
Modelling the Doughnut of social and planetary boundaries with frugal machine learning

Stefano Vrizzi

UB School of Economics
Universitat de Barcelona
vrizzi@ub.edu

Daniel W. O'Neill

UB School of Economics
Universitat de Barcelona
oneill@ub.edu

Abstract

The ‘Doughnut’ of social and planetary boundaries has emerged as a popular framework for assessing environmental and social sustainability. Here, we provide a proof-of-concept analysis that shows how machine learning (ML) methods can be applied to a simple macroeconomic model of the Doughnut. First, we show how ML methods can be used to find policy parameters that are consistent with ‘living within the Doughnut’. Second, we show how a reinforcement learning agent can identify the optimal trajectory towards desired policies in the parameter space. The approaches we test, which include a Random Forest Classifier and Q -learning, are frugal ML methods that are able to find policy parameter combinations that achieve both environmental and social sustainability. The next step is the application of these methods to a more complex ecological macroeconomic model.

1 Introduction

1.1 The Doughnut of social and planetary boundaries

Economic growth, typically measured by rising GDP, remains the primary goal for most national governments. However, research shows that once basic human needs are met, further increases in GDP do not improve wellbeing [1]. Moreover, this pursuit of economic growth exacerbates environmental pressures, as GDP is tightly coupled to environmental indicators such as CO₂ emissions and material use [2].

In response, a number of ‘post-growth’ approaches have emerged, whose aim is to improve human wellbeing within environmental limits [3]. One approach, which has been gaining traction with policymakers, is the ‘Doughnut’ of social and planetary boundaries developed by economist Kate Raworth [4, 5]. The Doughnut visualises sustainability in terms of a doughnut-shaped space where resource use is high enough to meet people’s basic needs, but not so high it transgresses planetary boundaries.

An active area in post-growth research is the development of *ecological macroeconomic models* [6]. These models evaluate the performance of different policies on a wide range of environmental and social indicators, such as those in the Doughnut. Within these models, the goal is not to maximise GDP, but instead to achieve the threshold values for each of the social variables, without exceeding the boundary values for the environmental variables [7].

1.2 Applying machine learning to ecological macroeconomic models

Machine learning (ML) can potentially be used to improve ecological macroeconomic models in several ways (e.g. design, computational efficiency, calibration, policy search). Here, we focus on

two core aspects where ML can be used with a model of the Doughnut: 1) policy search, and 2) transition to a desired policy state. We show proof-of-concept examples for each aspect (see Results, sections 3.1 and 3.2 respectively), using a toy model (see Methods and Appendix, section A.1). Our goal is to validate frugal ML tools on a computational model where the ground truth is known and observable (Fig. 1). Ultimately, the aim is to scale up these ML tools to a more complex ecological macroeconomic model, such as COMPASS (the Comprehensive Model of Planetary And Social Sustainability) [8]. COMPASS allows for policies to be assessed against the Doughnut’s multiple environmental and social indicators.

2 Methods

While true ecological macroeconomic models such as COMPASS contain many variables [8], we devise a simple consumer-resource model with one environmental indicator and one social indicator (section A.1 in Appendix) to explore our idea. Here, we optimise only two model parameters: consumption c and efficiency η , which reflect two important policy levers for achieving sustainability. We define a vector $\mathbf{v} = [v_1, \dots, v_N]$ of components $v_n \in \mathbb{R}$ describing N model performance scores, defined as the N sustainability and socio-economic indicators of choice averaged over time (section A.2 in Appendix). In our case, for simplicity, we study one environmental and one socio-economic performance score, i.e. $N = 2$. We summarise aggregate performance by combining individual scores into a single ‘Doughnut’ score (Fig. 1):

$$D = \mathbf{w}^T \mathbf{v} + P(\mathbf{v}) \mathbf{w}^T \text{ReLU}(\mathbf{v}), \quad (1)$$

where $\mathbf{w} \in [0, 1]^N$ is a weight vector satisfying $\sum_{n=1}^N w_n = 1$ that scales the relative importance of individual performance scores, while $P(\mathbf{v}) = \prod_{n=1}^N H(v_n) - 1$ is a binary term based on the Heaviside step function H . We classify the model output into two cases: either all objectives are met ($\forall n, v_n > 0 \Rightarrow P(\mathbf{v}) = 0$) or not ($\exists n : v_n \leq 0 \Rightarrow P(\mathbf{v}) = 1$). In the first case, we say that the model dynamics ‘fall *within* the Doughnut’ (class ‘+’); $D > 0$ and it scales with the magnitude of all indicators. In the second case, model dynamics ‘fall *outside* the Doughnut’ (class ‘-’), and only negative indicators contribute to D , hence $D \leq 0$.

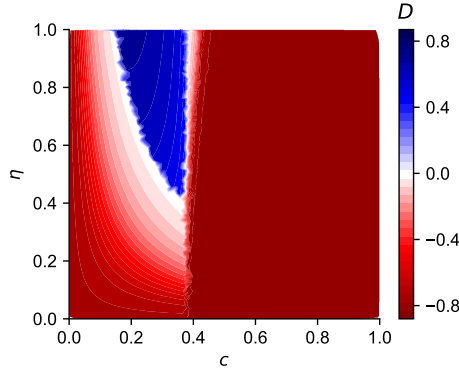


Figure 1: **‘Doughnut’ score of a computational toy model with respect to the two model parameters of interest: consumption c and efficiency η .** We consider a simple model (section A.1 in Appendix) and compute an aggregate performance measure, the Doughnut score D (eq. 1). We consider a toy model because the ground truth of its Doughnut score’s parameter-dependence is easily observable in two dimensions, unlike for complex economic models. The blue area represents the ‘Doughnut’, i.e. the desired model output, which identifies the range of the corresponding desired model input parameters.

3 Results

3.1 Locating the Doughnut: socio-economic and environmental policy search

We assume that we cannot access Fig. 1 (e.g. because of computational cost or high dimensionality). First of all, we want to find the model input parameter ranges that produce the outcomes of interest

(the Doughnut), which may cover a small portion of the parameter space. A frugal ML tool in this case is a Random Forest Classifier (RFC), with model parameter inputs as features, and a binary output indicating whether the result is within or outside the Doughnut (see Methods). The RFC correctly identifies the region of desired parameter values (Fig. 2b), despite the high class imbalance and a max depth capped at 3 nodes. Moreover, it also provides interpretable decision paths (Fig. 2a). Further details may be found in the Appendix (section A.3.1).

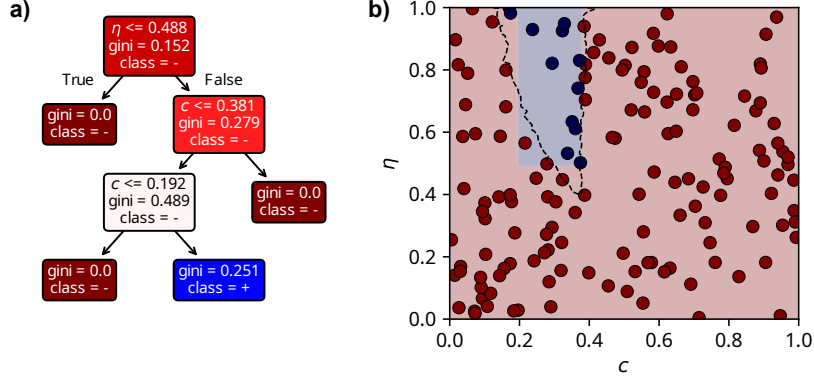


Figure 2: **Output from a Random Forest Classifier trained to find model input parameter values producing desired model outputs (i.e. within the Doughnut).** Colour coding: red indicates the ‘-’ class (i.e. model output falling outside the Doughnut), blue indicates the ‘+’ class (i.e. within the Doughnut). **a)** Decision path of a single tree showing the requirements to reach the Doughnut (blue box): high efficiency ($\eta > 0.488$), sustainable consumption ($c \leq 0.381$), and satisfying consumption needs ($c > 0.192$). **b)** Decision surface of the Random Forest Classifier represented as a background shade, where the colour indicates the predicted class. The circles are unseen test data, where the colour indicates the true label. The black dashed line circumscribes the area corresponding to the Doughnut ($D = 0$ in Fig. 1).

Since decision paths from individual trees can vary remarkably and be sensitive to noise, we propose a novel, frugal algorithm (Appendix A.4) that queries the entire forest, leveraging the RFC. The algorithm correctly identifies the desired model input parameter ranges (Fig. 3b). The advantage with respect to RFC classification alone (Fig. 2) is that parameter ranges can be ranked and presented in a tabular form (Fig. 3a), which is both scalable and interpretable (one column, one parameter). The parameter ranges are defined by individual trees’ thresholds, while the ranking is based on an ‘agreement’ score computed from individual trees’ classification, weighted by an accuracy measure on test data (Appendix A.4). This analysis can be complemented by feature importance ranking, which RFCs provide (Fig. 6 in Appendix).

3.2 Reaching the Doughnut: policy transition path

In addition to locating the Doughnut, we can also study the optimal way to achieve the model output of interest from a given starting point. To do so, we train a reinforcement learning (RL) agent (Q -learning [9]) to maximise reward (eq. 5-6 in Appendix A.3.2) based on the Doughnut score (eq. 1). The agent successfully reaches the desired model outputs, while avoiding the unwanted ones (Fig. 4a-b). Since the transition path here is long, with no gradient in local reward at the start, the discount factor γ plays a key role (Fig. 4a-b).

4 Discussion

We briefly discuss two important points: 1) measuring performance, and 2) ML techniques. Concerning measuring performance, we use a scalar, aggregate variable (D , eq. 1). This aggregate variable is not fully consistent with the Doughnut, which is a ‘strong sustainability’ framework [7]. Strong sustainability implies that higher performance on one indicator should not compensate for lower performance on another indicator [10]. While our approach ensures that all objectives are met, it is not strictly consistent with strong sustainability, and thus we continue to explore additional approaches.

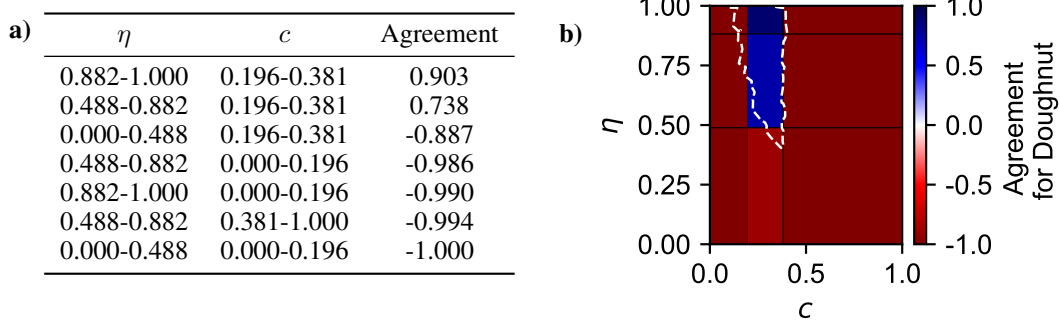


Figure 3: **Agreement scores on model output from model input parameter ranges.** We define ‘agreement’ (see Appendix A.4) as a score ranging from -1 (strong agreement for model output falling outside the Doughnut) to $+1$ (strong agreement for model output falling inside the Doughnut). Parameter ranges come from the RFC’s trees (Appendix A.4). **a)** Parameter ranges ranked by ‘agreement’ in tabular form, yielding an interpretable and scalable solution (each model input parameter adds a column). **b)** Visual representation of the tabular form, to verify the agreement scores against the ground truth. The white dashed line indicates the Doughnut ($D = 0$ in Fig. 1).

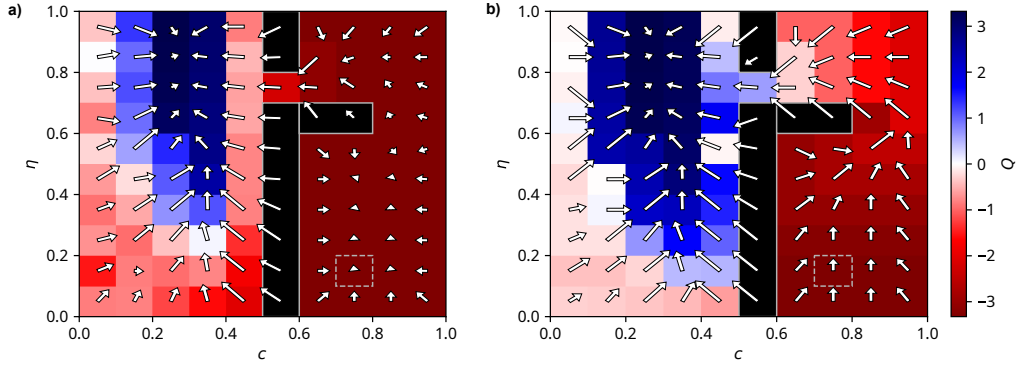


Figure 4: **RL policy of the Q -learning agent after training.** Each box is a state s defined by discretising the space of model input parameters c and η . The colour bar indicates the Q -value of action a_{stay} after training. Arrows indicate the average action in each state. The agent successfully learns to avoid hypothetical barriers (i.e. strongly undesirable model outputs, shown in black), which were artificially set to further challenge the agent. The dashed grey line indicates the starting point of the agent. Panels **a** and **b** differ by discount factor ($\gamma = 0.5$ and $\gamma = 0.8$, respectively, see eq. 5 in Appendix). The discount factor prioritises long-term over short-term rewards.

Regarding ML techniques, each comes with its advantages and drawbacks [11, 12]. For instance, our policy search (section 3.1) can be optimised (e.g. 13 and section A.3.1 in Appendix). In our policy-transition path search (section 3.2), RL can be confronted with the curse of dimensionality in designing states and action spaces [14], which affects the agent’s exploration abilities. However, RL can also offer further advantages, such as hierarchical RL [15] or co-operative agents (e.g. each agent optimises a different dimension of the Doughnut).

5 Conclusions

Achieving sustainability is a complex and multi-dimensional challenge. Here, we have applied two standard ML techniques to highlight the potential of ML to improve ecological macroeconomic modelling. Specifically, we have presented two simple examples: 1) finding socio-economic policies (i.e. model input parameters) that are consistent with the Doughnut of social and planetary boundaries), and 2) identifying the optimal policy-transition path (i.e. traversing the model input parameter space). The next step is the application of these ML tools to COMPASS [8], a more complex ecological macroeconomic model that allows real policies to be assessed against the Doughnut’s indicators.

Acknowledgments

We would like to thank T. Domingos for his insightful comments on this paper. This research was funded by the European Union in the framework of the Horizon Europe Research and Innovation Programme under grant agreement number 101137914 (MAPS: ‘Models, Assessment, and Policies for Sustainability’). It received additional support from the AI4theSciences COFUND programme (Horizon 2020 grant agreement number 945304).

References

- [1] David Collste, Sarah E. Cornell, Jorgen Randers, Johan Rockström, and Per Espen Stoknes. Human well-being in the Anthropocene: limits to growth. *Global Sustainability*, 4:e30, 2021.
- [2] Helmut Haberl, Dominik Wiedenhofer, Doris Virág, Gerald Kalt, Barbara Plank, Paul Brockway, Tomer Fishman, Daniel Hausknost, Fridolin Krausmann, Bartholomäus Leon-Gruchalski, et al. A systematic review of the evidence on decoupling of GDP, resource use and GHG emissions, part II: synthesizing the insights. *Environmental Research Letters*, 15(6):065003, 2020.
- [3] Giorgos Kallis, Jason Hickel, Daniel W. O’Neill, Tim Jackson, Peter A. Victor, Kate Raworth, Juliet B. Schor, Julia K. Steinberger, and Diana Ürge Vorsatz. Post-growth: the science of wellbeing within planetary boundaries. *The Lancet Planetary Health*, 9(1):e62–e78, 2025.
- [4] Kate Raworth. *Doughnut Economics: Seven Ways to Think Like a 21st-Century Economist*. Random House, London, 2017.
- [5] Andrew L Fanning and Kate Raworth. Doughnut of social and planetary boundaries monitors a world out of balance. *Nature*, 646(8083):47–56, 2025.
- [6] Lukas Hardt and Daniel W O’Neill. Ecological macroeconomic models: assessing current developments. *Ecological Economics*, 134:198–211, 2017.
- [7] Daniel W O’Neill, Andrew L Fanning, William F Lamb, and Julia K Steinberger. A good life for all within planetary boundaries. *Nature Sustainability*, 1(2):88–95, 2018.
- [8] Jefim Vogel, Rob Van Eynde, Lorenzo Di Domenico, Seán Fearon, Luzie Dallinger, Tina Beigi, Kevin Dillman, and Daniel W O’Neill. COMPASS: A social-ecological macroeconomic model for guiding pathways to equitable human wellbeing within planetary boundaries. Technical report, ToBe Research Project, 2025.
- [9] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.
- [10] Eric Neumayer. *Weak versus Strong Sustainability: Exploring the Limits of Two Opposing Paradigms*. Edward Elgar, Cheltenham, UK, 4th edition, 2013.
- [11] Christoph Molnar. *A Guide for Making Black Box Models Explainable*. 2018.
- [12] 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.
- [13] Francesco Lamperti, Andrea Roventini, and Amir Sani. Agent-based model calibration using machine learning surrogates. *Journal of Economic Dynamics and Control*, 90:366–389, 2018.
- [14] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. Bradford Books, 2018.
- [15] Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(4):341–379, 2003.

A Appendix

A.1 Computational model

We devise a simple dynamical system (eqs. 2 and 3) based on two time-dependent variables, i.e. an environmental budget indicator x_{env} and a social indicator x_{soc} :

$$\frac{dx_{\text{env}}}{dt} = r x_{\text{env}} (1 - x_{\text{env}}) H(x_{\text{env}} - x_{\text{env-crit}}) - \tilde{c}, \quad (2)$$

$$\frac{dx_{\text{soc}}}{dt} = x_{\text{soc}} (1 - x_{\text{soc}}) \eta \tilde{c} - \min(x_{\text{soc}}, c - \tilde{c}), \quad (3)$$

with $x_{\text{env}}, x_{\text{soc}} \in [0, 1]$. We also define critical thresholds ($x_{\text{env crit}}$ and $x_{\text{soc crit}}$) for both indicators. The environmental threshold represents the critical environmental budget to preserve natural resources; the socio-economic threshold represents minimum socio-economic goals. The environmental threshold affects the model dynamics through the Heaviside step function H , reflecting the tipping point for the collapse of environmental resources. Model dynamics qualitatively depend on three parameters that are fixed for an individual simulation: consumption c (which relates to the policy goal of sufficiency), efficiency η (which relates to the policy goal of resource efficiency), and an environmental budget regeneration rate r (e.g. the annual capacity of carbon sinks in climate change terms), here set to $r = 1.5$. Moreover, we define $\tilde{c} = \min(c, x_{\text{env}})$ to be actual consumption, as the environmental budget may be lower than consumption. In our analysis, we only focus on efficiency η and consumption c , as they are the only two parameters in our simple model that are of interest from a socio-economic and environmental policy perspective.

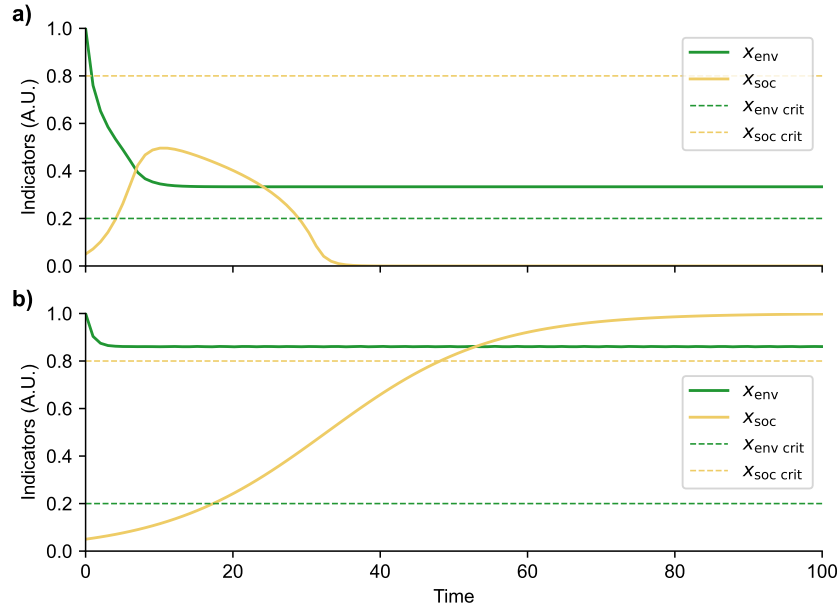


Figure 5: Examples of model behaviour from our chosen toy model. The two plots show the time dynamics of the socio-economic (x_{soc}) and environmental (x_{env}) indicators, with their respective critical thresholds ($x_{\text{soc crit}}$ and $x_{\text{env crit}}$), for two different model input parameter configurations (parameters c and η). **a)** While natural resources are not critically compromised, socio-economic targets are not met ($c = 0.42$, $\eta = 0.9$). **b)** Both indicators achieve their respective thresholds, i.e. socio-economic targets are met without compromising natural resources ($c = 0.2$, $\eta = 0.9$).

A.2 Scores from indicators

We define two performance scores ($N = 2$) from our computational model as the average of the environmental and social indicators respectively:

- Environmental performance $v_1 = \frac{1}{T} \int_0^T x_{\text{env}} - x_{\text{env-crit}}(t) dt$,
- Socio-economic performance $v_2 = \frac{1}{T} \int_0^T x_{\text{soc}} - x_{\text{soc-crit}} dt$,

where

- $x_{\text{env-crit}}$ is a tipping point below which the environmental resource ceases to regenerate.
- $x_{\text{soc-crit}}$ is the minimum social threshold required to meet basic needs (the equivalent of the ‘social foundation’ in the Doughnut).

A.3 Machine Learning

A.3.1 Random Forest Classifier

We sample synthetic data using from `emukit.core.initial_designs import RandomDesign`, with 500 samples, setting `np.random.seed(42)`. We split test and training data using from `sklearn.model_selection import train_test_split` with `random_state = 42`.

For the RFC, we applied the following hyperparameters: max depth = 3; 100 trees; stratified data for training. We set max depth to 3 to pursue frugality and interpretability. Future work may wish to compare the performance of this classification with performance from a hyperparameter-optimised RFC.

Stratified 5-fold cross-validation on the RFC achieves 0.968 ± 0.013 accuracy. When sampling the model a high number of times (e.g. 5000), the fraction of samples falling outside the Doughnut is 0.912, denoting high class imbalance.

We complement our RFC analysis by adding feature importance ranking (Fig. 6). Feature importance values can be obtained for free from RFCs. We find that the consumption parameter c impacts the classification of the model dynamics as ‘within’ or ‘outside’ the Doughnut more than the efficiency parameter η .

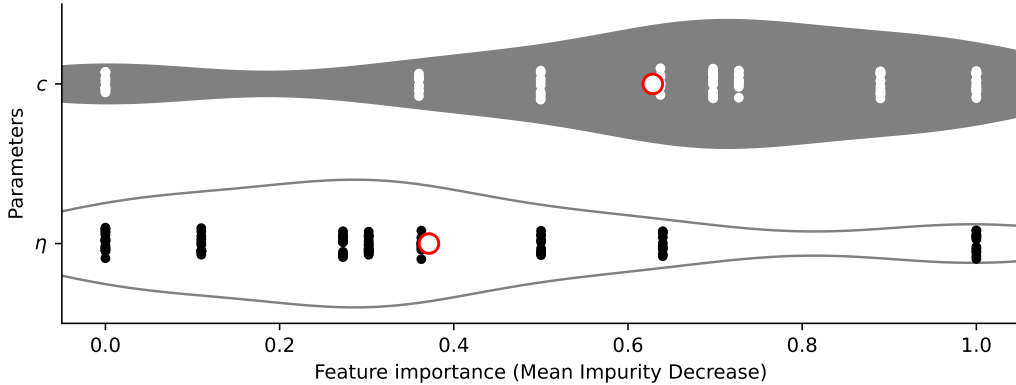


Figure 6: **Feature importance computed from the RFC.**

A.3.2 Q-learning

For the RL agent, we applied the following parameters: learning rate $\alpha = 0.1$, discount factor $\gamma = 0.5$, inverse temperature $\beta = 2$ for softmax decision rule, 30,000 training episodes, 50 steps.

RL states s are defined by discretising the parameter space in 10 values for each model input parameter (c and η).

At each time step t within an episode, the agent selects an action a_t according to a softmax decision rule:

$$P(a_t | c_t, \eta_t) = \frac{e^{\beta Q(c_t, \eta_t, a_t)}}{\sum_{a' \in \mathcal{A}} e^{\beta Q(c_t, \eta_t, a')}} \quad (4)$$

We define reward as:

$$r_t = R(c_{t+1}, \eta_{t+1}) + \gamma \max_{a' \in \mathcal{A}(c_{t+1}, \eta_{t+1})} Q(c_{t+1}, \eta_{t+1}, a'), \quad (5)$$

where

$$R(c_{t+1}, \eta_{t+1}) = D(c_{t+1}, \eta_{t+1}), \quad (6)$$

with D defined in eq. 1 and computed with the parameter values where the state s is centred.

Finally, the Q -value update rule is:

$$Q(c_t, \eta_t, a_t) \leftarrow Q(c_t, \eta_t, a_t) + \alpha (r_t - Q(c_t, \eta_t, a_t)). \quad (7)$$

A.4 Agreement score

We train a Random Forest Classifier on a binary classification task, to classify whether a given configuration of model parameter values produces a model regime falling inside or outside the Doughnut (see Methods).

We develop an *agreement* score leveraging the trained RFC to: 1) identify the parameter ranges that produce the desired model regimes, 2) rank them, and 3) present them in readily interpretable tabular form. The algorithm works by first identifying suitable parameter ranges, to then compute agreement for each range.

A.4.1 Identify split thresholds

In a trained RFC, each decision tree defines its own split thresholds for features. We collect all thresholds across trees and count their frequency.

A.4.2 Select split thresholds

For each feature, we sort the split thresholds by frequency. Within each feature i , starting from the most frequent split threshold, we merge all thresholds lying within a chosen ε_i ($\varepsilon = 0.02$ for all features, in our case), keeping the split threshold with the highest count. Merged split thresholds also accumulate their counts. We then set a percentage (25% in our case) as the minimal frequency threshold to retain split thresholds. All split threshold with a lower frequency are discarded. The remaining split thresholds define the selected features' ranges (in our case, model parameter ranges), which form multi-feature bins. Here, the selection step, which might seem optional, not only helps to deal with a more reasonable number of feature splits, but also plays a central role in creating large-enough value ranges to bin test data and compute test accuracy in later steps of the algorithm, especially for high-dimensional input spaces (Fig. 7). This selection step can be further optimised to ensure that all bins contain a minimum number of test data points.

A.4.3 Compute agreement within feature ranges

We sample a high number (e.g. 100,000) of model input parameter configurations as input feature values. We then bin them into the bins obtained in the previous step. For each bin b and for each tree τ we collect the predicted class of each sample within that bin. Next, for each tree, we average the predicted classes $\hat{c}_n \in \{0, 1\}$ (with 1 indicating that the input features produce a Doughnut-compliant model output, 0 otherwise) across N_b samples within that bin. We define this measure as 'estimated

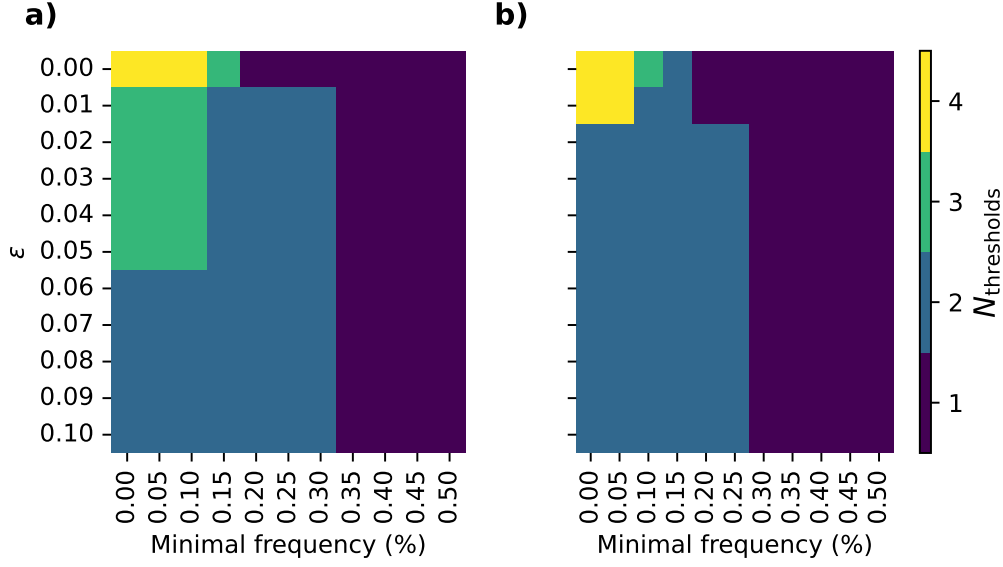


Figure 7: **Heatmap showing how the number of selected split-thresholds, collected among trees of a Random Forest Classifier trained on our task, changes according to the parameter ϵ and the minimal frequency.** Each plot refers to a different input model parameter: c in **a** and η in **b**. The higher the number of selected thresholds per parameter, the more granular, but also less interpretable.

Doughnut frequency' $\hat{f}_{\text{raw},\tau,b}$ for that bin:

$$\hat{f}_{\text{raw},\tau,b} = \frac{1}{N_b} \sum_{n=1}^{N_b} \hat{c}_n. \quad (8)$$

We then compute the raw accuracy $a_{\text{raw},\tau,b}$ for each tree, for all labelled test data within a given bin.

Since we are dealing with a classification task with only two classes, we can leverage both high and low accuracy. The further below 50% the test accuracy is, the stronger our confidence that the actual class is the one *not* predicted by the tree. We can therefore compute a 'useful' accuracy

$$a_{\text{useful},\tau,b} = 2 |a_{\text{raw},\tau,b} - 0.5|, \quad (9)$$

and a 'useful' estimated Doughnut frequency

$$\hat{f}_{\text{useful},\tau,b} = \begin{cases} \hat{f}_{\text{raw},\tau,b}, & a_{\text{raw},\tau,b} > 0.5 \\ 1 - \hat{f}_{\text{raw},\tau,b}, & a_{\text{raw},\tau,b} < 0.5 \end{cases} \quad (10)$$

for each tree τ and bin b .

Next, for each bin, we want to emphasise the frequency values of the Doughnut $\hat{f}_{\text{raw},\tau,b}$ from the trees with the highest useful accuracy. We therefore normalise useful accuracies across trees, bin-wise, using softmax:

$$a_{\text{norm},\tau,b} = \frac{e^{\beta_{\text{norm}} a_{\text{useful},\tau,b}}}{\sum_{\tau'} e^{\beta_{\text{norm}} a_{\text{useful},\tau',b}}}, \quad (11)$$

setting the inverse temperature parameter $\beta_{\text{norm}} = 1$.

We then measure agreement A for each bin by averaging the estimated Doughnut frequency across trees, weighted by their own normalised useful accuracy:

$$A_b = \sum_{\tau} a_{\text{norm},\tau,b} \cdot \hat{f}_{\text{useful},\tau,b}. \quad (12)$$

Finally, agreement is scaled to lie between $[-1, 1]$, parameter ranges are sorted, and presented in tabular form (Fig. 3a).

A.5 Hardware and computation

We used a Dell Precision 7760 laptop with a 2.60 GHz 16-core Intel Xeon(R) W-11955M processor. Generating the 500 samples (i.e. running the model 500 times) and training the classifier takes < 1.4 seconds. Running the agreement score algorithm (section A.4) adds less than 2.5 seconds. Training the Q -learning agent required less than 1 min 30 seconds.

The full research project involved trial-and-error to adapt and optimise the ML tools to the specific computational experiments. We only report the final version of our experiments.

A.6 Code

We implemented our statistical pipeline in Python 3.12.3. The code is available at https://github.com/stefanovrizzi/doughnut_ml.