

EcoRETINA ML Workbench PRO

One-Page Executive Summary

Purpose. EcoRETINA ML Workbench PRO is a computing workbench that combines empirical econometrics with modern machine learning. It provides data preparation, model estimation, statistical diagnostics, elasticity analysis, AI-assisted analysis, and real-time carbon-footprint auditing via CodeCarbon. The workflow is organized into four sequential operational tabs.

1. Data & Pre-Processing. Users can ingest CSV, XLSX, or JSON datasets and configure delimiters. The tab supports:

- Train/test splitting (default 80/20) and K -fold cross-validation (default $K = 5$).
- Distribution inspection and outlier treatment through clipping (Winsorization) or row deletion (truncation).
- Categorical-variable handling through reference-category selection, dropping columns, or one-hot/dummy encoding.
- Manual variable removal and feature scaling with `StandardScaler` (Z -score) or `MinMaxScaler` (0–1).
- *Cautionary note:* Scaling the target variable can make MAPE sharply increase when values approach zero; RMSE is recommended in that situation.

2. Algorithms, Hyperparameters & Feature Routing. Users select the dependent variable, name each run, configure parameters, and launch or stop model estimation.

- **EcoRETINA framework:** Supports alternative loss functions, linear/logit/probit estimation, robust covariance estimators (HC0–HC3), interaction terms, logarithmic and ReLU transformations, power expansions, and multiple approaches to handling zero-valued predictors, constrained by an R^2 ceiling.
- **Benchmarked models:** OLS, Lasso, Ridge, ElasticNet, XGBoost, Random Forest, and Multi-Layer Perceptrons (MLP), with model-specific hyperparameters.
- **Feature routing:** Separated into continuous and dummy variables for EcoRETINA, and a unified predictor list for other models.

3. Compare Results & Statistical Reports. The benchmark dashboard compares executed runs across fit, error, and environmental metrics: Train/Test R^2 , adjusted Test R^2 , RMSE, MAPE, R^2/CO_2 , and total CO_2 emissions. Optimal values are highlighted automatically. The detailed report window provides:

- Train/test performance divergence, execution time, and Shapiro–Wilk residual-normality testing.
- EcoRETINA marginal effects and elasticities evaluated at sample means.
- Parameter coefficients, t -statistics, p -values, confidence intervals, or tree/MLP feature importances.
- Closed-form estimated formulas and graphical diagnostics (actual vs. predicted, residual distributions, top elasticities).

4. Predict — New Data. A trained model can be selected and applied to an unlabelled dataset without the target variable. The system checks for required columns, applies learned transformations, generates a `Predicted_[Target]_[RunID]` column, provides an interactive viewer, and exports predictions to CSV.

AI Copilot & Operational Controls. The AI Copilot acts as an interactive econometrician connecting to leading LLM providers asynchronously. It analyzes runs, diagnoses overfitting and residual normality, suggests parameter adjustments, and executes actions via embedded “Magic Tags”. API keys are strictly held in volatile session memory. Reproducibility is ensured via an activity log with TXT export and a 15-state undo/redo buffer.

Bottom line: EcoRETINA ML Workbench PRO is designed as an end-to-end research environment: *prepare data* $\rightarrow$ *configure and run econometric/ML models* $\rightarrow$ *compare predictive, statistical, and*

environmental performance $\rightarrow$ *inspect detailed diagnostics and elasticities* $\rightarrow$ *generate out-of-sample predictions*, with AI assistance and reproducibility controls integrated throughout.

EcoRETINA

Comprehensive Operational Manual & Econometric Guide

Maxime ELEK **Jules BERNADAC**
Mines Saint-Étienne Mines Saint-Étienne
ICAE, Universidad Complutense ICAE, Universidad Complutense
maxime.elek@gmail.com jules.bernadac@gmail.com

August 2026

Cloud Application: <https://ecoretina-cloud.onrender.com/> | **GitHub Repository:**
<https://github.com/Eco-Retina/Eco-Retina>

Acknowledgements: The authors warmly acknowledge previous contributors to Eco-RETINA, as well as colleagues and faculty at ICAE and Universidad Complutense of Madrid for their feedback and support.

Contents

1	Introduction	3
2	System Controls, Activity Logging, and AI Copilot	5
2.1	Top Control Header	5
2.2	System Activity Log & Tutorial	5
2.3	AI Copilot Engine	5
2.3.1	Prerequisites, Authentication, and Security	5
2.3.2	Supported Models and Automated Capabilities	6
3	Tab 1: Data & Pre-Processing	6
3.1	1. Dataset Ingestion	6
3.2	2. Sample Split Strategies	7
3.3	3. Outlier Management	7
3.4	4. String & Categorical Variables	7
3.5	5. Drop Any Variable(s) & 6. Scaling / Normalization	7
4	Tab 2: Algorithms, Hyperparameters, and Feature Routing	8
4.1	Algorithm Hyperparameters	9
4.1.1	EcoRETINA	9
4.1.2	Linear, Regularization, and Machine Learning Models	10
4.2	Feature Selection Routing	10
5	Tab 3: Compare Results & Detailed Statistical Reports	10
5.1	Global Benchmark Dashboard	10
5.2	Interactive Row Context Menu	11
5.3	Detailed Statistical Report Window	11
6	Tab 4: Predict (Inference on New Data)	12

1 Introduction

EcoRETINA is a lightweight graphical user interface (GUI) software designed to streamline the modeling, estimation, inference, and out-of-sample predictive comparison of alternative econometric and machine learning algorithms.

This distribution is a cloud-based, portable adaptation of the broader EcoRETINA framework [1]. While the general version requires full local setup on a physical machine, this cloud version offers high versatility by running directly within the browser without any local installation. It features the exact same core model architectures, but exposes a curated, simplified set of hyperparameters for faster online workflows.

Methodological Note

Cloud Resource Note: Because this web version is hosted on a Render server with limited RAM, it is optimized for small to medium-sized datasets. For very large datasets or memory-intensive computations, the standalone desktop version is recommended.

The platform natively integrates real-time computational carbon footprint auditing via **CodeCarbon** [5], multi-provider Large Language Model (LLM) analytical assistance, detailed regression diagnostics, and interactive elasticity analysis.

The graphical interface is structured around a top utility header, a persistent background logging system, an interactive AI Copilot sidebar, and four sequential operational tabs:

1. **1. Data & Pre-Processing:** File ingestion, sample splitting strategies, distribution inspections, outlier mitigation, categorical dummy generation, manual feature dropping, and feature scaling.
2. **2. Algorithms & Params:** Interactive algorithm dashboard, hyperparameter parameterization, dependent variable assignment, custom run labeling, and dedicated feature routing.
3. **3. Compare Results:** Multi-run comparative matrix, dynamic performance indicators, historical CSV exports, and statistical report windows.
4. **4. Predict (New Data):** Out-of-sample inference pipeline and augmented CSV prediction exports.

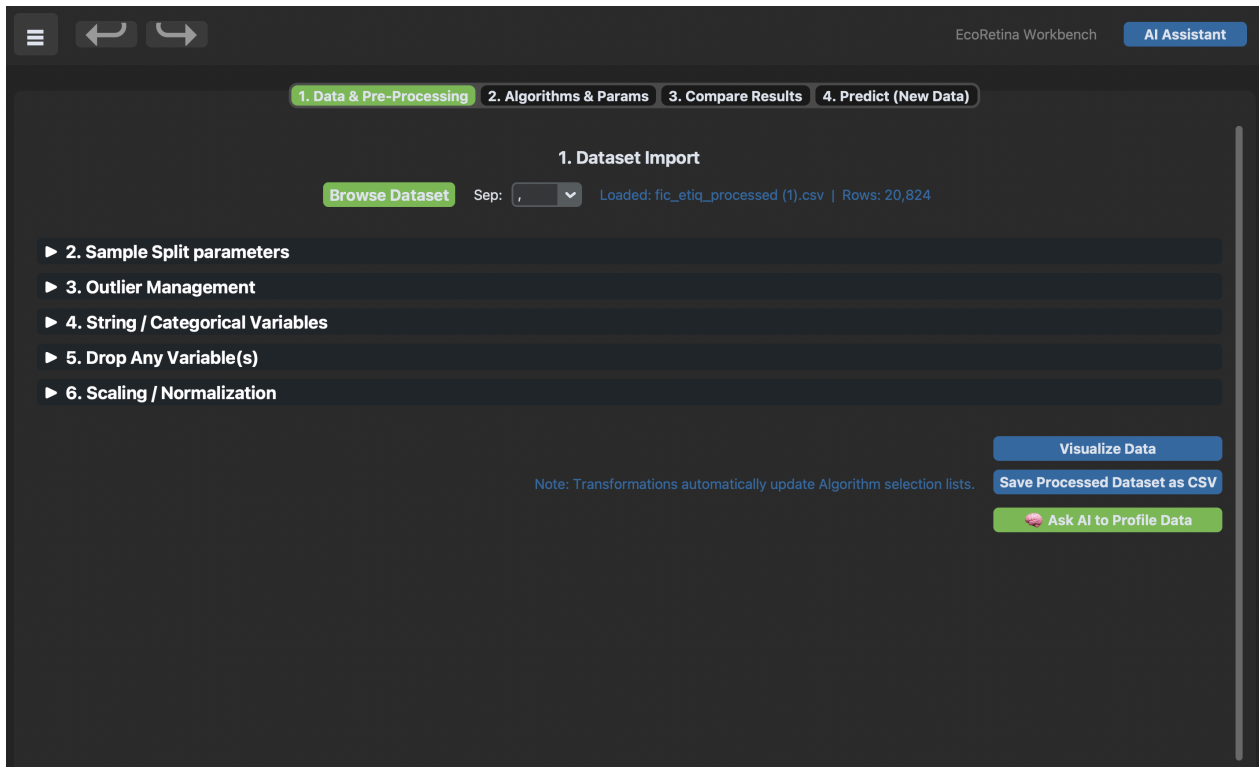


Figure 1: Global Workbench interface with status indicators and top controls

2 System Controls, Activity Logging, and AI Copilot

2.1 Top Control Header

- **Navigation Menu Button ($\equiv$):** Opens a popup menu to switch between the *Main Workspace*, the dedicated *System Activity Log*, and the interactive *Tutorial & Documentation* module.
- **Undo Button ($\leftrightarrow$):** Reverts data mutations to the exact preceding dataset state (maintains a memory buffer of up to 15 sequential states).
- **Redo Button ($\hookrightarrow$):** Restores an operation reversed by the Undo command.
- **AI Assistant Button:** Toggles the right-hand analytical Copilot panel.

2.2 System Activity Log & Tutorial

Accessible via the main menu, the **System Activity Log** provides a console tracking every data mutation, parameter initialization, pipeline execution, and inference step.

- **Export Log to TXT:** Saves the operational trail with timestamps for auditability and research reproducibility.
- **Clear Log:** Flushes the current session log buffer.

2.3 AI Copilot Engine

The **AI Copilot** serves as an interactive econometrician and real-time operational assistant. It connects to leading Large Language Model (LLM) providers via dedicated asynchronous worker threads, ensuring uninterrupted interface responsiveness and token-by-token response streaming.

2.3.1 Prerequisites, Authentication, and Security

To activate the assistant, users must authenticate with an active API key from their preferred AI platform:

- **Obtaining Access Credentials:** API tokens can be generated via official developer dashboards:
 - *Google AI Studio:* Keys typically prefixed with `AIza...`
 - *OpenAI Platform:* Keys typically prefixed with `sk-...`
 - *Anthropic Console:* Keys prefixed with `sk-ant-...`
 - *Groq Cloud Console:* Keys prefixed with `gsk_...`
 - *Mistral AI Platform:* Standard API bearer tokens.
- **Interface Setup:** Open the sidebar by clicking **AI Assistant**, select the target provider, paste the API key into the secure password-masked entry, and click **Connect**.
- **Security & Ephemeral Storage:** Keys are held strictly in volatile session memory. They are never written to disk, logged in activity trails, or shared with third parties beyond the direct HTTPS endpoint of the chosen provider.
- **Provider Switching:** Clicking **Disconnect** instantly purges active keys and conversational history from runtime memory, allowing seamless switching across models without restarting the software.

2.3.2 Supported Models and Automated Capabilities

The engine supports the latest cost-effective and high-throughput production models:

- **LLM Backends:** Google Gemini (`gemini-2.5-flash`), OpenAI (`gpt-4o-mini`), Anthropic Claude (`claude-3-5-sonnet`), Mistral AI (`mistral-large-latest`), and Groq (`llama-3.3-70b-versatile`).
- **Analyze Last Run:** Automatically extracts regression outputs (R^2 , RMSE, MAPE, Shapiro-Wilk p -values, CO₂ emissions) to synthesize overfitting diagnostics and residual normality assessments.
- **Auto-Tune Hyperparameters:** Compares in-sample versus out-of-sample loss metrics to formulate three concrete parameter adjustments (e.g., learning rates, regularization penalties, tree depths).
- **Autonomous Action Execution (Magic Tags):** Beyond standard text generation, the agent can actively modify the dataset and workspace state (such as dropping collinear variables, removing missing values, applying dummy encodings, or launching training runs) via discrete embedded command tags.

3 Tab 1: Data & Pre-Processing

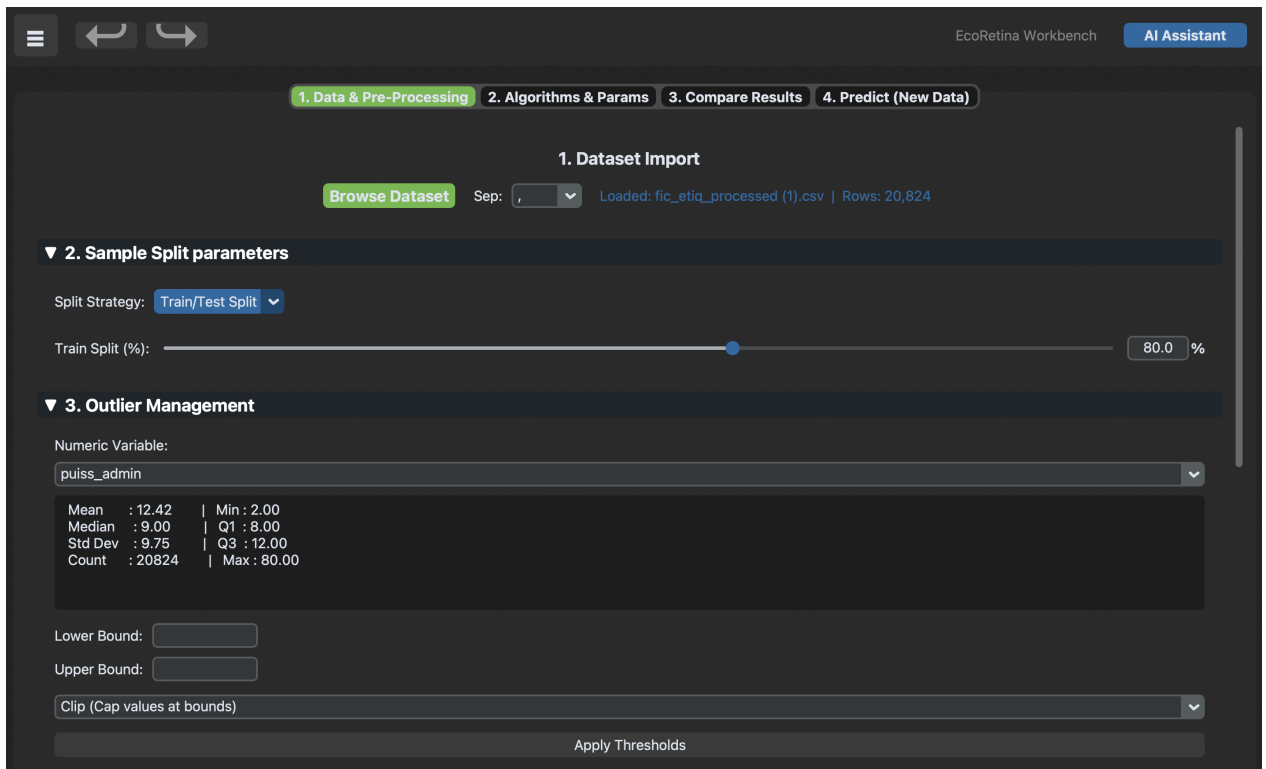


Figure 2: Sampling configuration, distribution inspections, outlier rules, and variable encoding

3.1 1. Dataset Ingestion

- **Browse Dataset Button:** Ingests files formatted as `.csv`, `.xlsx`, or `.json`.
- **Sep (Delimiter Dropdown):** Allows manual delimiter selection (`Auto`, `,`, `;`, `\t`, `|`). The `Auto` mode samples initial lines to determine the most frequent delimiter.

- **Ask AI to Profile Data Button:** Summarizes missing values, types, and descriptive statistics, prompting the Copilot for a data audit.

3.2 2. Sample Split Strategies

- **Train/Test Split:** Divides the dataset into training and evaluation sets.
 - *Train Split (%) Slider & Field:* Configures the split ratio from 50.0% to 100.0% (default: 80.0%). Setting 100% evaluates metrics in-sample.
- **K-Fold Cross Validation:** Partitions the dataset into K equal subsets to guard against sample selection bias.
 - *Number of Folds (K):* Configures the integer fold count (default: $K = 5$).

3.3 3. Outlier Management

- **Numeric Variable Selector:** Inspects any continuous variable and displays the count, mean, median, standard deviation, quartiles (Q_1, Q_3), min, and max.
- **Lower Bound & Upper Bound Inputs:** Sets numerical threshold boundaries $[\underline{B}, \overline{B}]$.
- **Action Menu:**
 - *Clip (Cap values at bounds):* Performs Winsorization by replacing values $x_i < \underline{B}$ with $\underline{B}$ and $x_i > \overline{B}$ with $\overline{B}$, retaining sample size N .
 - *Drop (Delete rows outside bounds):* Performs Truncation by removing rows where $x_i \notin [\underline{B}, \overline{B}]$.
- **Apply Thresholds Button:** Applies transformations and updates available downstream variable lists.

3.4 4. String & Categorical Variables

- **String Variable Selector:** Displays distinct categories and absolute instance counts.
- **Reference Category Dropdown:** Selects the baseline category c_{ref} omitted during One-Hot encoding to prevent exact dummy variable collinearity (the Dummy Variable Trap).
- **Drop Column Button:** Drops non-numeric columns from the dataset.
- **Encode (Dummies) Button:** Converts the categorical column into $k - 1$ binary (0/1) indicator variables.

3.5 5. Drop Any Variable(s) & 6. Scaling / Normalization

- **Drop Selected Button:** Multi-select listbox (using **Ctrl** or **Shift**) to prune redundant predictors.
- **Variables to Scale Menu:** Selects transformation scope: *All Numeric Predictors (No Target)*, *Target Variable ONLY*, or *All Numeric Variables*.
- **Method Dropdown:**
 - *StandardScaler (Z-Score):* Centers and standardizes data such that $\mu = 0$ and $\sigma = 1$: $z = (x - \mu)/\sigma$.
 - *MinMaxScaler (0-1):* Scales values into the unit interval: $x_{\text{scaled}} = (x - x_{\min})/(x_{\max} - x_{\min})$.

Cautionary Note

Scaling the Target Variable (Y) around zero can cause the Mean Absolute Percentage Error (MAPE) to increase sharply due to near-zero division. Under such conditions, evaluate model fit using Root Mean Squared Error (RMSE).

4 Tab 2: Algorithms, Hyperparameters, and Feature Routing

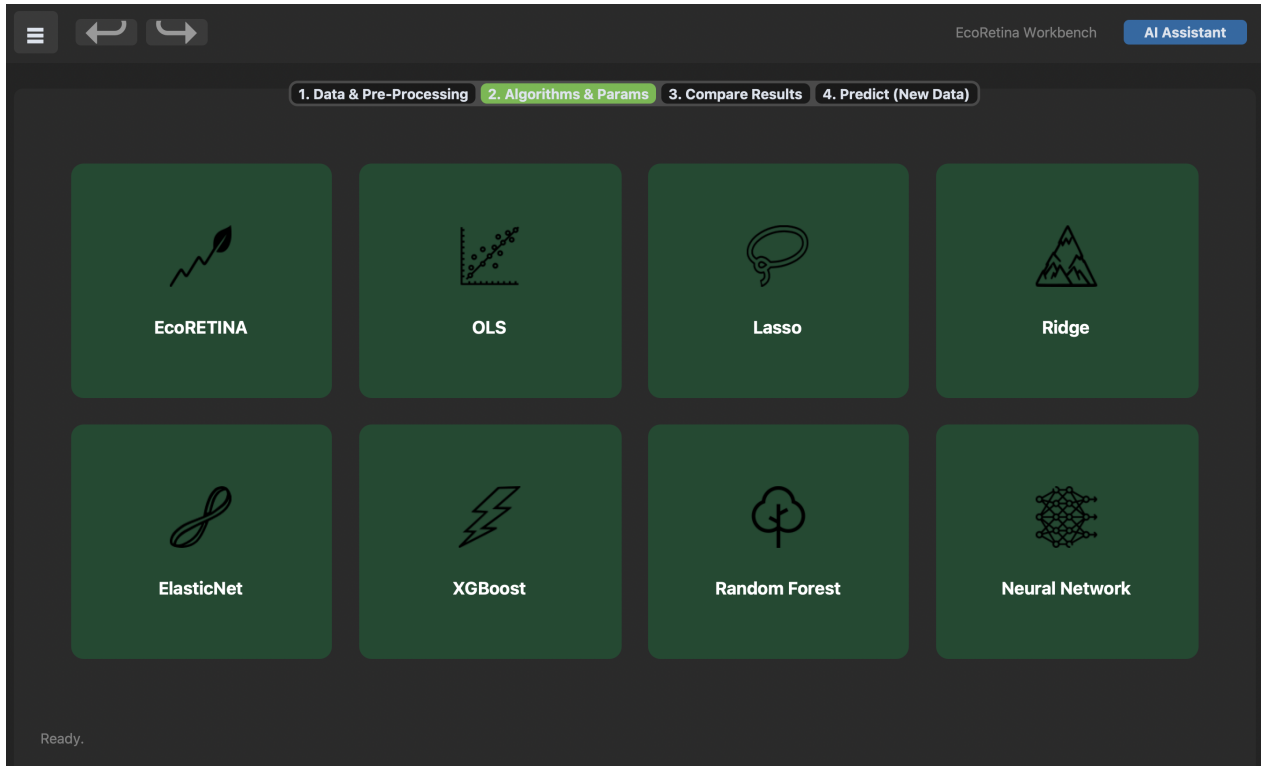


Figure 3: Algorithm Dashboard

Selecting an algorithm card opens a dedicated parameter configuration panel:

- **Target Variable (Y) Selector:** Sets the dependent column.
- **Run Name Entry:** Custom alphanumeric identifier for the run. If left empty, the system automatically assigns a timestamped label (e.g., EcoRETINA_143022).
- **Top Header Buttons:**
 - **► Run Model:** Launches estimation in a background worker thread.
 - **Stop Button:** Aborts execution without crashing the application.
 - **View Report Button:** Directly opens the detailed analytical report for the most recent run of the selected algorithm family.

4.1 Algorithm Hyperparameters

4.1.1 EcoRETINA

The screenshot displays the EcoRETINA Workbench interface. At the top, there are navigation tabs: '1. Data & Pre-Processing', '2. Algorithms & Params' (which is active), '3. Compare Results', and '4. Predict (New Data)'. Below the tabs, the 'EcoRETINA' logo is centered. To the right of the logo are buttons for 'Run model' (green) and 'View Report' (blue). The 'Target Variable (Y):' is set to 'puiss_admin'. The 'Run Name:' is 'Auto (EcoRETINA_...)'. Below these, there are several configuration options: 'Loss:' (mse), 'Reg Type:' (linear), 'Cross Dummy:' (False), 'Cov Type:' (nonrobust), 'Zero Handling:' (prevent_division), 'Add Logs (ln):' (False), 'PaAdd ReLU:' (False), 'Max R²:' (0.99), 'Grid:' (0.005), 'Seed:' (8), 'Max Inst.:' (100000), and 'Max Reg:' (100). There are also checkboxes for 'Select All Continuous' and 'Select All Dummies'. Below these are two lists of variables: 'Continuous Variables' and 'Dummy Variables', both containing the same list of variables: 'puiss_admin', 'puiss_max', 'conso_urb_93', 'conso_exurb', 'conso_mixte', 'co2_mixte', 'co_typ_1', 'hc', 'nox', 'hcnnox', 'ptcl', and 'masse_ordma_min'. The interface is dark-themed with green accents.

Figure 4: EcoRetina algorithm

- **Loss:** Objective optimization metric (mse, mae, MAPE, AIC, BIC).
- **Reg Type:** Estimation framework: linear (OLS linear basis), logit, or probit.
- **Cross Dummy:** (True/False) Enables interaction terms between continuous predictors and categorical dummy indicators ($X_{\text{cont}} \times D$).
- **Cov Type:** Robust covariance standard error estimation method: nonrobust, HC0, HC1, HC2, or HC3 (White heteroskedasticity-consistent estimators). [4]
- **Zero Handling (handle_zeros):** Governs how continuous features with zero entries ($X_j = 0$) are processed during non-linear transformation:
 - *prevent_division* (Default): Identifies columns containing zero values (cols_with_zeros_) and algorithmically filters out all candidate interactions requiring negative exponents (X_j^p with $p < 0$) on those specific features. This eliminates ZeroDivisionError and undefined arithmetic while preserving all observations (N) and retaining unperturbed original zero values for positive powers, linear terms, and interaction products.
 - *translate*: Applies a uniform affine coordinate shift $X' = X + |\min(X)| + 1.0$ (or custom epsilon) across the entire variable to ensure strictly positive support.
 - *drop_rows*: Automatically prunes any row where a continuous feature equals zero.
- **Add Logs (ln):** (True/False) Adds natural log-transformed features $\ln(X)$.
- **Add ReLU:** (True/False) Generates rectified linear basis activations $\max(0, X - c)$ to capture structural regime changes.

- **Params (list):** Power expansion parameter grid, formatted as `[-1.0, 0.0, 1.0]`.
- **Max R²:** Threshold ceiling (default: 0.99) to halt candidate generation upon convergence and avoid multicollinearity.
- **Grid, Seed, Max Inst., Max Reg:** Fine-tunes optimization step sizes, pseudorandom generator initialization, maximum observation iterations, and maximum retained candidate regressors.

4.1.2 Linear, Regularization, and Machine Learning Models

- **OLS:** `Fit Intercept (True/False)` includes an unconstrained constant vector β_0 .
- **Lasso (L_1), Ridge (L_2), ElasticNet ($L_1 + L_2$):**
 - **Alpha:** Penalty parameter magnitude controlling regularization shrinkage.
 - **L1 Ratio:** (ElasticNet only) Weight between L_1 and L_2 penalties (0.5 represents an equal balance).
 - **Max Iterations & Tolerance:** Numerical solver convergence criteria.
 - **Solver:** (Ridge only) Optimization routine (`auto`, `svd`, `cholesky`, `lsqr`, `sag`, `saga`).
- **XGBoost:** [3] `N Estimators` (number of gradient boosted trees), `Max Depth` (maximum tree depth), `Learning Rate ( $\eta$ )`, `Subsample` (row subsampling ratio), `Colsample By Tree` (feature subsampling ratio), `Gamma` (minimum split loss reduction), `Reg Alpha ( $L_1$ )`, and `Reg Lambda ( $L_2$ )`.
- **Random Forest:** `N Estimators`, `Max Depth` (0 signifies unlimited expansion), `Min Samples Split`, `Min Samples Leaf`, and `Max Features` (`1.0`, `sqrt`, `log2`).
- **Neural Network (MLP):** `Hidden Layers` (e.g., 100,50), `Activation` (`relu`, `tanh`, `logistic`, `identity`), `Solver` (`adam`, `sgd`, `lbfgs`), `Alpha ( $L_2$  penalty)`, and `Learning Rate Init`.

4.2 Feature Selection Routing

- **For EcoRETINA:** Two independent, mutually exclusive listboxes isolate **Continuous Variables** from **Dummy Variables**.
- **For Other Models:** A unified **Predictor Variables (Features)** listbox selects active explanatory columns.

—

5 Tab 3: Compare Results & Detailed Statistical Reports

5.1 Global Benchmark Dashboard

The comparison table tracks and contrasts all executed runs across ten metrics:

1. **Run Name & Algo:** Run identifier and model architecture.
2. **Train R² & Train MAPE:** In-sample goodness-of-fit and percentage error.
3. **Test R² & Test Adj R²:** Out-of-sample explanatory power, adjusted for feature degrees of freedom:
$$\bar{R}^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1} \quad (1)$$
4. **Test RMSE & Test MAPE:** Out-of-sample loss metrics.
5. **R²/CO₂ Ratio & CO₂ (kg):** Computational environmental telemetry computed via CodeCarbon.

Methodological Note

Best Metric Highlighting: The platform scans the table across all columns and highlights optimal metrics with an asterisk (*). Higher values are optimal for R^2 and R^2/CO_2 ; lower values are optimal for RMSE, MAPE, and CO_2 .

5.2 Interactive Row Context Menu

Right-clicking (or two-finger tapping on macOS) any row in the Treeview opens an action menu:

- **View Statistical Report:** Launches the comprehensive multi-tab analytical report window.
- **Rename Run:** Opens an input modal to rename the run ID across memory, table rows, and inference dropdowns.
- **Delete Row (or Delete/Backspace keys):** Removes the selected run from memory and the visual table, recalculating optimal metric asterisks.
- **Export Comparison CSV:** Downloads the benchmark matrix as a CSV file.

5.3 Detailed Statistical Report Window

The popup report displays three diagnostic tabs:

1. Statistical Report Tab:

- *Performance Metrics:* Train vs. Test divergence, execution time, and the **Shapiro-Wilk residual normality test** ($p > 0.05$ indicates normally distributed errors).
- *Econometric Elasticities & Marginal Effects (EcoRETINA continuous features):* [?] Numerical evaluation at sample means:

$$\text{Marginal Effect} = \left. \frac{\partial \hat{Y}}{\partial X_j} \right|_{\bar{X}}, \quad \text{Elasticity (\%)} = \left. \frac{\partial \hat{Y}}{\partial X_j} \right|_{\bar{X}} \times \left(\frac{\bar{X}_j}{\bar{Y}} \right) \quad (2)$$

- *Significance Tables:* Displays parameter coefficients $\hat{\beta}_j$, Student's t -statistics, p -values, and 95% confidence intervals (or feature importance magnitudes for trees and neural nets).

2. Visual Formula Tab: Displays the estimated linear equation:

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^p \hat{\beta}_j X_j \quad (3)$$

3. Graphical Analysis Tab: Displays three integrated diagnostic plots:

- *Actual vs. Predicted:* Scatter plot against the 45° line with a 95% confidence interval band ($\pm 1.96 \cdot \hat{\sigma}_\epsilon$).
- *Residuals Distribution:* Error histogram with empirical Shapiro-Wilk p -values.
- *Top 10 Elasticities / Importances:* Horizontal bar chart showing directional impacts (green for positive, red for negative).

6 Tab 4: Predict (Inference on New Data)

- **1. Browse New Dataset Button:** Ingests an unlabelled evaluation dataset (omitting the target variable Y).
 - **2. Select Trained Model Dropdown:** Selects a trained run from the session registry. Missing required columns are flagged automatically.
 - **3. Run Prediction Button:** Applies the learned model transformations and appends a new prediction column: `Predicted_[Target]_[RunID]`.
 - **4. Visualize Results Button:** Displays an interactive data viewer for the augmented dataset.
 - **5. Export Predictions to CSV Button:** Exports the inference dataset with predicted values to a CSV file.
-

Acknowledgements and Contributions

The authors express their deepest gratitude to **Prof. Teodosio Pérez-Amaral** (Universidad Complutense de Madrid / ICAE) for his guidance, supervision, and methodological vision throughout this project. We also extend our warmest thanks to **Javier Capilla**, **Ángel Alcaraz**, **Ángel Valarezo**, and **Alfredo García-Hiernaux** for their foundational work on the Eco-RETINA algorithm, as well as to **Jules Bernadac** and **Maxime Elek** for software development, cloud deployment, and empirical benchmark evaluations.

References

- [1] Capilla, J., Alcaraz, A., Valarezo, Á., García-Hiernaux, A., & Pérez-Amaral, T. (2025). *Eco-RETINA: A Green Flexible Algorithm for Model Building*. ICAE Working Paper Series, WP 2501, Universidad Complutense de Madrid.
- [2] Wooldridge, J. M. (2020). *Introductory Econometrics: A Modern Approach* (7th ed.). Boston, MA: Cengage Learning.
- [3] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.
- [4] White, H. (1980). A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity. *Econometrica*, 48(4), 817–838.
- [5] Lacoste, A., Luccioni, A., Schmidt, V., & Dandres, T. (2019). Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*.