



Universidade Estadual Paulista "Júlio de
Mesquita Filho"
CAMPUS DE GUARATINGUETÁ

Giovana Ramon Luiz Neto

**APLICAÇÃO DE *MACHINE LEARNING* PARA
MAPEAMENTO DE REGIÕES DE
ESTABILIDADE: Análise da performance do
modelo através de ajuste de hiperparâmetros e
threshold.**

Guaratinguetá/Brasil

2024

Giovana Ramon Luiz Neto

**APLICAÇÃO DE *MACHINE LEARNING* PARA
MAPEAMENTO DE REGIÕES DE ESTABILIDADE:
Análise da performance do modelo através de ajuste de
hiperparâmetros e *threshold*.**

Dissertação de Mestrado apresentada em 2024
ao Conselho da Pós Graduação da Faculdade
de Engenharia e Ciências do Campus de Gua-
ratinguetá, Universidade Estadual Paulista
"Júlio de Mesquita Filho", como parte dos
requisitos para obtenção do título de Mestre
em Física e Astronomia.

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Programa de Pós Graduação em Física e Astronomia

Orientador: Prof^o Dr. Rafael Sfair

Guaratinguetá/Brasil

2024

L953a	Luiz Neto, Giovana Ramon Aplicação de <i>Machine Learning</i> para mapeamento de regiões de estabilidade: análise da performance do modelo através de ajuste de hiperparâmetros e threshold / Giovana Ramon Luiz Neto – Guaratinguetá, 2024. 94 f : il. Bibliografia: f. 88-94 Dissertação (Mestrado) – Universidade Estadual Paulista, Faculdade de Engenharia e Ciências de Guaratinguetá, 2024. Orientador: Prof. Dr. Rafael Sfair de Oliveira 1. Aprendizado do computador. 2. Exoplanetas. 3. Astronomia. I. Título. CDU 52(043)
-------	---

GIOVANA RAMON LUIZ NETO

**ESTA DISSERTAÇÃO FOI JULGADA ADEQUADA PARA A OBTENÇÃO DO TÍTULO DE
“MESTRA EM FÍSICA”**

**PROGRAMA: FÍSICA E ASTRONOMIA
CURSO: MESTRADO**

APROVADA EM SUA FORMA FINAL PELO PROGRAMA DE PÓS-GRADUAÇÃO


Prof. Dr. Ernesto Vieira Neto
Coordenador

BANCA EXAMINADORA:



Documento assinado digitalmente

RAFAEL SFAIR DE OLIVEIRA

Data: 27/08/2024 14:40:21-0300

Verifique em <https://validar.iti.gov.br>

Prof. Dr. RAFAEL SFAIR DE OLIVEIRA
Orientador - UNESP



Documento assinado digitalmente

VALERIO CARRUBA

Data: 22/08/2024 20:53:08-0300

Verifique em <https://validar.iti.gov.br>

Prof. Dr. VALERIO CARRUBA
UNESP



Documento assinado digitalmente

RITA DE CASSIA DOMINGOS

Data: 22/08/2024 20:11:24-0300

Verifique em <https://validar.iti.gov.br>

PROF^a. Dr^a. RITA DE CASSIA DOMINGOS
Faculdade de Engenharia de São João da Boa Vista

AGOSTO de 2024

Eu dedico este trabalho a Felipe Machado Mathídios dos Santos.

DADOS CURRICULARES

GIOVANA RAMON LUIZ NETO

NASCIMENTO 17/05/2001 - Taubaté-SP

FILIAÇÃO Antonio Luiz Neto
Cristiane Ramon de Carvalho Luiz Neto

2019 - 2022 Bachalerado em Física
Universidade Estadual Paulista "Júlio de Mesquita Filho"

2023 - 2024 Mestrado em Física e Astronomia
Universidade Estadual Paulista "Júlio de Mesquita Filho"

Agradecimentos

Em primeiro lugar gostaria de agradecer ao meu orientador Prof. Dr. Rafael Sfair, que me acompanha desde o primeiro ano da graduação. Sou grata por toda atenção e esforço que dispõe aos seus orientandos, tornando nossa trajetória mais leve. Agradeço por todas as oportunidades, bem como pelo incentivo e aprendizado.

Agradeço à minha família que desde cedo fomenta meu interesse pela ciência e apoia minha decisão ao escolher física como profissão. Por estar ao meu lado em todos os momentos difíceis e me ajudar a seguir em frente.

Sou grata a todos os professores e professoras que transpuseram meu caminho até aqui, em especial ao projeto Astronomia no Sítio que marcou o início do meu interesse pela ciência e a astronomia. Gostaria de expressar minha profunda gratidão, ao professor Felipe, por me acompanhar desde o ensino fundamental, sempre me apoiando e incentivando em cada etapa da minha jornada acadêmica. Seu apoio foi fundamental na decisão sobre a faculdade, e o amor por astronomia e divulgação científica foram desencadeados por fagulhas depositadas por você. Obrigada pelas oportunidades e pelo trabalho com o planetário, sua dedicação foi inspiradora e permanecerá comigo.

Aos meus amigos da pós-graduação Raí e Tiago Francisco que em meio ao trabalho estão sempre presentes para tornar as coisas mais leves e divertidas, obrigada por todas as risadas e também pela disposição sempre que precisei de ajuda. Em especial ao Tiago que aceitou embarcar nesse projeto e ao longo de um ano, desenvolvemos muitas coisas juntos, sou grata pela parceria e dedicação durante deste período.

Aos meus irmãos e irmãs da vida, Milena, Breno, Ismael, Thaís, Íris e Giovanna, capazes de aturar o sumiço e estresse de uma estudante de pós-graduação e mesmo assim sempre estarem lá para conversar e se distrair. Estes capazes de conversar de assuntos sérios e em meio a isso arrancar risadas inimagináveis. Vocês fizeram parte disso e sempre serei grata.

Como último agradecimento, expresso a minha gratidão ao meu noivo, José Eduardo, por trazer significado a minha vida, ser meu suporte em todos os momentos difíceis, trazer alegria sempre que possível durante meu dia, compartilhar a complicada tarefa de ser adulto e se organizar e por todas as conversas profundas. Você é o responsável por eu acreditar que sou capaz, seu apoio foi fundamental para a conclusão deste trabalho.

Este trabalho contou com o apoio das seguintes entidades:

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil - UNESP
- Universidade Estadual Paulista "Júlio de Mesquita Filho" DFG - German Research
Foundation - Projeto 446102036

"Existe uma teoria que diz que, se um dia alguém descobrir exatamente para que serve o Universo e por que ele está aqui, ele desaparecerá instantaneamente e será substituído por algo ainda mais estranho e inexplicável. Existe uma segunda teoria que diz que isso já aconteceu."

Douglas Adams

Resumo

O número de exoplanetas cresce a cada ano e o mapeamento das regiões de estabilidade ao redor desses corpos é um estudo significativo, uma vez que leva a identificação da presença de potenciais satélites e sistemas de anéis. A fim de realizar essa investigação, usualmente são empregados conjuntos de simulações numéricas de N-corpos, caros computacionalmente. O objetivo principal deste trabalho foi aplicar modelos de *Machine Learning* para classificar mapas de regiões de estabilidade ao redor de sistemas planetários hipotéticos em órbitas do tipo-S. Foram desenvolvidas 100.000 simulações numéricas adimensionais do problema de três corpos (estrela-planeta-partícula) através do integrador Rebound/IAS15. Cada sistema explorava nove características diferentes, incluindo os elementos orbitais da partícula e do planeta, assim como a razão de massa do sistema. As simulações foram classificadas como instáveis ou estáveis e a amostra construída foi utilizada como conjunto de treino para os modelos de *Machine Learning*. Foram empregados cinco algoritmos diferentes de *Machine Learning*, e o melhor desempenho foi alcançado pelo *XGBoost*, com uma *accuracy* igual a 98,48%, uma *recall* e *precision* equivalentes a 94% para partículas estáveis e de 99% para partículas instáveis. A análise foi estendida para o mapeamento da estabilidade da região da *Sailboat*, utilizando quatro conjuntos de simulações numéricas diferentes, cada um composto por 300.000 condições iniciais. O intuito dessa fração do trabalho, foi identificar o limite da existência da estabilidade da *Sailboat* para diferentes razões de massa, excentricidades do corpo perturbador, inclinações e argumentos do pericentro da partícula, confirmando o estudo presente na literatura e acrescentando novas investigações para o comportamento da região. Através do emprego do melhor modelo para o *XGBoost*, foi atingida uma *accuracy* superior a 97% para todos os casos avaliados. Os algoritmos de *Machine Learning* reduziram o tempo computacional gasto em simulações numéricas para o problema de três corpos em uma previsão de alguns minutos.

Palavras-chave: machine learning; mapas de estabilidade; dinâmica orbital.

Abstract

The number of exoplanets increases every year and mapping stable regions around these bodies is a significant study, because it allows verification of the potential presence of satellites or even ring systems. These investigations are typically performed numerically by solving a large set of computationally expensive N-body simulations. The main objective of this work was to implement Machine Learning to predict maps of stable regions of S-type orbits around hypothetical planets. In total, 100,000 numerical simulations of three body problem (star-planet-test particle) were run using the Rebound package and IAS15 integrator. These simulations cover nine distinct features, including the orbital elements of the particle and the planet, as well as the mass ratio of the system. The simulations were classified as either stable or unstable, forming the dataset employed for training machine learning models. We applied five different supervised algorithms, our best performance was achieved utilizing the XGBoost, with an accuracy of 98.48%, a recall and precision of 94% for stable particles and 99% for unstable particles. The analysis was extended to mapping the stability of the Sailboat region, using four sets of different numerical simulations, each one composed of 300,000 initial conditions. The aim of this section of the work was to identify the limit of existence of the stability of the Sailboat for different mass ratios, eccentricities of the disturbing body and inclination and argument of periapsis of the particle, corroborating the studies present in the literature and adding new investigations of the behavior of the region. By applying the best model for XGBoost, an accuracy greater than 98% was obtained for all cases evaluated. Machine learning algorithms significantly reduce computational time in three-body simulations, it can predict a diagram of initial conditions within minutes.

Keywords: machine learning; stability maps; orbital dynamics.

Lista de Figuras

Figura 1 – Fluxograma representando as etapas deste trabalho.	20
Figura 2 – Exemplos de espécies da flor de Iris, classificadas através do algoritmo <i>Decision Tree</i>	21
Figura 3 – <i>Decision Tree</i> para a classificação da flor Iris.	22
Figura 4 – Algoritmo de histograma.	25
Figura 5 – Distribuição das classes de acordo com o <i>threshold</i>	30
Figura 6 – Representação da curva ROC de três modelos hipotéticos.	31
Figura 7 – Matriz da confusão para uma classe binária.	32
Figura 8 – Histograma da distribuição dos planetas em unidades de raio de Hill.	34
Figura 9 – Comparação entre os mapas de estabilidade.	36
Figura 10 – Variação das métricas de acordo com os valores de hiperparâmetros.	38
Figura 11 – Variação das métricas de acordo com os valores do hiperparâmetro <i>max depth</i>	39
Figura 12 – Variação da <i>accuracy</i> de acordo com o <i>threshold</i>	40
Figura 13 – Variação das métricas da classe estável de acordo com o <i>threshold</i>	41
Figura 14 – Variação das métricas da classe instável de acordo com o <i>threshold</i>	41
Figura 15 – Variação da <i>precision</i> e <i>recall</i> para as classes estáveis e instáveis de acordo com o <i>threshold</i>	42
Figura 16 – Desempenho dos cinco modelos testados.	43
Figura 17 – Matriz da confusão para <i>Decision Tree</i> e <i>Random Forest</i>	44
Figura 18 – Matriz da confusão para o <i>HGB</i> , <i>LightGBM</i> e <i>XGBoost</i>	45
Figura 19 – Importância dos atributos para o modelo <i>XGBoost</i>	47
Figura 20 – Regiões de estabilidade para satélites prógrados e retrógrados hipotéticos.	48
Figura 21 – Comparação entre os mapas de estabilidade.	49
Figura 22 – Performance para o <i>XGBoost</i> sobre os sistemas prógrado e retrógrado.	50
Figura 23 – Comparação entre os mapas de estabilidade.	51
Figura 24 – Desempenho do modelo sobre o sistema PDS110b.	52
Figura 25 – Comparação entre os mapas de estabilidade.	54
Figura 26 – Desempenho do modelo sobre o conjunto de satélites Inuit.	54
Figura 27 – Diagrama das condições iniciais de $e \times a$, exibindo a região da <i>Sailboat</i>	56
Figura 28 – Semieixo maior da estrela secundária (a_s) em relação à razão de massa do sistema binário (μ)	58
Figura 29 – Variação das métricas de acordo com os hiperparâmetros <i>eta</i> e <i>scale pos weight</i>	62
Figura 30 – Variação das métricas de acordo com o hiperparâmetro <i>max depth</i>	63
Figura 31 – Variação das métricas de acordo com os valores do <i>Threshold</i>	64

Figura 32 – Evolução da estabilidade da <i>Sailboat</i> para diferentes valores de μ	66
Figura 33 – Evolução da estabilidade da <i>Sailboat</i> para diferentes valores de μ	67
Figura 34 – Região da <i>Sailboat</i> para um sistema binário com $\mu = 0, 12$	68
Figura 35 – Variação das métricas de acordo com os valores dos hiperparâmetros <i>eta</i> e <i>scale pos weight</i>	70
Figura 36 – Variação das métricas de acordo com os valores do hiperparâmetro <i>max</i> <i>depth</i>	70
Figura 37 – Variação das métricas de acordo com os valores do <i>Threshold</i>	71
Figura 38 – Excentricidade máxima do corpo perturbador	72
Figura 39 – Evolução da região da <i>Sailboat</i> para diferentes excentricidades do corpo perturbador	73
Figura 40 – Variação das métricas de acordo com os valores dos hiperparâmetros <i>eta</i> e <i>scale pos weight</i>	75
Figura 41 – Variação das métricas de acordo com os valores do hiperparâmetro <i>max</i> <i>depth</i>	76
Figura 42 – Variação das métricas de acordo com os valores do <i>threshold</i>	76
Figura 43 – Regiões de estabilidade para diferentes inclinações orbitais da partícula.	78
Figura 44 – Evolução da <i>Sailboat</i> variando as inclinações das partículas.	79
Figura 45 – Variação das métricas de acordo com os hiperparâmetros <i>eta</i> e <i>scale pos</i> <i>weight</i>	81
Figura 46 – Variação das métricas de acordo com os valores do hiperparâmetro <i>max</i> <i>depth</i>	81
Figura 47 – Variação das métricas de acordo com os valores do <i>Threshold</i>	82
Figura 48 – Intervalo do argumento do pericentro da partícula onde existe a região da <i>Sailboat</i>	83
Figura 49 – Diagrama das condições iniciais ($e \times a$) para diferentes valores do argumento do pericentro.	84

Lista de Tabelas

Tabela 1 – Grade de elementos orbitais.	34
Tabela 2 – Resultados para o <i>hyperparameter tuning</i> para o <i>Decision Tree</i>	37
Tabela 3 – Resultados para o <i>hyperparameter tuning</i> para o <i>Random Forest</i>	37
Tabela 4 – Resultados para o <i>hyperparameter tuning</i> para o <i>LightGBM</i>	37
Tabela 5 – Resultados para o <i>hyperparameter tuning</i> para o <i>HGB</i>	37
Tabela 6 – Resultados para o <i>hyperparameter tuning</i> para o <i>XGBoost</i>	37
Tabela 7 – Valores para a área sob a curva ROC (AUC).	46
Tabela 8 – Elementos orbitais dos satélites de Saturno que pertencem ao grupo Inuit.	53
Tabela 9 – Sistemas binários de planetas anões e de seus satélites do Sistema Solar	59
Tabela 10 – Intervalos de parâmetros livres para a análise de diferentes razões de massa.	61
Tabela 11 – Resultados para o <i>hyperparameter tuning</i> para o <i>XGBoost</i>	61
Tabela 12 – Grade de parâmetros livres.	68
Tabela 13 – Resultados para o <i>hyperparameter tuning</i> para o <i>XGBoost</i>	69
Tabela 14 – Parâmetros livres para o caso da variação da inclinação orbital da partícula.	74
Tabela 15 – Resultados para o <i>hyperparameter tuning</i> para o <i>XGBoost</i>	74
Tabela 16 – Parâmetros livres analisados para o caso da variação do argumento do pericentro.	80
Tabela 17 – Resultados para o <i>hyperparameter tuning</i> para o <i>XGBoost</i>	80

Índice

1	INTRODUÇÃO	15
2	MACHINE LEARNING	18
2.1	Algoritmos	20
2.1.1	Decision Tree	21
2.1.2	Bagging	22
2.1.2.1	Random Forest	23
2.1.3	Boosting	23
2.1.3.1	XGBoost	24
2.1.3.2	LightGBM	24
2.1.3.3	HGB	24
2.1.4	Algoritmo Genético	25
2.2	Classes desbalanceadas	26
2.3	Busca de hiperparâmetros	27
2.3.1	Hiperparâmetros implementados	28
2.4	Busca de <i>threshold</i>	30
2.5	Métricas	31
3	REGIÕES DE ESTABILIDADE AO REDOR DE PLANETAS	33
3.1	Simulações Numéricas	33
3.2	Performance	36
3.2.1	Busca de hiperparâmetros e threshold	36
3.2.2	Análise das métricas	42
3.2.3	Importância dos atributos	46
3.3	Aplicações	47
3.3.1	Estabilidade ao redor de exoplanetas gigantes	47
3.3.2	Sistema PDS110b	50
3.3.3	Sistema de satélites Inuit	52
3.4	Comentários finais	55
4	ESTABILIDADE DA REGIÃO DA <i>SAILBOAT</i>	56
4.1	Sistemas Binários	57
4.2	Planetas anões	58
4.3	Simulações Numéricas	59
4.4	Análise da região <i>Sailboat</i> para diferentes razões de massa	61
4.4.1	Busca de Hiperparâmetros e <i>Threshold</i>	61

4.4.2	Resultados obtidos com o <i>XGBoost</i>	64
4.5	Análise da região <i>Sailboat</i> para o corpo excentricamente perturbado	68
4.5.1	Busca de Hiperparâmetros e <i>Threshold</i>	69
4.5.2	Resultados obtidos com o <i>XGBoost</i>	71
4.6	Análise da região <i>Sailboat</i> para diferentes valores da inclinação da partícula	74
4.6.1	Busca de Hiperparâmetros e <i>Threshold</i>	74
4.6.2	Resultados apresentados pelo <i>XGBoost</i>	77
4.7	Análise da região <i>Sailboat</i> com a variação do argumento do pericentro	79
4.7.1	Busca de Hiperparâmetros e <i>Threshold</i>	80
4.7.2	Resultados obtidos com o <i>XGBoost</i>	82
4.8	Comentários Finais	84
5	CONSIDERAÇÕES FINAIS	86
	REFERÊNCIAS	88

1 Introdução

O mapeamento de regiões de estabilidade ao redor de planetas é uma área de grande interesse dentro da dinâmica orbital, motivando estudos com diferentes configurações orbitais e gravitacionais. Um sistema binário é composto por dois corpos que orbitam um centro de massa externo, quando existe um terceiro corpo são possíveis três órbitas distintas: tipo-P, tipo-S e tipo-T. Uma órbita tipo-P ocorre quando o terceiro corpo orbita o centro de massa do sistema, enquanto no caso de uma órbita do tipo-S, o terceiro corpo orbita apenas um dos componentes binários com o segundo componente comportando-se como um perturbador (Rabl and Dvorak, 1988). Por fim, uma órbita do tipo-T é aquela onde o terceiro corpo é co-orbital com o segundo corpo localizado nos pontos Lagrangianos L4 ou L5 (Cuntz, 2013).

A descoberta de exoplanetas aumenta anualmente, a fim de explorar a existência de satélites ou sistemas de anéis ao redor desses planetas, são usualmente realizadas simulações de N-corpos. Diversos estudos estenderam esta análise, Holman and Wiegert (1999) investigaram numericamente a estabilidade de órbitas do tipo-S em um problema elíptico de três corpos. Como resultado, foi obtida uma expressão empírica para o semieixo maior crítico de uma partícula estável, relacionada a distância binária, a razão de massa e a excentricidade, sendo que inicialmente a partícula se encontrava em uma órbita circular e planar. Os autores analisaram um intervalo da razão de massa equivalente a $\mu = [0 - 0,9]$ e excentricidade binária $e = [0 - 0,8]$.

Neto and Winter (2001) examinaram a captura gravitacional no sistema Sol-Urano-satélite. Nesse estudo, foi investigado um diagrama das condições iniciais do semieixo maior em relação a excentricidade para identificar regiões estáveis ao redor de Urano. Além disso, foram consideradas partículas com várias inclinações iniciais e argumentos do pericentro.

Domingos et al. (2006) verificaram a estabilidade de satélites hipotéticos orbitando exoplanetas com razão de massa $\mu = 10^{-3}$. Como resultado, o trabalho derivou uma expressão empírica para o limite do semieixo maior em termos do raio de Hill, da excentricidade do planeta e do satélite, considerando órbitas prógradas e retrógradas.

Winter et al. (2010) estudaram possíveis regiões estáveis no sistema Plutão-Caronte, a análise do estudo envolveu o emprego de simulações numéricas para investigar a estabilidade de órbitas do tipo-S ao redor de Plutão e Caronte, considerando partículas em órbitas prógradas e retrógradas. Para o caso prógrado, os autores identificaram três regiões estáveis, de forma inesperada, uma das regiões localizada entre Plutão e Caronte foi a *Sailboat*, explorada pela primeira vez neste estudo.

Rieder and Kenworthy (2016) conduziram simulações de N-corpos para verificar a estabilidade de um sistema de anéis hipotético ao redor do candidato a planeta J1407b, impondo restrições para os parâmetros orbitais e físicos do sistema. Para o sistema PDS110b, Pinheiro and Sfair (2021) desenvolveram milhões de simulações numéricas do problema de três corpos, refinando parâmetros que segundo os dados observacionais permanecem indefinidos, tais quais massa e excentricidade do planeta, inclinação e tamanho do anel.

Para muitos sistemas, empregar um grande conjunto de simulações numéricas pode ser caro computacionalmente, além da integração exigir uma ampla demanda de tempo. Para otimizar esse processo, algoritmos de *Machine Learning* (ML) podem ser aplicados. A identificação de regiões de estabilidade com o emprego de *Machine Learning* já foi demonstrado em estudos anteriores. Tamayo et al. (2016) aplicam algoritmos de ML para prever a estabilidade de sistemas planetários compactos, os quais são compostos por dois planetas orbitando uma estrela central. Nesse trabalho, uma *accuracy* igual a 90% é alcançada com os modelos empregados. Os resultados de Tamayo et al. (2016) indicam que essa nova abordagem é em três ordens de magnitude mais rápida do que simulações diretas de N-corpos. Lam and Kipping (2018) empregam *Deep Neural Networks* (DNN) para prever a estabilidade de órbitas do tipo-P, inicialmente coplanares e circulares, para planetas circum binários com uma *accuracy* de até 86%.

Neste trabalho, propomos a aplicação de modelos de *Machine Learning* para classificar mapas de regiões de estabilidade ao redor de sistemas planetários hipotéticos em órbitas do tipo-S. Além disso, temos como objetivo analisar através do ML a estabilidade da região da *Sailboat*, variando parâmetros físicos e restringindo o intervalo em que a *Sailboat* pode existir.

Como os algoritmos supervisionados de ML requerem um conjunto de treino para aprender o comportamento dos dados a serem previstos, foram desenvolvidas, para as duas abordagens, diferentes simulações numéricas adimensionais do problema de três corpos elíptico. Essas simulações foram realizadas pelos colaboradores do trabalho, Tiago Francisco Pinheiro Leal Lins e Dr. Rafael Sfair.

Nosso trabalho explora um largo intervalo de parâmetros físicos e orbitais, os quais incluem a razão de massa do sistema e diversos elementos orbitais da partícula e do planeta, dessa forma, resolver numericamente esse problema seria pouco viável.

O trabalho foi organizado de maneira que o Capítulo 2 apresenta os conceitos principais de *Machine Learning* empregados, como também a breve definição dos cinco algoritmos analisados (seção 2.1). O Capítulo 3 descreve a construção do conjunto de dados por meio de simulações numéricas para o primeiro cenário apresentado, a atuação dos algoritmos de ML sobre a amostra e a comparação das previsões obtidas a partir do nosso modelo de *Machine Learning* com os resultados numéricos exibidos nos trabalhos de Domingos et al. (2006) e Pinheiro and Sfair (2021), assim como para o sistema de satélites

Inuit de Saturno. O Capítulo 4, apresenta a construção das simulações numéricas para a análise da região da *Sailboat*, bem como o desempenho do modelo de ML sobre esse conjunto de dados. Finalmente, o Capítulo 5 contém as considerações finais referentes ao estudo todo.

2 Machine Learning

De acordo com [Samuel \(1959\)](#), *Machine Learning* (ML) é o campo de estudo que dá aos computadores a habilidade de aprender sem ser explicitamente programado. Entende-se, portanto, que através dos algoritmos de ML a máquina é capaz de realizar uma tarefa por meio da experiência de aprendizado. O usuário pode oferecer a solução designada ao problema processado, sendo esse cenário conhecido como aprendizado supervisionado; já quando nenhuma correspondência é dada tem-se o caso não-supervisionado. Além disso, o aprendizado supervisionado pode ser caracterizado de forma que os rótulos dos dados são valores numéricos contínuos ou discretos, configurando uma regressão ou uma classificação, respectivamente.

As técnicas de *Machine Learning* vem ganhando grande destaque dentro da astronomia nas últimas décadas, com aplicações nas áreas de classificação de estrelas e galáxias, como vemos no projeto *Galaxy Zoo* ([Lintott et al., 2008](#)). Em dinâmica orbital, a utilização dos algoritmos de ML é emergente, entretanto recentemente diversos trabalhos de impacto foram publicados. O estudo de [Carruba et al. \(2022\)](#) realiza uma revisão completa da literatura disponível sobre dinâmica de asteroides e avalia segundo as métricas atuais as aplicações de *Machine Learning* em outros subcampos astronômicos.

Podemos citar alguns estudos de relevância, como [Smullen and Volk \(2020\)](#) que usaram o modelo *GBoost* para automaticamente classificar objetos recém-descobertos do Cinturão de Kuiper (KBOs) em quatro classes. [Carruba et al. \(2020\)](#) empregaram algoritmos de *Machine Learning* de classificação para identificar novos membros de famílias de asteroides, com base na distribuição orbital da família anteriormente conhecida. Eles compararam o resultado de nove algoritmos de classificação, e o método *Extremely Randomized Trees (ExtraTree)* apresentou a maior precisão, permitindo recuperar até 97% dos familiares identificados.

[Carruba et al. \(2021a\)](#) demonstra pela primeira vez a aplicação de *Artificial Neural Network (ANN)* para identificar automaticamente o comportamento de órbitas de asteroides afetadas pela ressonância de movimento médio M1:2 com Marte. O estudo confirmou que a ressonância M1:2 afeta principalmente as órbitas das famílias de asteroides Massalia, Nysa, e Vesta.

[Li et al. \(2022\)](#) aplicam *Artificial Neural Networks (ANNs)* para prever as trajetórias em ressonância 2:3, cobrindo um ciclo completo de libração. Neste trabalho é considerado um caso intermediário de movimento regular em um sistema não integrável: o comportamento de objetos na ressonância de movimento médio 2:3 com Netuno. O estudo pode auxiliar na classificação da grande população de KBOs a serem descobertos

em sistemas futuros.

Por fim, em um trabalho recente, Carruba et al. (2023b) utilizaram métodos de *Machine Learning* para prever as classes de mais de 68.000 asteroides sem nenhuma informação taxonômica prévia, detectando 409 novos possíveis asteroides do tipo V. O trabalho analisou os dados do *Dark Energy Survey (DES)*, correlacionando com outros levantamentos.

A função de classificação do *Machine Learning* utiliza um conjunto de treino, onde os rótulos são conhecidos, durante o aprendizado. Assim, ocorre uma generalização das classes através da associação do padrão presente no conjunto de dados com cada classe e consecutivamente é possível realizar a predição de amostras desconhecidas. Neste trabalho, os dados foram divididos em treino, validação e teste, sendo que os elementos destinados à validação são definidos pela imparcialidade na avaliação da eficácia. Mediante este processo de validação, é possível refinar os hiperparâmetros e limites do modelo, discutidos nas seções 2.3 e 6, respectivamente. Além disso, a validação auxilia na identificação de *overfitting* e *underfitting*, (Shalev-Shwartz, 2014). O *overfitting* ocorre quando o desempenho do modelo é perfeito sobre o conjunto de treino, enquanto não se comporta bem sobre os dados de teste, ou seja, o algoritmo super aprende as relações entre os padrões dos dados de treino mas não tem capacidade de generalização para novas amostras. O *underfitting* representa o cenário em que o modelo não consegue estabelecer relação entre as variáveis e os padrões dos dados no próprio processo de treino.

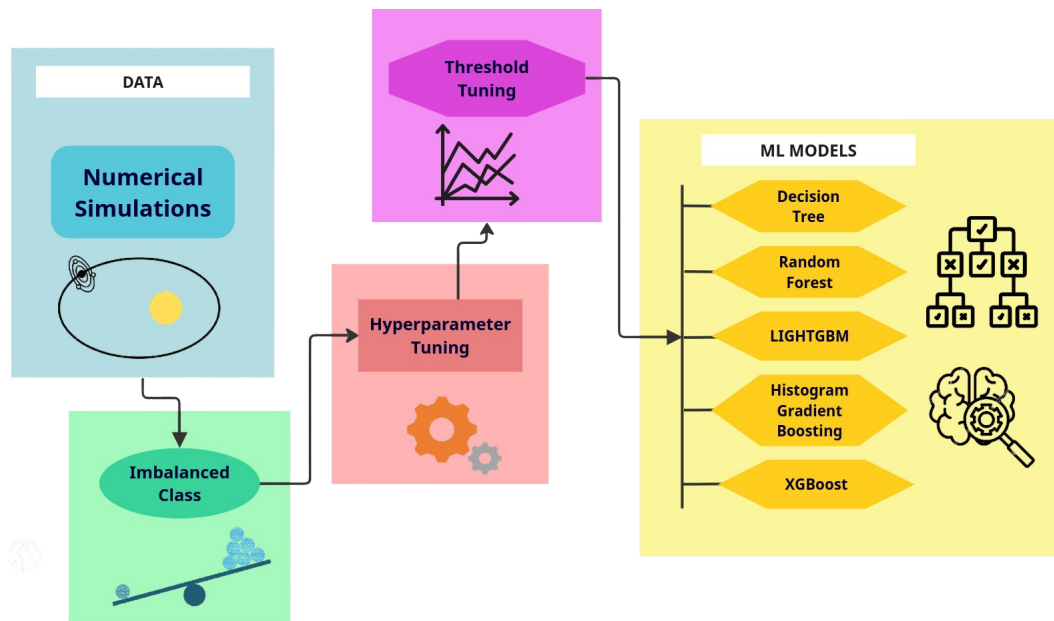
No cenário do uso de *Machine Learning*, temos parâmetros atualizados durante o processo de aprendizado e aqueles definidos em uma etapa anterior ao treino, conhecidos como hiperparâmetros. A avaliação dos melhores hiperparâmetros para o modelo pode ser superajustada aos dados de treinamento, resultando naqueles que traduzem um "modelo ideal", para isso é utilizado o conjunto de validação, entretanto essa abordagem esgota uma parcela do conjunto de dados (Ojala and Garriga, 2009). Para lidar com esse cenário utilizamos a técnica de *cross-validation*, na qual a amostra de treino é separada em subconjuntos que serão treinados e validados separadamente, dispensando a necessidade de um conjunto destinado somente a validação. Dessa forma, o desempenho é relatado pela média dos valores decorrentes do *cross-validation*, estimando o erro verdadeiro do modelo (Fushiki, 2011).

No nosso estudo, o conjunto de treino foi dividido em cinco subconjuntos com o intuito de escolher o melhor modelo, a partir do treino em diferentes configurações e por meio da avaliação da *cross-validation*. Com o desempenho mais adequado, foi feito o retreino utilizando o conjunto de treino em toda a sua extensão e verificando a eficácia a partir do conjunto de teste, cujos valores são desconhecidos para o algoritmo.

As principais etapas desse trabalho são descritas na Figura 1, na qual foi desenvolvido primeiramente um conjunto de dados por meio de simulações numéricas destinado ao

treino, validação e teste do modelo de ML, como detalhado na seção 3.1. Em sequência realizamos a reamostragem dos dados para lidar com as classes desbalanceadas (seção 2.2) e após isso, a busca de hiperparâmetros e *threshold*, cujas definições foram apresentadas nas seções 2.4 e 2.3 e os respectivos resultados na seção 3.2.1. Por fim, testamos cinco diferentes algoritmos a fim de designar aquele que obtivesse o melhor desempenho, exibidos na seção 3.2. Realizamos o mesmo procedimento para o conjunto de dados referente a estabilidade da região da *Sailboat*, seção 4.3, entretanto, nesse caso foi aplicado somente o modelo cujo desempenho foi superior segundo os resultados da seção 3.2.

Figura 1 – Fluxograma representando as etapas deste trabalho.



2.1 Algoritmos

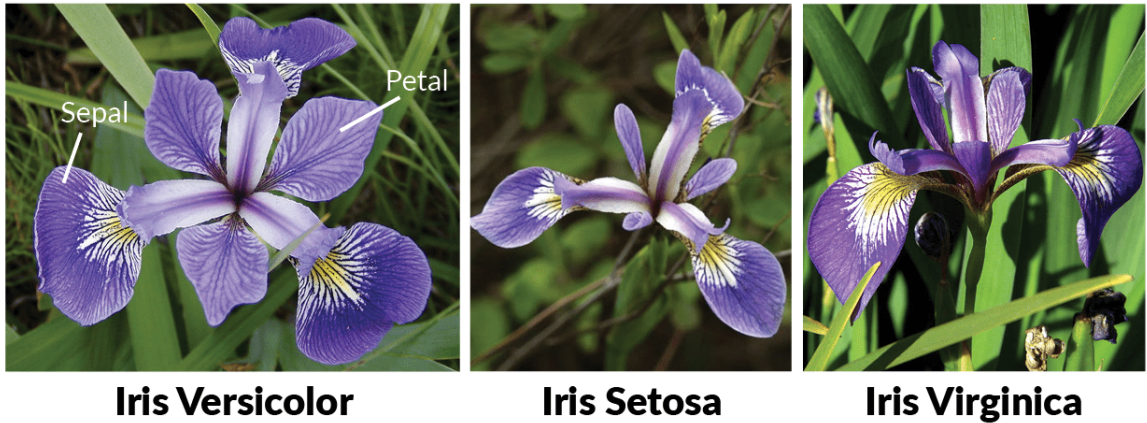
Foram implementados cinco algoritmos de *Machine Learning*, sendo eles: *Decision Tree*, *Random Forest*, *Extreme Gradient Boosting (XGBoost)*, *Histogram Gradient Boosting (HGB)*, e *Light Gradient Boosting Machine (LightGBM)*. Segundo Géron (2022), um conjunto de preditores realiza uma predição melhor que um único preditor. Dessa forma, a maioria dos algoritmos empregados são caracterizados como *ensemble methods*, compostos por um conjunto de árvores de aprendizado. Além disso, também foi executado o uso de Algoritmo Genético para avaliar os melhores algoritmos que se encaixariam no conjunto de dados disponível. É importante destacar que outros algoritmos foram testados, como *Artificial Neural Network (ANN)*, entretanto a eficácia, tempo de treino e gasto computacional não foram compensatórios comparados aos modelos selecionados. Nas seções a seguir será descrito brevemente os conceitos fundamentais de cada um dos métodos citados.

2.1.1 Decision Tree

Decision Tree é um algoritmo de *Machine Learning* capaz de realizar tanto tarefas de classificação quanto de regressão. Foi introduzido pela primeira vez em [Quinlan \(1986\)](#), que caracteriza as árvores de decisão como uma sequência de nós responsáveis por realizar os testes de atributos, os nós geram ramificações para cada possível resultado alcançando as folhas, que representam as classes da amostra. A classificação tem seu início no nó raiz da árvore, onde seu atributo correspondente é testado e o melhor ramo é escolhido como resultado, em sequência o processo é repetido em sub-ramos até que uma folha (classe) seja encontrada ([Mitchell, 1997](#)).

Um exemplo famoso retirado de [Géron \(2022\)](#) para o entendimento de uma árvore de decisão é dado pela verificação da classe de uma flor de Íris, exibida na Figura 2 como referência. Começamos pelo nó raiz onde é designado o atributo de comprimento da pétala da flor, se esse é menor que 2,45 cm, então o próximo nó gera uma folha que prevê a classe como Íris-Setosa. Caso o comprimento seja maior que 2,45cm, surge um nó que não é folha, e, portanto, um novo teste de atributo é feito. O qual investiga a hipótese da largura da pétala ser menor que 1,75 cm, se sim, sua flor é provavelmente uma Íris-Versicolor, se não, é provável uma Íris-Virginica.

Figura 2 – Exemplos de espécies da flor de Iris, classificadas através do algoritmo *Decision Tree*.



Fonte: www.lac.inpe.br/rafael.santos/r.html

Cada nó segue uma regra específica para determinada característica da amostra $x_i < t_i$, onde t_i é um valor limite de um único atributo (x_i) no i -ésimo nó. A escolha do melhor valor para t_i depende da impureza dos dois subconjuntos subsequentes, e pode ser determinado pela característica que reproduz a função de menor custo $J(k, t_i)$ para classificação ([Géron, 2022](#)), sendo essa

$$J(k, t_i) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}, \quad (2.1)$$

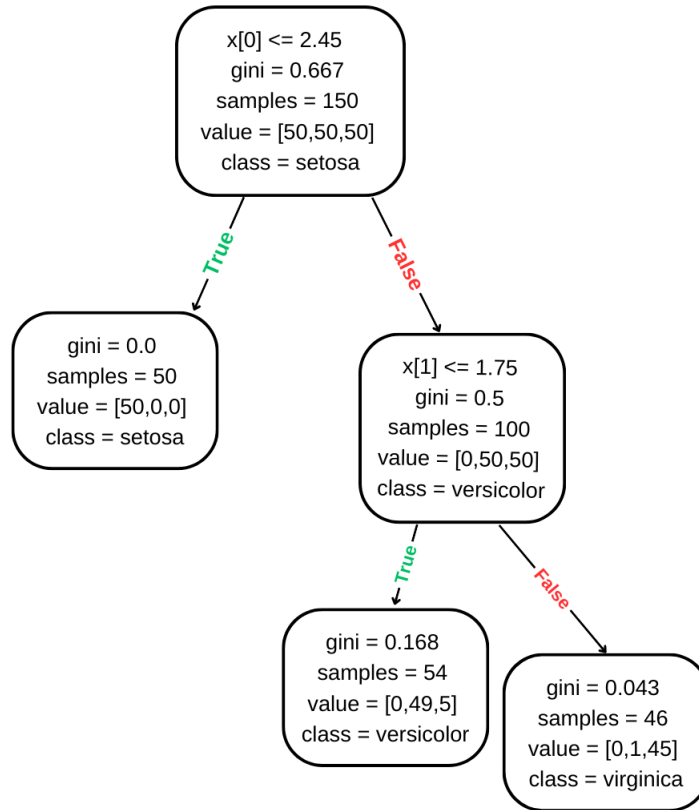
onde m é o número de amostras no nó e G é o valor da Impureza de Gini (equação 2.2) do subconjunto à esquerda e à direita,

$$G_i = 1 - \sum_{k=1}^n p_{i,k}^2, \quad (2.2)$$

$p_{j,k}$ é a probabilidade da classe k .

Voltando ao exemplo da Iris, podemos observar na Figura 3 que no nó raiz ($i = 0$) temos a regra x (comprimento da pétala) sendo verificada, caso ela seja verdadeira, a impureza de Gini resultante é igual a zero e a predição da classe é feita (setosa). Caso seja falsa, uma nova regra é formada no nó $i = 1$ e em sequência novos valores para a impureza são calculados até que seja encontrado o menor valor aceitável, a fim de determinar a classe da amostra.

Figura 3 – *Decision Tree* para a classificação da flor Iris.



2.1.2 Bagging

Como mencionado no início desta seção, um conjunto de preditores é conhecido como um *ensemble method*, muitas vezes o emprego de um grupo de classificadores pode aumentar a acurácia e a robustez do modelo. Uma abordagem consiste em aplicar o mesmo algoritmo como preditor em um *ensemble* sobre diferentes subconjuntos dos dados de

treinamento (Géron, 2022). Caso as amostras forem retiradas com reposição, o método é conhecido como *Bagging* (Breiman, 1996). Uma vez realizada todas as previsões para cada um dos algoritmos, o resultado será dado a partir da votação da classe majoritária para o caso de métodos de classificação e pela média para regressão.

2.1.2.1 Random Forest

Um exemplo de algoritmo que realiza classificação via *Bagging* é o *Random Forest*, introduzido por Breiman (2001). Este método consiste em um conjunto de *Decisions Trees*, no qual cada árvore é construída por um algoritmo A aplicado sobre um subconjunto de treino S e um vetor aleatório θ , onde θ é idêntica e independentemente distribuído em alguma amostra (Shalev-Shwartz, 2014). Dessa maneira, a previsão é obtida pela maioria dos votos sobre as previsões das árvores individuais. A aplicação do *Random Forest* diminui a correlação entre as árvores, reduzindo, portanto, a possibilidade de *overfitting* no treino.

Dado $X = x_1, \dots, x_n$ uma amostra de treinamento com rótulos $Y = y_1, \dots, y_n$, e sendo x' o conjunto desconhecido para o algoritmo, o objetivo do método de *Random Forest* é encontrar a função de predição $f(X)$, determinada pela função de perda $L(Y, f(X))$ e definida para minimizar o valor de perda $E_{XY}(L(Y, f(X)))$.

De acordo com Cutler et al. (2011), uma escolha usual para a função de perda L no caso de classificação é dada pela *zero-one loss*

$$L(Y, f(X)) = I(Y \neq f(X)) = \begin{cases} 0, & \text{se } Y = f(X) \\ 1, & \text{outros casos.} \end{cases} \quad (2.3)$$

A função f é construída conforme os "*base learners*" e, no cenário da classificação, representa as classes preditas,

$$f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} \sum_{j=1}^J I(y = h_j(x)), \quad (2.4)$$

onde $\mathcal{Y}$ é o conjunto de valores possíveis para Y e $h_j(x)$ determina os denominados *base learners*.

2.1.3 Boosting

O método de *Boosting* caracteriza-se como um *ensemble*, no qual a ideia fundamental é agrupar preditores fracos em um único preditor forte (Zhou, 2012). A título de exemplo, considerando uma amostra x , separada em três subconjuntos x_1, x_2 e x_3 , um preditor fraco é aquele que consegue acertar a classificação de dois dos três grupos e erra a classificação de um deles. O objetivo do método de *Boosting* é treinar os preditores fracos sequencialmente, de modo que cada iteração corrige os erros da anterior até que seja alcançado um forte preditor (Géron, 2022).

2.1.3.1 XGBoost

O algoritmo *XGBoost* (*Extreme Gradient Boosting*) é um conjunto de *decision trees* desenvolvido por [Chen and Guestrin \(2016\)](#), construído através do conceito de *Gradient-Boosted Tree* (GBT) de [Friedman \(2001\)](#) com a intenção de ampliar o ganho de velocidade. O mecanismo de *Gradient Boosting* funciona adicionando preditores sequencialmente a um conjunto com o intuito de corrigir o erro do anterior. Entretanto, a correção é feita a partir de erros residuais, os quais representam a diferença entre os valores reais e as classes preditas, eles são utilizados para atualizar o modelo iterativamente ([Géron, 2022](#)). O processo realizado pelo *XGBoost* é uma implementação eficiente do GBT, a cada iteração o algoritmo divide subconjuntos que podem ser utilizados para construir árvores em paralelo, otimizando o processo de iteração. Além disso, o método incorpora técnicas de regularização para evitar o *overfitting*.

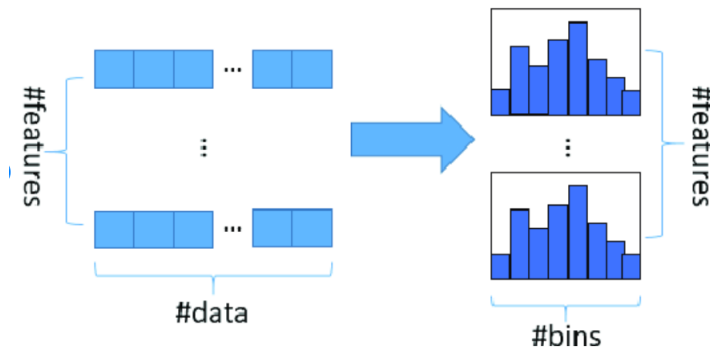
2.1.3.2 LightGBM

O algoritmo *LightGBM* é um tipo de *Gradient Boosting Decision Tree* (GBDT) que surgiu em 2016, como parte do projeto da Microsoft *Distributed Machine Learning Toolkit* (DMTK) ([Ke et al., 2017](#)). Ao lidar com grandes conjuntos de dados, o GBDT pode possuir uma eficiência insatisfatória, uma vez que para cada atributo se percorre toda a amostra para estimar o ganho de informação de todos os possíveis pontos de divisão. Com a intenção de resolver esse problema, [Ke et al. \(2017\)](#) introduziram as técnicas de *Gradient-Based One-Side Sampling* (GOSS) e *Exclusive Feature Bundling* (EFB). Com o método GOSS é possível excluir porções dos dados, de forma que uma estimativa bastante precisa do ganho de informação é feita utilizando apenas uma parcela da amostra. Já o EFB agrupa características mutuamente exclusivas, que raramente assumem valores diferentes de zero simultaneamente, para reduzir o número de atributos. O *LightGBM* é, portanto, definido como uma implementação do GBT com a adição dos métodos de GOSS e EFB ([Ke et al., 2017](#)).

2.1.3.3 HGB

Assim como surgiu o *LightGBM*, o *Histogram Gradient Boosting* (HGB) foi desenvolvido com o intuito de aumentar a eficiência e reduzir o tempo e gasto computacional exigido por modelos de *gradient boosting*. Dessa forma, como o nome sugere, o algoritmo faz uso de histogramas para agrupar os dados contínuos em um número constante de intervalos, conforme mostrado na Figura 4 ([Ibrahim et al., 2023](#)).

Figura 4 – Algoritmo de histograma.



Fonte: Ibrahim et al. (2023)

A implementação do HGB reduz os valores das características da amostra, consequentemente as divisões nas árvores diminuem, resultando na aceleração e melhora no processo das árvores de decisão do modelo.

2.1.4 Algoritmo Genético

O Algoritmo Genético é inspirado pelos princípios biológicos da seleção natural e da evolução Darwiniana das espécies. Dessa forma, o conjunto de dados é submetido a um método de otimização que analisa parâmetros de diferentes modelos de *Machine Learning*. O intuito é testar os métodos a cada geração, utilizando um processo de busca paralela e adaptativa baseado no princípio de sobrevivência dos mais aptos e na reprodução.

O método funciona em um ciclo no qual ocorre a seleção para reprodução dos indivíduos mais aptos da população atual, os mesmos são emparelhados e os genes combinados para criar novos indivíduos. Por fim, passam pelo processo de mutação, onde uma parcela dos genes é mutada, criando diversidade na população (Goldberg, 1989). Nesse contexto, os indivíduos se referem aos modelos de ML, os genes aos hiperparâmetros do modelo e a geração, a iteração realizada pelo Algoritmo Genético. A busca inteligente retorna o algoritmo de desempenho superior e consecutivos hiperparâmetros que melhor se adequam a amostra testada (Chen et al., 2004).

Em estudos anteriores o uso de Algoritmo Genético demonstrou ser uma técnica útil para identificação dos melhores algoritmos a serem explorados para determinado problema. No trabalho de Carruba et al. (2021b), há a aplicação de Algoritmo Genético para selecionar o modelo e o conjunto de treino mais adequado para a identificação de asteroides pertencentes às ressonâncias seculares z_1 e z_2 . Carruba et al. (2021b) implementaram a biblioteca *Tpot* do *Python* (Le et al. (2019); Olson et al. (2016)), utilizando 5 gerações, 20 populações e 5 subgrupos para a validação cruzada. Como resultado, foram obtidos três modelos de *ensemble* diferentes: *Random Forest*, *Extra Trees* e *Gradient Boosting*. A fim de verificar os algoritmos selecionados, aplicaram a técnica de *pooling sample*. Neste trabalho,

utilizamos a biblioteca *Tpot* com 6 gerações e 30 populações, realizando a aplicação do Algoritmo Genético pelo menos 10 vezes, a fim de verificar se o resultado seria coerente ou aleatório.

2.2 Classes desbalanceadas

Muitos algoritmos de *Machine Learning* não são efetivos na generalização de dados desbalanceados, isto é, aqueles que possuem mais pontos pertencentes a uma classe do que a outra. Dessa forma, amostras desbalanceadas levam a vieses na classificação e empobrecem o desempenho do modelo. A fim de solucionar o problema são empregadas técnicas de reamostragem como *undersampling* e *oversampling*. *Undersampling* é um método que remove dados da classe majoritária, entretanto, essa aplicação pode apresentar uma desvantagem ao passo que informações são perdidas. No caso do *oversampling*, uma fração da classe minoritária é acrescida até que a amostra esteja equilibrada.

O método SMOTE é conhecido por, recentemente, ter ganhado grande espaço no emprego de técnicas de *oversampling* (Zheng et al., 2015). Ademais, outros métodos podem ser citados, tais como: *Random Oversampling*, *Borderline SMOTE* e *ADASYN*. Neste trabalho, avaliamos as quatro técnicas de *oversampling* mencionadas, uma vez que o conjunto de dados gerado via simulações numéricas apresentou classes desequilibradas.

O *Random Oversampling* é caracterizado por replicar aleatoriamente dados dentro da amostra com o intuito de nivelar a distribuição de classes. No entanto, essa abordagem pode levar ao *overfitting*, uma solução para isso é apresentada pelo *Synthetic Minority Oversampling Technique* (SMOTE). A técnica de *oversampling* sobre a classe minoritária foi proposta por Chawla et al. (2002), onde são gerados exemplos ‘sintéticos’ para a reamostragem. O *oversampling* é feito a partir de cada amostra da classe minoritária, introduzindo exemplos sintéticos ao longo da linha de segmentos que unem todos os k vizinhos mais próximos.

O *Borderline-SMOTE*, identifica as fronteiras de cada classe e evita utilizar os pontos fora da curva (pontos de ruído) da classe minoritária para realizar a reamostragem da mesma (Han et al., 2005). Por fim, o método *Adaptive Synthetic* (ADASYN) gera amostras sintéticas para a classe minoritária considerando a distribuição ponderada para essa classe específica.

Segundo Carruba et al. (2023a), para ser considerado um caso grave de desequilíbrio, a razão entre a classe minoritária e a majoritária deve ser igual ou maior que 1:100. No cenário da aplicação de ML para o estudo de asteroides ressonantes há grandes desafios, ao passo que há uma grande razão de desequilíbrio entre o número de asteroides em órbitas librando comparado ao resto, chegando a 1:270. Entretanto, verificamos na seção 3.1 que a proporção para os dados analisados neste trabalho é de 1:10, já para a amostra

apresentada na seção 4.5 temos um caso com razão de desequilíbrio próximo de 1:100. Carruba et al. (2023a) concluíram que métodos de otimização sensíveis ao custo devem sempre ser considerados quando viáveis, uma vez que podem melhorar o desempenho do modelo ou não ter efeito significativo. No entanto, métodos de reamostragem para classificação são recomendados para conjuntos de dados com grave desequilíbrio.

2.3 Busca de hiperparâmetros

Existem dois tipos de parâmetros presentes na modelagem de *Machine Learning*, aqueles atualizados durante o processo de aprendizado (como os pesos em alguns algoritmos) e aqueles definidos em uma etapa anterior ao treino, identificados como hiperparâmetros (Yang and Shami, 2020). A arquitetura do modelo é caracterizada pela escolha de hiperparâmetros, o que leva ao desempenho do modelo depender diretamente deles (Weerts et al., 2020).

Com o intuito de construir um modelo de ML eficaz é necessário realizar a variação de uma gama de possibilidades de hiperparâmetros. A seleção pode advir do padrão do algoritmo utilizado, ser definida manualmente pelo usuário ou mediante um processo de ajuste (Probst et al., 2018), denominado *hyperparameter tuning*. O ajuste de hiperparâmetros procura minimizar o erro esperado para a generalização do algoritmo e o induz sobre um espaço de busca de hiperparâmetros com possíveis configurações a serem testadas, usualmente avaliando-as sobre um conjunto de teste separado ou realizando reamostragem (Bischl et al., 2012). Dessa maneira, a técnica de *hyperparameter tuning* pode ser desenvolvida de duas principais maneiras: *Grid Search* e *Random Search*.

O método de *Grid Search* (GS), é uma das formas mais comuns utilizadas para explorar as configurações de hiperparâmetros. O GS efetua todas as possíveis combinações de hiperparâmetros, em vista disso é considerada uma técnica de busca exaustiva (Injadat et al., 2020). As combinações são feitas através do produto Cartesiano entre conjuntos finitos definidos pelo usuário (Hutter et al., 2019). O processo para empregar a técnica de GS é caracterizado por: iniciar o método com um amplo espaço de pesquisa, em seguida limitar o espaço conforme os resultados anteriores que obtiveram o melhor desempenho e, por fim, repetir esse procedimento até que a configuração ideal seja alcançada.

A implementação do *Grid Search* é simples e eficaz, entretanto, apresenta uma grande desvantagem no cenário de um espaço de configurações de hiperparâmetros de alta dimensão. Uma vez que, conforme o número de hiperparâmetros cresce, o número de avaliações aumenta exponencialmente, conhecido como a maldição da dimensionalidade (Claesen et al., 2014).

A técnica de *Random Search* (RS) é similar ao GS, no entanto, seleciona combinações para avaliar. O método de RS escolhe aleatoriamente um número pré-definido de amostras

com base no limite inferior e superior dos valores de hiperparâmetros definidos pelo usuário. Diferente do *Grid Search*, o RS utiliza um número fixo de combinações, o que reduz a complexidade computacional. Dessa forma, é uma técnica útil quando o espaço de hiperparâmetros possui alta dimensão (Bergstra and Bengio, 2012).

2.3.1 Hiperparâmetros implementados

Neste estudo, implementamos a técnica de *Grid Search* nos algoritmos avaliados, com exceção do *Random Forest*, no qual o resultado foi superior com a aplicação do *Random Search*. Como aplicamos algoritmos baseados em árvores de decisão, os hiperparâmetros disponíveis são similares, havendo discrepância principalmente entre a implementação de *bagging* e *boosting*. A seguir estão listados e definidos todos os hiperparâmetros empregados para cada um dos algoritmos:

1. **max_depth**: controla o nível de profundidade que a árvore pode crescer (Pedregosa et al., 2011). Quanto maior a profundidade, mais padrões do conjunto de treinamento serão captados, entretanto, existe um limite no qual o algoritmo pode aprender excessivamente levando ao *overfitting*. O **max_depth** possui números inteiros e positivos como valores de entrada ou 'None', os quais irão definir o nível da profundidade (caso **max_depth** = 3, a profundidade vai até o nível 3). Caracterizado como um hiperparâmetro essencial para árvores de decisão, foi implementado na análise dos algoritmos: *Decision Tree*, *Random Forest*, *XGBoost* e *HGB*.
2. **min_samples_split**: representa o número mínimo de amostras para dividir um nó. Como discutido na seção 2.1.1, o nó representa a seção da árvore de decisão onde um atributo do conjunto de dados é avaliado, um nó pode ser dividido em novos nós de acordo com certas condições a serem avaliadas. Para o **min_samples_split**, caso o número de amostras em um nó seja menor que o valor definido pelo hiperparâmetro, não haverá uma divisão e o nó se tornará um nó folha, ou seja, a decisão da classificação é tomada e a árvore é encerrada. Este hiperparâmetro aceita como entrada números inteiros, interpretados como o número mínimo de amostras, além disso, *floats* também são considerados, nesse caso a entrada é uma porcentagem da amostra (Pedregosa et al., 2011). Entre os cinco algoritmos testados, esse hiperparâmetro foi utilizado para *Decision Tree* e *Random Forest*.
3. **min_samples_leaf** : o hiperparâmetro **min_samples_leaf** define o número mínimo de amostras necessário para compor um nó folha. Dessa forma, caso o número indicado pelo **min_samples_leaf** esteja presente em um nó, então mais nenhuma divisão será realizada e o nó será designado como folha. Isso permite que as divisões não sejam realizadas indefinidamente (Pedregosa et al., 2011). As opções de entrada

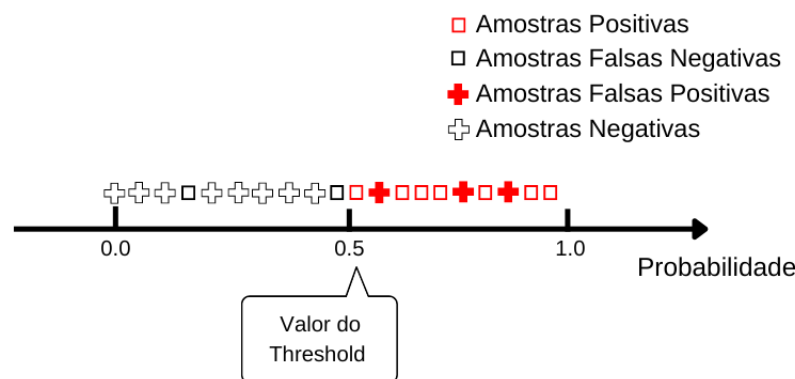
para o `min_samples_leaf` são números inteiros ou *floats*. Esse hiperparâmetro foi implementado nos algoritmos *Decision Tree* e *Random Forest*.

4. **max_features**: define o número máximo de atributos a serem considerados ao procurar a melhor divisão do nó na árvore de decisão. As entradas podem ser: números inteiros, *float*, *sqrt*, *log2* e *None*, onde a opção *sqrt* permite que o algoritmo considere apenas a raiz quadrada do número total de atributos para cada divisão e o *log2* faz com que o algoritmo considere o logaritmo de base 2 de um número total de características para cada divisão. o hiperparâmetro **max_features** foi aplicado nos algoritmos *Decision Tree* e *Random Forest*.
5. **n_estimators**: como o *Random Forest* é caracterizado por um conjunto de árvores de decisão, é necessário definir o número de árvores presentes na floresta, dessa forma o **n_estimators** determina esse limite. A entrada para esse hiperparâmetro é representada por números inteiros (Pedregosa et al., 2011).
6. **booster**: o hiperparâmetro **booster** indica qual *booster* será utilizado, isto é, o tipo de modelo a ser executado em cada iteração. Possui três opções: *gbtree*, *gblinear* ou *dart*. Os hiperparâmetros *gbtree* e *dart* empregam modelos baseados em árvore, enquanto o *gblinear* usa modelos lineares. Esse hiperparâmetro foi aplicado apenas nos algoritmos de *boosting*: *XGBoost* e *LightGBM*.
7. **learning_rate**: representa a taxa de aprendizado do *boosting*, atuando como um fator multiplicativo nos pesos dos atributos das árvores. O **learning_rate** pode reduzir os pesos dos atributos a cada iteração, tornando o modelo mais robusto, auxiliando ainda para evitar o *overfitting*. A entrada para esse hiperparâmetro é definida por *floats*. Esse hiperparâmetro foi aplicado em todos os modelo de *boosting* avaliados, entretanto é válido comentar que no caso do *XGBoost*, o **learning_rate** recebe o nome de *eta*.
8. **scale_pos_weight**: controla o equilíbrio entre os pesos positivos e negativos, sendo útil no cenário de classes desbalanceadas. Um valor maior que 0 deve ser utilizado no caso em que o desequilíbrio entre as classes é alto, uma vez que auxilia na convergência mais rápida. Esse hiperparâmetro foi implementado somente no algoritmo *XGBoost*.
9. **max_iter**: define o número máximo de iterações no processo de *boosting*, ou seja, o número máximo de árvores para classificação binária. Para classificação multi classe, é construído um número de árvores iguais à **n_classes** por iteração. Esse hiperparâmetro foi aplicado no algoritmo *HGB*.

2.4 Busca de *threshold*

O *threshold* no cenário da classificação binária determina como o modelo irá classificar uma amostra em uma das duas classes. O valor do *threshold* pode ser modificado, o que influencia diretamente no desempenho e eficácia do modelo. Em geral, o *threshold* padrão dos algoritmos é igual a 0,5, dessa forma, se um ponto do conjunto de dados pertencente a classe positiva possui probabilidade $\geq 0,5$, ele é classificado como positivo conforme exibido na Figura 5, caso contrário é definido como negativo. Além disso, se as amostras positivas são classificadas como negativas são consideradas falsas negativas, e quando as negativas são preditas positivas são definidas como falsas positivas.

Figura 5 – Distribuição das classes de acordo com o *threshold*.



No caso de classes desbalanceadas, utilizar o *threshold* igual a 0,5 pode ser inadequado, uma vez que não há uma distribuição igual para cada classe (Zou et al., 2016). A busca para um valor ótimo para o *threshold* pode ser realizada através da curva *The Receiver Operating Characteristic* (ROC), a qual representa graficamente a taxa de Verdadeiros Positivos (*True Positive*) em relação a de Falsos Positivos (*False Positive*) (Figura 6) de acordo com a probabilidade, entre 0 e 1, do *threshold* (Géron, 2022). O termo Verdadeiro Positivo se refere aos dados pertencentes a classe positiva que foram classificados corretamente, enquanto o Falso Positivo aqueles que compõem a classe negativa e foram preditas incorretamente.

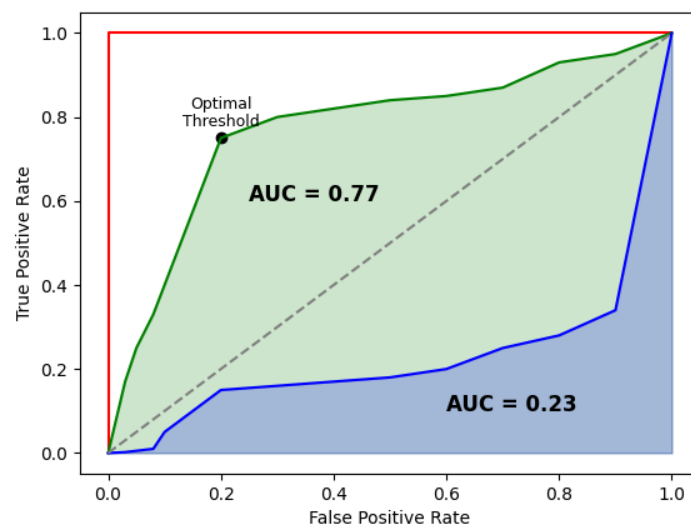
A curva ROC sempre é iniciada no ponto (0,0), onde o *threshold* é igual a 1, ou seja, todos pontos do conjunto de dados foram preditos como pertencentes a classe negativa. O final da curva é indicado no ponto (1,1), com *threshold* 0, indicando que todos os pontos foram classificados como da classe positiva. A área sob a curva ROC é definida como *Area Under the Curve* (AUC), calculando a habilidade do modelo em diferenciar as classes.

A Figura 6 exibe um exemplo de uma curva ROC, onde as curvas vermelha, verde e azul identificam a performance de três modelos diferentes. Um modelo perfeito (curva vermelha), é aquele que sempre prevê corretamente 100% da classe positiva, dessa forma a

área sob a curva será 1. Bons modelos irão apresentar um valor AUC próximo a 1. No caso apresentado na figura 6, a curva verde possui um AUC igual a 0,77, representando que há uma probabilidade de sucesso de 77% das classes positivas e negativas serem distinguidas. A linha tracejada possui um AUC equivalente a 0,5 e o modelo não é capaz de diferenciar as classes, haverá, portanto, uma predição aleatória. Já a curva azul, representa um AUC de 0,23, significando que a maior parte das classes positivas da amostra foram preditas como negativas.

O valor ótimo para o *threshold* (ponto preto na Figura 6) é definido como o ponto mais próximo ao canto superior esquerdo da curva ROC, ponto (0,1), ou aquele que maximiza a distância da linha de referência (linha tracejada).

Figura 6 – Representação da curva ROC de três modelos hipotéticos. A curva vermelha está associada a um modelo ideal, as curvas verde e azul são referentes a classificadores bons e ruins, respectivamente. A linha tracejada é a linha de referência.

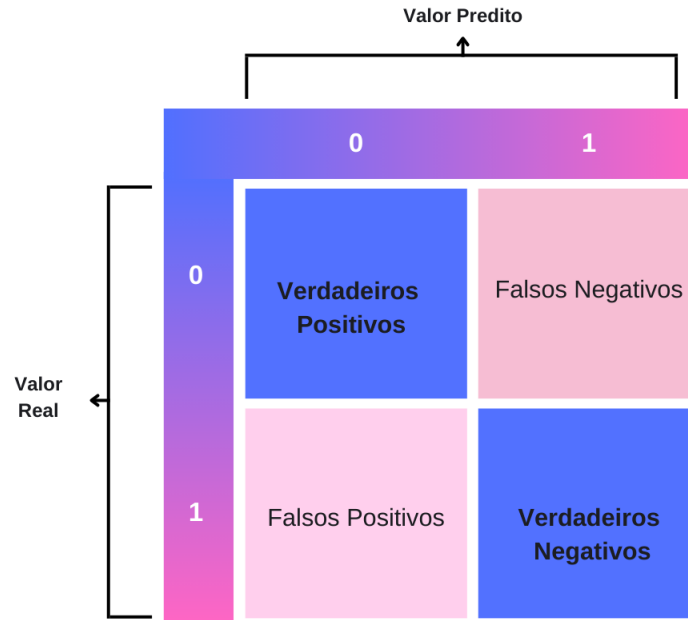


2.5 Métricas

Métricas de avaliação são empregadas a fim de mensurar o desempenho dos modelos de *Machine Learning*. Desse modo, esses valores são calculados a partir da matriz da confusão, uma tabela estatística que demonstra os resultados da previsão feita pelo algoritmo sobre o conjunto de teste, indicando quanto da amostra avaliada foi classificada correta e incorretamente (Stehman, 1997). Conforme exibido na Figura 7, as linhas da matriz da confusão representam os valores reais do conjunto de dados, enquanto as colunas retratam os valores preditos. No exemplo da Figura 7, temos uma classificação binária, na qual a matriz da confusão é dividida em quatro classes: Verdadeiro Positivo e Verdadeiro Negativo, onde o algoritmo previu corretamente tanto a classe positiva quanto a negativa,

Falso Positivo e Falso Negativo, em que as classes positivas e negativas foram preditas incorretamente.

Figura 7 – Matriz da confusão para uma classe binária.



Mediante o resultado das quatro categorias da matriz da confusão citadas é possível calcular as métricas: *accuracy*, *precision*, *recall* e *f1-score*. A *accuracy* mede a proporção do conjunto de dados predita corretamente,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.5)$$

por meio desse valor, avaliamos quão eficaz o modelo é em determinar corretamente cada classe em relação ao conjunto. A *recall* analisa quanto da amostra pertencente à classe positiva são corretamente preditas como positivas,

$$Recall = \frac{TP}{TP + FN}. \quad (2.6)$$

Já a *precision* calcula a proporção do conjunto predito como positivo que é realmente positivo,

$$Precision = \frac{TP}{TP + FP} \quad (2.7)$$

e, por fim, a *f1-score* é uma métrica que verifica a contribuição da *precision* e da *recall* combinando ambas por meio de uma média harmônica,

$$F1 = 2 \frac{precision \times recall}{precision + recall}. \quad (2.8)$$

3 Regiões de estabilidade ao redor de planetas

O número de exoplanetas cresce a cada ano e o estudo da estabilidade de sistemas planetários externos nos leva a questão se eles podem hospedar exoluas ou exoanéis. Os trabalhos que abordam esse tema são usualmente realizados numericamente mediante um grande conjunto, computacionalmente caro, de simulações de N-corpos. Neste estudo, nós propomos a aplicação de *Machine Learning* para prever mapas de estabilidade de regiões de órbitas do tipo-S ao redor de planetas.

Para identificar as regiões de estabilidade, foram realizadas simulações de sistemas compostos por uma estrela, um planeta e uma partícula, onde cada um deles possuía 9 atributos representando as condições iniciais do planeta, da partícula e a razão de massa do sistema. Esse capítulo descreve brevemente as simulações numéricas desenvolvidas, assim como a atuação dos algoritmos de ML.

3.1 Simulações Numéricas

Com o intuito de investigar a estabilidade orbital de um sistema de três corpos (estrela, planeta e partícula), foram realizadas simulações numéricas adimensionais através do integrador Rebound/IAS15 ([Rein and Spiegel, 2015](#)).

A fim de estabelecer a extensão da variação dos parâmetros trabalhados, analisamos inicialmente o levantamento disponível em [Schneider et al. \(2011\)](#), o qual contém 4.175 exoplanetas confirmados. Uma vez que a razão de massa do sistema (μ) é dada por

$$\mu = \frac{m_p}{M_s + m_p} \quad (3.1)$$

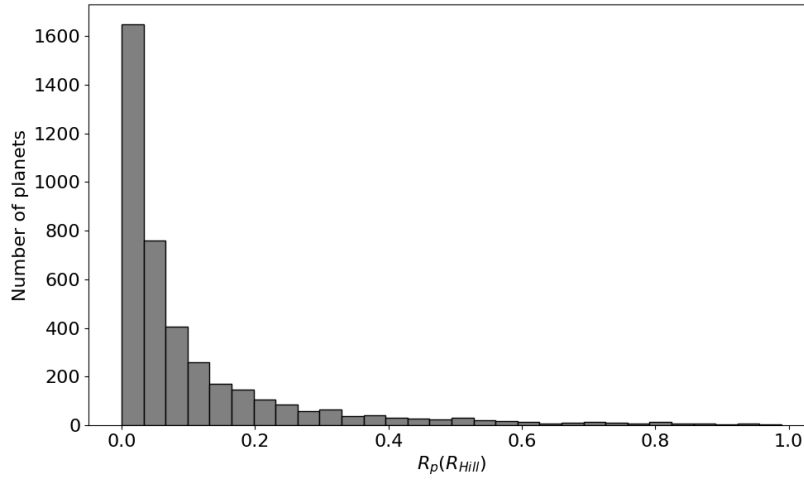
onde m_p é a massa do planeta e M_s a massa da estrela. Observamos que cerca de 4.150 (99,6%) dos exoplanetas possuem uma razão de massa equivalente à $\mu \leq 10^{-2}$. Portanto, μ foi definido na simulação entre 10^{-5} e 10^{-2} . Além disso, a massa do planeta foi estabelecida como $m_p = \mu$, e a massa da estrela como $M_s = 1 - \mu$.

A esfera de Hill é determinada como a região onde um corpo tem maior influência gravitacional sobre a presença de perturbações gravitacionais de outros corpos ([Hamilton and Burns, 1992](#)). Dessa forma, sendo o semieixo maior do planeta igual à 1, o raio de colisão r_c pode ser calculado em função do raio de Hill via,

$$r_c = 0,05(1 - e_p) \sqrt[3]{\frac{\mu}{3}} \quad (3.2)$$

sendo que o raio de colisão é equivalente a 5% do raio de Hill, representando 55,6% dos raios planetários conhecidos como vemos na Figura 8.

Figura 8 – Histograma da distribuição dos planetas confirmados em unidades de seus raios de Hill baseado em um levantamento composto por 4.150 planetas diferentes (Schneider et al., 2011).



Como exibido na Tabela 1, a excentricidade (e_p) e a anomalia verdadeira (f_p) do planeta foram distribuídos aleatória e uniformemente nos intervalos $e_p = [0 - 0,99]$ e $f_p = [0^\circ - 360^\circ]$. Os elementos orbitais restantes para o planeta foram definidos como zero. Enquanto isso, a partícula foi modelada em uma órbita do tipo-S ao redor do planeta e perturbada gravitacionalmente pela estrela. Assim, o semieixo maior das partículas assume valores de maneira aleatória e uniforme no intervalo de $1,1r_c$ à $1r_{Hill}$, a excentricidade de 0 à 0,99, a inclinação de 0° e 180° , por fim, o argumento do pericentro, a longitude do nodo ascendente e a anomalia verdadeira entre 0° e 360° .

Tabela 1 – Grade de elementos orbitais.

Elemento orbital	Planeta	Partícula
a	1	$1,1r_c - 1r_{Hill}$
e	0 - 0,99	0 - 0,99
i ($^\circ$)	0	0 - 180
f ($^\circ$)	0 - 360	0 - 360
ϖ, Ω ($^\circ$)	0	0 - 360

As simulações numéricas foram integradas em um intervalo de tempo equivalente à 10^4 períodos orbitais do planeta. No total, foram realizadas 10^5 simulações, nas quais 11,83% das partículas permaneceram estáveis durante toda integração, enquanto 88,17% foram definidas como instáveis. Entre as partículas instáveis, 47,22% da amostra colidiu com o planeta, enquanto 40,95% foi ejetada do sistema. No modelo, uma partícula ejetada era aquela que possuía $a > 1$ ou atingia uma órbita hiperbólica ($e \geq 1$).

Os resultados das simulações numéricas foram empregados para construir um conjunto de treino para os algoritmos de *Machine Learning*. A amostra de dados foi dividida conforme as condições iniciais do sistema, sendo elas 9 características: a razão de

massa do sistema, o semieixo maior, a inclinação, o argumento do pericentro, longitude do nodo ascendente da partícula, excentricidade e anomalia verdadeira do planeta e da partícula. Os resultados numéricos para cada condição inicial foram rotulados como estáveis e instáveis, oferecendo uma correspondência a ser aprendida e posteriormente predita pelo algoritmo de ML.

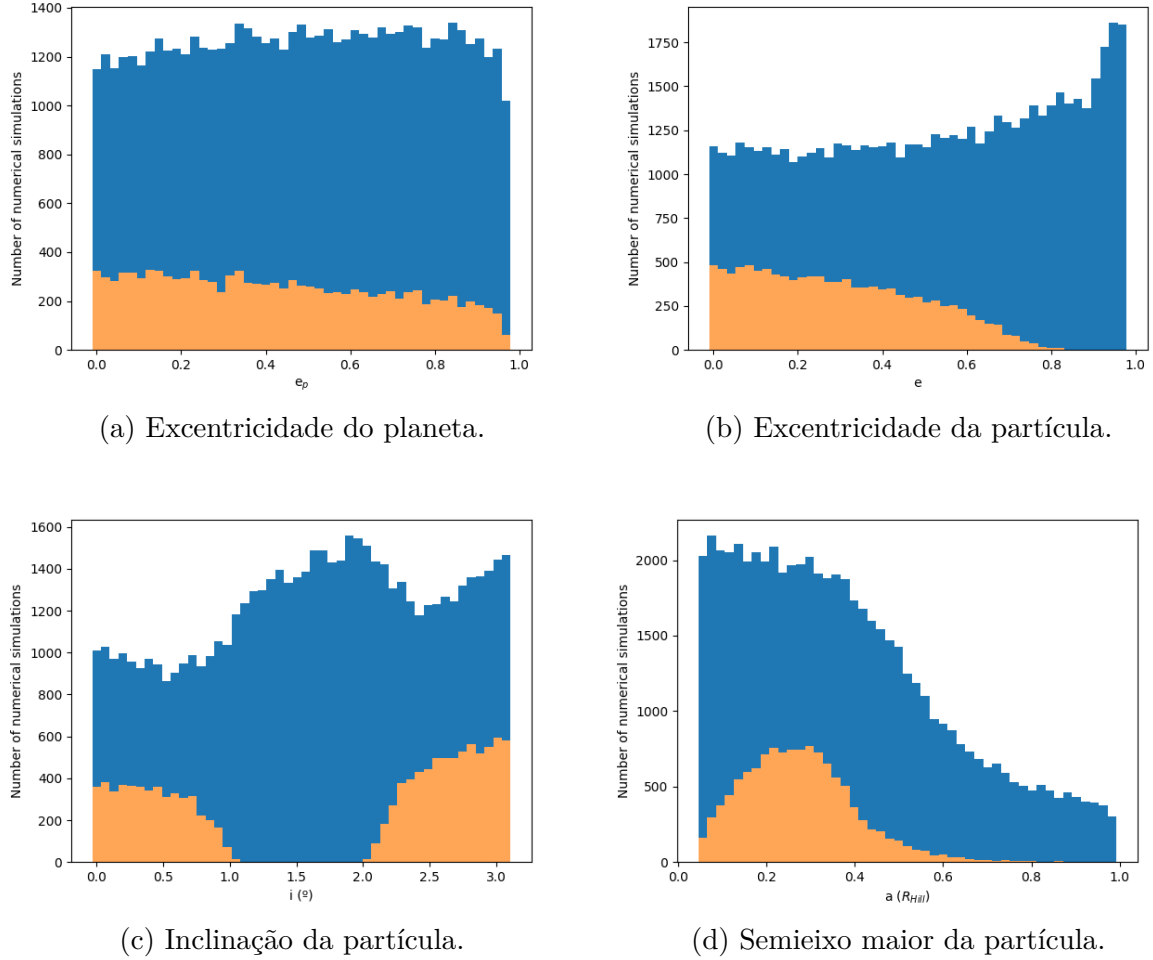
A fim de demonstrar a proporção de algumas características do conjunto de dados, realizamos a análise presente na Figura 9. A Figura 9a mostra o histograma do número de simulações numéricas em relação à excentricidade do planeta, já a Figura 9b exibe a excentricidade da partícula. Observa-se que a quantidade de sistemas instáveis cresce com o aumento da excentricidade, uma vez que órbitas excêntricas experimentam fortes perturbações gravitacionais em seu pericentro, aumentando o número de partículas colididas. Além disso, praticamente não existem partículas estáveis quando a excentricidade inicial é $e > 0,8$.

A inclinação das partículas em relação ao plano do planeta também exerce grande influência no tamanho da região estável. A Figura 9c mostra que órbitas retrógradas tendem a ter mais sistemas estáveis quando comparadas a órbitas prógradas, e praticamente não há sistemas estáveis para inclinações entre 80° e 110° .

A Figura 9d exibe a distribuição das partículas estáveis e instáveis em relação ao semieixo maior das partículas em unidades do raio de Hill do planeta. Próximo ao planeta, há uma abundância de partículas colidindo, dessa forma para partículas inicialmente posicionadas distantes do planeta há o aumento do número de sistemas estáveis, atingindo um pico em $0,3 r_{\text{Hill}}$. Após o pico, há um declínio e poucos sistemas permanecem estáveis próximos a $0,7 r_{\text{Hill}}$, onde a maioria das partículas começa a ser ejetada.

Outros atributos como a razão de massa do sistema, o argumento do pericentro, o argumento do nodo ascendente e a anomalia verdadeira apresentaram uma distribuição regular.

Figura 9 – Histograma da distribuição dos sintemas instáveis (azul) e estáveis (laranja) em relação aos valores da excentricidade do planeta, da excentricidade, da inclinação e do semieixo maior da partícula.



3.2 Performance

Através dos resultados obtidos pelas simulações numéricas foi possível treinar e validar os algoritmos de *Machine Learning*. Nesse contexto, foram aplicados modelos de classificação seguindo uma abordagem binária, na qual a partícula seria prevista como estável ou instável. A amostra final composta com 10^5 simulações foi dividida em três sub-conjuntos: 80% para treino e validação e 20% para teste. A melhor performance foi obtida utilizando o *XGBoost*, sem técnicas de reamostragem e com um *threshold* igual a 0,4.

3.2.1 Busca de hiperparâmetros e threshold

A busca de hiperparâmetros foi empregada através do uso de *Grid Search* para os cinco algoritmos avaliados, com exceção do *Random Forest*, no qual foi aplicado *Random*

Search. Como discutido na seção 2.3, o GS realiza o produto Cartesiano entre conjuntos de valores finitos definidos pelo usuário. As grades de configurações implementadas neste estudo estão descritas nas Tabelas 2, 3, 4, 5, 6, os hiperparâmetros aplicados foram detalhados na seção 2.3.1. Além disso, verificamos os hiperparâmetros selecionados como aqueles que melhor otimizaram os algoritmos.

Tabela 2 – Resultados para o *hyperparameter tuning*, explorados através de *Grid Search* para o *Decision Tree*.

Hiperparâmetros	Espaço de busca	Valores selecionados
<code>max_depth</code>	None, 3, 4, 5	None
<code>min_samples_split_options</code>	2, 3, 4	4
<code>min_samples_leaf_options</code>	1, 2, 3	3
<code>max_features_options</code>	None, sqrt, log2	None

Tabela 3 – Resultados para o *hyperparameter tuning*, explorados através de *Grid Search* para o *Random Forest*.

Hiperparâmetros	Espaço de busca	Valores selecionados
<code>n_estimators_options</code>	100, 200	200
<code>max_depth</code>	None, 3, 4, 5	None
<code>min_samples_split_options</code>	2, 3, 4	3
<code>min_samples_leaf_options</code>	1, 2, 3	2
<code>max_features_options</code>	None, sqrt, log2	None

Tabela 4 – Resultados para o *hyperparameter tuning*, explorados através de *Grid Search* para o *LightGBM*.

Hiperparâmetros	Espaço de busca	Valores selecionados
<code>boosting_options</code>	gbdt, dart	gbdt
<code>learning_rate_options</code>	0.1, 0.2, 0.3, 0.5, 0.6, 0.7	0.2

Tabela 5 – Resultados para o *hyperparameter tuning*, explorados através de *Grid Search* para o *HGB*.

Hiperparâmetros	Espaço de busca	Valores selecionados
<code>max_iter_options</code>	100, 200	200
<code>learning_rate_options</code>	0.1, 0.2, 0.3, 0.4	0.2
<code>max_depth_options</code>	None, 5, 6, 7	None

Tabela 6 – Resultados para o *hyperparameter tuning*, obtidos através de *Grid Search* para o *XGBoost*.

Hiperparâmetros	Espaço de busca	Valores selecionados
<code>booster</code>	gbtree, dart	gbtree
<code>eta</code>	0.2, 0.3, 0.4, 0.5, 0.6	0.2
<code>scale_pos_weight</code>	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.	0.4
<code>max_depth</code>	1, 2, 3, 4, 5, 6, 7, 8, 9	8

Através da busca realizada para o *XGBoost*, podemos observar como a mudança dos valores dos hiperparâmetros afeta o desempenho do modelo nas Figuras 10 e 11. Apresentamos nos gráficos a variação das métricas *accuracy* (vermelho), *recall* e *precision* para a classe estável (verde e roxo), e *recall* e *precision* para a classe instável (azul e rosa). Nessa análise, apenas o valor do hiperparâmetro indicado foi modificado e o restante permaneceu no padrão do algoritmo. Para o caso do hiperparâmetro *eta*, os valores para a classe instável são aproximadamente constantes em torno de 99%, enquanto para a classe estável nota-se uma pequena variação com valores máximos entre 0,1 e 0,2. Como discutido na seção 2.3, o *eta* controla a taxa de aprendizado do modelo, portanto, valores pequenos para esse hiperparâmetro resultam em uma velocidade de treinamento lenta, mas também um modelo mais robusto que realiza pequenos ajustes a cada iteração.

O comportamento para a variação do *scale_pos_weight* e *max_depth* é similar ao observado para o *eta*, onde para o *scale_pos_weight* os pontos máximos para a classe estável se encontram em torno de 0,4 e 0,7, e para a classe instável as porcentagens permanecem constantes ao redor de 99%. Esse hiperparâmetro ajusta o peso das classes, no cenário da classificação binária valores menores que 1 são utilizados em dados desequilibrados e previne que ele favoreça excessivamente a classe majoritária. No caso do *max_depth*, os maiores valores das métricas se concentram em 5, 7 e 9, indicando que grandes profundidades para as árvores são vantajosas para essa amostra.

Figura 10 – Variação das métricas de acordo com os valores de hiperparâmetros.

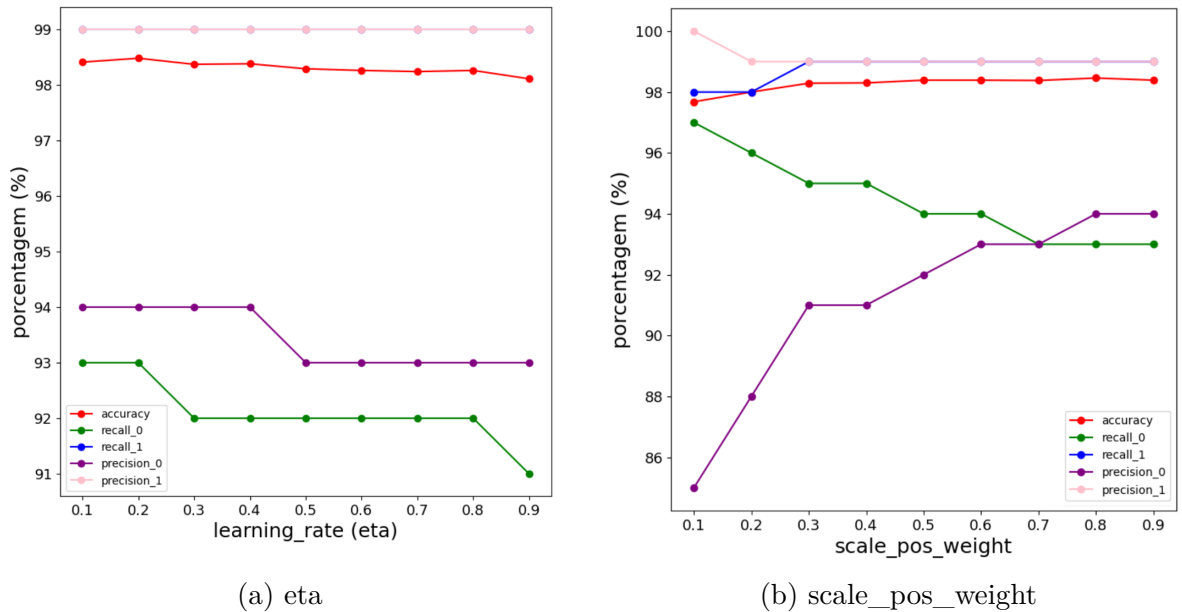
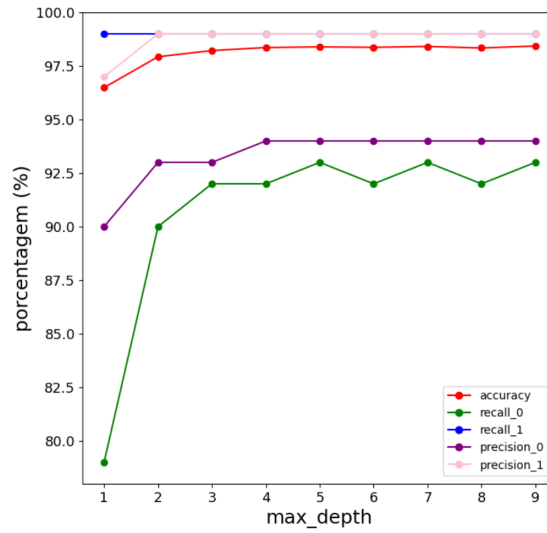
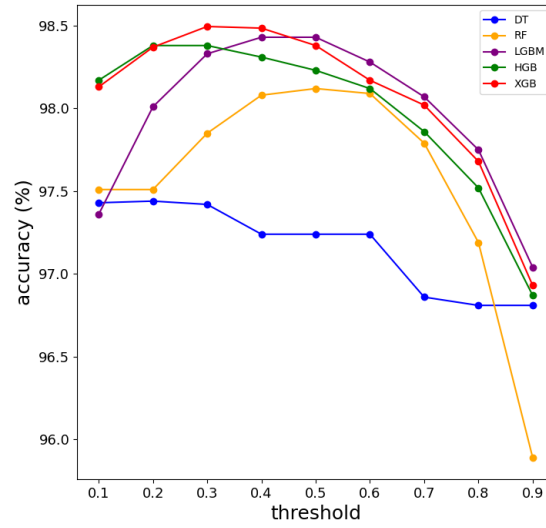


Figura 11 – Variação das métricas de acordo com os valores do hiperparâmetro *max depth*.

A busca de *threshold* foi feita, como discutido na seção 2.4, entre os valores 0 e 1, havendo o afinilamento do intervalo de acordo com as métricas obtidas. É válido ressaltar que a busca do *threshold* foi realizada após a determinação dos melhores hiperparâmetros, portanto, todos os resultados apresentados a seguir utilizam os valores de hiperparâmetros exibidos nas Tabelas 2, 3, 4, 5 e 6. Os resultados para a variação dos *thresholds*, com um passo igual a 0,1 para cada um dos algoritmos, são observados nas Figuras 12, 13 e 14, na qual as siglas apresentadas são referentes aos modelos: *Decision Tree* (DT), *Random Forest* (RF), *Histogram Gradient Boosting* (HGB), *Light Gradient Boosting Machine* (LGBM), *Extreme Gradient Boosting* (XGB). Na Figura 12, os algoritmos XGB e LGBM representados pelas cores vermelha e roxa, respectivamente, possuem os maiores valores para a *accuracy* no intervalo entre 0,3 e 0,5.

Figura 12 – Variação da *accuracy* de acordo com o *threshold*

Na Figura 13a, a tendência de valores mais altos para a *precision* da classe estável se concentra no intervalo de *threshold* de 0,1 a 0,2, dominado pelos algoritmos LGBM e RF. Enquanto, na Figura 13b o inverso ocorre e os maiores valores para a *recall* se encontram entre 0,8 e 0,9, onde os modelos RF, LGBM, HGB e XGB se igualam, havendo uma discrepância apresentada somente pelo DT. O *threshold* se relaciona a probabilidade do ponto pertencer à classe positiva ou negativa, como discutido na seção 2.4. Quando consideramos a classe estável positiva e a instável negativa, a *precision* alcança valores altos minimizando a taxa de Falsos Positivos (equação 2.7), ou seja, a probabilidade de predição da classe negativa é reduzida, levando à redução do *threshold*, confirmado pela Figura 13a. Já para a *recall* (equação 2.6) o contrário ocorre, para maximizar o seu alcance é preciso diminuir a taxa de Falsos Negativos, e assim reduzir a probabilidade na classificação de valores positivos por meio da elevação do *threshold*, o que se verifica na Figura 13b.

A Figura 14, exibe os resultados obtidos para a classe instável, na qual os valores mais altos para a *precision* se concentram entre 0,7 a 0,9 para o XGB e LGBM. Em vista disso, há concordância com o resultado observado para a classe estável, uma vez que a *precision* é agora calculada em referência aos Verdadeiros Negativos, e a taxa de Falsos Negativos é minimizada, levando ao aumento do *threshold*. Enquanto a *recall* exibe o inverso pelo mesmo raciocínio e os maiores percentuais são apresentados para o intervalo de *threshold* entre 0,1 e 0,3 para os algoritmos XGB, LGBM e HGB.

Figura 13 – Variação das métricas *precision* e *recall* para a classe estável de acordo com o *threshold*.

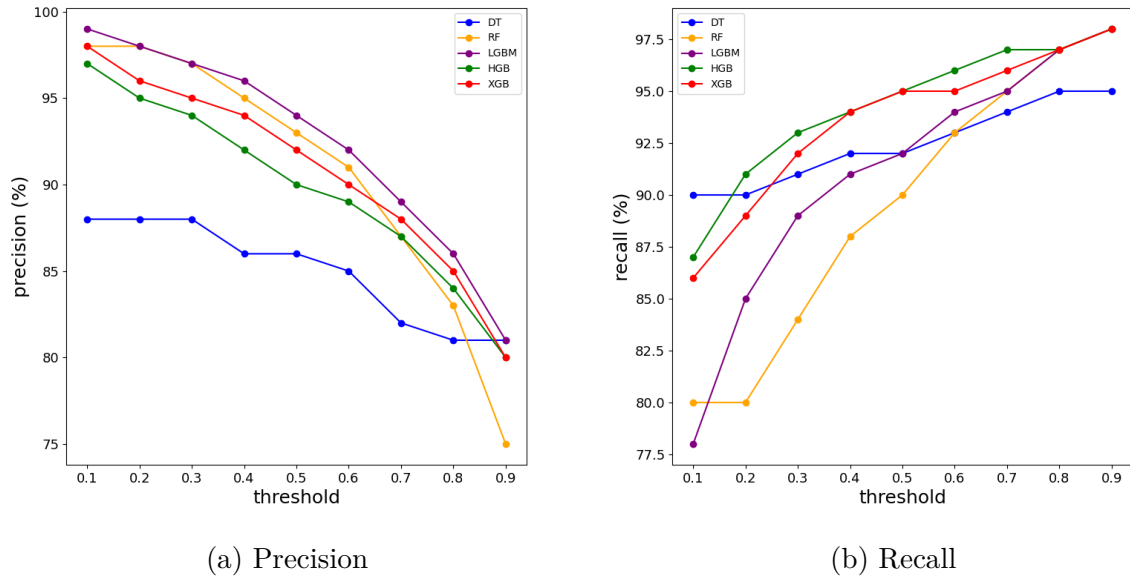
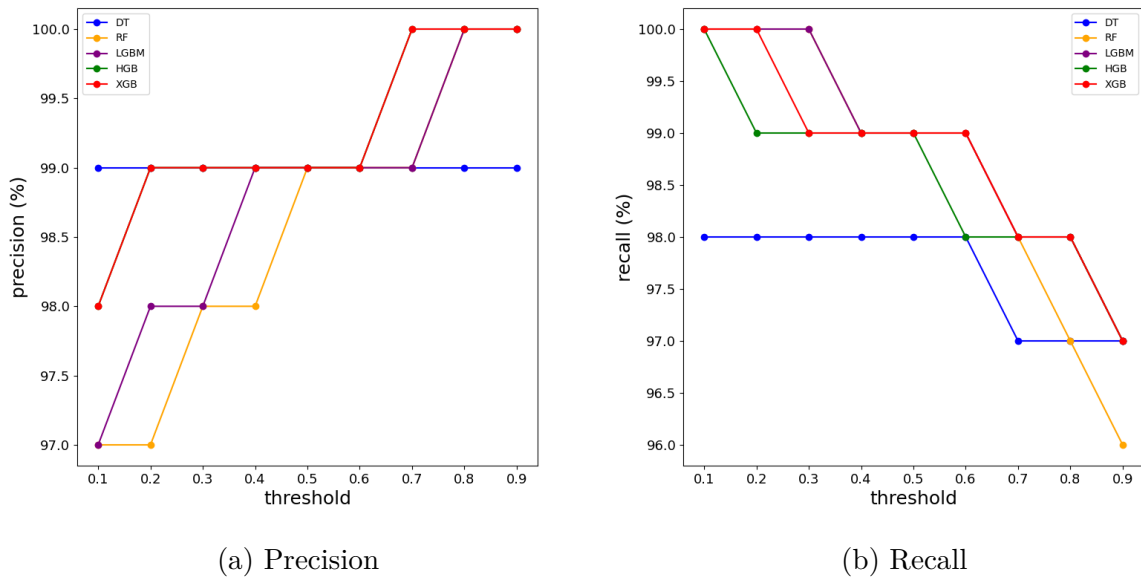


Figura 14 – Variação das métricas *precision* e *recall* para a classe instável de acordo com o *threshold*.

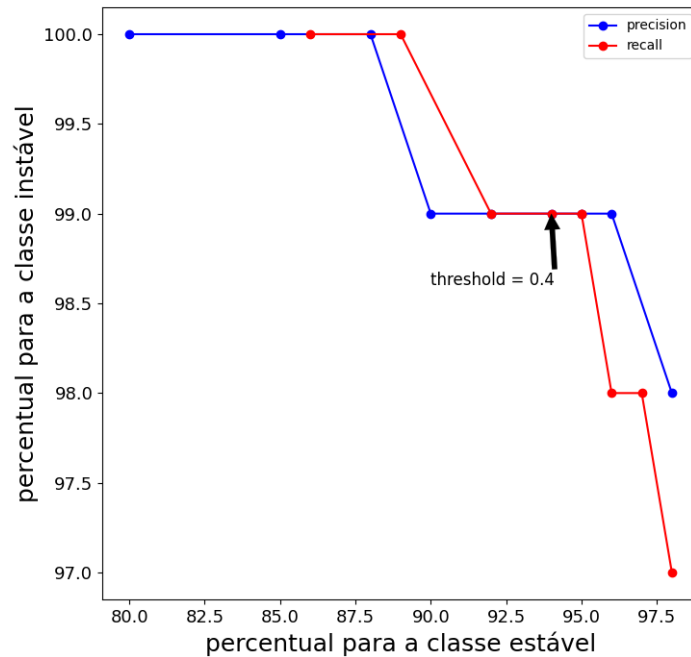


Os resultados demonstram a importância da seleção do *threshold*, uma vez que a sua mudança influencia diretamente na previsão do modelo. A fim de obter o equilíbrio entre as conclusões apresentadas, o melhor *threshold* escolhido para cada algoritmo foi aquele que demonstrou desempenho satisfatório para as métricas *accuracy*, *precision* e *recall*, bem como para ambas as classes estável e instável. Dessa forma, os valores selecionados para os

cinco algoritmos foram: *Decision Tree* = 0,1, *Random Forest* = 0,4, *LightGBM* = 0,45, *HGB* = 0,35, e *XGBoost* = 0,4.

Com o intuito de exemplificar a escolha do melhor *threshold*, observamos na Figura 15 a variação da *precision* (curva azul) e da *recall* (curva vermelha) para as classes estável e instável. As curvas se encontram em alguns pontos do gráfico, as regiões coincidentes próximas ao canto superior direito configuram os valores ideais para a atuação do modelo. Dessa forma, o ponto que apresenta essas características e pertence ao mesmo *threshold* é aquele em (99,94), ou seja, possui *precision* e *recall* iguais a 99% para a classe instável e 94% para a estável. O valor do *threshold* nesse ponto é igual a 0,4.

Figura 15 – Variação da *precision* e *recall* para as classes estáveis e instáveis de acordo com o *threshold*.



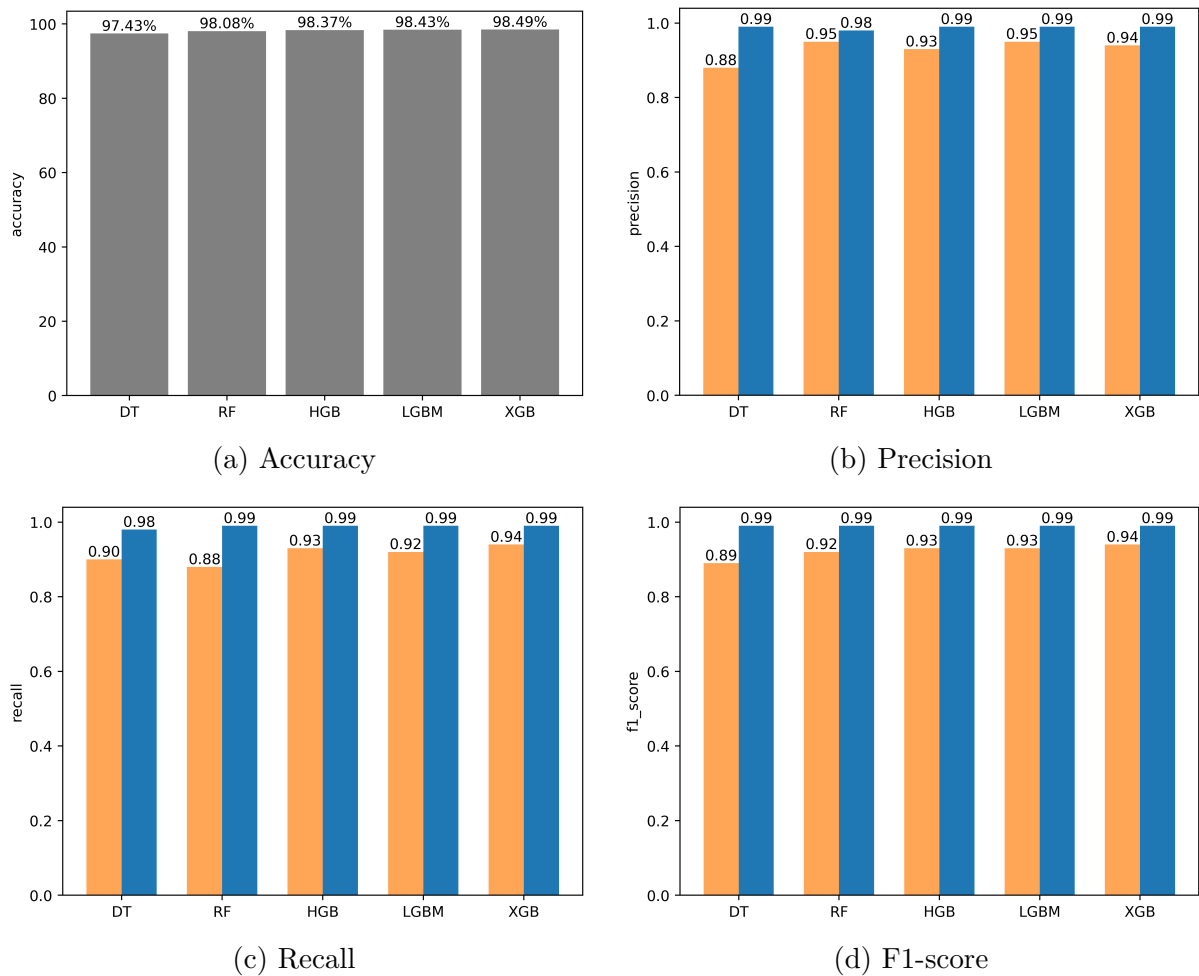
O resultado foi verificado mediante o uso de Algoritmo Genético. Como descrito na seção 2.1.4, essa técnica retorna o algoritmo de melhor desempenho para um dado conjunto de dados, assim como os hiperparâmetros mais indicados segundo a análise feita em cada iteração presente no processo do Algoritmo Genético.

3.2.2 Análise das métricas

Na seção 2.5 foram apresentadas as quatro métricas empregadas para avaliar o desempenho, dessa maneira o resultado alcançado para os cinco algoritmos utilizados é exibido na Figura 16, na qual as siglas apresentadas são referentes aos modelos: *Decision Tree* (DT), *Random Forest* (RF), *Histogram Gradient Boosting* (HGB), *Light Gradient*

Boosting Machine (LGBM), *Extreme Gradient Boosting* (XGB). Todos os algoritmos tiveram resultados comparáveis, no entanto, na Figura 16a o *XGBoost* se destaca com a acurácia igual à 98,49%, enquanto o menor valor (97,43%) foi alcançado pelo *Decision Tree*, revelando uma diferença de apenas 1,05%. Uma vez que a classe instável é 7,5 vezes maior que a classe estável, essa desproporção exerce uma influência sobre o cálculo da acurácia, feito considerando quão corretamente o modelo prevê as classes instáveis e estáveis sobre todo o conjunto de dados.

Figura 16 – O desempenho dos cinco modelos testados conforme as métricas *accuracy*, *precision*, *recall* e *f1-score*. As classes estáveis e instáveis são representadas pela cor laranja e azul, respectivamente.

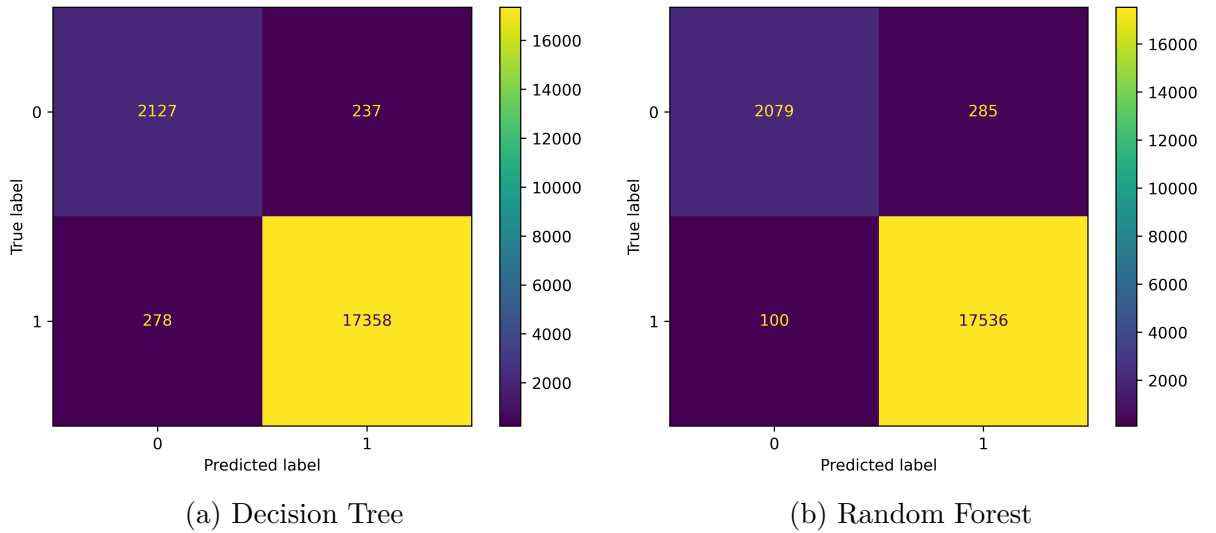


Nas Figuras 16b e 16c, vemos o resultado para a *precision* e a *recall* respectivamente, onde a cor laranja se refere a classe estável e a azul à classe instável. Com exceção da *precision* para o *Random Forest* e da *recall* para o *Decision Tree*, ambas iguais à 98%, todos os modelos alcançaram 99% de *precision* e *recall* no caso da classe instável. Já para classe estável, os modelos de *Random Forest* e *LightGBM* atingiram o maior valor para *precision*, classificando corretamente 95% da sua previsão. Entretanto, para a *recall* e *F1-score* (Figuras 16c e 16d), a melhor atuação para a classificação da classe estável é

obtida pelo *XGBoost*, igual a 94% para ambas. Em vista do cenário geral, através das quatro métricas avaliadas, o *XGBoost* é o modelo que obteve o melhor desempenho entre os cinco algoritmos testados.

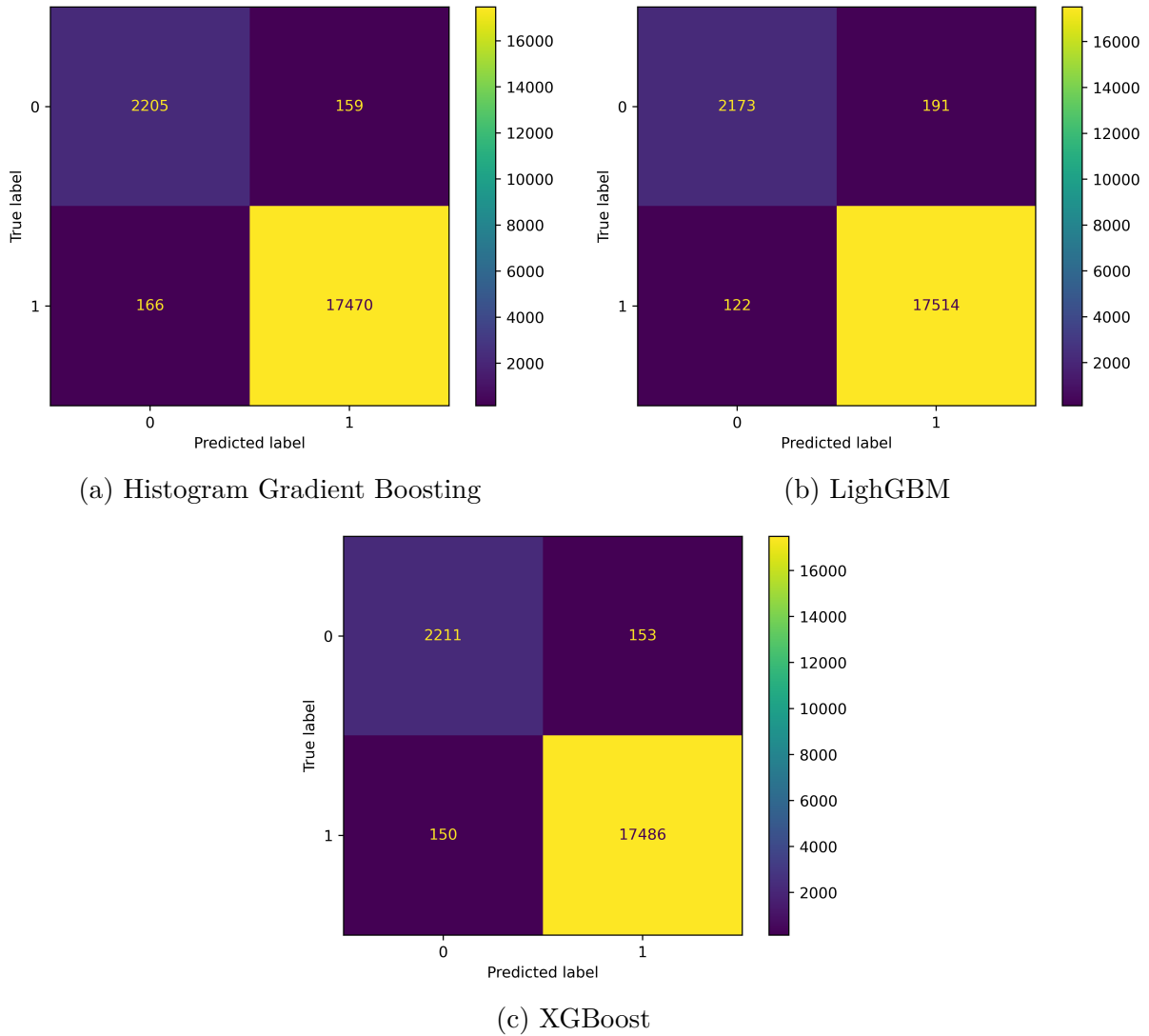
As Figuras 17 e 18 exibem a matriz da confusão obtida para cada um dos algoritmos. Como descrito na seção 2.5, através da matriz da confusão é possível analisar estatisticamente a previsão dos modelos em termos do conjunto de teste. Nesse caso, a classe 0 representa as partículas estáveis e a classe 1 as partículas instáveis. Em geral, todos os modelos erraram a previsão de poucos pontos da amostra de teste, de modo que aquele que apresentou uma maior discrepância foi o *Decision Tree*, com 19.485 classes corretamente classificadas de um total de 20.000.

Figura 17 – Resultados das matrizes da confusão para os melhores modelos dos algoritmos *Decision Tree* e *Random Forest*.



Por outro lado, o modelo *XGBoost*, que demonstrou a maior acurácia, falhou na previsão de somente 303 partículas, nas quais 153 pertenciam à classe estável e 150 à classe instável. Enquanto o *XGBoost* obteve a maior quantidade de acertos (2.211) para a classe estável, os modelos *Random Forest* e *LightGBM* previram incorretamente apenas 100 e 122 pontos respectivamente do conjunto de partículas instáveis, corroborando o resultado obtido para a *precision* igual a 0,95 para ambos e 0,01 maior que aquela alcançada para o *XGBoost*.

Figura 18 – Resultados das matrizes da confusão para os melhores modelos dos algoritmos *HGB*, *LightGBM* e *XGBoost*.



Outra maneira de avaliar o desempenho de um modelo de ML é calculando a área sob a curva ROC, como discutido na seção 2.4. Para a classificação binária, uma curva ROC perfeita é aquela onde a área sob a curva é igual à 1. Isto posto, a Tabela 7 demonstra que o valor que mais se aproximou de uma curva ideal, entre os cinco modelos, foi alcançado pelo *XGBoost* ($AUC = 0,9978$), como esperado. Os demais resultados não apresentaram grande divergência, sendo que o mais distante foi o *Decision Tree* com AUC igual à 0,9554.

Tabela 7 – Valores para a área sob a curva ROC (AUC) para o melhor modelo de cada algoritmo.

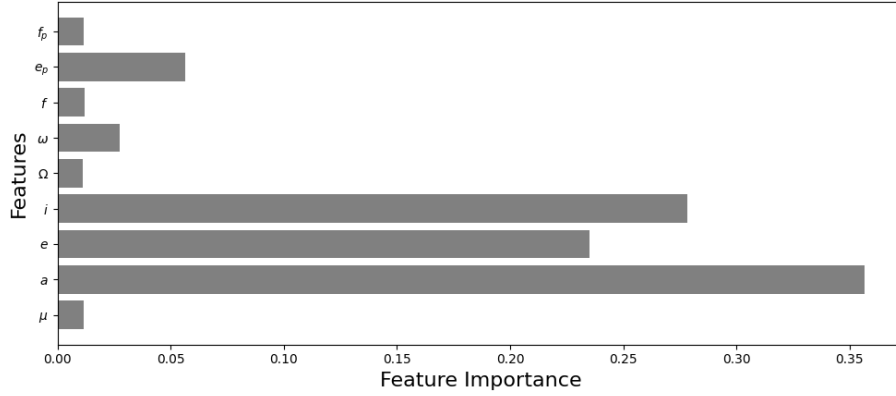
Modelo	AUC value
Extreme Gradient Boost	0,9978
Histogram Gradient Boosting	0,9974
Light Gradient boosting model	0,9975
Random Forest	0,9957
Decision Tree	0,9554

3.2.3 Importância dos atributos

Um retorno obtido pelo *XGBoost* é a importância atribuída a cada característica do conjunto de dados. O algoritmo faz essa medida calculando quão útil é cada atributo em reduzir a impureza durante as divisões presentes nas árvores de decisão. A Figura 19 mostra que as características com maior relevância são elementos orbitais da partícula, sendo eles o semieixo maior, a excentricidade e a inclinação com valores de importância iguais a 0,3567, 0,2783, 0,2351, respectivamente. A excentricidade do planeta é o quarto atributo de maior influência com um valor equivalente à 0,0564, seguido pelo argumento do pericentro da partícula (0,0274), os outros parâmetros possuem pouca atuação sobre a classificação, com valores de importância abaixo de 0,01.

O mosaico 9 na seção 3.1 discute a distribuição das condições iniciais para as quatro características principais do conjunto de dados, indicando fisicamente como a estabilidade do sistema pode ser alterada a partir da variação da excentricidade do planeta e da partícula, da inclinação e o semieixo maior da partícula. Portanto, a interferência causada por estes atributos na classificação das árvores de decisão presentes no *XGBoost*, está de acordo com a análise física realizada. Há uma redução significativa do número de sistemas estáveis para valores de semieixo maior acima de $0,6r_{Hill}$. Além disso, praticamente não há partículas estáveis com inclinação de 90° e excentricidade $e \geq 0,8$.

Figura 19 – Importância dos atributos para o modelo *XGBoost*. A representação dos elementos orbitais para a partícula é dada por: semieixo maior (a), excentricidade (e), inclinação (i), argumento do pericentro (ϖ), longitude do nodo (Ω), e anomalia verdadeira (f). Além disso, a excentricidade do planeta e a anomalia verdadeira são denotados por e_p e f_p , respectivamente. A razão de massa do sistema é definida por μ .



3.3 Aplicações

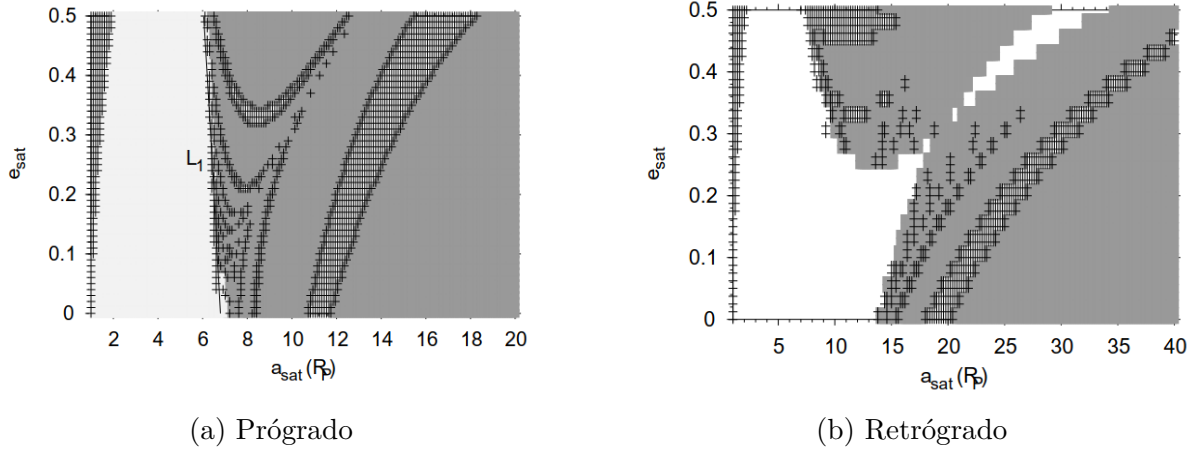
Neste capítulo, aplicamos o melhor modelo de *Machine Learning* alcançado com o propósito de reproduzir as regiões de estabilidade obtidas por Domingos et al. (2006), Pinheiro and Sfair (2021) e também, para um grupo específico de luas de Saturno conhecido como Inuit. Nesse cenário, foram gerados como conjuntos de teste para o ML, quatro mapas de estabilidade distintos ($a \times e$), construídos com 10.200 condições iniciais, nas quais as partículas de teste foram uniformemente distribuídas em um intervalo do semieixo maior inicial de $1,1 r_p$ à $1 r_{\text{Hill}}$ com um passo de $\Delta a = 0,1 r_p$ e excentricidade inicial entre 0 e 0,5, com $\Delta e = 0,005$.

3.3.1 Estabilidade ao redor de exoplanetas gigantes

O estudo realizado por Domingos et al. (2006), desenvolve diversos mapas de estabilidade via simulações numéricas, com o intuito de estabelecer os limites das regiões estáveis ao redor de exoplanetas com órbitas próximas à estrela, os quais possuem o semieixo maior igual à $0,1 \text{ u.a.}$ A integração era interrompida sempre que havia colisão entre o satélite e o planeta e entre o satélite e a estrela ou também quando a energia planetocêntrica se tornava positiva, caracterizando uma ejeção. Como resultado, o trabalho estabelece que a região de estabilidade possui dois extremos, um interno (a_I) e outro externo (a_E), definidos a partir de uma expressão para o semieixo maior crítico de exosatélites em função da excentricidade do satélite e do planeta com duas inclinações diferentes para órbita, equivalentes à 0° e 180° . A Figura 20, disponível no estudo de Domingos et al. (2006), demonstra os resultados das simulações numéricas para um planeta com órbita

circular.

Figura 20 – Regiões de estabilidade para satélites prógrados (figura à esquerda) e satélites retrógrados (figura à direita) hipotéticos com condições iniciais a_{sat} em relação a e_{sat} , retiradas do artigo elaborado por Domingos et al. (2006). A região branca indica a região estável. A borda externa desta região corresponde à curva de velocidade zero associada a L1. A cor cinza indica as regiões instáveis. O símbolo + se refere à colisão do satélite.



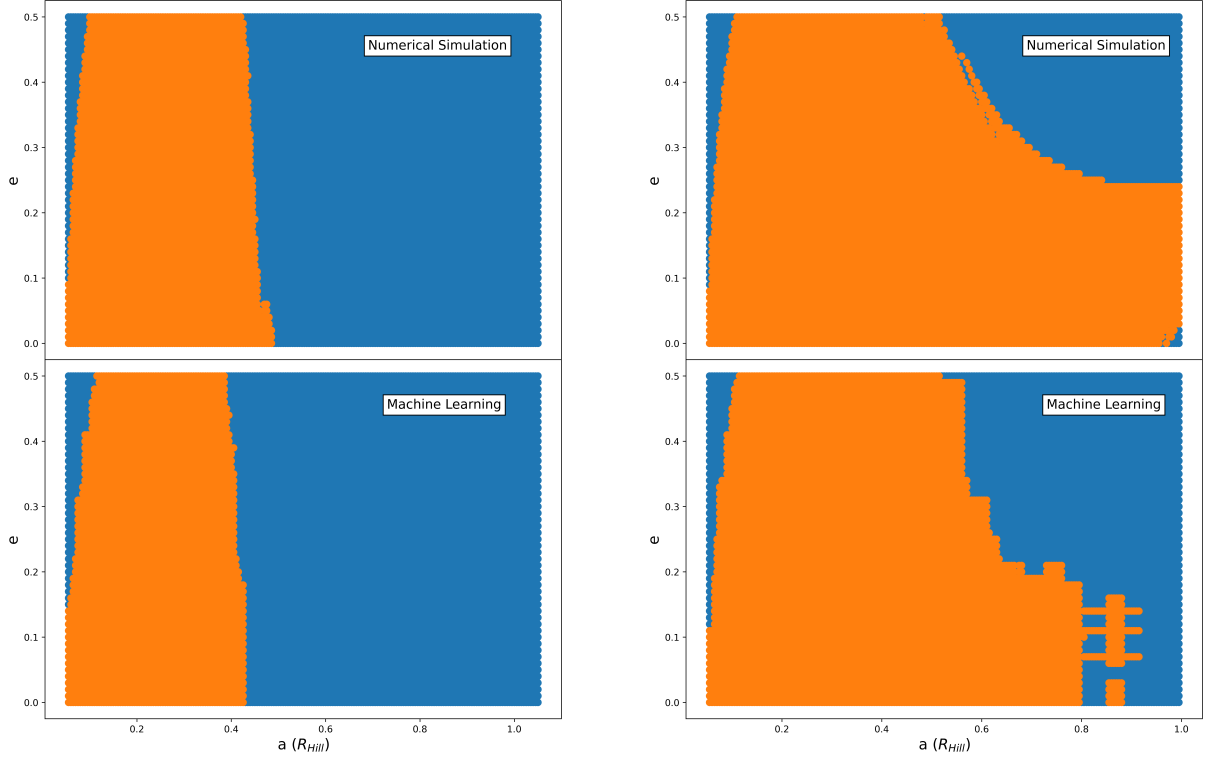
Fonte: (Domingos et al., 2006)

A fim de reproduzir os resultados de Domingos et al. (2006), foram desenvolvidos mapas ($a \times e$) dos casos prógrados e retrógrados com 10200 condições iniciais. Para caracterizar o sistema, a razão de massa foi definida como $\mu = 10^{-3}$, e o raio do planeta igual à $r_p = 0,05r_{Hill}$, sendo que o planeta possuía uma órbita circular ao redor estrela. Nesse caso, foram consideradas as colisões entre o satélite e o planeta, e a ejeção foi definida sempre que a órbita era hiperbólica, ou quando o semieixo do satélite era maior que o do planeta.

Das 10200 simulações numéricas, para o caso prógrado 54,92% das partículas foram caracterizadas como instáveis, enquanto 45,08% permaneceram estáveis. Já para o sistema retrógrado, 21,15% da amostra se manteve estável durante toda a simulação e 73,35% colidiu ou foi ejetada, representando um conjunto mais desequilibrado em relação ao anterior.

Na Figura 21 observamos a comparação entre os resultados das simulações numéricas e as previsões executadas pelo *Machine Learning*, nas quais a região azul representa as partículas instáveis, e a laranja as partículas estáveis. As imagens da primeira coluna são referentes aos sistemas prógrados, enquanto as da segunda coluna aos sistemas retrógrados. Mediante a análise visual, nota-se que o ML classifica corretamente a estrutura principal do mapa de estabilidade de ambos os cenários, entretanto para o caso retrógrado falha em prever alguns pontos da borda.

Figura 21 – Comparação entre os mapas de estabilidade resultantes das simulações numéricas e aqueles previstos utilizando o modelo *XGBoost*. A primeira coluna indica o sistema prógrado e a segunda o retrógrado. Os pontos em laranja representam a região estável, e os pontos em azul a região instável.

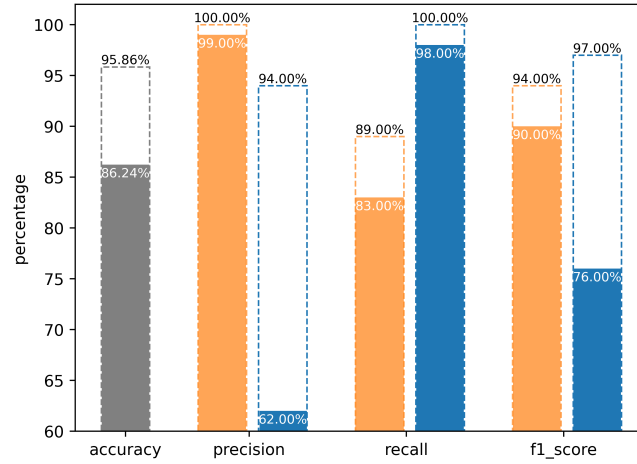


A Figura 22 exibe o desempenho do modelo de *XGBoost*, avaliando as métricas obtidas para essa amostra de dados, onde a linha tracejada representa o sistema retrógrado e as barras cheias o prógrado. A barra cinza indica a *accuracy*, enquanto as barras laranjas caracterizam a classes estável e aquelas em azul a classe instável. Para o sistema prógrado, o ML reproduziu a estabilidade da região com uma *accuracy* igual a 95,86%. No caso das partículas estáveis, o modelo classificou incorretamente apenas 5 pontos de 3.447 do conjunto, definindo uma *precision* igual 100%. A *recall* atingida para o cenário estável foi de 89%, coerente com o fato de que no resultado exibido pelas simulações numéricas a faixa referente a região estável é um pouco mais larga que aquela predita pelo ML. Para as partículas instáveis foi alcançada uma *recall* igual a 100%, ou seja, todas as partículas instáveis reais foram classificadas como instáveis. No entanto, como o modelo falha em prever os pontos na fronteira entre a região estável e instável, isso leva a uma *precision* de 94% para esta classe.

Das quatro aplicações reproduzidas nesta seção, o caso retrógrado (linhas tracejadas na Figura 22) apresentou o pior resultado para a *accuracy*, igual a 86%. O modelo classificou incorretamente apenas 44 partículas como estáveis de um total de 6.200, obtendo uma *precision* de 99%. Além disso, a *recall* para a classe instável foi igual 98%. Entretanto, o modelo não demonstrou uma boa generalização para a estrutura no limite de estabilidade,

fazendo com que partículas com semieixo maior que $0,8 r_{\text{Hill}}$ também fossem classificadas como instáveis.

Figura 22 – Performance obtida para o *XGBoost* sobre os sistemas prógrado e retrógrado, estabelecido em Domingos et al. (2006). As linhas tracejadas indicam o sistema prógrado, e as barras cheias o sistema retrógrado. A cor cinza representa a *accuracy*, a laranja as classes estáveis e a azul as classes instáveis.



3.3.2 Sistema PDS110b

No estudo desenvolvido por Pinheiro and Sfair (2021), foi verificada a estabilidade de um possível sistema de anéis ao redor do candidato à planeta PDS110b. Utilizamos o conjunto de dados construído para o sistema PDS110b para analisar o desempenho do modelo de ML, sendo que a amostra foi composta por $1,3 \times 10^6$ simulações numéricas do problema de três corpos, cada uma representando uma condição inicial do sistema. Dessa forma, foi definido o cenário onde o planeta tem massa equivalente à 10 massas de Júpiter, excentricidade 0,2 e a inclinação do anel é igual à 150° . Esses parâmetros correspondem a um possível sistema de anéis retrógrados ao redor de PDS110b.

Na Figura 23 é exibida a comparação entre os resultados das simulações numéricas e a classificação do modelo *XGBoost* para o sistema PDS110b, a região laranja representa as partículas estáveis, a em azul indica as partículas instáveis e a linha preta representa o raio mínimo para o anel observado (Pinheiro and Sfair, 2021). Novamente, via inspeção visual, percebe-se que o ML reproduz corretamente a estrutura principal do mapa de estabilidade e perde alguns pontos na fronteira entre as partículas estáveis e instáveis. Através da Figura 24, nota-se que a *accuracy* (barra em cinza na figura) obtida foi igual a 95,27%, caracterizando um resultado satisfatório para um sistema retrógrado em comparação com a aplicação anterior. O sistema presente no estudo de Domingos et al. (2006), possuía uma proporção de 70% partículas instáveis para o caso retrógrado, já nesse conjunto de dados, as classes são equilibradas, onde o número de partículas estáveis é igual a 5.044 e o de partículas instáveis é 4.599.

A *precision* e a *recall* apresentaram-se em torno de 94% e 97% para ambas as classes, a pequena diferença entre elas se deve a falha na generalização das bordas da região estável, onde foram classificados incorretamente 159 pontos do conjunto instável e 297 do estável. Além disso, algumas das partículas previstas de maneira errônea pertenciam aos pontos fora da curva, encontrados além da região de estabilidade na Figura 23. É válido ressaltar que as simulações numéricas atingiram um período orbital equivalente a 10^4 períodos orbitais, entretanto, algumas das partículas estáveis podem se tornar instáveis com um tempo de integração maior.

Figura 23 – Comparação entre os resultados obtidos pelas simulações numéricas e a previsão feita pelo modelo *XGBoost*. Os pontos azuis representam a região instável, enquanto aqueles em laranja representam a região estável, a linha preta indica o raio mínimo do anel observado.

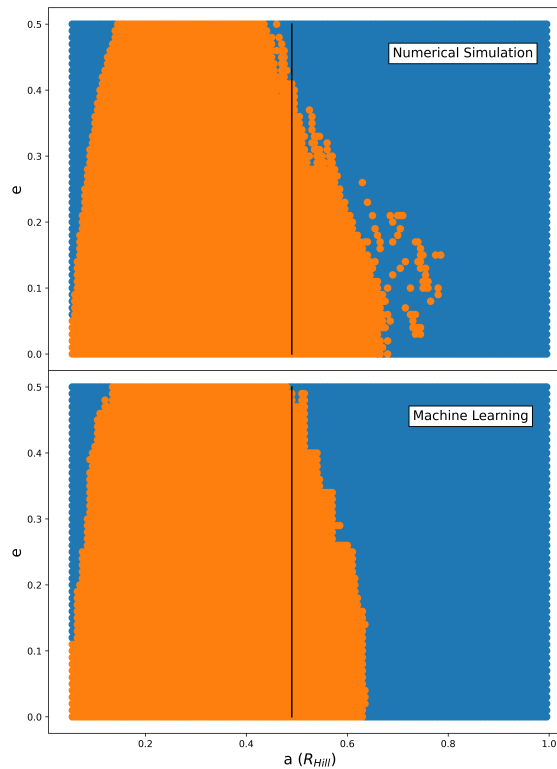
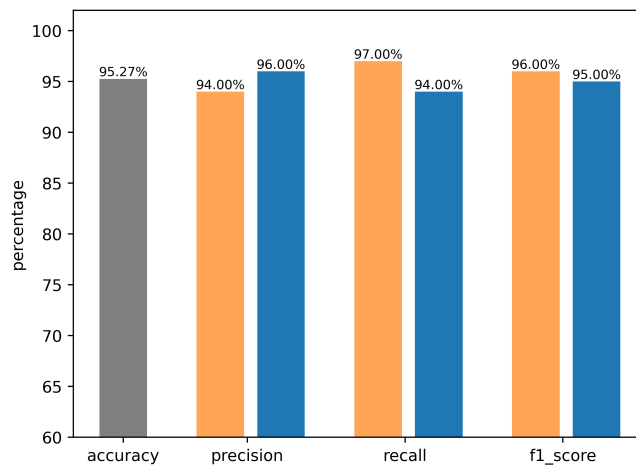


Figura 24 – Desempenho do modelo sobre o sistema PDS110b através das métricas: *accuracy*, *precision*, *recall* e *F1-score*. A barra cinza representa a *accuracy*, as barras em laranja indicam a classe estável e aquelas em azul a classe instável.



3.3.3 Sistema de satélites Inuit

A última aplicação desenvolvida neste capítulo é referente aos satélites Inuit de Saturno. Alguns dos satélites irregulares de Saturno são classificados em três diferentes grupos de acordo com sua inclinação e semieixo, os grupos Inuit, Gaulês e Nórdico. Satélites irregulares são geralmente objetos e excêntricos e acredita-se que sejam remanescentes da formação do Sistema Solar que foram capturados por Saturno ou sejam fragmentos de colisão (Ashton et al., 2021). As órbitas excêntricas e inclinadas sugerem a captura e inclinações aleatórias para estes satélites Carruba et al. (2002) analisam a estabilidade de satélites irregulares e demonstram com integrações de longo prazo (10^9 anos) que muitas órbitas de alta inclinação são instáveis devido a perturbações solares seculares. No caso do conjunto conhecido como Inuit, os satélites são prógrados com distâncias de Saturno entre 0,16 e 0,3 raios de Hill de Saturno e inclinação no intervalo entre 40° e 50° (Grav and Bauer (2007), Sheppard et al. (2018)). A tabela 8 demonstra os elementos orbitais dos satélites que compõem esse grupo.

Foram gerados mapas de estabilidade para os satélites Inuit cobrindo o intervalo de inclinação dos corpos, com 10.200 simulações numéricas do problema de três corpos, o qual foi composto pelo Sol, Saturno e uma partícula teste. A inclinação inicial da partícula de teste foi definida como 45° , e o argumento do pericentro, a anomalia verdadeira e a longitude do nodo ascendente foram aleatoriamente selecionados entre 0° e 360° com uma distribuição uniforme.

Tabela 8 – Elementos orbitais dos satélites de Saturno que pertencem ao grupo Inuit. Alguns dos satélites Inuit recentemente descobertos não foram incluídos nessa tabela, uma vez que as excentricidades orbitais que apresentam não são bem determinadas. (<https://carnegiescience.edu/>).

Satélite	a (r_{Hill})	e	i ($^{\circ}$)
Kiviuq	0,1803	0,334	45,71
Ijiraq	0,1805	0,316	46,44
S/2019 S1	0,1825	0,623	44,38
Paaliaq	0,2466	0,364	45,13
Siarnaq	0,2844	0,295	45,56
Tarqeq	0,2922	0,160	46,09
S2428b	0,2835	0,472	44,43
T522499	0,2824	0,242	48,11

A Figura 25 mostra os resultados numéricos e os obtidos através da predição do *Machine Learning* para esse exemplo, onde os pontos em laranja representam a região estável, os pontos em azul, a região instável, e os pontos em preto indicam os satélites pertencentes ao grupo Inuit. Como exibe a Figura 26, a melhor *accuracy* (barra em cinza na figura) foi alcançada para essa amostra de dados, entre as quatro testadas, equivalente à 96,73%. Para o caso estável obtivemos uma *precision* e uma *recall* de 97% e 93%, respectivamente. No geral, o ML classificou incorretamente apenas 333 pontos de um total de 10.200, sendo a maioria deles referentes a partículas dispostas na fronteira da região de estabilidade.

Figura 25 – Comparação entre os resultados referentes às simulações numéricas e a classificação realizada pelo modelo de *XGBoost* para o caso dos satélites Inuit de Saturno. A região azul representa as partículas instáveis, a laranja as partículas estáveis e os pontos pretos indicam os satélites do grupo Inuit.

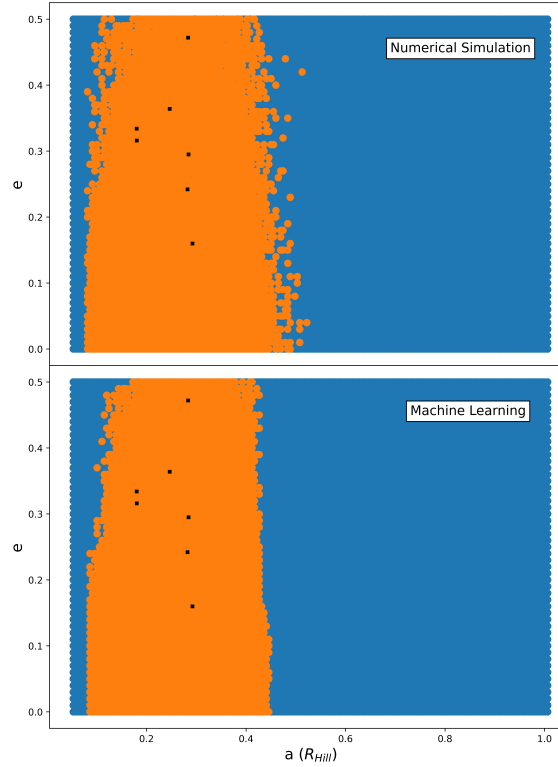
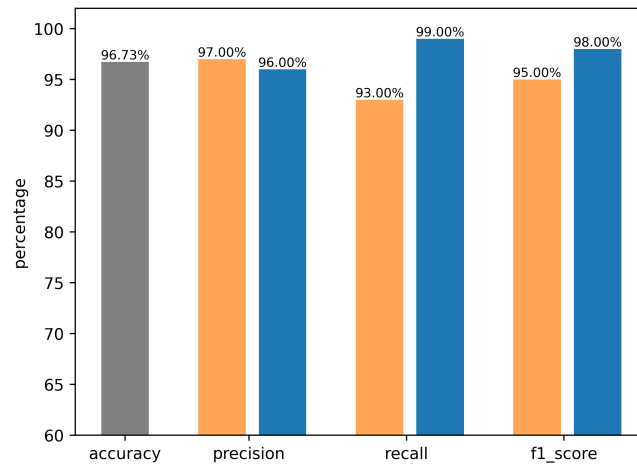


Figura 26 – Desempenho do modelo sobre o conjunto de satélites Inuit de Saturno, medido através das métricas: *accuracy*, *precision recall* e *F1-score*. A barra cinza demonstra a *accuracy*, as barras em laranja representam a classe estável e aquelas em azul indicam a classe instável.



3.4 Comentários finais

O propósito deste capítulo refletiu-se na aplicação de modelos de *Machine Learning* para gerar mapas de predição de regiões de estabilidade ao redor de sistemas planetários hipotéticos em órbitas do tipo-S. Dessa maneira, avaliar o desempenho dos algoritmos segundo a variação de hiperparâmetros e *threshold* a fim de determinar o modelo com as melhores métricas em relação ao conjunto de dados disponível.

O trabalho explorou uma grande gama de parâmetros orbitais e físicos do problema de três corpos adimensional composto por estrela, planeta e partícula. Esses parâmetros incluíam a razão de massa do sistema, o semieixo maior, a inclinação, o argumento do pericentro, a longitude do nodo ascendente da partícula, a excentricidade e a anomalia verdadeira do planeta e da partícula. No total, foram integrados 10^5 sistemas numéricos classificando as partículas em estáveis e instáveis. A amostra de dados construída foi empregada para compor o conjunto de treino para os algoritmos de ML.

Do total de simulações numéricas obtidas, 11,83% das partículas permaneceram estáveis, enquanto, 47,22% colidiu com o planeta e 40,95% foi ejetada do sistema, caracterizando um conjunto de dados desbalanceado, entretanto, a razão de desequilíbrio é de 1:10, o que não caracteriza um desequilíbrio grave. Portanto, a utilização de *oversampling* não demonstrou melhora significativa nessa amostra e não foi considerada nos resultados finais.

Nós utilizamos cinco algoritmos de ML: *Random Forest*, *Decision Tree*, *XGBoost*, *Light Gradient Boosting* e *Histogram Gradient Boosting*. Além disso, o desempenho do modelo derivou diretamente da busca de hiperparâmetros e *threshold*, acarretando em uma ampla variação dos mesmos com o intuito de avaliar aqueles que desencadearam o modelo mais eficaz. Dessa forma, o melhor desempenho foi performado pelo *XGBoost*, com uma *accuracy* igual a 98,48%, *precision* e *recall* para a classe estável e instável equivalentes a 94% e 99% respectivamente. Esse resultado foi confirmado através da utilização de Algoritmo Genético.

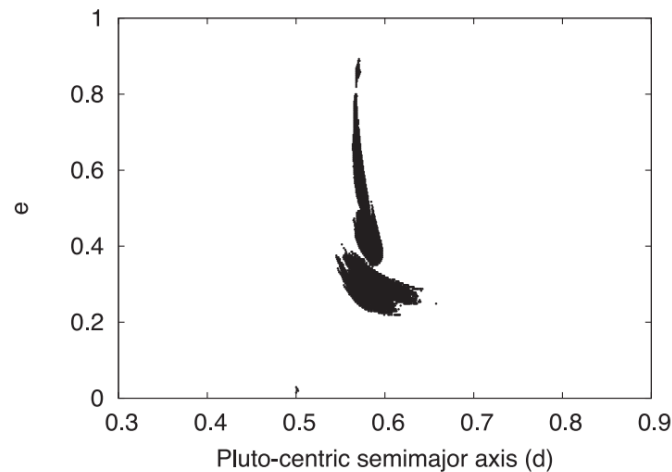
Visando verificar o desempenho do modelo definido em mapas de estabilidade, reproduzimos os resultados numéricos dos trabalhos de Domingos et al. (2006), Pinheiro and Sfair (2021) e do grupo de satélites Inuit de Saturno, obtendo uma acurácia de até 96,73%.

O melhor modelo do *XGBoost* demonstrou ser eficaz na identificação das regiões de estabilidade, perdendo apenas a classificação de alguns pontos na fronteira entre a região estável e instável para alguns casos de aplicação. Entretanto, apresentou métricas satisfatórias para prever regiões de estabilidade variando a ordem de 10^9 parâmetros físicos e orbitais, reduzindo o gasto computacional exigido pelas simulações numéricas.

4 Estabilidade da região da *Sailboat*

O planeta anão Plutão e seu satélite Caronte formam um sistema binário com razão de massa equivalente a aproximadamente 0,12 e separação de 19.595km (Brozović and Jacobson, 2024). Winter et al. (2010) exploram as regiões estáveis no sistema Plutão-Caronte, considerando partículas em órbitas prógradas e retrógradas. No estudo foram identificadas três regiões diferentes, a primeira composta por partículas teste orbitando Caronte com semieixo maior de até $0,2d$ e excentricidade menor que 0,6, onde d é a distância entre Plutão e Caronte. A segunda região foi caracterizada por partículas orbitando Plutão, com semieixo maior de até $0,5d$. A terceira região foi identificada como a *Sailboat*, localizada em um ponto intermediário entre Plutão e Caronte com $a_p = [0,5d - 0,7d]$ e $e_p = [0,2 - 0,9]$, onde a_p e e_p são o semieixo maior e a excentricidade da partícula, a região pode ser visualizada na Figura 27 retirada do trabalho de Giuliatti Winter et al. (2014).

Figura 27 – Diagrama das condições iniciais de $e \times a$, exibindo a região da *Sailboat*. Os parâmetros iniciais das partículas são $\omega = 0^\circ$ e $I = 0^\circ$.



Fonte: (Giuliatti Winter et al., 2014).

O nome para a última região vem do seu formato que remete a um barco à vela. Giuliatti Winter et al. (2014) também confirmaram a estabilidade da região da *Sailboat* através de Seções de Poincaré. Os autores classificaram essa região associada com a família “BD” de órbitas periódicas (Broucke, 1968), derivada de um problema de três corpos planar, circular e restrito.

Giuliatti Winter et al. (2014) analisaram os limites da região da *Sailboat* para o sistema Plutão-Caronte, explorando diferentes configurações da inclinação orbital da partícula (I) e argumento do pericentro (ω). O resultado do trabalho mostrou que a

Sailboat sobrevive com $I < 90^\circ$ e essa região está confinada dentro de dois intervalos para o argumento do pericentro: $\omega = [-10^\circ - 10^\circ]$ e $\omega = [160^\circ - 200^\circ]$, com máxima extensão da *Sailboat* em ω igual a 0° ou 180° .

O objetivo deste capítulo é realizar um entendimento profundo, através da aplicação do modelo de *Machine Learning* com maior desempenho, selecionado na seção 3.2, para prever a estabilidade da região da *Sailboat* variando parâmetros físicos e orbitais e restringindo o intervalo de sobrevivência em que a região pode existir. Esses parâmetros incluem a razão de massa do sistema binário, a excentricidade do corpo secundário, a inclinação da partícula e o argumento do pericentro.

No presente capítulo foi apresentada uma breve introdução sobre sistemas binários (seção 4.1) e sistemas binários compostos por planetas anões (seção 4.2), a fim de justificar o estudo sobre a região da *Sailboat*. Foram produzidas 1,2 milhões de simulações numéricas do problema de três corpos discutidas na seção 4.3. Essa amostra de dados foi empregada para criar o conjunto de treino para o algoritmo de *Machine Learning*, com o intuito de prever as partículas estáveis ao redor do corpo central. Para aprimorar o desempenho do modelo, este trabalho teve ênfase na seleção dos melhores hiperparâmetros e *threshold* apropriados para o problema em questão.

4.1 Sistemas Binários

Sistemas binários consistem em duas estrelas orbitando um centro de massa em comum, o tipo de interação entre as estrelas leva a classificação do sistema em binárias separadas, semi-separadas ou binárias de contato.

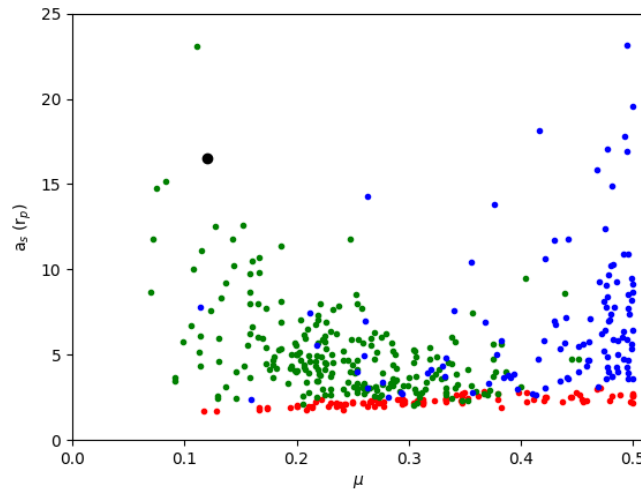
- Binárias separadas: ambas as estrelas estão dentro do seu Lóbulo de Roche, região em torno de cada estrela em que as suas forças gravitacionais dominam, dessa forma não há troca de material.
- Binárias semi-separadas: uma das estrelas preenche o Lóbulo de Roche da companheira, fazendo com que haja transferência de material para a estrela companheira.
- Binárias de contato: as estrelas preenchem o Lóbulo de Roche uma das outras, o que leva a transferência de material de ambas as partes.

A Figura 28 mostra o semieixo maior da estrela secundária (a_s), em unidades de raio da estrela primária (r_p), em relação à razão de massa do sistema binário (μ). O gráfico foi construído a partir de um levantamento composto por 489 sistemas binários eclipsantes, utilizando dados de três catálogos. Os dados são classificados em 100 sistemas do tipo binárias de contato, *Catalog of Early-type Contact Binaries*, (Bondarenko and Perevozkina, 2004), 232 em binárias semi-separadas, *Catalog Semi-detached Eclipsing Binaries*, (Surkova

and Svechnikov, 2004), e 157 do tipo binárias separadas, *Catalog of DMS-type Eclipsing Binaries*, (Svechnikov and Perevozkina, 2004).

Observamos na Figura 28 que a concentração ao redor da razão de massa próxima a do sistema Plutão-Caronte (ponto em preto) são sistemas que compõe binárias separadas. Como nesse tipo de sistema há transferência de massa, a região é provavelmente instável. Entretanto, existem quatro sistemas para binárias separadas com razão de massa $\mu < 0,25$ e distância de separação de $2r_p$ a $8r_p$, que provavelmente contém uma região do tipo *Sailboat*. Giuliatti Winter et al. (2014) demonstraram que a região da *Sailboat* pode existir para diferentes distâncias entre Plutão e Caronte.

Figura 28 – Semieixo maior da estrela secundária (a_s), em unidades de raio da estrela primária (r_p), em relação à razão de massa do sistema binário (μ). Os pontos em azul representam os sistemas para Binárias Separadas, em verde os sistemas Semi-Separados, e em vermelho as binárias de Contato. O ponto em preto representa o sistema Plutão-Caronte.



4.2 Planetas anões

Planetas anões¹ diferem dos planetas por não possuírem a capacidade de limpar a região circundante de suas órbitas. A Tabela 9 exibe os maiores planetas anões confirmados, ou candidatos a planeta anão, que contém pelo menos um satélite, onde μ é referente a razão de massa dada por,

$$\mu = \frac{m_2}{m_1 + m_2} \quad (4.1)$$

sendo m_1 a massa do satélite e m_2 a massa do planeta anão, indicados pela Tabela 9. Todos os corpos listados são objetos transnetunianos.

¹ O termo e a definição foram atribuídos durante a 26ª Assembleia Geral da *International Astronomical Union (IAU)*.

Os sistemas Orcus-Vanth e Varda-Ilmarë têm razão de massa similar ao de Plutão-Caronte, com $\mu = 0,16$ e $0,083$, respectivamente. Dessa forma, isso sugere que ambos binários provavelmente possuem uma região da *Sailboat* semelhante aquela do sistema de Plutão.

Tabela 9 – Sistemas binários de planetas anões e de seus satélites do Sistema Solar. Fonte: (a) Brozović and Jacobson (2024), (b) Brown and Butler (2023), (c) Holler et al. (2021), (d) Ragozzine and Brown (2009), (e) Morgado et al. (2023), (f) Kiss et al. (2019), (g) Grundy et al. (2019), (h) Grundy et al. (2015), (i) Sheppard et al. (2018).

Sistema Binário	μ	a (km)	e
Plutão e Caronte ^a	0,12	19.600	0,000161
Eris e Dysnomia ^{b,c}	0,0050	37.273	0,0062
Haumea e Hi'iaka ^d	0,012	49.880	0,0513
Makemake e S/2015 (136472) 1 ^d	0,002	21.100	desconhecido
Quaoar e Weywot ^e	0,00433	13.289	0,056
Gonggong e Xiangliu ^f	desconhecido	24.021	0,2908
Orcus e Vanth ^b	0,16	8.999,8	0,00091
Salacia e Actaea ^g	0,033	5.724	0,0098
Varda e Ilmarë ^h	0,083	4.809	0,0181
Sistema (532037) 2013 FY27 ⁱ	desconhecido	9.800	desconhecido

O sistema Orcus-Vanth pertence ao Cinturão de Kuiper, localizado em uma ressonância 2:3 com Netuno, Vanth é relativamente grande comparado a Orcus, diferente de outros KBOs que possuem um satélite pequeno. Orcus é caracterizado por um diâmetro de 910km, enquanto Vanth possui 475km de diâmetro. Varda é um objeto com 705km de diâmetros e $1,27 \text{ g/cm}^3$ de densidade, enquanto Ilmarë tem aproximadamente a metade do tamanho de Varda com 361 km de diâmetro.

Investigar as regiões de estabilidade ao redor desses corpos pode auxiliar missões espaciais futuras e campanhas de observação, uma vez que através desse estudo é possível identificar os locais prováveis para descoberta de novos objetos ou sistemas de anéis de poeira.

4.3 Simulações Numéricas

Com o intuito de analisar a estabilidade da região da *Sailboat* para diferentes sistemas binários, foram produzidas simulações numéricas pelos colaboradores do trabalho. Os resultados alcançados foram utilizados como conjunto de treino para o algoritmo de *Machine Learning*, a fim de classificar as partículas em estáveis e instáveis, e consecutivamente encontrar o valor crítico da razão de massa para o sistema binário onde existiria a região da *Sailboat*.

Foram desenvolvidas simulações do problema de três corpos elíptico com o integrador REBOUND/IAS15 (Rein and Spiegel, 2014), com a variação das condições orbitais iniciais e razão de massa do sistema binário. Cada simulação foi composta por um sistema binário adimensional com uma partícula teste orbitando o corpo central em uma órbita do tipo-S e perturbada gravitacionalmente pelo corpo perturbador.

O intervalo de tempo da integração foi caracterizado por $10^4 \times 2\pi$ unidades de tempo. Conforme Winter and Vieira Neto (2001) demonstram, 10^4 períodos orbitais são suficientes para classificar uma partícula como estável ou instável. Em todas as simulações numéricas, o corpo central foi definido como um ponto de massa, sendo a massa dada por $M_c = 1 - \mu$, onde μ é a razão de massa do sistema. O corpo perturbador possuía a massa igual a $m_p = \mu$, o semieixo maior $a_p = 1$, e o raio de colisão (r_p) definido como 10% do seu raio de Hill (equação 3.2).

As simulações numéricas foram compostas por 300.000 sistemas binários hipotéticos, a razão de massa (μ), o semieixo maior da partícula teste (a) e a excentricidade (e) foram aleatoriamente selecionados de acordo com os intervalos: $\mu = [0,01 - 0,3]$, $a = [0,45 - 0,7]$ e $e = [0 - 0,99]$, respectivamente. Os outros elementos orbitais para o corpo perturbador e para a partícula foram definidos como zero.

O estudo foi estendido para verificar como o limite da região da *Sailboat* varia de acordo com diferentes excentricidades iniciais do corpo perturbador, inclinações orbitais da partícula e argumentos do pericentro da partícula. Foram gerados três conjuntos de dados, compostos por 300.000 sistemas, o primeiro conjunto incluiu variações na excentricidade do corpo perturbador (e_p), de 0 a 0,99. A segunda abordagem variou a inclinação orbital da partícula, $I = [0^\circ - 180^\circ]$, por fim, o terceiro conjunto examinou variações no argumento do pericentro da partícula, $\omega = [0^\circ - 360^\circ]$.

Sempre que havia colisão da partícula com o corpo perturbador ou ejeção do sistema, a simulação numérica era interrompida. A ejeção foi definida como quando a partícula atingia uma órbita hiperbólica ($e \geq 1$) ou quando seu semieixo maior era $a > 1$, partículas que sobreviviam durante toda a simulação eram consideradas estáveis.

Os quatro conjuntos de dados foram empregados como conjunto de treino para o algoritmo *XGBoost* (*Extreme Gradient Boosting*), como analisado no capítulo 3.3, o *XGBoost* demonstrou desempenho superior na resolução do problema dos três corpos. Cada conjunto de dados passou por técnicas de reamostragem apresentadas no capítulo 2. Além disso, foram aplicados ajustes de hiperparâmetros e *threshold* com a intenção de otimizar o desempenho do modelo específico para cada conjunto de dados.

4.4 Análise da região *Sailboat* para diferentes razões de massa

Com o intuito de analisar a variação da região *Sailboat* de acordo com a razão de massa do sistema, foi construído um conjunto de dados composto pelos parâmetros livres: razão de massa do sistema (μ), semieixo maior da partícula (a) e excentricidade (e).

Tabela 10 – Intervalos de parâmetros livres para a análise de diferentes razões de massa.

Parâmetros livres	Intervalo
μ	0,01 - 0,24
a	0,45 - 0,7
e	0 - 0,99

O conjunto do total foi caracterizado por 300.000 amostras, onde 77,98% das partículas colidiram, 0,75% foram ejetadas do sistema, e 21,27% permaneceram estáveis durante toda a simulação. O conjunto apresentou leve desequilíbrio entre as classes estável e instável, apesar da razão de desequilíbrio ser menor que 1:100, definido como aquele indicado para a utilização de reamostragem por Carruba et al. (2023a). Entretanto, foram aplicados métodos de sobreamostragem a fim de analisar se havia melhora do desempenho do modelo.

4.4.1 Busca de Hiperparâmetros e *Threshold*

A busca de hiperparâmetros foi realizada através do uso de *Grid Search* para o *XGBoost*. As grades de configurações implementadas neste caso estão descritas na Tabela 11, assim como os hiperparâmetros selecionados para a otimização do modelo.

Tabela 11 – Resultados para o *hyperparameter tuning*, obtidos através de *Grid Search* para o *XGBoost*.

Hiperparâmetros	Espaço de busca	Valores selecionados
booster	gbtree, dart	gbtree
eta	0.2, 0.3, 0.4, 0.5, 0.6	0.6
scale_pos_weight	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.	1.0
max_depth	1, 2, 3, 4, 5, 6, 7, 8, 9	9

As Figuras 29 e 30 demonstram como a alteração dos valores dos hiperparâmetros afeta o desempenho do modelo. Os gráficos exibem a variação das métricas *accuracy* (vermelho), *recall* e *precision* para a classe estável (verde e roxo), e *recall* e *precision* para a classe instável (azul e rosa). Nessa análise, apenas o valor do hiperparâmetro indicado foi modificado e os outros parâmetros permaneceram no padrão do algoritmo.

Para o caso do hiperparâmetro **eta**, os valores para a classe instável apresentam uma pequena variação em torno de 99%, enquanto para a classe estável há um aumento

gradual das métricas que se estabilizam em 95%, após 0,6. Como apresentado na seção 2.3, o hiperparâmetro `eta` representa a taxa de aprendizado do modelo, dessa forma o valor 0,6 é relativamente alto, significando que há um grande ajuste dos pesos a cada iteração, aumentando a velocidade do aprendizado.

O comportamento para o `scale_pos_weight` exibe uma alternância maior para a classe instável, de modo que para o valor 0,8 há um ponto de coincidência entre a *recall* e *precision* para ambas as classes, denotando um valor que seria ótimo para esse caso.

Por fim, para o `max_depth` (Figura 30) todas as métricas crescem e se estabilizam em torno do `max_depth` igual à 7, sendo os valores ideais em torno de 7, 8 ou 9. Uma vez que esse hiperparâmetro indica a profundidade da árvore, a escolha de valores em torno de 9 representam uma grande profundidade, resultando em uma maior complexidade presente nas árvores do modelo.

Figura 29 – Variação das métricas de acordo com os hiperparâmetros *eta* e *scale pos weight*.

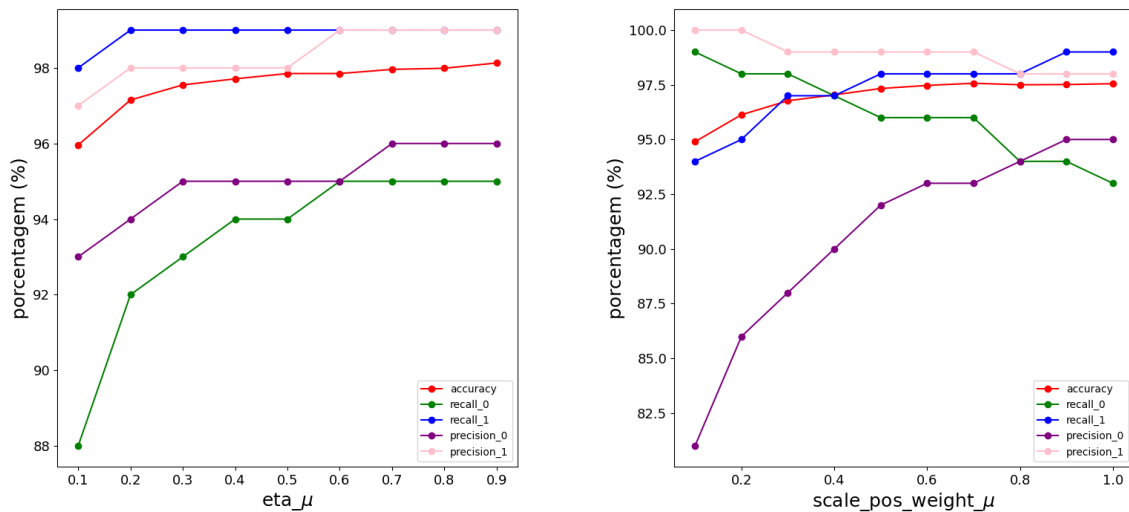
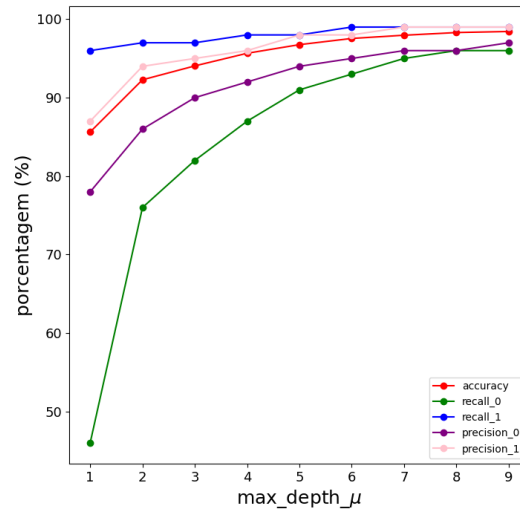
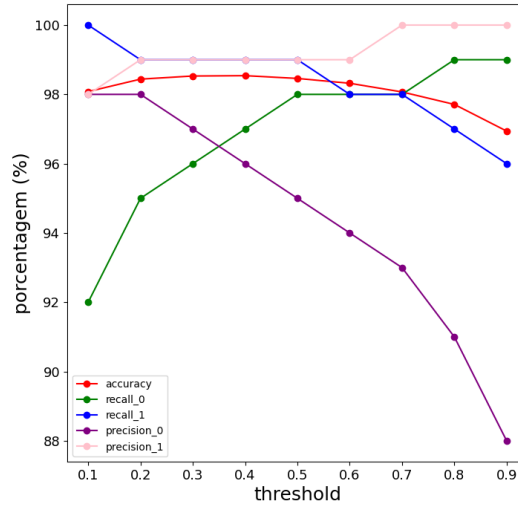


Figura 30 – Variação das métricas de acordo com o hiperparâmetro *max depth*.

A variação do *threshold* foi feita entre os valores 0 e 1, é válido ressaltar que a busca foi realizada após a determinação dos melhores hiperparâmetros, portanto, todos os resultados apresentados a seguir utilizam os valores de hiperparâmetros exibidos na Tabela 11, assim como o ajuste realizado com a utilização da técnica de sobreamostragem, *Random Oversampling*.

A Figura 31 exibe o desempenho das métricas de acordo com os valores de *threshold* com um passo igual a 0,1. As curvas em roxo e verde representam a *recall* e a *precision* para a classe estável, observamos que há um grande decaimento da *precision* e um crescimento da *recall* após o *threshold* igual a 0,3.

O *threshold* indica a probabilidade do ponto pertencer à classe positiva ou negativa, como discutido na seção 2.4. Sendo a classe estável positiva e a instável negativa, a *precision* atinge valores altos, minimizando a taxa de Falsos Positivos, o que acontece para valores pequenos de *threshold*. O contrário ocorre para a *recall*, indicada pela curva verde para a classe estável no gráfico, para maximizar o seu alcance, é preciso diminuir a taxa de Falsos Negativos, por meio da elevação do *threshold*. Portanto, um ponto ideal é aquele onde ocorre o encontro das duas curvas, como há uma constância entre as curvas da *recall* e *precision* para a classe instável, isso acontece na região entre 0,3 e 0,4 para ambas as classes. O valor do *threshold* selecionado como aquele que melhor otimizou o modelo para a amostra de dados disponível foi igual a 0,31.

Figura 31 – Variação das métricas de acordo com os valores do *Threshold*.

4.4.2 Resultados obtidos com o *XGBoost*

Como discutido na seção 3.2, o modelo selecionado com o melhor desempenho foi o *XGBoost*. Após realizada a busca de hiperparâmetros e *threshold* e empregada a técnica *Random Oversampling*, nosso modelo mais eficaz alcançou uma *accuracy* igual a 98,54%.

Para a classe estável, foi obtida uma *recall* de 97%, enquanto para a classe instável, a *recall* foi equivalente a 99%, indicando que o modelo acertou 97% da amostra estável total e 99% da amostra instável total. Além disso, a *precision* foi igual à 97% para a classe estável e 99% para a classe instável, indicando que 97% dos pontos classificados como estáveis e 99% dos pontos classificados como instáveis foram corretamente preditos.

Os resultados da previsão do *Machine Learning* são exibidos nas Figuras 32 e 33, que apresentam diagramas das condições iniciais ($e \times a$) para a partícula teste orbitando o corpo primário do sistema binário. As razões de massa do sistema são exibidas com um passo igual a $\Delta\mu = 0,01$. Cada diagrama possui 247.500 condições iniciais, onde o semieixo maior da partícula e a excentricidade percorrem os intervalos descritos na Tabela 10 com $\Delta a = \Delta e = 0,001$.

Para $\mu \leq 0,04$, o sistema binário possui uma região estável ao redor do corpo central que se estende até $a = 0,7$. No intervalo $a = [0,55 - 0,68]$, as partículas estáveis alcançam altas excentricidades, um pouco abaixo de 1. Quando a razão de massa do sistema binário é igual a $\mu = 0,05$, a região estável é separada em duas regiões distintas, a primeira região é definida por $a < 0,52$ e $e \leq 0,4$, já a segunda está em um intervalo de $a = [0,52 - 0,7]$ e $e < 1$, conhecido como região da *Sailboat*. Conforme o valor de μ cresce, a região da *Sailboat* diminui gradualmente até desaparecer completamente, com uma razão de massa máxima igual a $\mu = 0,22$.

Em $\mu = 0,10$, a região da *Sailboat* é separada em duas, uma região possui uma largura maior e existe nos intervalos iguais a $a = [0,52 - 0,67]$ e $e = [0,2 - 0,8]$, enquanto a região menor se encontra em órbitas excêntricas com $e > 0,85$.

Para sistemas com $\mu = 0,15$, a região principal da *Sailboat* se encontra confinada em $a = [0,52 - 0,63]$ e $e = [0,22 - 0,7]$, essa região se torna menos excêntrica e perde o formato de vela. Para $\mu = 0,18$, a região estável é encontrada no intervalo $a = [0,5 - 0,6]$ com $e < 0,5$, indicando um deslocamento gradual próximo ao corpo central e uma redução da excentricidade máxima.

Em $\mu = 0,22$, novas regiões estáveis aparecem em $a = [0,5 - 0,6]$ com $e < 0,5$, essa região se expande e se funde com a região remanescente da *Sailboat* em $\mu = 0,23$, na qual o intervalo do semieixo maior vai de 0,5 até 0,52 e a excentricidade é aproximadamente 0,4.

Figura 32 – Evolução da estabilidade das regiões da *Sailboat* (em preto) para diferentes valores de razão de massa $\mu = [0, 13 - 0, 24]$ e partículas em órbitas coplanares ao redor do corpo central, com $\omega = \Omega = f = 0^\circ$.

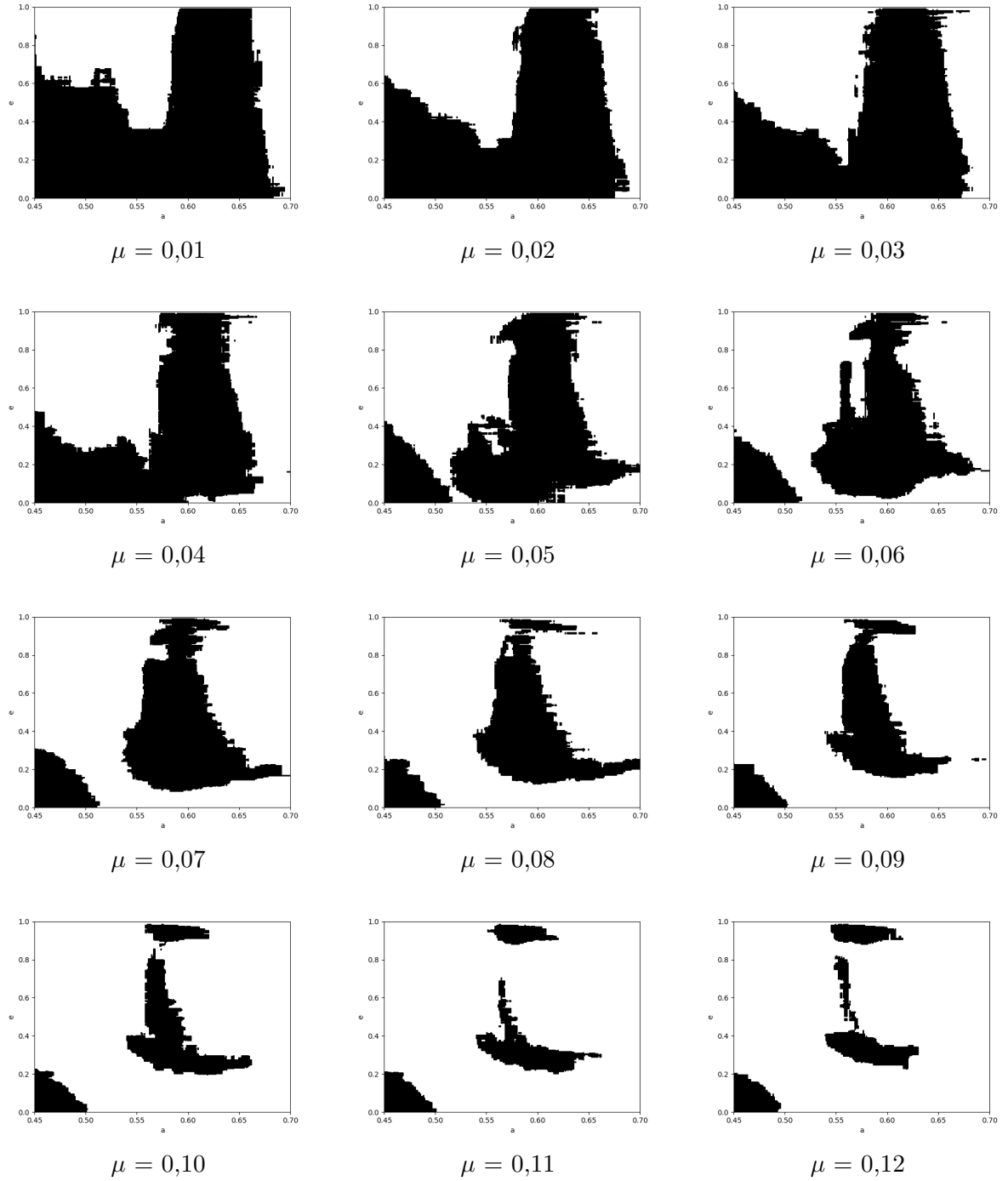
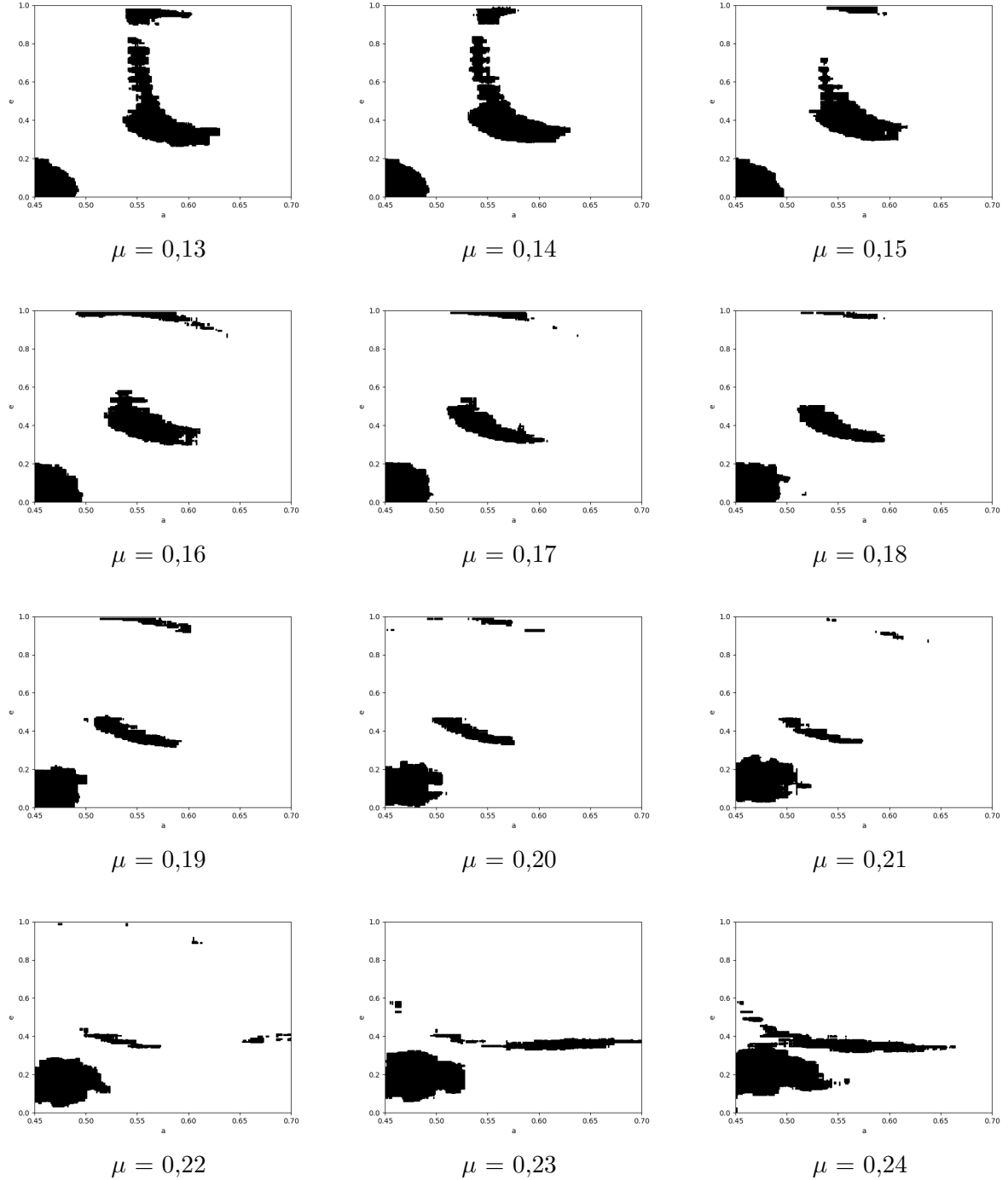


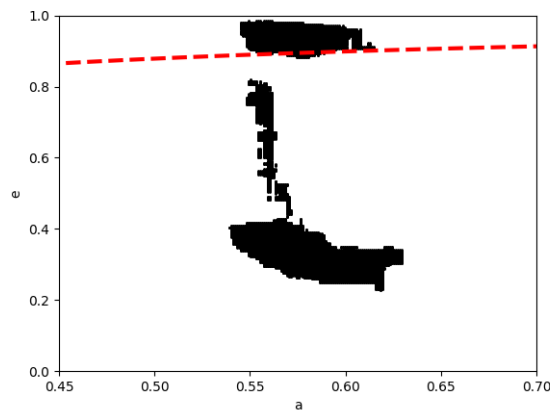
Figura 33 – Evolução da estabilidade das regiões da *Sailboat* (em preto) para diferentes valores de razão de massa $\mu = [0, 13 - 0, 24]$ e partículas em órbitas coplanares ao redor do corpo central, com $\omega = \Omega = f = 0^\circ$.



Algumas órbitas estáveis excêntricas ($e > 0,85$) presentes na região da *Sailboat* podem se tornar instáveis de acordo com o tamanho do corpo central. A Figura 34 demonstra a região da *Sailboat* para um sistema binário com $\mu = 0,12$ (o valor de μ mais próximo do sistema Plutão-Caronte), a linha tracejada vermelha indica onde o pericentro da partícula é correspondente ao raio de Plutão. De acordo com a figura, todas as partículas

com $e > 0,9$ colidiram com Plutão e se tornam instáveis. Esse resultado está coerente com os resultados numéricos presentes em (Giuliatti Winter et al., 2014), e pode ser comparado com a Figura 27, apresentada no início deste capítulo, no qual a região da *Sailboat* é encontrada em $a = [0,5 - 0,7]$ e $e = [0,2 - 0,9]$.

Figura 34 – Região da *Sailboat* para um sistema binário com $\mu = 0,12$ (o caso mais próximo do sistema Plutão-Charonte, onde Charonte tem excentricidade). A linha tracejada vermelha representa o pericentro da partícula a mesma distância do raio de Plutão. Não há outra região estável no intervalo $a = [0,3 - 0,7]$, não exibido no gráfico.



4.5 Análise da região *Sailboat* para o corpo excentricamente perturbado

A segunda abordagem empregada foi caracterizada pela análise da estabilidade da região da *Sailboat* através da variação da excentricidade do corpo perturbador (e_p). O conjunto de dados utilizado foi composto por quatro parâmetros livres indicados na Tabela 12:

Tabela 12 – Grade de parâmetros livres.

Parâmetro livre	Intervalo
μ	0,01 - 0,23
a	0,45 - 0,7
e	0 - 0,99
e_p	0 - 0,99

Através do estudo desses parâmetros, é possível investigar como mudanças na excentricidade do corpo perturbador afetam a estabilidade da região da *Sailboat*. A amostra foi composta por 1,77 % de partículas estáveis, e 98,23 % instáveis, nas quais 95,39 % foram ejetadas e 2,84 % colidiram com o corpo perturbador.

Para lidar com o desequilíbrio do conjunto de dados, foi empregada a técnica de sobreamostragem *ADASYN* (*Adaptive Synthetic Sampling*), discutida na seção 2.2, a qual recuperou o modelo ótimo para essa amostra.

4.5.1 Busca de Hiperparâmetros e *Threshold*

A busca de hiperparâmetros foi realizada através do uso de *Grid Search*. As grades de configurações implementadas neste caso estão descritas na Tabela 11, os hiperparâmetros selecionados como aqueles que melhor otimizam o modelo são exibidos na Tabela 13.

Tabela 13 – Resultados para o *hyperparameter tuning*, obtidos através de *Grid Search* para o *XGBoost*.

Hiperparâmetros	Valores selecionados
booster	gbtree
eta	0.6
scale_pos_weight	0.8
max_depth	9

As Figuras 35 e 36 demonstram como a alteração dos valores dos hiperparâmetros afeta o desempenho do modelo. Os gráficos exibem a variação das métricas *accuracy* (vermelho), *recall* e *precision* para a classe estável (verde e roxo), e *recall* e *precision* para a classe instável (azul e rosa).

Nessa análise, apenas o valor do hiperparâmetro indicado foi modificado e os outros parâmetros permaneceram no padrão do algoritmo, assim como nenhuma técnica de sobreamostragem foi empregada para a realização desses gráficos. Como esse conjunto de dados possui um desequilíbrio grave com razão próxima de 1:100, a implementação do ajuste do *threshold* e do *ADASYN* caracterizam uma grande influência no desempenho das métricas, havendo um aumento na porcentagem após a aplicação dos mesmos (Figura 37).

Para o caso do hiperparâmetro **eta**, a porcentagem das métricas para a classe instável é praticamente constante em torno de 100%, enquanto para a classe estável, a *recall* tem um baixo desempenho com valores máximos em 85% e a *precision* atinge 90% para **eta** entre 0,3 e 0,6. Como discutido na seção 2.3, valores baixos como 0,3 para o **eta** indicam um aprendizado mais lento, já 0,6 seria um valor relativamente alto, acelerando um pouco o processo de iteração.

O comportamento para o **scale_pos_weight** é similar ao do **eta** para a classe instável, e exibe uma grande alternância para a classe estável, de modo que para o valor 0,6 há um ponto de coincidência entre a *recall* e a *precision* para ambas as classes, denotando um valor que seria ótimo para esse caso.

Para o **max_depth** (Figura 36) todas as métricas crescem e se estabilizam em torno do **max_depth** igual a 7, sendo os valores ideais em torno de 7, 8 ou 9. Indicando que as

árvores que maximizam a eficácia do modelo possuem alta profundidade.

Figura 35 – Variação das métricas de acordo com os valores dos hiperparâmetros η e $scale_pos_weight$.

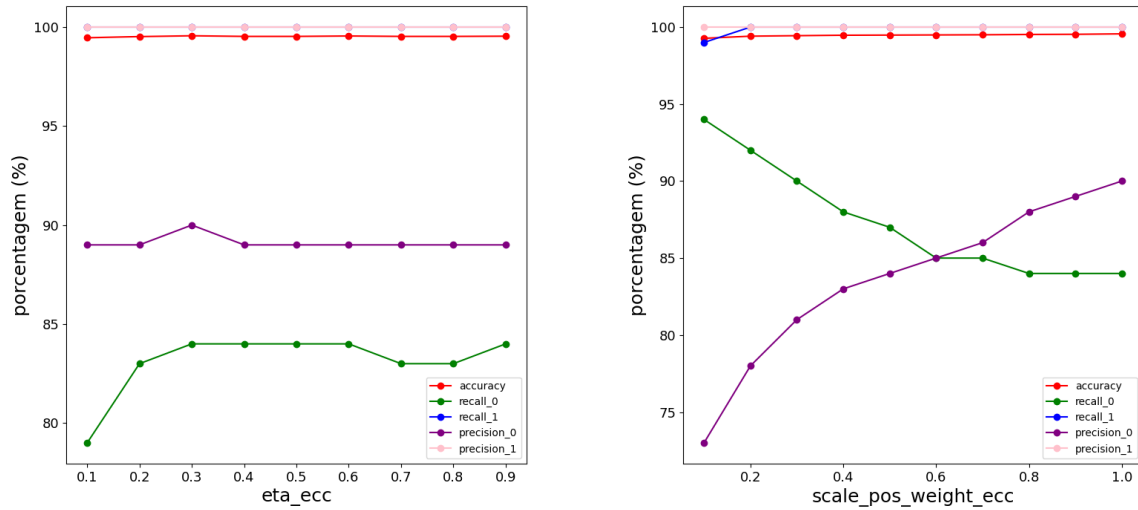
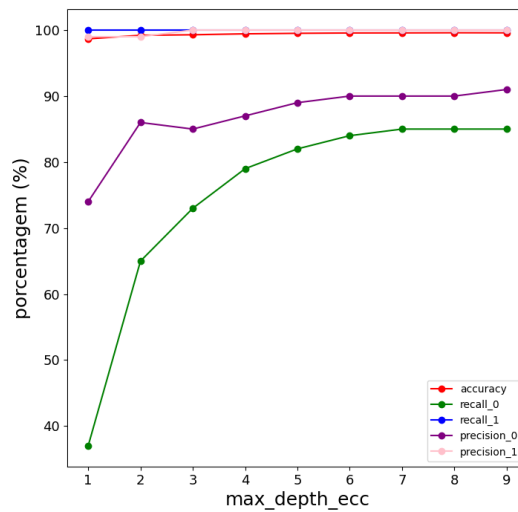


Figura 36 – Variação das métricas de acordo com os valores do hiperparâmetro max_depth .

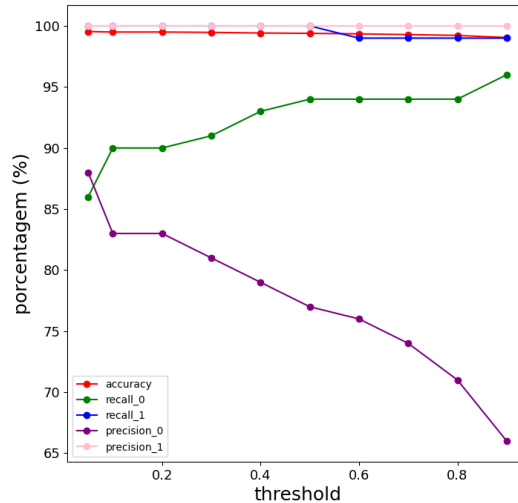


A variação do *threshold* foi feita entre os valores 0 e 1 com passo igual a 0,1. Todos os resultados apresentados a seguir utilizam os valores de hiperparâmetros exibidos na Tabela 13, assim como o ajuste realizado com a utilização da técnica de sobreamostragem ADASYN.

A Figura 31 exibe o desempenho das métricas de acordo com os valores de *threshold*. As curvas em roxo e verde representam a *recall* e a *precision* para a classe estável, observamos que as curvas se encontram somente no início do eixo x , para valores de

threshold menores que 0,1. Indicando que o ideal para esse conjunto de dados é minimizar a taxa de Falsos Positivos, uma vez que a porcentagem da classe estável (classe positiva) na amostra é de apenas 1,77%. Portanto, o valor escolhido para o *threshold* foi igual a 0,05.

Figura 37 – Variação das métricas de acordo com os valores do *Threshold*.



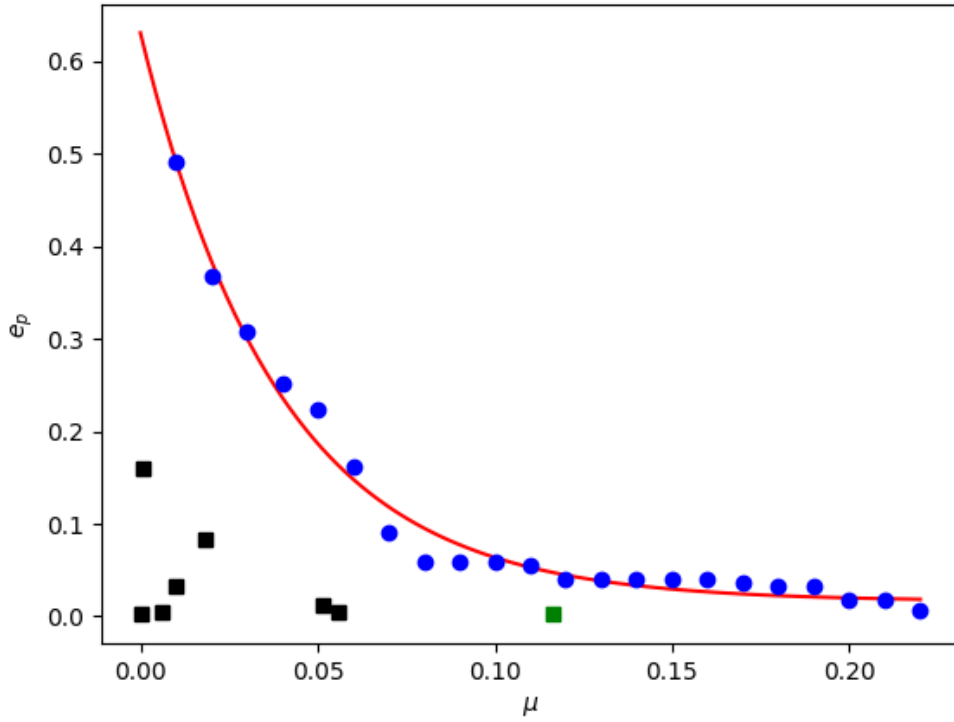
4.5.2 Resultados obtidos com o *XGBoost*

A aplicação do *XGBoost* com a seleção dos melhores hiperparâmetros, *threshold* e sobreamostragem, levou a uma *accuracy* igual a 99,55%. Para a classe estável, a *recall* foi igual a 87%, enquanto para a classe instável a *precision* foi aproximadamente 100%.

O algoritmo de *Machine Learning* foi utilizado para prever mapas de estabilidade ($a \times e$) com $\Delta\mu = 0,01$ e $\Delta e_p = 0,001$, definindo o valor máximo para e_p , onde a região da *Sailboat* desaparece completamente.

A Figura 38 mostra a excentricidade máxima do corpo perturbador (pontos em azul) para diferentes razões de massa nas quais a região da *Sailboat* existe. A curva vermelha representa o ajuste obtido utilizando o método dos Mínimos Quadrados, cada quadrado preto corresponde a um sistema binário de planetas anões e seus satélites, como listado na Tabela 9, já os quadrados verdes caracterizam o sistema Plutão-Caronte. A figura demonstra que todos os planetas anões indicados na Tabela 9 podem exibir uma região da *Sailboat*, sugerindo regiões potenciais para hospedar satélites ou sistemas de anéis de poeira.

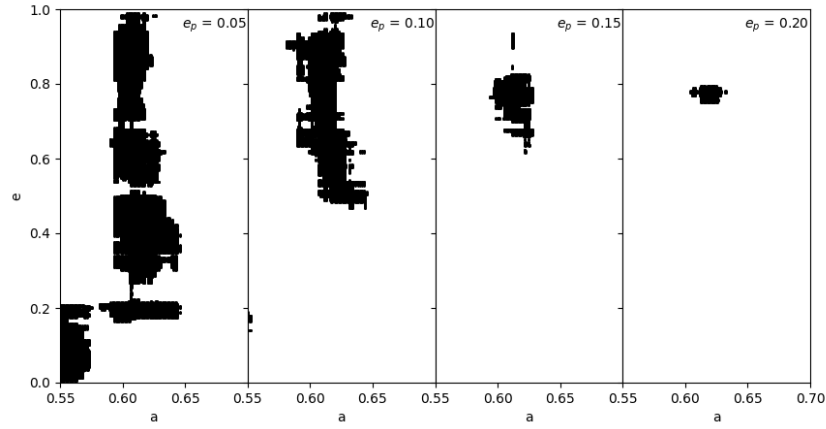
Figura 38 – Excentricidade máxima do corpo perturbador (pontos azuis) em relação a razão de massa do sistema. A curva vermelha representa o modelo ajustado obtido utilizando o método dos Mínimos Quadrados. Os quadrados pretos denotam os sistemas binários dos planetas anões e seus satélites, enquanto o quadrado verde representa o sistema Plutão-Caronte.



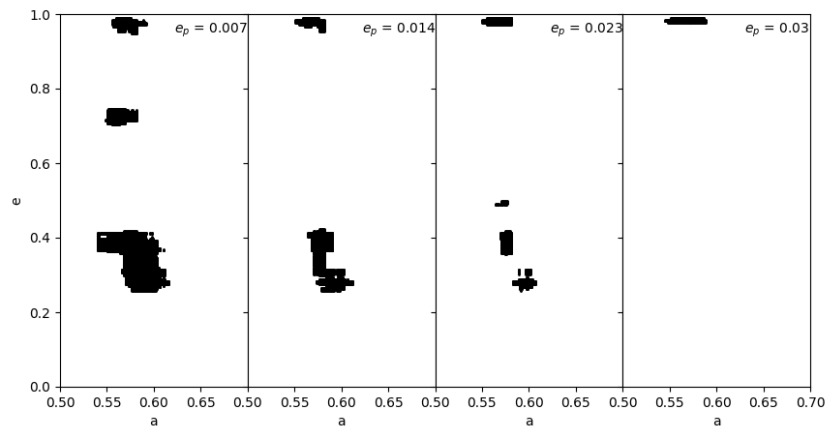
Pequenas variações nos valores da excentricidade do corpo perturbador são suficientes para reduzir significativamente o limite da região da *Sailboat*. A Figura 39 exibe diagramas $e \times a$, mostrando a evolução da região da *Sailboat* conforme a excentricidade do corpo perturbador varia para sistemas binários com razões de massa $\mu = 0,05$, $0,12$ e $0,22$. O decrescimento da excentricidade do corpo perturbador faz com que partículas com baixa excentricidade na região da *Sailboat* se tornem instáveis. Para o caso de $\mu = 0,05$ e $e_p = 0,2$, apenas partículas com excentricidades ao redor de $e \sim 0,8$ permanecem estáveis. Já quando $\mu = 0,12$ com $e_p = 0,03$ e $\mu = 0,22$ com $e_p = 0,007$, somente partículas com valores de excentricidade próximos a 1 sobrevivem.

Considerando que partículas com $e > 0,9$ se tornam instáveis devido a colisões com o corpo central, a excentricidade máxima possível para o corpo perturbador exibe mudanças significativas para certos valores de μ . Para valores de μ no intervalo de $0,12$ até $0,16$, a excentricidade máxima $e_{\max}$ varia de $0,04$ até valores no intervalo de $e_{\max} = [0,19 - 0,25]$. Quando $\mu = 0,17$, há um aumento para $e_{\max} = 0,036$. Para o caso de $\mu = 0,18$ e $\mu = 0,19$, os valores de $e_{\max}$ caem para $0,014$, já para um sistema com $\mu = 0,20$ e $\mu = 0,21$, temos $e_{\max} = 0,014$. Finalmente, para $\mu = 0,22$, alcançamos $e_{\max}$ igual a $0,007$.

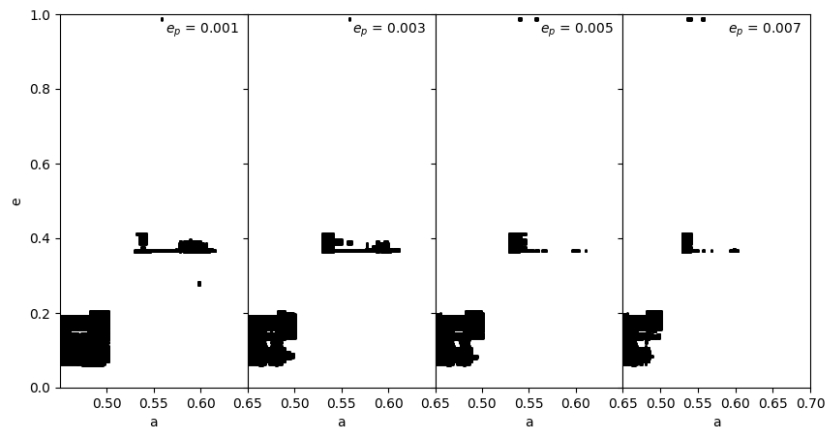
Figura 39 – Mapas de estabilidade ($e \times a$) exibindo a evolução da região da *Sailboat* para diferentes excentricidades do corpo perturbador, para sistemas binários com razões de massa iguais a $\mu = 0,05$, $0,12$ e $0,22$.



(a) $\mu = 0,05$



(b) $\mu = 0,12$



(c) $\mu = 0,22$

4.6 Análise da região *Sailboat* para diferentes valores da inclinação da partícula

Com o intuito de analisar o limite da região da *Sailboat* de acordo com a variação da inclinação orbital da partícula (I), foi estudado um conjunto de dados com 300.000 amostras. Os parâmetros livres foram definidos pela Tabela 14:

Tabela 14 – Intervalos de parâmetros livres analisados para o caso da variação da inclinação orbital da partícula.

Parâmetros livres	Intervalo
μ	0,01 - 0,23
e	0 - 0,99
a	0,45 - 0,7
I	0° - 180°

A amostra é composta por 20,11% de sistemas estáveis e 79,89% instáveis, nos quais 79,75% das partículas foram ejetadas e apenas 0,14% colidiu. Foi desenvolvida a busca de hiperparâmetros e *threshold* para esse caso, assim com a aplicação da técnica *Synthetic Minority Oversampling Technique (SMOTE)* demonstrou melhora no desempenho do modelo.

4.6.1 Busca de Hiperparâmetros e *Threshold*

A busca de hiperparâmetros foi empregada com o uso de *Grid Search*. As grades de configurações implementadas neste caso estão descritas na Tabela 11, e os hiperparâmetros selecionados como aqueles que melhor otimizam o modelo são exibidos na Tabela 15.

Tabela 15 – Resultados para o *hyperparameter tuning*, obtidos através de *Grid Search* para o *XGBoost*.

Hiperparâmetros	Valores selecionados
booster	gbtree
eta	0.6
scale_pos_weight	1.0
max_depth	9

As Figuras 40 e 41 exibem como a alteração dos valores dos hiperparâmetros influencia o desempenho do modelo. Os gráficos exibem a variação das métricas *accuracy* (vermelho), *recall* e *precision* para a classe estável (verde e roxo), e *recall* e *precision* para a classe instável (azul e rosa).

O hiperparâmetro **eta** apresentado na Figura 40, mostra que a porcentagem das métricas para a classe instável sofrem uma pequena variação em torno de 98%, enquanto

para a classe estável, a *recall* atinge 90% após 0,5 e a *precision* se mantém constante em 93% e cai somente para *eta* igual a 0,9. Valores para esse hiperparâmetro em torno de 0,5 e 0,8 indicam uma taxa de aprendizado mais acelerado, permitindo que o modelo se ajuste mais rapidamente aos dados.

O comportamento para o *scale_pos_weight* demonstra uma alternância maior, as curvas para a *precision* e *recall* das classes estável e instável coincidem em 0,7. Uma vez que o *scale_pos_weight* ajusta o equilíbrio entre as classes, a escolha do valor 0,7 caracteriza uma leve correção nos pesos da classe minoritária em relação a majoritária.

Para o *max_depth* (Figura 41) todas as métricas crescem e se estabilizam em torno do *max_depth* igual à 7, sendo os valores ideais em torno de 8 ou 9.

Figura 40 – Variação das métricas de acordo com os valores dos hiperparâmetros *eta* e *scale pos weight*.

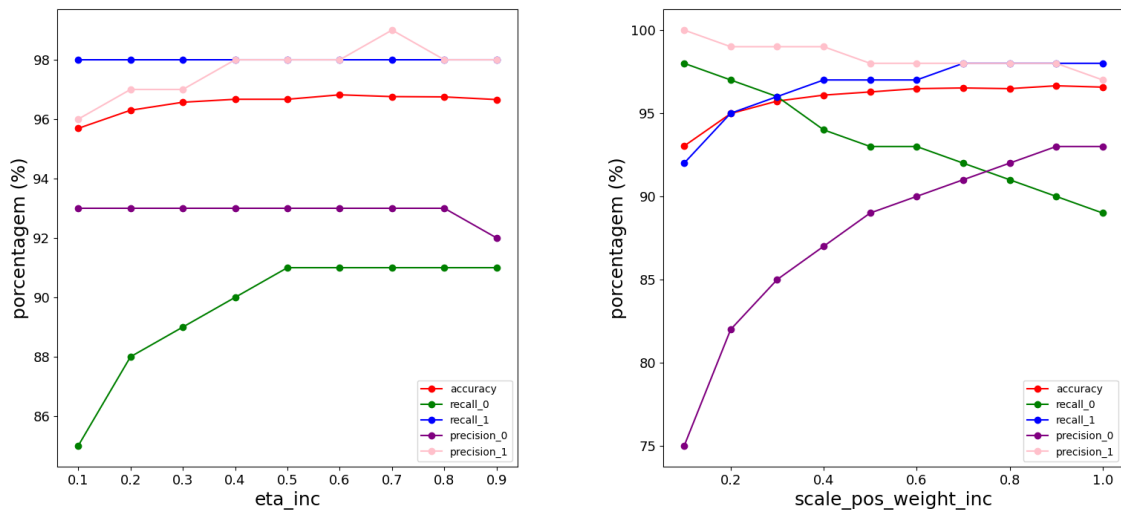
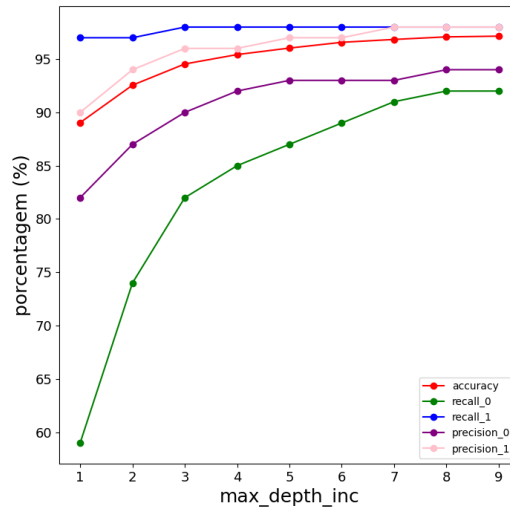
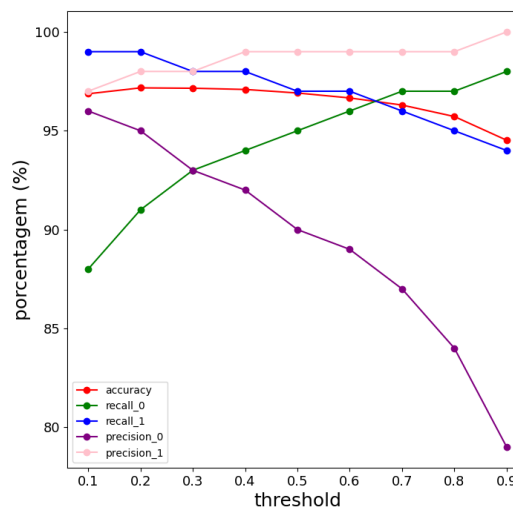


Figura 41 – Variação das métricas de acordo com os valores do hiperparâmetro *max depth*.

O *threshold* foi variado entre 0 e 1 com passo igual à 0,1. Os resultados apresentados a seguir utilizam os valores de hiperparâmetros exibidos na Tabela 15, assim como o ajuste realizado com a utilização da técnica de sobreamostragem *SMOTE*.

A Figura 42 exibe o desempenho das métricas de acordo com os valores de *threshold*. As curvas em roxo e verde representam a *recall* e a *precision* para a classe estável, e as curvas em rosa e azul, a *recall* e *precision* para a classe instável. Observamos que as quatro curvas citadas se encontram entre si no ponto entre os valores 0,3 e 0,4. Levando a escolha do *threshold* igual a 0,31 para esse caso, uma vez que após esse valor, a *precision* para a classe estável decai, indicando que a capacidade do modelo em minimizar Falsos Positivos diminui conforme o valor do *threshold* aumenta.

Figura 42 – Variação das métricas de acordo com os valores do *threshold*.

4.6.2 Resultados apresentados pelo *XGBoost*

O melhor modelo para o *XGBoost* alcançado obteve uma *accuracy* igual a 97,15%, com uma *precision* de 93% para os casos estáveis e uma *recall* de 98% para os casos instáveis. Empregando o modelo, foram gerados mapas das condições iniciais ($a \times e$) para intervalos da inclinação da partícula de $I = [0^\circ - 180^\circ]$, com um passo de $\Delta I = 1^\circ$. O processo foi repetido para cada $\mu \leq 0,22$, com um passo de $\Delta\mu = 0,01$. A figura 43 exibe onde pode ser encontrada a região estável considerando o intervalo $a = [0,4 - 0,7]$ para diferentes valores de razão de massa e inclinação orbital da partícula.

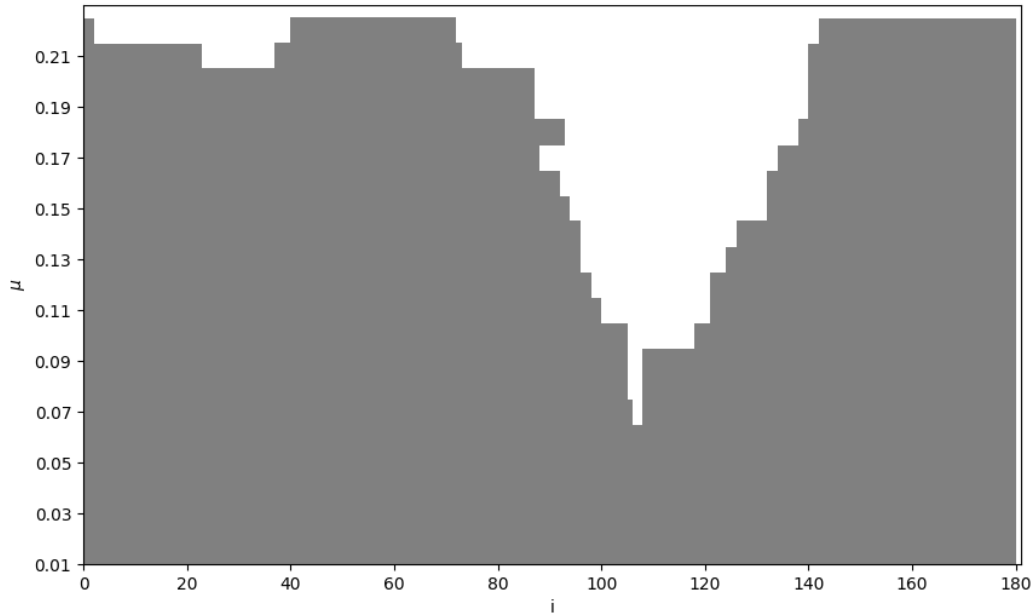
Para o caso de $\mu = 0,22$, a região da *Sailboat* é extremamente pequena, dessa forma um aumento da inclinação de apenas 2° é suficiente para fazer desaparecer essa região inteira. Para $\mu = 0,21$, a *Sailboat* sobrevive com uma inclinação de até 20° . Em ambos os casos, novas regiões estáveis começam a aparecer para os valores de inclinação em torno de 35° , até aproximadamente 70° . A medida que a inclinação se aproxima de $I \sim 140^\circ$, um regime retrógrado surge, crescendo até atingir sua estrutura máxima em $I = 180^\circ$.

Quando $\mu < 0,2$, a região da *Sailboat* é robusta em relação as variações da inclinação da partícula e pode ser encontrada até inclinações próximas a 90° . Para μ entre 0,17 e 0,2, essa região pode sobreviver até inclinações de $I = [85^\circ - 95^\circ]$. A região da *Sailboat* também pode existir para órbitas retrógradas onde $\mu \leq 0,16$. Quando a razão de massa se encontra entre 0,11 e 0,16, a região pode permanecer estável para inclinações no intervalo $I = [90^\circ - 100^\circ]$. Esses resultados são consistentes com o que foi apresentado em [Giulianti Winter et al. \(2013\)](#), cujo trabalho demonstra que a região da *Sailboat* pode sobreviver no sistema Plutão-Charonte para inclinações de até $I = 100^\circ$.

A região da *Sailboat* sobrevive até $I \sim 105^\circ$ para razões de massa no intervalo $\mu = [0,07 - 0,1]$. Conforme o valor de μ diminui, uma região retrógrada aparece em inclinações cada vez mais baixas. Para $\mu = [0,15 - 0,2]$ a região se encontra no intervalo de $I = [130^\circ - 140^\circ]$, quando $\mu = [0,11 - 0,14]$, essa região surge no intervalo $I = [120^\circ - 130^\circ]$, por fim em $\mu = [0,07 - 0,1]$, uma região retrógrada aparece quando $I = [108^\circ - 118^\circ]$.

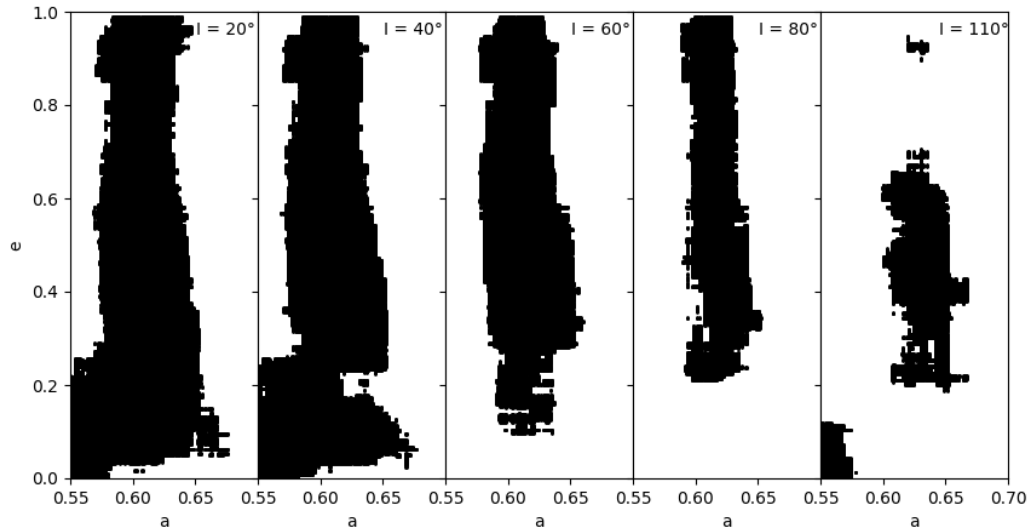
Para μ igual a 0,05 e 0,06, a região da *Sailboat* diminui de tamanho conforme a inclinação decresce, mas nunca desaparece, atingindo seu tamanho mínimo em uma inclinação específica e se expandindo novamente. Em ambos os casos, a região da *Sailboat* alcança seu tamanho mínimo em $I = 110^\circ$.

Figura 43 – Regiões de estabilidade para diferentes inclinações orbitais da partícula. As áreas em cinza indicam os valores para a inclinação onde regiões estáveis podem ser encontradas entre o intervalo $a = [0,4 - 0,7]$ para diferentes valores de μ .

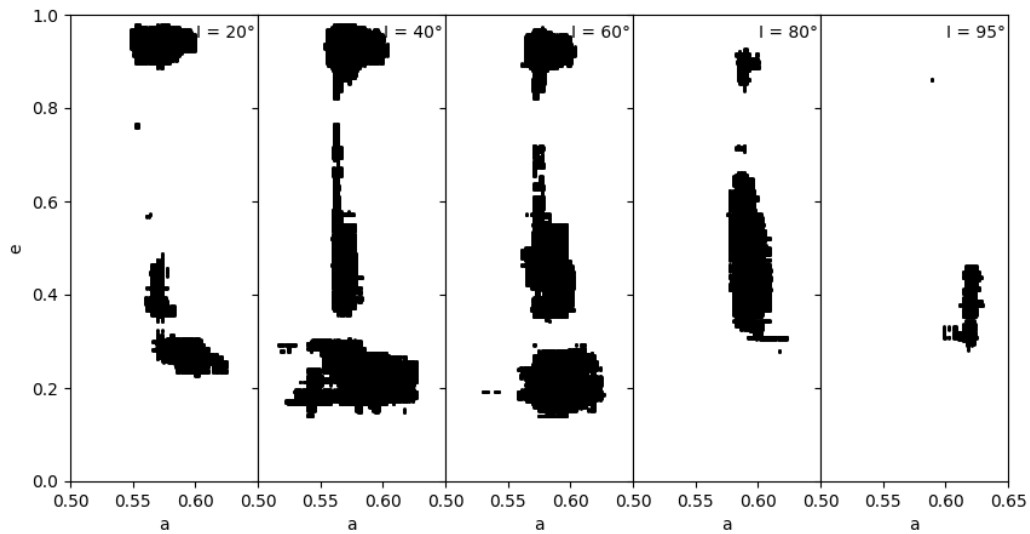


A Figura 44 mostra a evolução da *Sailboat* para duas razões de massa diferentes, $\mu = 0,05$ e $0,12$, considerando as inclinações orbitais da partícula iguais a 30° , 60° , 90° , 95° (para $\mu = 0,12$) e 110° (para $\mu = 0,05$).

Figura 44 – Mapas de estabilidade ($e \times a$) exibindo a evolução da região da *Sailboat* variando as inclinações das partículas para sistemas binários com razões de massa de $\mu = 0,05, 0,12$.



(a) $\mu = 0,05$



(b) $\mu = 0,12$

4.7 Análise da região *Sailboat* com a variação do argumento do pericentro

Nesta seção, nós exploramos como a variação do argumento do pericentro da partícula afeta a região da *Sailboat*. O conjunto de dados utilizado possuía 300.000 amostras, das quais 11,5% das partículas sobreviveram durante toda a simulação, 87,32% foram

ejetadas e 1,18% colidiram com o corpo perturbador. O conjunto possuía os parâmetros livres dados pela Tabela 16:

Tabela 16 – Parâmetros livres analisados para o caso da variação do argumento do peri-centro.

Parâmetros livres	Intervalo
μ	0,01 - 0,23
a	0,4 - 0,7
e	0 - 0,99
ω	0° - 360°

Para essa amostra realizamos a busca de hiperparâmetros e *threshold* e empregamos a técnica *Random Oversampling* a fim de alcançar o modelo de *XGBoost* mais ótimo.

4.7.1 Busca de Hiperparâmetros e *Threshold*

A busca de hiperparâmetros foi empregada com o uso de *Grid Search* para o *XGBoost*. As grades de configurações implementadas neste caso estão descritas na Tabela 11, e os hiperparâmetros selecionados como aqueles que melhor otimizam o modelo são exibidos na Tabela 17.

Tabela 17 – Resultados para o *hyperparameter tuning*, obtidos através de *Grid Search* para o *XGBoost*.

Hiperparâmetros	Valores selecionados
booster	gbtree
eta	0.6
scale_pos_weight	1.0
max_depth	9

As Figuras 45 e 46 exibem como a alteração dos valores dos hiperparâmetros alteram o desempenho do modelo. Os gráficos exibem a variação das métricas *accuracy* (vermelho), *recall* e *precision* para a classe estável (verde e roxo), e *recall* e *precision* para a classe instável (azul e rosa).

Nestes gráficos, apenas o valor do hiperparâmetro indicado foi modificado e os outros parâmetros permaneceram no padrão do algoritmo, assim como nenhuma técnica de sobreamostragem foi empregada.

A porcentagem das métricas para o hiperparâmetro **eta** para a classe instável é praticamente constante em torno de 99%, enquanto para a classe estável, a *recall* atinge 91% após 0,5 e a *precision* se mantém constante em 94% no intervalo entre 0,5 e 0,8.

O comportamento para o **scale_pos_weight** demonstra uma pequena variação das curvas para a *precision* e *recall* da classe, mantendo-se constante em 98% no intervalo

de 0,5 a 1,0, enquanto para a classe estável há uma grande variação, sendo que as curvas para a *recall* e *precision* se encontram em 0,7.

A Figura 30 exibe os valores para o *max_depth*, onde todas as métricas crescem e se estabilizam em torno de 7.

Figura 45 – Variação das métricas de acordo com os hiperparâmetros *eta* e *scale pos weight*.

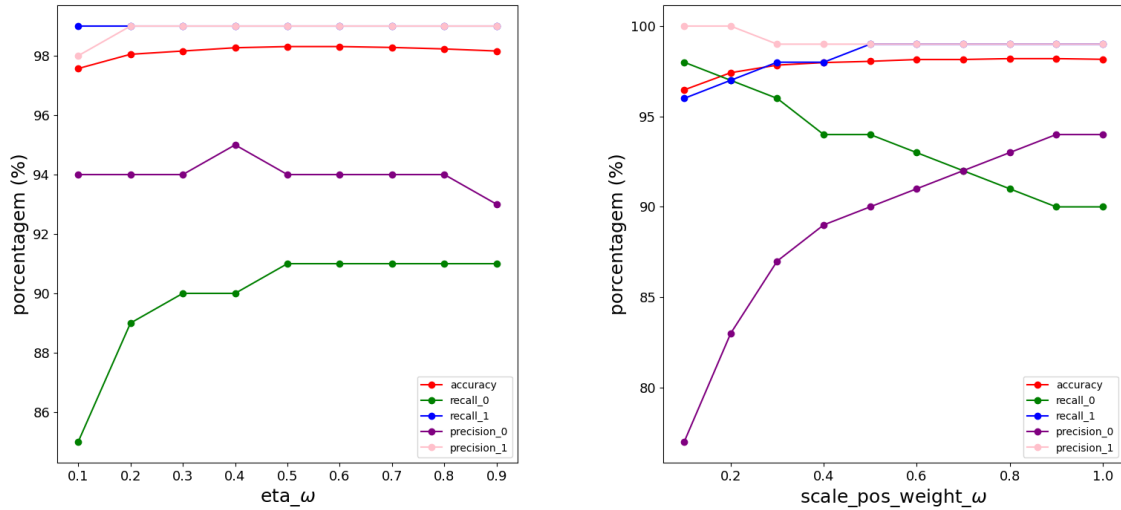
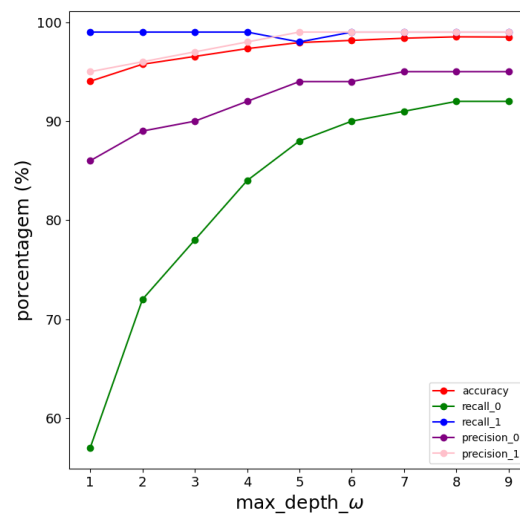


Figura 46 – Variação das métricas de acordo com os valores do hiperparâmetro *max depth*.

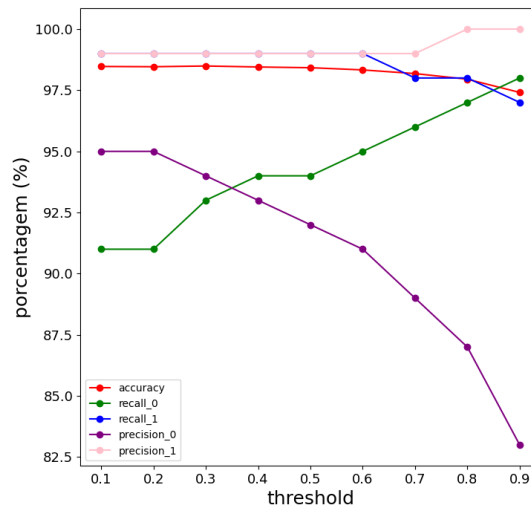


A busca do *threshold* foi empregada no intervalo de 0 a 1 com passo igual a 0,1. Os resultados apresentados a seguir utilizam os valores de hiperparâmetros exibidos na Tabela 17 e o ajuste com a utilização da técnica de sobreamostragem *Random Oversampling*.

A Figura 42 exibe o desempenho das métricas de acordo com os valores de *threshold*. As curvas em roxo e verde representam a *recall* e a *precision* para a classe estável, e as

curvas em rosa e azul a *recall* e a *precision* para a classe instável. Observamos que as quatro curvas citadas se encontram entre si no ponto entre os valores 0,3 e 0,4. Levando a escolha do *threshold* igual a 0,31 para esse caso, uma vez que após esse valor, a *precision* para a classe estável decai, indicando que a capacidade do modelo em minimizar Falsos Positivos diminui conforme o valor do *threshold* aumenta.

Figura 47 – Variação das métricas de acordo com os valores do *Threshold*.



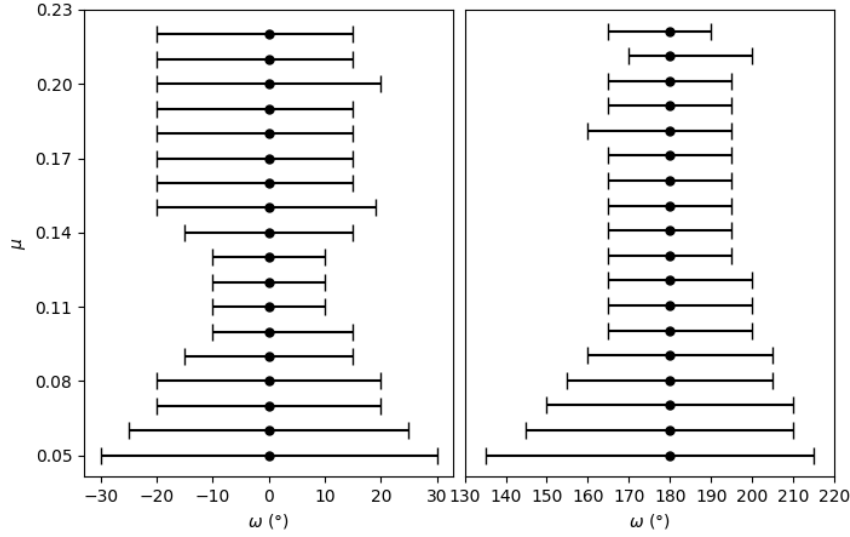
4.7.2 Resultados obtidos com o *XGBoost*

O modelo otimizado para o algoritmo *XGBoost* obteve uma *accuracy* igual a 98,50%, já a *recall* e a *precision* para a classe estável foram equivalentes a 94% e 93%, respectivamente. Para o caso instável, a *recall* e a *precision* foram iguais a 99%.

Giuliatti Winter et al. (2014) demonstram que a ilha da *Sailboat* existe entre o sistema Plutão-Caronte em dois intervalos específicos para o argumento do pericentro ($\Delta\omega$): $\Delta\omega = \pm 10^\circ$ com tamanho máximo em $\omega = 0^\circ$; e $\Delta\omega = \pm 20^\circ$ centrado ao redor de $\omega = 180^\circ$. Nossos resultados também revelam que a região da *Sailboat* pode ser encontrada nesses dois intervalos de ω , atingindo a máxima extensão e em $\omega = 0^\circ$ e 180° , independente da razão de massa.

Na Figura 48, as barras representam o intervalo aproximado do argumento do pericentro onde existe a região da *Sailboat* de acordo com o valor de μ . O gráfico à esquerda mostra os valores de ω ao redor de 0° , e o gráfico à direita ao redor de 180° .

Figura 48 – Intervalo do argumento do pericentro da partícula onde existe a região da *Sailboat* com máxima extensão em $\omega = 0^\circ$ (gráfico à esquerda) e $\omega = 180^\circ$ (gráfico à direita).



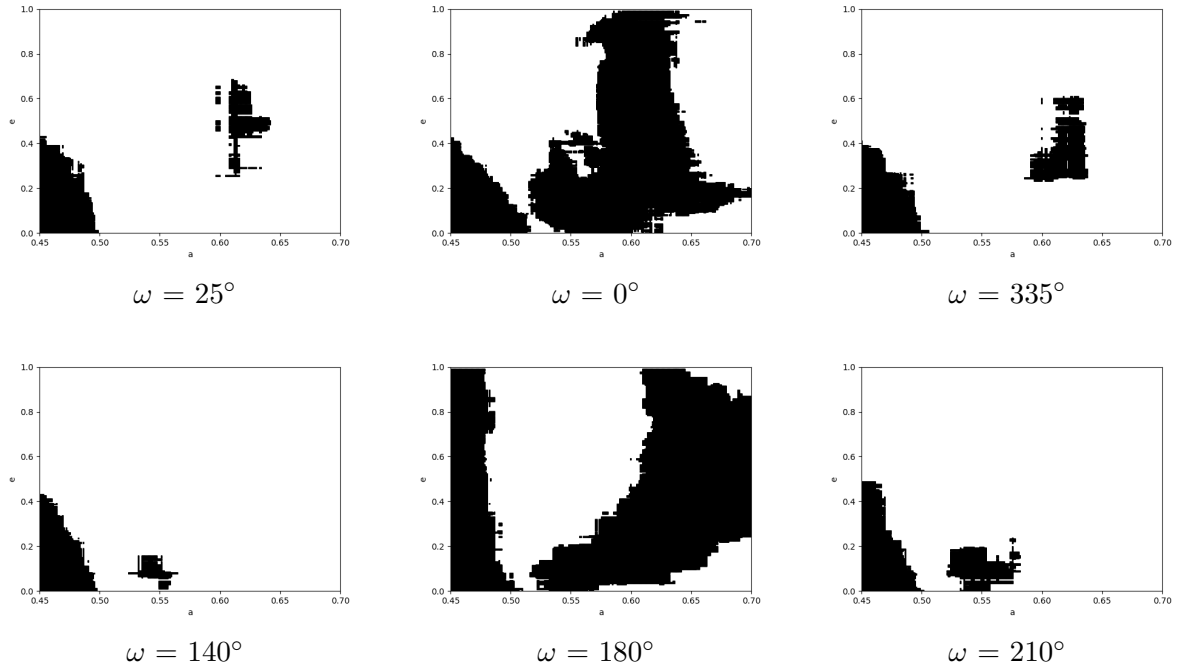
Na Figura 48, para certos valores de μ , os intervalos do argumento do pericentro são assimétricos ao redor de 0° ou 180° , diferente do resultado de [Giulianti Winter et al. \(2014\)](#). É possível que essa diferença seja resultado da generalização do algoritmo, que pode apresentar a classificação incorreta de algumas partículas estáveis, principalmente devido ao número de partículas estáveis se tornar muito pequeno no limite de ω .

O tamanho desses intervalos não aumenta proporcionalmente com μ , para valores de ω ao redor de 0° , quando $\mu < 0,08$, o intervalo vai de $\Delta\omega$ aproximadamente $\pm 20^\circ$ até $\pm 30^\circ$. Quando μ está entre 0,1 e 0,14, esse intervalo é reduzido para aproximadamente $\pm 10^\circ$, coerente com o resultado de [Giulianti Winter et al. \(2014\)](#) para o sistema Plutão-Caronte. Além disso, para $\mu \geq 0,15$, $\Delta\omega$ aumenta novamente para cerca de $\pm 20^\circ$.

Considerando um intervalo centrado em $\omega = 180^\circ$, a região da *Sailboat* pode sobreviver entre diferentes intervalos de $\Delta\omega$, dependendo do valor de μ . Para $\mu = 0,05$, o intervalo é igual a $\omega = [135^\circ - 215^\circ]$, quando μ é 0,06, o intervalo decresce para $\omega = [145^\circ - 210^\circ]$. Conforme μ aumenta para 0,07, o intervalo se torna $\pm 30^\circ$, e para $\mu = 0,08$, ele se estreita ainda mais para $\pm 25^\circ$, já quando $\mu \leq 0,09$, o intervalo é limitado para $\omega = [160^\circ - 205^\circ]$. No intervalo em que $0,14 \geq \mu \geq 0,10$, a região da *Sailboat* existe entre $\omega = [165^\circ - 200^\circ]$, para $\mu \geq 0,15$, o intervalo está ao redor de $\pm 15^\circ$.

A Figura 49 mostra a variação da região da *Sailboat* para diferentes argumentos do pericentro da partícula.

Figura 49 – Diagrama das condições iniciais ($e \times a$) para um sistema binário com $\mu = 0,05$ e diferentes valores do argumento do pericentro.



4.8 Comentários Finais

A *Sailboat* é uma região estável peculiar, característica de órbitas do tipo-S ao redor do corpo mais massivo em um sistema binário. O objetivo desse capítulo refletiu-se na aplicação do algoritmo *XGBoost* para estudar a estabilidade da região da *Sailboat* e analisar o seu limite de existência, mediante quatro abordagens diferentes. Além disso, avaliar o desempenho do algoritmo de acordo com a busca de hiperparâmetros, *threshold* e avaliação das métricas, com o intuito de atingir o modelo mais eficaz para cada conjunto de dados específico.

Foram desenvolvidas 300.000 simulações do problema de três corpos elíptico para cada uma das quatro abordagens. Esses conjuntos de dados foram utilizados como conjunto de treino do modelo de *Machine Learning* para prever a estabilidade de partículas em sistemas binários.

Para o primeiro conjunto, variamos a razão de massa do sistema binário, o semieixo maior da partícula e a excentricidade. Foi atingida uma *accuracy* igual a 98,54%, e uma *recall* e *precision* superior a 97% para todas as classes. Nossos resultados demonstraram que a região da *Sailboat* existe em qualquer sistema binário com $\mu = [0,05 - 0,22]$. Para $\mu = 0,05$, a *Sailboat* se encontra em $a = [0,5 - 0,7]$, já para $\mu = 0,22$, a região existe em $a = [0,5 - 0,55]$ e $e = [0,3 - 0,4]$. Uma nova região surge quando $\mu = 0,22$ e se funde com a *Sailboat* conforme a razão de massa cresce.

O segundo conjunto incluiu variações na excentricidade do corpo perturbador, de 0 a 0,99, assim como os parâmetros: e , μ e a . A aplicação do XGBoost levou a uma *accuracy* igual à 99,55%, uma *recall* de 87% para a classe estável e *precision* de aproximadamente 100% para a classe instável. Para essa amostra, a razão de massa igual a $\mu = 0,05$ leva a excentricidade máxima da região *Sailboat* de aproximadamente 0,23. Quando $\mu = 0,22$, a excentricidade máxima foi igual a $e_p \sim 0,007$. Além disso, verificamos que pequenas mudanças na excentricidade do corpo perturbador são suficientes para que a região desapareça.

A terceira abordagem variou a inclinação orbital da partícula, $I = [0^\circ - 180^\circ]$. O melhor modelo para o XGBoost alcançado obteve uma *accuracy* igual à 97,15%, e o desempenho da *recall* e *precision* para a classe estável e instável foram 97% e 93%, respectivamente. Para esse caso, a região da *Sailboat* sobrevive mesmo para órbitas retrógradas, com $\mu \leq 0,16$, quando $\mu \leq 0,6$ a região diminui de tamanho, mas não desaparece completamente.

Por fim, a quarta amostra examinou variações no argumento do pericentro da partícula, $\omega = [0^\circ - 360^\circ]$. Nesse caso, foi alcançada uma *accuracy* igual à 98,5%, a *recall* e a *precision* para a classe estável e instável foram equivalentes a 94% e 99%, respectivamente. Os resultados demonstraram que a região da *Sailboat* existe para dois intervalos de ω , variando de tamanho de acordo com os valores de μ , a máxima extensão ocorreu quando $\omega = 0^\circ$ e 180° .

5 Considerações finais

O propósito deste estudo baseou-se na atuação de modelos de *Machine Learning* para gerar mapas de predição de regiões de estabilidade de partículas em órbitas do tipo-S ao redor de planetas hipotéticos (Capítulo 3), assim como avaliar a estabilidade da região da *Sailboat*, variando parâmetros físicos e restringindo seu intervalo de existência (Capítulo 4). Dessa maneira, avaliamos o desempenho dos algoritmos segundo a variação de hiperparâmetros e *threshold* para cada amostra analisada, a fim de determinar o modelo com as melhores métricas em relação ao conjunto de dados disponível.

Para o primeiro cenário, o estudo explorou uma grande gama de parâmetros orbitais e físicos do problema de três corpos. No total, foram integrados 10^5 sistemas numéricos classificando as partículas em estáveis e instáveis. A amostra de dados construída foi empregada para compor o conjunto de treino para os algoritmos de ML. Nós testamos cinco algoritmos de ML: *Random Forest*, *Decision Tree*, *XGBoost*, *Light Gradient Boosting* e *Histogram Gradient Boosting*. O *XGBoost* demonstrou ser o modelo com desempenho superior, com uma *accuracy* igual a 98,48%.

Por fim, com a intenção de verificar a eficácia do modelo obtido, reproduzimos os resultados numéricos dos trabalhos de Domingos et al. (2006), Pinheiro and Sfair (2021) e do grupo de satélites Inuit de Saturno, obtendo uma acurácia de até 96,73%.

Para a segunda análise, foram desenvolvidas milhares de simulações do problema de três corpos elíptico para quatro casos diferentes. Dessa forma, foi verificado como o limite da região da *Sailboat* varia de acordo com diferentes razões de massa do sistema binário, excentricidades iniciais do corpo perturbador, inclinações orbitais da partícula e argumentos do pericentros da partícula. Uma vez que o *XGBoost* apresentou maior desempenho, o mesmo foi empregado para realizar a predição dos quatro conjuntos avaliados. No total, realizamos a previsão da ordem de 10^9 condições iniciais, atingindo uma *accuracy* superior a 97% para todos os casos.

O *XGBoost* pode perder desempenho na classificação de pontos de fronteira como observamos em alguns casos de aplicação desse trabalho, que tiverem uma *accuracy* menor. Entretanto, o modelo cumpriu de forma eficaz o objetivo do estudo em identificar regiões de estabilidade para diferentes cenários com milhares de condições iniciais.

Os algoritmos de *Machine Learning* demonstraram ser uma poderosa ferramenta para redução do tempo e gasto computacional requeridos para simulações numéricas de problema de três corpos. Um mapa de estabilidade cobrindo uma grande gama de parâmetros físicos e orbitais e com milhares de condições iniciais pode ser gerado em apenas um minuto.

Como resultado do trabalho, está em desenvolvimento um site cujo objetivo é realizar a previsão de mapas de estabilidades para condições iniciais oferecidas pelo usuário, utilizando o treinamento do *XGBoost* salvo a partir das simulações numéricas construídas.

REFERÊNCIAS

- E. Ashton, B. Gladman, and M. Beaudoin. Evidence for a recent collision in saturn’s irregular moon population. *The Planetary Science Journal*, 2(4):158, 2021.
- J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13:281–305, 03 2012.
- B. Bischl, O. Mersmann, H. Trautmann, and C. Weihs. Resampling methods for meta-model validation with recommendations for evolutionary computation. *Evolutionary computation*, 20:249–75, 02 2012. doi: 10.1162/EVCO_a_00069.
- I. Bondarenko and E. Perevozkina. VizieR online data catalog: Catalog of early-type contact binaries (bondarenko+, 1996). *VizieR Online Data Catalog*, pages V–120, 2004.
- L. Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.
- L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001.
- R. A. Broucke. Periodic orbits in the restricted three-body problem with earth-moon masses. Technical Report 32-1168, Jet Propulsion Laboratory, California Institute of Technoly, 1968.
- M. E. Brown and B. J. Butler. Masses and densities of dwarf planet satellites measured with alma. *The Planetary Science Journal*, 4(10):193, 2023.
- M. Brozović and R. A. Jacobson. Post-new-horizons orbits and masses for the satellites of pluto. *The Astronomical Journal*, 167(6):256, 2024.
- M. Brozović and R. A. Jacobson. Post-new-horizons orbits and masses for the satellites of pluto. *The Astronomical Journal*, 167(6):256, 2024.
- V. Carruba, J. A. Burns, P. D. Nicholson, and B. J. Gladman. On the inclination distribution of the jovian irregular satellites. *Icarus*, 158(2):434–449, 2002. ISSN 0019-1035. doi: <https://doi.org/10.1006/icar.2002.6896>. URL <https://www.sciencedirect.com/science/article/pii/S001910350296896X>.
- V. Carruba, S. Aljbaae, R. C. Domingos, A. Lucchini, and P. Furlaneto. Machine learning classification of new asteroid families members. *Monthly Notices of the Royal Astronomical Society*, 496(1):540–549, 05 2020. ISSN 0035-8711. doi: 10.1093/mnras/staa1463. URL <https://doi.org/10.1093/mnras/staa1463>.

- V. Carruba, S. Aljbaae, R. C. Domingos, and W. Barletta. Artificial neural network classification of asteroids in the M1:2 mean-motion resonance with Mars. *Monthly Notices of the Royal Astronomical Society*, 504(1):692–700, 04 2021a. ISSN 0035-8711. doi: 10.1093/mnras/stab914. URL <https://doi.org/10.1093/mnras/stab914>.
- V. Carruba, S. Aljbaae, and R. D. C. Domingos. Identification of asteroid groups in the z_1 and z_2 nonlinear secular resonances through genetic algorithms. *Celestial Mechanics and Dynamical Astronomy*, 133, 06 2021b. doi: 10.1007/s10569-021-10021-z.
- V. Carruba, S. Aljbaae, R. C. Domingos, M. Huaman, and W. Barletta. Machine learning applied to asteroid dynamics. *Celestial Mechanics and Dynamical Astronomy*, 134(4), Aug. 2022. ISSN 1572-9478. doi: 10.1007/s10569-022-10088-2. URL <http://dx.doi.org/10.1007/s10569-022-10088-2>.
- V. Carruba, S. Aljbaae, G. Caritá, M. V. F. Lourenço, B. S. Martins, and A. A. Alves. Imbalanced classification applied to asteroid resonant dynamics. *Frontiers in Astronomy and Space Sciences*, 10, 2023a. ISSN 2296-987X. doi: 10.3389/fspas.2023.1196223. URL <https://www.frontiersin.org/journals/astronomy-and-space-sciences/articles/10.3389/fspas.2023.1196223>.
- V. Carruba, J. I. B. Camargo, S. Aljbaae, . co authors, and D. Collaboration. Main belt asteroids taxonomical information from dark energy survey data, 2023b. URL <https://arxiv.org/abs/2311.03613>.
- N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- P.-W. Chen, J.-Y. Wang, and H.-M. Lee. Model selection of svms using ga approach. In *2004 ieee international joint conference on neural networks (ieee cat. no. 04ch37541)*, volume 3, pages 2035–2040. IEEE, 2004.
- T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
- M. Claesen, J. Simm, D. Popovic, Y. Moreau, and B. D. Moor. Easy hyperparameter search using optunity. *CoRR*, abs/1412.1114, 2014. URL <http://arxiv.org/abs/1412.1114>.
- M. Cuntz. S-type and p-type habitability in stellar binary systems: A comprehensive approach. i. method and applications. *The Astrophysical Journal*, 780(1):14, 2013.
- A. Cutler, D. Cutler, and J. Stevens. *Random Forests*, volume 45, pages 157–176. Springer, 01 2011. ISBN 978-1-4419-9325-0. doi: 10.1007/978-1-4419-9326-7_5.

- R. C. Domingos, O. C. Winter, and T. Yokoyama. Stable satellites around extrasolar giant planets. *Monthly Notices of the Royal Astronomical Society*, 373(3):1227–1234, 11 2006. ISSN 0035-8711. doi: 10.1111/j.1365-2966.2006.11104.x. URL <https://doi.org/10.1111/j.1365-2966.2006.11104.x>.
- J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
- T. Fushiki. Estimation of prediction error by using k-fold cross-validation. *Statistics and Computing*, 21:137–146, 2011.
- A. Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. "O'Reilly Media, Inc.", 2022.
- S. Giuliatti Winter, O. Winter, E. Vieira Neto, and R. Sfair. Stable regions around pluto. *Monthly Notices of the Royal Astronomical Society*, 430(3):1892–1900, 2013.
- S. M. Giuliatti Winter, O. Winter, E. Vieira Neto, and R. Sfair. A peculiar stable region around pluto. *Monthly Notices of the Royal Astronomical Society*, 439(4):3300–3307, 2014.
- D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1989. ISBN 0201157675.
- T. Grav and J. Bauer. A deeper look at the colors of the saturnian irregular satellites. *Icarus*, 191(1):267–285, 2007.
- W. Grundy, S. Porter, S. Benecchi, H. Roe, K. Noll, C. Trujillo, A. Thirouin, J. Stansberry, E. Barker, and H. Levison. The mutual orbit, mass, and density of the large transneptunian binary system varda and ilmarë. *Icarus*, 257:130–138, 2015.
- W. Grundy, K. Noll, H. Roe, M. Buie, S. Porter, A. Parker, D. Nesvorný, H. Levison, S. Benecchi, D. Stephens, et al. Mutual orbit orientations of transneptunian binaries. *Icarus*, 334:62–78, 2019.
- D. P. Hamilton and J. A. Burns. Orbital stability zones about asteroids II. The destabilizing effects of eccentric orbits and of solar radiation. , 96(1):43–64, Mar. 1992. doi: 10.1016/0019-1035(92)90005-R.
- H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.
- B. Holler, W. Grundy, M. Buie, and K. Noll. The eris/dysnomia system i: The orbit of dysnomia. *Icarus*, 355:114130, Feb. 2021. ISSN 0019-1035. doi: 10.1016/j.icarus.2020.114130. URL <http://dx.doi.org/10.1016/j.icarus.2020.114130>.

- M. J. Holman and P. A. Wiegert. Long-term stability of planets in binary systems. *The Astronomical Journal*, 117(1):621, 1999.
- F. Hutter, L. Kotthoff, and J. Vanschoren, editors. *Automatic machine learning: methods, systems, challenges*. Challenges in Machine Learning. Springer, Germany, 2019. ISBN 978-3-030-05317-8. doi: 10.1007/978-3-030-05318-5.
- M. Ibrahim, H. AbdelRaouf, K. M. Amin, and N. Semary. Keystroke dynamics based user authentication using histogram gradient boosting. *IJCI. International Journal of Computers and Information*, 10(1):36–53, 2023.
- M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami. Multi-split optimized bagging ensemble model selection for multi-class educational data mining. *CoRR*, abs/2006.05031, 2020. URL <https://arxiv.org/abs/2006.05031>.
- G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.
- C. Kiss, G. Marton, A. H. Parker, W. M. Grundy, A. Farkas-Takács, J. Stansberry, A. Pal, T. Müller, K. S. Noll, M. E. Schwamb, et al. The mass and density of the dwarf planet (225088) 2007 or10. *Icarus*, 334:3–10, 2019.
- C. Lam and D. Kipping. A machine learns to predict the stability of circumbinary planets. *Monthly Notices of the Royal Astronomical Society*, 476(4):5692–5697, 2018.
- T. T. Le, W. Fu, and J. H. Moore. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, 36(1):250–256, 06 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz470. URL <https://doi.org/10.1093/bioinformatics/btz470>.
- X. Li, J. Li, Z. J. Xia, and N. Georgakarakos. Machine-learning prediction for mean motion resonance behaviour – The planar case. *Monthly Notices of the Royal Astronomical Society*, 511(2):2218–2228, 01 2022. ISSN 0035-8711. doi: 10.1093/mnras/stac166. URL <https://doi.org/10.1093/mnras/stac166>.
- C. J. Lintott, K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. , 389(3):1179–1189, Sept. 2008. doi: 10.1111/j.1365-2966.2008.13689.x.
- T. Mitchell. Learning, machine. *Publisher: McGraw Hill*, 1997.

- B. Morgado, B. Sicardy, F. Braga-Ribas, J. Ortiz, H. Salo, F. Vachier, J. Desmars, C. Pereira, P. Santos-Sanz, R. Sfair, et al. A dense ring of the trans-neptunian object quaoar outside its roche limit. *Nature*, 614(7947):239–243, 2023.
- E. V. Neto and O. Winter. Time analysis for temporary gravitational capture: satellites of uranus. *The Astronomical Journal*, 122(1):440, 2001.
- M. Ojala and G. C. Garriga. Permutation tests for studying classifier performance. In *2009 Ninth IEEE International Conference on Data Mining*, pages 908–913, 2009. doi: 10.1109/ICDM.2009.108.
- R. S. Olson, R. J. Urbanowicz, P. C. Andrews, N. A. Lavender, L. C. Kidd, and J. H. Moore. Automating biomedical data science through tree-based pipeline optimization. *CoRR*, abs/1601.07925, 2016. URL <http://arxiv.org/abs/1601.07925>.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- T. F. Pinheiro and R. Sfair. Constraining the nature of the possible extrasolar pds110b ring system. *Astronomy & Astrophysics*, 652:A149, 2021.
- P. Probst, B. Bischl, and A.-L. Boulesteix. Tunability: Importance of Hyperparameters of Machine Learning Algorithms. *arXiv e-prints*, art. arXiv:1802.09596, Feb. 2018. doi: 10.48550/arXiv.1802.09596.
- J. R. Quinlan. Induction of decision trees. *Machine learning*, 1:81–106, 1986.
- G. Rabl and R. Dvorak. Satellite-type planetary orbits in double stars : a numerical approach. , 191:385–391, Feb. 1988.
- D. Ragozzine and M. E. Brown. Orbits and masses of the satellites of the dwarf planet haumea (2003 el61). *The Astronomical Journal*, 137(6):4766, 2009.
- H. Rein and D. S. Spiegel. ias15: a fast, adaptive, high-order integrator for gravitational dynamics, accurate to machine precision over a billion orbits. *Monthly Notices of the Royal Astronomical Society*, 446(2):1424–1437, 11 2014. ISSN 0035-8711. doi: 10.1093/mnras/stu2164. URL <https://doi.org/10.1093/mnras/stu2164>.
- H. Rein and D. S. Spiegel. Ias15: a fast, adaptive, high-order integrator for gravitational dynamics, accurate to machine precision over a billion orbits. *Monthly Notices of the Royal Astronomical Society*, 446(2):1424–1437, 2015.
- S. Rieder and M. A. Kenworthy. Constraints on the size and dynamics of the j1407b ring system. *Astronomy & Astrophysics*, 596:A9, 2016.

- A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.
- J. Schneider, C. Dedieu, P. Le Sidaner, and I. Savalle, Renaud e Zolotukhin. Defining and cataloging exoplanets: the exoplanet. eu database. *Astronomy & Astrophysics*, 532:A79, 2011.
- S. Shalev-Shwartz, Shai e Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- S. S. Sheppard, Y. R. Fernandez, and A. Moullet. The albedos, sizes, colors, and satellites of dwarf planets compared with newly measured dwarf planet 2013 fy27. *The Astronomical Journal*, 156(6):270, 2018.
- R. A. Smullen and K. Volk. Machine learning classification of Kuiper belt populations. *Monthly Notices of the Royal Astronomical Society*, 497(2):1391–1403, 07 2020. ISSN 0035-8711. doi: 10.1093/mnras/staa1935. URL <https://doi.org/10.1093/mnras/staa1935>.
- S. V. Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, 62(1):77–89, 1997.
- L. Surkova and M. Svechnikov. VizieR online data catalog: Semi-detached eclipsing binaries (surkova+, 2004). *VizieR Online Data Catalog*, 2004.
- M. Svechnikov and E. Perevozkina. VizieR online data catalog: Catalog of dms-type eclipsing binaries (svechnikov+, 1999). *VizieR Online Data Catalog*, page 5, 2004.
- D. Tamayo, A. Silburt, D. Valencia, K. Menou, M. Ali-Dib, C. Petrovich, C. X. Huang, H. Rein, C. Van Laerhoven, A. Paradise, et al. A machine learns to predict the stability of tightly packed planetary systems. *The Astrophysical Journal Letters*, 832(2):L22, 2016.
- H. J. Weerts, A. C. Mueller, and J. Vanschoren. Importance of tuning hyperparameters of machine learning algorithms. *arXiv preprint arXiv:2007.07588*, 2020.
- O. Winter and E. Vieira Neto. Time analysis for temporary gravitational capture-stable orbits. *Astronomy & Astrophysics*, 377(3):1119–1127, 2001.
- S. G. Winter, O. Winter, A. F. Guimarães, and M. Silva. Exploring s-type orbits in the pluto–charon binary system. *Monthly Notices of the Royal Astronomical Society*, 404(1): 442–450, 2010.
- L. Yang and A. Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020. ISSN 0925-2312.

doi: <https://doi.org/10.1016/j.neucom.2020.07.061>. URL <https://www.sciencedirect.com/science/article/pii/S0925231220311693>.

Z. Zheng, Y. Cai, and Y. Li. Oversampling method for imbalanced classification. *Computing and Informatics*, 34(5):1017–1037, 2015.

Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*, volume 14. Chapman and Hall/CRC, 06 2012. ISBN 9781439830031. doi: 10.1201/b12207.

Q. Zou, S. Xie, Z. Lin, M. Wu, and Y. Ju. Finding the best classification threshold in imbalanced classification. *Big Data Research*, 5:2–8, 2016.