Proceedings of the International Conference on Industrial Engineering and Operations Management

Publisher: IEOM Society International, USA DOI: 10.46254/NA10.20250066
Published: June 17, 2025

Bridging the Black Box: Data Mining Enhanced Machine Learning Interpretability for Exoplanet Discovery

V. Ann Schrimpsher, Ana Wooley and Anyama Tettey
Department of Industrial & Systems & Engineering Management
University of Alabama in Huntsville
Huntsville, AL, USA
vas0001@uah.edu, acw0047@ush.edu, aht0005@uah.edu

Hieu Pham
Professor, Department of Information Systems, Supply Chains, and Analytics
University of Alabama in Huntsville
Huntsville, AL, USA
htp0004@uah.edu

Sampson Gholston
Chair & Professor, Department of Industrial, Manufacturing, and Systems Engineering
University of Texas at Arlington
Arlington, TX, USA
sampson.gholston@uta.edu

Abstract

Data Mining for Artificial Intelligence (AI): Achieving Transparency Without Sacrificing Performance:
The widely accepted notion that there is an inherent trade-off between accuracy and interpretability in artificial intelligence (AI) models is often misleading. While black-box machine learning (ML) models are commonly viewed as essential for achieving high predictive and classification performance, this paper demonstrates that deterministic data mining (DM) techniques can enhance ML interpretability without compromising accuracy or efficiency. Using the Periodic Motion Detection (PMD) algorithm, we show that deterministic methods can improve transparency in ML decision-making processes without affecting computational time or sacrificing accuracy and precision, particularly in large, noisy real-world datasets. To validate its performance, the PMD process was applied to 4,000 light curves from the Kepler Space Telescope, each containing 15,000 data points, to extract known exoplanet signals. Its performance was compared to that of eleven ML models. After normalizing results for GPU computation, PMD processed a single light curve in just 0.021 seconds—at least an order of magnitude faster than all ML models tested. Additionally, it achieved an accuracy of 93.23%, precision of 98.76%, recall of 87.55%, and specificity of 98.9%, matching or exceeding the performance of the ML models. PMD requires minimal preprocessing and no iterative training, maintaining low design complexity and computational costs. Rather than replacing ML, this study emphasizes how deterministic methods can serve as an efficient interpretability layer for black-box models. By integrating DM techniques like PMD, AI practitioners can gain valuable insights into black box decision processes while preserving the predictive power of machine learning.

Keywords
Machine learning, deterministic programs, artificial intelligence, data mining, black box

1. Introduction

Artificial Intelligence (AI) methods, especially Machine Learning (ML), have proven to be invaluable tools for processing large-scale data. One of the greatest advantages of ML algorithms is their universality—the ability to be applied to a wide range of problems and datasets. These models autonomously build analytical frameworks by continuously learning from data and adapting their structures to optimize performance. Importantly, they do so without requiring explicitly programmed rules, which allows them to process vast amounts of data and identify complex relationships between variables. This capability enables insights to be drawn from systems' activities and interactions, even when the relationships between variables are unclear or previously unknown. However, this remarkable ability to autonomously uncover patterns comes with significant trade-offs. ML models often require long training and retraining cycles, leading to high computational demands (Ray 2019; Wu et al. 2020).

Additionally, their black-box nature—where the decision-making process is opaque—can undermine trust in their outputs, which hinders decision-making.To address these concerns, efforts have been made to increase model transparency through the integration of additional algorithmic layers, such as those used in explainable AI (XAI). While these techniques, like Local Interpretable model-agnostic Explanations (LIME) and Shapley Additive Explanations (SHAP), aim to improve interpretability, they are often met with skepticism. Concerns about increased processing time and computational overhead, as well as potential trade-offs with accuracy and precision, have led to their limited adoption in practice (Rudin 2019; Zhou and Kantarcioglu 2020; Bell et al. 2022; Kamwa et al. 2011). Despite the appeal of XAI techniques, their impact on model performance and efficiency remains a key challenge.

All ML models require high-quality training data to establish optimal weight and bias values. To build an accurate classification model, they must be trained on large, diverse datasets to ensure they can generalize well across different inputs. While this iterative learning process improves model performance over time, it also increases preprocessing and training times (Ray 2019). Additionally, ML models rely on real-world datasets for training, meaning that a portion of the available data is used for model development rather than direct analysis. If the distribution of data points varies significantly across different datasets, the model may require frequent retraining to maintain accuracy, even when analyzing the same underlying relationships and patterns (Wu et al. 2020). Any algorithmic transparency layer, to be effective, must not increase the already intensive training time or require additional training data that could lead to overfitting.

One of the most widely recognized challenges in ML programming is its black-box nature—the inability to fully understand how the model makes decisions. ML models operate with opaque decision-making processes, often requiring external interpretability techniques to identify the factors that influence specific predictions. This lack of transparency can undermine trust in model outputs, making it difficult to validate results or diagnose errors. In some cases, misinterpretations of an ML algorithm's decision process have led to severe consequences (Rudin 2019). Increasing transparency in the black-box process would enhance confidence and trust in a model's predictions by helping users understand how decisions are made (Algolia 2025). Transparent models also foster better accountability, as it becomes easier to identify and correct biases or errors in decision-making, helping ensure fairness and the ethical use of AI (TechTarget 2021).

ML programs are computationally expensive, requiring significant resources to store models and process large-scale datasets. Many ML algorithms rely on high-performance Graphics Processing Unit (GPU) clusters to manage the complexity of their neural network architectures, which contributes to substantial processing costs. The primary computational burden arises from the size and depth of these neural networks, as well as the iterative training required for optimization. Since ML is often applied to large, noisy, and unlabeled datasets, any additional transparency layer must be capable of parsing these datasets without increasing computational overhead, all while maintaining high precision and reliability—particularly in scenarios requiring real-time analysis or large-scale data processing.

Deterministic models are lightweight, rule-based methods used for pattern recognition, dataset search, and decision-making. Unlike machine learning ML models, they do not require training or retraining and operate on well-defined logical frameworks, resulting in a significantly smaller computational footprint. However, they typically require more upfront design and development effort, and once implemented, they are less flexible in adapting to new or evolving data patterns. While they may lack the universality of ML algorithms, deterministic models offer clearer, more understandable processing steps and outcomes, which can enhance confidence in the decision-making process. Once developed, these models do not require iterative training, allowing them to operate efficiently without significantly increasing the processing time of ML models. Furthermore, deterministic algorithms have fully defined mathematical

structures, ensuring complete transparency and enabling quick processing of large datasets, all while maintaining a lower computational cost compared to traditional ML approaches.This paper applies the Periodic Motion Detection (PMD) algorithm, a deterministic data mining (DM) approach developed in previous work (Schrimpsher et al. 2025), for detecting periodic motion in large datasets. Unlike ML algorithms, which require extensive training, the PMD algorithm performs the same task with minimal preprocessing and computational overhead, while also providing transparency in the identification of exoplanet signals.

The goal of this paper is to demonstrate that the PMD algorithm can achieve the same results as ML algorithms without the need for training on the data, while offering equivalent or superior levels of accuracy, precision, specificity, and recall. Additionally, the PMD algorithm performs this task faster—by an order of magnitude—compared to ML models. To showcase this, the PMD algorithm is applied to 4,000 light curves from the Kepler Space Telescope, each containing 15,000 data points, to extract known exoplanet signals. The results are then compared to those of eleven different ML algorithms that have previously been used for the same process. Since the PMD algorithm performs as well as the ML algorithms without increasing preprocessing time or computational cost, it proves to be an ideal candidate for serving as a transparency layer that can be added to ML algorithms to enhance interpretability without sacrificing performance.

In this paper, I will first describe the eleven ML algorithms, providing a brief overview of their operation, and then explain the origin of the light curve dataset. I will detail the preprocessing steps involved for the ML models and compare them to the PMD's approach, demonstrating that the PMD does not significantly increase preprocessing time or introduce any additional risk of overtraining the data. I will also clarify the operation of the PMD algorithm, illustrating how it identifies exoplanet signals and highlights the specific data points that correspond to these signals, thus ensuring complete transparency in the output determination.

Furthermore, I will outline the methods for calculating computational expense, accuracy, precision, specificity, and recall, comparing the performance metrics of the PMD algorithm with those of the eleven ML models. The results will show that the PMD algorithm not only matches but in many cases exceeds the performance of the ML models across all metrics, while also demonstrating a computational speed advantage—performing at least an order of magnitude faster. This comparison will establish that the PMD algorithm can effectively perform the same tasks as ML models, providing greater transparency without sacrificing performance or increasing computational or preprocessing time.

1.1 Objectives

The objectives of this research are as follows:

- Demonstrate the potential of deterministic algorithms to enhance transparency by showing that deterministic models, such as the PMD algorithm, can provide clear and understandable decision-making steps, helping to clarify the output of ML models.
- Assess the performance of deterministic algorithms alongside ML models to show that deterministic algorithms can match or exceed ML results in terms of accuracy, precision, specificity, and recall.
- Evaluate the computational efficiency of the combined approach by demonstrating that deterministic algorithms can be paired with ML models without increasing preprocessing time or computational cost, making them an ideal transparency layer.
- Propose the benefits of integrating deterministic algorithms with ML models, arguing that pairing deterministic algorithms with ML models can offer the best of both worlds—enhanced interpretability and transparency without sacrificing performance or computational efficiency.

2. Machine Learning (ML) Model Review

ML is a subset of AI that focuses on developing systems capable of learning from data. ML techniques enable software applications to improve performance over time by identifying patterns, relationships, and structures within datasets. These models use historical data to make predictions, classify information, cluster data points, and reduce dimensionality. Given the diverse nature of datasets—varying in structure, type, and size—numerous ML algorithms have been developed to address different challenges.

This study evaluates the performance of seven ML algorithms in comparison to a deterministic data mining approach. The following section provides a brief overview of each algorithm, including its origins, strengths, and limitations.

The sigmoid activation function was first introduced in neural networks (NNs) by Norman Wiener in 1942 (Wiener 1965). It enables non-linearity in models, allowing neural networks to learn complex decision boundaries by converting linear outputs into probabilities. However, its major limitation is that it only produces outputs between 0 and 1, making it unsuitable for problems requiring multiple outputs beyond this range. Additionally, sigmoid suffers from the vanishing gradient problem, which can slow or halt learning during deep network training.

The Naïve Bayes (NB) classifier emerged in the early 1970s and was first applied in image recognition of landscapes in 1973 (Haralick et al. 1973). Today, it is widely used in text classification and spam detection. NB is a simple and fast algorithm that performs well on both small and large datasets. As a generative model, NB estimates the underlying distribution of data to classify new instances, distinguishing it from discriminative models like Sigmoid, which learn a direct boundary between classes. Unlike Sigmoid, NB allows for multiple probability outputs and does not suffer from the vanishing gradient problem. However, NB assumes feature independence, which is rarely true in real-world datasets, limiting its accuracy in complex applications.

Introduced in 1975, the rectified linear unit (ReLU) activation function addressed the vanishing gradient problem by replacing all negative input values with zero, ensuring efficient gradient propagation. Unlike Naïve Bayes, ReLU does not assume feature independence. However, due to dead neuron issues, it was initially considered less accurate than NB and Sigmoid. In 2010, Nair & Hinton refined the function in their work on Restricted Boltzmann Machines (Nair and Hinton 2010), leading to its widespread adoption. Today, ReLU is one of the most commonly used activation functions in neural networks.

In 1989, the Least Squares (LS) method (Mirzai et al. 1989) was introduced into ML as a form of regression analysis used to determine the best-fit line for a dataset. It is commonly applied in time-series analysis to model trends. However, LS is highly sensitive to outliers, meaning deviations in the dataset can significantly impact the regression results. Additionally, LS has poor extrapolation properties, making it unreliable when predicting values outside the observed range.

Support Vector Machines (SVMs) are supervised learning models used for classification and regression problems. They were introduced in 1992 by Vladimir Vapnik and Corinna Cortes (Cortes and Vapnik 1995). SVMs perform exceptionally well in high-dimensional spaces and are particularly effective when the number of features exceeds the number of samples. Additionally, SVMs are memory-efficient and can separate data using hyperplanes with strong generalization capabilities. However, SVMs struggle with large datasets, are prone to errors when handling noisy data, and require extensive training to optimize hyperparameters. Furthermore, their decision-making process is difficult to interpret, making them a less transparent choice compared to other models.

The Random Forest (RF) algorithm was trademarked by Leo Breiman and Adele Cutler in 1995. RF is an ensemble learning method that combines multiple decision trees to improve predictive performance. It is versatile, supporting both classification and regression tasks, and is widely used due to its simplicity, robustness, and ability to handle categorical and continuous data (Svetnik et al. 2003). However, RF has notable drawbacks, including high computational cost, long training times, and lack of interpretability, as it struggles to determine the significance of individual variables. Despite these limitations, RF remains one of the most frequently used ML models across various domains.

Developed in the 1990s, Convolutional Neural Networks (CNNs) became a breakthrough in deep learning, particularly in image classification and object detection (Kang et al. 2014). CNNs automate feature extraction without requiring manual intervention, making them ideal for recognizing patterns in visual data. Compared to RF, CNNs demonstrate superior accuracy in image processing tasks. However, they require high computational resources, large, labeled datasets, and extensive memory, making them less effective for sequential data analysis.

ML algorithms have evolved significantly over the years, with continuous advancements in their ability to process and analyze large datasets. From early techniques like the sigmoid activation function to more complex models like CNNs, the field has made remarkable progress. Despite these advancements, challenges remain, including issues related to interpretability, computational efficiency, and the potential for overfitting. As a result, there is still substantial room for growth and improvement in ML algorithms, with ongoing research focused on overcoming these limitations.

3. Methodology for Comparing Machine Learning and Data Mining

To demonstrate that the PMD process avoids the costs of training, retraining, black-box decision-making, and excessive computational time, we first explain the preprocessing steps for handling the large datasets used in this experiment. These steps will show that the PMD model's preliminary processes are far less time-consuming than the training phase of machine learning models. Additionally, an interpretation of the underlying PMD algorithm will reveal that its outputs are transparent, not black-box results. We then describe the equations used to calculate accuracy, precision, specificity, and recall metrics for both the PMD process and the eleven ML models, assessing their effectiveness in selecting exoplanet signatures from large, noisy, unlabeled datasets. Before the comparative analysis begins, we provide a description of the dataset used in the experiment.

3.1 Light Curve Dataset

A large data set containing a lot of real-world noise was needed to thoroughly test the algorithms; so that computational time could be fully measured, and accuracy rates would be comprehensively gauged. The chosen data sets consisted of 10,000 light curves containing 15,000 data points each from the Kepler space telescope taken during Q1 – Q17 part of its mission. A light curve is a set of intensity values, or star brightness measurements, taken at a predetermined time interval from an observed star or set of stars. This created a time-series graph where time was the independent or "x" variable, and Intensity values were the dependent or "y" value. The goal of the algorithms was to detect the passing of a exoplanet, or a transit, in front of the exoplanet's parent star. The algorithms should be able to determine when a transit occurs due to the drop in the star's intensity value. These light curves contain a lot of inherent random noise due to each star's unique fluctuations and the oscillations intrinsic to the instrumentation.

All naturally occurring transits were removed from the 10,000 light curves. Then 140 known transits from the National Aeronautics and Space Administration (NASA) database were injected into 5,000 light curves. The total number of transits within the data set was 560. This ensured that the programmers would know exactly where and when each transit took place. It also removed any doubt about the correctness of the calculated false-positive, false-negative, true-positive, and true-negative values. The development of these light curves was done by Jara-Maldonado et al. (2020).

3.2 Preprocessing Steps: Training and Retraining

Data preprocessing usually involves cleaning and transforming, feature extraction, and normalization of the data. These steps are necessary to better accomplish the classification of data (Subasi 2020). These preprocessing steps are required for any algorithms using data sets. All AI algorithms, like ML, require an additional preprocessing step called training.

Often training sets must be run multiple times to get a working ML model. The first time you train the model the accuracy is extremely low. Remember ML models learn as they go, so they are required to run through multiple data sets with enough data variability in order to generate an accurate model. You can however, over train a model, which leads to over-fitting. Over-fitting is when a model has achieved 100% or near 100% accuracy. The problem with over-fitting is the accuracy of the model will significantly decrease when evaluating new data sets. It amounts to the model memorizing the training data set instead of learning how to categorize the data within the set.

Many times, an ML model will have to be retrained. This occurs when the data variability is much higher or much lower in a new data set. Even if the new data set is of the same nature as the original training data sets and the model is looking for the same patterns and relationships. A large enough change in data variability will make the patterns and relationships unrecognizable to the ML model and the training process will have to begin again. This situation arose for the programmers training the ML models on the Kepler Light Curve data.

A star's brightness fluctuates naturally. Different stars' brightness fluctuates at different levels. Some stars are quieter, while other stars are more erratic. The ML models for this experiment were trained on stars that had medium variability. If light curves from more quiet or more erratic stars were used, the models would have had to be retrained (Jara-Maldonado et al. 2020).

The PMD process, which was used to analyze the light curves in this experiment, is a variation of a Data Mining (DM) model. Since it is not a form of AI, no training is required. There were, however, preprocessing steps that had to occur that altered the data in the light curve data set to make it feasible for analysis. To provide an overview of the PMD algorithm, as detailed in Schrimpsher et al. (2025), we begin by describing the data preprocessing steps that enabled the light curve data to be mined and the threshold parameter to be calculated for the test phase. The first step in the process was interpolation. Since the light curve data were sporadic, new data points were predicted using Equation (1) to generate evenly spaced intensity values. This step helped accentuate trends in the data, making them more distinct and easier to analyze.

$$f(c) = [f(a)]^{(b-c)/(b-a)} * [f(b)]^{(c-a)/(b-a)} \quad (1)$$

Where, a = time at initial intensity value
b = time at final intensity value
c = desired time, this is the time you want an intensity value for
$f(a)$ = intensity at point a
$f(b)$ = intensity at point b
$f(c)$ = interpolated intensity at point c.

Following interpolation, the normalization step was applied. This involved setting the mean of the light curve to zero and adjusting all intensity values to align with this new mean. The threshold parameter was then calculated to be 3 standard deviations (SD) from the mean. The 3-sigma threshold was chosen because it strikes a balance between the two primary types of statistical errors, providing an effective method for identifying significant events while minimizing false positives and negatives (Montgomery 2013).

The preprocessing steps performed by Jara-Maldonado et al. (2020) for the ML models are similar to those of the PMD with some caveats. First, linear interpolation was applied to the light curves where data gaps existed. Next, each light curve was folded so that all transits within a light curve overlapped. This allowed the data to be normalized and given values between 0 and 1. After that, the data was grouped into bins that consisted of the mean of all the data points, so that the data set size was reduced. In the end, there were 2048 bins. Each model was enhanced with a Discrete Wavelet Transform (DWT) which decomposed the number of data points. Some of the ML models' neural nets were altered to have different numbers of hidden layers and different numbers of neurons within each hidden layer. Each modified ML model was used in the comparison analysis.

Once the preprocessing steps were complete for the ML models, 60% of the preprocessed light curves were used to train the ML models. Each ML model was trained for 200 iterations for each of the 6,000 light curves assigned for training purposes. The 4,000 remaining light curves were used for testing

3.3 PMD Algorithm Overview

Understanding how an algorithm works is fundamental to ensuring transparency in a program's decision-making process, as it clarifies how results are obtained. Unlike machine learning models, the core function of the PMD algorithm is fully explainable and transparent.

The PMD algorithm used in this study was developed by Schrimpsher et al. (2025), with some modifications made to adapt it for the light curve dataset discussed in this paper. The process begins with preprocessing the light curves, where the data is normalized to standardize intensity measurements and calculate the threshold parameter. The core of the PMD algorithm is a partial fast Fourier transform (pFFT), designed to convert the data from Cartesian coordinates to polar form, enabling more efficient detection of periodic events. Once the data is transformed into polar form, it is divided into 15° segments, and for each segment, the mean intensity value is calculated. The algorithm computes the difference between the lowest intensity value within the segment and the calculated mean. If this difference exceeds the predefined threshold parameter (calculated during preprocessing), the segment is marked as a potential transit event. The threshold is designed to identify significant intensity fluctuations while minimizing noise.

Once the potential transit events are identified, the corresponding angle offsets are recorded. All segments are then folded based on their angle offsets. For example, if a potential transit event occurs at 35 degrees, all data points at 35 degrees are overlapped. This folding process aligns potential transit events across the dataset, improving the accuracy of transit detection. If multiple transit events are detected within the same time interval, they are labeled as true positives; if only one event is detected, it is classified as a true negative. The angle offsets are then used to determine the specific location along the timeline where the true-positive exoplanet signatures occur. First, the data is converted from polar coordinates back into Cartesian form. From there, the transit events can be traced along the timeline to pinpoint when an exoplanet signature was detected.

3.4 Experimental Setup and Performance Comparison

ML algorithms are known for their complex architectures, which not only demand significant development time but also result in longer runtimes for processing. In this study, we compared the runtime performance of eleven different ML algorithms from Jara-Maldonado et al. (2020) against the PMD process, using a dataset of 4,000 light curves. Each light curve contains 15,000 data points, representing time-series measurements of stellar brightness. The hypothesis of this comparison is that, due to the deterministic nature of the PMD process, it will be significantly faster than the ML algorithms while maintaining equivalent levels of accuracy, precision, specificity, and recall.

Table 1 lists the eleven ML algorithms included in the study, which are described in detail by Jara-Maldonado et al. (2020). It is important to note that some algorithm names, such as Sigmoid and ReLU, appear multiple times. This is because some models are variations of the same algorithm, with differences in the number of hidden layers and neurons. The numbers in parentheses next to the model names indicate the number of hidden layers, with the values separated by commas representing the number of neurons in each layer.

As mentioned earlier, the metrics accuracy, precision, specificity, and recall were used to evaluate the performance of all algorithms, confirming that the PMD method can perform at the same level as ML algorithms. Accuracy measures the percentage of times a model correctly predicts the outcome. Precision calculates the proportion of positive predictions that are correct. Specificity determines the percentage of negative predictions that are correctly classified. Recall measures the percentage of true positives correctly identified by the model. The formulas for computing these metrics are provided in Equations (2), (3), (4), and (5) below.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

Where, TP = true-positive
TN = true-negative
FP = false-positive
FN = false-negative

All ML models were executed on a computer with an Intel Core i7-7700 High Quality (HQ) CPU, 16.0 GB of RAM, Windows 10 operative system of 64 bits, and a NVIDIA GeForce 1060 graphics card by Jara-Maldonado et al. (2020). The PMD model used an Intel Core i5-9600 K CPU, 3.70 GHz × 6, 32.0 GB AMD Radeon RX 550 series, Linux (Pop!_OS 22.04 Long-Term Support (LTS)) 64-bit by the authors of this paper. Notice the computer used to run the PMD process did not use a graphic card like the computer used to run the ML models. The original runtime of the PMD process was 31 seconds. The runtime value for the PMD model in Table 1 has been normalized to the GPU processing speed to display a fair assessment between all the model runtimes.

4. Data Collection

The light curve data used in this study was originally sourced from the Kepler Space Telescope mission, publicly available through NASA's Mikulski Archive for Space Telescopes (MAST). This data was modified for the experiment discussed in Jara-Maldonado et al. (2020) to test the eleven ML algorithms presented in this paper. We obtained the modified dataset directly from Miguel Jara-Maldonado (miguel.jaramo@udlap.mx) and Vicente Alarcon-Aquino (vicente.alarcon@udlap.mx). The results from their experiment, detailed in Table 1, were used to compare the

performance metrics and computational runtimes of the PMD algorithm against those of the ML algorithms. This dataset was initially presented in Jara-Maldonado et al. (2020).

Table 1. Dataset experimental results, averaged from 100 executions for each model

Model	Accuracy (%)	Precision (%)	Specificity (%)	Recall (%)	Time (sec)
Sigmoid (5,2)	94.23	92.38	91.69	96.72	1.76
ReLU (5,2)	95.12	92.94	92.27	97.93	2.54
Sigmoid (1024)	92.65	89.72	88.13	97.1	46.94
ReLU (1024)	97.47	96.32	96.17	98.75	72.93
Sigmoid (64,32,8,1)	50.68	39.66	23.51	77.98	0.43
ReLU (64,32,8,1)	97.42	96.66	96.57	98.38	7.16
LS	49.62	0	99.89	0	0.35
Random Forest	98.08	97.49	97.41	98.73	1.16
CNN	99.13	99.16	99.13	99.09	22.93
Naïve Bayes	95.93	93.37	92.85	98.99	0.35
SVM	94.96	90.93	89.84	99.99	1.52

5. Results and Discussion

5.1 Numerical Results

Once the PMD runtime data was normalized to account for differences in hardware configuration, the PMD algorithm was able to analyze a single light curve in just 0.021 seconds (see Table 2). This is more than an order of magnitude faster than the runtimes of the ML models, which are presented in Table 1. To assess the statistical significance of these runtime differences, a paired t-test was conducted. The results, shown in Table 3 under the "Time" column, revealed that all p-values were significantly below 0.05, indicating that PMD's runtime was considerably faster than all the ML models tested.

Table 2. PMD model metric values for accuracy, precision, specificity, recall, and runtime.

Model	Accuracy (%)	Precision (%)	Specificity (%)	Recall (%)	Time (sec)
PMD	93.23	98.76	87.55	98.9	0.021

Table 3. The p-values for the time column were obtained from the paired t-test. All other metric's p-values were obtained from the one-sided equivalence test.

Model	Accuracy	Precision	Specificity	Recall	Time
Sigmoid (5,2)	greater	greater	greater	< 0.05	< 0.05
ReLU (5,2)	greater	< 0.05	greater	greater	< 0.05
Sigmoid (1024)	greater	< 0.05	greater	greater	< 0.05
ReLU (1024)	greater	greater	greater	greater	< 0.05
Sigmoid (64,32,8,1)	greater	greater	greater	greater	< 0.05
ReLU (64,32,8,1)	greater	greater	greater	greater	< 0.05
LS	greater	greater	< 0.05	greater	< 0.05
Random Forest	< 0.05	greater	greater	< 0.05	< 0.05
CNN	< 0.05	greater	greater	< 0.05	< 0.05
Naïve Bayes	greater	greater	greater	greater	< 0.05
SVM	greater	greater	greater	greater	< 0.05

The PMD metric values for accuracy, precision, specificity, and recall are shown in Table 2. When comparing these metrics to the results of the other ML models (see Table 1),

PMD outperformed most of the ML models in these areas. For the metrics where PMD did not show better performance, a one-sided equivalence test was conducted to verify that the metric levels between PMD and the ML models were equivalent. In these cases, the null hypothesis, known as the "inferiority" hypothesis, tests whether the new mean is lower than the lower bound of the equivalence range around the original mean (Pardo, 2014). Rejecting the null hypothesis indicates that the two means are equivalent, with the desired outcome being a p-value less than 0.05. The statistical test showed that all inferior means for all metrics were equivalent. The results of the statistical testing are summarized in Table 3.

The results from the PMD algorithm demonstrate impressive performance across multiple metrics, including accuracy, precision, specificity, and recall, as seen in Table 2. In comparison to the ML models, PMD outperformed most of them in these key areas, highlighting its potential as a highly effective tool for identifying exoplanet signals. For the metrics where PMD did not show superior performance, the one-sided equivalence test confirmed that PMD's results were statistically equivalent to those of the ML models. This suggests that, even in cases where PMD did not outperform the ML algorithms, it still delivered comparable results. The equivalence testing supports the conclusion that PMD can achieve comparable or superior performance to traditional machine learning models without sacrificing accuracy. Additionally, the significant difference in runtime, with PMD being over an order of magnitude faster, further underscores its potential as an efficient alternative for processing large datasets. These findings suggest that PMD not only offers transparency but also has the potential to be integrated with or used alongside ML models, providing a valuable balance between interpretability and computational efficiency. The results of these comparisons, combined with the equivalence testing, position PMD as a strong candidate for future applications in both standalone and hybrid systems.

5.3 Proposed Improvements

There were two major issues with using the PMD model as a classification model that will need to be addressed in the future. Recall that the PMD process requires the polar coordinate time-series data to be split into segments every 15°. On occasion in the data set, a transit would appear at or near the 15° mark. Splitting the data at a transit occurrence causes the model to miss the transit and give a false-negative reading. The second issue had to do with a star's variability. Sometimes a star's intensity levels would be extremely high for three or four data points. Since the mean of each segment of data is calculated independently, that mean value for that segment would be unusually high. This caused the difference between that particular mean and the lowest intensity value associated with that segment to surpass the predetermined threshold value and a false-positive response would be given.

6. Conclusion

ML algorithms are highly effective in classifying complex data, yet their black-box nature can diminish trust in their outputs, hindering decision-making processes. To address this, various XAI techniques have been integrated into ML models to enhance transparency and foster trust. However, skepticism remains about these methods, primarily due to concerns that they may reduce model performance, increase preprocessing and computational time, or even introduce overfitting, especially since many of these interpretability layers rely on the same datasets as the ML models themselves for training.

In this paper, we demonstrate that the PMD algorithm is a powerful tool for enhancing both the transparency and efficiency of ML models. By comparing the PMD algorithm with eleven different ML models, we show that PMD can achieve comparable, if not superior, performance, with an accuracy of 93.23%, precision of 98.76%, recall of 87.55%, and specificity of 98.9%. Moreover, PMD processes a single light curve at least an order of magnitude faster than the ML models tested. These results suggest that PMD can serve as an ideal XAI layer within an ML algorithm, improving the interpretability of ML algorithms without compromising performance, computational efficiency, or increasing preprocessing time.

While the PMD algorithm demonstrates impressive results, it is important to acknowledge certain limitations, particularly in its application as a classification model. Challenges related to data segmentation at transit occurrences and star variability were identified, causing false-negative and false-positive responses, respectively. These issues must be addressed in future work to enhance the robustness of the algorithm. Nonetheless, the PMD algorithm's ability to operate without training, combined with its efficiency and transparency, underscores its potential as a valuable complement to ML algorithms.

This study supports the idea that deterministic algorithms, such as PMD, can enhance ML models by providing greater interpretability without sacrificing accuracy or computational efficiency. While the models were tested separately, the

findings suggest that a hybrid system incorporating both approaches could offer the best of both worlds—leveraging the high predictive power of ML and the clarity of deterministic algorithms. This presents a promising avenue for future research.

Acknowledgements

The authors would like to thank Miguel Jara-Maldonado and Vicente Alarcon-Aquino for letting our team use the Kepler mission light curve data set that they acquired from the MAST data archive at the Space Telescope Science Institute (STScI). Funding for the Kepler mission is provided by the NASA Science Mission Directorate. STScI is operated by the Association of Universities for Research in Astronomy, Inc., under NASA contract NAS 526555.

References

Algolia, (n.d.), Building trust with AI transparency. *Algolia*, Available: https://www.algolia.com/blog/ai/building-trust-with-ai-transparency, Retrieved February 25, 2025.

Bell, A., Solano-Kamaiko, I., Nov, O. and Stoyanovich, J., It's just not that simple: an empirical study of the accuracy-explainability trade-off in machine learning for public policy. *Proceedings of the 2022 ACM conference on fairness, accountability, and transparency*, pp. 248-266, June 2022.

Cortes, C. and Vladimir, V., Support-vector networks, *Machine learning,* vol. 20, pp. 273-297, 1995.

Haralick, R., Shanmugam, K. and Dinstein, I., Textural feastures for image classification*, IEEE transactions on Systems, Man, and Cybernetics*, No. 6, Mov 1973.

Jara-Maldonado, M., Alarcon-Aquino, V., Rosas-Romero, R., Starostenko. O. and Ramirez-Cortes, J., Transiting exoplanet discovery using maching learning techniques: a survey, *Earth Science Informatics*, vol. 13 pp. 573-600, Jun 2020.

Kamwa, I., Samantaray, S. R. and Joós, G., On the accuracy versus transparency trade-off of data-mining models for fast-response PMU-based catastrophe predictors. *IEEE Transactions on Smart Grid*, vol. *3*(1), pp. 152-161, 2011.

Kang, L., Kumar, J., Ye, P., Li, Y. and Doermann, D., Convolutional neural networks for document image classification, *22nd International Conference on Pattern Recognition,* Stockholm, Sweden, pp. 3168-3172, 2014.

Mirzai, A., Cowan, C. and Crawford, T., Intelligent alignment of waveguide filters using a machine learning approach, *IEEE transactions on Microwave Theory and Techniques*, vol. 37(1), pp.166-173, 1989.

Montgomery, D. *Introduction to Statistical Quality Control 7th Edition*, pp. 245, 2013.

Nair V. and Hinton, G., Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th International Conference on Machine Learning*, pp. 807-814, Jun 2010.

Pardo, S., Equivalence and noninferiority Tests for Quality, Manufacturing, and Test Enginers, *CRC Press*, Boca Raton: 2014.

Ray, S., A quick review of machine learning algorithms, *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, Faridabad, India, pp. 35-39, 2019.

Rudin, C., Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, *Nat Mach Intell*, vol. 1(5), pp. 206-215, May 2019.

Schrimpsher, V., Gholston, S., Pham, H., Wooley, A. and Tettey, A., Tracking the Unseen: Autonomous Multi-Object Motion Analysis in Complex Environments, *Proceeding of the 10th North American Conference on industrial Engineering and Operations Management (IEOM),* Orlando, Florida, in press.

Subasi, A., Practical Machine Learning for Data Analysis Using Python, *Academic Press*, pp. 27-89, 2020.

Svetnik, V., Liaw, A., Tong, C., Culberson, J., Sheridan, R. and Feuston, B., Random forest: a classification and regression tool for compound classification and QSAR modeling, *J. Chem. Inf. Comput. Sci,* vol. 43(6), pp. 1947-1958, Nov 2003.

Wiener, N., Cybernetics, or the Control and Communication in the Animal and the Machine*, The MIT Press*, vol. 25, 1965.

Wu, Y., Dobriban, E. and Davidson, S., DeltaGrad: rapid retraining of machine learning models, *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, June 2020.

Zhou, Y., and Kantarcioglu, M., On transparency of machine learning models: A position paper, *AI for Social Good Workshop,* pp. 1-5, 2020.

TechTarget, AI transparency: What is it and why do we need it? *SearchCIO*, Available: https://www.techtarget.com/searchcio/tip/AI-transparency-What-is-it-and-why-do-we-need-it, December 1, 2021.

Biographies

V. Ann Schrimpsher is a doctoral student in the Department of Industrial and Systems Engineering and Engineering Management (ISEEM) at the University of Alabama in Huntsville (UAH). Holding both an MBA and a B.S. in Electrical Engineering from UAH. In their career, Ann has collaborated with the Department of Defense on real-time embedded systems, integrating satellite systems, ground-based radars, sea-based radars, missile guidance systems, and command and control centers. They served as the Technical Lead within the systems engineering department at Teledyne Brown Engineering and Boeing, overseeing key projects in this domain. As a Senior Software Engineer at Motorola, they contributed to the development of the Iridium satellite system, the first global satellite communications network. They also have experience as a Product Manager in the Testing Department at Honeywell, where they led the design of electrical systems for the company's private jet fleet. Their research interests span supply chain management, multi-modal systems, and machine learning, with a focus on advancing the integration and efficiency of complex systems.

Dr. Hieu Pham is an assistant professor of analytics in the Department of Information Systems, Supply Chain, and Analytics at the University of Alabama in Huntsville. His research interests include operations research, machine learning, and data analytics.

Dr. Sampson E. Gholston is the Chair of the Industrial, Manufacturing, and Systems Engineering Department at the University of Texas at Arlington (UTA). His research has been in applied statistics, quality engineering, and organizational performance. Prior to joining UTA, Dr. Gholston served as the Chair of the Industrial and Systems Engineering and Engineering Management at The University of Alabama in Huntsville. He worked at the GM Saturn Corporation as a Supplier Quality Engineer. Dr. Gholston has also consulted and contracted for Delphi Automotive, Huntsville Hospital, NASA Marshall Space Flight Center, General Motors, and other organizations.

Dr. Ana Wooley is an Assistant Professor in the Department of Industrial & Systems Engineering and Engineering Management (ISEEM) at The University of Alabama in Huntsville. She completed her PhD and Master's degree in Industrial and Systems Engineering at Auburn University. Her current research focuses on digital twin development, smart manufacturing and simulation of manufacturing systems.

Anyama Tettey holds a BSc in electrical engineering and master's degree in industrial and systems from Kwame Nkrumah University of Science and Technology, Ghana and North Carolina A&T State University respectively. She is currently pursuing her doctorate degree in Industrial and Systems Engineering and Engineering Management at the University of Alabama in Huntsville. Her research interests include the application of quality engineering principles and machine learning algorithms for the reliable performance of Large-Scale Complex Engineered Service Systems. She is a Senior member of the institute of Electrical and Electronics Engineers.