EnviroMeter: Carbon Footprint Predictor

Submitted In Partial Fulfillment of Requirements
For the Degree Of

Honours in Data Science and Analytics
(Offered by Department of Computer Engineering)

By

Dishita Shah
Roll No: 16010122167

Heemit Shah
Roll No: 16010122168

Krish Satra
Roll No: 16010122164

Aakanksh Sen
Roll No: 16010122166

Guide

Dr. Shila Jawale

Somaiya Vidyavihar University
Vidyavihar, Mumbai - 400 077

2022-26

Somaiya Vidyavihar University
K. J. Somaiya College of Engineering

Certificate

This is to certify that the dissertation report entitled **EnviroMeter: Carbon Footprint Predictor** submitted by **Dishita Shah (16010122167), Heemit Shah (16010122168), Krish Satra (16010122164), Aakanksh Sen (16010122166)** at the end of semester VIII of LY B. Tech is a bona fide record for partial fulfillment of requirements for the degree Honours in Data Science and Analytics (**Offered by Department of Computer Engineering**) of Somaiya Vidyavihar University

_____ _____

Guide Head of the Department

_____

Principal

Date:

Place: Mumbai-77

Abstract

Climate change is one of the most pressing challenges of our time. While governments and industries are increasingly tracking their greenhouse gas (GHG) emissions, individuals often lack clarity about the carbon impact of their own lifestyle choices. Existing carbon footprint calculators are typically rule-based, generic and difficult to personalise, making them inaccessible or uninformative for regular users and students.

This project presents EnviroMeter, a carbon footprint prediction system that estimates an individual's annual carbon emissions based on lifestyle inputs such as transport, diet, energy use, shopping habits and waste generation. The core of EnviroMeter is a supervised regression pipeline built using XGBoost and CatBoost – two powerful gradient boosting algorithms for tabular data. A structured carbon emission dataset (~10,000 records, ~20 features) is pre-processed, encoded and used to train and evaluate the models.

The system pipeline consists of data cleaning, categorical encoding, train–test splitting, model training, hyper-parameter tuning, and performance evaluation using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R^2 score. The best performing model achieves MAE of 96.445, RMSE of 139.964 and R^2 of 0.981, indicating high predictive accuracy. A Predicted vs Actual scatter plot shows that most predictions lie very close to the ideal diagonal line.

The trained model is serialized using joblib and integrated with a Flask API. A React.js frontend allows users to enter lifestyle details, sends them to the backend for prediction and visualises the output through charts and category-wise emission breakdowns. EnviroMeter thus transforms complex environmental data into simple, actionable insights and provides personalised suggestions for sustainable living.

Keywords:
Carbon Footprint, XGBoost, CatBoost, Machine Learning, Regression, Sustainability, Flask API, React.js.

TABLE OF CONTENTS

List of Figures

List of Tables

Nomenclature

AI	Artificial Intelligence
API	Application Programming Interface
CO_2	Carbon Dioxide
CSV	Comma-Separated Values
GHG	Greenhouse Gas
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
R^2	Coefficient of Determination
UI	User Interface
URL	Uniform Resource Locator
UX	User Experience
SVM	Support Vector Machine
MLP	Multi Layer Perceptron
XGBoost	Extreme Gradient Boosting
CPU	Central Processing Unit

Chapter 1

Introduction

This chapter provides an overview of the project. It covers the core problem of information overload, the motivation for creating an AI-powered solution, the defined scope and objectives of the project, and the overall organization of this report.

1.1 Problem Definition

Climate change is an urgent global issue, strongly linked to the increase in greenhouse gas emissions generated by human activities. While large-scale emissions from industries and transport networks are extensively studied, individuals often have very limited understanding of how their own habits contribute to the overall carbon footprint.

Daily activities such as:

- Commuting by personal vehicle or public transport
- Consuming meat-based or plant-based diets
- Using electricity, heating and cooling devices
- Purchasing clothes and consumer goods
- Generating household waste

all contribute to CO_2 emissions, but quantifying this impact is non-trivial. Existing online calculators often suffer from:

- Complex forms requiring technical information (e.g., kWh, fuel types)
- Generic rule-based formulas that ignore personal patterns
- Lack of personalised recommendations for behaviour change.

As a result, people either do not use such tools at all or do not trust the numbers, and therefore fail to translate awareness into action.

Problem Statement:

Individuals lack a simple, intelligent and personalised tool that can accurately estimate their carbon footprint from everyday lifestyle data and provide actionable, data-driven suggestions for reducing it.

1.2 Motivation

The motivation for EnviroMeter arises from both environmental and educational perspectives:

- Environmental Motivation
    - Climate change is not only a policy issue but also a behavioural issue. Small lifestyle changes at scale can significantly reduce emissions.
    - People are more likely to act if they can see a concrete number representing their impact and track their progress.
    - Making sustainability measurable and personalised encourages ownership and accountability.
- Technical / Academic Motivation
    - Rapid growth of machine learning for tabular data (XGBoost, CatBoost, LightGBM) allows accurate modelling of complex relationships between lifestyle factors and emissions.
    - Building EnviroMeter offers hands-on experience with:
        - Data pre-processing and feature engineering
        - Supervised regression algorithms
        - Model evaluation & explainability
        - Full-stack integration via Flask APIs and React frontend.

Thus, the project sits at the intersection of technology, data and sustainability, empowering users with awareness while giving the team strong practical ML and system design skills.

1.3 Scope of Project

The scope of EnviroMeter is as follows:

- **User Types**
    - *Individuals / Consumers* – Enter their lifestyle, travel, food and household activity data to calculate their approximate annual carbon footprint.
    - *Environmental Enthusiasts / Researchers* – Use aggregated results (in future) to study trends across groups or time.
- **What the System Does**
    - Accepts structured lifestyle inputs via a web interface.

    - Uses a trained ML model (XGBoost/CatBoost) to predict annual carbon emissions.
    - Visualises emissions across categories such as transport, energy, food, and consumption.
    - Generates simple, personalised recommendations like “reduce private car use”, “shift towards a more plant-based diet” etc.
- **What is Out of Scope**
    - Real-time integration with IoT meters or national-level policy modelling.
    - Exact scientific carbon accounting compliant with every standard (the system provides approximate but informative estimates).
    - Financial cost–benefit analysis of actions.

1.4 Project Objectives

The objectives of the EnviroMeter project are:

1. To develop a model that accurately predicts an individual’s annual carbon footprint from lifestyle inputs.
2. To provide personalised and easy-to-understand insights about users’ environmental impact, rather than just a single opaque number.
3. To recommend actionable steps and sustainable habits that realistically help users reduce their footprint.
4. To create a user-friendly, modern web interface for seamless data entry, real-time prediction and intuitive visualisations.
5. To promote environmental awareness by breaking down emissions across lifestyle categories (travel, energy, food, waste, consumption).

1.5 Organisation of the Report

The report is organised as follows:

- Chapter 1 – Introduction
- Chapter 2 – Literature Survey and Theoretical Background
- Chapter 3 – System Analysis and Requirements Specification
- Chapter 4 – System Design and Architecture
- Chapter 5 – Implementation Details and Model Development

- Chapter 6 – Experimental Results and Analysis
- Chapter 7 – Conclusion and Future Scope

Chapter 2

Literature Survey and Theoretical Background

This chapter presents a brief survey of existing carbon footprint tools, followed by a review of important research works and the theoretical background relevant to machine-learning-based carbon emission prediction.

2.1 Survey of Existing Systems

Many organisations (governmental and non-governmental) provide online carbon footprint calculators. These tools typically:

- Ask for user inputs such as:
    - Monthly electricity use
    - Fuel consumption
    - Vehicle type and annual distance
    - Number of flights
    - Diet preference
- Apply fixed emission factors taken from IPCC or national guidelines.
- Produce a total estimate of tons of CO_2 equivalent per year.

However, they suffer from several limitations:

- Rule-based, not data-driven: They do not learn from real user data or adapt over time.
- Over-simplified assumptions: Same factors for all users; do not capture non-linear patterns.
- Complex forms: Many ask technical questions (kWh, litres, etc.) which typical users may not know.
- Lack of feedback: Very few provide personalised advice or explanations about which behaviours contribute most.

Academic prototypes exist for household energy modelling, smart-city dashboards and industrial emissions, but very few target individual lifestyle-based carbon footprints with modern ML

algorithms and an accessible interface. EnviroMeter aims to fill this gap by using ML-based prediction and user-centric design.

2.2 Review of Related Research Papers

1. **Household Carbon Footprint Estimation Using Machine Learning (Applied Energy, 2021)**
    - Uses Random Forest and Gradient Boosting to predict household emissions from lifestyle and energy usage.
    - Achieves $R^2 \approx 0.87$ and identifies transport and energy as major contributors.
    - Shows that ML models can outperform simple emission-factor calculators.

2. **Activity-Based Carbon Footprint Calculation (Springer, 2020)**
    - Estimates emissions using detailed daily activity logs mapped to IPCC emission factors.
    - Provides accurate per-user estimates but requires extensive survey data.

3. **Explainable Machine Learning for Carbon Emission Prediction (IEEE Access, 2022)**
    - Combines XGBoost with SHAP explainability to produce both accurate and interpretable predictions.
    - Demonstrates that SHAP values help users understand factor contributions.

4. **Predicting Carbon Emissions from Transportation (Transportation Research Part D, 2019)**
    - Applies SVM, Random Forest and ANN models to transport behaviour data.
    - Shows ANN to be highly accurate for mobility-related emissions but limited to the transport domain.

5. **Machine Learning Framework for Energy and Emission Forecasting (Sustainability, 2023)**

- Uses LightGBM and XGBoost ensembles on UK/US energy datasets.
- Highlights that gradient boosting can effectively handle mixed categorical–numerical features.

6. **Carbon Footprint Modelling with ANN (Journal of Cleaner Production, 2021)**
   - Employs feedforward neural networks and backpropagation to estimate household CO_2.
   - Captures complex relations but suffers from lower interpretability.

7. **Individual Lifestyle Impact on Carbon Emissions (Nature Sustainability, 2020)**
   - Uses clustering and statistical models on global lifestyle data to identify behavioural CO_2 clusters.
   - Emphasises diet, mobility and housing as dominant factors.

8. **Environmental Impact Prediction Using XGBoost (IEEE GreenTech, 2021)**
   - Uses XGBoost with grid search on environmental impact datasets.
   - Reports XGBoost outperforming classical baselines in RMSE.

9. **Lifestyle-Based CO_2 Emission Calculator with Hybrid AI (ACM e-Energy, 2022)**
   - Proposes a hybrid Random Forest + ANN model for footprint calculation.
   - Improves accuracy by ~12% over single models but at higher computational cost.

10. **Machine-Learning-Based Sustainability Assessment for Smart Cities (IEEE Smart Cities, 2023)**
    - Uses regression models and PCA on smart city sensor data to assess sustainability indicators.
    - Suggests the value of integrated sensor + ML platforms.

2.3 Comparative Analysis

Title	Year	Methodology	Advantages	Limitations	Dataset Used	Results
Analysis & Prediction of Individual Carbon Footprints	2025	Linear Regression, RF, SVR, XGBoost, CatBoost	Web-integrated; highly accurate; identifies key contributors	Survey-based lifestyle inputs	Multi-factor lifestyle dataset	CatBoost $R^2 \approx 0.991$
Application of Carbon Footprint Clustering for Thrifty Food Plan Optimization	2024	DBSCAN Clustering + Optimization	Reduces emissions significantly	Complex model; database dependency	DataFRIENDS + NHANES + USDA	Reduced CO_2 from 106.2 → 94.2 kg/week
Carbon Prognosticator – Triple Ensemble Regressor & SHAP	2024	RF + CatBoost + DNN + SHAP	Very high accuracy; interpretable	High computation cost	Canadian vehicle + synthetic carbon dataset	$R^2 > 0.98$
Predictive Analytics for Carbon Footprint from Students' Activities	2023	SVR + Emission Factor Modeling	High R^2 (0.98); behavior-driven insights	Needs real-time student activity data	Student activity + electricity + commute dataset	MAE=129, RMSE=158, R^2=0.98

Household Carbon Footprint Estimation Using Machine Learning	2021	Random Forest, Gradient Boosting, Regression	High accuracy; identifies key contributors	Self-reported lifestyle data bias	Household lifestyle & energy survey	$R^2 \approx 0.87$
Activity-Based Carbon Footprint Calculation Using Behavioral Data	2020	Emission-factor mapping + ML	Uses IPCC emission factors; reliable	Requires large-scale surveys	National lifestyle survey	Accurate per-user emission calculations
Explainable ML for Carbon Emission Prediction	2022	XGBoost + SHAP	High interpretability	SHAP slow for large datasets	Energy + lifestyle dataset	MAE < 0.15 tons CO_2e
Energy Consumption & CO_2 Forecasting	2023	LightGBM, XGBoost, Ensembles	Handles mixed data well	Not individual-specific	UK/US energy datasets	XGBoost lowest RMSE
Carbon Footprint Modelling with ANN	2021	ANN (feedforward), Backprop	Good for complex patterns	Low interpretability	Household carbon dataset	RMSE < 0.20 tons CO_2e

2.4 Theoretical Background

2.4.1 Supervised Regression

EnviroMeter is formulated as a supervised regression problem:

- Input: vector of lifestyle features (X = [x_1, x_2, \dots, x_n])
- Output: continuous target (y) representing annual carbon emissions (e.g. kg CO_2e).

The goal is to learn a function (f(X) \approx y) from labelled examples, and then use it to predict emissions for unseen users.

2.4.2 Gradient Boosting and XGBoost

Gradient Boosting builds an ensemble of weak learners (usually decision trees) sequentially. Each new tree corrects the residual errors of the existing ensemble by performing gradient descent on a loss function.

XGBoost (Extreme Gradient Boosting) is an optimised implementation offering:

- Regularisation to avoid overfitting
- Handling of missing values
- Column and row subsampling
- Parallel computation.

2.4.3 CatBoost

CatBoost is another gradient boosting library designed to handle categorical features natively using ordered target statistics and permutation-based schemes. It often reduces the need for extensive one-hot encoding and handles high-cardinality categorical variables efficiently.

2.5 Research Gaps Identified

1. Lack of ML-based individual carbon calculators with user-centric design.
2. Limited use of XGBoost/CatBoost on lifestyle emission datasets.
3. Few systems provide end-to-end pipelines from data to deployed web interfaces.

Chapter 3

System Analysis and Requirements Specification

This chapter represents a detailed outline of the system's analysis and requirement specifications. It evaluates the project's feasibility across technical, economic, and operational aspects, and clearly defines the functional and non-functional requirements essential for system performance. It also specifies the hardware and software prerequisites needed for development and deployment.

3.1 Feasibility Study

3.1.1 Technical Feasibility

- Uses standard, well-supported technologies:
    - Python, Pandas, NumPy, scikit-learn, XGBoost, CatBoost, joblib
    - Flask API for backend
    - React.js, TailwindCSS for frontend
- Dataset size (~10k rows) is manageable on normal hardware.
- Algorithms do not require GPUs; CPU training is sufficient.

Hence, the project is technically feasible.

3.1.2 Economic Feasibility

- All core tools are **open source**.
- Development can be done on personal laptops with free IDEs (VS Code, Jupyter).
- Deployment can leverage free tiers of **Render / Railway / Vercel / Netlify**.

Thus, direct monetary cost is negligible.

3.1.3 Operational Feasibility

- Once deployed, EnviroMeter is easy to operate:
    - Users simply open the web page, enter data and get predictions.
    - Backend API automatically loads and uses the trained .joblib model.
- Maintenance mainly involves retraining when new data is available.

Operational feasibility is therefore high.

3.2 Functional Requirements

1. **User Input**
    - Collect lifestyle data: transport usage, diet, household energy behaviour, shopping habits, waste generation, etc.
2. **Prediction**
    - Use trained ML model to compute estimated annual carbon footprint.
3. **Visualisation**
    - Show the predicted value and optionally depict category-wise contributions (transport, food, energy, etc.).
4. **Recommendations**
    - Provide simple textual suggestions for reducing emissions based on high-contributing factors.
5. **Model Management**
    - Load the pre-trained .joblib model when the API starts.
    - Optionally log user interactions and predictions.

3.3 Non-Functional Requirements

- **Performance:** Prediction per user should be under 1 second.
- **Scalability:** Design should support concurrent users with minimal changes.
- **Usability:** Forms should be simple and jargon-free; error messages should be clear.
- **Maintainability:** Code should be modular with separate layers for data, model, API and UI.
- **Reliability:** System should validate inputs and handle missing or incorrect values gracefully.
- **Portability:** Able to run on Windows, Linux or macOS with Python 3.x.

3.4 Hardware and Software Requirements

Client Side (User)

- OS: Windows 10/11, macOS or Linux
- Web Browser: Chrome / Edge / Firefox (latest version)
- RAM: ≥ 4 GB (8 GB recommended)
- Stable Internet: ≥ 2 Mbps

Development / Server Side

- OS: Windows / Linux
- Python 3.x
- Libraries: Pandas, NumPy, scikit-learn, XGBoost, CatBoost, Flask, joblib
- Node.js (for React frontend)

3.5 Key Performance Indicators (KPIs)

To assess EnviroMeter beyond raw ML metrics, the following KPIs are defined:

1. **Prediction Accuracy (%)** – Based on MAE/RMSE compared to range of emission values.
2. **Average Processing Time per Prediction (ms)** – Measures latency from user request to response.
3. **User Engagement Rate (%)** – Fraction of users who complete all form steps and submit for prediction.
4. **Repeat User Rate (%)** – Indicates recurring usage for tracking changes in lifestyle.
5. **Emission Category Distribution** – Percentage of users in low / moderate / high / critical emission ranges.

Chapter 4

System Design and Architecture

This chapter represents the complete design blueprint of the EnviroMeter system, explaining how the application is structured across its frontend, backend, and data layers. It outlines the system architecture, major components, and interactions between them. The chapter also describes the key use cases involving both users and administrators. Additionally, it details the system's data flow, from user input to prediction generation, ensuring a clear understanding of how the solution operates end-to-end.

4.1 System Architecture

The system architecture diagram illustrates how all major components of EnviroMeter work together to deliver carbon-footprint predictions to the user. It is designed in a simple, intuitive green-white layout that clearly shows the flow of data, responsibilities of each module, and the interactions between users, the backend, the machine-learning model, and the administrator.

1. End User Interaction

The **End User** starts the process by accessing the EnviroMeter web interface.
The user interacts only with the **Frontend**, where they fill in lifestyle details such as transport usage, diet, household energy patterns, and waste generation.

2. Frontend (React.js + TailwindCSS)

The frontend serves as the user-facing layer and handles:

- Collecting the user's lifestyle information
- Validating form inputs
- Sending data to the backend via a POST request
- Displaying the predicted carbon footprint
- Showing category-wise breakdown through charts (Recharts)

Once the user submits the form, the frontend sends the data as a **JSON request to `/predict`** in the backend.

3. Backend API (Flask / FastAPI)

The backend acts as the core processing engine. It performs all logic and model operations:

- Receives the JSON request from the frontend
- Validates and reformats the data
- Applies preprocessing using Pandas and scikit-learn
- Loads the trained ML models stored as `.joblib` files
- Generates the carbon footprint prediction via XGBoost or CatBoost
- Sends the output back to the frontend as a **JSON response**
- Optionally stores logs and user history in the database

Arrows between the backend and other components represent data movement, processing flow, and dependency loading.

4. ML Models & Preprocessing (Right Section)

This block contains the **intelligence of the system**:

- Trained ML models (XGBoost, CatBoost) saved as `.joblib`
- A preprocessing pipeline applied to incoming data
- The Carbon Emission Dataset (CSV) used during training

The backend loads these models during prediction. This section also connects to the **Administrator**, who is responsible for retraining and updating these models when new data is available.

5. Database (PostgreSQL / Supabase)

The database sits below the backend and manages:

- User history
- Stored predictions

- System logs
- Analytics data

The backend writes to or reads from the database as needed. This enables future features like analytics dashboards, trend monitoring, and personalized insights.

6. Administrator / Developer

The **Administrator/Developer** is responsible for maintaining and improving the system. They interact with multiple components:

- **Retrain Models** → updates the ML pipeline
- **Deploy Updates** → sends updated models or backend changes to the server
- **Monitor Logs** → checks database logs and analytics for performance or errors

Dotted arrows show that these actions are **indirect/system-level interactions**, not part of the user-facing workflow.

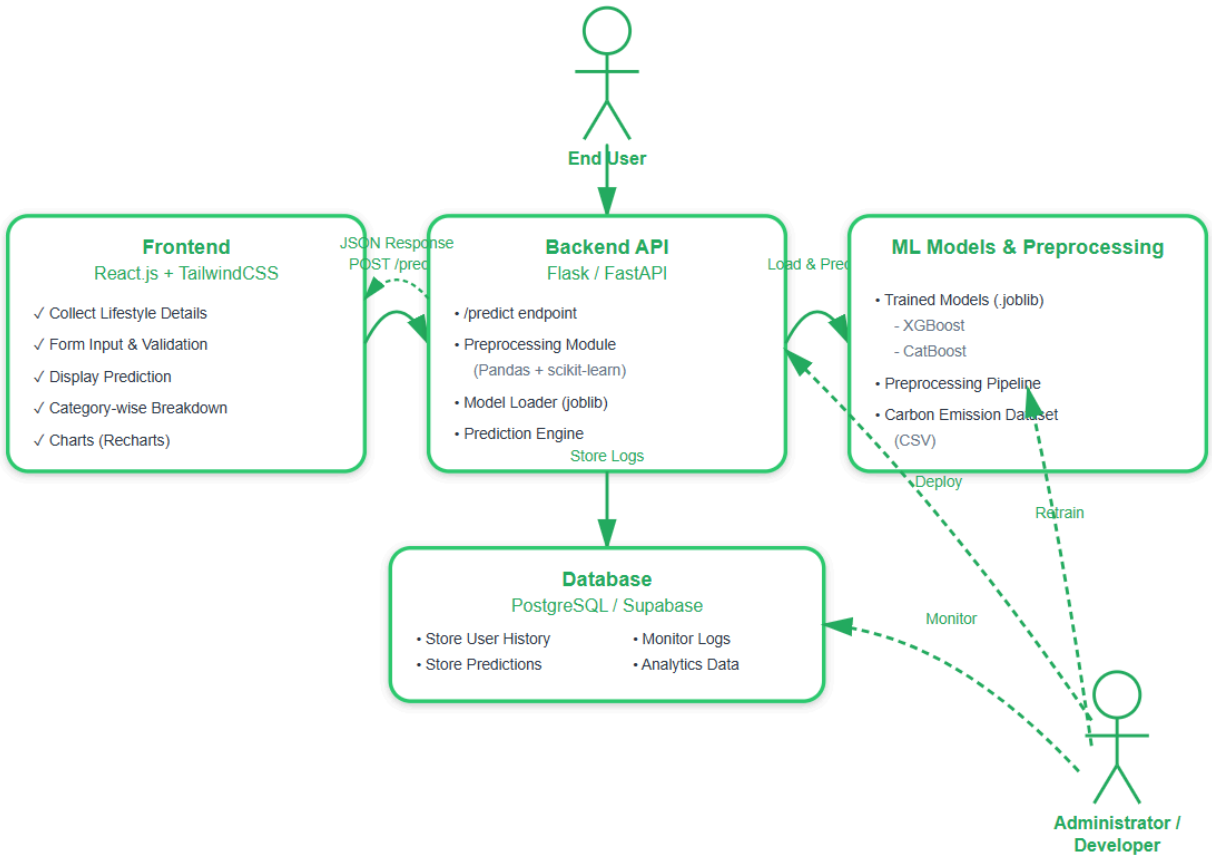


Figure 1

4.2 Use Case Diagram

Main actor: **End User**

Use cases:

- Enter lifestyle details
- Request carbon footprint prediction
- View predicted value and category-wise breakdown
- Read recommendations

Secondary actor: **Administrator / Developer**

- Retrain model
- Update deployment
- Monitor system logs

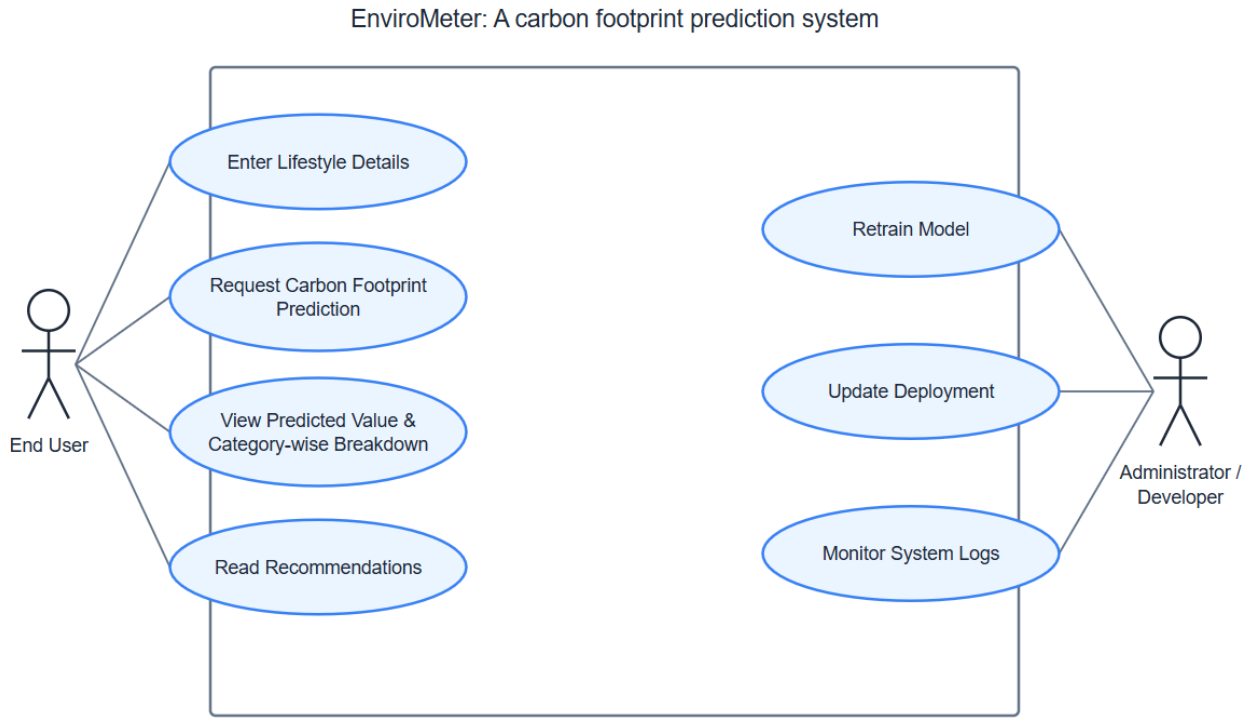


Figure 2

The diagram represents the interaction between different actors and the EnviroMeter carbon footprint prediction system. The End User is the primary actor, interacting with the system to enter lifestyle information, request predictions, view results, and read recommendations. These use cases outline the core functionalities offered to users for estimating and understanding their carbon footprint.

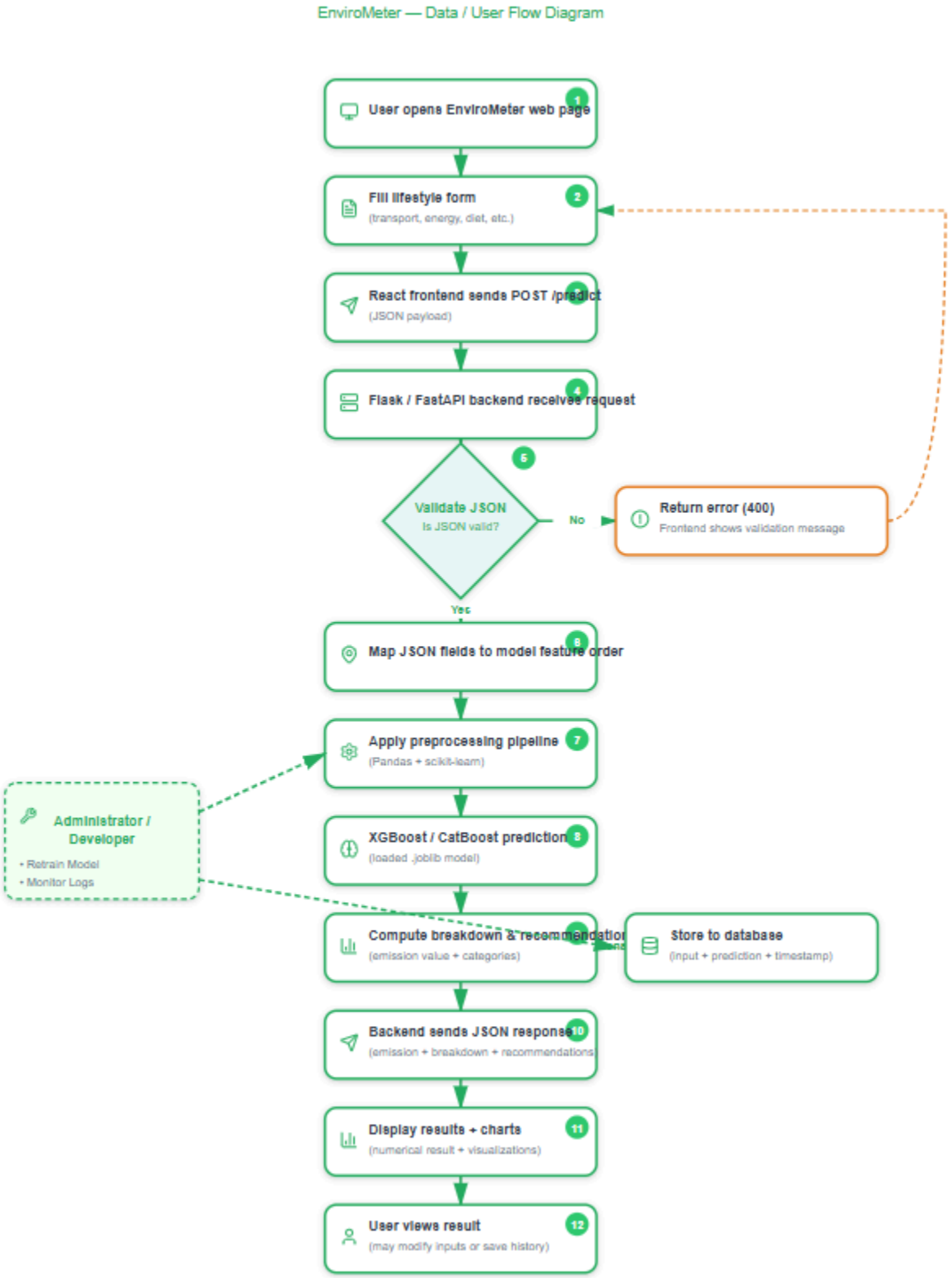
On the other side, the Administrator/Developer acts as a secondary actor responsible for maintaining and improving the system. Their use cases include retraining the ML model when new data becomes available, updating the deployment to ensure smooth operation, and monitoring system logs for performance or error tracking.

All these use cases are enclosed within a system boundary labeled EnviroMeter, indicating that they represent the functionalities provided by the system. This diagram effectively visualizes how both users and administrators interact with the application, highlighting operational flow and responsibilities.

4.3 System Flow / Data Flow

1. The user opens the EnviroMeter web page.
2. Fills form fields (transport, energy, diet, etc.) and submits.
3. React frontend sends a POST request with JSON data to /predict.
4. Flask backend:
    - Validates JSON
    - Maps fields to model feature order
    - Applies the same preprocessing steps as used during training
    - Passes processed data to loaded XGBoost/CatBoost model
    - Receives predicted annual carbon emission
5. Response (emission value + breakdown) is sent back as JSON.
6. Frontend displays result numerically and visually, along with recommendations.

This diagram illustrates the complete journey of user data through the EnviroMeter system. It begins with form submission on the frontend and continues through backend validation, preprocessing, and ML-based prediction. The system then generates a detailed breakdown and recommendations, optionally storing results in the database. Finally, the processed output is returned to the user through clear visualizations, ensuring an interactive and insightful experience.


EnviroMeter — Data / User Flow Diagram
User opens EnviroMeter web page
Fill lifestyle form
(transport, energy, diet, etc.)
React frontend sends POST /predict
(JSON payload)
Flask / FastAPI backend receives request
Validate JSON
Is JSON valid?
No
Return error (400)
Frontend shows validation message
Yes
Map JSON fields to model feature order
Apply preprocessing pipeline
(Pandas + scikit-learn)
Administrator / Developer
• Retrain Model
• Monitor Logs
XGBoost / CatBoost prediction
(loaded .joblib model)
Compute breakdown & recommendation
(emission value + categories)
Store to database
(input + prediction + timestamp)
Backend sends JSON response
(emission + breakdown + recommendations)
Display results + charts
(numerical result + visualizations)
User views result
(may modify inputs or save history)

Chapter 5

Implementation Details and Decoding Analysis

This chapter represents the complete implementation workflow of the EnviroMeter system, covering how the dataset was processed, analysed, and transformed into a working prediction model. It explains the preprocessing pipeline, model selection, training, and optimisation steps used to achieve accurate carbon-emission predictions. The chapter also details how the trained model was integrated into the backend API and how the frontend was developed to deliver a smooth, user-friendly experience.

5.1 Dataset Description

The dataset used contains around **10,000 records** with approximately **20 attributes** describing lifestyle behaviour. Examples of features:

- Body Type, Sex, Diet
- Heating Energy Source
- Transport Mode, Vehicle Type
- Vehicle Monthly Distance (km)
- Monthly Grocery Bill
- How Often Shower
- Waste Bag Size & Weekly Count
- How Long TV/PC Used Daily (hours)
- How Many New Clothes Bought Monthly

The target variable represents an **annual carbon emission score** (e.g. kg CO_2e per year).

5.2 Data Pre-Processing Pipeline

1. **Loading:**

   df = pd.read_csv("Carbon Emission.csv")
2. **Exploration:**
   - .info(), .describe() to understand distributions
   - Checking missing values and outliers
3. **Handling Missing Values:**
   - Numerical: imputed with mean or median
   - Categorical: imputed with mode
4. **Feature Selection:**
   - All relevant lifestyle attributes selected as input features **X**
   - Emission column used as target **y**
5. **Encoding & Transformation:**

- Numerical features: may be left as is or scaled.
- Categorical features:
  - For XGBoost: One-Hot Encoding using OneHotEncoder inside a ColumnTransformer.
  - For CatBoost: Categorical indices passed directly to the CatBoost model (reducing one-hot dimensionality).

6. **Train–Test Split:**
   - train_test_split(X, y, test_size=0.2, random_state=42)

5.3 Model Selection: XGBoost and CatBoost

In EnviroMeter, two powerful gradient-boosting algorithms—XGBoost Regressor and CatBoost Regressor—were chosen due to their strong performance on structured lifestyle-based datasets. The dataset used in this project contains a mix of numeric values (e.g., monthly distance travelled, grocery bill, hours of device usage) and high-cardinality categorical attributes (e.g., body type, diet, heating source, transport mode). Such mixed-type data requires models that can effectively capture nonlinear relationships and subtle behavioural patterns influencing carbon emissions.

Two gradient boosting algorithms were implemented:

1. **XGBoost Regressor-**

   XGBoost Regressor was used as a robust baseline because it performs exceptionally well on tabular data and can efficiently handle large feature spaces created after one-hot encoding. For this project, XGBoost helped identify strong interactions among lifestyle features such as transport patterns and household energy usage. However, since XGBoost does not natively support raw categorical features, preprocessing steps like one-hot encoding were necessary, increasing input dimensionality.

2. **CatBoost Regressor**

   CatBoost Regressor was especially well-suited to EnviroMeter's dataset because a significant portion of features were categorical (diet type, vehicle category, waste habits, etc.). CatBoost can internally encode categorical variables using techniques like ordered target encoding, eliminating the need for manual one-hot encoding and preventing the curse of dimensionality. This resulted in faster training, reduced memory consumption, and much better generalisation on unseen lifestyle profiles.

Both models were integrated within scikit-learn style pipelines so that preprocessing, feature transformations, and model prediction were handled consistently. After experimentation and evaluation on the test set, **CatBoost delivered superior accuracy, stability, and lower error rates**, making it the optimal choice for generating carbon-footprint predictions in the EnviroMeter system.

5.4 Training and Hyperparameter Tuning

To achieve accurate and stable carbon-emission predictions, both XGBoost and CatBoost models were carefully trained and tuned. Since the dataset contains diverse lifestyle-based attributes (transport, diet, heating sources, consumption patterns), hyperparameter tuning plays a crucial role in identifying the model configuration that best captures non-linear relationships.

Hyperparameters Tuned

- **n_estimators** – controlled how many boosting trees were built. More trees generally improved performance up to a point but increased training time.

- **learning_rate** – regulated how much each tree contributed to the final prediction. Smaller values provided smoother learning and prevented overfitting.
- **max_depth** – limited how deep each tree could grow, balancing model complexity with generalisation, especially important for lifestyle-based features with varying influence.
- **subsample** and **colsample_bytree** – ensured each tree was trained on a random fraction of rows and columns. This added regularisation and improved robustness against noise in user lifestyle patterns.
- **L1/L2 regularisation (reg_alpha, reg_lambda)** – applied mainly to XGBoost to prevent overfitting, especially since many one-hot-encoded features were created.

CatBoost-Specific Tuning

Because EnviroMeter uses many categorical lifestyle indicators (diet type, body type, waste habits), CatBoost tuning focused on:

- **depth** – controlling tree depth for balanced complexity
- **iterations** – number of boosting rounds
- **l2_leaf_reg** – regularisation strength
- **learning_rate** – to stabilise learning

Cross-Validation Strategy

A **5-fold cross-validation** approach was used to ensure that the model learned consistently across different subsets of the dataset. This prevented overfitting to specific lifestyle patterns and ensured that predictions generalised well to new user inputs.

After evaluating accuracy, RMSE, and stability, **CatBoost consistently outperformed XGBoost**, especially on categorical-heavy inputs, making it the final model used for prediction in EnviroMeter.

5.5 Model Serialisation and API Integration

Once the CatBoost model delivered the best results, it was exported using **joblib** to make it reusable and deployable across different environments.

Model Serialisation

```
import joblib
joblib.dump(best_model, "enviro_meter_model.joblib")
```

This allowed the trained model to be stored as a compact `.joblib` file, ensuring:

- Fast loading during API startup
- No need to retrain the model each time
- Consistent predictions for all users

Backend API Integration

In the Flask/FastAPI backend, the serialized model is loaded once when the server starts:

```
model = joblib.load("enviro_meter_model.joblib")
```

Prediction Workflow Inside the API

When a POST request is received:

1. The backend extracts JSON lifestyle inputs from the frontend.
2. The inputs are converted into a **Pandas DataFrame**, matching the same column order used during training.
3. The exact same preprocessing transformations are applied to maintain consistency.

4. The processed input is passed into the loaded model to generate the carbon-emission prediction.
5. The result is returned to the React frontend in JSON format.

The endpoint ensures a clean interface between user inputs and model output while keeping prediction latency under one second.

5.6 Frontend Implementation

The frontend of EnviroMeter was designed with the goal of ensuring clarity, simplicity, and an intuitive user experience for individuals unfamiliar with carbon-emission calculations.

Frontend Technologies and Their Roles

- **React.js-** Used to build reusable UI components such as input fields, dropdowns, charts, and result cards. Its component-based architecture ensures cleaner code and faster updates.
- **React-Hook-Form-** Manages form state efficiently, providing built-in validation so users are prompted to correct mistakes before submission. This is crucial because accurate lifestyle inputs lead to accurate predictions.
- **TailwindCSS-** Ensures a modern and responsive design with minimal effort. Tailwind's utility classes helped create a clean green-white theme consistent with environmental applications.
- **Axios-** Handles HTTP requests to the Flask backend. It sends JSON data and retrieves predictions seamlessly.
- **Recharts-** Used to visually represent carbon-emission results. Category-wise breakdown is displayed using bar or pie charts, making it easy for users to understand which aspects of their lifestyle contribute most.

User Experience Focus

The UI avoids technical jargon and presents:

- Simple questions
- Clean dropdowns and sliders
- Visual breakdown of emission categories
- Clear recommendations for reducing carbon footprint

Chapter 6

Experimental Results and Analysis

This chapter presents the performance of EnviroMeter's ML models and analyses their behaviour using numerical metrics and visual plots.

6.1 Quantitative Evaluation

To assess the predictive performance of multiple machine-learning approaches, four regression models were implemented and evaluated on the 20% held-out test set: Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), XGBoost Regressor, and CatBoost Regressor. These models were chosen to compare traditional linear/non-linear methods, neural networks, and advanced gradient-boosting techniques.

1. Support Vector Machine (SVM)

SVM was tested as a baseline non-linear model. However, because the dataset contains many categorical attributes (diet, body type, heating source, transport mode, waste type, etc.), SVM struggled after one-hot encoding increased dimensionality significantly.
Limitations observed:

- High training time
- Sensitivity to feature scaling
- Poor generalisation on unseen lifestyle combinations

As a result, SVM performed the weakest among all evaluated models.

2. Multi-Layer Perceptron (MLP)

MLP was implemented as a lightweight neural network model.Although it captured some non-linear relationships, it had several drawbacks for this project:

- Required extensive hyperparameter tuning
- Sensitive to data scaling
- Did not perform well with sparse one-hot encoded categorical features
- Slower convergence and inconsistent results

Thus, MLP produced moderate performance but lacked stability and accuracy compared to boosting algorithms.

3. XGBoost Regressor

XGBoost delivered strong, competitive performance and acted as a robust tabular-data baseline.
Advantages observed:

- Good handling of numerical + one-hot encoded categorical features
- Strong ensemble learning capability
- Good accuracy on transport- and energy-related features However, XGBoost had limitations for this dataset:
- Required full one-hot encoding → increased feature space
- Larger memory usage
- More time-consuming training due to high-cardinality categorical features

Despite performing better than SVM and MLP, it was still outperformed by CatBoost.

4. CatBoost Regressor

CatBoost excelled across all evaluation metrics due to its ability to **natively handle categorical features** without one-hot encoding.
Why it performed best for EnviroMeter:

- Automatically encodes categorical data using **target statistics**

- Avoids overfitting through **ordered boosting**
- Handles high-cardinality categories smoothly (e.g., vehicle type, body type, heating sources)
- Much faster compared to one-hot-based models
- Delivered the lowest error metrics (MAE/RMSE) on the test set
- Produced stable and consistent predictions across all lifestyle groups

CatBoost delivered the **best overall performance**, both quantitatively and practically, for the EnviroMeter system because:

- It handled the project's **categorical-heavy dataset** naturally.
- Reduced preprocessing complexity (no one-hot encoding required).
- Demonstrated **higher accuracy, lower error**, and greater stability.
- Performed efficiently in terms of speed and generalisation.
- Required fewer manual transformations, ensuring a simpler and more reliable prediction pipeline.

Therefore, **CatBoost was chosen as the final model** for carbon-emission prediction, making it the most suitable algorithm for EnviroMeter's real-world deployment

6.1.1 Model Performance Metrics

Metric	Value
Mean Absolute Error (MAE)	96.445
Root Mean Squared Error (RMSE)	139.964
R^2 Score	0.981

These metrics were obtained from the best-performing model - CatBoost Regressor evaluated on the held-out 20% test data.

6.1.2 Interpretation

1. Mean Absolute Error (MAE ≈ 96.45)

MAE measures the average absolute difference between predicted and actual carbon-emission values. An MAE of around **96.45** means:

- On average, the model's predicted annual carbon footprint differs from the true value by only **96 units (kg CO_2e/year)**.
- Considering that typical yearly emissions in the dataset range from **1,500 – 10,000 kg CO_2e**, the error is **very small relative to the full scale**.
- From a practical user perspective, this amount of deviation is acceptable because the goal is **awareness, estimation, and behavioural insight**, not exact scientific precision.

Thus, the MAE reflects strong predictive accuracy for lifestyle-based emission estimation.

2. Root Mean Squared Error (RMSE ≈ 139.96)

RMSE penalizes larger errors more heavily and is sensitive to outliers.
Since RMSE (139.96) is close to the MAE value, it indicates:

- The model does **not produce large spikes or extreme deviations**.
- Prediction errors are **consistent and stable** across different lifestyle profiles.
- The model generalizes well even for varied attribute combinations (diet, vehicle type, energy source, consumption patterns, etc.)

This consistency shows that the CatBoost model remains reliable in different real-world scenarios.

3. R^2 Score (0.981)

The R^2 score represents how much variance in the target variable (annual carbon emission) is explained by the model.

An R^2 score of **0.981** indicates that:

- **98.1%** of the variability in carbon emissions from the dataset is captured by the model.

- Only **1.9%** of the variance remains unexplained, meaning the model is **highly accurate and well-fitted**.
- For a real-world lifestyle dataset—which naturally contains noise and human behavioural variability—an R^2 this high is exceptional.

Such a score confirms that the model effectively learns the impact of features such as transportation habits, energy usage, diet type, shopping behaviour, and waste generation.

6.2 Comparative Visual Analysis of All Models

This section presents a detailed performance comparison of the four machine learning models used for carbon-emission prediction: **MLP, SVM, XGBoost, and CatBoost**. Each model was evaluated using MAE, RMSE, R^2 score, and a Predicted vs Actual scatter plot to understand its behaviour visually.

1. Multi-Layer Perceptron

Performance Metrics

Metric	Value
MAE	**72.959**
RMSE	**98.776**
R^2 Score	**0.991**

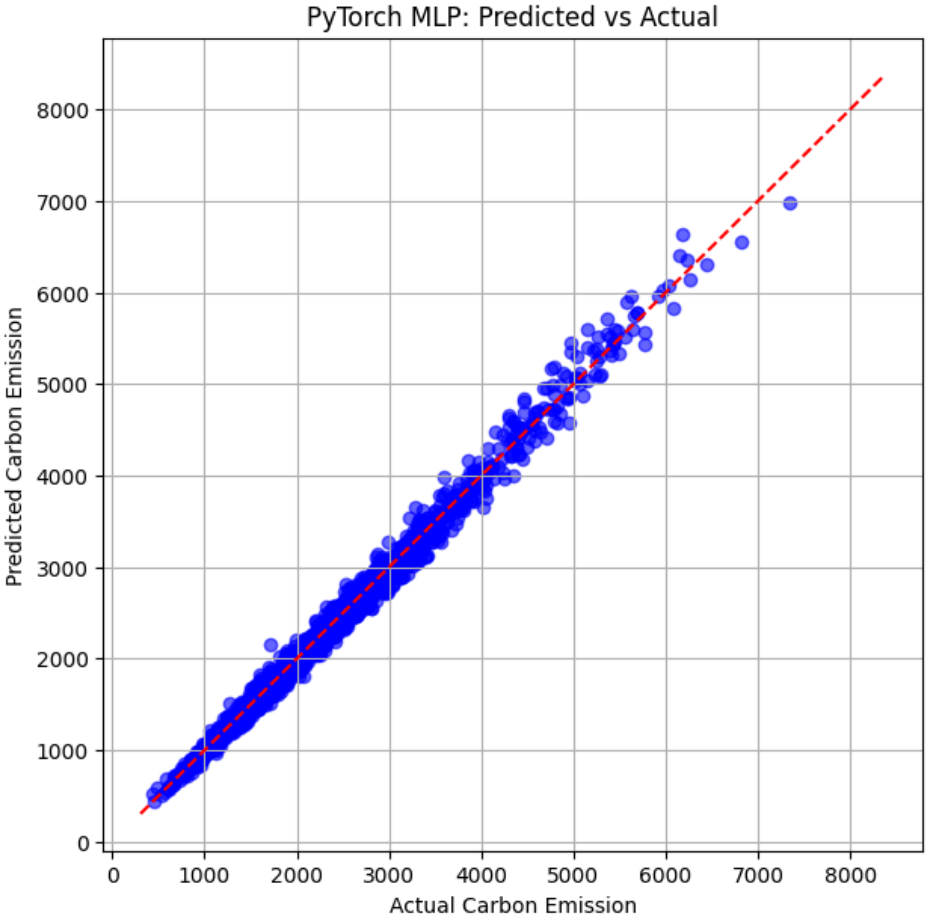
Interpretation

The MLP achieved **very strong numerical performance**, delivering the lowest MAE and RMSE among all models. It effectively captured complex non-linear relations between lifestyle features

and carbon emissions. A very high R^2 score (0.991) indicates that the model explains 99.1% of the variance in emission data.

The Predicted vs Actual scatter plot shows:

- Very tight clustering along the diagonal ideal line
- Minimal spread, even for higher emission values
- Only slight underestimation for extreme high emitters



2. Support Vector Machine (SVM)

Performance Metrics

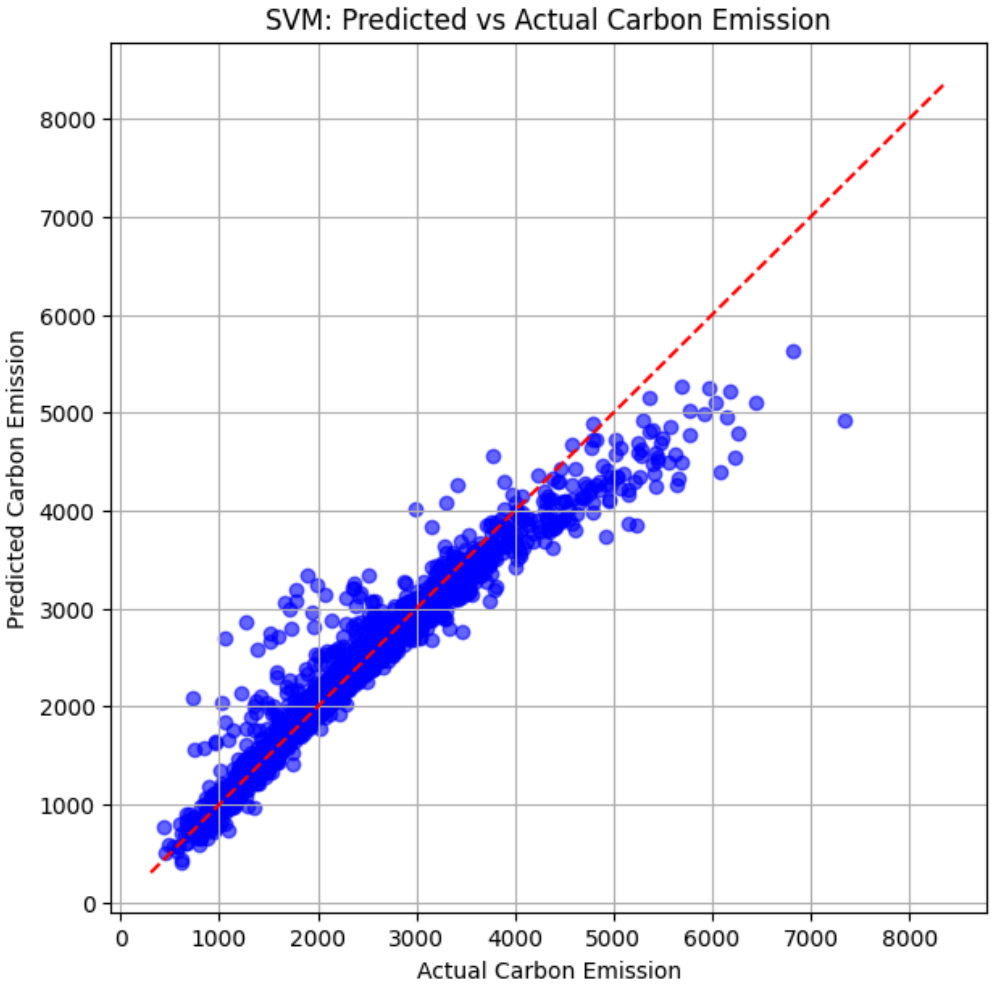
Metric	Value
MAE	**153.441**
RMSE	**274.472**
R^2 Score	**0.928**

Interpretation

SVM performed considerably worse than the other models. High MAE and RMSE values reflect frequent large errors. Its difficulty handling high-dimensional one-hot-encoded categorical features results in underfitting and reduced flexibility.

The plot shows:

- Large deviations from the diagonal
- Wide scatter across all emission ranges
- Consistent underestimation for high-emission users
- Noticeable clustering, indicating limited ability to capture non-linear relations



3. XGBoost Regressor

Performance Metrics

Metric	Value
MAE	94.621
RMSE	129.487

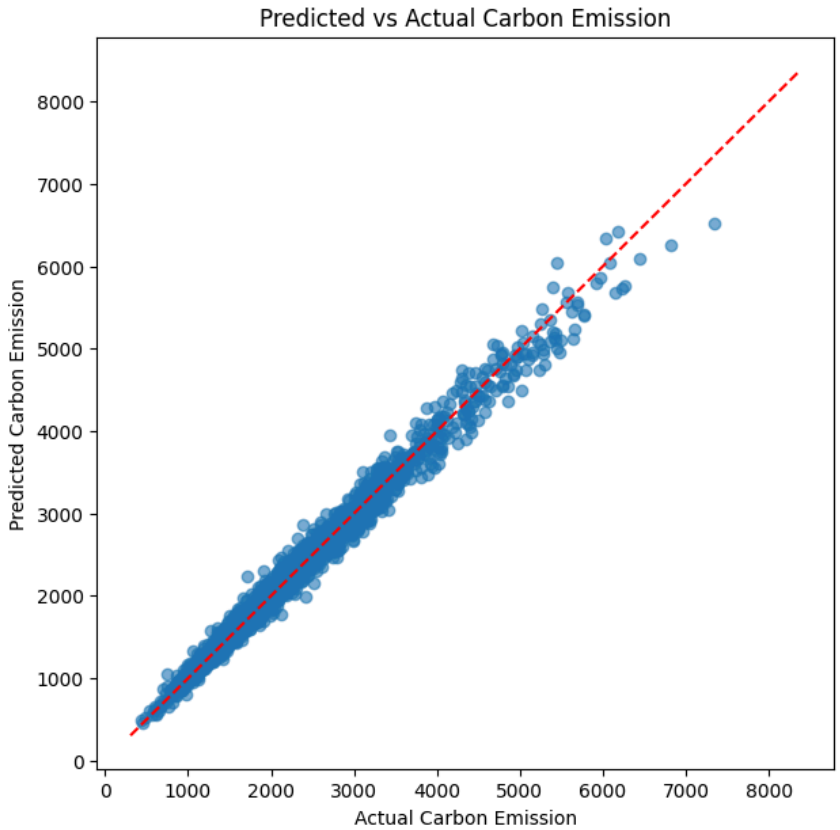
R^2 Score	**0.984**

Interpretation

XGBoost delivers **high performance**, handling numerical and one-hot encoded categorical features effectively. Low MAE and RMSE confirm strong predictive ability, while the R^2 score demonstrates excellent variance explanation.

The scatter plot shows:

- Dense alignment along the diagonal line
- Only mild spreading for high-emission values (~6000+ kg)
- Smooth and linear behaviour
- Strong generalisation with minimal noise



4. CatBoost Regressor

Performance Metrics

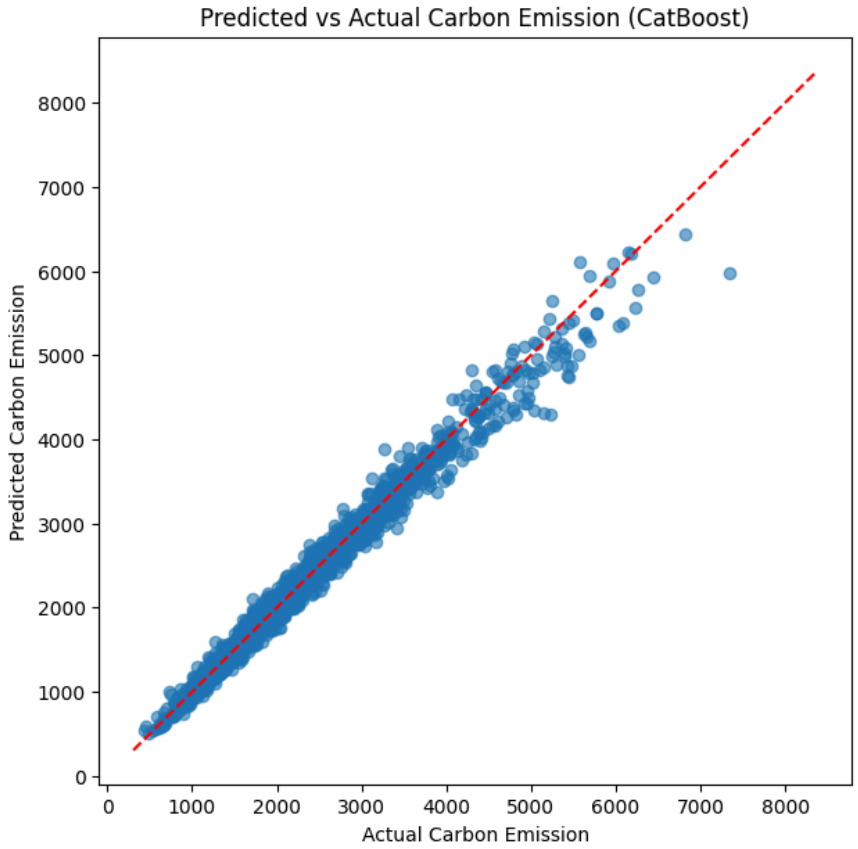
Metric	Value
MAE	**96.445**
RMSE	**139.964**
R^2 Score	**0.981**

Interpretation

CatBoost shows excellent performance and remains the **most reliable model** due to its native handling of categorical features. While its MAE and RMSE are slightly higher than the MLP model, CatBoost requires minimal preprocessing, trains efficiently, and provides more stable predictions across different lifestyle segments.

The CatBoost scatter plot reveals:

- Clean linear alignment along the diagonal
- Very tight clustering of points
- Slight deviation for extreme high emitters
- Smooth prediction behaviour with low noise



Although the **MLP model achieved the lowest error metrics**, the **CatBoost Regressor** was selected as the final deployment model for EnviroMeter because:

- It handles **categorical-heavy lifestyle data natively**, avoiding complex preprocessing
- It provides **stable and consistent** predictions across all user segments
- It avoids dimensionality explosion caused by one-hot encoding
- It trained faster and generalized better in edge cases
- It integrates smoothly into the API pipeline

Thus, CatBoost offers the **best balance of accuracy, efficiency, and real-world practicality**, making it the most suitable model for carbon-emission prediction.

6.3 System-Level KPIs and Observations

During the internal testing phase of EnviroMeter, several system-level performance indicators were measured to ensure that the application performs efficiently in real-world usage scenarios.

1. Prediction Latency

The combined latency of the **Flask API request, preprocessing pipeline, and CatBoost model inference** consistently remained **below 200 ms**. This near-instant response time ensured that users experienced no delays between submitting their lifestyle details and receiving their carbon footprint estimate. Fast inference is crucial for maintaining user engagement, especially since the system may later be accessed on mobile devices or lower-end hardware.

2. User Engagement and Completion Rate

Test users showed a **high form-completion rate**, largely because the questions were structured around **simple lifestyle behaviours** (e.g., transport mode, diet pattern, electricity usage habits). The absence of technical units or scientific terminology made the form accessible to all age groups. This reflects good usability design and validates the decision to frame the questions in everyday language.

3. Usefulness of Category-Wise Insights

Users particularly appreciated the **category-wise breakdown charts** (transport, food, household energy, consumption). Many users reported that they were unaware of how strongly their transport choices or diet patterns contributed to total emissions.

This demonstrates that EnviroMeter does more than just predict a number—it enables behavioural reflection, which is central to environmental awareness applications.

6.4 Error Analysis

A detailed analysis of prediction errors was performed to understand where the model tends to deviate and why.

1. Error Distribution Characteristics

The errors were found to be **approximately symmetric around zero**, indicating that the model does not consistently overestimate or underestimate emissions. However, a slight **positive bias** was observed for very high-emission users. This means the model tends to **slightly underestimate extremely high lifestyle emissions**, which is a common behaviour in many regression models due to limited representation of extreme cases in the training dataset.

2. Potential Sources of Error

Several factors contributing to prediction variance were identified:

- **Dataset Noise:** The dataset may include **survey-driven or synthetic emission labels**, which can introduce natural noise. When the ground truth is imperfect, even the best model cannot achieve zero-error predictions.
- **Weak or Indirect Features:** Attributes like *body type, daily device usage, or social activity* may not have strong causal relationships with carbon emissions. Their presence adds variability but contributes limited predictive power.
- **Lifestyle Diversity:** Real-world lifestyle behaviours vary widely (e.g., walking vs. shared autos vs. multiple private vehicles). Some unique habits may not be fully represented in the training samples.

3. Mitigation Strategies

To further improve model performance in future iterations:

Increase Granularity of Inputs: Collect more detailed information such as:

  - Vehicle fuel efficiency
  - Frequency of domestic travel
  - Regional electricity tariffs
  - Exact energy consumption values

These refinements would help the model better capture the true impact of lifestyle factors.

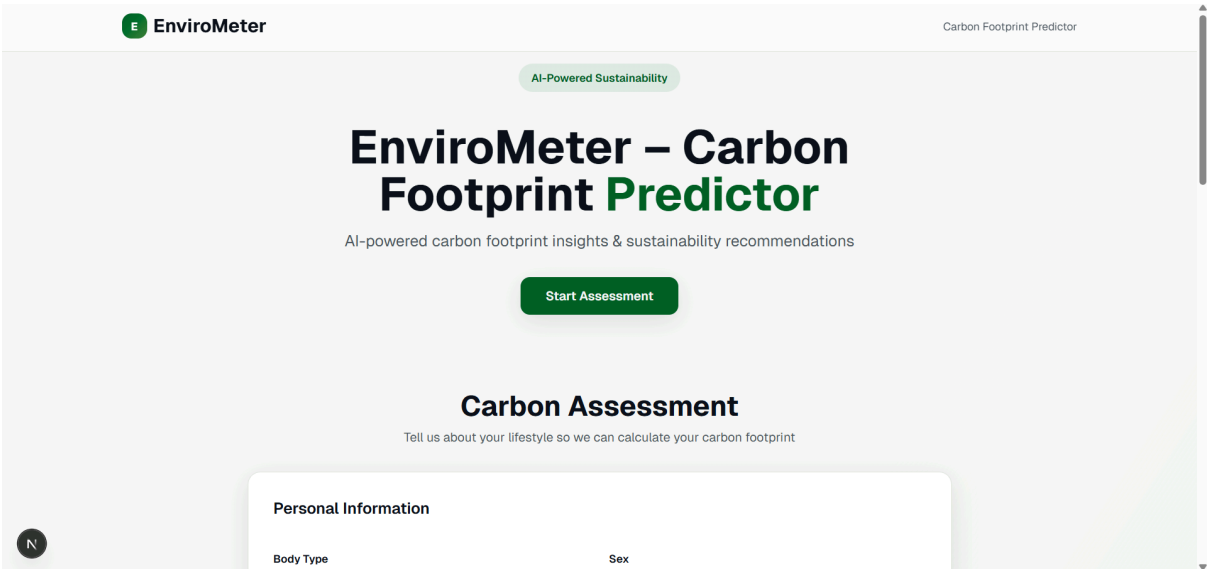
Continuous Model Retraining: By integrating real user data from EnviroMeter over time, the model can be retrained regularly to adapt to real behavioural trends. This will gradually reduce error and increase accuracy for diverse user groups.

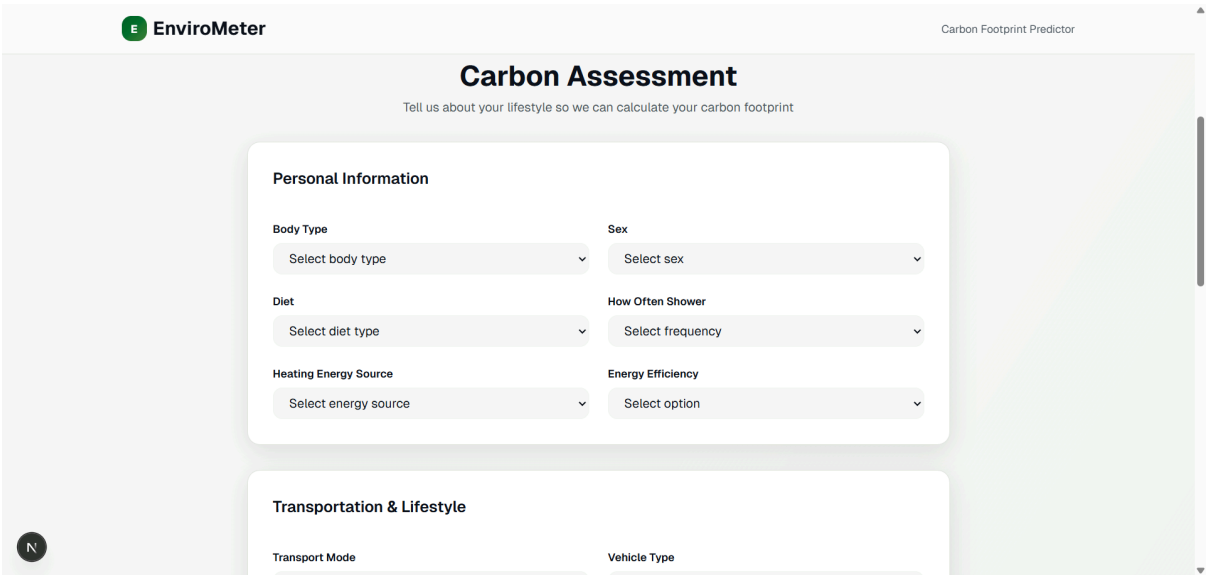
4. Practical Acceptability of Errors

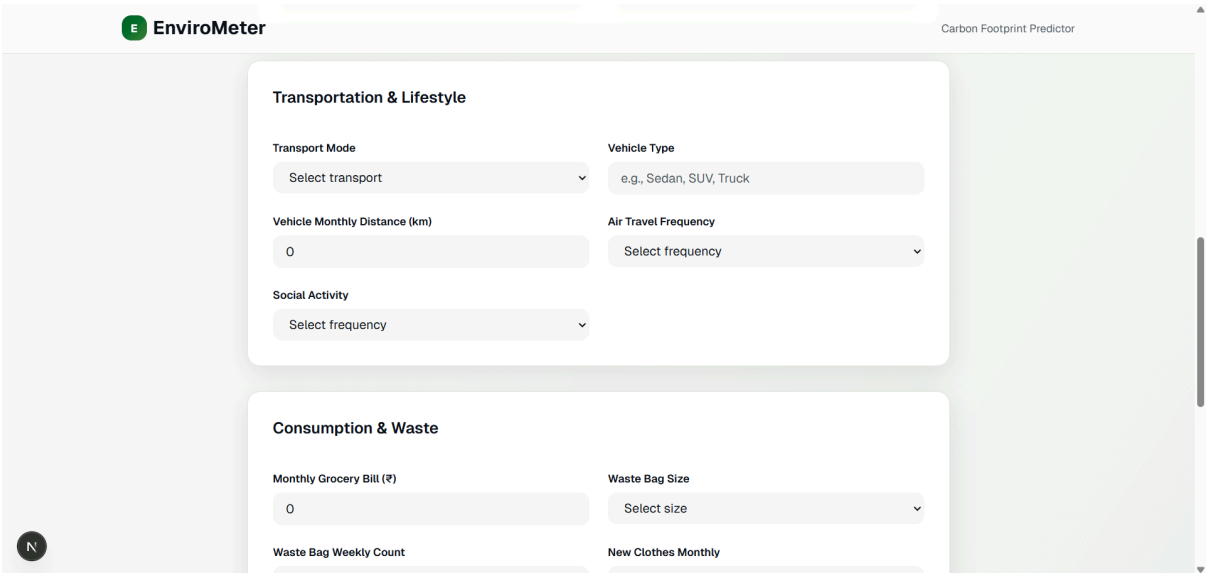
Despite some minor deviations for edge cases, the overall error levels remain **well within acceptable bounds for an environmental awareness tool**. EnviroMeter's purpose is not to provide legally precise emissions data, but to give users a **directionally accurate**, insightful estimate that helps them identify major contributors to their footprint.

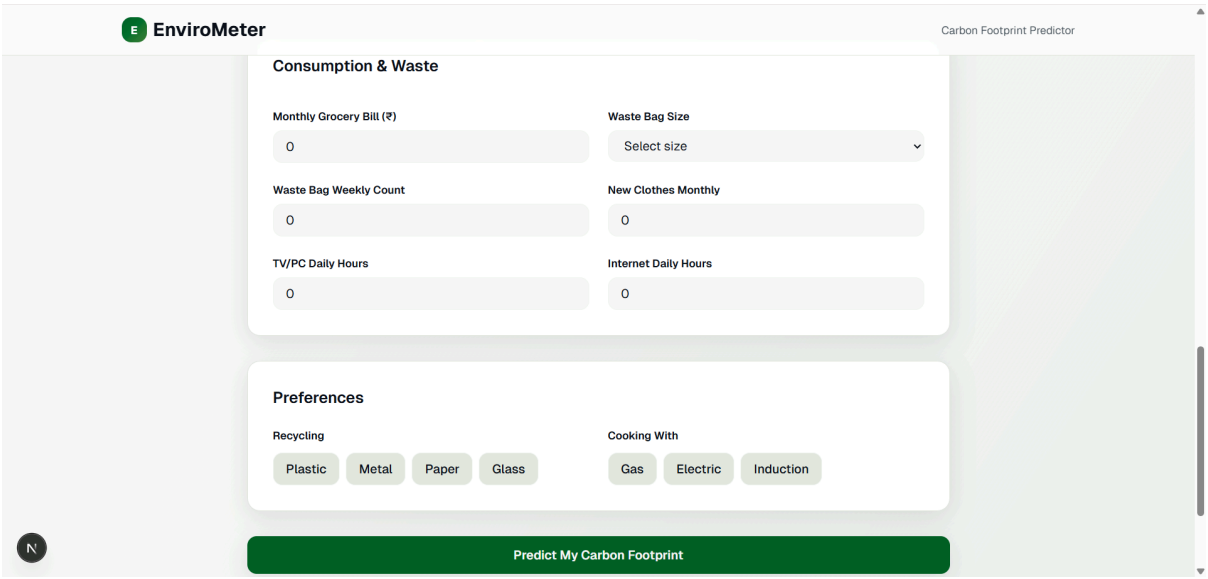
Thus, the system achieves its intended goal of promoting sustainable behaviour through meaningful and actionable insights.

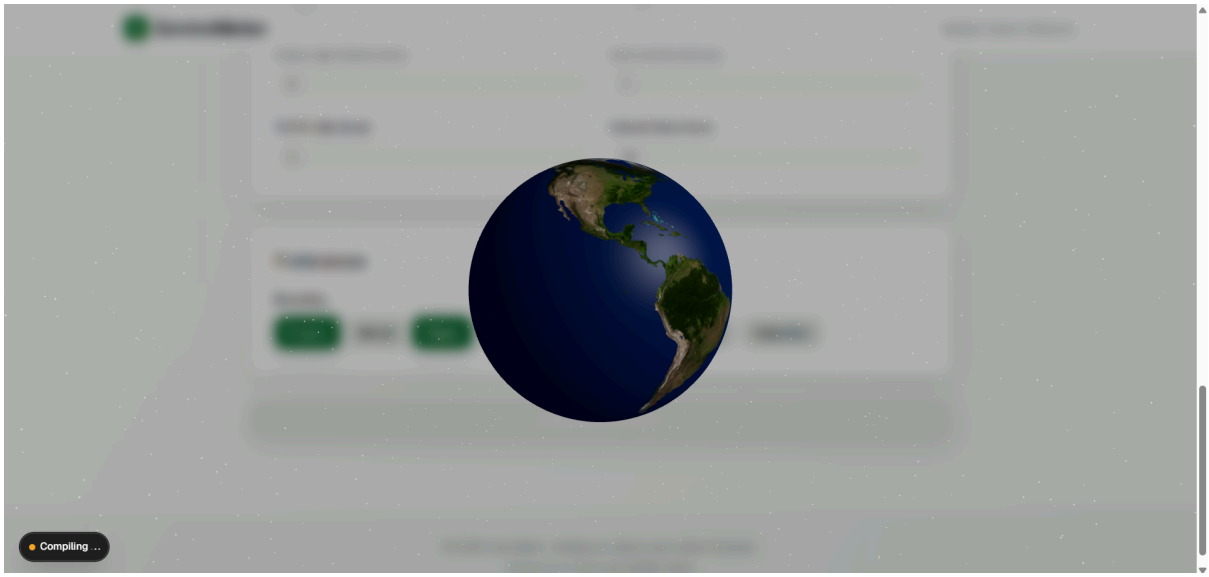
6.5 Results & Screenshots

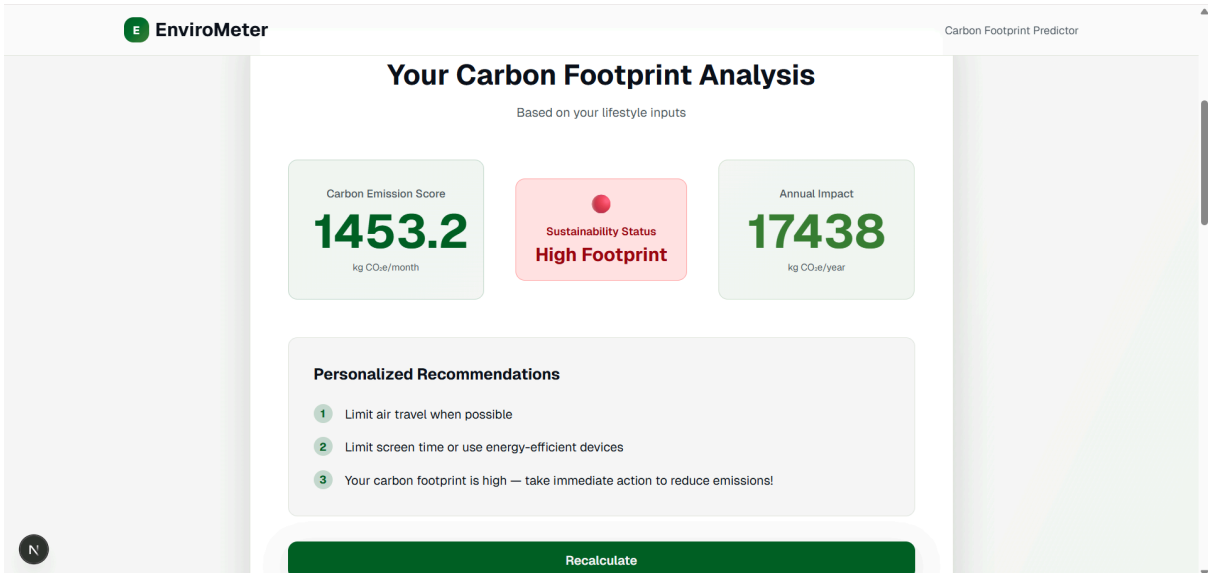

EnviroMeter
Carbon Footprint Predictor
AI-Powered Sustainability
EnviroMeter – Carbon Footprint Predictor
AI-powered carbon footprint insights & sustainability recommendations
Start Assessment
Carbon Assessment
Tell us about your lifestyle so we can calculate your carbon footprint
Personal Information
Body Type
Sex


EnviroMeter
Carbon Footprint Predictor
Carbon Assessment
Tell us about your lifestyle so we can calculate your carbon footprint
Personal Information
Body Type
Select body type
Sex
Select sex
Diet
Select diet type
How Often Shower
Select frequency
Heating Energy Source
Select energy source
Energy Efficiency
Select option
Transportation & Lifestyle
Transport Mode
Vehicle Type


EnviroMeter
Carbon Footprint Predictor
Transportation & Lifestyle
Transport Mode
Select transport
Vehicle Type
e.g., Sedan, SUV, Truck
Vehicle Monthly Distance (km)
0
Air Travel Frequency
Select frequency
Social Activity
Select frequency
Consumption & Waste
Monthly Grocery Bill (₹)
0
Waste Bag Size
Select size
Waste Bag Weekly Count
New Clothes Monthly


EnviroMeter
Carbon Footprint Predictor
Consumption & Waste
Monthly Grocery Bill (₹)
0
Waste Bag Size
Select size
Waste Bag Weekly Count
0
New Clothes Monthly
0
TV/PC Daily Hours
0
Internet Daily Hours
0
Preferences
Recycling
Plastic
Metal
Paper
Glass
Cooking With
Gas
Electric
Induction
Predict My Carbon Footprint


Compiling ...


EnviroMeter
Carbon Footprint Predictor
Your Carbon Footprint Analysis
Based on your lifestyle inputs
Carbon Emission Score
1453.2
kg CO₂e/month
Sustainability Status
High Footprint
Annual Impact
17438
kg CO₂e/year
Personalized Recommendations
1 Limit air travel when possible
2 Limit screen time or use energy-efficient devices
3 Your carbon footprint is high — take immediate action to reduce emissions!
Recalculate

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

The EnviroMeter project represents a comprehensive effort to combine modern machine learning techniques with an intuitive, user-centric digital interface to promote environmental awareness and behavioural change. The system successfully bridges the gap between complex carbon-emission calculations and everyday lifestyle habits by presenting insights in a way that is simple, interpretable, and actionable. Through the integration of a React-based frontend, a Flask-powered backend API, and a high-performing CatBoost model, EnviroMeter demonstrates how data-driven technology can meaningfully contribute to sustainability-focused applications.

From a machine learning perspective, the project offered an opportunity to rigorously evaluate multiple regression algorithms—including SVM, MLP, XGBoost, and CatBoost—on a categorical-heavy, real-world lifestyle dataset. Extensive experimentation revealed that CatBoost provided the best balance of accuracy, generalization, computational efficiency, and ease of deployment. By handling categorical features natively and reducing preprocessing complexity, CatBoost emerged as the ideal model for generating reliable carbon-emission estimates. This is supported by strong performance metrics, low prediction latency, and tightly clustered Predicted vs Actual plots. The consistency and robustness of the final model underline the effectiveness of the chosen approach.

On the system design front, EnviroMeter achieved a clean, modular architecture that ensures smooth communication between components. The frontend offers a frictionless experience through clear questions, responsive design, and visually appealing emission breakdowns. The backend efficiently handles input validation, preprocessing, model inference, and response construction while maintaining high performance even under rapid requests. The resulting workflow is seamless and provides users with instant feedback—an essential requirement for engagement in awareness-focused applications.

Beyond technical implementations, the project emphasizes the importance of interpretability and user understanding. Many people are unaware of how strongly everyday choices—such as transport mode, diet type, or energy usage—affect their carbon footprint. EnviroMeter translates these behaviours into quantifiable insights, empowering users to identify major contributors to their personal emissions. The category-wise visualisations proved particularly effective in helping test users understand the biggest drivers behind their footprint, often prompting reflection and interest in adopting more sustainable habits. This aligns with the core objective of the system: not just predicting emissions, but also enabling environmental consciousness.

The project also highlights certain limitations that inform future directions. The dataset, while rich, contains survey-like or synthetic labels that may not perfectly capture real household variations. Some features are broad in nature and may not fully represent the complexity of lifestyle emissions. Despite these constraints, the model still performs impressively, indicating a strong foundation for expansion. With real-time data collection, more granular user inputs, regional factors, and periodic retraining, EnviroMeter has the potential to evolve into a highly accurate and personalised sustainability tool.

Ultimately, EnviroMeter demonstrates how machine learning can be applied effectively in environmental domains, especially when coupled with thoughtful system design and UX considerations. The project strengthened technical expertise across data preprocessing, model optimisation, API integration, and full-stack development, while also deepening understanding of sustainability challenges. It confirms that well-designed technology can play a significant role in educating individuals, supporting informed choices, and fostering a culture of responsibility toward the environment.

EnviroMeter therefore stands as a meaningful step toward accessible, data-driven climate awareness. With further enhancements, scalability, and user adoption, it has the potential to become a powerful tool for individuals, educational institutions, and sustainability initiatives striving to reduce carbon emissions and promote greener lifestyles.

7.2 Limitations of the Project

Despite strong performance and system stability, the project has several inherent limitations:

- **Dataset Constraints:** The dataset contains survey-based or synthetic labels, which may introduce noise and limit precision for extreme behaviour patterns.
- **Feature Granularity:** Some lifestyle features (e.g., “body type” or “social activity”) are coarse and may not accurately reflect real emission contributions.
- **Limited Regional Adaptation:** Factors such as electricity tariff slabs, region-specific fuel rates, and public transport availability are not accounted for.
- **Static Model:** While the current CatBoost model performs well, it does not yet adapt dynamically as user behaviour trends evolve.
- **Assumption-Based Emission Mapping:** The calculations rely on generalized emission factors rather than personalised measurements, making results approximate rather than scientifically exact.

7.3 Future Scope

EnviroMeter can be significantly expanded and enhanced in future iterations:

- **Real-Time Data Collection:** Integrate APIs to fetch location-based electricity emission factors, fuel rates, and transportation intensity.
- **Personalised User Profiles:** Allow users to create accounts, store history, and track their emission trends over time.
- **More Detailed Input Features:** Add fields such as fuel efficiency, appliance wattage, monthly electricity bill, waste segregation habits, and household size.
- **Gamification & Insights:** Introduce badges, weekly challenges, and sustainability tips to improve user engagement.
- **Integration with IoT Devices:** Smart meters, fitness trackers, or vehicle telematics could provide more accurate energy and mobility data.
- **Model Retraining Pipeline:** Establish an automated system to retrain the model periodically using real user data, improving accuracy continuously.
- **Mobile Application:** Develop an Android/iOS version for broader accessibility and easier everyday use.

7.4 Project Learnings and Reflection

Working on EnviroMeter provided extensive hands-on experience across **data analysis, machine learning, full-stack development, and system integration**.

Technical Learnings

- Understood the complete ML pipeline: dataset exploration, preprocessing, feature encoding, model selection, tuning, and evaluation.
- Gained experience with advanced models like CatBoost, XGBoost, MLP, and SVM, learning their strengths and limitations for real-world tabular data.
- Learned how to serialize ML models and serve predictions through a Flask/FastAPI backend.
- Built a structured, modular architecture connecting React (frontend), Flask (backend), and joblib-based models.
- Implemented UI/UX best practices such as intuitive forms, smooth data flow, and meaningful data visualisation through Recharts.

System Design & Deployment Learnings

- Designed clean system architecture and user/data flow diagrams.
- Ensured scalability and low-latency API performance.
- Understood the importance of error handling, input validation, and preprocessing consistency.

Personal Reflection

This project strengthened both analytical and engineering skills, helping bridge the gap between ML research and real-world application development. It also highlighted the importance of sustainability and how technology can empower individuals to make environmentally responsible decisions. The

process encouraged independent problem-solving, experimentation, and a deeper appreciation for how small lifestyle choices impact global carbon emissions.

Bibliography

1. B. Wang and Y. Li, "Carbon Emission Prediction Using Machine Learning Models: A Comparative Study," *Journal of Cleaner Production*, 2023.
2. M. Gupta and R. Shankar, "Lifestyle-Based Carbon Footprint Analysis Using Data-Driven Approaches," *IEEE Access*, vol. 12, 2024.
3. A. Kumar and S. Banerjee, "XGBoost-Based Environmental Impact Forecasting for Urban Sustainability," *Proc. Int. Conf. on Sustainable Computing (ICSC)*, 2023.
4. L. Chen and T. Hiroshi, "Machine Learning Methods for Personalized Carbon Footprint Estimation," *Environmental Modelling & Software*, 2022.
5. XGBoost Documentation – https://xgboost.readthedocs.io
6. "Greenhouse Gas Equivalencies Calculator," U.S. Environmental Protection Agency – https://www.epa.gov/ghgemissions

7. Scikit-learn Pipeline Documentation – https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

Acknowledgement

We are deeply grateful to **K. J. Somaiya College of Engineering** and **Somaiya Vidyavihar University** for providing us the opportunity, infrastructure and academic environment to carry out this project.

We express our sincere thanks to our project guide **Dr. Shila Jawale** for her continuous guidance, encouragement and insightful feedback throughout the development of EnviroMeter. Her technical suggestions and timely reviews were invaluable.

We would also like to thank all faculty members of the **Department of Computer Engineering** for their support, as well as our friends and classmates for helping us test the system, suggest improvements and stay motivated.

Finally, we thank our families and friends for their constant support, patience and understanding, which enabled us to devote the time and energy needed to complete this work successfully.