transits found by BLS highlighted.

(b) Periodogram produced by BLS of a with-exoplanet star.

Figure 1: Light curve and periodogram of a with-exoplanet star.



(a) Light curve of a without-exoplanet star, with the transits found by BLS highlighted.

(b) Periodogram produced by BLS of a without-exoplanet star.

Figure 2: Light curve and periodogram of a without-exoplanet star.

LSTMs that present each training sequence forwards and backwards to two separate LSTMs, both of which are connected to the same output layer. Therefore, for every point in a given sequence, the network has information about the sequence both before and after the given point. The authors found that bidirectional networks outperform unidirectional ones, and that LSTM is both much faster and more accurate than standard recurrent neural networks.

2 Data Exploration

We first try to get an intuitive understanding of what the light curve of a with-exoplanet star looks like as opposed to that of a without-exoplanet star. Figure 1a is the light curve of a with-exoplanet star, and Figure 2a is the light curve of a without-exoplanet star (we explain the highlighted ranges in Section 4.1). The periodic transit of an exoplanet in Figure 1a seems quite clear visually, as opposed to Figure 2a where there does not appear to be a periodic transit.[1]

3 Methods

3.1 Feature Engineering and Preprocessing

For the classic machine learning methods and the deep learning methods, we first upsample the training data by repeating all the with-exoplanet data points 136 times, resulting in the training set

[1] We have handpicked these stars for this section, but we should note that the light curves of with-exoplanet stars are not so obvious in many cases.

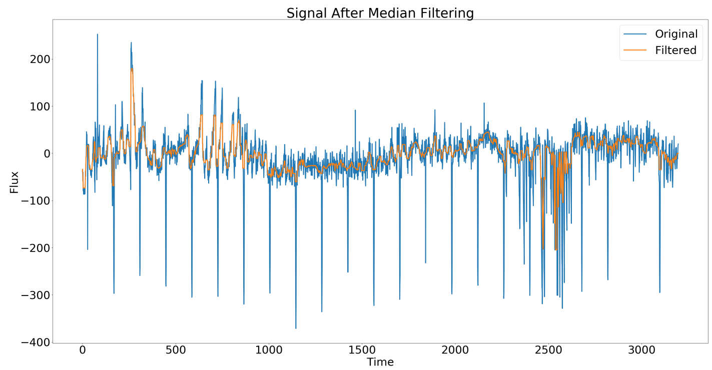


Figure 3: Flux signal before and after median filtering

comprising 5050 without-exoplanet and 5032 with-exoplanet stars. We then perform the following transformations using Scipy [3], PyWavelets [4], and Astropy [5] on the time series data to construct features for each star:

1. Discrete Fourier Transform to determine the coefficients of the 4 strongest frequencies
2. Discrete Wavelet Transform to determine the coefficients of the 4 strongest wavelets
3. Box Least Squares to determine depths, uncertainties, powers, periods, and durations of the maximum power period

Beyond simple duplication for upsampling, we also explored rotating the signals in time by some random integer to augment our data. However, as we trained our machine learning models, we found this yielded no improvement over simple duplication, so we stuck with simple duplication for upsampling.

In addition, for some analyses we explored smoothing of the flux signals as a preprocessing step. This was done by using a 12-hour sliding median window, i.e. the data in each window was reduced to its median. The 12-hour duration of the window and the median metric (as opposed to mean) were recommended by Professor Joshua Winn (Astrophysics Department) during our discussions with him regarding our project. In Figure 3 we show a flux-signal before and after median-filtering.

3.2 Performance Metrics

We use four different performance metrics: accuracy, area under the receiver operating characteristic curve (AUC), F1 score, and recall. The latter three metrics are particularly important, since accuracy alone does not give a full picture of performance. In particular, it will be extremely skewed simply because our data set is unbalanced (i.e., it consists of significantly more non-exoplanet stars than exoplanet stars). For example, simply classifying all stars as without-exoplanet would lead to $> 99\%$ accuracy in the test set.

AUC (which is between 0 and 1) can be interpreted as the probability that a random positive example (in our case, a with-exoplanet star) is classified more positively than a random negative example [8].

Recall is equal to the number of true positives divided by the number of actual positives. We place particular emphasis on this metric rather than, say, precision, since successfully detecting exoplanets is our main priority, and producing some false positives is acceptable. The F1 score is the harmonic average of precision and recall, where

$$\text{Precision} = \frac{\text{True Positive}}{\text{Total Predicted Positive}}, \qquad \text{Recall} = \frac{\text{True Positive}}{\text{Total Actual Positive}}.$$

In other words, precision gives the proportion of positive classifications that were correct, while recall gives the proportion of actual positive data points that were correctly classified [7]. For all four performance metrics, higher is better.

Although technically not a metric, we also use the ROC curve itself when evaluating our models. The ROC curve plots the true positive rate vs. the false positive rate at varying thresholds, and can thus be useful for approximating an optimal threshold at which to predict a data point as with-exoplanet or without-exoplanet. In general, the larger the area under this curve (i.e. the AUC described above), the better.

3.3 Models

In our baseline analysis, we perform Box Least Squares (BLS) using Astropy [6]. BLS models a transit as a periodic upside down hat with four parameters: period, duration, depth, and a reference time (see Figure 11 in the appendix). We use Astropy's `autopower` function to automatically determine the best periods at which to evaluate the power. We run this function with the following parameters:

- `duration=20`: This sets the granularity of the period search to 20 hours.
- `minimum_n_transits=4`: This restricts the period so that there must be at least 4 transits of the exoplanet within the light curve.
- `objective="snr"`: This sets the objective to be the signal-to-noise ratio (as opposed to the likelihood).

Following the results of Kovács et al., we classify each planet based on its SNR.

In our second analysis, using the upsampled training data, we construct the features described in Section 3.1 instead of working with the raw time series. We use the following classifiers from scikit-learn, with the default parameters unless otherwise specified:

1. Support Vector: Support vector classifier using a Radial Basis Function Kernel
2. Linear Support Vector: Support vector based linear separator
3. Random Forest: Bagging on tree sampling instances/features
4. Decision Tree: Tree-based classifier (N = 10)
5. K-Nearest Neighbors: Similarity analysis with k closest instances
6. Logistic Regression: Maps input to [0,1] range using logistic function, outputs classification probability

In our final analysis, also using the upsampled training data, we use Tensorflow [1] with Keras to create a long short-term memory recurrent neural network (LSTM RNN) as well as a convolutional neural network (CNN).

RNNs are generally effective for sequential data such as time series data. We used a very small sequential Keras model with the following layers (with default parameters unless otherwise specified):

1. Bidirectional LSTM layer with 4 neurons; outputs the output of the last neuron.[2]
2. Dense layer with 4 neurons.
3. Dropout layer; drops out 50% of inputs.
4. Dense layer with 1 neuron with sigmoid activation; outputs a number between 0 and 1.

We train this model using the Adam optimizer and with binary cross entropy as our loss function.

Figure 4 shows the architecture we ultimately used for our CNN (note that initially we performed 1D convolution - ie. didn't reshaped the data into images - so as to not disrupt the temporal relationships across the data, but that actually yielded worse performance than when data was reshaped into images in particular 47x68 "images").

[2] Note that, as explained in Section 1.1, the bidirectional layer actually consists of two separate networks with 4 neurons each, and each network outputs the output of its last neuron.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 45, 66, 64)	640
max_pooling2d_1 (MaxPooling2	(None, 22, 33, 64)	0
dropout_1 (Dropout)	(None, 22, 33, 64)	0
flatten_1 (Flatten)	(None, 46464)	0
dense_1 (Dense)	(None, 128)	5947520
dropout_2 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 2)	258

Total params: 5,948,418
Trainable params: 5,948,418

Figure 4: CNN Architecture

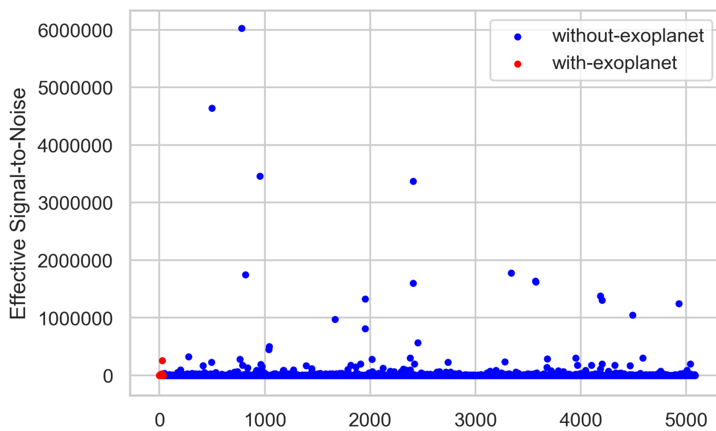


Figure 5: BLS signal-to-noise ratio for the training set.

4 Results

4.1 Box Least Squares

Figures 1 and 2 show some example results of BLS for a with-exoplanet and without-exoplanet star, respectively. The highlighted ranges in Figures 1a and 2a correspond to the periods with maximum power in Figures 1b and 2b, respectively. BLS seems to very accurately and precisely find the periodic transit in Figure 1a, but, as expected, has difficulty finding such a transit in Figure 2a.

The lowest SNR in the training set was 12.24, and the highest was approximately 6 million, which seems to be at odds with the findings of Kovács et al. We believe this is due to running BLS with different parameters (e.g. the granularity of our period search was 20 hours, and we used Astropy's default of 10 bins as opposed to the 100 bins used by Kovács et al.). Therefore, we sought to find an optimal SNR threshold ourselves. In Figure 5, we have plotted the SNRs for the training set. Unfortunately, there does not seem to be a clear distinction between with-exoplanet and without-exoplanet SNRs. Using the ROC curve on the training set as guidance, we handpicked an SNR threshold of 200 so that an SNR greater than 200 resulted in a with-exoplanet classification, and an SNR below 200 resulted in a without-exoplanet classification. Running this on the test set resulted in the performance metrics and confusion matrix shown in Figure 6.

Although this classifier does correctly detect all 5 with-exoplanet stars in the test set, it also results in many false positives, probably too many to be useful.

Accuracy	AUC	F1	Recall
0.447	0.790	0.031	1.0

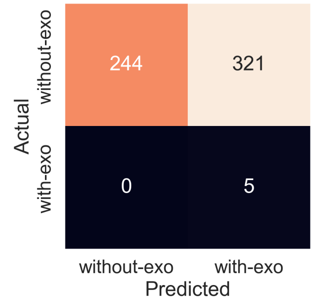


Figure 6: Performance of the baseline BLS classifier.

Model	Accuracy	Precision	Recall	F1-Score
SVC	99.12%	0	0	0
LinearSVC	72.81%	0.01923	0.6	0.03727
Random Forest	53.33%	0.01487	0.8	0.0292
Decision Tree	97.89%	0	0	0
MLP	84.21%	0.0107	0.4	0.02083
KNN	95.09%	0.04	0.2	0.06667
Logistic	51.05%	0.01418	0.8	0.02787

Figure 7: Performance of classical classifiers.

4.2 Classical Classifiers

In Figure 7, we show the performance metrics of our "classical classifiers" on the feature-engineered dataset (20 BLS features + 4 Fourier + 4 Discrete Wavelet). Although models like SVC, Decision Tree, and KNN have high accuracies, they are unable to achieve our ultimate goal, namely that of correctly classifying most/all of the with-exoplanets star in the test set, as is evident by their low recalls. In addition, although Random Forest classifies 4/5 with-exoplanet stars correctly, its accuracy is probably too low to be useful. Figure 12 in the appendix breaks these results down further through confusion matrices.

4.3 Deep Learning

The test set performance metrics for our two deep learning methods are shown below:

Model	Accuracy	AUC	F1	Recall
LSTM RNN	0.798	**0.759**	0.065	**0.8**
CNN	**0.971**	0.667	**0.211**	0.4

4.3.1 LSTM RNN

The loss and accuracy of the RNN LSTM over 10 epochs with a batch size of 32 are shown in Figure 8 (remember that the training set is upsampled while the test set is not). The results are not spectacular, but they are better than our other models so far, and the RNN LSTM did successfully detect four of the five with-exoplanet stars. We did try several larger architectures, both wider and deeper, but this quickly resulted in severe overfitting (even with our very small architecture, we found that a dropout layer helped reduce overfitting). Although larger networks produced better *accuracy*, this seemed to be because the network was essentially "memorizing" the light curves of the with-exoplanet stars in the training set rather than finding characteristic patterns among them. Larger networks had very good accuracy on the training set, and also on the test set since simply classifying all stars as without-exoplanet resulted in $> 99\%$ accuracy. However, looking at our other metrics (recall, AUC, F1 score), we decided a small architecture was best since it was able

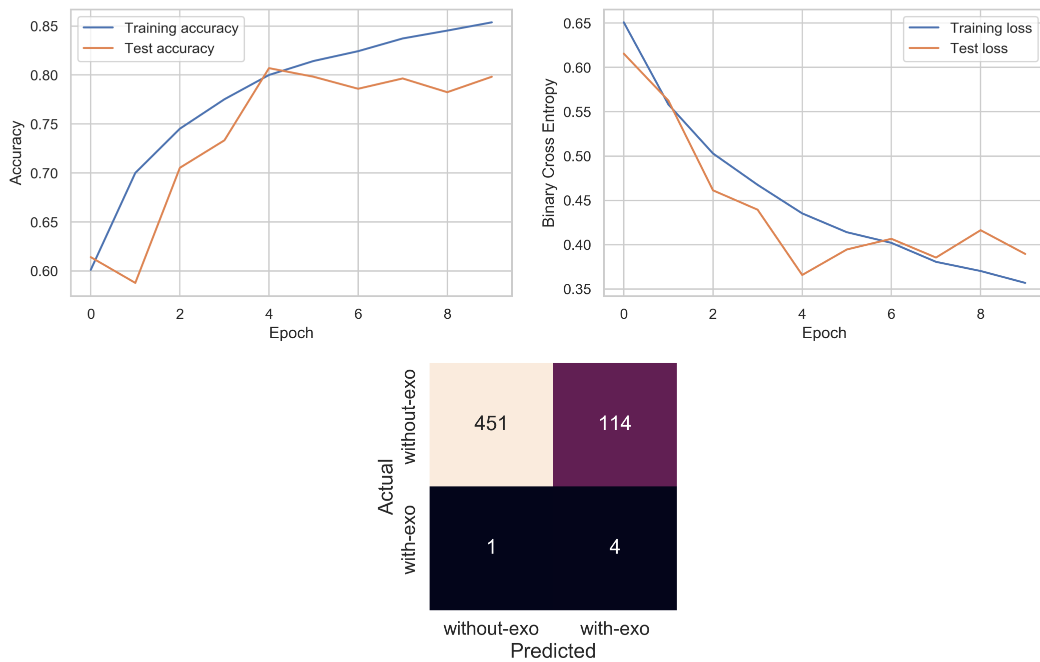
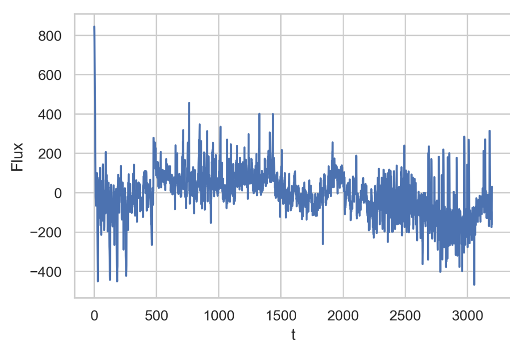
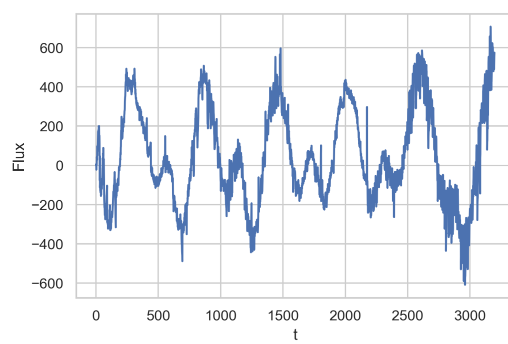


Figure 8: Performance of the LSTM RNN classifier.



(a) False negative light curve.



(b) Most confident false positive light curve.

Figure 9: Light curves incorrectly classified by the LSTM RNN classifier.

to correctly classify most of the with-exoplanet stars. Unfortunately, it may simply be infeasible to very effectively train such a network with so few with-exoplanet stars in the training set.

To get a better understanding of the LSTM RNN's performance, we have plotted the light curve of the one false negative and the light curve of the most confidently classified false positive in Figure 9. At least with our (admittedly inexperienced) eyes, these incorrect classifications seem reasonable. There does not appear to be a clear periodic transit in the false negative light curve, and there does appear to be a quite clear periodic transit in the false positive light curve. However, after speaking with astrophysics professor Joshua Winn, we believe the false positive light curve may be that of an eclipsing binary star, which is actually made up of two stars that periodically eclipse each other. These curves are distinguished by their periodic V-shapes, so with more training data, it should be possible to distinguish such stars through machine learning methods.

4.3.2 CNN

In addition to training an LSTM RNN, we also trained a CNN. Initially, we decided to construct a 1D CNN so as to not perturb the temporal correlations present in the data. However, to our surprise, we found that the CNN performed better when the data was arranged into 2D "images" (ie. each training instance was reshaped into a 47x68 array). We believe that although reshaping the data into an image might disrupt some temporal relationships, most if not all data instances change at high enough frequencies that each row of the images can capture multiple periods in the data. Therefore, the 2D format mostly preserves the periodicity of the light curves, while also allowing the neural network to "see" multiple parts of the light curve at the same time. Ultimately, as shown in the confusion matrix in Figure 10, the CNN was able to classify 2 out of the 5 with-exoplanet stars in the test set correctly, while still classifying most of the without-exoplanet stars correctly (551 out of 565, ie. 97.1% accuracy).

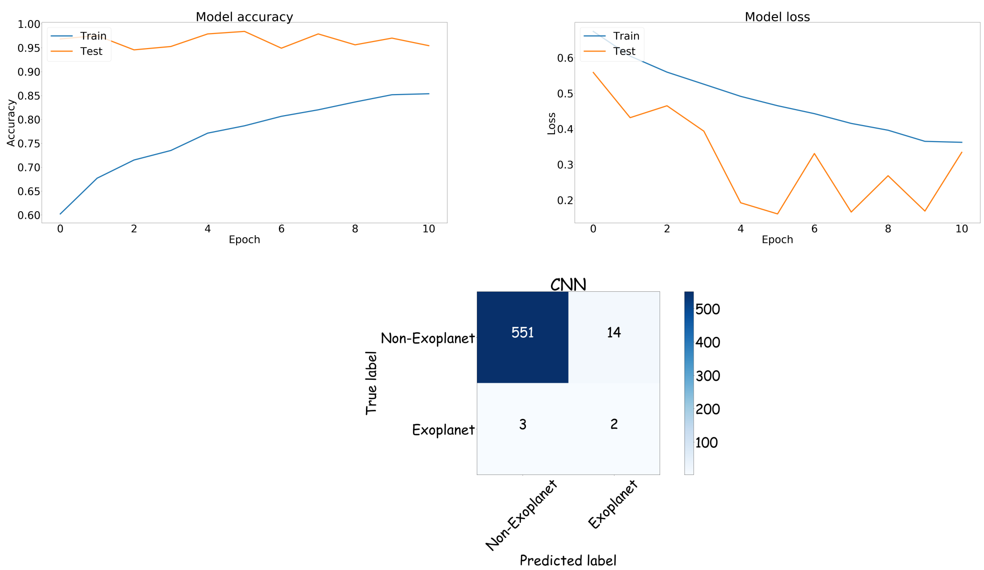


Figure 10: Performance of the CNN classifier.

5 Discussion and Conclusion

Although none of our models had spectacular results, deep learning methods definitely seem promising, significantly outperforming our baseline BLS model in three out of four of our performance metrics, and coming close in the last one (recall). Our LSTM RNN only produced one false negative while maintaining a decent level of accuracy in the test set. We believe further experimentation with data augmentation, including rotation, noise addition, and possibly scaling, could produce even better results with our very limited set of with-exoplanet data. It could also be interesting to examine what *kinds* of planets are classified well by our machine learning methods. Our data set did not include descriptive information about the stars or exoplanets, but other data sets with such information could help guide the process of feature engineering and model selection.

For future study, one could experiment with more sophisticated neural network architectures. For example, Karim et al. [11] were able to achieve state-of-the-art performance in time series classification using a LSTM Fully Convolutional Network, which combines the outputs of an LSTM network and a convolutional network.

In addition, the issue of an extremely unbalanced data set with few with-exoplanet light curves may soon be solved by the Transiting Exoplanet Survey Satellite (TESS), which launched in April 2018 and will be surveying 200,000 of the brightest stars near the sun over the course of two years [2]. It will cover a sky area 400 times larger than that monitored by the Kepler telescope, and will provide invaluable data for exoplanet discovery.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ...Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation*, 265-283. Retrieved from https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

[2] About TESS. (n.d.). Retrieved from https://www.nasa.gov/content/about-tess

[3] https://www.scipy.org/

[4] https://pywavelets.readthedocs.io/en/latest/

[5] Robitaille, T. P., Tollerud, E. J., Greenfield, P., Droettboom, M., Bray, E., Aldcroft, T., ...Streicher, O. (2013). Astropy: A community Python package for astronomy. *Astronomy & Astrophysics, 558*. doi:10.1051/0004-6361/201322068

[6] Box least squares (BLS) periodogram. (n.d.). Retrieved from http://docs.astropy.org/en/stable/stats/bls.html

[7] Classification: Precision and Recall. (n.d.). Retrieved February 17, 2019, from https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

[8] Classification: ROC Curve and AUC. (n.d.). Retrieved February 17, 2019, from https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[9] Exoplanet Hunting in Deep Space. (2017, April 12). Retrieved from https://www.kaggle.com/keplersmachines/kepler-labelled-time-series-data

[10] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks, 18*(5-6), 602-610. doi:10.1016/j.neunet.2005.06.042

[11] Karim, F., Majumdar, S., Darabi, H., & Chen, S. (2018). *LSTM Fully Convolutional Networks for Time Series Classification. IEEE Access, 6*, 1662-1669. doi:10.1109/access.2017.2779939

[12] Kepler Guest Observer Program. (n.d.). Retrieved from https://keplerscience.arc.nasa.gov/K2/GuestInvestigationsC03.shtml

[13] Kovács, G., Zucker, S., & Mazeh, T. (2002). A box-fitting algorithm in the search for periodic transits. *Astronomy & Astrophysics, 391*(1), 369-377. doi:10.1051/0004-6361:20020802

6 Appendix

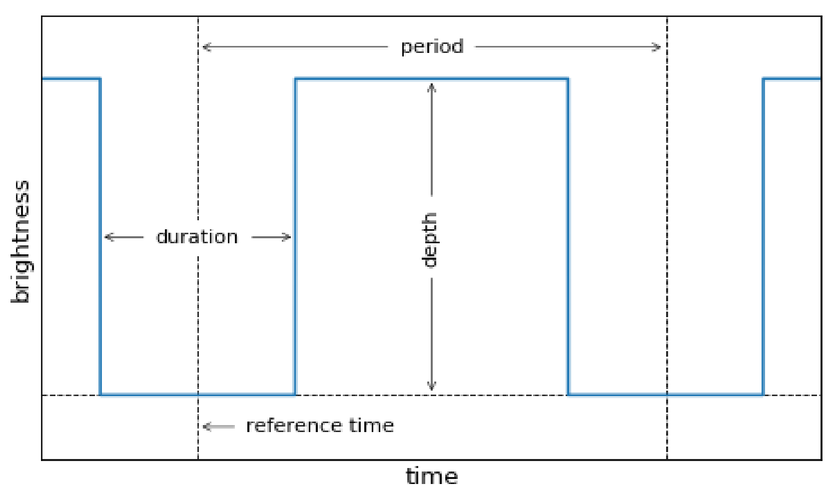


Figure 11: The box least squares model [6].

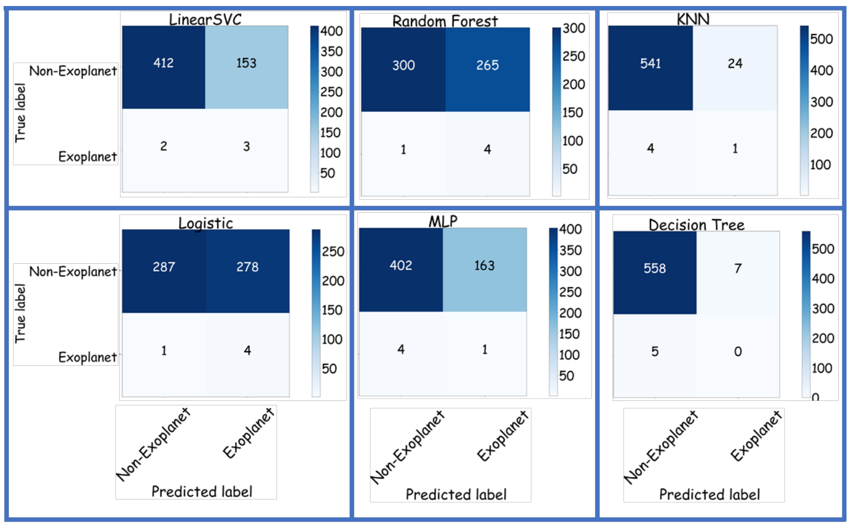


Figure 12: Confusion Matrices