Supplementary Information for:

Body mass prediction in biped dinosaurs using supervised learning

Alejandro Otero[1] and Ailen Borya[2]

[1] CONICET - División Paleontología de Vertebrados (Anexo Laboratorios), Facultad de Ciencias Naturales y Museo (UNLP), La Plata (1900), Argentina. alexandros.otero@gmail.com;

[2] Especialización en Geociencias de Exploración y Desarrollo de Hidrocarburos, Facultad de Ciencias Naturales y Museo (UNLP), La Plata (1900), Argentina. ailenborya@gmail.com;

Content

1. Model setup (page 2).
2. Dataset balance via SMOGN (page 5).
3. Hyperparameters for random forests (page 6).
4. Figure S1 (page 7).
5. Figure S2 (page 8).
6. Figure S3 (page 9).
7. Figure S4 (page 10).
8. Figure S5 (page 11).
9. Performance on the test set (page 11).
10. Figure S6 (page 14).
11. Figure S7 (page 15).
12. Figure S8 (page 16).
13. Figure S9 (page 17).
14. Figure S10 (page 18).
15. Figure S11 (page 19).
16. Table S5 (page 20).
17. Table S9 (page 22).
18. Table S10 (page 23).
19. Figure S12 (page 25).
20. Scripts (page 27).
21. References not included in main text (page 28).

1. Model setup

Hyperparameter optimization. Machine learning algorithms automatically learn internal parameters from data, but some parameters (the hyperparameters) can be defined prior to training. These determine model structure and learning behavior, strongly influencing performance. Hyperparameter optimization (tuning) automates the search for optimal combinations, reducing manual intervention and improving reproducibility (Yang & Shami, 2020; Elgeldawi et al., 2021; Ballell et al., 2022; Foster et al., 2022; Linchamps et al., 2023; Liu et al., 2023).

Several optimization strategies exist, including Grid Search, Random Search, and Bayesian Optimization (Bergstra et al., 2011; Syarif et al., 2016; Wicaksono & Supianto, 2018; Liashchynskyi & Liashchynskyi, 2019; Madhav & HimanshuRawat, 2021). Here, Grid Search was used, which evaluates all possible hyperparameter combinations specified by the user. This exhaustive method is computationally demanding and best suited for small datasets (Elgeldawi et al., 2021; Ballell et al., 2022; Linchamps et al., 2023).

Model evaluation was performed using k-fold cross-validation (CV) to minimize underfitting and overfitting. In this approach, the training data are divided into k subsets; the model is iteratively trained on k-1 folds and validated on the remaining one, and results are averaged across folds. In Scikit-learn, ‘GridSearchCV’ implements this process, taking as input the model (e.g., random forests), candidate hyperparameter values, and the number of folds (cv = 5 in this study).

It is important to distinguish this internal k-fold cross-validation from the overarching data partitions detailed method section (i.e., the globally withheld 10% validation set and the 75:25 train-test split). While the 75:25 split provides a single, fixed division for baseline algorithm training and evaluation, relying solely on a single static split during hyperparameter tuning can introduce optimization bias, where the selected hyperparameters inadvertently overfit to that specific data configuration. By employing k-fold cross-validation internally on the training subset, the data is dynamically re-partitioned. This iterative approach maximizes the utility of the relatively small paleontological dataset, ensuring that the optimal hyperparameter combination is genuinely robust and generalizable across different permutations of the data, rather than being an artifact of a single random split.

After training the random forests model with the full training set (Table S8) under default and optimized hyperparameters for both femur length and circumference predictors, a sensitivity analysis was conducted to evaluate the influence of individual hyperparameters on model performance. This analysis allowed us to assess the robustness of the model against variations in parameter values (i.e., ‘max. depth’, ‘max. features’, ‘max. samples’, ‘min. samples leaf’, ‘min. samples split’ and ‘n. estimators’) and to identify those exerting the strongest control on mean

squared error (MSE). Based on the results of this sensitivity analysis, the random forests models were subsequently re-trained using refined hyperparameter configurations to ensure optimal and stable performance.

The output of the sensitivity analysis of hyperparameters are presented as plots showing the performance (in terms of MSE) versus different values for each hyperparameter for both training and testing sets in order to identify the value for which the error is minimal.

Hyperparameter optimization was not exhaustively performed for the sequential neural networks models in this study due to both methodological and data-related considerations. The dataset used for model training comprises approximately 1000 observations with a limited number of predictors, a regime in which neural networks are particularly susceptible to overfitting when excessive architectural flexibility (=complexity) is introduced (e.g., Aggarwal, 2018; Chollet, 2018). Under these conditions, extensive hyperparameter tuning can lead to artificially inflated performance estimates that do not generalize beyond the training data. Instead, network architectures and training parameters were selected based on established best practices for regression problems with small-to-moderate sample sizes, including a limited number of hidden layers, conservative neuron counts, and standard optimization settings. This approach emphasizes model stability and interpretability rather than maximal performance (Aggarwal, 2018). Moreover, random forests models are inherently less sensitive to hyperparameter choices and exhibit strong performance with minimal tuning. Given this asymmetry, exhaustive hyperparameter optimization of the neural networks could bias the comparison in favor of the more flexible model, thereby reducing the fairness of cross-algorithm evaluation. By constraining the sequential neural networks to a parsimonious architecture and avoiding aggressive hyperparameter tuning, we prioritize robust generalization and ensure that observed differences in performance between neural networks and random forests reflect intrinsic methodological differences rather than unequal levels of optimization. This strategy aligns with the study's goal of comparative model assessment rather than absolute performance maximization.

Regularization. It refers to a set of techniques designed to control model complexity and improve generalization by penalizing overly flexible solutions. Unlike hyperparameter optimization, which aims to identify an optimal configuration of model settings, regularization operates during model training by constraining how much information a model can encode. This is particularly important in regression problems where complex models, such as neural networks, are prone to overfitting (Srivastava et al., 2014; Goodfellow et al., 2016; Aggarwal, 2018; Chollet, 2018; Ying, 2019; Ketkar & Moolayil, 2021).

Overfitting occurs when a model captures noise or idiosyncratic patterns in the training dataset rather than the underlying signal, resulting in excellent training performance but poor predictive

accuracy on unseen data (Dietterich, 1995; Ying, 2019). In practice, there is always a performance gap between training and test datasets, but this gap becomes especially pronounced when model complexity is high relative to dataset size, which is a common scenario in paleontological and biological datasets (Cifuentes-Alcobendas & Domínguez-Rodrigo, 2019; Ballell et al., 2022; Marques et al., 2025a). Although increasing the size of the training dataset is the most effective way to mitigate overfitting, this is often not feasible. In such cases, regularization provides a principled alternative by limiting the effective capacity of the model. By restricting the magnitude of model parameters or introducing stochastic constraints during training, regularization forces the optimization process to prioritize the most robust and generalizable patterns in the data. Models that are constrained in this way are less likely to memorize spurious relationships and more likely to capture biologically meaningful signals relevant to body mass prediction. In the present study, the dataset comprises a relatively small number of observations and a limited set of predictor variables. Accordingly, two widely adopted and well-established regularization strategies for neural networks were implemented: L2 weight regularization and dropout.

L2 regularization (also known as weight decay) constrains model complexity by penalizing large weight values. This is achieved by adding a term to the loss function proportional to the sum of the squared weights, thereby discouraging extreme parameter values and promoting smoother, more stable model solutions (Goodfellow et al., 2016; Aggarwal, 2018; Chollet, 2018; Ying, 2019; Ketkar & Moolayil, 2021; see Hsiang et al., 2019; Ha & Kim, 2023 for applications in paleontology). As a result, L2 regularization reduces variance while preserving the ability of the network to learn nonlinear relationships between skeletal measurements and body mass.

Dropout provides an alternative, complementary form of regularization by introducing stochasticity during training. When applied to a neural networks layer, dropout randomly sets a fraction of the layer's output units to zero at each training step, which prevents the network from becoming overly reliant on specific neurons and encourages the development of redundant, distributed representations. During inference, dropout is disabled and activations are appropriately rescaled, ensuring consistent output magnitudes (Srivastrava et al., 2014; Goodfellow et al., 2016; Aggarwal, 2018; Ying, 2019). In effect, dropout approximates the averaging of predictions from an ensemble of subnetworks, which enhances robustness and reduces overfitting, and has been applied in different paleontological fields, such as taphonomy (Cifuentes-Alcobendas & Domínguez-Rodrigo, 2019) and microfossil classification (Ozer et al., 2023b).

Together, L2 regularization and dropout provide an effective and computationally efficient strategy for improving the generalization performance of neural networks models in settings characterized by limited sample sizes, while maintaining interpretability and comparability with alternative approaches such as random forests regression.

2. Dataset balance via SMOGN.

SMOTE (Synthetic Minority Over-sampling TEchnique) was developed to address class imbalance in classification tasks. It operates by identifying minority class samples and generating synthetic observations along the line segments connecting each minority instance to its nearest neighbours in feature space, effectively expanding the representation of underrepresented classes in a discrete label context (Chawla et al., 2002). Because SMOTE was designed for categorical targets, it defines ‘minority’ and ‘majority’ exclusively in terms of class membership.

SMOGN (Synthetic Minority Over-sampling Technique for Regression with Gaussian Noise) was developed by Branco et al. (2017) as a direct extension of the SMOTE philosophy to regression problems, where the target variable is continuous rather than categorical. In regression, there is no discrete minority class; instead, rarity is defined by the distribution of the continuous target variable. SMOGN identifies rare values using a relevance function that quantifies the importance of target values based on their frequency in the training distribution, typically assigning higher relevance to values in the tails of the distribution (Branco et al., 2017; Ribeiro, 2011).

SMOGN interpolates both predictor and target values simultaneously, generating synthetic cases with continuous body mass values derived from the weighted combination of neighbouring training instances (see Branco et al., 2017 for algorithm and experimental evaluation). In terms of resampling strategy, SMOGN combines oversampling of rare target-value regions with undersampling of frequent target-value regions, producing a more balanced distribution across the full continuous response range. When the local neighbourhood of a rare instance is too small to generate reliable synthetic samples via interpolation alone, SMOGN introduces Gaussian noise perturbations as an alternative synthetic generation strategy. This prevents degenerate interpolation in sparse regions of the predictor space, a common situation with paleontological datasets where large-bodied extinct taxa have few morphological neighbours in the training distribution (Branco et al., 2017).

In the present dataset, large-bodied taxa are underrepresented relative to small-bodied extant birds. Because body mass is a continuous variable, SMOTE cannot be applied directly. SMOGN addresses this by identifying large body mass values as rare (high-relevance regions of the target distribution) and generating synthetic training cases in those regions while simultaneously reducing overrepresentation of the modal (small-bodied) mass range. This rebalancing strategy is particularly appropriate for allometric regression problems spanning several orders of magnitude, where extreme values carry disproportionate biological significance but are statistically underrepresented (Ribeiro, 2011; Torgo et al., 2013; Branco et al., 2017;).

3. Hyperparameters for random forests.

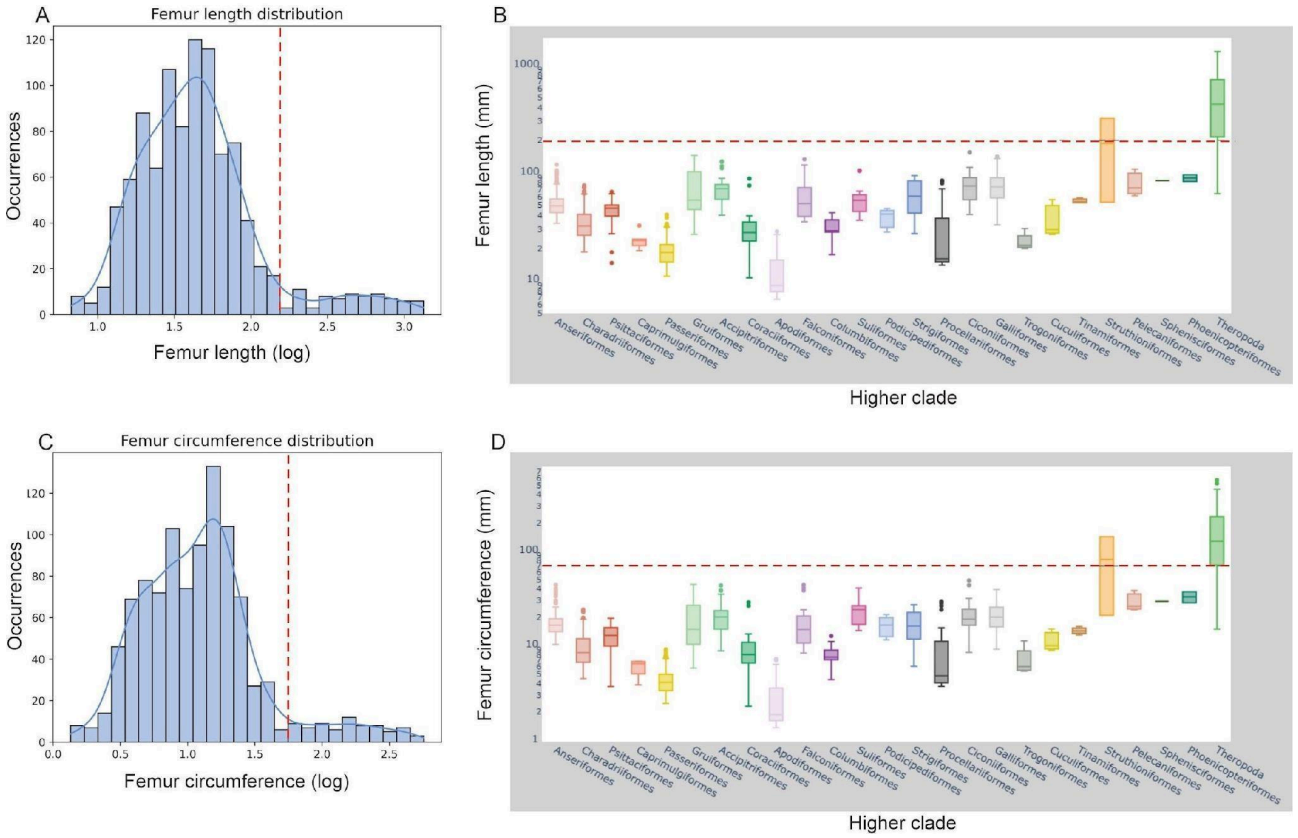
In simple terms, random forests work by aggregating and averaging a set of multiple decision trees. Each tree in the ensemble is built from a sample drawn with replacement (that is, bootstrap sample) from the training set. For each tree in the forest, the **'max_features'** creates a random subset of the variables to be considered when splitting a tree node. Both bootstrapping and feature sampling decrease the variance of the estimation.

Another important hyperparameter is **'n_estimators'**, which controls the number of trees in the forest. More trees yields better results, although it also increases the computational requirements. **'max_depth'** is the maximum depth of each tree. If this parameter is set to 'None', the tree branches will split until the output is pure, increasing the risk of overfit. **'min_samples_split'** controls the number of samples required to split an internal node, whereas **'min_samples_leaf'** is the minimum number of samples required to be at a leaf node. '**Bootstrap'** can be either True or False. If False, the whole dataset is used to build each tree. If bootstrap is True, **'max_samples'** controls the number of samples to draw from X to train each base estimator. **'oob_score'** means out-of-bag samples to estimate the generalization score.

Hyperparameter's values con be found here:
https://scikit-learn.org/stable/modules/ensemble.html#forest

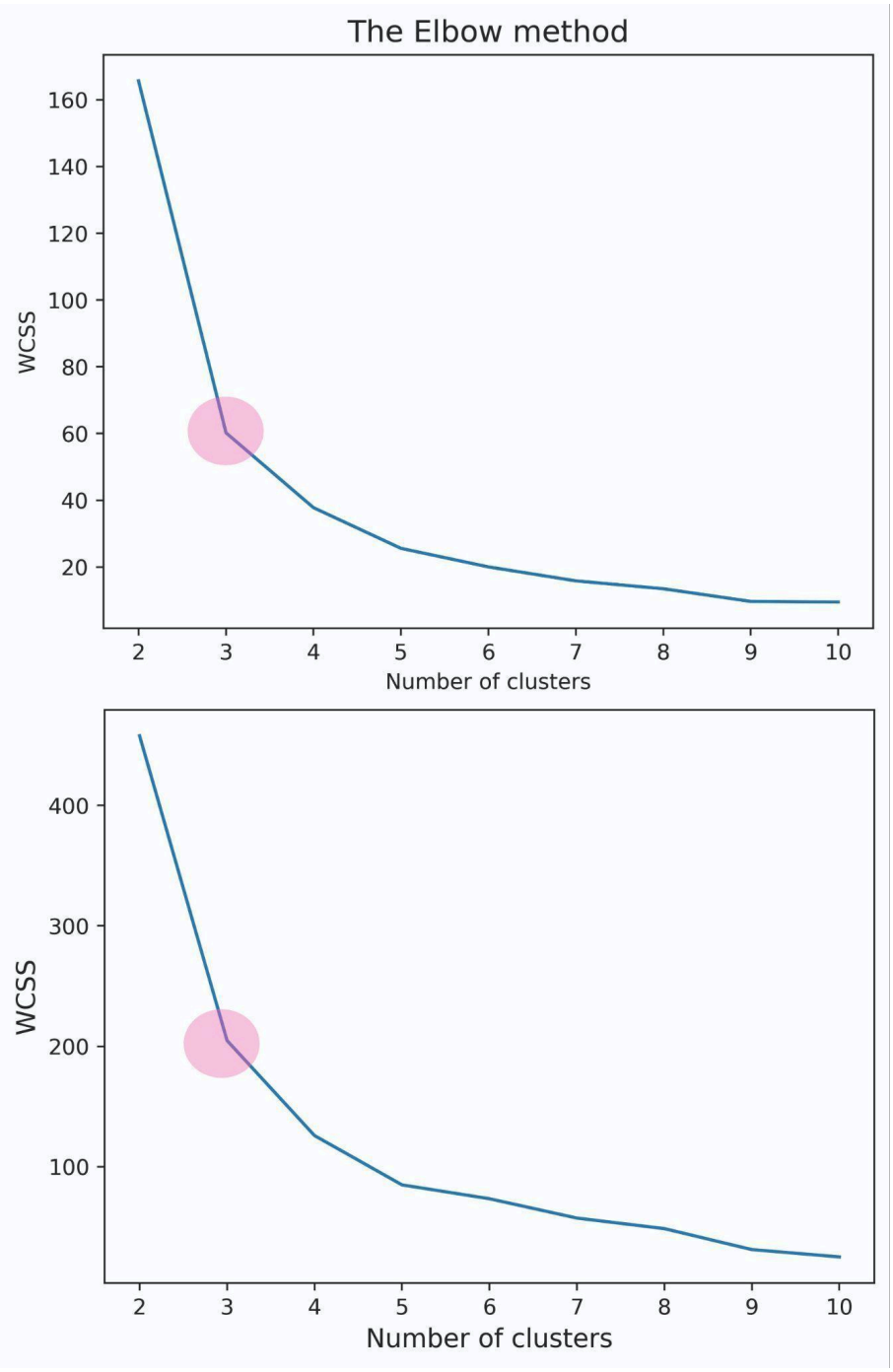
4. Figure S1.

Visual representation of the structure of the full training dataset. The histograms in A and C show the skewed distribution of the frequency of the femur length and circumference (both in log 10), respectively, highlighting femur length and circumference at 200 mm and 70 mm, respectively (vertical dotted lines). The boxplots in B and D show the disaggregated structure, highlighting femur length and circumference at 200 mm and 70 mm, respectively (horizontal dotted lines). Most taxa are placed below these thresholds. The y-axis in B and D is presented in log 10 scale for ease of visualization. The clade name 'Theropoda' refers to non-avian theropods.



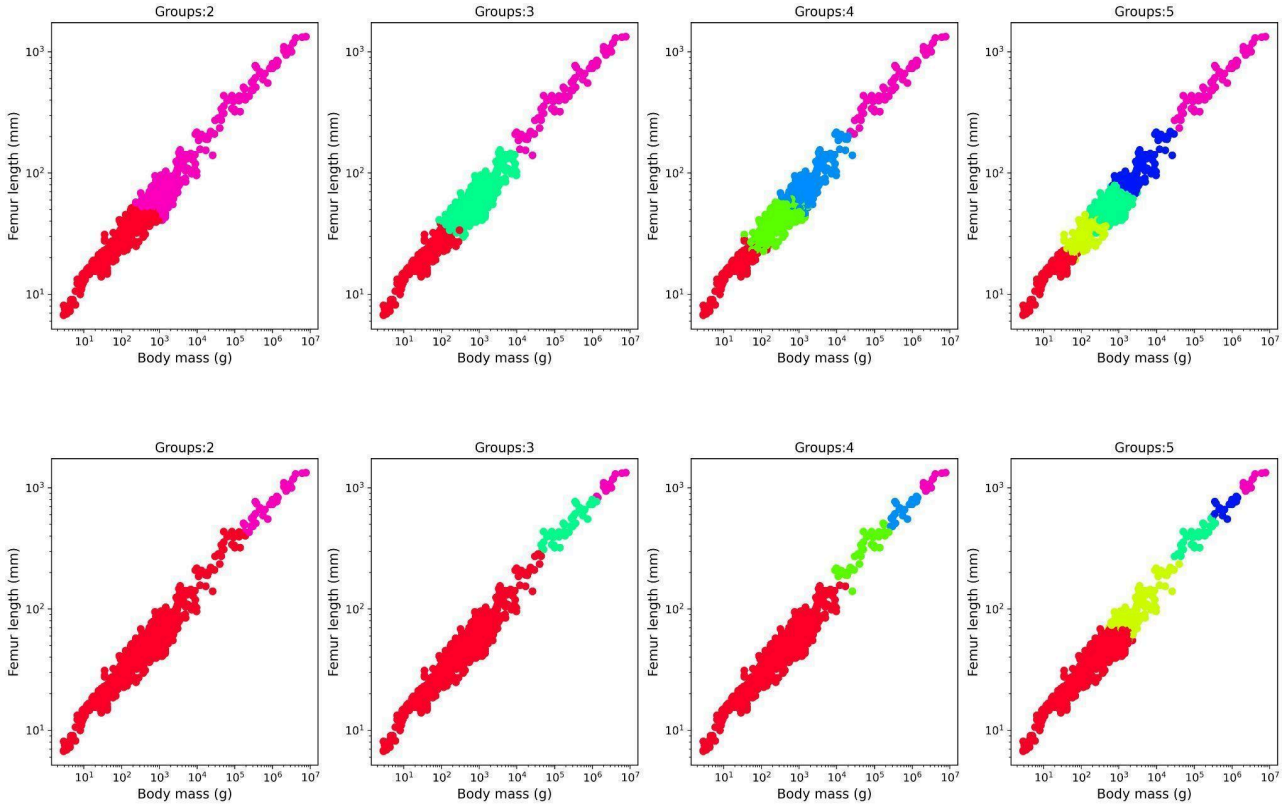
5. Figure S2.

Elbow method for cluster identification via within-cluster sum of squares (WCSS). The highest inflection is given at 3 clusters for both log10-transformed (top) and standardized (bottom) data.



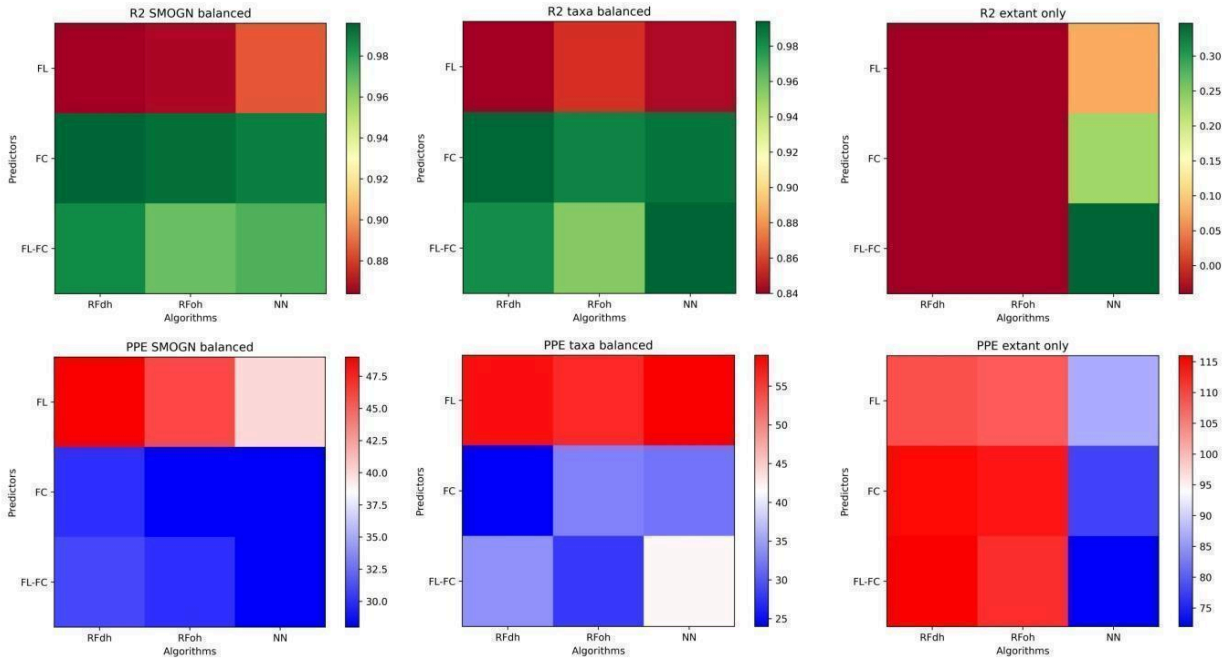
6. Figure S3.

Scatter plot showing the distribution of femur length measurements for the full training dataset against body mass after clusterization in a log10 (top) and standardization (bottom) spaces for the clusters with the highest values after silhouette method.



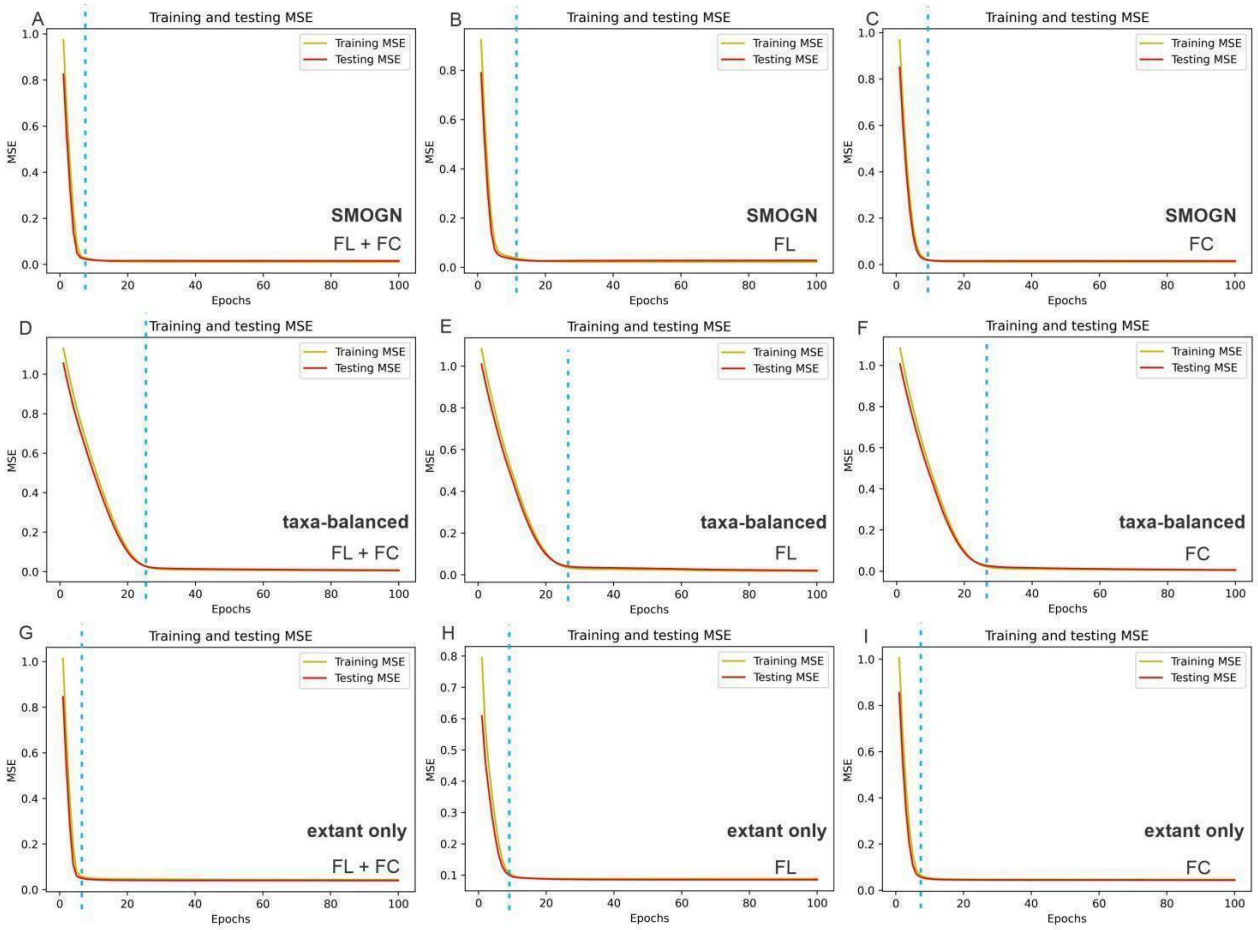
7. Figure S4.

Heatmaps showing overall performance of algorithms (random forests and neural networks) versus predictors (FL-FC combined, FL alone, FC alone) in terms of R^2 (upper row) and PPE (bottom row) for the validation set after training with various balanced datasets. Performance is similar for SMOGN and taxa-balanced datasets (first and second columns), dropping for the extant-only dataset (third column). Abbreviations: NN, neural networks; RFdf, random forests with default hyperparameters; RFoh, random forests with optimized hyperparameters.



8. Figure S5.

Neural networks' learning curves (epoch versus MSE) for predictor combinations under SMOGN, taxa-balanced and extant-only training datasets. The vertical dashed line depicts the epoch at which algorithm reaches performance stability under an error plateau. The taxa-balanced iterations reach such stability after epoch 20. Abbreviations: FC, femur circumference; FL, femur length; MSE, mean squared error.



9. Performance on the test set.

Overall, performance on the test set confirms the patterns observed in validation analyses while clarifying the internal generalization behaviour of both algorithms under controlled distributional conditions. Below are exposed the performances (MSE and R^2) of the training datasets across algorithms (Tab. S10).

Across the full, SMOGN-balanced, and taxa-balanced datasets, both random forests (RF) and neural networks (NN) models achieved high explanatory power when trained with combined predictors (FL + FC) or femoral circumference (FC) alone. R^2 values frequently exceeded 0,98 and reached as high as 0,996–0,997 (RF and NN, FC and combined predictors), confirming that

nonlinear ensemble and parametric learners effectively capture the dominant allometric signal when variance breadth is preserved.

As in the validation set, femoral circumference consistently outperformed femoral length alone. FL-only models showed lower R^2 (e.g., NN FL ≈ 0,876 in the original dataset), reinforcing the weaker scaling signal embedded in length relative to circumference. This pattern aligns with biomechanical expectations, as circumference more directly reflects weight-bearing cross-sectional properties and thus body mass scaling (e.g., Anderson et al., 1985; Campione & Evans, 2014).

SMOGN balancing produced modest fluctuations but did not compromise performance in RF models, which maintained $R^2 \geq 0,99$ in most configurations. This reflects the variance-reduction properties of bagging and feature subsampling (Breiman, 2001b; Biau & Scornet, 2016). Neural networks showed slightly greater sensitivity to SMOGN in the test set (e.g., NN FL + FC $R^2 \approx$ 0,967), consistent with their reliance on gradient-based global optimization, where synthetic sampling can alter loss landscape geometry (Goodfellow et al., 2016). Nevertheless, explanatory power remained high.

Taxa balancing similarly preserved strong predictive performance across algorithms. Notably, NN models under taxa balancing achieved some of the highest R^2 values (up to 0,997 for FL + FC), suggesting that redistributing clade representation does not impair, and may even stabilize, parametric learning when variance coverage is maintained.

A stark contrast emerges under extant-only training. For random forests, R^2 values on the test set collapse to ≈ 0,06–0,08 across predictor configurations, indicating near-total loss of explanatory power. This mirrors the instability observed previously when fixed train-test splits were imposed and confirms that the issue is structural rather than stochastic.

The explanation lies in the interaction between restricted variance and tree-based partitioning. When trained exclusively on extant taxa, RF learns a regression surface constrained to a narrow body mass regime. Even within the extant-only test partition, random splitting can produce uneven size distributions between training and test subsets. Because tree-based models predict via terminal node averages bounded by training responses, inadequate coverage of the upper range within the training split forces predictions toward local means. This dramatically reduces explained variance when the test subset contains relatively larger individuals. In small or variance-restricted datasets, such partition imbalance can yield artificially low R^2 despite unchanged algorithmic settings. This phenomenon reflects high variance in performance estimates under limited morphospace coverage (Hastie et al., 2009).

Neural networks exhibit a different pattern. Under extant-only training, NN models retain moderate explanatory power ($R^2 \approx$ 0,63–0,73 for combined predictors and FC), substantially outperforming

RF in this restricted regime. Because NNs learn a continuous parametric function rather than piecewise constant partitions, they can interpolate more smoothly across sparse regions of predictor space. However, performance remains markedly inferior to full-dataset training, reflecting increased bias under variance truncation.

Removing extreme extant values (notably *Struthio camelus*) dramatically improves performance in both algorithms. For RF, R^2 increases from ≈0,06–0,08 to ≈0,74–0,81; for NN, R^2 increases to ≈0,85–0,87 for combined predictors. This improvement highlights the disproportionate influence of leverage points in variance-restricted contexts.

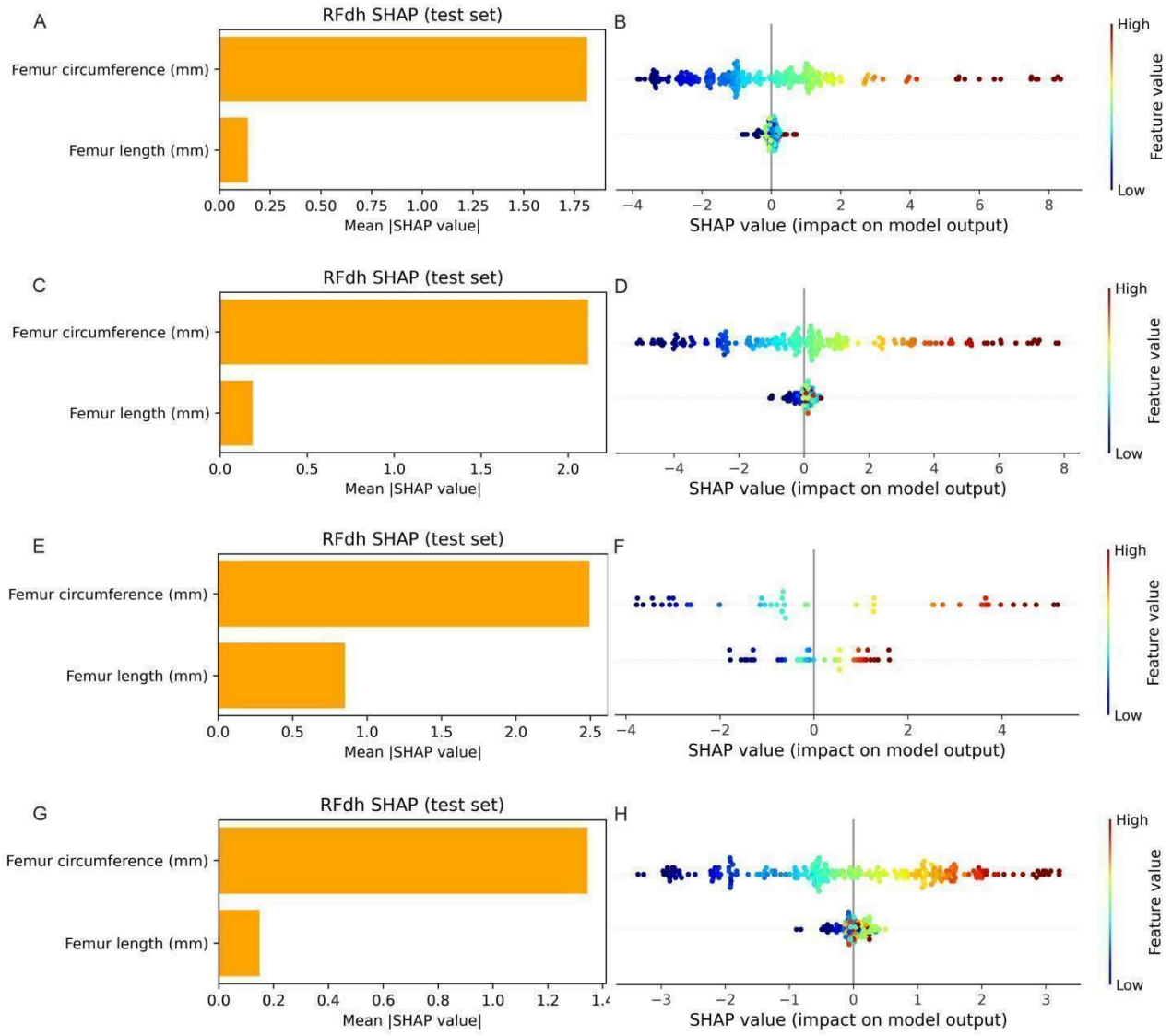
For random forests, extreme values can dominate upper-node splits, leading to poorly generalized terminal partitions when data are sparse. For neural networks, extreme values exert strong influence on gradient updates, distorting weight optimization and increasing generalization error (Goodfellow et al., 2016). Removing such leverage points stabilizes both partitioning and gradient dynamics, reducing variance in parameter estimates and improving interpolation performance.

Summing up, the test set confirms that both RF and NN models perform exceptionally well when trained on datasets preserving broad morphological and body mass diversity. However, restricting training to extant taxa introduces severe instability, particularly for tree-based ensembles. Even within distributionally matched test data, limited variance and leverage effects can induce near-complete predictive degradation.

These results further substantiate a central conclusion of this study: in modelling continuous allometric traits, morphospace breadth and variance structure are primary determinants of generalization performance. Algorithmic sophistication cannot compensate for truncated biological variance. Preserving wide size regimes, through inclusion of extinct taxa and balanced representation, remains essential for stable, biologically meaningful predictive modelling.

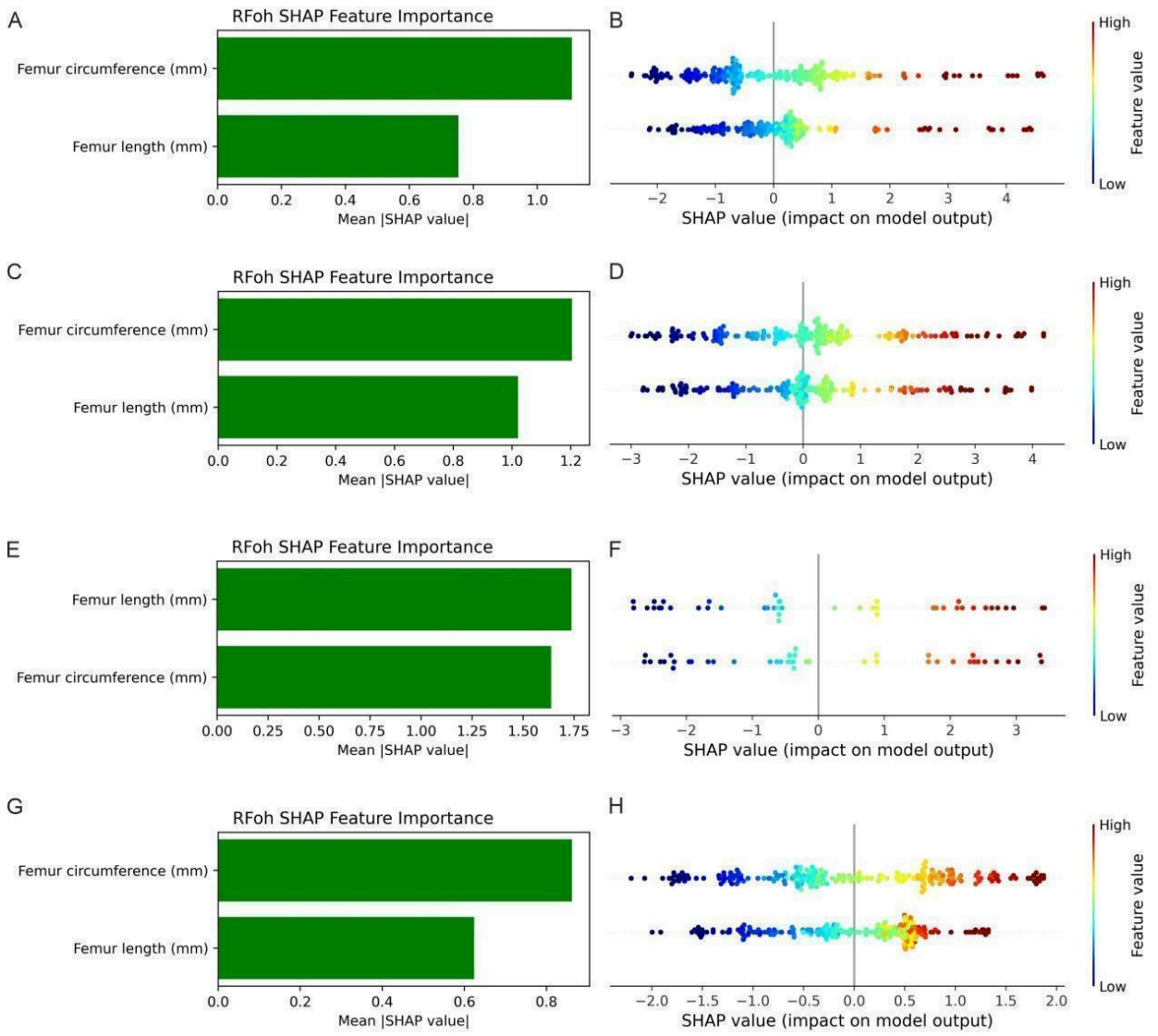
10. Figure S6.

Random forests with default hyperparameters (RFdh). Feature importance output via SHAP for the **test set** (= 25% of the data after 75:25 split) after training with the full training dataset (A, B), after training with the SMOGN-balanced dataset (C, D), after training with the taxa-balanced dataset (E, F) and after training with the extant-only dataset (G, H). Femur circumference emerges as the most influential predictor across algorithms, with high influence across all training datasets.



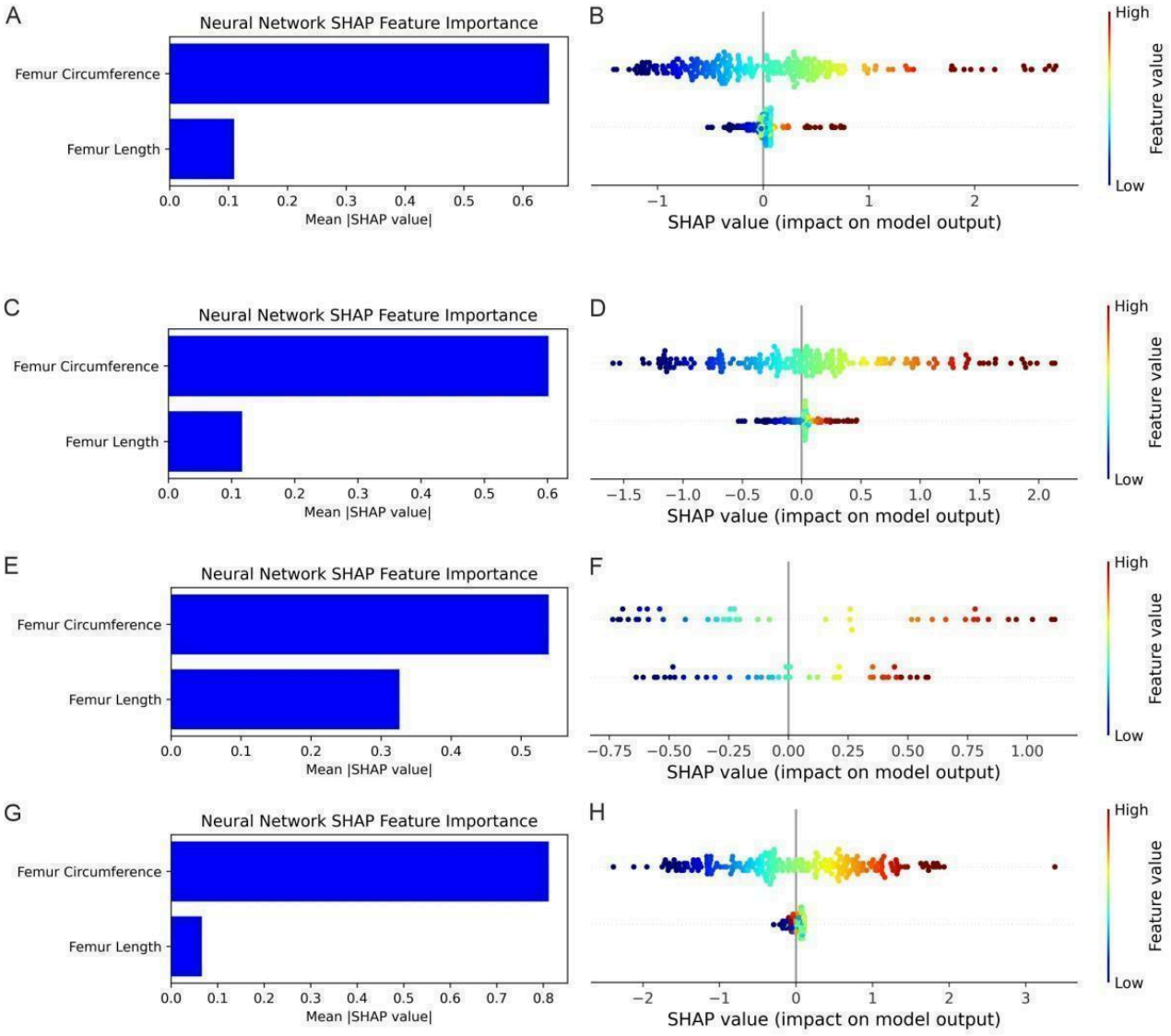
11. Figure S7.

Random forests with optimized hyperparameters (RFoh). Feature importance output via SHAP for the **test set** (= 25% of the data after 75:25 split) after training with the full training dataset (A, B), after training with the SMOGN-balanced dataset (C, D), after training with the taxa-balanced dataset (E, F) and after training with the extant-only dataset (G, H). Femur circumference emerges as the most influential predictor across algorithms, but its influence is not as dominant as seen in RFdh (see Figure S6).



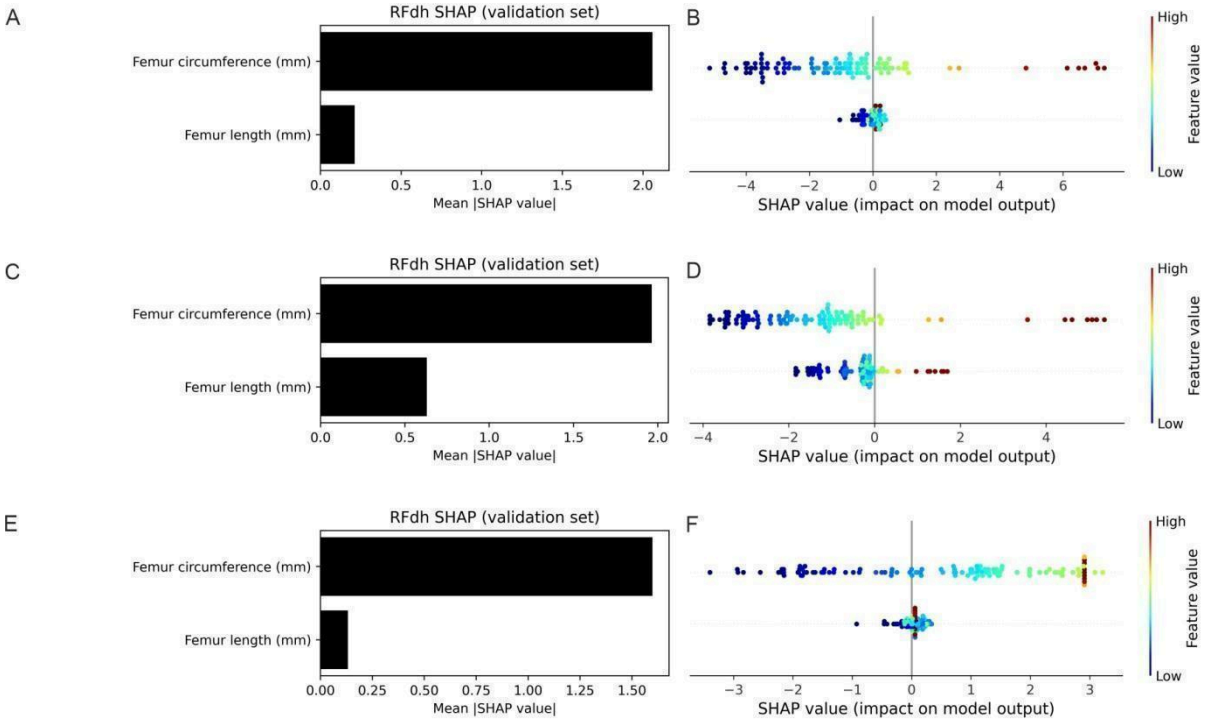
12. Figure S8.

Neural networks. Feature importance output via SHAP for the **test set** (= 25% of the data after 75:25 split) after training with the full training dataset (A, B), after training with the SMOGN-balanced dataset (C, D), after training with the taxa-balanced dataset (E, F) and after training with the extant-only dataset (G, H). Femur circumference emerges as the most influential predictor in the full training, SMOGN-balanced and extant-only datasets. The taxa-balanced dataset (E, F) shows a less-dominant femoral circumference in prediction.



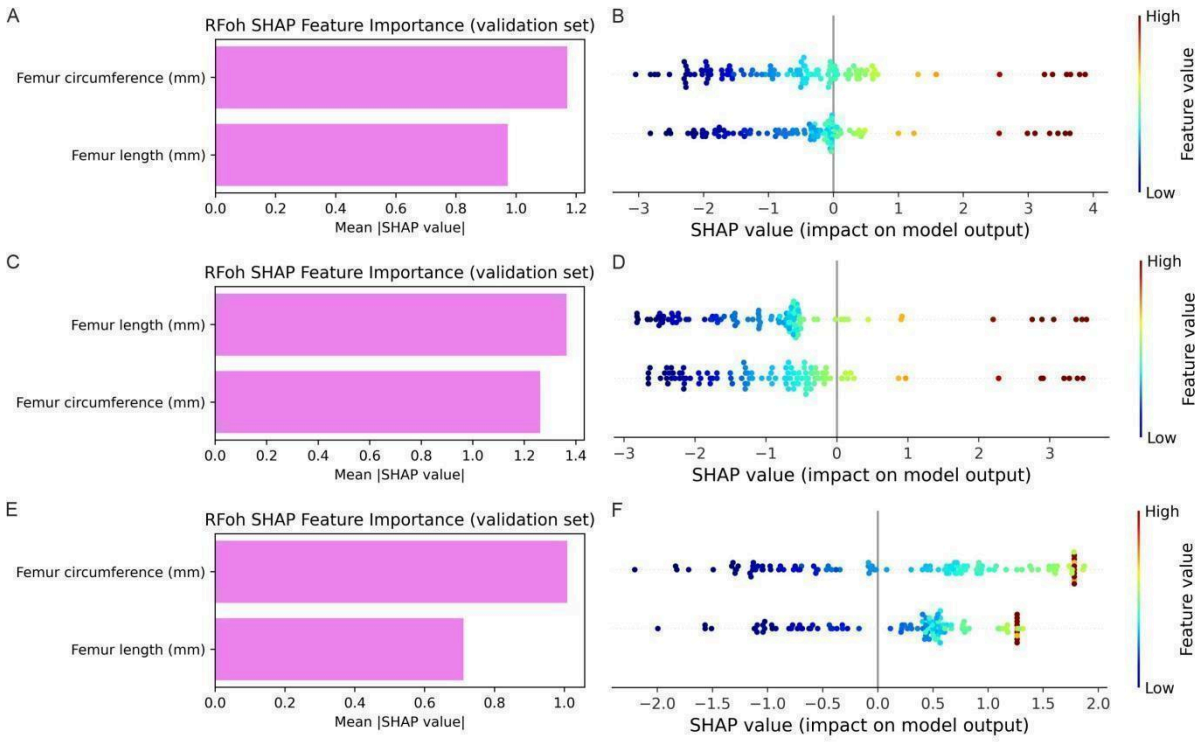
13. Figure S9.

Random forests with default hyperparameters (RFdh). Feature importance output via SHAP for the **validation set** (10% held out from the original raw dataset) after training under the SMOGN-balanced dataset (A, B), after training under the taxa-balanced dataset (C, D) and after training under the extant-only dataset (E, F). Femur circumference emerges as the most influential predictor across algorithms, with high influence across all training datasets.



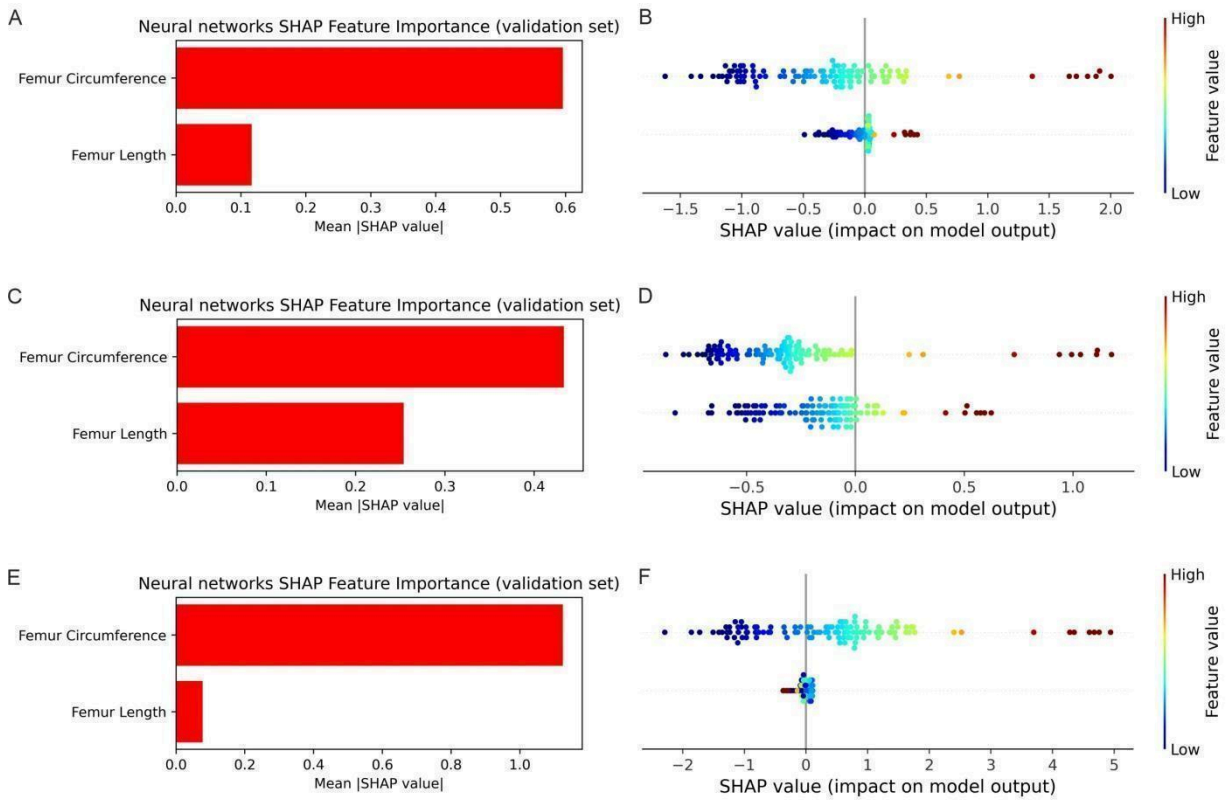
14. Figure S10.

Random forests with optimized hyperparameters (RFoh). Feature importance output via SHAP for the **validation set** (10% held out from the original raw dataset) after training under the SMOGN-balanced dataset (A, B), after training under the taxa-balanced dataset (C, D) and after training under the extant-only dataset (E, F). Femur circumference emerges as the most influential predictor in SMOGN (A, B) and extant-only (E, F) training datasets, reversing the pattern in taxa-balanced (C, D) training dataset, in which femoral length is the most influential predictor, although almost as similar as femoral circumference.



15. Figure S11.

Neural networks. Feature importance output via SHAP for the **validation set** (10% held out from the original raw dataset) after training under the SMOGN-balanced dataset (A, B), after training under the taxa-balanced dataset (C, D) and after training under the extant-only dataset (E, F). Femur circumference emerges as the most influential predictor in SMOGN (A, B) and extant-only (E, F) training datasets, showing less influence in the taxa-balanced (C, D) training dataset.



16. Table S5.

Statistics of the full training dataset before and after log 10 transformation and standardization. Abbreviations: Count., total count of rows/samples; Max., maximum value; Mean, average value; Min., minimum value; Std., standard deviation; 75%, 50%, 15%, samples quartiles.

Before log and standardization			
	Femur length (mm)	**Femur circumference (mm)**	**Body mass (g)**
Count	1009	1009	1009
Mean	82,86938553	26,01245788	61313,19554
Std	161,5866298	58,1196688	438399,8421
Min	6,71	1,35	2,9
25%	24,2	6,05	64,8
50%	40,89	11,75	334
75%	66,35	19,5	1120
Max	1333	570	7700000
After log transformation			
Count	1009	1009	1009
Mean	1,649671642	1,087721653	2,557723362
Std	0,398365863	0,444828081	1,104752395
Min	0,82672252	0,130333768	0,462397998

25%	1,383815366	0,781755375	1,811575006
50%	1,611617111	1,070037867	2,523746467
75%	1,821840927	1,290034611	3,049218023
Max	3,124830149	2,755874856	6,886490725
After standardization transformation			
Count	1009	1009	1009
Mean	0	0	0
Std	1	1	1
Min	-0,471556055	-0,424549704	-0,13991953
25%	-0,363263225	-0,343641969	-0,139778265
50%	-0,259923754	-0,245519822	-0,139163909
75%	-0,102283077	-0,112108131	-0,137370136
Max	7,740433516	9,36442553	17,43265887

17. Table S9.

Cluster statistics after K-means analysis. Abbreviations: BM, body mass; FC, femur circumference; FL, femur length.

Groups	N°of specimens	FL mean	FL std	FC mean	FC std	BM mean	BM std
1	416	21,96639423	7,025989994	5,419134615	2,018156434	66,290625	58,17806139
2	515	62,1867767	24,30704067	18,54293204	7,690354674	1255,556117	1465,054986
3	78	544,2435897	314,7472367	185,1615385	124,4238329	784497,7692	1393487,328

18. Table S10.

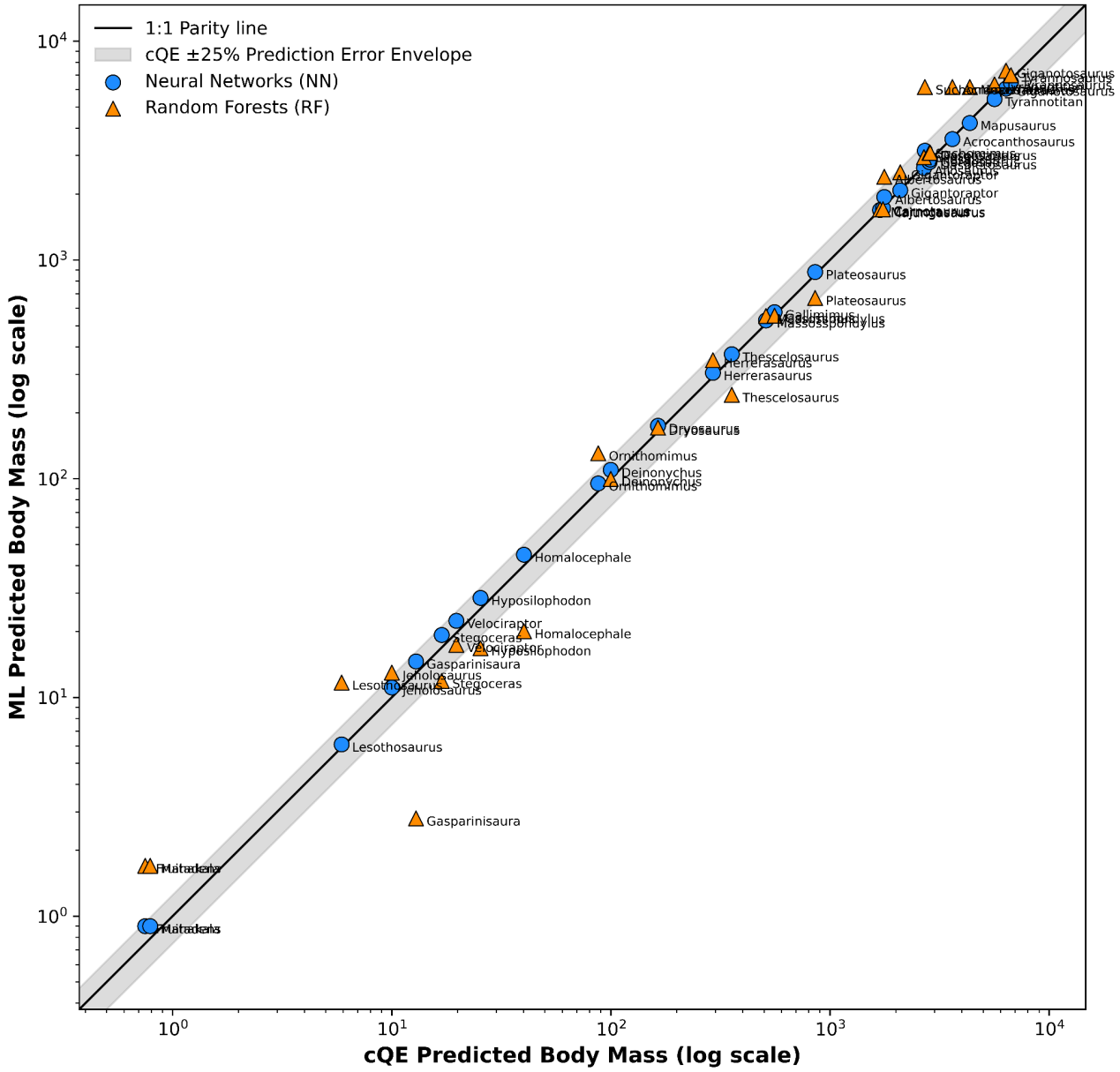
Performance metrics (R^2 and MSE) for the test set resulting from the 75:25 train-test split across machine learning algorithms implemented in this contribution. Abbreviations: BM, body mass; FC, femur circumference; FL, femur length; NN, neural networks; RFdf, random forests with default hyperparameters; RFoh, random forests with optimized hyperparameters.

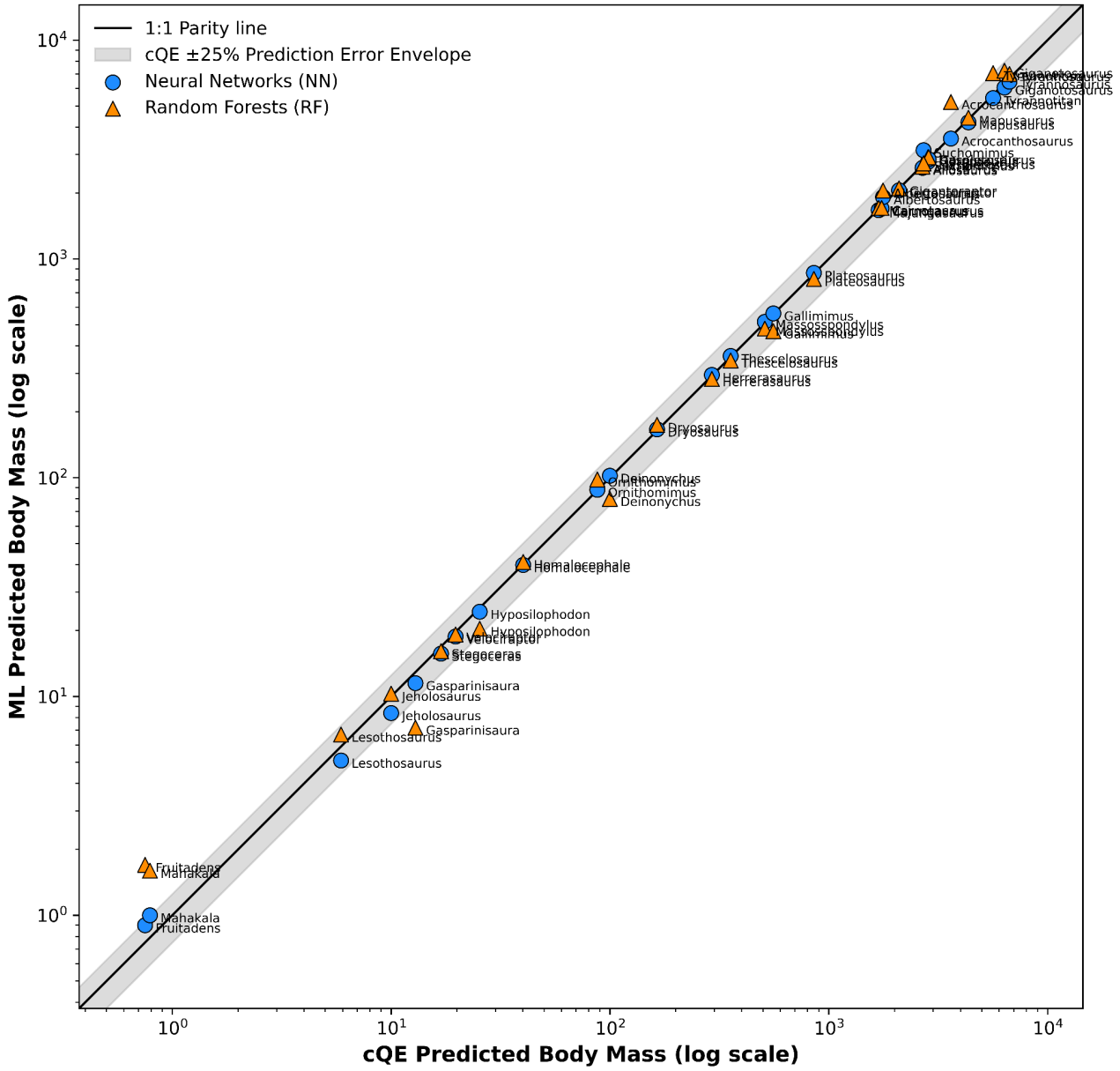
Test set				
MSE				
Variables	**RFdh**	**RFdh SMOGN balanced**	**RFdh taxa balanced**	**RFdh extant**
FL + FC	850731165,9	2383027490	1604064661	78814723,01
FC	405455808,2	3090787607	1620808659	79110047,9
FL	8270573283	3285446225	19820418105	78647761,5
	RFoh	**RFoh SMOGN balanced**	**RFoh taxa balanced**	**RFoh extant**
FL + FC	2234326591	3449970561	6428625397	78376546,9
FC	1517441272	3057814298	6738704179	78895953,3
FL	5510240816	6817286824	15448977254	77835718,4
	NN	**NN SMOGN balanced**	**NN taxa balanced**	**NN extant**
FL + FC	913534144,5	18036088149	828100930,7	23155729,4
FC	399026093,9	9109487048	1668641569	31140175,6
FL	14396664265	47139048979	50013543724	62775694,6

R2				
	RFdh	**RFdh SMOGN balanced**	**RFdh taxa balanced**	**RFdh extant**
FL + FC	0,992	0,995	0,994	0,06
FC	0,996	0,994	0,994	0,06
FL	0,929	0,994	0,935	0,068
	RFoh	**RFoh SMOGN balanced**	**RFoh taxa balanced**	**RFoh extant**
FL + FC	0,98	0,993	0,979	0,071
FC	0,987	0,994	0,978	0,065
FL	0,952	0,987	0,95	0,078
	NN	**NN SMOGN balanced**	**NN taxa balanced**	**NN extant**
FL + FC	0,992	0,967	0,997	0,725
FC	0,996	0,983	0,994	0,631
FL	0,876	0,915	0,838	0,256

19. Figure S12

On the top, Benchmarking under independent, non-circular 3D volumetric-density targets (Campione & Evans, 2020, appendix S1). Eliminating proxy-derived target leakage resolves the artificial linearity into realistic biological scatter reflecting clade-specific variation in skeletal proportions, while neural networks maintain robust scaling trends across large-bodied forms. On the bottom, benchmarking under a hybrid training configuration combining cQE- and VD-derived targets. This mixed regime demonstrates a mediating effect where neural networks achieve strong alignment within the cQE error envelope, whereas random forests display localized, clade-specific deviations at size extremes (e.g., sub-kilogram taxa and hyper-robust theropods).


1:1 Parity line
cQE ±25% Prediction Error Envelope
Neural Networks (NN)
Random Forests (RF)
ML Predicted Body Mass (log scale)
cQE Predicted Body Mass (log scale)
10^0
10^1
10^2
10^3
10^4
Gigantoraptor
Tyrannotitan
Mapusaurus
Acrocanthosaurus
Plateosaurus
Gallimimus
Thescelosaurus
Herrerasaurus
Ornithomimus
Deinonychus
Homalocephale
Hyposilophodon
Velociraptor
Stegoceras
Gasparinisaura
Jeholosaurus
Lesothosaurus



20. Scripts.

The Python resources provided in this contribution include:

- **'eda_unsupervised_code'** (.IPYNB). Notebook to run preprocessing, exploratory data analysis and unsupervised (clustering) algorithms.
- **'rfoh_code'** (.IPYNB). Notebook to run random forests algorithm with optimized hyperparameters.
- **'rfdh_code'** (.IPYNB). Notebook to run random forests algorithm with default hyperparameters.
- **'nn_code'** (.IPYNB). Notebook to run neural networks algorithm.

- **'bm_calculator'** (.IPYNB). Notebook to calculate body mass in biped dinosaurs. Check the 'readme.txt' file for instructions.

To execute the scripts files, the notebooks (.IPYNB) and training file (.py) must be in the same folder. Suggested platforms to run it are presented below. Both are free:

- Google® Colab® (not open source, does not require installation): https://colab.google, or
- Jupyter Notebook© or Jupyter Lab© via Anaconda© Distribution (open source, does require installation): https://www.anaconda.com/download

21. References not included in the main text.

Ribeiro, R.P. (2011). Utility-based regression (Doctoral dissertation). University of Porto, Porto, Portugal.

Torgo, L., Ribeiro, R.P., Pfahringer, B., & Branco, P. (2013). SMOTE for regression. In L. Correia, L.P. Reis, & J. Cascalho (Eds.), Progress in Artificial Intelligence. EPIA 2013. Lecture Notes in Computer Science, vol. 8154, pp. 378–389. Springer, Berlin. https://doi.org/10.1007/978-3-642-40669-0_33