Identifying Exoplanet Companions using Radial Velocity Data with Deep Learning

Rohan S. Nagavardhan

November 2019

Abstract

In 1995, the radial velocity method was conceived to find the first extrasolar planet around a Sun-like star in the constellation Pegasus. This method has since been responsible for the detection of a huge number of planets over the past decades. NASA and other groups such as ESO and the SDSS, are amassing vast catalogs of radial velocity data for stars with instruments such as HARPS and MARVELS. The use of Non-Linear Least Squared Regression with the curve fitting function of SciPy helped to analyze radial velocity graphs to extract accurate orbital parameters of stars reflected in NASA's Exoplanet Archive. In this work, we utilized the XGBoost Classifier in order to label whether the orbital parameters for the first planet around a star has other companions through the theory of planet to planet interactions. If a certain star does not have any planets, its change in radial velocity is attributed to a spectroscopic anomaly or a presence of another star. Due to the small data set, the test set and the complete data set were used to evaluate the model's performance. Accuracy percentages as well as the F-measure scores of the model for each evaluation were calculated. After training, the model received an accuracy percentage of *61.7978%* on the test set and *81.2004%* on the entire data set. The average F-measure scores of the three labels on both evaluations reflect the accuracy percentages: *0.61* & *0.81* on the test set and complete data set respectively. These results substantiate the use of orbital parameters from radial velocity analysis and planet to planet interactions for the purpose of exoplanet companion identification.

1 Introduction

Exoplanet detection is an endeavor that has spanned over a decade and a half. Since the discovery of the first exoplanet in 1995, the number of known exoplanets has reached over four thousand and new planets are constantly being found [6]. Deep learning is a relatively recent development in the field of computer science and artificial intelligence. Deep learning technologies have been applied to many different disciplines, such as astronomy, medicine, and weather. Astronomy is a science of observation and naturally lends itself towards large amounts of data collection; therefore, astronomy is a perfect field for application of deep learning algorithms.

Artificial intelligence has recently been used to discover exoplanets leveraging the transit photometry detection method [9]. However, there existed a niche research area in using artificial intelligence in combination with the radial velocity detection method which became the focus of this work. Radial velocity is a spectroscopic detection technique that analyzes periodic shifts in a star's spectrum in order to find a hidden planet [12]. This shift is caused by gravitational and Keplerian laws in action. The planet's gravitational force on the star causes the star to "wobble" and this fundamental theory was the basis of my rationale. If the planet affects the star, then multiple planets must be affecting each other through planet to planet interactions [7].

Using this same rationale, orbital parameters for the first planet (if any) were extracted from the NASA's radial velocity data; then, a deep learning model was created to detect whether there existed companions to that first planet (if any) assuming planet to planet interactions had an affect on orbital parameters (i.e. eccentricity of orbit). Based on the input parameters, the algorithm outputs a "0", "1" or "2" corresponding to a certain classification scheme. A "0" meant no planet exists around the star and the change in spectrum was due to external factors. A classification of "1" means the first planet has no companions. Finally, a classification of "2" means that the algorithm predicts that the first planet has companion planets. SciPy's curve fitting algorithm gave a relatively accurate extraction of orbital parameters [5]. Due to our limited data set as a result of data preprocessing and cleaning I used an XGBoost classifier which outperforms conventional machine learning models on small to medium data sets [3]. Model performance statistics will be discussed further in the Results section.

2 Background

An extrasolar planet, or exoplanet, is defined as any planet that orbits other stars that are not our Sun. Since the detection of the first extrasolar planet around a Sun-like star in 1995, the number of exoplanets detected has grown exponentially, leading to increased investment in exoplanet research. Over the past decade, astronomers have developed various methods to detect exoplanets. The first extrasolar planet, 51 Pegasi b, was detected using the radial velocity of the star; while in more recent times, radial velocity has been complemented by the transit photometry method [6]. As technology and accuracy of sensors has increased, astronomers have defined many methods to detect exoplanets. Below is a list of some of the methods of exoplanet detection. I will further discuss the first two methods as they are more relevant to my project and other literature.

1. Radial Velocity
2. Transit Photometry
3. Astrometry
4. Gravitational Microlensing
5. Direct Imaging

2.1 Radial velocity

The radial velocity method was the first prominent method for detecting exoplanets and is still to this day a very effective method. The radial velocity method (also Doppler Spectroscopy Method) uses the fundamental principles of the Doppler effect but applies them to light rather than sound. The summation of gravitational forces in an n-planet system creates a "wobble" that can be directly observed but more generally is observed through spectroscopy. This "wobble" is the star orbiting around the center of mass of an n-planet system. As we look at the star system, the star moves towards and away from us and this causes a shift in the stellar spectrum of the star. If this pattern of red-shift and blue-shift happens regularly, there is a high probability that an exoplanet is causing this shift in the spectrum [12]. The orientation of the star system relative to observation from Earth is one key limitation of this method. If the system is oriented edge-on the Doppler signature is maximized; however, in an observation where the orientation is "face-on" the subsequent Doppler signature is minimized. The orientation of the observations cannot be controlled;

therefore, some star observations have varying strength in Doppler signature. Using a spectroscopic method such as radial velocity gives astronomers a wealth of information about the mass of the planet which is useful for future planet habitability.

2.2 Transit photometry

Transit Photometry is a very promising exoplanet detection method and has helped NASA in detecting many exoplanets in recent history. Transit Photometry analyzes the orbit of the exoplanet over time and its effect on the flux or brightness of a star. When the exoplanet orbits, from our observation, the planet eventually crosses in front of the star causing a small dip in the star's flux. Transit photometry can pick out many planets because it is an analysis of change in flux and therefore multi-planet systems have a more concrete effect on the flux of a star since multiple planets can cause varying amounts of dampening in flux measurements. This method also allows us to understand the composition of the planet, as light that passes through the planet's atmosphere can be directly observed from telescopes on Earth. Additionally, photometry can help astronomers to gain understanding about the radius of the extrasolar planet. Just like the other detection measurements, transit photometry has one big limitation. The method can only detect an exoplanet if a transit occurs; however, many transits never occur from our observations leaving many objects hidden from discovery. When complemented with a spectroscopic method, transit photometry's efficiency and accuracy is bound to increase; additionally, when the two methods work in conjunction the amount of planetary data is increased.

3 Materials and Methods

In this section, I will discuss the methods of data collection and data analysis that I had to conduct in my project. Additionally, I will examine the various deep learning models that I used for this work.

3.1 Data Analysis

Using the NASA Exoplanet Science Institute's bulk download from the NASA Exoplanet Archive, I was able to access radial velocity data for most stars in the archive, the data was listed in the form of `wget` snippets which contained a direct URL to the radial velocity data for each star [1]. I extracted each URL in the file and saved it to a separate text file

that I would be able to access using Python's I/O features [10].

Most of the code and data analysis in this project will be done using the Python programming language due to its easy of reading syntax and the extensive list of libraries that boost the functionality of language [10]. The requests[1] library is an easy to use HTTP library for the Python programming language. The files had headers and using simple string methods, I extracted the name of the star that was written in the header of the data file. This decision would facilitate file naming and make researching individual stars significantly faster. Since the data had header which included unnecessary information, I had to parse the files and extract just the data. The files were successively named employing the naming conventions mentioned above and saved in the file's original TBL format. The file format was difficult to deal with so I decided to convert all the files into a CSV file (comma separated values) so that data analysis and manipulation would be easier with the `pandas` library. After all the data files were downloaded, I needed to gain an understanding of the data. The data that I pulled from the archive is classified as time series data: a data set that follows a quantity of measurement over a period of time. The quantity that NASA has measured is called radial velocity (RV). Time was measured in a unit called Julian Dates, designed to measure time since January 1, 4713 BC Greenwich noon. For data visualization, I wrote a Python script that would take the radial velocity data in CSV format and plot each point in the data set over the time implemented with the `matplotlib` library [4]. NASA's data files summed to about 900 files, in order to view the change in radial velocity for each star in the archive, I created a script that would generate and save all the graphs of Radial Velocity v. Julian dates in a separate file directory. The nature of radial velocity observations yields periodic results; therefore, the resulting plots should generate graphs similar to Figure 3. However, after running the Python script on all the data files, I was surprised to notice that they lacked any sinusoidal or periodic pattern. If the curve was indeed periodic, then the period could be calculated with the help of an algorithm. NASA's Exoplanet Archive provides a GUI application that can calculate the period based on the data that the user feeds in [1]. The application made use of the Lomb Scargle Algorithm; an algorithm capable of detecting periodicity in unevenly spaced time series data [11]. The algorithm returns an array of power and a subsequent array of frequencies, when graphed displays various signals that the algorithm has uncovered in the data set. I decided to go with the strongest signal (or the frequency with the highest corresponding power) as it was likely that this signal was caused by a major astronomical body. Even if no astronomical body was present around the star I

[1] `https://github.com/psf/requests`

was measuring, it would help later when building a machine learning model. I implemented this algorithm into my program with the assistance of a library called `astropy` [2]. Using the library's `LombScargle` class, I was able to write a script that would plot a periodogram plot which was a graph of the period and its power. For the star, 47 UMa, the period of the strongest signal is around 10^3 days with the highest power of < 0.8. Using this period, I was able to construct a radial velocity curve like Figure 3 but where the radial velocity was changing relative to orbital phase measured in days. Using phase constrains radial velocity measurements to one orbital cycle allowing us to better view the radial velocity curve and its periodic shape. The planet pictured is 51 Pegasi b which has an orbital period of 4.230785 days which is evident by the graph's halt a little after four days [6]. After running the same script for all the data files in my database directory, I analyzed the radial velocity plots for most stars and concluded that with the inclusion of a phase or orbital period, the curve roughly followed a sinusoidal pattern.

From radial velocity literature, I discovered that Equation 1 was the function I would use to fit my radial velocity v. phase graphs.

$$V_r(t) = K_1[\cos(f_1 + \omega_1) + e_1 \cos(\omega_1)] + V_{r0} \tag{1}$$

The radial velocity semi-amplitude (K_1) is like the amplitude of a sine function; it represents the distance from the equilibrium to the crest of the sine curve. The true anomaly (f_1) is an angle at a given position along an orbit. The eccentricity (e_1) is the measure of how far the orbit deviates from a true circle ($e_i = 0$). The argument of periapsis (ω_1) represents the angle at between the position closest to the star and the intersection between orbit and plane of reference called the ascending node. The barycentric radial velocity (V_{r0}) represents the velocity of the center of mass of the orbit.

After understanding how the mathematical model worked, I needed to be able to model all my data using that same model. This type of curve fitting is classified under nonlinear curve fitting. Nonlinear curve fitting was difficult with the data I had and throughout the summer I implemented many solutions using different Python libraries (i.e. TensorFlow, lmfit, and PyMC3) striving to achieve the best results. I, eventually, used the Python library, SciPy, because of its ease of use and its ability to fit non-linear functions using the curve fit function in the `scipy.optimize` class [5] . The function uses Least Squared Minimization in order to find the best fit for the data and in order to do this it takes in an initial guess of parameters and then changes them using an optimization algorithm that will return the best fit parameters. Each parameter was written to a CSV file, after the SciPy fit was completed [5]. The list of parameters was the data that I would use to train my model. The data was

not clean enough to be used for training a machine learning model and therefore I had to go through the data and find any of the outliers and points that would mess with the accuracy of the model later down the line. I decided that I would build a model that would classify orbital parameters into three categories: "0", "1" or "2". A "0" meant no planet exists around the star and the change in spectrum was due to external factors. A classification of "1" means the first planet has no companions. Finally, a classification of "2" means that the algorithm predicts that the first planet has companion planets. Since I was purposefully classifying the parameters, I needed the correct answers of "0s", "1s" and "2s" for each set of orbital parameters; this class of machine learning is called *Supervised machine learning*. With the NASA Exoplanet Archive, I was able to build a query that would retrieve those labels from the actual archive for all the planets that I had. As I sifted through the data, the data contained repeated values and outliers; therefore, I removed the duplicates and in order to remove outliers, I constricted the values of each orbital parameter. Preliminary model tests were indicating that the model was unable to classify "2s" properly. This was an indication that I needed more data. The bulk download file I got from the archive only contained about 1/4 of the actual number of planets in the archive. The query that I used earlier allowed me to add some more data on the remaining planets that were present in the archive. The data that I was returned by the query was somewhat incomplete; therefore, I had to once again strip down the data set until all the data was all complete. Since the model was lacking in accurately classifying "2s", I decided to only add orbital parameters with labels of "2" in order to create a more even distribution of labels. This addition helped model accuracy tremendously as described in the Results section of this work.

3.2 XGBoost Classifier

I trained a model with the use of an algorithm called XGBoost; specifically, the algorithm's classification utility. As opposed to traditional methods in the deep learning field, XGBoost provides state-of-the-art performance with small to medium data sets [3]. The XGBoost algorithm is an enhanced variant of a common machine learning algorithm called Decision Trees. Decision Trees contains a flow of logic and this flow is formed through the use of branches and leaves. Upon training the decision tree, the algorithm generates if-this-else-that statements that constitute the logic of the model. These types of algorithms are multi-purpose as they can be used in regression problems and classification problems.

XGBoost is an enhanced version of basic tree algorithms because they make use of Gradient Boosting [3, 8] which explains the high speed and performance of the algorithm [8]. The

term gradient boosting is comprised of two different processes. Boosting is when weak models are combined in iterations to create a stronger model and a better fit of the training data. Traditional machine learning and deep learning models train by using a learning method called gradient descent; the term gradient in gradient boosting stems from this method. Gradient boosting algorithms work by fitting a new classifier model by minimizing the losses of the older classification models rather than changing the weights of the each successive classifier in conventional boosted algorithms [8]. This process enhances the training and performance of decision tree models creating stronger and better fit models of real world data.

Minimizing training time is an important aspect of building a deep learning model. Deep learning requires the normalization of the data set before training because of a natural variation in the data set; this process allows models to learn faster. By sifting through the data, I realized there was variations in the orbital period data series which I flagged as a potential problem when training the XGBoost model. Instead of using standard statistical normalization techniques, I decided to use the absolute value of the natural log of the orbital period. This technique essentially made sure that the orbital period was within similar ranges of the other features instead of the great variation I saw before normalization. This simple change to the data set boosted model accuracy which will be further discussed in the Results section.

4 Results

As opposed to other types of machine learning models (i.e. linear regression or logistic regression), performance of a classification model can not be measured solely through the use of an accuracy percentage. In order to measure my model's performance, I used both the accuracy percentages and F-measure scores. Additionally, I used a confusion matrix to analyze which labels the classifier was misclassifying for further analysis of the model's performance.

Due to the nature of the bulk download data on the NASA Exoplanet Archive, the data that I had collected ended up with duplicates which needed to be removed [1]. Although the data set would have a greater number of points, the model would overfit because the it has already seen some of the duplicated data points. I mitigated the effects of reduced amounts of data by evaluating the model on the test set and the full data set. Confusion Matrices allow me to understand the performance of the model by seeing how it is classifying the

input data. In my case, the confusion matrix shows the how the model is categorizing the orbital parameters in to the 3 given classification labels. The confusion matrix of the test set in Figure () shows that when a given a input vector of orbital parameters ($\vec{p}$ [2]) has a *true label* of '0', the classifier accurately classified it as '0' seventeen times in the entire test set. Even though it was classifying '0' a majority of the training time, the model classified eleven '0's as '1' instead. This misclassification is reflected again in the row analyzing the classification of the label, '2', in the data; this is likely caused by minimal amounts of data for both the two classification labels. When trained on the complete data set, the results of the confusion matrix indicated a better model and a subsequent increase in accuracy. The model was doing a better job at differentiating between the three labels and thus classifying majority of $\vec{p}$'s accurately.

To evaluate the model's performance, further, I calculated the model's accuracy percentage and F-measure score for both evaluations. The accuracy percentage measures how many orbital parameters the algorithm correctly classifies out of the test set (or data set). Although accuracy percentage will indicate the strength of the model, it is quite often never a true indication of how well a model has performed. On the test set evaluation, our model scored an 61.7978%; out of the entire test set, it was able to accurately classify more than 50% of the orbital parameters in the test set. This accuracy is indicative that that even with the available data, the model was decently accurate at classifying all the $\vec{p}$'s in the test set. The accuracy score for the training evaluation on the complete data set is far more promising. Upon testing the model on the entire data set, we achieved an accuracy of 81.2004%. With a loss of only 10% of the total data, the accuracy decreased illustrating the importance of data in this specific machine learning problem. The F-measure score measures the performance of the model using both precision and recall. Precision measures the proportion of positive results that are actually positive; however, recall is the ability to classify positive results correctly. Our model achieved a F-measure score of *.61* on the test set and *.81* on the entire data set. The precision and recall of the model on both training evaluations indicate a decently strong model in identifying exoplanet companions.

5 Discussion

Conventional machine learning models require large amounts of data for training. The performance of state of the art models is dependent on the amount of training data present

[2] $\vec{p}$ refers to the input vector of orbital parameters

for the model to learn from. However, certain machine learning problems have a lack of valuable data sets as opposed to others. In the exoplanet detection space, transit photometry data is far more abundant because of its important role it has played in recent history. The bulk data from the NASA Exoplanet Archive had a radial velocity data file associated to a star from the archive. Some stars were less observed than others; therefore, files for one star contained substantially less data points than another. This made a non-linear regression difficult and thus some RV files were removed from the analysis directory. I searched for stars not already present in the bulk RV data from the NASA Exoplanet Archive, but the amount of data of orbital parameters still lacked sufficient data points. My model's performance is merely a stepping stone and will improve as space agencies discover more planets using the radial velocity method.

There are many instruments in the world, today, that measure the spectroscopic shifts of stars to derive the radial velocity of stars that data scientists will later use. Some examples of such instruments include HARPS at the La Silla Observatory. The SDSS has created a radial velocity survey called MARVELS which observes a star for a set period and then derives the radial velocity for that star from the spectrum. Astronomers and engineers are always making new telescopes and instruments to help observe the universe taking advantage of newer technology that was not present at the start (i.e. JWST replacing the Hubble Telescope). If radial velocity instruments get upgrades in the near future, astronomers could possibly see a spike in new radial velocity data that will help my model learn and classify more accurately. Even without the presence of upgraded radial velocity instruments, astronomers can create simulations using orbital mechanics in order to generate simulated radial velocity data. Simulated radial velocity can be conducted on a laptop or desktop removing any observational obstacles out of the way. We would be able to produce large amounts of data with relative ease and add the simulated data to existing radial velocity catalog in order to create bigger data sets for future machine learning models targeting this problem or other exoplanet related problems.

6 Conclusion

In this work, we presented an approach to exoplanet companion identification using the radial velocity method with the assistance of deep learning technology. Our XGBoost Classifier is capable of identifying whether a certain set of orbital parameters has any companion planets. Our model performs well on both the test and full data set indicating we have built a strong

model for exoplanet detection using extracted orbital parameters from radial velocity. The results of this work substantiate the use of orbital parameters extracted from radial velocity analysis and the influence of companion planets on those orbital parameters (via planet to planet interactions) for exoplanet companion identification.

This research has made use of the NASA Exoplanet Archive, which is operated by the California Institute of Technology, under contract with the National Aeronautics and Space Administration under the Exoplanet Exploration Program. This research made use of Astropy,[3] a community-developed core Python package for Astronomy.

References

[1] RL Akeson, X Chen, D Ciardi, M Crane, J Good, M Harbut, E Jackson, SR Kane, AC Laity, S Leifer, et al. The nasa exoplanet archive: data and tools for exoplanet research. *Publications of the Astronomical Society of the Pacific*, 125(930):989, 2013.

[2] Astropy Collaboration, T. P. Robitaille, E. J. Tollerud, P. Greenfield, M. Droettboom, E. Bray, T. Aldcroft, M. Davis, A. Ginsburg, A. M. Price-Whelan, W. E. Kerzendorf, A. Conley, N. Crighton, K. Barbary, D. Muna, H. Ferguson, F. Grollier, M. M. Parikh, P. H. Nair, H. M. Unther, C. Deil, J. Woillez, S. Conseil, R. Kramer, J. E. H. Turner, L. Singer, R. Fox, B. A. Weaver, V. Zabalza, Z. I. Edwards, K. Azalee Bostroem, D. J. Burke, A. R. Casey, S. M. Crawford, N. Dencheva, J. Ely, T. Jenness, K. Labrie, P. L. Lim, F. Pierfederici, A. Pontzen, A. Ptak, B. Refsdal, M. Servillat, and O. Streicher. Astropy: A community Python package for astronomy. , 558:A33, October 2013.

[3] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[4] John D Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90, 2007.

[5] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[6] Michel Mayor and Didier Queloz. A jupiter-mass companion to a solar-type star. *Nature*, 378(6555):355, 1995.

[3] `http://www.astropy.org`

[7] Kyle A. Pearson. A search for multi-planet systems with tess using a bayesian n-body retrieval and machine learning, 2019.

[8] Ilan Reinstein. Xgboost, a top machine learning method on kaggle, explained, 2017.

[9] Christopher J Shallue and Andrew Vanderburg. Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(2):94, 2018.

[10] G. van Rossum. Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, May 1995.

[11] Jacob T VanderPlas. Understanding the lomb–scargle periodogram. *The Astrophysical Journal Supplement Series*, 236(1):16, 2018.

[12] Jason T Wright. Radial velocities as an exoplanet discovery method. *Handbook of Exoplanets*, pages 1–13, 2017.