Creating an Automated Classifier to Approximate the By-eye Selection of HAT Exoplanet Candidates

Will Coulton
Department of Physics
Princeton University
wcoulton@princeton.edu

Joshua Wallace
Department of Astrophysical Sciences
Princeton University
joshuajw@princeton.edu

Abstract

Machine learning techniques are applied to aid in the binary classification of data from the Hungarian-made Automated Telescope (HAT) arrays. The HAT arrays are designed to detect periodic dimmings of stars, potentially caused by orbiting planets around those stars. There are many signals, both physical and spurious, that can approximate planet signals, so it is important to weed out as many false positives as possible. After an automated pipeline removes probable false planets, a final manual by-eye selection is made to create the final set of planet cadidates. We train classifier models to approximate this by-eye selection using ∼30,000 labelled data points with the hope of finding a sufficiently well-performing model that can stand as a proxy for or even entirely replace the manual portion of the selection pipeline.

1 Introduction

The field of exoplanets ("extra-solar planets", or planets around other stars) has established itself as the hot new field in astrophysics. Since the discovery of the first exoplanets in the early 1990's [15], nearly 3000 exoplanets have been discovered [6], with potentially thousands more exoplanets remaining unconfirmed or thus far undiscovered in existent data. This explosion in exoplanet discovery has been fueled by the funding and construction of many exoplanet detection surveys. The majority of exoplanets have been found by the space-based *Kepler* telescope and its primary [3] and "K2" surveys [10]. Ground-based surveys, while not as sensitive to smaller planets as *Kepler* (due to atmospheric distortion of image quality and other problems associated with observing from the ground), are sufficiently sensitive to discover Jupiter-sized planets on short-period orbits. Such planets are called "hot Jupiters" because they are the similar in size to Jupiter but are sufficiently close in to the stars they orbit (much closer than Mercury is to our sun) that the star heats them up to several thousands of degrees Celsius.

Since ground-based surveys are much cheaper to run per area of sky monitored than spaced-based surveys, the majority of known hot Jupiters have been discovered by large-scale ground-based surveys. Of the few hundred hot Jupiters that have been found to date, the plurality (∼100) have been found by Princeton's own Hungarian-made Automated Telescope (HAT) collaboration, headed by Prof. Gaspar Bakos of the Department of Astrophysical Sciences. This collaboration runs two surveys. The longest-running of the two surveys is HATNet, a collection of five telescopes in Arizona and two telescopes in Hawaii [7]. The other survey is HATSouth, which consists of three locations with eight telescopes each: Chile, Namibia, and Australia [2]. A third survey, HATPI, which will consist of 63 telescopes in Chile, is currently being constructed [4]. The telescopes themselves are small as telescopes go: only about 10 cm across, they are more similar to camera lenses than the "normal" telescopes usually associated with astronomy. Figure 1 shows a picture of four of the HAT-South telescopes.

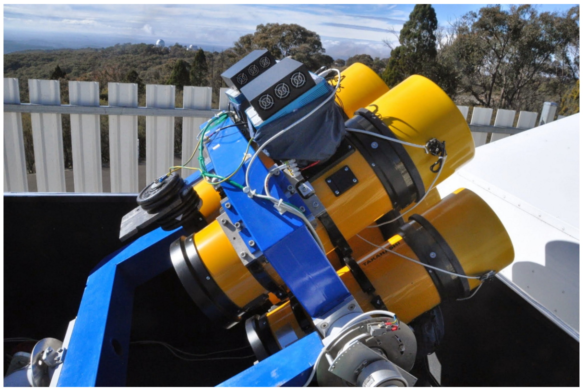

Figure 1: The four HATSouth telescopes at Siding Springs Observatory, Australia. Photo credit Gaspar Bakos. From [5].

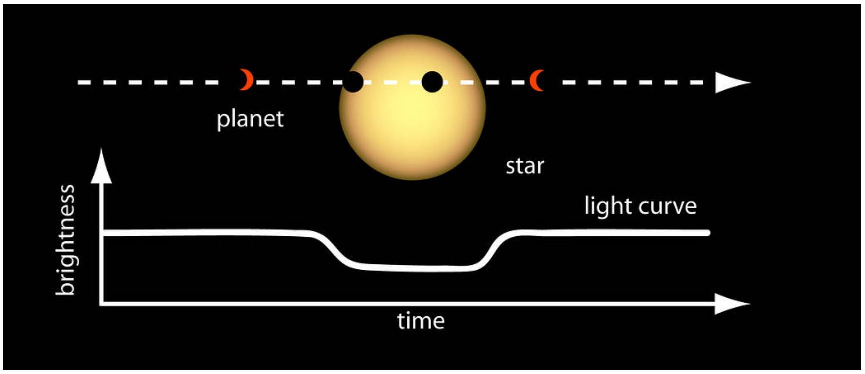

Figure 2: An example "light curve" (brightness as a function of time) from a star that hosts a transiting exoplanet. As the planet crosses in front of the star, the measured brightness from the star is less than the out-of-transit value. Partial eclipses as the planet is transitioning from being out-of-transit to fully in transit put a slope in the light curve between in-transit and out-of-transit magnitudes. Public domain image from [1].

The detection technique used by HAT*, *Kepler*, and many other exoplanet surveys is the "transit" technique. Planets are far too faint compared to their host stars to be able to directly image around most stars. Instead, in the transit technique, the brightness of stars is monitored as a function of time (usually by taking periodic long-exposure photographs of large areas of the sky). When a planet crosses in front the star it orbits, the amount of light detected from that star is decreased. This is demonstrated cartoon-style in Figure 2. Transits are found in data using matched filtering (with either a box- or trapezoid-shaped filter, matching the signal from a transiting exoplanet), folding the data on a variety of periods and phases. When a star is found to have periodic dips in brightness, it is labeled as a potentially planet-hosting star. However, transiting planets are not the only source of periodic signals like that seen in Figure 2. Starspots rotating in and out of view also provide periodic dips in brightness and binary stars that eclipse each other in a grazing way (so the amount of light that goes missing during an eclipse is about the same as a planet) are two examples of astrophysical false positive signals. Various cuts are made to help ensure that a prospective planet is neither any of these false positives nor a spurious signal. The various parameters of the fit to the transit signal (depth, time of ingress, length of transit) are often good indicators to separate between planet and not. These cuts are automated. However, after the cuts, there are still false positives that are not apparent to the pipeline but are obvious to a trained human eye. Thus, the final step in the HAT pipeline is a manual, by-eye examination of the signal. Planet candidates that pass this step then are sent for follow up (and hopefully confirmation!) at bigger telescopes.

Other than this manual step, the vetting of possible planet candidates is completely automatic. This single manual step is quite labor intensive, and prevents a fully automatic characterization of planet detection efficiency (what fraction of real planets get correctly identified), which is necessary for the calculation of statistics related to the occurrence rate of these planets. Thus, it would be very nice to have an automated approximation to the by-eye work that has occurred. This is the purpose of this project.

2 Related Work

The premier (and only statistically robust) calculation of a hot Jupiter occurrence rate from transit data is that from the primary *Kepler* mission [9]. Because of the exquisite quality of the data from the *Kepler* telescope, their detection efficiency for hot Jupiters is near 100% and they are better able to automatically identify false positives than the HAT collaboration is from their surveys. Because of this, the *Kepler* pipeline needs much less manual vetting of planet candidates and thus the *Kepler* team has not needed to worry about the manual step as the HAT collaboration does. However, as mentioned before, the HAT collaboration has found more hot Jupiters than any other collaboration, and so if we can get a good handle on our detection efficiency we have the potential to provide the most precise measure of the occurrence rate of these planets, which will help inform planet formation theory.

Unfortunately, since the *Kepler* work did not need to worry so much about detection efficiencies and pipeline misclassifications, they did not develop the hard statistical/machine learning tools that would be useful for this work. However, a recent astrophysics study [12] that used a very similar time-series data set as the HAT surveys used a random forest classifier to classify 450,000 eclipsing binaries (binary stars that transit each other, similar to how a planet transits a star) from their data.

3 Data

The data used are (currently) unpublished and proprietary, though there are efforts being made to make the data public. They were extracted from the HAT database using a script provided by Joel Hartman[1], Research Scientist at Princeton University. The database itself currently has no reference.

The data consists of 30,445 prospective planet candidates identified by the pipeline. Of these, 1024 passed the manual vetting (positive cases) and 29,421 failed (negative cases). The number of positive cases is perhaps too low to train a truly robust model, but it is unfortunately all we have to work with. The process of discovering new planets is one of constant attrition, and there are unfortunately more cases of false positive signals than of real signals. There are 92 features that consist of such measured quantities as the fit parameters to the transit dip (transit depth, length, etc.), quantities related to the power spectrum from the matched filtering, measured quantities for the star (brightness in various colors), and signal to noise measures. All of these quantities are continuous, not categorical. There are some missing values: 284 "inf" (0.01% of all values) and 4461 "nan" (0.2% of all values). The "inf" values are mostly concentrated in a single column (feature) of the data, while the "nan" values tend to cluster in the same rows (objects). These data were imputed using the SoftImpute package with 30 components [8]. The soft-impute package preforms a thresholded singular value decomposition of the masked data set and then uses this representation to impute the missing values.

4 Methods

4.1 Unsupervised Learning

We used some unsupervised learning techniques to see if there were any obviously exploitable structures in the data that aligned well with the classifications. We use the K-means, principal component analysis (PCA), Gaussian mixture model (GMM), and latent Dirichlet allocation (LDA) algorithms from `scikit learn`[11]. Data are scaled prior to the PCA analysis, and features are re-centered to avoid negative values for the LDA algorithm. Clusters and cluster membership are examined to

[1] jhartman@astro.princeton.edu

see if any clusters have predominately and most of the positive examples. We also use the PCA-transformed data in supervised learning models.

4.2 Supervised Learning

We used several supervised classifiers to attempt to predict the class label for our data set. We consider the following classfiers from `scikit learn`[11]: logistic regression (with L_1 and L_2 penalties), naive Bayes, SVM (with polynomail and RBF kernels) and four variations on decision trees; random forest, extra-trees, gradient boosted trees and adaboost trees. We used mutual inforamation to select the best n-features, where n was choosen by 25-fold cross-validation. For all the methods with hyperparameters, we fit them using 25-fold cross-validation. Finally we combined all of these classifiers together and used them in a combined manner. For the combined classifier we used two methods, first where the classifiers 'voted' on the label with the label set by the majority vote. In the second method we weighted the predicted labels by the classifier's prediction probability.

4.3 Focus on One Algorithm: Logistic Regression

This section benefitted from [13]. The logistic function is defined as

$$f(\sigma) = \frac{e^{\sigma}}{e^{\sigma}+1} = \frac{1}{e^{-\sigma}+1}. \tag{1}$$

In many cases, including logistic regression, it is useful to set σ equal to a polynomial to allow for more flexible fitting to data. As an example, we use first-degree polynomial $\sigma = kx - b$. The parameter k controls the "steepness" of the logistic fit and b is used as a translation parameter, with x now becoming the independent variable,

$$f(x) = \frac{1}{e^{-(kx-b)}+1}. \tag{2}$$

One can think of the logistic function as modeling exponential growth with a cap. An example of such a situation is population growth in a location with a maximum "carrying capacity" (i.e., the environment can only support a maximum number of individuals). Indeed, the logistic function is used as a tool modeling real-world population growth [14].

In statistical modeling, logistic regression is usually used to model a binary classification/outcome on a continuous domain. Logistic regression can be extended to more than two classifications/outcomes in the form of multinomial logistic regression. Since the problem addressed in this work is a binary classification problem, we focus on "normal" logistic regression.

After a logistic function has been fit to given data, either a soft or a hard classification can be made of any subsequent data. For a soft classification, a Bernoulli distribution can be used to assign probabilities of the two classifications using the value obtained from the logistic function. More specifically, if $f(x)$ is the value of the logistic fit,

$$P(Y = y|x) = f(x)^y(1 - f(x))^{1-y}, \tag{3}$$

where y is the classification (either 0 or 1 in the binary case), x is the data, and Y is the assigned classification. The nature of the logistic function is that as x ventures far away from the value of $x = b$ (the translation parameter/median of the distribution), the probability will converge on a single classification being dominant, while at b there is a 50/50 probability of both classifications. For hard classification, a cutoff value for probability is chosen to assign a single classification to the data. For example, if a greater than 0.2 probability that $Y = 1$ is chosen as the cutoff value, then all data points with $P(Y = 1|x) > 0.2$ will be assigned $Y = 1$, and all other data points will be assigned $Y = 0$. In practice, the cutoff value is a hyperparameter that can (and should!) be tuned to give the best classification results.

5 Results

5.1 Unsupervised Learning

There were no unsupervised methods that provided obvious structure that matched the classifications. For the K-means method, a variety of k-values were used. In each case, the positive cases

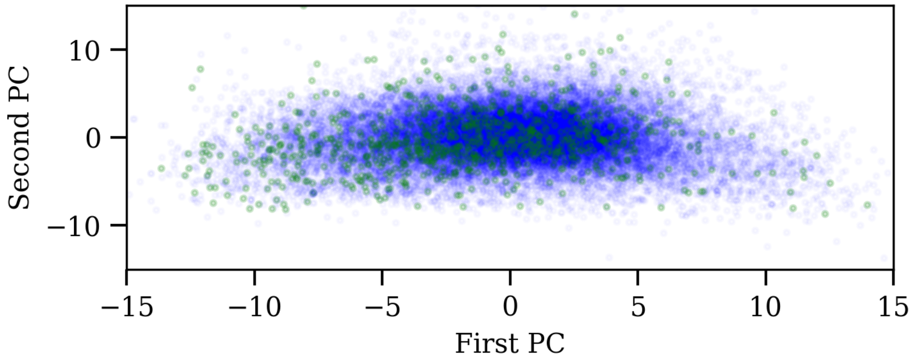

Figure 3: The data projected on the first two principal components. The positive cases are in green and the negative cases are in blue. Note that the scales of the axes are the same; they have been stretched and squished to better fit in the single-column format of this paper. The transparency of the green points is less than the blue points so that they could be better visible, since there are about 30 times more blue points than green points. There is no obvious separation between the two groups of points, though the green points do have larger variance than the blue points along the first principal component (22.6 versus 17.7). In the axis labels, "PC" = principal component.

were divided among the categories in rough proportion to the total number of cases in each category. A similar thing happened with LDA. Since LDA is a mixed membership model, we decided to assign hard membership if an object had larger than a certain threshold (e.g., 0.8 or 0.9) membership in a particular topic. Members of a particular topic were distributed between positive and negative examples in rough proportion to the actual distributions of the examples in the data set. GMM, for small values of `n_components`, was able to do better than just randomly pulling from the dataset for its category assignment. For example, for `n_components` = 4, one of the categories (using a posterior probability threshold of 0.9) had ∼25% of its members being positive examples and another category had only ∼0.1% of its members being positive examples. If the GMM category assignment was nothing more than random draws from the data, then both of these percentages would be around 3.3%. (The other two categories have ∼1–2% of their members being positive examples). Thus, GMM was able to uncover some features in the data that allowed for a small differentiation between positive and negative examples. Unfortunately, only ∼10% of the positive examples end up in the positive-heavy category, which itself still consists ∼75% of negative examples, so the GMM categories themselves are not sufficiently discriminatory to be used as our model.

For PCA, the first principal component explains 20% of the variance, then 11%, 8.5%, and 7.8% for the next three principal components. Figure 3 shows the projection of the data onto the first two principal components. The green circles are the positive cases and blue dots are the negative cases. Although the variance between the two sets of points is different (22.6 along the first principal component for the green points versus 17.7 for the blue), from just the first two principal components there is no exploitable separation between the positive and negative cases that allows for classification. Running K-means (with several different values of k) on the PCA-decomposed values also revelead no exploitable structure for classification. Running GMM on the PCA-decomposed values provided clustering with worse predictive power than GMM on the normal data (the "best" cluster only had ∼13% of its members being positive cases, instead of ∼25% as was the case before).

5.2 Supervised Learning

We focused on four metrics to assess our models. These were the precision, recall, negative predictive value (NPV; the "precision" of the negative cases), and specificity (the "recall" of the negative cases). It should be noted that our data set is very imbalanced, with many more negative cases (29,420) than positive cases (1024). The result of this was that for almost all of estimators we achieved high NPV and specificity, thus the more interesting metrics are the precision and recall. Also, since we care more about the accuracy positive cases (these are what go on to hopefully be

Classifier	NPV	Specificity	Precision	Recall
NaiveBayes	0.984	0.967	0.369	0.555
GradientBoosting	0.979	0.992	0.628	0.386
RandomForest	0.975	0.988	0.429	0.263
ExtraTrees	0.974	0.987	0.407	0.252
AdaBoost	0.973	0.992	0.502	0.202
LogisticReg	0.969	0.995	0.3767	0.080

Table 1: The best fit results for our classifiers. The hyper-parameters were fit with cross validation and their efficiency was evaluated using cross validation with 25 folds. We found that the Naive Bayes, with a Gaussian model, and the Gradient Boosting decision trees fit the data best. Note random guess would give Recall and precision rates of ≈ 0.03.

Classifier	NPV	Specificity	Precision	Recall
All classifiers - Hard	0.969	0.995	0.368	0.077
All classifiers - Soft	0.969	0.995	0.365	0.074
Best Two - Hard	0.979	0.992	0.634	0.383
Best Two - Soft	0.978	0.992	0.612	0.373

Table 2: The best fit results for our combined classifiers. The classifiers were fit above and then combined. Their efficiency was evaluated using cross validation with 25 folds. We found that combining the best two classifiers gave the best result, but that it was not better than the individual classifiers.

planet discoveries after all), precision and recall are more important than NPV and specificity (although the values of each are related to each other).

In Table 1 we present the average precision and recall of our classifiers on the with-held data set averaged from 25 fold cross-validation. The results show that the best methods are the Naive Bayes and Gradient Boosting methods. Unfortunately we did not have sufficient computing rescources to fit the SVM methods to our data (after 160 cpu hours the SVM model was not fit). In Table 2 we show the results of combining our classifiers and using soft and hard voting methods. We found that we achieved the best results in this method by only using the best two classifiers rather than all of the classifiers however this was not an improvement on the separate classifiers.

6 Discussion and Conclusion

Despite the presence of different physical effects in our data, we were unable to train an unsupervised classifier to disentangle these effects. It is likely that these physical processes appear too similar in our current data set. The unsupervised methods still provided an interesting exploration of the data. The supervised methods provided significantly better results with all of our classifiers vastly outperformed random guessing. The best methods were the Naive Bayes and the Gradient boosted method, a priori we expected the AdaBoosting method to perform best and are unsure why this did not perform better.

Our classification problem is quite difficult as the boundary between accepted and rejected candidates is hard to define with many of the accepted cases being difficult, even for a human, to distinguish. Thus we were happy with the levels of performance obtained by our classifiers, though further improvement is still desired. We hope that by performing more careful feature engineering and by including the images, which our features were derived from, that we can improve further. Due to time limitations we were unable to fully study the effect of changing the probability to accept and reject candidates. It would interesting to train our models to find the best cutoff probability, this could be treated as a hyperparameter and selected by cross-validation.

In future work we would like to more carefully examine how to weight the classifiers when combining them; currently all the classifiers have equal weight in the voting, despite the fact that some of the classifiers work much better than the others. Beyond this we plan to incorporate these classifiers in a statistical model to estimate the occurrence of hot Jupiters, a step that had been impossible for the HAT data with human selection of candidates.

Acknowledgments

We are grateful to Joel Hartman for providing a script to extract the necessary data from the HAT database. We are also grateful to the entire HAT team (PI: Gaspar Bakos) for their many dedicated years of constant observations and the huge pile of astronomical data they've collected. We are grateful to Adrian Price-Whelan for the excellent examples of PCA he provided in AST 542, which inspired part of our work here.

References

[1] NASA Ames. Light curve of a planet transiting its star. `https://www.nasa.gov/mission_pages/kepler/multimedia/images/transit-light-curve.html`. Accessed: 2017-04-27.

[2] G Bakos, C Afonso, T Henning, A Jordán, M Holman, RW Noyes, PD Sackett, D Sasselov, Gábor Kovács, Z Csubry, et al. Hat-south: a global network of southern hemisphere automated telescopes to detect transiting exoplanets. *Proceedings of the International Astronomical Union*, 4(S253):354–357, 2008.

[3] William J Borucki, David Koch, Gibor Basri, Natalie Batalha, Timothy Brown, Douglas Caldwell, John Caldwell, Jørgen Christensen-Dalsgaard, William D Cochran, Edna DeVore, et al. Kepler planet-detection mission: introduction and first results. *Science*, 327(5968):977–980, 2010.

[4] HATPI collaboration. The hatpi project. `https://hatpi.org/`. Accessed: 2017-04-27.

[5] HATSouth collaboration. The hatsouth exoplanet survey. `https://hatsouth.org/operations.html`. Accessed: 2017-04-27.

[6] exoplanets.org. Exoplanets.org main page. `http://exoplanets.org/`. Accessed: 2017-04-21.

[7] JD Hartman, G Bakos, KZ Stanek, and RW Noyes. Hatnet variability survey in the high stellar density "kepler field" with millimagnitude image subtraction photometry. *The Astronomical Journal*, 128(4):1761, 2004.

[8] T. Hastie, R. Mazumder, J. Lee, and R. Zadeh. Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares. *ArXiv e-prints*, October 2014.

[9] Andrew W Howard, Geoffrey W Marcy, Stephen T Bryson, Jon M Jenkins, Jason F Rowe, Natalie M Batalha, William J Borucki, David G Koch, Edward W Dunham, Thomas N Gautier III, et al. Planet occurrence within 0.25 au of solar-type stars from kepler. *The Astrophysical Journal Supplement Series*, 201(2):15, 2012.

[10] Steve B Howell, Charlie Sobeck, Michael Haas, Martin Still, Thomas Barclay, Fergal Mullally, John Troeltzsch, Suzanne Aigrain, Stephen T Bryson, Doug Caldwell, et al. The k2 mission: Characterization and early results. *Publications of the Astronomical Society of the Pacific*, 126(938):398, 2014.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[12] I Soszyński, M Pawlak, P Pietrukowicz, A Udalski, MK Szymański, Ł Wyrzykowski, K Ulaczyk, R Poleski, S Kozlowski, DM Skowron, et al. The ogle collection of variable stars. over 450 000 eclipsing and ellipsoidal binary systems toward the galactic bulge. *arXiv preprint arXiv:1701.03105*, 2017.

[13] Wikipedia. Logistic regression. `https://en.wikipedia.org/wiki/Logistic_regression`. Accessed: 2017-05-16.

[14] Wikipedia. Population growth. `https://en.wikipedia.org/wiki/Population_growth`. Accessed: 2017-05-16.

[15] A. Wolszczan and D. A. Frail. A planetary system around the millisecond pulsar PSR1257 + 12. *Nature*, 355:145–147, January 1992.