3 Methodology

3.1 Introduction

This subsection presented a detailed exposition of the methodological framework adopted in this research. The workflow is organized into two principal phases:

- **Exploratory Modeling Approaches:** Sequential implementation and assessment of individual machine learning methods to establish performance baselines and extract insights.
- **Integrated Ensemble Pipeline:** Design and deployment of a unified pipeline that amalgamates preprocessing, feature transformation, hyperparameter optimization, model stacking, and residual correction.

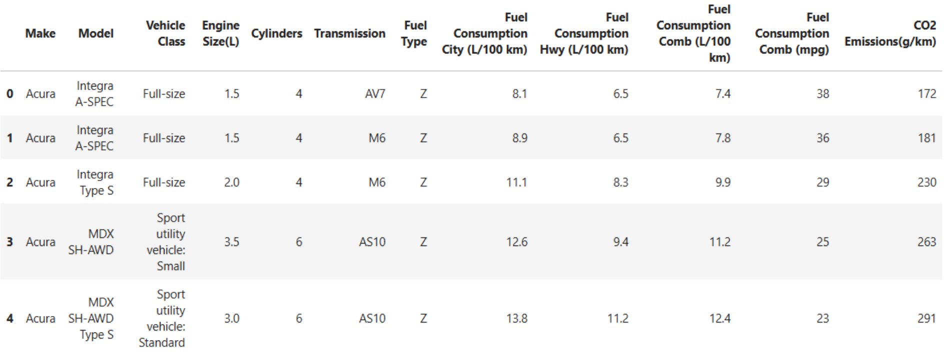
Each section provides a clear exposition of preprocessing, model construction, training strategies, and evaluation metrics.

3.2 Dataset Description

It comprises detailed information about 27813 vehicles, which includes vehicle attributes and their corresponding CO_2 emissions. The dataset includes the following major attributes:

1. **Make:** Manufacturer name
2. **Model:** Vehicle model
3. **Vehicle Class:** Type (SUV, sedan, truck, etc.)
4. **Engine Size (L):** Engine size (in liters)
5. **Cylinders:** Number of engine cylinders
6. **Transmission:** Type of transmission (automatic, manual, etc.)
7. **Fuel Type:** Type of fuel used (gasoline, diesel, electric, etc.)
8. **Fuel Consumption (City, Hwy, Combined(in L/100 km and mpg):** Measured in L/100 km and mpg (miles per gallon)
9. CO_2 **Emissions (g/km):** Target variable for prediction

Dataset at a glance:



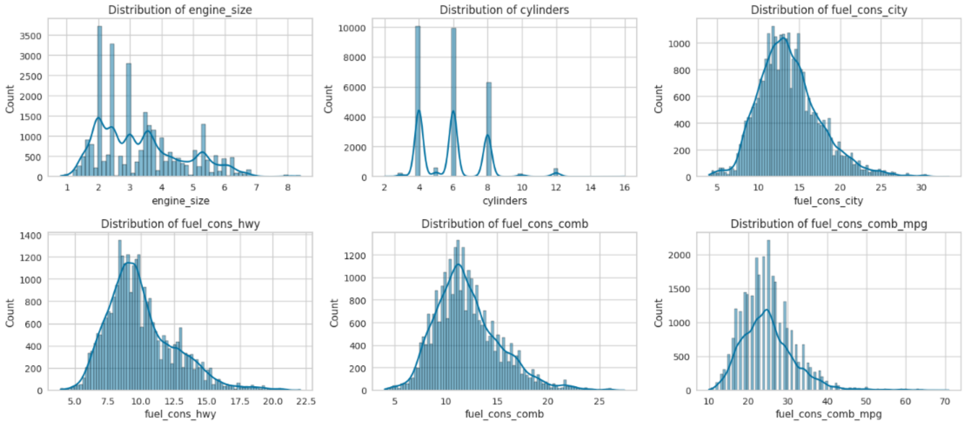
	Make	Model	Vehicle Class	Engine Size(L)	Cylinders	Transmission	Fuel Type	Fuel Consumption City (L/100 km)	Fuel Consumption Hwy (L/100 km)	Fuel Consumption Comb (L/100 km)	Fuel Consumption Comb (mpg)	CO2 Emissions(g/km)
0	Acura	Integra A-SPEC	Full-size	1.5	4	AV7	Z	8.1	6.5	7.4	38	172
1	Acura	Integra A-SPEC	Full-size	1.5	4	M6	Z	8.9	6.5	7.8	36	181
2	Acura	Integra Type S	Full-size	2.0	4	M6	Z	11.1	8.3	9.9	29	230
3	Acura	MDX SH-AWD	Sport utility vehicle: Small	3.5	6	AS10	Z	12.6	9.4	11.2	25	263
4	Acura	MDX SH-AWD Type S	Sport utility vehicle: Standard	3.0	6	AS10	Z	13.8	11.2	12.4	23	291

A combination of categorical and continuous features makes this dataset suitable for mixed-data modeling approaches.

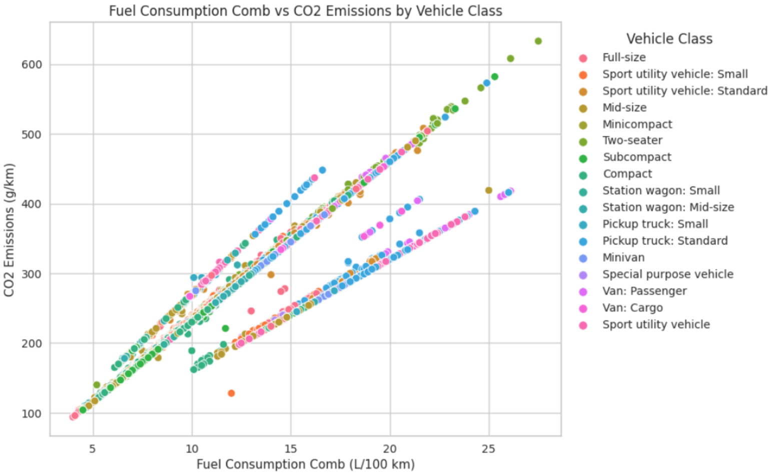
3.3 Exploratory Data Analysis (EDA)

Comprehensive EDA provided a foundational understanding of feature distributions, relationships, and potential modeling challenges.

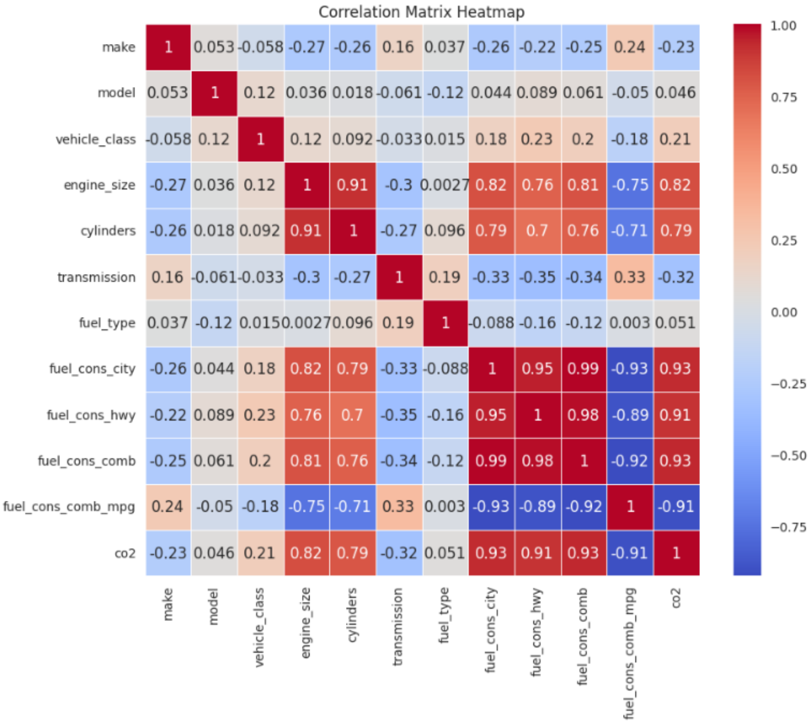
- **Univariate Distributions:**
  - Histograms and kernel density estimates for continuous variables (engine size, cylinders, fuel consumption, CO_2 emissions).
  - Frequency plots for categorical variables (fuel type, transmission, vehicle class).



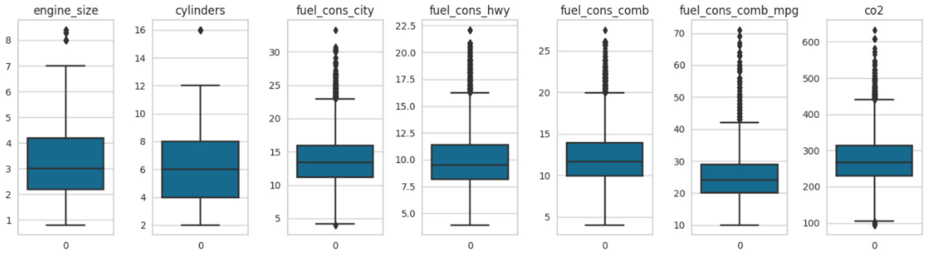
- **Bivariate Relationships:**
  - Scatterplots illustrating linear and non-linear relationships (e.g., combined fuel consumption vs. CO_2).
  - Boxplots to compare emission distributions across categorical groups.



- **Correlation Analysis:** Pearson correlation heatmap among numerical features revealed strong associations, especially between combined fuel consumption and CO_2 emissions (r > 0.85).



- **Outlier Detection:** Interquartile range (IQR) method flagged extreme emission values; outliers were retained to preserve model generality but downweighted by robust scaling.



Findings from EDA guided feature engineering and informed model choice.

3.4 Data Preprocessing and Feature Engineering

A consistent preprocessing workflow was implemented via scikit-learn's Pipeline and ColumnTransformer.

- **Missing Value Imputation:** Median substitution for numerical features; new category ”Unknown” for missing categorical entries.
- **Categorical Encoding:** One-Hot Encoding for nominal features.

- **Feature Scaling:** StandardScaler applied to all numerical variables to standardize means and variances.

- **Target Normalization:** PowerTransformer(method='yeo-johnson') transformed the CO_2 emission distribution towards Gaussianity.

All transformers were fitted exclusively on training data to avoid information leakage.

3.5 Exploratory Modeling Approaches

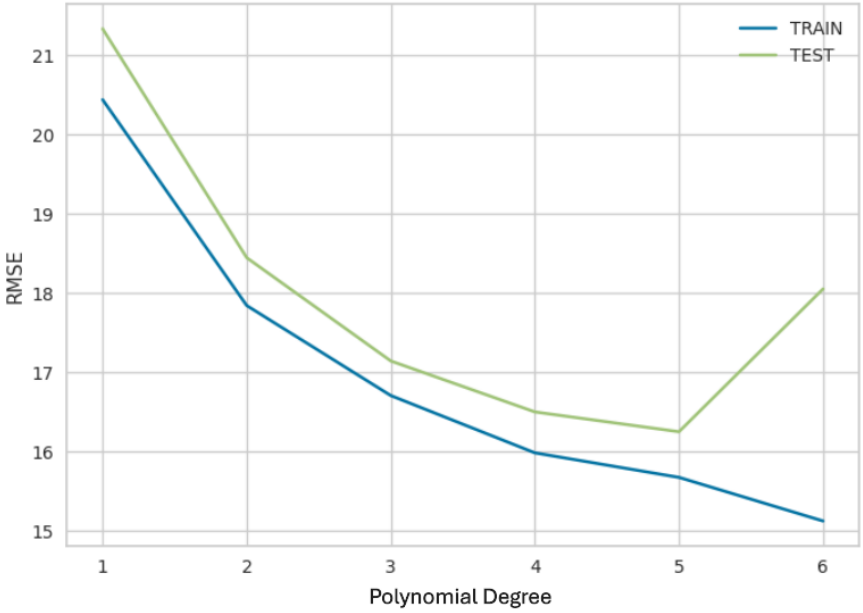
To build intuition and benchmark predictive capability, a spectrum of algorithms was trained on the preprocessed dataset. All models used the same training–testing split (80/20) and performance metrics (R^2, MAE, MSE, RMSE).

3.5.1 Linear Regression Variants

- **Simple Linear Regression:** Single-predictor model (engine size vs. CO_2). *Test performance:* $R^2 = 0.6785$, MAE = 28.82 g/km, RMSE = 37.58 g/km. This baseline demonstrates a modest fit, indicating that engine size alone explains only about 68% of the variance in emissions.

- **Multiple Linear Regression:** Incorporates all continuous features (engine size, cylinders, fuel consumption metrics). *Test performance:* $R^2 = 0.8965$, MAE = 11.86 g/km, RMSE = 21.20 g/km. The inclusion of multiple predictors substantially improves accuracy, capturing nearly 90% of the variance.

- **Polynomial Regression (Degree 4):** Extends multiple regression with polynomial terms up to fourth order to model non-linear relationships. *Test performance:* $R^2 = 0.9355$, MAE = 7.50 g/km, RMSE = 16.83 g/km. The higher-degree terms effectively capture curvature in the data, further reducing prediction error.



3.5.2 Regularized Linear Models

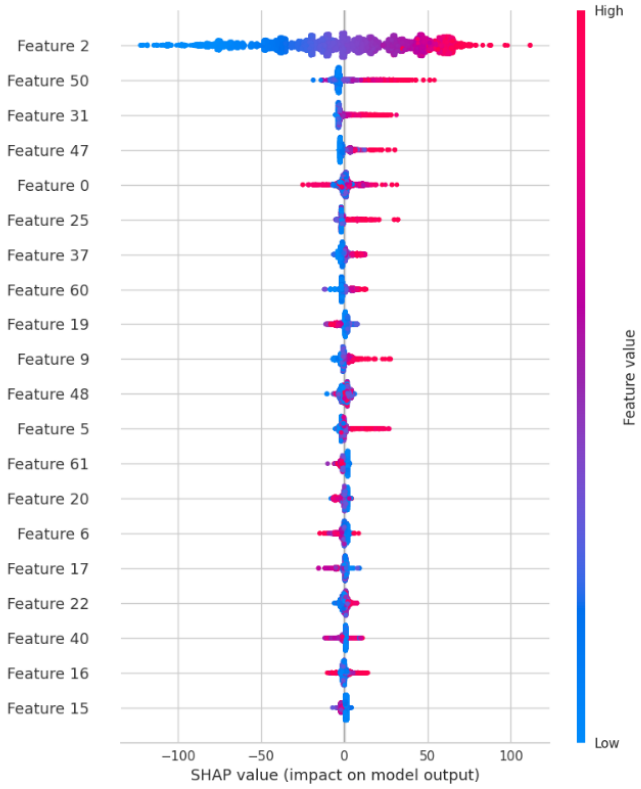
- **Ridge Regression (L2):** Adds an L2 penalty to mitigate multicollinearity among predictors. *Test performance:* $R^2 = 0.9282$, MAE = 8.44 g/km, RMSE = 17.76 g/km. Regularization yields a smoother solution, with only a slight decrease compared to the polynomial model, but improved stability.

- **Lasso Regression (L1):** Utilizes an L1 penalty to enforce sparsity and perform implicit feature selection. *Test performance:* $R^2 = 0.9030$, MAE = 11.87 g/km, RMSE = 20.64 g/km. The model simplifies the predictor set at the cost of some predictive power.

3.5.3 Tree-Based and Ensemble Methods

- **Random Forest Regressor:** Aggregates multiple decision trees to capture complex feature interactions. *Test performance:* $R^2 = 0.9718$, MAE = 3.71 g/km, RMSE = 11.13 g/km. Demonstrates strong non-linear modeling capability and robustness to outliers.

- **XGBoost Regressor:** Implements gradient boosting with regularization and tree pruning. *Test performance:* $R^2 = 0.9707$, MAE = 3.91 g/km, RMSE = 11.35 g/km. Slightly lower variance explained than Random Forest, but comparable error metrics.



- **LightGBM Regressor:** Utilizes histogram-based boosting for faster training on large feature spaces. *Test performance:* $R^2 = 0.9715$, MAE = 4.04 g/km, RMSE = 11.20 g/km. Matches XGBoost in accuracy with improved computational efficiency.

- **CatBoost Regressor:** Provides native handling of categorical features and ordered boosting. *Test performance:* $R^2 = 0.9714$, MAE = 4.04 g/km, RMSE = 11.22 g/km. Comparable to other boosting methods, with minimal preprocessing for categorical data.

Synthesis: Insights from these experiments—particularly feature importance rankings and error patterns—guided the construction of the final ensemble pipeline.

3.6 Final Ensemble Pipeline

The ensemble strategy developed in this study embodies a multi-stage learning system carefully designed to leverage the strengths of different gradient boosting models, enhance their predictive capabilities through a meta-learner, and correct residual errors with a final adjustment model. This section provides an expanded discussion on the design philosophy, training workflow, and rationale behind each module of the final pipeline.

3.6.1 Pipeline Architecture

The complete ensemble pipeline consists of three fundamental layers:

- **Preprocessing Layer:** Standardizes and encodes the input data to ensure model compatibility and improve learning stability.
- **Stacked Base Learners:** Parallel training of three highly efficient gradient boosting algorithms to generate diverse yet complementary predictions.
- **Meta-Learning and Residual Correction Layer:** Aggregates base learner outputs via a neural network meta-learner and subsequently applies Ridge regression to model any remaining systematic errors.

The modularity of the pipeline ensures scalability, interpretability, and minimal information leakage across the workflow.

3.6.2 Detailed Description of Components

- **Data Preprocessing**
  - **Numerical Features:** Standardization using StandardScaler to ensure zero mean and unit variance.
  - **Categorical Features:** Transformation using OneHotEncoder, preserving unknown categories to manage unseen inputs gracefully.
  - **Target Variable:** Normalized using PowerTransformer (Yeo–Johnson method) to mitigate skewness and heteroscedasticity, facilitating better convergence during training. All transformations were encapsulated in a ColumnTransformer and fitted exclusively on the training dataset.
- **Base Learners** The first layer comprises three diverse and complementary gradient boosting models:
  - **LightGBM Regressor:**

    - Fast training using histogram-based algorithms.
    - Leaf-wise tree growth enhances model complexity management.
  - **XGBoost Regressor:**
    - Regularized boosting prevents overfitting.
    - Sophisticated tree-pruning and parallelized learning mechanisms.
  - **CatBoost Regressor:**
    - Native support for categorical variables without explicit encoding.
    - Utilizes ordered boosting for reduced prediction bias.

  These learners were independently optimized for their hyperparameters through Bayesian optimization, ensuring optimal balance between bias and variance.

- **Meta-Learner** A shallow Multi-Layer Perceptron (MLP) Regressor was employed as the meta-learner:
  - **Architecture:** Single hidden layer with tuned neuron counts.
  - **Activation:** ReLU for hidden layers; Linear activation for output layer.
  - **Optimizer:** Adaptive gradient descent (Adam) with a tuned learning rate.
  - **Loss Function:** Used Mean Squared Error(MSE) for loss function.

  The meta-learner synthesized the out-of-fold predictions from the base learners to minimize the generalization error.

- **Residual Correction Model** After stacking, residuals (errors between predicted and actual normalized targets) were modeled using a Ridge Regression:
  - **Objective:** Capture and correct systematic biases missed by the ensemble.
  - **Regularization:** L2 penalty to control model complexity and prevent overfitting.

  This final adjustment ensured that even subtle residual structures were accounted for, maximizing final predictive precision.

3.6.3 Hyperparameter Optimization

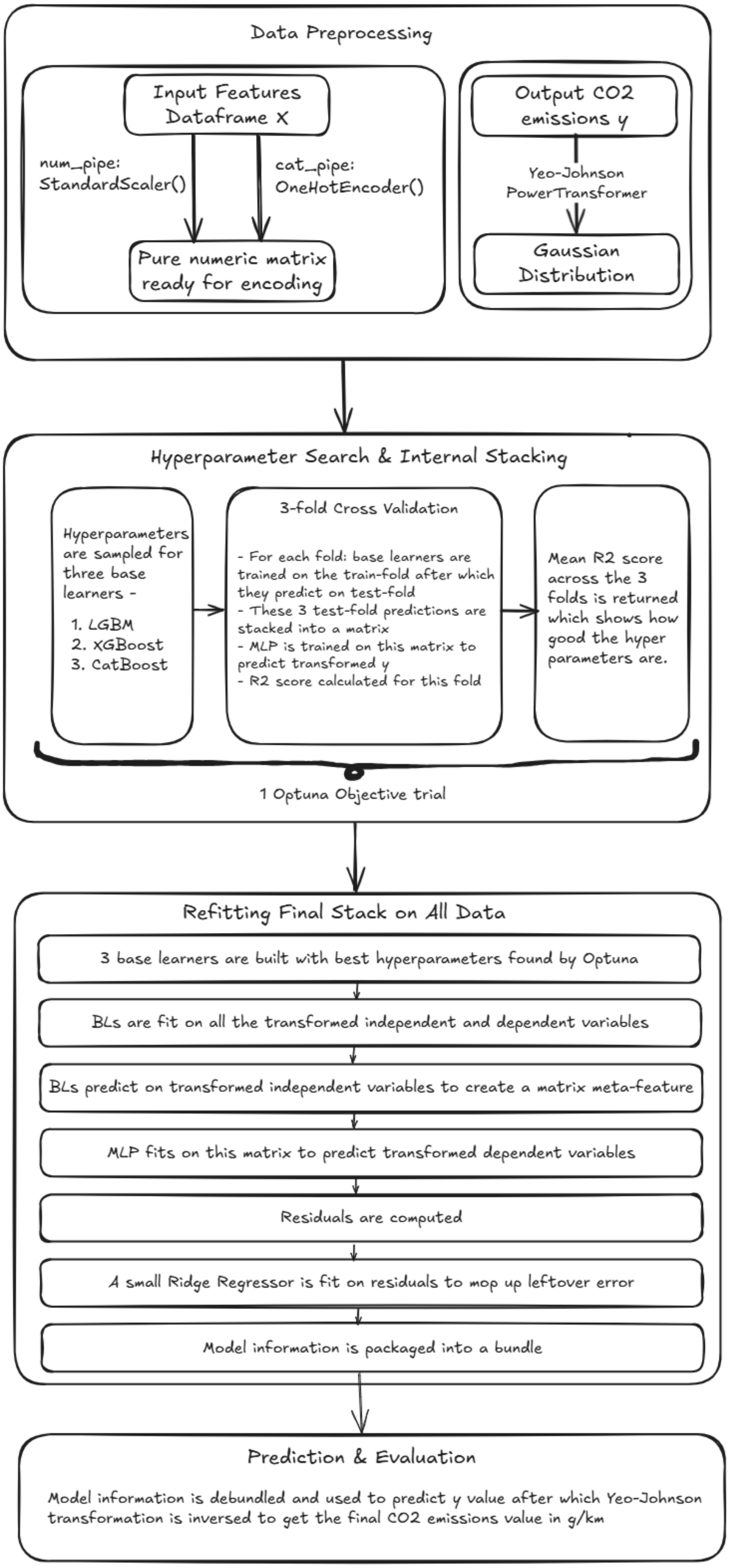
Bayesian optimization with Optuna maximizes mean cross-validated R^2. The search space includes:

- Number of trees, learning rate, tree depth for each base learner
- Hidden layer size, regularization coefficient, and learning rate for the MLP

Optimization runs employ 3-fold cross-validation to balance computational cost and robustness.

3.6.4 Training and Prediction Workflow

- Fit preprocessor and target transformer on training data.
- Optimize base learners and MLP via Optuna's cross-validated trials.
- Retrain tuned models on complete training data.
- Generate stacked features and fit meta-learner.
- Train Ridge regressor on meta-learner residuals.
- Predict by applying transforms to new data, obtaining base predictions, meta combination, residual adjustment, and inverse-transform of targets.



Data Preprocessing

Input Features Dataframe X
num_pipe: StandardScaler()
cat_pipe: OneHotEncoder()
Pure numeric matrix ready for encoding
Output CO2 emissions y
Yeo-Johnson PowerTransformer
Gaussian Distribution

Hyperparameter Search & Internal Stacking

Hyperparameters are sampled for three base learners -

1. LGBM
2. XGBoost
3. CatBoost

3-fold Cross Validation

- For each fold: base learners are trained on the train-fold after which they predict on test-fold
- These 3 test-fold predictions are stacked into a matrix
- MLP is trained on this matrix to predict transformed y
- R2 score calculated for this fold

Mean R2 score across the 3 folds is returned which shows how good the hyper parameters are.

1 Optuna Objective trial

Refitting Final Stack on All Data

3 base learners are built with best hyperparameters found by Optuna

BLs are fit on all the transformed independent and dependent variables

BLs predict on transformed independent variables to create a matrix meta-feature

MLP fits on this matrix to predict transformed dependent variables

Residuals are computed

A small Ridge Regressor is fit on residuals to mop up leftover error

Model information is packaged into a bundle

Prediction & Evaluation

Model information is debundled and used to predict y value after which Yeo-Johnson transformation is inversed to get the final CO2 emissions value in g/km

3.7 Model Evaluation Metrics

The evaluation of the predictive models has been performed using a comprehensive set of statistical metrics designed to measure different aspects of model performance and generalization capability.

3.7.1 Primary Evaluation Metrics

- **Coefficient of Determination (R^2 Score)**:
The R^2 statistic measures the proportion of variance in the observed target variable that is explained by the model. Formally,

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{1}$$

where y_i are the true emission values, $\hat{y}_i$ are the model predictions, and $\bar{y}$ is the mean of the observed values.
An R^2 of 1 shows a perfect prediction; on the other hand, a value of 0 implies that the model will perform similar to predicting the mean. Negative R^2 values can occur when the model fits worse than the horizontal line at $\bar{y}$. In this study, high R^2 values (above 0.90) signify strong explanatory power of our ensemble pipeline.

- **Mean Absolute Error (MAE)**:
The MAE quantifies the absolute average deviation between actual and predicted values:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|. \tag{2}$$

By treating all errors equally, MAE provides an interpretable which is measured in the same units as the target, CO_2 emissions, (g/km). A lower MAE reflects more accurate and reliable predictions, with reduced average bias.

- **Mean Squared Error (MSE)**:
The MSE is used to calculate the average of squared deviations, thereby penalizing larger errors more severely:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{3}$$

Since outliers contribute quadratically, MSE is sensitive to occasional large mispredictions. It is widely used in optimization procedures due to its smooth, differentiable properties.

- **Root Mean Squared Error (RMSE)**:
The RMSE is defined as the square root of the MSE, translating the penalized error back into the original measurement scale:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \tag{4}$$

RMSE helps in retaining the sensitivity of MSE to large errors but yields values that are directly comparable to the original units of CO_2 emissions. A smaller RMSE indicates that large deviations are minimal, reflecting high model precision.

- **Metric Selection Rationale**:
  Together, these metrics provide a comprehensive evaluation:
  - R^2 assesses overall variance explained.
  - MAE measures average model bias in real units.
  - MSE and RMSE emphasize the impact of large errors, guiding optimization towards both accuracy and robustness.

  Reporting all four metrics ensures that model performance is evaluated from both statistical and practical perspectives, aligning with best practices in environmental data science.

3.7.2 Validation Strategy

To ensure that the reported performance reflects genuine predictive capability rather than overfitting to a specific data split, a two-pronged validation scheme was adopted:

- **Hold-out Test Set**: A randomly selected 20% subset of the full dataset was withheld before any model training or hyperparameter tuning. This "hold-out" set remained untouched during all optimization phases and was used exclusively for the final evaluation of each model. By evaluating data that the model has never seen, this approach provides an unbiased estimate of real-world performance.
- **3-Fold Cross-Validation**: The other 80% of data that remained was subjected to 3-fold cross-validation during model development:
  1. The training portion was partitioned into three approximately equal folds.
  2. In each iteration, two folds were used to fit the model, and the third fold served as the validation set.
  3. Performance metrics (e.g., R^2, MAE, RMSE) were computed on the validation fold.
  4. The process was repeated such that each fold acted once as the validation set.

  The cross-validation scores were then averaged to yield a robust estimate of model generalization and to guide hyperparameter selection.

Stability Assessment: In addition to mean performance, the standard deviation of cross-validation scores was recorded. A low standard deviation will indicate that the model's performance will be consistent across different data partitions, suggesting reliable behavior under varying data samples.

Overall, this combined strategy of hold-out testing and structured cross-validation balances the need for an unbiased final evaluation with comprehensive use of available data during model development.

3.7.3 Interpretation Guidelines

Metric	**Interpretation**
High R^2	Strong explanatory power; high variance captured
Low MAE	Small average prediction error
Low MSE and RMSE	Accurate predictions; penalization of large errors
Low Cross-Validation Std Dev	Stability across different data splits

Table 1: Interpretation of evaluation metrics

3.8 Summary

This chapter has delineated the evolution from initial individual models to an advanced ensemble pipeline, emphasizing rigorous preprocessing, thorough EDA, and sophisticated optimization techniques. The final architecture is designed to maximize predictive accuracy and ensure robust generalization for CO_2 emission forecasting.