Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **527,** 6439–6454 (2024)
Advance Access publication 2023 November 22

https://doi.org/10.1093/mnras/stad3603

Asteroids co-orbital motion classification based on Machine Learning

Giulia Ciacci,[1] Andrea Barucci ,[1]★ Sara Di Ruzza[2] and Elisa Maria Alessi[3]

[1]*IFAC-CNR, Istituto di Fisica Applicata 'Nello Carrara', Consiglio Nazionale delle Ricerche, via Madonna del Piano 10, I-50019 Sesto Fiorentino (FI), Italy*
[2]*Dipartimento di Matematica e Informatica, Università di Palermo, Via Archirafi 34, I-90123 Palermo, Italy*
[3]*IMATI-CNR, Istituto di Matematica Applicata e Tecnologie informatiche 'E. Magenes', Consiglio Nazionale delle Ricerche, Via Alfonso Corti 12, I-20133 Milano, Italy*

Accepted 2023 November 15. Received 2023 October 25; in original form 2023 September 15

ABSTRACT

In this work, we explore how to classify asteroids in co-orbital motion with a given planet using Machine Learning. We consider four different kinds of motion in mean motion resonance with the planet, nominally *Tadpole* at L_4 and L_5, *Horseshoe* and *Quasi-Satellite*, building three data sets defined as Real (taking the ephemerides of real asteroids from the JPL Horizons system), Ideal and Perturbed (both simulated, obtained by propagating initial conditions considering two different dynamical systems) for training and testing the Machine Learning algorithms in different conditions. The time series of the variable θ (angle related to the resonance) are studied with a data analysis pipeline defined *ad hoc* for the problem and composed by: data creation and annotation, time series features extraction thanks to the TSFRESH package (potentially followed by selection and standardization) and the application of Machine Learning algorithms for Dimensionality Reduction and Classification. Such approach, based on features extracted from the time series, allows to work with a smaller number of data with respect to Deep Learning algorithms, also allowing to define a ranking of the importance of the features. Physical interpretability of the features is another key point of this approach. In addition, we introduce the SHapley Additive exPlanations for Explainability technique. Different training and test sets are used, in order to understand the power and the limits of our approach. The results show how the algorithms are able to identify and classify correctly the time series, with a high degree of performance.

Key words: methods: numerical – celestial mechanics – minor planets, asteroids: general.

1 INTRODUCTION

In the last decades, the use of Artificial Intelligence (AI) for data analysis has significantly increased in scientific applications, in particular thanks to its sub-field known as Machine Learning (ML), where an algorithm is said to improve its performance on a specific task by experience (e.g. Hastie et al. 2009b; Jordan & Mitchell 2015). More recently, many authors started to use such methods in astronomy and Solar system science (e.g. Ball & Brunner 2010; Ivezić et al. 2014). Although well-known and broadly applied in several contexts, we recall here the general concepts of AI and ML, for the sake of completeness. With AI we mean methods by which a computer makes decisions or discoveries that would usually require human intelligence, while with ML we mean automated processes that learn by examples in order to classify, predict, discover or generate new data. Part of ML is the class of algorithms known as *Deep Learning* (DL) which is based on artificial neural networks (e.g. LeCun, Bengio & Hinton 2015; Goodfellow, Bengio & Courville 2016). ML and DL are the key of the success of AI nowadays. There are three classes of ML algorithms (see e.g. Hastie, Tibshirani & Friedman 2009a for more details): *supervised learning*, where a labelled data set is used to help to train and tune the algorithm, with the goal to create a map that links inputs to outputs; *unsupervised learning*, where no labels are provided and the goal is to discover hidden patterns allowing the data to speak for itself; *reinforcement learning*, where an agent learns by interacting with an environment and modifying its behaviour to maximize its reward. It is important to keep in mind that this line between classes can occasionally become hazy and fluid because numerous applications frequently combine them in inventive and unique ways (e.g. self-supervised learning, see Liu et al. 2021).

These approaches are firmly established in astronomy and an important survey of the state of art can be found in Fluke & Jacobs (2020), who analyse the published articles in the last years. They highlight applications in many sub-fields of astronomy where ML could be used for several activities, as classification, regression, clustering, forecasting, generation of data, discovering, development of new scientific insights. Fluke & Jacobs (2020) also classify the different fields of astronomy where ML is used as 'emerging', 'progressing', and 'established', depending on the progress of its use.

The first approach in astronomy to Principal Component Analysis (PCA), an algorithm devoted to Dimensionality Reduction, which is nowadays a standard technique, was introduced in the 1980s for morphological classification of spiral galaxies (e.g. Whitmore 1984), in the 1990s for quasar detection (e.g. Francis et al. 1992) and spectral classification (e.g. Singh, Gulati & Gupta 1998), while more recent

★ E-mail: a.barucci@ifac.cnr.it

applications with ML have been done for discovering extra-solar planets (e.g. Shallue & Vanderburg 2017; Pearson, Palafox & Griffith 2018), for studying gravitationally lensed systems (e.g. Lanusse et al. 2017; Pourrahmani, Nayyeri & Cooray 2018; Jacobs et al. 2019) and for discovering and classifying transient objects (e.g. Connor & van Leeuwen 2018; Farah et al. 2018). For a complete and detailed bibliography about all the ML applications in the astronomical fields we suggest a careful reading of Fluke & Jacobs (2020).

The analysis of motion of the Solar system bodies is considered one 'progressing' field of application of ML. Several authors in the last years studied problems related to Solar system objects as, for example, applications to TransNeptunian objects (e.g. Chen et al. 2018), or detection and classification of asteroids through taxonomies of spectrophotometry, as studied in Erasmus et al. (2017, 2018).

One 'emerging' field concerns asteroid dynamics (e.g. Carruba et al. 2022). Indeed, the numerical propagation of asteroids' orbits, based on continuous improved information, implies a large volume of data, that requires fast and novel methods to be analysed. For example, in Smirnov & Markov (2017), the authors use ML methods to identify three-body mean motion resonance asteroids in the main belt without requiring numerical integration. They use proper elements which are quasi-integral of motion that are stable for a long time (e.g. Knezevic & Milani 1994; Knezevic, Lemaître & Milani 2002), and use four different supervised ML methods as reported in Hastie, Tibshirani & Friedman (2009a). The authors compare their results with the ones of the previous paper by Smirnov & Shevchenko (2013) remarking that, with the new approach, the identification of the objects trapped in mean motion resonance is very good and the procedure requires few seconds, while the numerical integration requires days and weeks. Very recently, Smirnov (2023) provides a new open-source package for identifying objects trapped in mean motion resonances (MMR). The main objective they have is to distinguish resonant and non-resonant orbits, but they do not aim at distinguishing different classes of 1:1 MMR, like we will do here.

Other new works comparing results from ML algorithms with previous known asteroid classifications are, for example, Smullen & Volk (2020), where the authors classify objects of the Kuiper belt into four classes based on their dynamics Carruba, Aljbaae & Lucchini (2019), where hierarchical clustering algorithms for supervised learning are applied to identify 6 new families and 13 new clustering of asteroids (Carruba et al. 2020), where ML classification algorithms are used to identify new families of asteroids based on the orbital distribution in the parameters [a, e, $\sin(i)$, where a, e, i are, respectively, the semimajor axis, the eccentricity, and the inclination of the asteroid orbit] of previous known family objects.

Some other very interesting and recent works explore the use of ML to classify regular or chaotic motions. For example, Kamath (2022) studies and classifies orbits in Poincaré maps: the major challenge of this problem is solved by creating high-quality training sets with few mislabelled orbits and converting the coordinates of the points into features that are discriminating, despite the apparent similarities between orbits of different classes. Celletti et al. (2022) use DL methods, such as convolutional neural networks (CNNs), to show how it is possible to classify different types of motion, starting from time series, without any prior knowledge of the dynamics. Indeed, the identification of a motion usually requires a knowledge and the solution of the differential equations governing the dynamical system. Instead using CNNs trained on one dynamical model, the type of motion could be predicted, for example, from observational data.

All these examples show how ML algorithms are increasingly used in astronomy, as well as in dynamical systems and in particular in celestial mechanics.

The aim of the study is to classify the co-orbital behaviour that can be described within the planar approximation of the Circular Restricted Three-Body Problem (CR3BP). Leveraging on the recent work (Di Ruzza, Pousse & Alessi 2023), we focus on asteroids that are in 1:1 MMR with a given planet of the Solar system. The data considered are ephemerides of real asteroids that are catalogued by the Minor Planet Center or different works (e.g. Mikkola et al. 2004; Kinoshita & Nakai 2007; Wajer 2010; Christou & Asher 2011; Ćuk et al. 2012; De la Fuente Marcos & De la Fuente Marcos 2012; Wajer & Kròlikowska 2012; De la Fuente Marcos & De la Fuente Marcos 2014; Qi & Qiao 2022) in the same way as *Tadpole*, *Horseshoe*, or *Quasi-Satellite*, and ephemerides created *ad hoc* by propagation of the CR3BP equations of motion and equations of motion corresponding to a more complex dynamical model, starting from well-defined initial conditions. We apply ML methods to classify, through specific features, the time series of a specific angular variable obtained in this way. In the spatial case, the dynamics is much richer and more complex, because transitions and compound motions can occur (as explained for instance in Namouni 1999; Christou 2000). This is why we prefer to leave it for the next phase of the work.

The current paper is organized as follows: In Section 2, we recall the averaged problem of circular restricted three-body problem for the co-orbital motion in the planar case and how the approximation can be applied to classify co-orbital objects in the Solar system. In Section 3, it is explained how the training and testing data are generated. In Section 4, the whole algorithmic pipeline is detailed, while in Section 5 the results are given together with a critical analysis on the procedure. In Sections 6 and 7, a possible future direction is proposed and the conclusions are drawn.

2 COPLANAR CO-ORBITAL ASTEROIDS IN THE SOLAR SYSTEM

The main idea considered by Di Ruzza, Pousse & Alessi (2023) was to show how an integrable approximation of the restricted three-body problem can be applied to describe the dynamics of real natural objects and the goal was to provide a general catalogue of co-orbital objects in the Solar system in the coplanar case and a tool to visualize them.

We recall here the general setting and main features that will be important for the present work. More details can be found in Pousse & Alessi (2022) and Di Ruzza, Pousse & Alessi (2023). The theoretical model is the Planar Circular Restricted Three-Body Problem where a massless body is interacting by gravitational attraction with two massive bodies. The Hamiltonian describing the motion of the massless body can be written as

$$\mathcal{H}\left(\boldsymbol{r},\dot{\boldsymbol{r}},\lambda_{\mathrm{p}}\right)=\frac{\|\dot{\boldsymbol{r}}\|^{2}}{2}-\frac{\mu}{\|\boldsymbol{r}\|}-\frac{(\mu+\mu_{\mathrm{p}})\,\varepsilon}{\left\|\boldsymbol{r}-\boldsymbol{r}_{\mathrm{p}}\left(\lambda_{\mathrm{p}}\right)\right\|}+(\mu+\mu_{\mathrm{p}})\,\varepsilon\,\boldsymbol{r}\cdot\boldsymbol{r}_{\mathrm{p}}\left(\lambda_{\mathrm{p}}\right), \tag{1}$$

where $\boldsymbol{r},\dot{\boldsymbol{r}}\in\mathbb{R}^{2}$ are, respectively, the heliocentric position and velocity vectors of the massless body (the asteroid); μ, μ_p are the mass parameters of the massive primary body (the Sun) and of the massive secondary body (the planet), respectively;

$$\varepsilon:=\frac{\mu_{\mathrm{p}}}{\mu+\mu_{\mathrm{p}}}$$

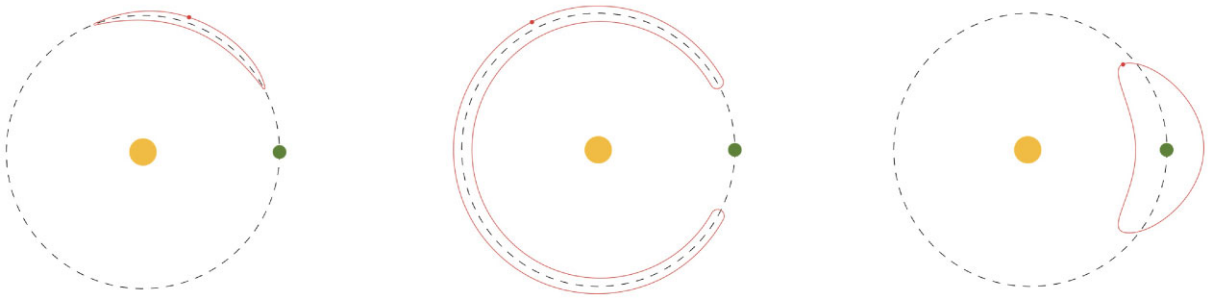


Figure 1. In red, a sketch of the tadpole motion (left), horseshoe motion (centre), quasi-satellite motion (right), in the synodic reference system. The yellow circle represents the Sun and the green one the planet.

is a dimensionless parameter characterizing the mass ratio of the Sun–planet system; the heliocentric vector $\boldsymbol{r}_{\rm p}(\lambda_p)$ denotes the position of the planet, for a given value of the mean longitude λ_p, which follows the solution of the two-body problem for the Sun–planet system. Usually, the Hamiltonian (1) is analysed in the synodic reference frame rotating with the planet. It is well-known that the problem admits five equilibrium points, called Lagrangian points and denoted by L_j for $j = 1, \ldots, 5$. If ε is small enough, we could rewrite the Hamiltonian (1) as

$$\mathcal{H}\left(\boldsymbol{r}, \dot{\boldsymbol{r}}, \lambda_{\rm p}\right) = \mathcal{H}_{\rm K}\left(\boldsymbol{r}, \dot{\boldsymbol{r}}\right) + (\mu + \mu_{\rm p})\,\varepsilon\,\mathcal{H}_{\rm P}\left(\boldsymbol{r}, \lambda_{\rm p}\right),$$

where $\mathcal{H}_{\rm K}$ is the unperturbed Kepler motion of the massless body (around the Sun) and $\mathcal{H}_{\rm P}$ is the perturbation depending on the gravitational influence of the planet and, then, we consider the averaged problem with respect to the fast angle $\lambda_{\rm p}$ obtaining the new Hamiltonian

$$\overline{\mathcal{H}} = \mathcal{H}_{\rm K} + \overline{\mathcal{H}}_{\rm P},$$

where $\overline{\mathcal{H}}_{\rm P}$ is the average over the period of revolution of the planet with respect to the fast angle λ_p.

We assume that the particle and the secondary are in a 1:1 MMR, that is, their orbits have the same value of semimajor axis. Within this approximation, the problem can be studied by means of the action-angle variables (θ, u), defined as follows:

$$\theta := \lambda - \lambda_{\rm p}$$

is the resonant angle (being λ the mean longitude of the asteroid) and

$$u := \sqrt{\frac{a}{a_{\rm p}}} - 1$$

is its conjugated action whose modulus measures the distance to the exact Mean Motion Resonance, with a and $a_{\rm p}$ being the semimajor axis of the asteroid and of the planet orbit, respectively; the exact 1:1 MMR is obtained for $(\dot{\theta}, u) = (0, 0)$. Note that the angle θ is the same used in other works on co-orbital dynamics, e.g. Morais (1999), Nsvorny et al. (2002), Mikkola et al. (2006), and Qi & Qiao (2022).

In this system, the quantity

$$\Gamma = \sqrt{a_{\rm p}}\left(1 - \sqrt{1 - e^2}\right)$$

is a first integral of the problem, being e the eccentricity of the asteroid orbit. For different values of $\Gamma \in [0 : \sqrt{a_{\rm p}}]$, the phase portrait in resonant variables (θ, u) allows to understand the whole co-orbital motion structure. In the planar circular case we can have three types of co-orbital motion, depicted in Fig. 1 in the synodic reference system. The tadpole (TP) motion (on the left) stemming from L_j with $j = 4, 5$ is such that θ experiences a periodic oscillation around a given $\theta_j(\Gamma)$ satisfying $23.9^\circ < (-1)^j\theta_j(\Gamma) < 180^\circ$; the horseshoe (HS) motion (in the middle), stemming from L_3 is such that θ oscillates around 180° with a large amplitude that decreases as long as Γ increases; the quasi-satellite (QS) regime (on the right) is such that θ librates around zero for $\Gamma > 0$.

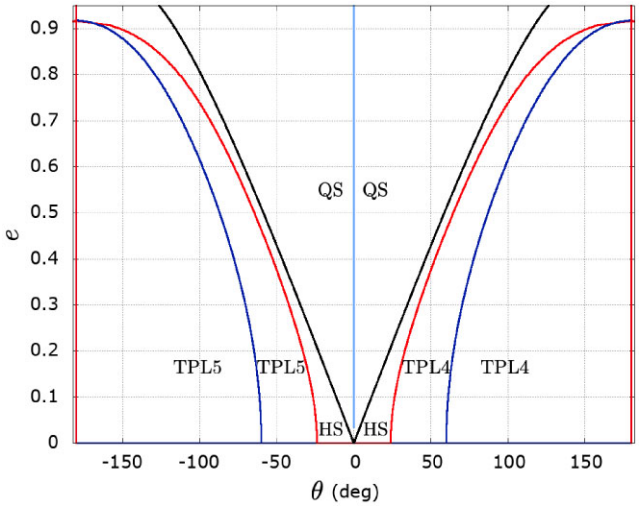


Figure 2. The (θ, e)-map of the co-orbital motion defined by the section $u = 0$. The black and red thick curves stand, respectively, for the singularity of collision and the crossing of the separatrices that originate from L_3 (thick red curve). They divide the map in three regions. The QS domain is between the dark curves; the HS region, split in two parts, is between the separatrix (red curve) and the dark curve; the TP regions are inside the separatrices (respectively, TPL4 for positive values of the angle θ and TPL5 for negative values of the angle θ).

In the given phase space, the co-orbital trajectories are solutions located in the neighborhood of $u = 0$ and such that θ oscillates around the given value. The crossing with the section $u = 0$, that corresponds to $a = a_{\rm p}$, provides a way to understand the global evolution of the dynamics at varying Γ, or equivalently, the eccentricity e of the asteroid's orbit. In this way it is possible to derive a (θ, e)-map, represented in Fig. 2, that allows to classify the different domains of co-orbital motion. We remark that, in first approximation, this map is invariant with respect to the mass parameter ε, so it has the same features for all the planets.

In the upper panels of Fig. 3, the graphs of the evolution of the time series (t, θ) of the three real examples of asteroids in the different regimes TP, HS, QS are plotted. In these cases, the evolution appears very regular, while in bottom panels, three less regular cases are reported for comparison.

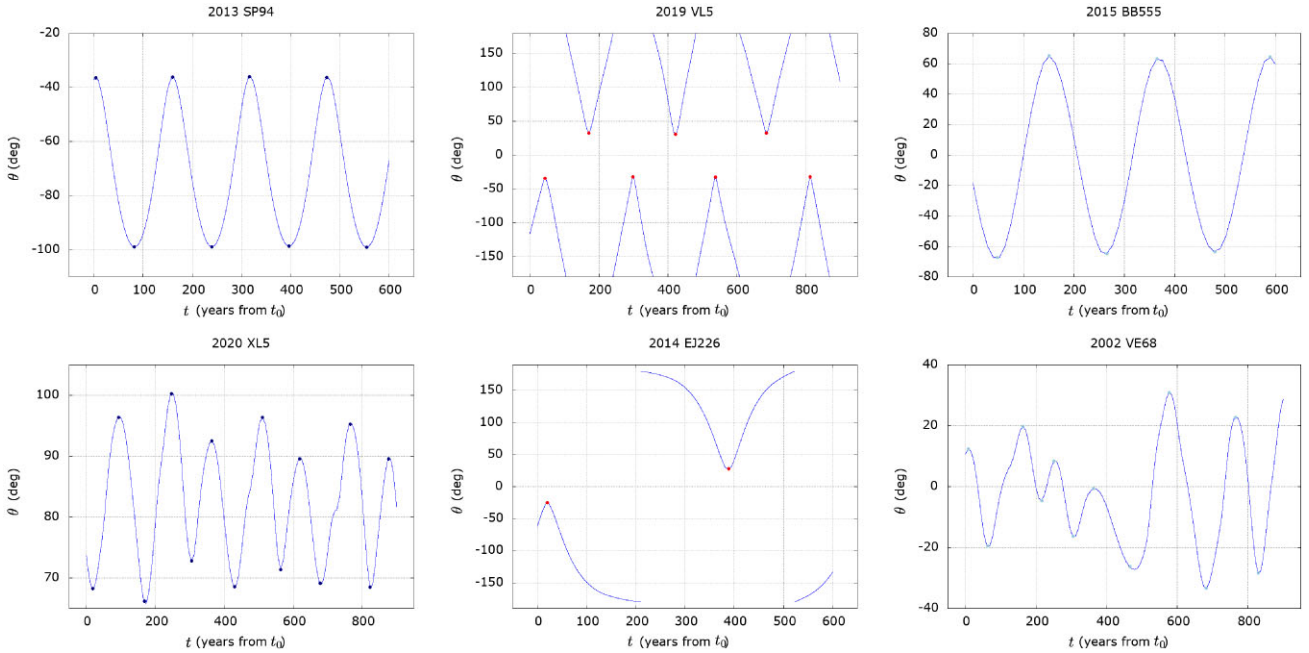


Figure 3. Upper: evolution of the angle θ versus time of three real asteroids in a regular co-orbital motion; from left to right, respectively, TP with Jupiter, HS with Earth, QS with Jupiter. Bottom: evolution of the angle θ versus time of three real asteroids in co-orbital motion with non-regular oscillations; from left to right, respectively, TP with Earth, HS with Jupiter, QS with Venus.

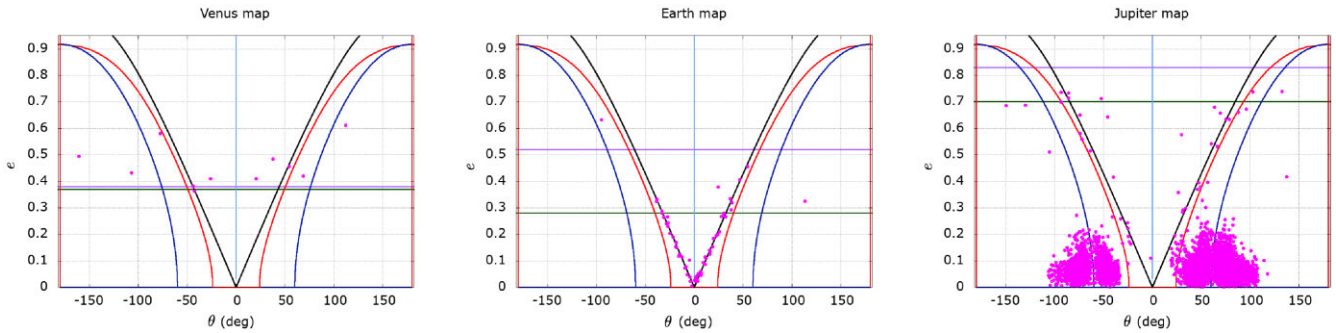


Figure 4. The (θ, e)-maps for the three planets; from left to right, respectively, Venus, Earth, and Jupiter. The points in magenta represent the distribution of co-orbital asteroids in the (θ, e)-map at a reference date, while the two horizontal lines stand for the eccentricities of an object in co-orbital motion with the considered planet P when it crosses the orbit of the inner and the outer planet (respectively in green and purple) with respect to P. The figures are already used in Di Ruzza, Pousse & Alessi (2023).

It is important to underline that the analysis done in the current work, and described in the next Sections, takes specifically into account the time evolution of the resonant angle θ. Subsequently, we will exploit the time series (t, θ) in order to recognize the different kinds of co-orbital regime as shown in Fig. 3.

In Di Ruzza, Pousse & Alessi (2023), co-orbital asteroids of Venus, Earth, and Jupiter have been analysed to show a practical application of the (θ, e)-map just explained. After a suitable filtering on the asteroid orbital elements in order to fulfil the resonance condition and the quasi-coplanar configuration at a given epoch, the ephemerides of asteroids have been computed by means of JPL HORIZONS API service (Giorgini et al. 1996; Giorgini & Yeomans 1999; Standish 1999; NASA 2022) for an interval of time of about 900 yr. The ephemerides data of real asteroids have been compared with the theoretical model and a very good correspondence has been found. Asteroids in quasi-coplanar co-orbital motion with Venus, Earth, and Jupiter have been catalogued according to their co-orbital dynamics and their representation can be seen in Fig. 4. A very refined analysis has been done checking *by hands* if the time series (t, θ) of each asteroid (as represented in Fig. 3) was in agreement with its position in the (θ, e)-map (Fig. 4). The results presented in Di Ruzza, Pousse & Alessi (2023) are very promising for TP, HS, and QS motion: under given assumptions, data of real observations fit very well with theory. The analysed series comprised also transitions (TR) between different co-orbital regimes as well as the compound (CP) motion (a particular combination between QS and HS dynamics).[1] In this case, the map was not able to accurately catch the behaviour,

[1]We refer to Namouni (1999) and Namouni, Christou & Murray (1999) for more details about the appearance of these kinds of motion.

Table 1. Summary of the data available.

Series	HS	QS	TPL4	TPL5	Total
Real	14	15	11	10	50
Ideal simulated	668	528	581	222	1999
Perturbed simulated	61	54	147	85	347

as expected, since TR and CP are proper of the three-dimensional model, not of the planar one.

At this point, an automatic tool capable of distinguishing the different co-orbital regimes becomes essential in order to improve our study. Indeed, in the future we aim to extend the analysis for a longer time span (order of thousands of years or more), to consider the spatial problem including asteroids with very high inclination and to understand better and classify TR and CP motions. All these information would be desirable to create a complete catalogue of asteroids in co-orbital motion with all the planets in the Solar system.

For these reasons, an ML approach in this problem is highly recommended in order to deal with a huge number of very long time series that can exhibit very rich dynamical behaviours. The aim of the present and coming works is to become able to manage any kind of ephemerides data of real asteroids, for short, medium, and long time-scales also when transitions between different co-orbital motions occur or when new kinds of motion appear, as, for example, the compound motions. In what follows, we will consider only TP, HS, and QS orbits since the foundations of the work are the results obtained in Di Ruzza, Pousse & Alessi (2023). In particular, we will classify co-orbitals motions belonging to the four classes QS, HS, TPL4 (a tadpole around the equilibrium position L_4), and TPL5 (a tadpole around the equilibrium position L_5).

3 DATA

Let us underline that our final goal is to be able to recognize, through the use of ML, co-orbital dynamics of real asteroids for short, medium, and long time-scales also when transitions between different co-orbital motions occur or when new kinds of motion appear, as, for example, the compound motions.

The data described in this section are the basis to outline the work done by the ML algorithms. As mentioned before, the information used in this work is the time evolution of the angle θ, computed considering three different sources of data, as summarized in Table 1.

In general, training an ML algorithm requires large amounts of data in order to provide accurate predictions. In our case, obtaining numerous time series of real asteroids with regular trends and clearly attributable to a single class (QS, HS, TPL4, TPL5) is not straightforward as real cases may present some complex behaviours, sometimes making labelling difficult and unclear. In particular, a high number of asteroids among those considered can escape from the given resonance or experience a co-orbital transitions.

We start our work by using the time series of asteroids reported in table 3, 4, 5 of the paper Di Ruzza, Pousse & Alessi (2023), that refer to ephemerides of real asteroids, obtained through the JPL Horizons system with a full dynamical model.

Looking at those tables, it is evident that most of the asteroids exhibit motions with different co-orbital dynamics and, as previously stated, these cases must be excluded so that, as shown in Table 1, the real cases data set used in the current work turns out to be composed by only 50 series, that is an absolutely insufficient number for a training set.

To overcome this issue, a data set containing simulated data of ideal cases is introduced. This kind of data can be produced by using suitable model and initial conditions (as depicted in the following) in order to get the four desired classes. It is possible to obtain as many cases as we need and we produced a total number of 1999 time series of ideal cases. This data set allows us to train the ML models with a consistent number of cases with well-known labels (i.e. motion clearly attributable to a single class), leaving the real cases data set for testing purposes.

On the other hand, to have more data to evaluate the performance of our pipeline, we decided to increase the number of cases that can be used. To this aim, we generated time series deviating from the ideal ones by perturbing the model used to generate ideal cases. This process only partially enlarges the number of cases to be used; in fact, by adding perturbations, the time series become more similar to real cases and most of them must be eliminated because escapes from the resonance or transitions between different co-orbital regimes appear. For this reason, the number of perturbed cases can not be as large as the ideal ones. As reported in the last row of Table 1, the total number of produced perturbed series is 347.

A detailed description of how the data are obtained is provided below.

(1) Ephemerides data of real asteroids are obtained from the JPL HORIZONS system (Giorgini et al. 1996; Giorgini & Yeomans 1999; Standish 1999; NASA 2022), following the approach adopted in Di Ruzza, Pousse & Alessi (2023). In this case, from the data base analyzed in Di Ruzza, Pousse & Alessi (2023), we have selected 50 asteroids that exhibit a regular tadpole, horseshoe, quasi-satellite behaviour, that is, we excluded the compound motions and transitions. In this case, the simulated data cover an interval of time equal at most to 900 yr. We refer to these data as *real data*. Note that from this set we have excluded the cases that are catalogued in a different way by the Minor Planet Center or other authors (see Di Ruzza, Pousse & Alessi 2023, section 4.3 or Greenstreet, Gladman & Juric` 2023).

(2) Ideal cases of TP, HS, QS motions are generated by propagating the equations of motion of the Circular Restricted Three-Body Problem with initial conditions obtained from the (θ, e)-map in the corresponding orbital domain (see Fig. 2). In this case, the initial condition in the synodic reference system is computed starting from the heliocentric orbital elements (a, e, i, ω, Ω, M) in the inertial system, by assuming the initial semimajor axis *an* equal to 1, the eccentricity e given by the map, the initial inclination i, the longitude of the ascending node Ω, and the mean anomaly M equal to 0 and the argument of pericentre ω equal to θ. In this case, the simulated data cover an interval of time equal to 3000 yr. We refer to these data as *ideal simulated data* and we produced a total number of 1999 time series of such cases.

(3) Perturbed cases from the ideal cases are computed by propagation by means of the REBOUND software (Rein & Liu 2012), considering a dynamical model that accounts for Sun, Moon, and the planets from Mercury to Mars. We have assumed physical units and the ecliptic plane at J2000 as the reference plane and initial conditions for the massive bodies from the JPL Horizons system at t_0 = JD 2305537.5. Since we are interested in orbits that are in co-orbital motion with a given planet in a coplanar approximation, the easiest choice is to assume that the planet is the Earth, so that we can take as initial condition for the virtual asteroid $a = 1$ AU and $i = \Omega = M = 0$. The other orbital elements e, ω are computed using the theoretical (θ, e) −map. That is, assuming, for instance, a quasi-satellite orbit, we know from the map that θ and e should

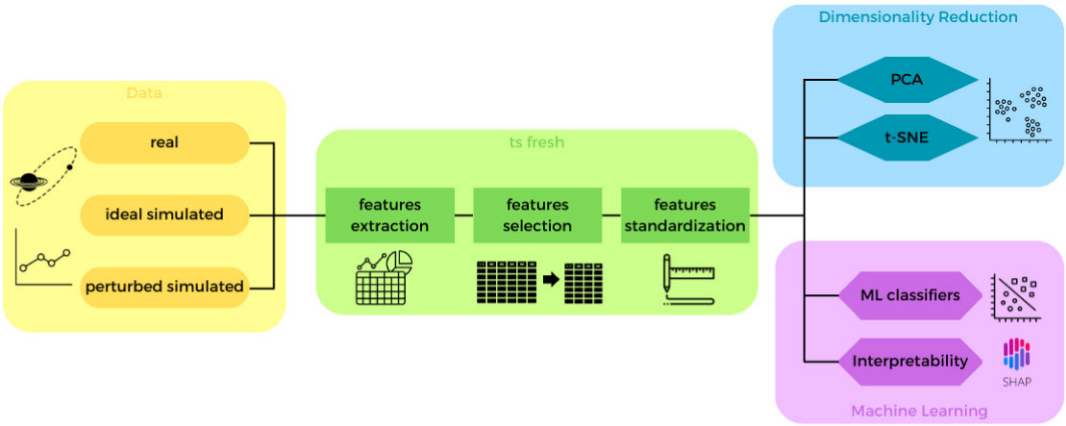


Figure 5. Data analysis workflow. The first step is the time series preparation, followed by the TSFRESH python package block where features are extracted and possibly selected and standardized. The final step regards the Machine Learning analysis performed using Dimensionality reduction algorithms (PCA and t-SNE) and classification algorithms (SVM, Random Forest, and XGBoost).

belong to a well-defined range (see Fig. 2). So, $\theta = \theta^*$ is given by this range and

$$\theta^* = \lambda - \lambda_{\rm planet} \equiv \lambda - \lambda_{\rm Earth}$$

by definition. The unknown is thus λ, that is,

$$\lambda = \theta^* + \lambda_{\rm Earth} = \theta^* + \omega_{\rm Earth} + \Omega_{\rm Earth} + M_{\rm Earth}.$$

Since by definition $\lambda = \omega + \Omega + M$, and we assume $\Omega = M = 0$, we get

$$\omega = \theta^* + \lambda_{\rm Earth} = \theta^* + \omega_{\rm Earth} + \Omega_{\rm Earth} + M_{\rm Earth}.$$

The simulated data cover an interval of time equal to 3000 yr. We refer to these data as *perturbed simulated data* and we produced a total number of 347 time series for this data set. They present variations to the ideal cases that resemble the behaviour of real objects, although no further perturbations have been added.

We are aware that if we had included Jupiter, the dynamics would have been more realistic in the long term. But, many orbits would have escaped from the resonant regimes or moved to a different one. It is certainly fundamental to develop a tool that can handle the co-orbital dynamics to the maximum extent, but we believe that research advances step by step. Without understanding how to develop an effective tool for the simplest, although not trivial, case, it is not possible to pave the way for a general tool, that will be able to classify all the possible situations in an accurate way and this is why we have considered this data set as an augmented data set to test the algorithms.

We note that data produced as described in point (2) and (3) above could be also interpreted as a good test of the results obtained in the previous paper Di Ruzza, Pousse & Alessi (2023). Indeed, we have chosen initial conditions (θ, e) in the (θ, e)-map and propagated them in order to obtain the desired kind of co-orbital motion.

4 DATA ANALYSIS WORKFLOW

As shown in Fig. 5, our data analysis workflow can be conceptually divided in three macro blocks. The first step consists in preparing and labelling the data described in Section 3, i.e. the output of the propagation of orbital elements of the asteroids. The data are collected in out format files: each file is associated with a single asteroid and it contains seven columns corresponding, respectively, to time (in Julian date), elapsed time in years (starting from t_0), semimajor axis a, eccentricity e, inclination i, resonant angle θ, and associated action u. The filenames contain acronyms useful to recognize the name of the asteroid, the kind of co-orbital motion, the planet that the asteroid is in resonance with and the kind of propagation used to get the data [points (1), (2), and (3) in Section 3]. In this way, files can be easily shared if required. It is important to stress that in this work we focus only on the time evolution of the variable angle θ, but the other information can turn out to be useful for future analysis.

These tabular data are passed to the next block, where the TSFRESH python package (e.g. Christ et al. 2018) provides a systematic time series feature extraction thanks to the combination of established algorithms from statistics, time series analysis, signal processing, and non-linear dynamics.

Before giving the extracted features to the Machine Learning classification algorithms, two additional steps can be applied: selection and standardization. Selection can be performed thanks to TSFRESH, which represents a robust feature selection algorithm (e.g. Li et al. 2017), while standardization can be obtained by any kind of library such as SCIKIT-LEARN pre-processing functions (e.g. Pedregosa et al. 2011).

The final classification step (last two blocks in Fig. 5) is performed in two parallel branches, with two classes of ML algorithms involved, namely, Dimensionality Reduction and Classification algorithms.

Before moving into a deeper explanation of all the details regarding the steps involved in the data analysis workflow, it is worth noting how our approach based on features extraction and standard Machine Learning algorithms is very well suited for our case where we have two constraints: data numerosity and physical interpretability. Both these constraints encourage an approach based on Machine Learning algorithms where the requirement on the number of data to train the algorithm is less tight with respect to Deep Learning. At the same time, thanks to the features extraction, a time series of any length can be converted into a finite number of features, all of them holding a physical meaning. This physical meaning is deeply important, because not only at the end of the whole data analysis workflow it is possible to identify the most important features responsible for a good time series classification (Feature Importance), but in addition we can look at the discriminating features between the different classes of signals, recovering a physical understanding of such processes.

4.1 Features extraction and selection: the TSFRESH open-source package

In order to train an ML model, features need to be extracted from the data. In our case a total of 789 features are extracted from each time series representing the time evolution of the angle $\theta(t)$ by the Python package TSFRESH (e.g. Christ et al. 2018). For a detailed description of the meaning of each feature please refer to Christ et al. (2023).

After feature extraction, usually, it is worth to introduce a step of *Feature Selection*. This step can be performed in different ways or not performed at all. However, in general, it has been demonstrated (e.g. Guyon, Elisseeff & Kaelbling 2003) that Feature Selection can improve ML performances. Therefore, we decided to implement such step in our workflow using a built-in function of TSFRESH, which provides a feature selection method based on Mann–Whitney Test. In our case, this step reduces the number of features to 239.

4.2 Features standardization

Again, pre-processing data is an essential step to achieve good classification performance, with the importance of data standardization (or normalization) for improving the performance of ML algorithms described in many studies as stated in Singh & Singh (2020). In our study, features are standardized using the SCIKIT-LEARN function StandardScaler (e.g. Pedregosa et al. 2011).

4.3 Dimensionality reduction

The process of transforming data from a high-dimensional space into a low-dimensional space with the goal of keeping the low-dimensional representation as close as possible to the inherent dimension of the original data is known as *Dimensionality Reduction*. There exist many different ML algorithms able to perform such transformation on data. In this work, we focus on two of them, namely, *Principal Components Analysis* (e.g. Cozzolino, Power & Chapman 2019) and *t-distributed Stochastic Neighbor Embedding* (t-SNE; e.g. Van der Maaten & Hinton 2008; Arora, Hu & Kothari 2018; Kobak & Berens 2019). PCA and t-SNE operate in two different ways: PCA is a linear method that seeks to preserve as much variance as possible and the global structure of the data, while t-SNE is a non-linear optimized technique that concentrates on preserving local similarities between data points. Additionally, PCA uses a well-known transformation making it a deterministic technique. On the other hand, t-SNE is a stochastic optimized method, which tend to preserve points which are close to each other. However, the method does not construct an explicit function that maps high-dimensional points to a low-dimensional space, but it just optimizes low-dimensional positions of the data points directly. Since it does not define a data transformation function, the method cannot be applied to newer data, but a newer optimization must run.

Both algorithms are Dimensionality Reduction techniques particularly well suited for the visualization of high-dimensional data sets as in this case, where, after the feature selection step, the number of features is still above 200. The utility of such kind of algorithms is twofold: on the one hand they can be used as unsupervised learning methods which allow to visualize the data distribution in two dimension, providing a deep insight on whether and, in case, how the data can be divided in the higher dimensional space. Moreover, they usually can give an idea of how the classifiers will perform. Indeed, well clustered data visualized by Dimensionality Reduction methods are usually well classified by ML algorithms, whereas the contrary is not necessarily true, meaning there could be data with a low degree of clustering where the classification algorithms still perform very well.

4.4 ML classification

We use three ML algorithms: Support Vector Machine (SVM; e.g. Cervantes et al. 2020), Random Forest (RF; e.g. Biau & Scornet 2016), and XGBoost (XGB; e.g. Chen & Guestrin 2016). We evaluate the performances of these algorithms with different combinations of training and test sets, as reported here:

(i) trained on real data and tested on real data;
(ii) trained on ideal simulated data and tested on real data;
(iii) trained on ideal simulated data and tested on perturbed simulated data;
(iv) trained on ideal simulated data and tested on real and perturbed simulated data.

4.4.1 Cross-Validation

When evaluating the performances of an ML model, it is highly important to validate its stability. This step is called *validation* and it consists in making sure that the model has learned the right patterns of the data and it is not picking up too much noise. In other words, it evaluates the model's ability to generalized on unseen data.

In Machine Learning, the most used validation technique is *Cross-Validation* (CV). It consists in splitting the data set into multiple subsets, usually called 'folds', then training the model on some of the folds and evaluating it on the remaining fold. This process is repeated multiple times, each time changing the remaining fold. The result is the mean score of all the performed tests. This allows to train and test the model on different data partitions, providing a robust and unbiased estimate of a model's performance.

There are many types of Cross-Validation; for this work we use a technique named *k-folds Cross-Validation* (e.g. Fushiki 2011), where the data set is divided in k folds and $k-1$ folds are used as training set and the remaining one as test set.

4.4.2 Hyperparameters tuning

When dealing with an ML model, one of the main aspects of designing the structure is a step called *Hyperparameters Tuning*, which consists in finding the best combinations of hyperparameters' models in order to achieve the best performance. Unfortunately, there are no rules or formulas to calculate these parameters, and an approach based on an extensive exploration of the hyperparameters' space along with some experience is the only way to find them, making hyperparameters tuning a computationally long and tedious process. In Python, many techniques have been developed to automate the tuning of hyperparameters and in this work we apply two of them: *GridSearchCV* and *RandomizedSearchCV*. Both these techniques make use of *k-fold Cross-Validation*.

4.4.3 SHAP: features interpretability

Machine Learning models are frequently considered 'black boxes', which make their interpretation challenging. In order to understand the main features that affect the output of the model, we can leverage on Explainable Machine Learning techniques that can unravel some of these aspects (e.g. Roscher et al. 2020). One very promising technique is the SHapley Additive exPlanations, more commonly known as SHAP (e.g. Lundberg & Lee 2017; Lundberg et al. 2018,

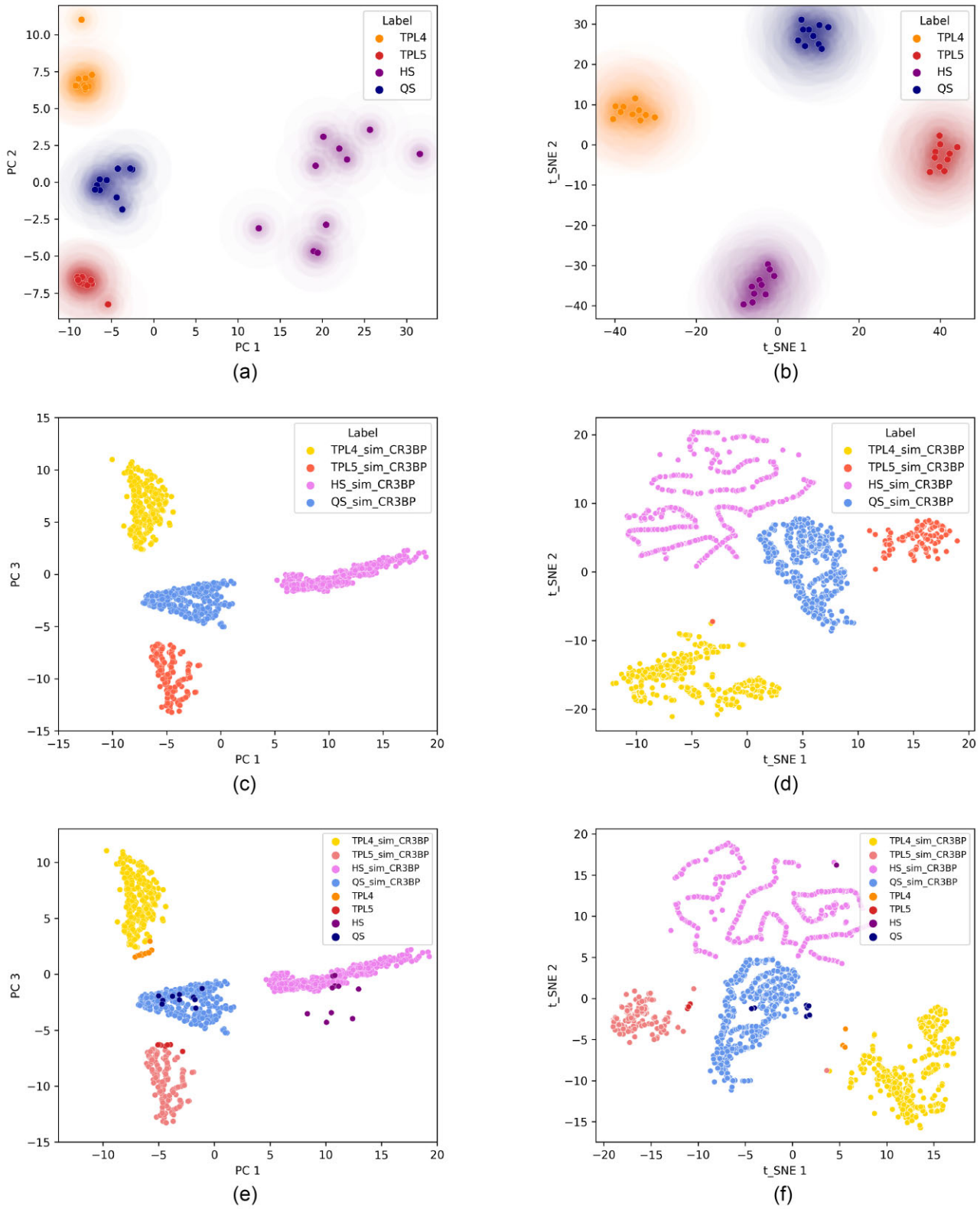


Figure 6. PCA and t-SNE of selected and standardized features extracted from: real data (a) and (b); ideal simulated data (c) and (d); overlapping between ideal simulated and real data clusters (e) and (f). In this last case it is worth to note as the orange points representing the real TPL4 cases overlap the yellow points representing the simulated TPL4 cases; the red points representing the real TPL5 cases overlap the light-red points representing the simulated TPL5 cases; the purple points representing the real HS cases overlap the violet points representing the simulated HS cases; the blue points representing the real QS cases overlap the light-blue points representing the simulated QS cases.

Table 2. Machine Learning multi-class classifiers results obtained with different combinations of training and test sets divided by algorithm. Because this is a multi-class classification, AUC, Recall, Precision, and f1 are averaged. In the Average AUC the acronym 'ovo' stands for one-versus-one and it computes the average AUC of all possible pairwise combinations of classes.

Training set	Test set	Accuracy (per cent)	Balanced Acc. (per cent)	'ovo' Average AUC	Average recall	Average precision	Average f1
			Support Vector Machine				
Real	Real	100	100	1.0	1.0	1.0	1.0
Ideal	Real	98.0	98.3	0.995	0.980	0.981	0.980
Ideal	Perturbed	100	100	1.0	1.0	1.0	1.0
Ideal	Real + perturbed	99.7	99.7	0.999	0.997	0.998	0.997
			Random Forest				
Real	Real	100	100	1.0	1.0	1.0	1.0
Ideal	Real	98.0	98.3	0.998	0.980	0.981	0.980
Ideal	Perturbed	100	100	1.0	1.0	1.0	1.0
Ideal	Real + perturbed	99.5	99.2	1.0	0.995	0.995	0.995
			XGBoost				
Real	Real	100	100	1.0	1.0	1.0	1.0
Ideal	Real	98.0	97.7	1.0	0.980	0.981	0.980
Ideal	Perturbed	100	100	1.0	1.0	1.0	1.0
Ideal	Real + perturbed	99.7	99.8	1.0	0.997	0.998	0.997

2020; Van den Broeck et al. 2022; Mitchell, Frank & Holmes 2022). It is based on Shapley values, which use game theory to assign credit for a model's prediction to each feature or feature value, increasing the transparency and the interpretability of Machine Learning models (e.g. Molnar 2022). In particular SHAP is known for its 'Consistency' property. SHAP values do not change when the model changes unless the contribution of a feature changes. This means that even when the model architecture or parameters change, SHAP values still offer a coherent interpretation of the behaviour of the model.

In our case, SHAP is applied to the ML models used for time series classification.

5 RESULTS

The results are presented in the following, according to the considered techniques.

5.1 Unsupervised ML: PCA and t-SNE

As stated in Section 4.3, Dimensionality Reduction techniques can be used to discover whether a high-dimensional data set presents separate clusters when projected in lower dimensional space (e.g. bi-dimensional). Therefore, the first step of our analysis has been to perform PCA and t-SNE on the features extracted from the real time series (real data) to see if they would cluster into four separated groups corresponding to four classes: QS, HS, TPL4, TPL5 (described in Section 2). PCA and t-SNE visualizations show four well separated clusters, as can be appreciated in Figs 6(a) and (b), respectively, where real data are considered.

Next, we performed PCA and t-SNE on the ideal simulated data to determine whether the trend of clustering in the four groups was also present in this data set. As it can be appreciated in Figs 6(c) and (d), clusters are still well visible.

Finally, given the positive results of the previous tests, we have applied the Dimensionality Reduction techniques on a data set containing both the real and ideal simulated data expecting an overlap between the real and simulated clusters for each class. The encouraging results of this analysis are reported in Figs 6(e) and (f). It is worth to observe that in these plots, PCA and t-SNE show the overlapping between real and simulated data clusters. In particular, the orange points representing the real TPL4 cases overlap the yellow points representing the simulated TPL4 cases; the red points representing the real TPL5 cases overlap the light-red points representing the simulated TPL5 cases; the purple points representing the real HS cases overlap the violet points representing the simulated HS cases; finally, the blue points representing the real QS cases overlap the light-blue points representing the simulated QS cases. This overlapping between clusters of real and simulated data in the reduced space confirms that the features extracted from these two data sets are similar and meaningful. In particular, these results confirm our expectations that both data sets are extracted from the same data distribution, making them suitable for the deeper machine learning analysis shown hereafter.

5.2 Supervised ML

While Dimensionality Reduction techniques allow to visualize high-dimensional data and eventual clusters within them, supervised ML algorithms provide an actual classification of the data. In our case, six classification metrics are considered to evaluate the supervised ML algorithms performances: *Accuracy, Balanced Accuracy, ROC AUC, Recall, Precision, f1*. A full description of the metrics can be found in SCIKIT-LEARN (2023a)

It is worth to note how some ML algorithms do not require features normalization, such as Random Forest, while for some others, such as Support Vector Machine, the normalization step strongly improves the classification performances (e.g. Singh & Singh 2020; Ozsahin et al. 2022). This peculiarity can be ascribed to the intrinsic differences in the working principles at the basis of each algorithm.

As was already noted, another crucial step that is typically (but not always) necessary to enhance classification performances is features selection. Our data shows that this is not the case; the outcomes are unaffected by the pre-processing stage. It should be highlighted, nevertheless, that this step generally needs to be preserved in the data analysis workflow. This is not the case for our data, results not being affected by this pre-processing step. However, it should be noted that in general such step must be kept in the data analysis workflow, evaluating its importance case by case. Concerning our

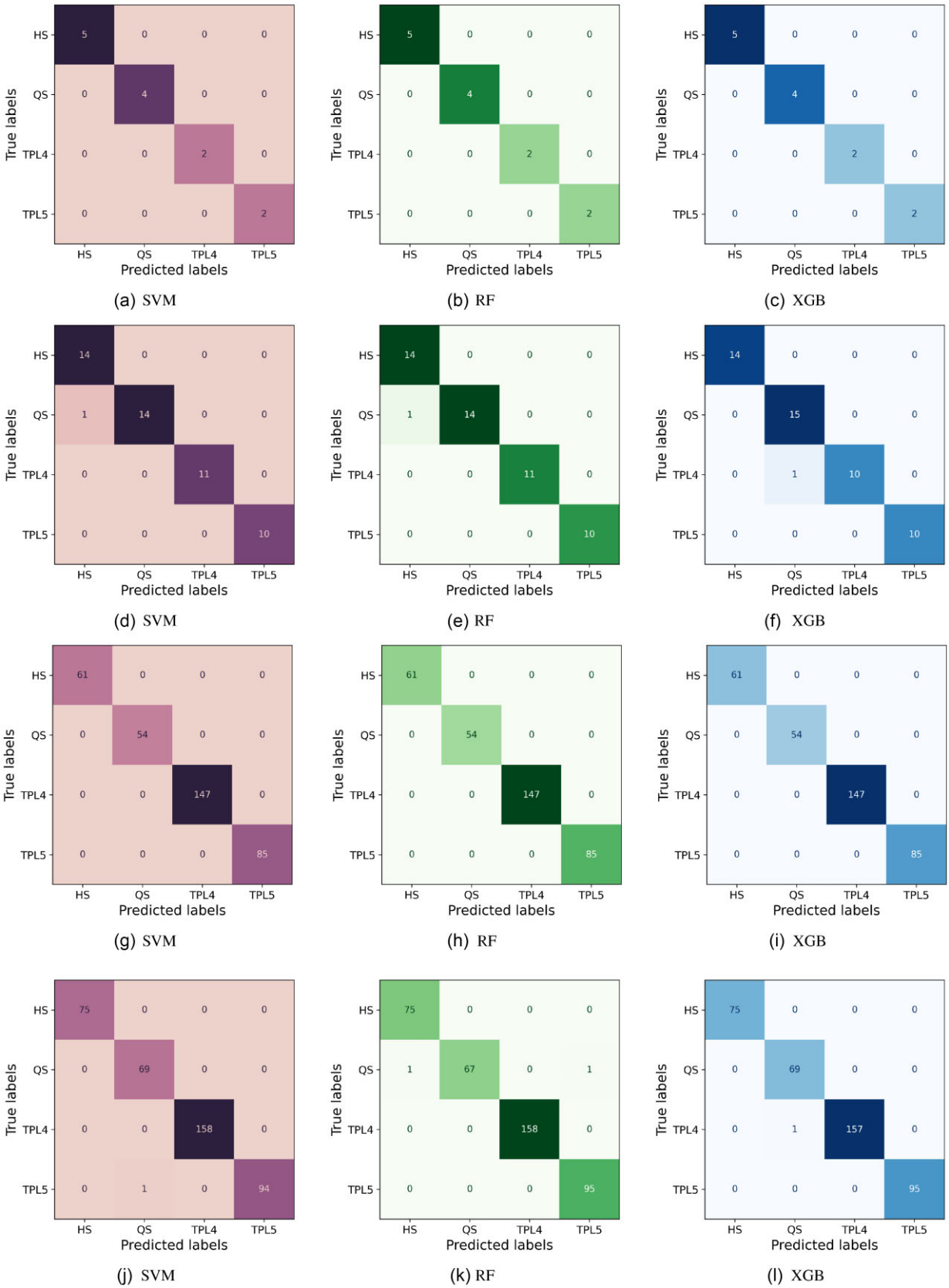


Figure 7. Confusion matrix for SMV (a), RF (b), and XGB (c) algorithms when trained and tested on real data. Confusion matrix for SMV (d), RF (e), and XGB (f) algorithms when trained on ideal simulated data and tested on real data. Confusion matrix for SMV (g), RF (h) and XGB (i) algorithms when trained on ideal simulated data and tested on perturbed simulated data. Confusion matrix for SMV (j), RF (k), and XGB (l) algorithms when trained on ideal simulated data and tested on real and perturbed simulated data.

Table 3. Machine Learning selected hyperparameters. A full description of their meaning can be found, for instance, in xgboost (2023a) and SCIKIT-LEARN (2023b, c).

Algorithm hyperparameters	Training set – test set			
	Real – real	Ideal – real	Ideal – pert.	Ideal – real + pert.
Support Vector Machine				
C	0.0001	1	0.001	1
Gamma	0.0001	0.001	0.1	0.001
Kernel	linear	linear	linear	linear
Random Forest				
n° estimators	190	100	300	300
XGBoost				
Colsample bytree	0.668	0.668	0.668	0.668
Learning rate	0.0765	0.0765	0.0765	0.0765
Max depth	5	5	5	5
Min child weight	1	1	1	1
n° estimators	70	70	70	70
Subsample	0.409	0.409	0.409	0.409

work, the results reported in this section are then relative to data sets containing all the extracted features.

5.2.1 *Test results*

The classification performances of the three used supervised ML algorithms (SVM, RF, and XGB, see Section 4.4) are reported in Table 2 for four different combinations of training and test sets. Although the motivations behind the chosen approach have already been partially described above, we remark the following observations. First of all, the real cases data set is limited, therefore it is impossible to give a clear answer regarding the generalization capability of our models to unseen data when trained and tested on real data. For this particular reason we introduced the ideal and perturbed simulated data sets, where the ideal one is intended for training purposes leaving the perturbed one to testing ones.

The hypothesis regarding the use of the ideal simulated as training set is confirmed by the fact that the classifiers trained in this way classify correctly the real series with an accuracy that reach 98 per cent. Lastly, classifiers trained on ideal simulated data and tested on perturbed simulated data obtain an accuracy of 100 per cent for all algorithms, while a slightly lesser accuracy is achieved testing on real and perturbed data.

All classification results are reported in Fig. 7, where confusion matrices for each performed test are presented. A *Confusion Matrix* is a type of visualization particularly well suited for evaluating the performance of an ML algorithm. The rows of the matrix represent the actual labels of the test set while the columns represent the labels predicted by the algorithm. Accordingly, the corrected predictions can be found along the diagonal of the matrix and the wrong ones outside of it.

In Table 3, they are reported all the selected hyperparameters for each performed test divided by algorithm.

5.2.2 *Cross-Validated results*

As introduced in Section 4.4.1, Cross-Validation is a crucial step to evaluate the model's ability to generalize on unseen data and it provides a more accurate evaluation of the model's performance.

Results obtained with a five-fold Cross-Validation are reported in Table 4, where we test on different combinations of the three data sets described in Section 3.

The mean accuracy relative to the real cases data set is quite high, but as already mentioned in the previous paragraph this may be due to the very limited dimensions of the data set. In fact, this case is the one with the highest CV error score (4 per cent) appearing on the table. Adding the ideal simulated data set, not only increases the mean accuracy (up to 99.9 per cent for XGB) but it also decreases the CV error score by an order of magnitude (0.09 per cent for XGB).

The third row of Table 4 is relative to the combination of the two simulated data sets, where we reach extremely high accuracy and quite low CV error score for all algorithms.

Finally, the algorithms' performances is cross-validated using all the available data. Although this is the case with the highest number of series and highest variability we still achieve remarkably good results with a mean accuracy that reaches 99.9 per cent (for RF and XGB) and overall low CV error score.

It is important to note how in the current section we report extremely good results, sometimes reaching up to 100 per cent accuracy, but these high numbers should not mislead the reader. The main purpose of this work is to demonstrate that our approach based on features extraction and Machine Learning algorithms works. For this reason, we have considered about 2400 series with quite regular trends and belonging to only four possible classes. Increasing the number of series, the number of classes or the irregularity of the series trends may lead to a worsening of the performances.

In other words, in this work we establish that our approach perfectly works in the most basic settings and, considering the extremely satisfactory results obtained, we plan to extend our goal to a more complete analysis increasing the complexity of the data in future works.

5.2.3 *Features importance*

Features Importance is one of the key points when using a Machine Learning algorithm for an application, where the interpretation and/or explanation of the results are as much important as finding good classification/regression results. The term *Features Importance* relates to methods for scoring each input feature given to the model

Table 4. Machine Learning multi-class classifiers results obtained in five-fold Cross Validation. Training sets and test sets contain, respectively, 80 per cent and 20 per cent of the data set. Standard deviation reported in parentheses. In the Average AUC the acronym 'ovo' stands for one-versus-one and it computes the average AUC of all possible pairwise combinations of classes.

Data set	Train	Test	Accuracy (per cent)	Balanced acc. (per cent)	'ovo' AUC	Precision	Recall	f1
Support Vector Machine								
Real	40	10	98.0 (± 4.0)	98.3 (± 3.3)	0.994(±0.011)	0.987(±0.027)	0.980 (± 0.040)	0.980 (± 0.040)
Real + ideal	1639	410	99.3 (± 1.3)	99.4 (± 1.0)	0.999(±0.001)	0.993(±0.012)	0.993 (± 0.013)	0.993 (± 0.014)
Ideal + pert.	1877	469	99.95 (± 0.09)	99.97 (± 0.07)	0.999(±0.001)	0.999(±0.001)	0.999 (± 0.001)	0.999 (± 0.001)
Real + Ideal + pert.	1917	179	99.42 (± 1.17)	99.53 (± 0.94)	0.999(±0.001)	0.994(±0.010)	0.994 (± 0.010)	0.994 (± 0.010)
Random Forest								
Real	40	10	98.0 (± 4.0)	98.3 (± 3.3)	0.995 (± 0.009)	0.985 (± 0.030)	0.980 (± 0.040)	0.979 (± 0.041)
Real + ideal	1639	410	99.9 (± 0.2)	99.9 (± 0.2)	1.0 (± 0.0)	0.999 (± 0.002)	0.999 (± 0.002)	0.999 (± 0.002)
Ideal + pert.	1877	469	100.0 (± 0.0)	100.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)
Real + ideal + pert.	1917	179	99.92 (± 0.17)	99.92 (± 0.17)	1.0 (± 0.0)	0.999 (± 0.002)	0.999 (± 0.002)	0.999 (± 0.002)
XGBoost								
Real	40	10	98.0 (± 4.0)	98.3 (± 3.3)	1.0 (± 0.0)	0.985 (± 0.030)	0.980 (± 0.040)	0.979 (± 0.041)
Real + ideal	1639	410	99.95 (± 0.10)	99.96 (± 0.08)	1.0 (± 0.0)	0.999 (± 0.001)	0.999 (± 0.001)	0.999 (± 0.001)
Ideal + pert.	1877	469	100.0 (± 0.0)	100.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)	1.0 (± 0.0)
Real + Ideal + pert.	1917	179	99.96 (± 0.08)	99.97 (± 0.07)	1.0 (± 0.0)	0.999 (± 0.001)	0.999 (± 0.001)	0.999 (± 0.001)

Figure 8. Common important features of the three supervised ML algorithms ranked by SHAP and Feature Importance tools.

based on how useful they are when predicting a target variable; the scores indicate what we call 'importance' of each feature. A higher score indicates that the particular feature will have a greater impact on the model. There are many ways to assign scores to the features; in our case we have used two different approaches: one based on a function provided by the algorithm library (e.g. SCIKIT-LEARN 2023b, 2023d; xgboost 2023b) and the other based on Shapley Values calculated by the SHAP package.

It is important to keep in mind that each algorithm has a tendency to weight features in a different way, even though some of them may be the same across all algorithms. In our case, it appears that there are no features common to all three algorithms, although we can find some common ones when comparing the algorithms two at a time. These common features are reported in Fig. 8.

Let us recall that, in this work, we have used three different classification algorithms: Random Forest, Support Vector Machines and XGBoost. Our results, reported in Figs 9(a)–(d), show that, for RF and SVM, most features are quite difficult to interpret, while the features ranking provided by XGBoost (Figs 9e and f) propose a more straightforward and interpretable explanation of the model. For XGBoost in particular, the two approaches for Features Importance point out two similar pools of features, where 7 out of 10 are the same. In addition, as shown in Figs 9(e) and (f), both approaches rank in the top positions features whose physical meaning is quite easy to deduct from their name, such as *theta sum values, theta standard deviation, theta mean,* and *theta variance*. Additionally, for XGBoost in Fig. 10, two other SHAP plots are shown: a *summary plot* where each feature's bar has a division into colours based on importance for each class and a *beeswarm plot*. A beeswarm plot is a data visualization tool used to display a summary of how the top features impact the model's output. Each point in the scatterplot represents a data point from the data set, the vertical line represents the baseline value, which may be the model's average prediction or the expected value of the output. The position of the point in relation to the vertical line reveals whether a feature makes a positive (increasing the prediction) or negative (decreasing the prediction) contribution to the prediction and this position is determined by the Shapley value of the data point. What is important to understand is that the farther a point is from the vertical line, the higher its impact will be on the output of the model, regardless of whether it is on the left or on the right side of the plot. For a more detailed explanation of the plot please refer to SHAP (2023).

6 FUTURE PERSPECTIVE: TIME SERIES WITH TRANSITION BETWEEN TRENDS, AN APPROACH BASED ON SLIDING WINDOWS

We are aware that the general case of time series observed could comprise different kinds of motion (such as the ones described and used in this work) due to transitions. In order to move towards this more complex real scenario, we have begun to work to identify regions in the time series where the kind of motion is of the same

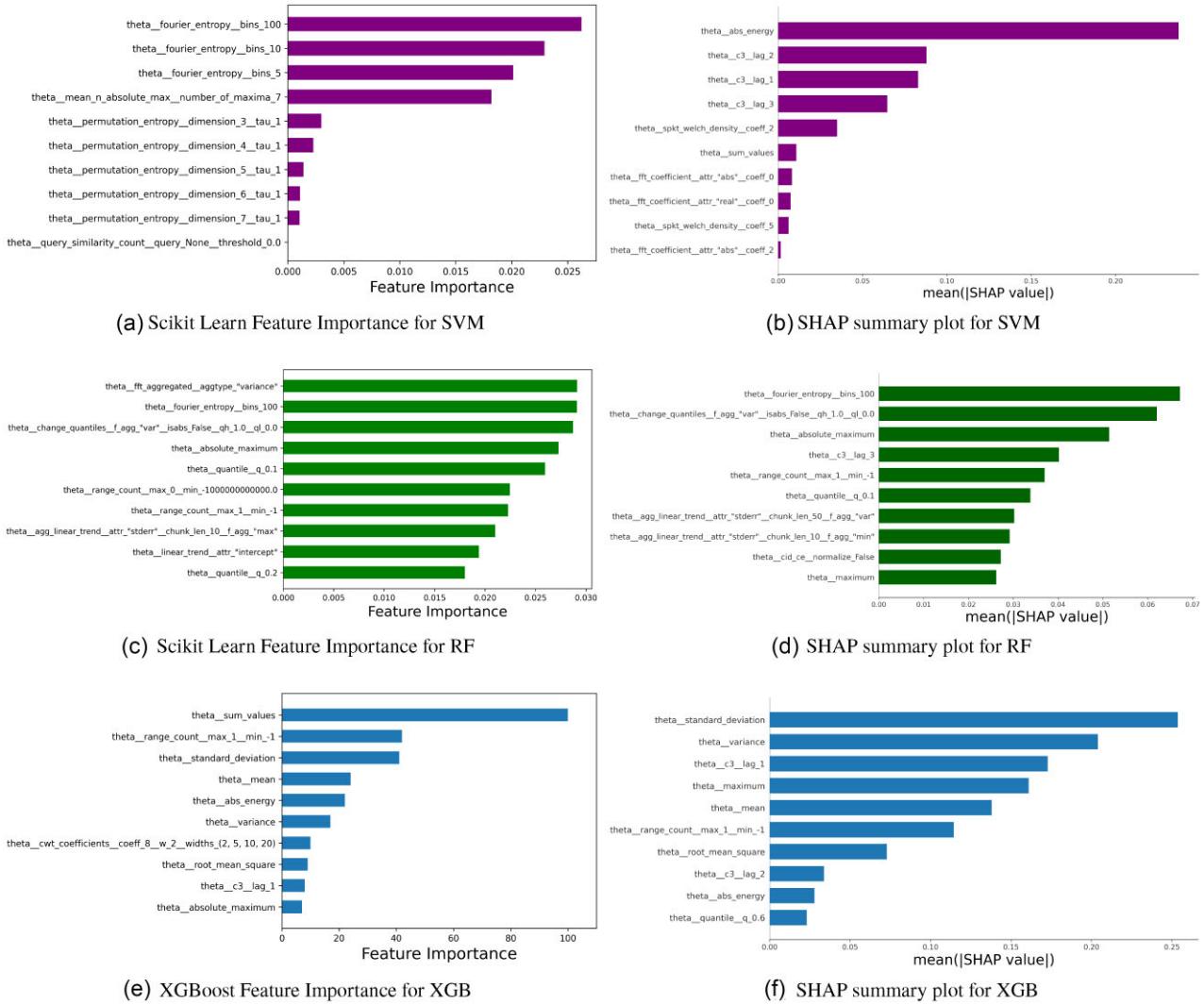


Figure 9. Feature Importances for the three different Machine Learning Algorithms, evaluated with SCIKIT-LEARN packages, and SHAP. In SHAP plots, the x-axis shows the features average impact on model output magnitude.

type. This capability would allow our data analysis pipeline to deal with any kind of scenario. As first approach, we have decided to leverage on standard packages for time series data analysis in the case of segmentation of non-stationary signals (e.g. Truong, Oudre & Vayatis 2020) and anomaly detection (e.g. Gensler & Sick 2018). We have performed some preliminary tests and some results are reported in this section and in the figure below. Our aim here is to give a possible direction for the next works.

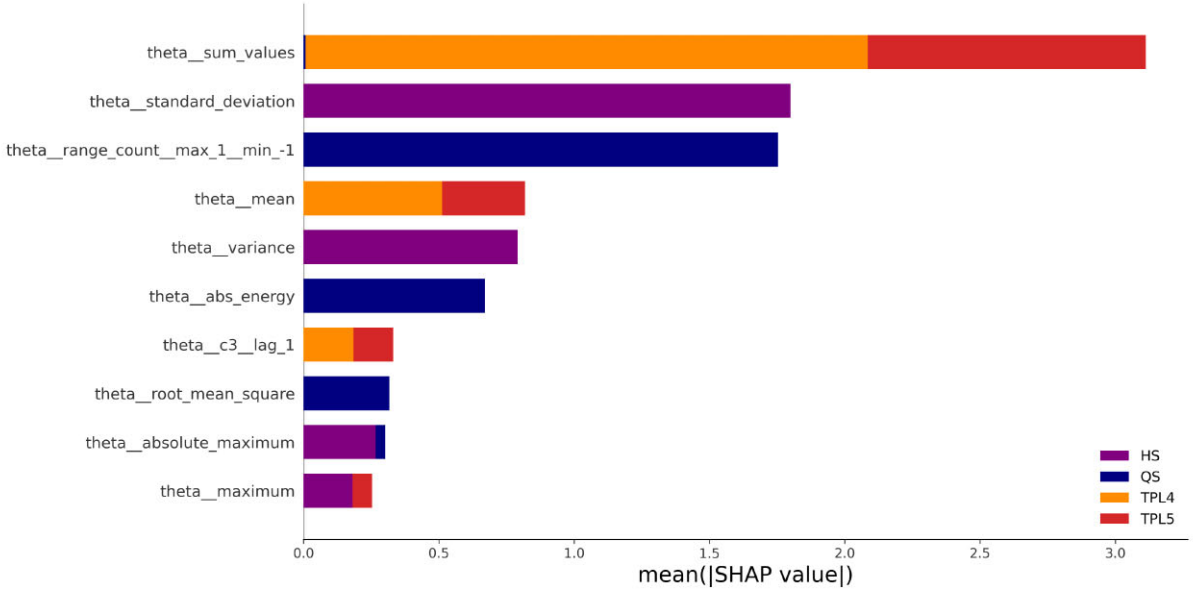
The results show that it is possible to arrange a semi-automatic division of the time series in the different trends, looking for example at the average over a fixed window length (in this case made of 8500 points) sliding over the $|\theta(t)|$ signal. The signal's mean of a window is compared to the mean of the following window; if the difference between those two values exceeds a certain threshold (empirically determined), a transition is detected.

However, despite the results can be useful and sometimes impressive (see Fig. 11), we have to investigate further how to generalize the definition of the time windows. This will be left to a future work.

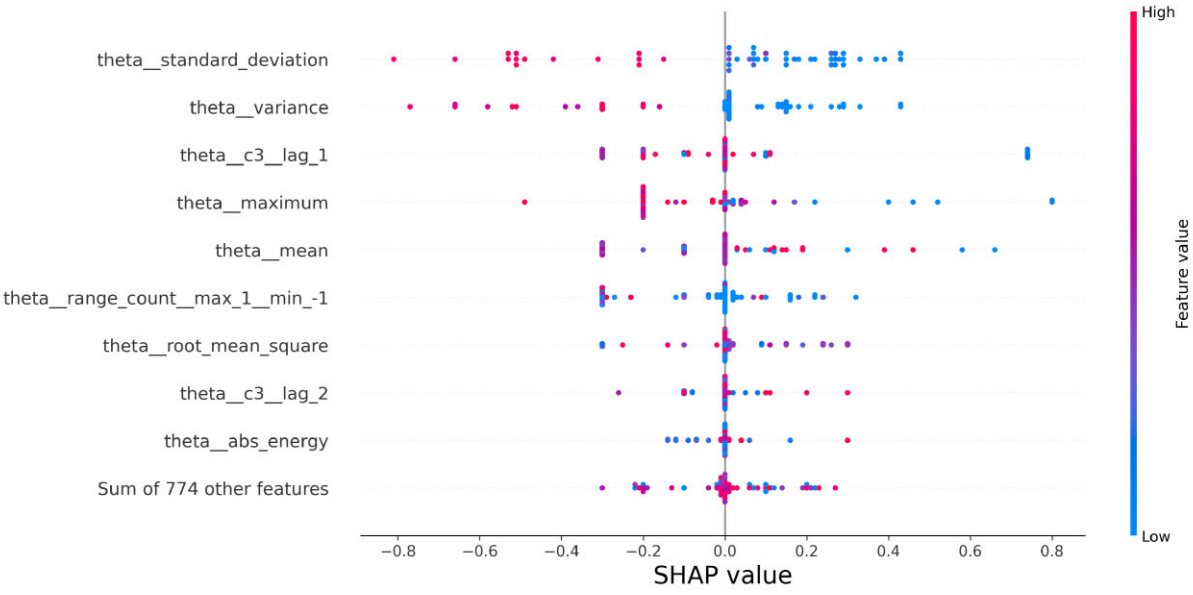
7 CONCLUSIONS

This work deals with the problem of classification of asteroids in co-orbital motion with a given planet using a Machine Learning approach. The main parameter analysed to determine the type of co-orbital motion is a suitable angle θ, that is defined following the assumption of the Planar Circular Restricted Three-Body Problem and its averaged approximation. The time evolution of θ allows to identify if the asteroid is in Tadpole motion, distinguishing between TPL4 (around the equilibrium point L_4) and TPL5 (around the equilibrium point L_5), HS motion or QS motion. We produce three different kinds of data set called real, ideal simulated, and perturbed simulated in order to apply Machine Learning algorithms. The data sets are formed by time series of the angle θ, that consist in its evolution in time for short and medium time-scale (about 900 yr for ephemerides data of real asteroids and 3000 yr for simulated cases).

The Python package TSFRESH is applied to such time series, extracting meaningfully features, which are selected and, if needed, standardized. Then, a Machine Learning pipeline based on algorithms for Dimensionality Reduction and Classification, is built, with



(a) SHAP summary plot for XGB with colour division based on importance for each class. The x-axis shows the features average impact on model output magnitude.



(b) SHAP beeswarm plot for XGB

Figure 10. SHAP results for XGBoost algorithm. On the top the summary plot while on the bottom the beeswarm plot.

the features extracted as input. The results show the power of such approach, with very well evident clusters in Dimensionality Reduction visualization plot and classification accuracy above 99 per cent. This paper aims to define a methodological approach to such kind of data, serving as a backbone model for further studies, where more and more complex cases are faced.

Our motivation was to develop a tool that can support an improvement and refinement of the theoretical method proposed in Di Ruzza, Pousse & Alessi (2023). To verify that the averaged approximation of the Circular Restricted Three-Body Problem can catch the real dynamics of co-orbital objects over significant time spans, under well-defined assumptions, we need a fast and automatic tool that can classify given time series. In this way, we will be able to compare the prediction of the averaged approximation with ephemerides data. The approach proposed here is to analyse time series obtained by propagation under different dynamical models. In general, the same ML pipeline can be applied to clone orbits and to longer and more complex time series, on condition that transitions can be identified. This aspect will be the focus of a future work.

Finally, we would like to remark that the short-term analysis can be useful to the space engineering field, in particular to select good candidates for a scientific mission, given the very high interest that asteroids are now receiving, not only for planetary defense purposes, but also as possible natural resources larders.

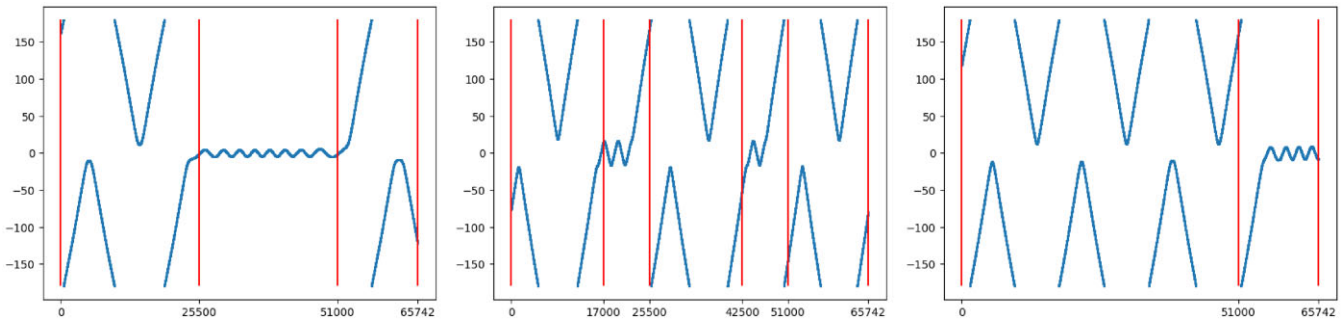


Figure 11. Three cases of real time series data: evolution of the resonant angle θ versus time; in the three cases several transitions between QS and HS regimes occur.

ACKNOWLEDGEMENTS

Authors express their gratitude to Tiago Azevedo and Pietro Liò from the University of Cambridge (UK), Michela Baccini, Chiara Marzi, Fabrizio Argenti, and Simone Marinai from the University of Florence (Italy), Stefano Diciotti from the University of Bologna (Italy), Alessandro Mecocci from the University of Siena (Italy), for fruitful discussion and advices on data analysis and to Lona Ceccherini for her support.

DATA AVAILABILITY

The data underlying this article will be shared on request to the corresponding author.

REFERENCES

Arora S., Hu W., Kothari P. K., 2018, in Bubeck S., Perchet V., Rigollet P., eds, Proc. Machine Learning Research, Vol. 75, Proc. 31st Conf. on Learning Theory. PMLR, p. 1455
Ball N. M., Brunner R. J., 2010, Int. J. Mod. Phys. D, 19, 1049
Biau G., Scornet E., 2016, Test, 25, 197
Van den Broeck G., Lykov A., Schleich M., Suciu D., 2022, J. Artif. Intell. Res., 74, 851
Carruba V., Aljbaae S., Lucchini A., 2019, MNRAS, 488, 1377
Carruba V., Aljbaae S., Domingos R. C., Lucchini A., Furlaneto P., 2020, MNRAS, 496, 540
Carruba V., Aljbaae S., Domingos R. C., Huaman M., Barletta W., 2022, Celest. Mech. Dyn. Astron., 134, 36
Celletti A., Gales C., Rodriguez-Fernandez V., Vasile M., 2022, Sci. Rep., 12, 1890
Cervantes J., Garcia-Lamont F., Rodríguez-Mazahua L., Lopez A., 2020, Neurocomputing, 408, 189
Chen T., Guestrin C., 2016, in Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. p. 785
Chen Y.-T. et al., 2018, PASJ, 70, S38
Christ M., Braun N., Neuffer J., Kempa-Liehr A. W., 2018, Neurocomputing, 307, 72
Christ et al., 2023, tsfresh github documentation. https://tsfresh.readthedocs.io/en/latest/ [last access Sep. 15 2023]
Christou A. A., 2000, Icarus, 1400, 1
Christou A., Asher D., 2011, MNRAS, 414, 2965
Connor L., van Leeuwen J., 2018, AJ, 156, 256
Cozzolino D., Power A., Chapman J., 2019, Food Anal. Methods, 12, 2469
Ćuk M. et al., 2012, MNRAS, 426, 3051
De la Fuente Marcos C., De la Fuente Marcos R., 2012, MNRAS, 427, 728
De la Fuente Marcos C., De la Fuente Marcos R., 2014, MNRAS, 445, 2985
Di Ruzza S., Pousse A., Alessi E. M., 2023, Icarus, 390, 115330
Erasmus N., Mommert M., Trilling D. E., Sickafoose A. A., van Gend C., Hora J. L., 2017, AJ, 154, 162
Erasmus N., McNeill A., Mommert M., Trilling D. E., Sickafoose A. A., van Gend C., 2018, ApJS, 237, 19
Farah W. et al., 2018, MNRAS, 478, 1209
Fluke C. J., Jacobs C., 2020, WIREs Data Min. Knowl. Discovery, 10, e1349
Francis P., Hewett P. C., Foltz C. B., Chaffee F. H., 1992, ApJ, 398, 476
Fushiki T., 2011, Stat. Comput., 21, 137
Gensler A., Sick B., 2018, Pattern Anal. Appl., 21, 543
Giorgini J., Yeomans D., 1999, NASA TECH BRIEFS NPO-20416, On-Line System Provides Accurate Ephemeris and Related Data.
Giorgini J. D. et al., 1996, BAAS, 28, 1158
Goodfellow I., Bengio Y., Courville A., 2016, Deep Learning. MIT Press
Greenstreet S., Gladman B., Juric` M., 2023, preprint (arXiv:2309.06609v1)
Guyon I., Elisseeff A., Kaelbling L. P., 2003, J. Mach. Learn. Res., 3, 1157
Hastie T., Tibshirani R., Friedman J., 2009a, The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd edn. Springer-Verlag, Berlin
Hastie T., Tibshirani R., Friedman J. H., Friedman J. H., 2009b, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Vol. 2. Springer-Verlag, Berlin
Ivezić Ž., Connolly A. J., VanderPlas J. T., Gray A., 2014, in Statistics, Data Mining, and Machine Learning in Astronomy. Princeton Univ. Press, Princeton
Jacobs C. et al., 2019, MNRAS, 484, 5330
Jordan M. I., Mitchell T. M., 2015, Science, 349, 255
Kamath C., 2022, Int. J. Data Sci. Anal. Available at: https://ui.adsabs.harvard.edu/abs/2023arXiv230513329K
Kinoshita H., Nakai H., 2007, Celest. Mech. Dyn. Astron., 98, 181
Knezevic Z., Milani A., 1994, in Milani A., di Martino M., Cellino A.eds, Vol. 160, Proc. IAU Symp. Asteroids, Comets, Meteors 1993. Kluwer, Dordrecht, p. 143
Knezevic Z., Lemaître A., Milani A., 2002, in Bottke W. F., Cellino A., Paolicchi P., Binzel R. P., eds. Asteroids III .University of Arizona Press, Tucson. p. 603
Kobak D., Berens P., 2019, Nat. Commun., 10, 5416
Lanusse F., Ma Q., Li N., Collett T. E., Li C.-L., Ravanbakhsh S., Mandelbaum R., Pczos B., 2017, MNRAS, 473, 3895
LeCun Y., Bengio Y., Hinton G., 2015, Nature, 521, 436
Li J., Cheng K., Wang S., Morstatter F., Trevino R. P., Tang J., Liu H., 2017, ACM Comput. Surv., 50, 1
Liu X., Zhang F., Hou Z., Mian L., Wang Z., Zhang J., Tang J., 2021, IEEE Trans. Knowl. Data Eng., 35, 857
Lundberg S. M., Lee S.-I., 2017, in Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R., eds, Advances in Neural Information Processing Systems 30. Curran Associates, Inc., p. 4765
Lundberg S. M. et al., 2018, Nat. Biom. Eng., 2, 749
Lundberg S. M. et al., 2020, Nat. Mach. Intell., 2, 2522
Van der Maaten L., Hinton G., 2008, J. Mach. Learn. Res., 9
Mikkola S. et al., 2004, MNRAS, 351, L63
Mikkola S. et al., 2006, MNRAS, 369, 15

Mitchell R., Frank E., Holmes G., 2022, GPUTreeShap: massively parallel exact calculation of SHAP scores for tree ensembles. PeerJ Computer Science ,
Molnar C., 2022, Interpretable Machine Learning, 2nd edn., Bookdown
Morais M. H. M., 1999, A&A, 350, 318
NASA, 2022, https://ssd-api.jpl.nasa.gov/doc/horizons.html (accessed October 18).
Namouni F., 1999, Icarus, 137, 293
Namouni F., Christou A. A., Murray C. D., 1999, Phys. Rev. Lett., 83, 2506
Nesvorny D. et al., 2002, Celest. Mech. Dyn. Astron., 82, 323
Ozsahin D. U., Mustapha M. T., Mubarak A. S., Ameen Z. S., Uzun B., 2022, in 2022 Int. Conf. on Artificial Intelligence in Everything (AIE). p. Lefkosa, 87
Pearson K. A., Palafox L., Griffith C. A., 2018, MNRAS, 474, 478
Pedregosa F. et al., 2011, J. Mach. Learn. Res., 12, 2825
Pourrahmani M., Nayyeri H., Cooray A., 2018, ApJ, 856, 68
Pousse A., Alessi E. M., 2022, Nonlinear Dyn., 108, 959
Qi Y., Qiao D., 2022, AJ, 163, 211
Rein H., Liu S. F., 2012, A&A, 537, A128
Roscher R., Bohn B., Duarte M. F., Garcke J., 2020, IEEE Access, 8, 42200
SHAP, 2023, beeswarm plot. https://shap.readthedocs.io/en/latest/example_notebooks/api_examples/plots/beeswarm.html (accessed October 18)
Scikit-Learn, 2023a, Metrics and scoring: quantifying the quality of predictions. https://scikit-learn.org/stable/modules/model_evaluation.html (accessed October 18)
Scikit-Learn, 2023b, SVC Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed October 18)
Scikit-Learn, 2023c, Random Forest Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed October 18)
Scikit-Learn, 2023d, Feature importances with a forest of trees. https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html (accessed October 18)
Shallue C. J., Vanderburg A. M., 2017, AJ, 155
Singh D., Singh B., 2020, Appl. Soft Comput., 97, 105524
Singh H. P., Gulati R. K., Gupta R., 1998, MNRAS, 295, 312
Smirnov E., 2023, Astron. Comput., 43, 100707
Smirnov E. A., Markov A. B., 2017, MNRAS, 469, 2024
Smirnov E. A., Shevchenko I. I., 2013, Icarus, 222, 220
Smullen R. A., Volk K., 2020, MNRAS, 497, 1391
Standish E. M., 1999, Interoffice Memorandum 312.F-98-048, JPL Planetary and Lunar Ephemerides, DE405/LE405. Jet Propulsion Laboratory, Pasadena, California
Truong C., Oudre L., Vayatis N., 2020, Signal Process., 167, 107299
xgboost, 2023a, XGBclassifier. https://xgboost.readthedocs.io/en/latest/python/python_api.html (accessed October 18)
xgboost, 2023b, xgboost.plot_importance. https://xgboost.readthedocs.io/en/stable/python/python_api.html#module-xgboost.plotting (accessed October 18)
Wajer P., 2010, Icarus, 209, 488
Wajer P., Kròlikowska M., 2012, Acta Astron., 62, 113
Whitmore B. C., 1984, ApJ, 278, 61

This paper has been typeset from a TEX/LATEX file prepared by the author.