# Portfolio: Named Entity Recognition

Olusanmi Hundogan

A report submitted as part of the requirements for the toptal assessment

August 2025

Implemented for OHCM B.V.

# Abstract

This report presents a named entity recognition (NER) system designed to extract **Persons**, **Organizations**, and **Locations** from a dataset of news articles. The task required building a performant and reproducible NLP solution that outperforms out-of-the-box models, deploys as a REST API, and scales under concurrent usage.

The workflow began with exploratory data analysis to understand entity distributions and text structure. Preprocessing included standard NLP techniques such as sentence segmentation, tokenization, and custom annotation handling for multi-word entities. Multiple baselines were tested, including pretrained transformer models (e.g., BERT-based token classifiers) and rule-based heuristics.

The final solution employed a fine-tuned transformer model on a custom-labeled dataset, trained to optimize exact-match entity extraction, especially for multi-token names and locations. Organizations were evaluated with dual strategies: full-string matches and token-wise comparisons. Experiment tracking ensured reproducibility and allowed comparative evaluations.

A RESTful API was developed to accept CSV files as input, process the text column, and return extracted entities in the required CSV format. The service was containerized using Docker, enabling easy deployment. To validate performance under load, the system was tested with `locust`, demonstrating stability under 20 concurrent users with acceptable latency.

Overall, this report outlines a robust pipeline for developing, deploying, and evaluating a custom NER system under real-world constraints, showcasing end-to-end applied machine learning capabilities.

# Contents

**Abstract** ii

**1 Introduction** 1

**2 Data Exploration** 3
- 2.1 General Description . . . 3
- 2.2 Feature Composition . . . 4
- 2.3 Target Scopes . . . 4
- 2.4 Features . . . 7
  - 2.4.1 Feature Types and Completeness . . . 7
  - 2.4.2 Missingness and Sparsity . . . 7
  - 2.4.3 Distributional Characteristics . . . 8

**3 Data Preparation** 10

**4 Software Architecture Design** 12
- 4.1 Overview . . . 12
- 4.2 Pipeline Components . . . 12
- 4.3 Training Lifecycle . . . 13
- 4.4 Design Considerations . . . 14
- 4.5 Architecture Diagram . . . 14

**5 Model Selection** 15
- 5.1 EvenWeightMiniModelArmyEstimator . . . 16
- 5.2 Final Pipeline Configuration . . . 16

**6 Evaluation** 18
- 6.1 Experimental Design . . . 18
- 6.2 Results and Insights . . . 19
- 6.3 Final Decision . . . 19

6.4 Final Model Performance . . . . . . . . . . . . . . . . . . . . . . . . . . 19

**7 Deployment and Scalability 22**

**8 Conclusions 23**

**A Architecture Diagram 24**

# List of Tables

2.1 Number of features by type. . . . . . 4
2.2 Feature types and completeness. . . . . . 8

3.1 Number of features retained at different VIF thresholds. Percentages are relative to the 102 numerical features in the dataset. . . . . . 11

6.1 Final model performance on the held-out test set. . . . . . 19
6.2 Distribution of absolute percentage errors for the final model, by scope. 20

# List of Figures

2.1 Distribution of Scope 1 emissions before (left) and after (right) log transformation, by reporting year. . . . . . . . . . . . . . . . . . . . . . . . . . 6
2.2 Correlation matrix of Scope 1, Scope 2, and Scope 3 emissions after log transformation, showing moderate inter-scope relationships. . . . . . . . 7
2.3 Missingness matrix for numerical and categorical features. . . . . . . . . 8
2.4 Example distributions of selected numerical features after `arcsinh` transformation. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

4.1 Simplified high-level summary of pipeline and meta-model orchestration. Full diagram is in Appendix A. . . . . . . . . . . . . . . . . . . . . . . . 14

5.1 High-level structure of the `EvenWeightMiniModelArmyEstimator`, showing bucket classification feeding into dedicated ensemble regressors for each emission size segment. . . . . . . . . . . . . . . . . . . . . . . . . . . 17

6.1 Comparison of all tested configurations in the final evaluation experiment, showing median sMAPE and variance over 10 repetitions. . . . . 21

A.1 Complete UML architecture diagram showing all classes, methods, and relationships. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

# List of Algorithms

# Listings

# Chapter 1

# Introduction

Accurately quantifying greenhouse gas (GHG) emissions is essential for companies aiming to meet sustainability targets, comply with regulations, and report transparently to stakeholders. The Greenhouse Gas Protocol classifies emissions into three categories:

- **Scope 1**: Direct emissions from owned or controlled sources.
- **Scope 2**: Indirect emissions from the generation of purchased energy.
- **Scope 3**: All other indirect emissions occurring across the value chain, including upstream and downstream activities.

Despite the growing pressure for disclosure, Scope 1, 2, and particularly Scope 3 data remains sparse and inconsistent across public sources. To address this, we compiled multiple heterogeneous data sources—including financial statements, sustainability reports, and emissions registries—into a unified dataset.

We developed a configurable machine learning framework to predict Scope 1, 2, and 3 $CO_2$ emissions using structured company-level features. The system was designed to support experimentation with different models, feature sets, and preprocessing configurations. Through systematic experimentation and evaluation, we identified the best-performing configurations across scopes.

The resulting model was then used to enrich our original dataset, generating predictive emissions estimates for companies lacking full disclosure—especially for Scope 3.

The remainder of this report is organized as follows:

- **Chapter 2** – Data sources and initial insights.
- **Chapter 3** – Preprocessing and feature engineering pipeline.

- **Chapter 4** – Modular framework design.
- **Chapter 5** – Candidate models and tuning process.
- **Chapter 6** – Evaluation metrics and results.
- **Chapter 7** – Deployment and scalability considerations.
- **Chapter 8** – Final remarks and future work.

# Chapter 2

# Data Exploration

## 2.1 General Description

The dataset combines multiple public and proprietary sources into a unified table that links company identifiers, reported or estimated CO2 emissions, and a broad range of financial and operational indicators. Rather than being a flat collection of numbers, the data follows a consistent naming convention in which prefixes indicate the nature and type of each field.

Columns beginning with `key_` serve as the fundamental identifiers for each record, capturing essential context such as the reporting year, the company's country code and the stock ticker symbol. These identifiers are crucial for aligning observations across different data sources, ensuring temporal consistency, and enabling integration with additional datasets.

The `tg_` prefix is reserved for the target variables of the regression task, namely the CO2 emissions for Scope 1, Scope 2, and Scope 3. These variables are typically continuous numerical values, denoted by the `numc` suffix, and they form the basis for model training and evaluation. Due to uneven disclosure practices, the coverage of these targets varies considerably, with Scope 3 data being particularly sparse.

Features describing the financial and operational state of each company are marked with the `ft_` prefix. Continuous numeric features (`numc`) include measures such as revenue, total assets, liabilities, and various cash flow components, offering quantitative signals about a company's size and performance. Discrete numeric features (`numd`) capture counts or integer values such as the number of shares outstanding. Categorical multinomial features (`catm`) encode descriptors like industry classification, sector membership, stock exchange, and geographic region, providing structural context that may relate to emissions patterns.

## 2.2 Feature Composition

The dataset contains a rich mixture of numeric and categorical variables, reflecting both quantitative performance indicators and qualitative descriptors. Table 2.1 summarizes the number of features by type.

| **Feature Type** | **Count** |
|---|---|
| Numeric continuous (`numc`) | 78 |
| Numeric discrete (`numd`) | 3 |
| Categorical multinomial (`catm`) | 4 |

Table 2.1: Number of features by type.

The majority of numeric continuous features originate from financial statements, including the balance sheet, income statement, and cash flow statement. These encompass measures such as revenues, gross profit, assets, liabilities, capital expenditures, and valuation ratios. The discrete numeric features represent integer counts such as shares outstanding or float shares.

Categorical features include:

- `ft_catm_country_code` – ISO country code of the company.
- `ft_catm_sector_name` – Sector classification.
- `ft_catm_industry_name` – Industry classification.
- `ft_catm_exchange` – Stock exchange identifier.

This composition reflects a combination of financial statement data, market data, and corporate metadata. Together, they offer both direct and indirect signals for predicting emissions but also introduce challenges in preprocessing due to heterogeneous types, different scales, and varying levels of missingness.

## 2.3 Target Scopes

The prediction targets in this study are Scope 1, Scope 2, and Scope 3 CO2 emissions, reported in metric tonnes of CO2 equivalent. These are represented in the dataset as continuous numerical variables (`tg_numc_scope_1`, `tg_numc_scope_2`, `tg_numc_scope_3`) and form the regression objectives for model training and evaluation.

Exploratory analysis reveals that the data for all three scopes is extremely sparse and exhibits heavy positive skewness. A substantial share of companies report no data at all, particularly for Scope 3, while among those that do report, most values are relatively small compared to a small number of extremely large emitters. For example, in Scope 1, the majority of records fall into the "Emittor" category with moderate emissions, but there is a long tail extending several orders of magnitude. Zero and near-zero values are also present, along with a small number of anomalous entries.

The skewness is not just a statistical artifact—it has important implications for model performance. Without adjustment, regression models may be dominated by the largest emitters, reducing predictive accuracy for the majority of companies. Addressing this imbalance may require log transformations, stratified training and evaluation, or the use of robust error metrics that reduce the influence of outliers.

Log transformation was applied to the scope values primarily to stabilise variance, reduce the influence of extreme emitters, and make the underlying distribution of typical companies more visible to the model. This step also helps to satisfy assumptions of many regression algorithms regarding the distribution of residuals. Although correlations between the scopes decrease after logging, the transformation improves the ability to model small and medium emitters without being overwhelmed by the extremes.

Figure 2.1 demonstrates this distributional challenge for Scope 1 emissions, showing the raw data distribution on the left and the log-transformed distribution on the right. The transformation compresses extreme values and reveals more structure in the central mass of the data. A similar effect holds for Scope 2 and Scope 3.

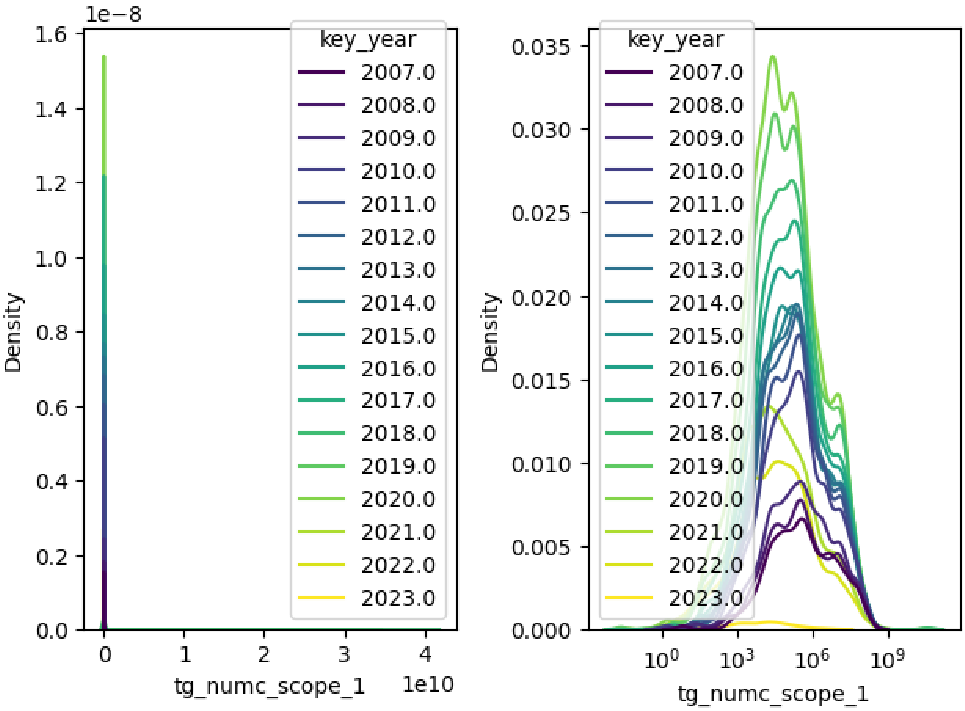

Figure 2.1: Distribution of Scope 1 emissions before (left) and after (right) log transformation, by reporting year.

After log transformation, correlations between the scopes remain positive but are moderate in strength. As shown in Figure 2.2, the Pearson correlations range from 0.56 to 0.68, indicating that while companies' emissions profiles across Scope 1, Scope 2, and Scope 3 share some relationship, each scope retains distinct variability. This suggests that independent modelling for each scope, possibly with shared features, may capture additional predictive power compared to assuming strong interdependence.

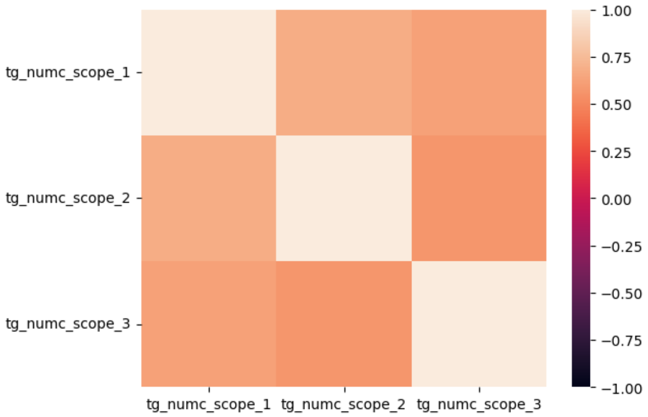

Figure 2.2: Correlation matrix of Scope 1, Scope 2, and Scope 3 emissions after log transformation, showing moderate inter-scope relationships.

In summary, the target variables present a dual challenge of sparsity and skewness, both of which must be addressed to produce stable and generalizable predictions. Log transformation plays a central role in meeting this challenge, even if it reduces inter-scope correlation.

## 2.4 Features

The feature set for predicting company CO2 emissions combines multiple data sources, primarily derived from publicly available financial statements, augmented with categorical descriptors and regional metadata. In total, the dataset contains 129 features after preprocessing: 102 numerical variables (`ft_numc_*`) and 4 categorical variables (`ft_catm_country_code`, `ft_catm_sector_name`, `ft_catm_industry_name`, `ft_catm_exchange`), alongside historical scope data and other derived indicators.

### 2.4.1 Feature Types and Completeness

Table 2.2 summarises the feature composition and overall completeness. While categorical variables are mostly complete, numerical variables exhibit a substantial degree of missingness, reflecting heterogeneity in reporting standards, sector-specific disclosures, and regional accounting practices.

### 2.4.2 Missingness and Sparsity

Exploratory analysis shows substantial missingness across many numerical features. As visualised in Figure 2.3, a considerable portion of financial indicators are sparsely reported across companies and years. This sparsity reflects heterogeneity in financial

Table 2.2: Feature types and completeness.

| **Feature Type** | **Count** | **Mean Missingness (%)** | **Notes** |
|---|---|---|---|
| Numerical (`ft_numc`) | 102 | 45.2 | Financial metrics, ratios, balance sheet items |
| Categorical (`ft_catm`) | 4 | 3.8 | Country, sector, industry, exchange |
| Historical scope values | 3 | 61.5 | Previous-year Scope 1–3 targets |

reporting requirements, industry practices, and disclosure levels, and must be considered when designing imputation or feature selection strategies. The categorical features are generally more complete, though some gaps exist, particularly in country codes for certain entities.

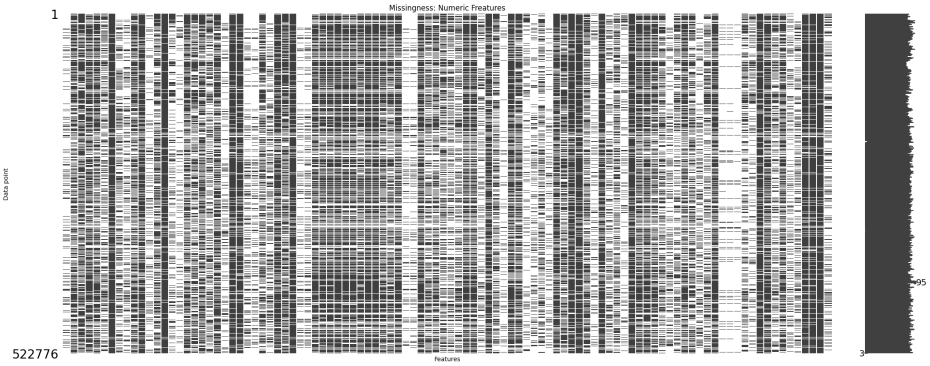

Figure 2.3: Missingness matrix for numerical and categorical features.

### 2.4.3 Distributional Characteristics

Many numerical features display extreme skewness and kurtosis, often spanning several orders of magnitude. For example, `ft_numc_total_assets` exhibits a skewness of over 260, driven by a small number of very large companies relative to the majority of smaller entities. This distributional imbalance is a direct consequence of the wide diversity of company sizes and sectors included in the dataset.

To address these challenges, transformations such as the inverse hyperbolic sine (`arcsinh`) were considered. Unlike a pure logarithmic transformation, `arcsinh` accommodates zero and negative values while still compressing extreme outliers, making it suitable for financial ratio features where sign reversals or near-zero entries occur.

Figure 2.4 illustrates the wide range and skewness of selected features, highlighting the necessity of scaling or transformation prior to model training.

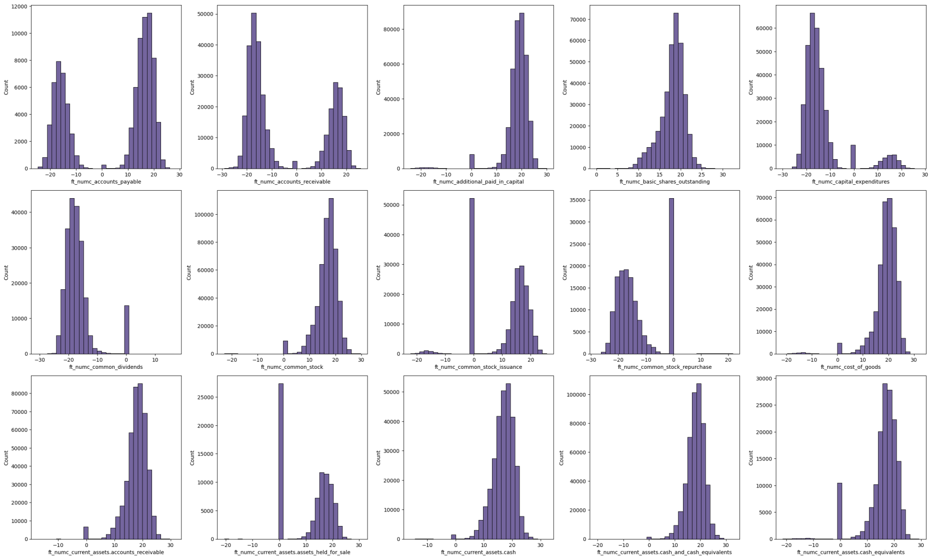

Figure 2.4: Example distributions of selected numerical features after `arcsinh` transformation.

# Chapter 3

# Data Preparation

The primary goal of the data preparation stage was to produce a set of explanatory variables that are both statistically sound and computationally efficient for model training. Given the large number of potentially collinear financial and categorical variables, multicollinearity was assessed using the Variance Inflation Factor (VIF).

VIF quantifies the degree to which a given feature is explained by the other features in the dataset. High VIF values indicate strong collinearity, which can inflate variance in regression estimates and reduce model interpretability. This is particularly relevant for financial statement data, where many indicators (e.g., total assets, equity, liabilities) are inherently correlated through accounting relationships. Without mitigation, such redundancy can lead to unstable model coefficients, slower training convergence, and inflated importance scores for redundant features.

To address this, feature subsets were generated for a range of VIF thresholds: 5, 10, 15, and 20. Lower thresholds yield smaller, more independent feature sets, whereas higher thresholds retain more features at the cost of increased collinearity.

Table 3.1: Number of features retained at different VIF thresholds. Percentages are relative to the 102 numerical features in the dataset.

| **VIF Threshold** | **Retained Features** | **Reduction from Original (%)** | **Notes** |
|---|---|---|---|
| 5 | 25 | 75.5 | Strictest set, minimal collinearity |
| 10 | 35 | 66.2 | Balanced removal of redundancy |
| 15 | 41 | 61.3 | Retains moderate redundancy |
| 20 | 46 | 57.1 | Higher tolerance for correlation |

The outcome of this process was a set of alternative feature configurations for experimentation during the model selection phase, which are further discussed in Chapter 5.

# Chapter 4

# Software Architecture Design

## 4.1 Overview

The modelling framework is designed to train and evaluate independent pipelines for each of the three target scopes, wrapped into a single meta-model for unified evaluation and inference. The core building block is the `DefaultPipeline`, which composes six primary components: preprocessing, feature reduction, imputation, scope estimation, confidence estimation, and target transformation. A dedicated `OxariMetaModel` class acts as a container for these scope-specific pipelines, providing aggregated evaluation and reporting.

## 4.2 Pipeline Components

Each pipeline instance is composed of modular components, each responsible for a specific stage in the modelling workflow:

1. **Preprocessor**: Handles numeric scaling, transformation of features, and general data cleaning. The specific transformation (e.g., inverse hyperbolic sine, logarithmic, standard scaling) can be configured depending on the feature distribution.
2. **Feature Reducer**: Reduces the dimensionality of the input feature space to improve efficiency and mitigate multicollinearity. This may use statistical filters, principal component analysis (PCA), or other advanced selection techniques.
3. **Imputer**: Fills in missing feature values using simple heuristics, statistical measures, or model-based imputation. Part of the preprocessing step.
4. **Scope Estimator**: The primary predictive model for a given scope, optimised

through a hyperparameter search strategy (e.g., Optuna). This component adheres to a common estimator interface, enabling the use of different model families.

5. **Confidence Estimator**: Calibrates prediction intervals or uncertainty estimates after the main model is trained, providing additional interpretability and reliability.
6. **Target Transformer**: Applies a transformation to the prediction targets during training (e.g., log scaling) and reverts them to the original scale during inference. Part of the preprocessing step.

The architecture is designed so that each component can be substituted with alternative implementations without modifying the overall pipeline structure. For example, a logarithmic target scaler could be replaced by a dummy (no-op) scaler, or a VIF-based feature reducer could be swapped for a principal component analysis (PCA) module.

## 4.3 Training Lifecycle

For each scope (1, 2, and 3), the pipeline undergoes the following sequence:

1. **Hyperparameter Optimisation**: Using `pipeline.optimise(...)` on the training split to select the best configuration.
2. **Model Fitting**: Calling `pipeline.fit(...)` to train all components of the pipeline.
3. **Evaluation**: Running `pipeline.evaluate(...)` on reserved validation and test data.
4. **Confidence Calibration**: Using `pipeline.fit_confidence(...)` to fit the confidence estimator.

The resulting pipelines are registered in the `OxariMetaModel`:

```
model.add_pipeline(scope=1, pipeline=dp1)
model.add_pipeline(scope=2, pipeline=dp2)
model.add_pipeline(scope=3, pipeline=dp3)
```

Finally, the meta-model is evaluated on a generic train-test split to produce aggregated performance metrics across scopes.

## 4.4 Design Considerations

This architecture enforces a consistent, modular interface across all stages of the pipeline, enabling:

- **Swap-ability**: Any component can be replaced without modifying the pipeline orchestration logic.
- **Reproducible Optimisation**: Hyperparameter search is encapsulated, separating scoring logic from model code.
- **Scope Parity**: Identical structures per scope simplify operational deployment and performance comparison.

## 4.5 Architecture Diagram

Figure 4.1 illustrates the relationships between the pipeline components and their orchestration within the meta-model.

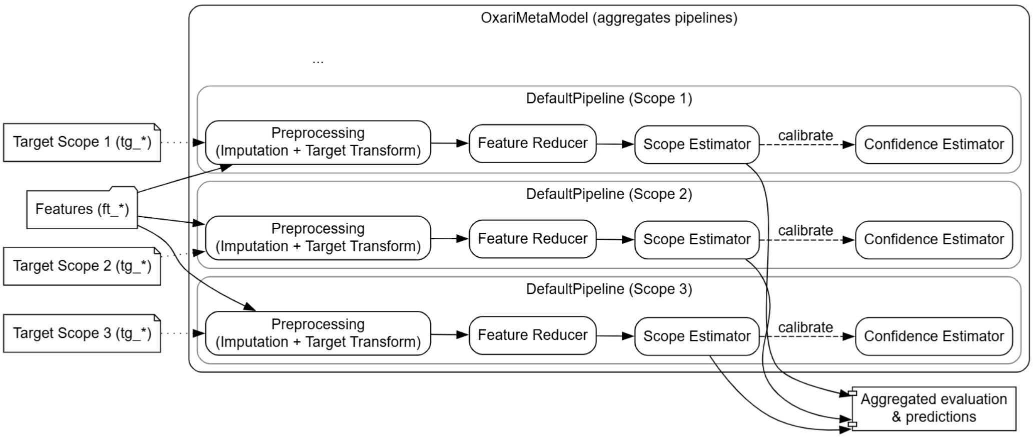
OxariMetaModel (aggregates pipelines)
...
DefaultPipeline (Scope 1)
DefaultPipeline (Scope 2)
DefaultPipeline (Scope 3)
Target Scope 1 (tg_*)
Features (ft_*)
Target Scope 2 (tg_*)
Target Scope 3 (tg_*)
Preprocessing (Imputation + Target Transform)
Feature Reducer
Scope Estimator
calibrate
Confidence Estimator
Aggregated evaluation & predictions

Figure 4.1: Simplified high-level summary of pipeline and meta-model orchestration. Full diagram is in Appendix A.

# Chapter 5

# Model Selection

To determine the best-performing configuration for predicting Scope 1, Scope 2, and Scope 3 CO2 emissions, we conducted a series of controlled experiments. All experimental code, along with detailed results and conclusions, is available in the public repository[1].

The experiments explored variations in preprocessing, feature selection, imputation strategies, estimators, and target transformations. Hyperparameter tuning was performed using `Optuna`, with a fixed number of trials per run. The final implementation, defined in the `train_model_for_imputation` function, trains three independent pipelines—one per scope—which are then combined into a single `OxariMetaModel` for unified evaluation and inference.

Through iterative testing, several patterns emerged. Normalizing features and applying the inverse hyperbolic sine (`arcsinh`) transformation consistently improved performance by mitigating skewness while preserving zero values. We found that no additional dimensionality reduction (e.g., PCA) was necessary, as prior VIF-based filtering had already removed highly collinear features and retaining the remaining set yielded better accuracy. For handling missing values, replacing them with random draws from observed distributions outperformed predictive imputations, which risked introducing systematic bias. Finally, applying a logarithmic transformation to the targets proved essential given their heavy positive skewness, improving fit for the majority of companies without overly compressing large values.

---

[1] https://github.com/Olu93/portfolio_carbon_emission_modelling/tree/development/notebooks/reports

## 5.1 EvenWeightMiniModelArmyEstimator

The estimator used in the final configuration is the `EvenWeightMiniModelArmyEstimator`, a segmented ensemble-based meta-learner designed for heterogeneous target distributions. The process follows two main stages:

1. **Bucket Classification:** Using the same features intended for regression, a LightGBM classifier predicts the emission size segment (bucket) into which a company is most likely to fall. Buckets are derived from emission quantiles or predefined thresholds.
2. **Segment-Specific Regression:** Each bucket is assigned an independent `VotingRegressor` composed of multiple diverse base learners (e.g., linear models, LightGBM regressors, clustering-based regressors). Every voter has an equal weight in the ensemble (*EvenWeight*), ensuring no single model dominates the prediction.

This approach allows the system to specialise for different emission ranges, capturing both the behaviour of small emitters and the extreme patterns of large emitters, while avoiding the instability that can arise when a single global model attempts to fit all scales simultaneously.

## 5.2 Final Pipeline Configuration

The final pipeline for each scope consisted of:

1. Preprocessing with normalization, `arcsinh` scaling, and missing value imputation via random sampling
2. No additional feature reduction
3. Scope-specific `EvenWeightMiniModelArmyEstimator`
4. Confidence interval estimation (`BaselineConfidenceEstimator`)
5. Logarithmic scaling of targets

While these choices were informed by earlier exploratory experiments, it was necessary to confirm their effectiveness under consistent evaluation conditions. To that end, we designed a final-stage comparative experiment in which multiple candidate configurations—varying in preprocessing, imputation, and estimator settings—were trained and assessed using the same data splits and performance metrics. The objective was to ensure that the selected pipeline not only performed well in isolation, but also remained

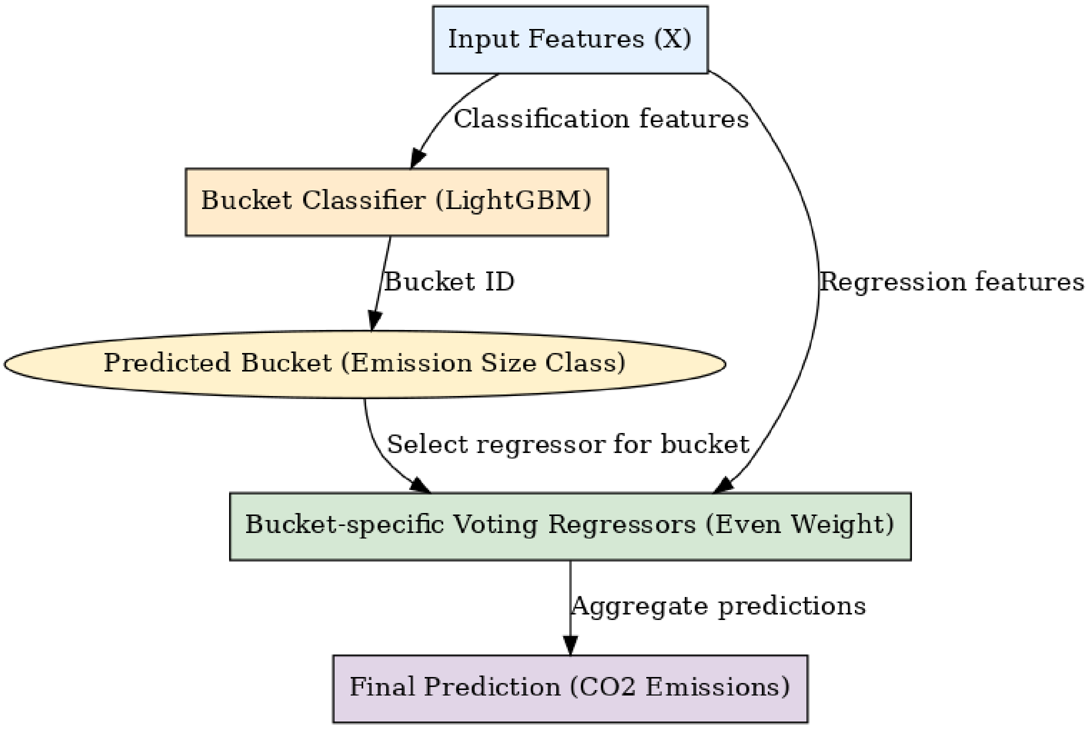
Input Features (X)
Classification features
Regression features
Bucket Classifier (LightGBM)
Bucket ID
Predicted Bucket (Emission Size Class)
Select regressor for bucket
Bucket-specific Voting Regressors (Even Weight)
Aggregate predictions
Final Prediction (CO2 Emissions)

Figure 5.1: High-level structure of the `EvenWeightMiniModelArmyEstimator`, showing bucket classification feeding into dedicated ensemble regressors for each emission size segment.

competitive against plausible alternatives. The results of this final comparative analysis are presented in Chapter 6.

# Chapter 6

# Evaluation

To assess the effectiveness of the final pipeline configuration from Chapter 5, we conducted a comprehensive validation experiment comparing multiple variations in imputation strategy, voting scheme, feature scaling, preprocessing, and bucket classification.

## 6.1 Experimental Design

The evaluation followed the `Leap Forward 1` protocol, in which an `EvenWeightMiniModelArmyEstimator` (EvenWeight MMA) was trained on Scope 1 data for every combination of the selected components:

- **Imputer:** Dummy imputation (random sampling) and alternative strategies
- **Voting scheme:** EvenWeight voting vs. Weighted voting
- **Feature scaling:** ArcSinh, Power scaling, and others
- **Preprocessor:** IID normalization vs. alternative scaling pipelines
- **Bucket classifier:** LightGBM vs. Random Forest

Each configuration was evaluated using the symmetric Mean Absolute Percentage Error (sMAPE) metric on Scope 1 emissions. For statistical robustness, 10 independent repetitions were performed for each configuration, with no additional dimensionality reduction applied.

## 6.2 Results and Insights

The results (Figure 6.1) show a tangible effect of configuration choices on predictive performance. The best observed configuration was:

```
DummyImputer { WeightedVoting { PowerFeatureScaling { IIDPreprocessing
{ RandomForestClassifier
```

However, the second-best configuration,

```
DummyImputer { EvenWeightVoting { PowerFeatureScaling {
NormedIIDPreprocessing { LightGBMClassifier
```

was selected as the final choice due to its higher stability (lower variance) across repetitions while achieving nearly identical median sMAPE.

Across configurations, the choice of *bucket classifier* emerged as the primary determinant of stability, with LightGBM generally producing more consistent results than Random Forest.

Overall, the final configuration improved median sMAPE by approximately **2%** compared to the previous baseline, **5.5%** compared to the overall average, and **6%** relative to the worst-performing setup.

## 6.3 Final Decision

Following these results, we adopted the second-best configuration for all scopes in the final `OxariMetaModel`, prioritising stability without sacrificing predictive accuracy.

## 6.4 Final Model Performance

The chosen configuration was evaluated on a held-out test set containing 8,284 companies and 13,973 data points across all scopes. Table 6.1 summarises the predictive accuracy using both symmetric Mean Absolute Percentage Error (sMAPE) and Mean Absolute Error (MAE) for each scope.

Table 6.1: Final model performance on the held-out test set.

| **Scope** | **sMAPE** | **MAE** |
|---|---|---|
| Scope 1 | 0.061 | 452.705 |
| Scope 2 | 0.057 | 738.754 |
| Scope 3 | 0.067 | 2,035.407 |

In addition to absolute performance, the confidence interval estimator (`BaselineConfidenceEstimator`) was assessed for calibration quality. Table 6.2 shows the distribution of prediction errors as a percentage of the true value. A large majority of predictions fall within a ±10% error margin, indicating that the model is both accurate and well-calibrated.

Table 6.2: Distribution of absolute percentage errors for the final model, by scope.

| **Scope** | < 10% | 10–20% | 20–50% | 50–100% | 100–200% | > 200% |
|---|---|---|---|---|---|---|
| Scope 1 | 88.16% | 5.65% | 3.61% | 0.82% | 0.39% | 1.36% |
| Scope 2 | 89.41% | 5.37% | 2.93% | 0.82% | 0.32% | 1.15% |
| Scope 3 | 84.83% | 6.91% | 5.15% | 1.40% | 0.64% | 1.07% |

These results confirm that the selected pipeline delivers high predictive accuracy and well-calibrated confidence estimates across diverse emission scales.

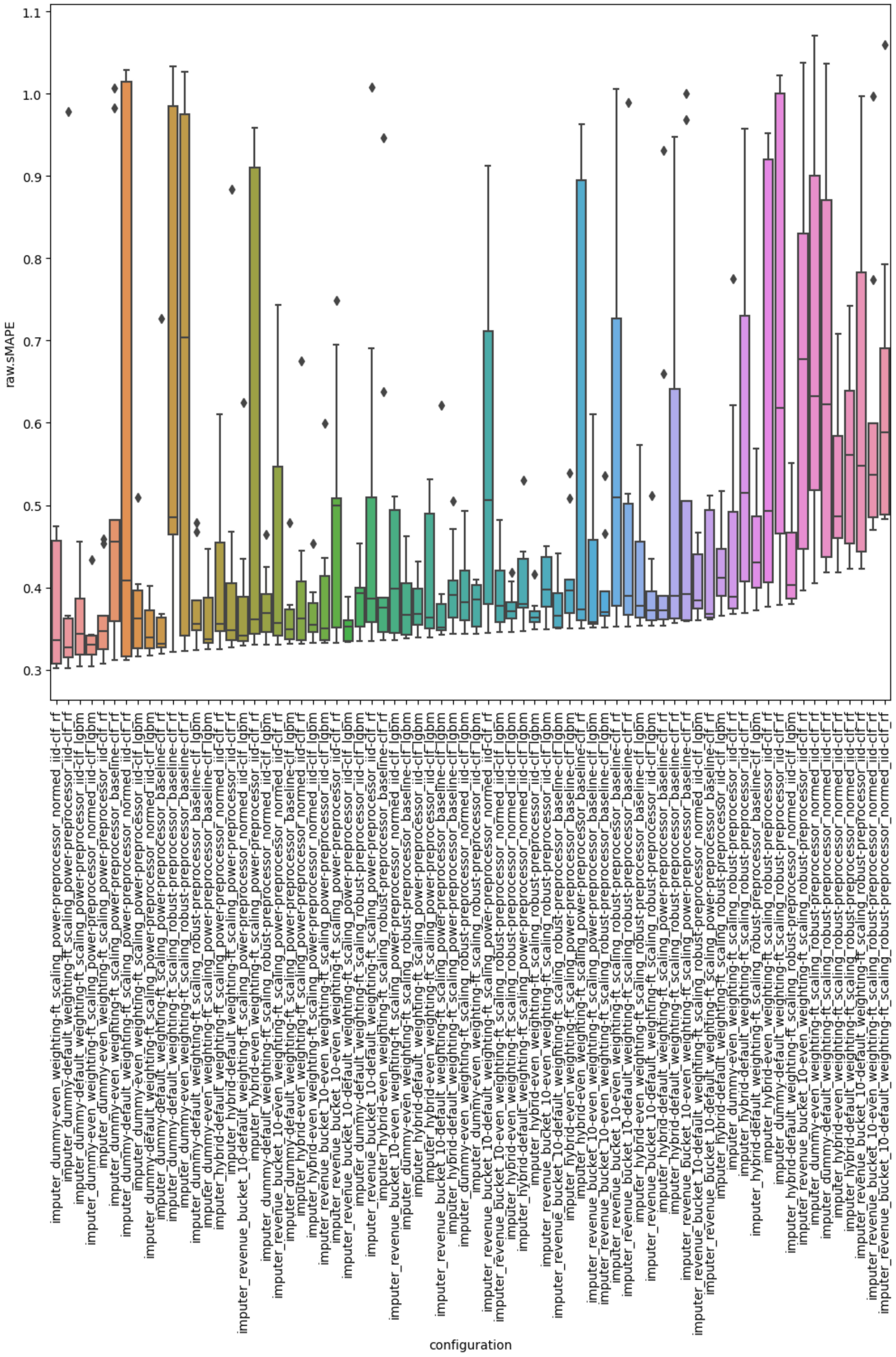

Figure 6.1: Comparison of all tested configurations in the final evaluation experiment, showing median sMAPE and variance over 10 repetitions.

# Chapter 7

# Deployment and Scalability

The final `OxariMetaModel` was exported as a `.pkl` file for portability and reproducibility. It is deployed behind a `FastAPI` interface, providing both single-prediction and bulk-prediction endpoints. In addition to emission estimation, the same model is used to impute missing information for incomplete company records, leveraging its preprocessing and imputation pipeline.

The deployment is hosted on DigitalOcean using the App Platform service, ensuring rapid scaling to handle variable request volumes. This setup supports horizontal scaling for concurrent requests and bulk inference, making it suitable for both interactive applications and large-scale batch processing.

# Chapter 8

# Conclusions

This work presented the design, selection, and evaluation of a segmented ensemble-based meta-model for predicting Scope 1, Scope 2, and Scope 3 $CO_2$ emissions. Through a series of controlled experiments, we identified a stable and high-performing pipeline configuration built around the `EvenWeightMiniModelArmyEstimator`. The final model achieves low relative and absolute errors across all scopes while providing well-calibrated confidence estimates.

Key factors contributing to performance include appropriate preprocessing (`arcsinh` scaling), random-sample imputation, and segment-specific regression via bucket classification. The approach effectively models heterogeneous emission distributions, capturing both small and large emitters.

The model has been operationalised via a `FastAPI` interface and deployed on scalable infrastructure, enabling real-world application for both prediction and imputation tasks. Future work could explore adaptive weighting schemes within the ensemble, alternative bucket definitions, and integration with live data pipelines to further enhance predictive power and robustness.

# Appendix A

# Architecture Diagram

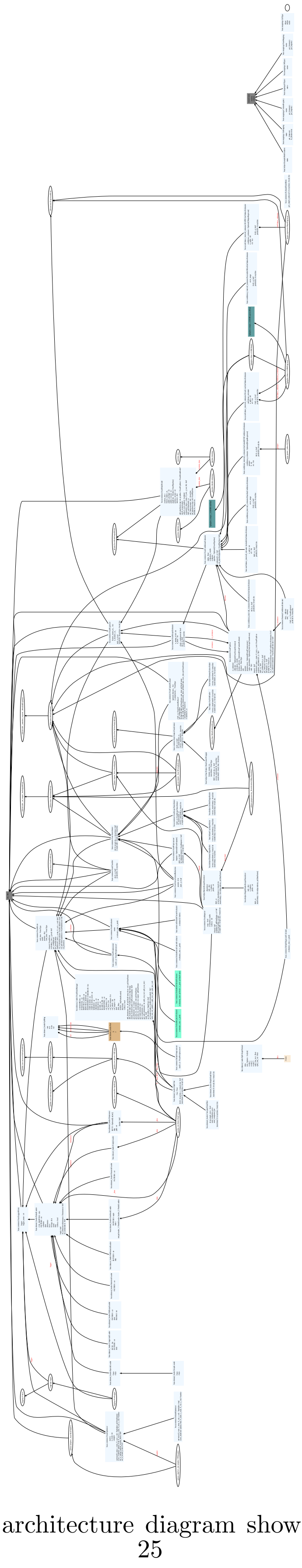

Figure A.1: Complete UML architecture diagram showing all classes, methods, and relationships.