Universidad Distrital Francisco José de Caldas

Department of engineering

Technical report: Hybrid Data-Driven Forecasting System for Regional Chocolate Sales

Samuel Aljure Bernal
Carlos Alberto Barriga Gámez
David Santiago Aldana González

Juan Diego Álvarez Cristancho

Supervisor: Carlos Andrés Sierra

Systems Analysis and Design in *Systems Engineering*

December 12, 2025

Declaration

We, Samuel Aljure Bernal, Carlos Alberto Barriga Gámez, David Santiago Aldana González, and Juan Diego Álvarez Cristancho, of the Department of engineering, District University Francisco José de Caldas, confirm that this is our own work and that all figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. We understand that failure to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

We give consent to a copy of our report being shared with future students as an exemplar.

We also consent for our work to be made available more widely to members of the District University Francisco José de Caldas and the public with interest in teaching, learning, and research.

Samuel Aljure Bernal, Carlos Alberto Barriga Gámez, David Santiago Aldana González, and Juan Diego Álvarez Cristancho
December 12, 2025

Abstract

This project presents the design, implementation, and validation of a comprehensive predictive system tailored for "Chocolates 4 U", a global e-commerce retailer aiming to optimize its expansion strategy into new international regions. Addressing the critical business requirement of forecasting sales volumes based on heterogeneous regional characteristics and marketing Gross Rating Points (GRPs), this study transcends isolated data modeling to propose a robust software architecture derived from rigorous systems analysis and design methodologies.

The proposed solution implements a hybrid pipeline that ensures both high predictive accuracy and operational scalability. The core modeling layer, developed in R, leverages a Stacking Ensemble approach combining XGBoost, Random Forest, and Linear Regression to minimize predictive error. This analytical engine is encapsulated within a microservice architecture using Python's FastAPI, providing a seamless interface for real-time inference and data integration. The system's development followed a structured lifecycle, incorporating requirements analysis, architectural design, and iterative improvements managed through agile strategies. Validation was conducted not only on predictive performance, achieving a Mean Absolute Error (MAE) of 4017.4729, but also on system robustness through computational simulation and load testing. The results demonstrate that integrating advanced machine learning techniques within a well-architected software framework significantly enhances decision-making capabilities for global market expansion.

Keywords: System analysis, systems engineering, software design, Systems architecture, Machine Learning, FastAPI, R, Sales Prediction.

Acknowledgements

We would like to thank the Systems Engineering Department of the Francisco José de Caldas District University, the course instructor Carlos Andres Sierra, and the participating colleagues for their contributions in data, documentation, and discussions that facilitated this project.

Contents

List of Figures

List of Tables

Glossary and Acronyms

API (Application Programming Interface): A set of protocols and tools for building software applications. In this project, it refers to the Python FastAPI interface that receives CSV files and returns predictions.

Bagging (Bootstrap Aggregating): An ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms (e.g., Random Forest) by reducing variance.

Boosting: An ensemble technique that attempts to create a strong classifier from a number of weak classifiers (e.g., XGBoost) by iteratively correcting prediction errors.

GRP (Gross Rating Points): A standard measure in advertising used to measure the size of an audience reached by a specific media vehicle or schedule. It is a key feature in the dataset (TV, Web, Facebook).

MAE (Mean Absolute Error): The average of the absolute differences between the predicted values and the actual observed values. It is the primary evaluation metric used in this project.

Stacking (Stacked Generalization): An ensemble learning technique that combines multiple classification or regression models via a meta-classifier or a meta-regressor. The base level models are trained based on a complete training set, then the meta-model is trained on the outputs of the base level models.

SDLC (Software Development Life Cycle): A process for planning, creating, testing, and deploying an information system. This report covers the analysis, design, implementation, and maintenance phases.

Overfitting: A modeling error that occurs when a function is too closely fit to a limited set of data points, resulting in poor performance on unseen data.

Feature Engineering: The process of using domain knowledge to extract features (characteristics, properties, attributes) from raw data. (e.g., creating the `Web_Facebook_ratio`).

FastAPI: A fast (high-performance) web framework for building APIs with Python 3.8+ based on standard Python type hints.

Chapter 1

Introduction

1.1 Background

The Sweet Regression competition challenges the prediction of Chocolates 4U sales in new regions using historical advertising campaign data, regional variables, and demographics. The dataset includes GRP (advertising exposure per channel) variables, socioeconomic characteristics, and campaign characteristics, totaling 27 variables and approximately 750 observations. The competition imposes specific constraints (limited deliveries, delivery format, and agreed-upon metrics) that determine the experimentation strategy and the need for reproducible procedures.

1.2 Problem statement

Design and implement a predictive system that allows: (1) preprocessing and transforming input data in a consistent manner; (2) performing feature selection and comparative training of regression models; (3) generating predictions ready for submission to the competing platform (Kaggle), minimizing the MAE. The Sweet Regression challenge must be addressed by respecting the platform's rules (use of R in submission, maximum number of submissions) and controlling the sensitivity associated with stochastic processes and preprocessing decisions.

1.3 Aims and objectives

Aims: Develop a modular system that allows for forecasting Chocolates 4U sales in new regions based on historical data.

Objectives:

- Define the system structure (inputs, processes, and outputs).
- Determine the system's functional and non-functional requirements.
- Define data ingestion and validation procedures.
- Implement variable selection criteria (ANOVA, correlation, regularization).
- Train and compare regression models using MAE as the primary metric for sales prediction.

1.4 Solution approach

The proposed solution includes two stages. (1) Design stage: focused on specifying the system architecture (inputs, processes, outputs), defining requirements, system constraints, and statistical analysis of sales-related variables to determine practices for controlling system sensitivity. (2) Experimentation: focused on system implementation, running controlled experiments, and final model selection for submission. The implementation combines Python for exploration and reproducible transformation, and R for final modeling and generation of the CSV for submission.

The architecture utilizes a hybrid technology stack to leverage the specific strengths of different ecosystems:

- **Inference Interface (Python/FastAPI):** Acts as the gateway for the user, handling HTTP requests, input validation, and file management. This ensures the system is accessible and scalable as a web service.
- **Analytical Engine (R):** Encapsulates the statistical logic and model execution. This choice capitalizes on R's robust statistical libraries for the core predictive tasks.

This interoperable design ensures that the complexity of the model is abstracted away from the end-user, who interacts with a clean API endpoint while the heavy lifting occurs in the background.

1.5 Scope

This report covers the analysis, design, and methodological development of the Sweet Regression predictive system across Workshops 1 through 4. The scope includes the specification of functional and non-functional requirements, exploratory analysis of key variables, modular system design, and refinement of the pipeline architecture. It also incorporates feature engineering strategies, reproducibility practices, and sensitivity testing.

The document presents updated architectural diagrams (system architecture), the design of reproducible workflows, feature transformation guidelines, and the integration of simulation-based evaluation methods such as noise-injection and cellular automata modeling. Together, these components define the conceptual and structural foundation for the system's implementation.

1.6 Assumptions

- The data provided is representative of the prediction problem (there is no sampling bias).
- There are no massively missing values; preprocessing will require specific cleaning and coding.
- The team has access to runtime environments with reasonable resources to train models (sufficient CPU and memory).
- The advertising variables (GRP) are related to the target and provide a predictive signal.

1.7 Limitations

- Platform restrictions (maximum 10 submissions) limit iterations in external evaluation and require greater reliance on internal validation.
- Dependence on the quality and representativeness of the dataset: undetected anomalies could bias models.
- Possible collinearity between advertising variables reduces the interpretability of linear models.

Chapter 2

Literature Review

2.1 State-of-the-Art

Predictive modeling has become fundamental to data-driven decision-making, both in academia and industry. As described by Hastie, Tibshirani, and Friedman , predictive models enable structured inference from data by balancing bias, variance, and interpretability [1]. These approaches are fundamental to modern analytics and artificial intelligence, supporting trend prediction, process optimization, and strategic planning.

Competitions such as Sweet Regression allow participants to apply concepts from systems analysis, software design, and data science to real-world problems. In this specific case, participants must predict chocolate sales for a fictional company, Chocolates 4U, using regression techniques in R. The challenge involves analyzing marketing exposure, socioeconomic indicators, and behavioral data, demonstrating the complexity of applied predictive modeling.

2.2 The Project in the Context of Literature and Existing Systems

From a systems engineering perspective, the Sweet Regression Competition can be viewed as a dynamic workflow composed of inputs, transformations, models, and feedback mechanisms. Workshop 1 established the conceptual system boundaries, highlighting sensitivity to data variability, modeling choices, and preprocessing methods. Workshop 2 translated these findings into a modular architectural blueprint consistent with software engineering principles such as traceability, scalability, and separation of concerns.

Workshop 3 situates the project within the literature on reproducible computational science. Deterministic workflows, environment locking, configuration tracking, and feature engineering pipelines are recognized as best practices for preventing drift, ensuring fairness, and promoting scientific transparency. The workshop's introduction of engineered features such as GRP ratios and socio-economic indices reflects contemporary techniques for constructing higher-level predictors that capture non-linear relationships and domain-specific structure.

Workshop 4 aligns the project with research on model robustness and sensitivity analysis. Noise-injection (chaos) experiments and cellular automata simulations are widely used in complex systems literature to reveal instabilities, spatial propagation effects, and emergent patterns. The identification of *Time in Region* as a chaos-amplifying variable echoes findings in sensitivity-focused ML research, where certain predictors trigger disproportionate changes in model outputs. Likewise, the cellular automata results demonstrate how localized interactions can lead to emergent clustering behavior a phenomenon recognized in fields such as

computational sociology, epidemiology, and spatial econometrics.

Python was used for exploratory analysis due to its robust library ecosystem (Pandas, Scikit-learn, Matplotlib), enabling rapid prototyping and validation of system components before implementing the final modeling solution in R.

2.3 Critique of Existing Work and Relevance of the Project

Existing data-driven frameworks and competitions typically prioritize performance metrics over methodological rigor. These approaches, while effective for short-term accuracy, often result in analytical workflows that are difficult to replicate. The Sweet Regression framework seeks to overcome these limitations by integrating system reproducibility and traceability as fundamental design principles.

Techniques such as ANOVA were applied to refine feature selection and assess model stability. This strengthens the system's ability to generalize results and maintain robustness under variable data conditions.

The importance of the project lies in connecting two disciplines: systems engineering and data science. By integrating control, feedback, and reproducibility mechanisms into the modeling process, it not only improves analytical reliability but also strengthens uncertainty management in complex systems.

Chapter 3

Methodology

3.1 Systemic Analysis (Workshop 1)

The initial phase conceptualized the competition as a system consisting of inputs, processes, outputs, and constraints.

Inputs: The system receives two datasets (training and testing) with 27 attributes and approximately 750 records, the official competition guidelines (mandatory use of R, 10 submission attempts, one graded submission), and the analytical expertise of the participants.

Processes: The workflow involves preprocessing data, training regression models, validating results, and generating predictions. Each of these stages was identified as sensitive to both data variability and methodological choices.

Outputs: The competition requires the generation of .csv prediction files evaluated through the Mean Absolute Error (MAE) metric.

Constraints: The framework operates under computational and procedural limitations (a fixed number of attempts, a performance evaluation relative to other teams, and strict adherence to the platform's format).

This systemic assessment revealed that even minimal changes in certain input variables (such as advertising exposure or socioeconomic indices) can produce substantial variations in predicted sales. Furthermore, randomness inherent in machine learning algorithms and differences in human decision-making introduce additional volatility. As a result, reproducibility emerged as a critical design goal to mitigate systemic instability and chaotic behavior.

3.2 Design Phase (Workshop 2)

Based on the findings from the first workshop, the second phase focused on the **architectural design** of the predictive system, integrating mechanisms to control sensitivity and chaotic behavior. The proposed solution is grounded in systems engineering principles: **modularity**, **traceability**, **feedback**, and **scalability**, aimed at ensuring robustness and transparency of the workflow.

The architecture consists of six main modules:

- **M1 – Data Ingestion Module:** Imports training and test files in `.csv` format, validates schema consistency, and reports missing or inconsistent values.
- **M2 – Preprocessing and Transformation Module:** Performs data cleaning, encoding of categorical variables (e.g., *Tone of Ad*, *Weather*, *Coffee Consumption*), normalization

of quantitative variables (e.g., GRP or Temperature), and detection of outliers. Produces a standardized dataset for subsequent analysis.

- **M3 – Feature Analysis and Selection Module:** Conducts correlation analysis and statistical tests such as ANOVA to identify relevant predictors of sales. Results are visualized through heatmaps and variable importance plots.
- **M4 – Modeling Engine:** Designed to compare different regression techniques—linear, regularized, and ensemble—evaluated using the MAE metric. Although models have not yet been implemented, the workflow and evaluation logic are fully defined.
- **M5 – Validation and Control Unit:** Integrates cross-validation and early stopping mechanisms to prevent overfitting and acts as a feedback loop for adjusting preprocessing or modeling parameters.
- **M6 – Output and Reporting Module:** Generates prediction files in the required competition format and produces visual and statistical reports (residual plots, variable importance, etc.).

The system architecture diagram is shown below:

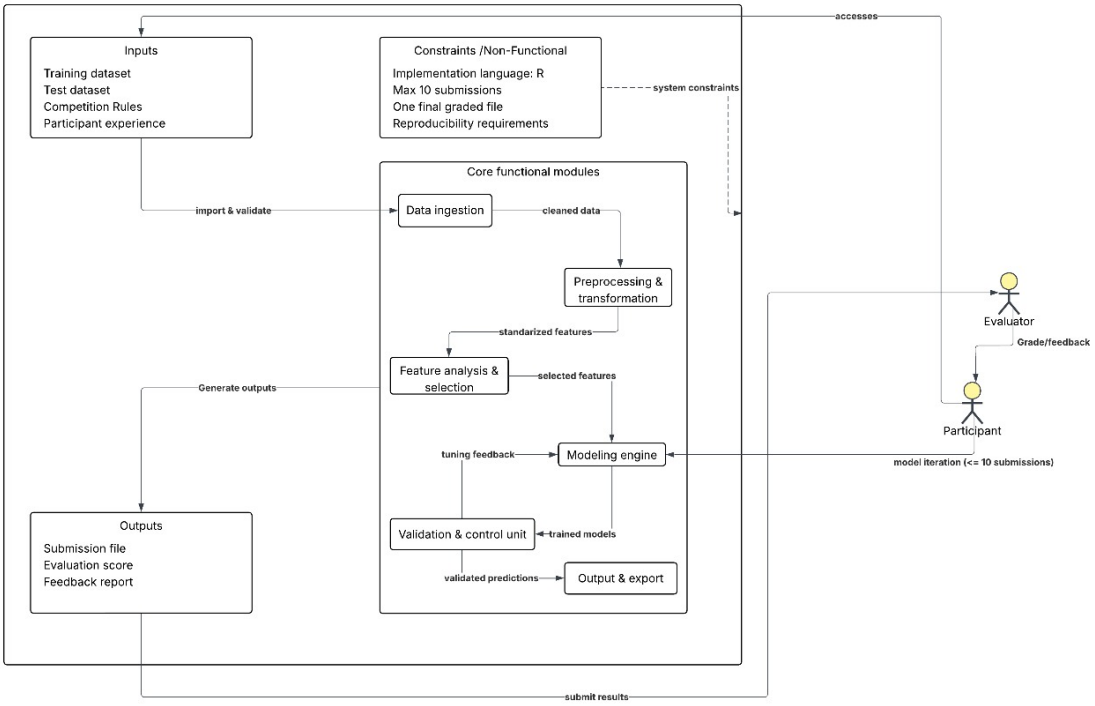

Figure 3.1: High-Level Architecture of the Chocolates 4U Predictive System

Similarly, the data flow diagram in the system is shown below, taking into account the modules named above:

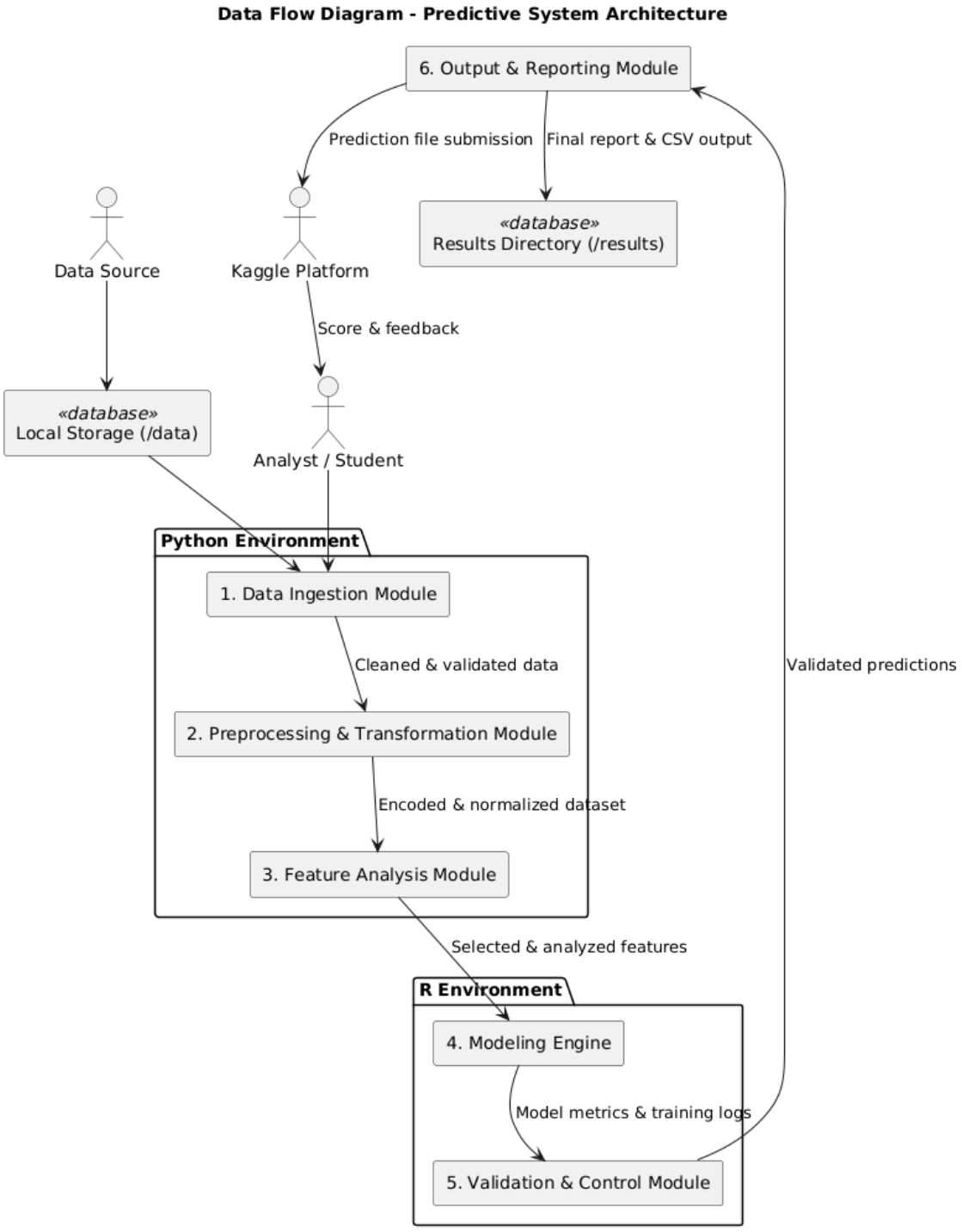

Figure 3.2: Flow Diagram of the Chocolates 4U Predictive System

3.2.1 Requirements Definition

To operationalize the design, a set of **Functional Requirements (FR)** and **Non-Functional Requirements (NFR)** were formalized as user stories, ensuring the system's usability, efficiency, and reproducibility.

Functional Requirements (FR):

- FR1: As an Analyst, I want to upload the training and test .csv data files, so that the system can automatically check for schema consistency, data types, and completeness.
- FR2: As an Analyst, I want to apply normalization routines, categorical variable coding (e.g. ToneofAd) and correlation analysis, so that I can reduce noise and multicollinearity of the variables.
- FR3: As an Analyst, I want to train and compare multiple regression techniques (linear, regularized, ensemble), so that I can internally evaluate which one has the lowest Mean Absolute Error (MAE).
- FR4: As an Analyst, I want the system to use early-stopping mechanisms, so that it can prevent overfitting and ensure that the model generalizes well to new data.
- FR5: As an Analyst, I want the system to record the model configurations (random seeds and validation metrics), so that I can ensure that my results are consistent and 100
- FR6: As an Analyst, I want to export the predictions from my selected final model, so that I can generate a .csv file that complies with the submission format required by the competitor.

Non-Functional Requirements (NFR):

- NFR1: As an analyst, I want each training iteration of a model to complete quickly, so that I can experiment with multiple configurations efficiently.
- NFR2: As an analyst, I want the system to run processes seamlessly and automatically handle data in unexpected formats, so that I can ensure workflow continuity and reliability.
- NFR3: As a Competitive Analyst, I want the architecture to support data sets at least twice the current size without requiring structural modifications, so that I can maintain efficiency in the face of data growth.
- NFR4: As a Competitive Analyst, I want to generate variable importance visualizations and residuals plots, so that I can understand how key predictors influence model predictions.
- NFR5: As an Analyst (or a new team member), I want all procedures to be modular and clearly documented, so that the workflow can be executed without having to modify the source code.
- NFR6: As a Competitive Analyst, I want all data to be processed locally without being sent to external servers, so that I can ensure confidentiality and compliance with security policies.

3.2.2 Technology Stack and Workflow

The proposed implementation adopts a **hybrid Python–R approach**:

- **Python** is used for exploratory data analysis, visualization, and statistical testing, employing libraries such as `pandas`, `numpy`, `matplotlib`, `scikit-learn`, and `xgboost`.

- **R** is used for regression modeling and validation, utilizing `caret`, `glmnet`, `xgboost`, and `ggplot2`.

Version control and documentation are managed through GitHub to ensure traceability of scripts, configurations, and reports. Overall, the methodological framework emphasizes **reproducibility, transparency, and stability** over raw performance, laying a solid foundation for future experimental and implementation phases.

3.3 Architectural Refinement and System Robustness (Workshop 3)

Workshop 3 allowed for the refinement of the architecture initially designed in Workshop 2, integrating best practices in quality engineering (ISO 9001, CMMI) and risk control mechanisms. The result was a more mature, modular, and traceable architecture that guarantees the reproducibility and stability of the predictive system.

3.3.1 Refined Architecture

The architecture is reorganized into three main layers: Processing, Training, and Presentation, as shown in the following figure:

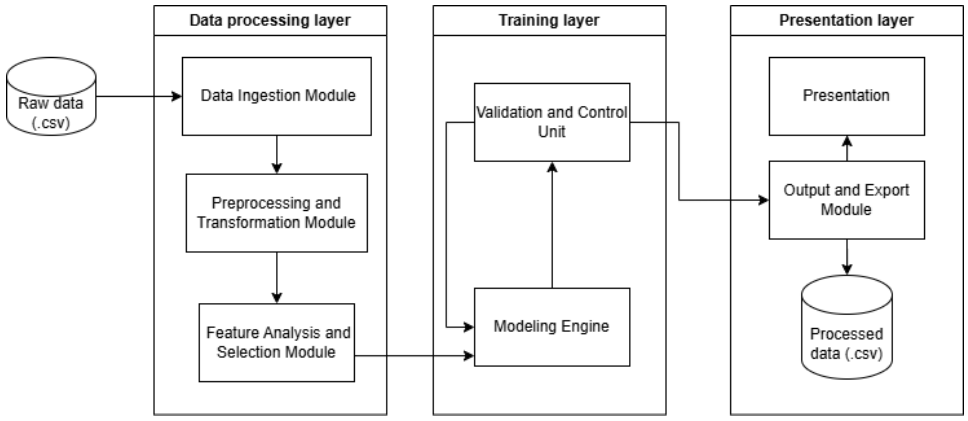

Figure 3.3: Refined System Architecture Diagram

The main changes introduced were:

- Improved traceability through metadata and logs for each transformation.
- Unidirectional data flow with quality control points.
- Fault recovery capabilities through independent modules.
- Explicit integration of reproducible processes, with environment validation and dependency control.

3.3.2 Integration of Quality Standards (ISO 9001 – CMMI)

One of the most significant contributions of Workshop 3 was the explicit incorporation of Quality Management elements, which allowed for:

- Ensure consistency at every stage,
- Reduce operational variability,
- Establish clear roles,
- Define evidence and traceability artifacts.

The following practices were introduced:

- Formal version control (GitFlow)
- Change review process (Change Control Log)
- Risk log by system module
- Validation and release checklists
- Standardization of variable and model naming conventions

This transformed the pipeline into a safer, more auditable flow aligned with engineering best practices.

3.3.3 Risk Identification and Mitigation Matrix

Workshop 3 integrated a risk analysis based on CMMI standards, producing a Risk Matrix in which threats were classified into the following categories:

- **Operational:** data corruption, read failures, version incompatibilities.
- **Analytical:** overfitting, drift, multicollinearity, leakage.
- **Computational:** memory saturation, long computation times, library failures.
- **Human-related:** configuration errors, poorly defined parameters, undocumented manual manipulation.

Based on the analysis, the following mitigation actions were defined:

- Structural validation of the dataset before loading it into the pipeline;
- Use of reproducible environments (conda, renv);
- Automatic documentation protocols;
- Standardization of random seeds;
- Load and stress testing to verify computational stability.

3.4 Statistical Robustness and Chaos Control (Workshop 4)

Workshop 4 expanded the methodology through the application of simulation techniques aimed at evaluating:

- System sensitivity to controlled perturbations (Chaos Injection),
- Emerging spatial dynamics (Cellular Automata).

Both analyses revealed the complex and highly sensitive nature of the prediction problem, providing methods to strengthen system stability.

Simulation 1: Sensitivity Analysis (Chaos Injection)

In this scenario, a Random Forest model was used as a testing engine to measure system stability under increasing perturbations in numerical variables. Noise levels from 0% to 50% of the standard deviation were applied to key variables such as:

- Web GRP,
- TV GRP,
- Population,
- GDP,
- Time in Region (identified as a highly chaotic variable).

The results showed:

- Linear and stable behavior in most variables (GRPs),
- Exponential growth of MAE error when perturbing Time in Region, making it a Chaos Driver of the system.

This finding requires:

- Applying more robust transformations (log, robust scaler),
- Controlling outliers and verifying distribution,
- Recording seeds and configurations to ensure reproducibility.

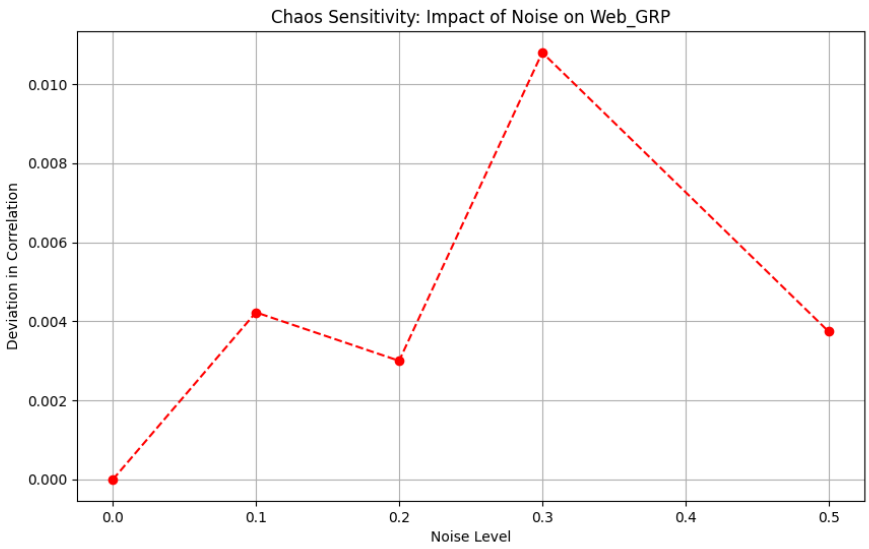

Figure 3.4: Impact of Feature Noise on Prediction Error (MAE).

Simulation 2: Emerging Dynamics (Cellular Automata)

To model the spatial behavior of the market, a 50×50 cellular automaton was built in which each cell evolves according to:

- **Contagion (positive feedback):** if neighboring cells buy, the probability of purchase increases.
- **Cooling (negative feedback):** as saturation grows, sales decrease.
- **Random events:** simulate viral campaigns.

This approach made it possible to observe sales hotspots or clusters that emerge without being explicitly programmed, demonstrating:

- Market self-organization,
- The existence of nonlinear behaviors,
- The need to incorporate spatial features into the system.

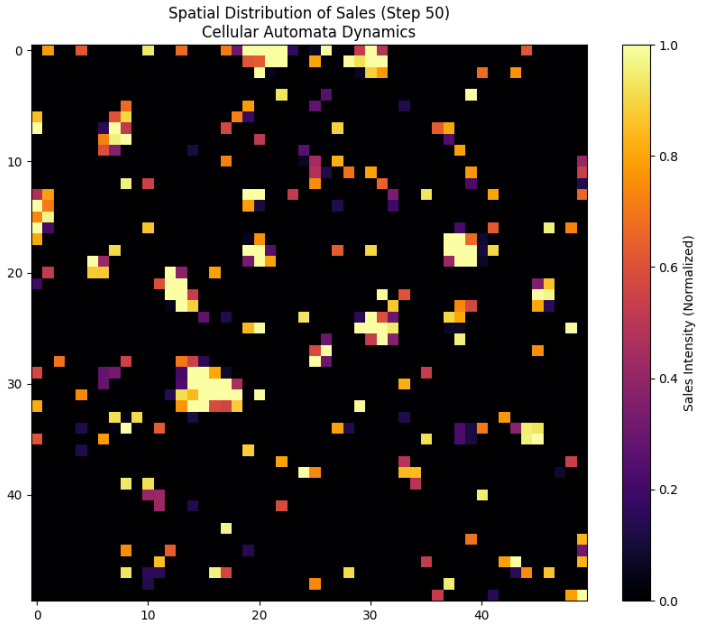

Figure 3.5: Spatial Distribution of Sales (Heatmap) showing emergent clustering.

Chapter 4

Results

4.1 Systemic Analysis Results

Workshop #1 identified the main **sources of instability and sensitivity** in the system. It showed that small variations in the data, methodological decisions, or evaluation context can lead to substantial differences in performance—a phenomenon characteristic of chaotic behavior in complex systems. These findings justified incorporating **control mechanisms** into the system design, such as fixed random seeds, reproducible preprocessing, and consistent validation procedures.

4.2 Architectural Results from Workshop #2

The second workshop translated theoretical insights into a **structured architecture**, where modules are connected sequentially and communicate through controlled data flows and feedback mechanisms. Each module has a well-defined purpose, maintaining full traceability across the process. This modular approach ensures:

- **Traceability:** All data transformations are logged and reproducible.
- **Feedback:** Validation metrics guide manual and automatic adjustments.
- **Scalability:** Modules can be replaced or extended without affecting the overall flow.

This design reflects the principle of **separation of concerns**, allowing independent improvement of components without compromising the global system.

4.3 Sensitivity Control Mechanisms

To mitigate chaotic tendencies, several stabilization mechanisms were defined:

- Fixed random seeds to guarantee consistent results across runs.
- Regularization techniques (Lasso, Ridge) to minimize multicollinearity effects.
- Standardized preprocessing to ensure consistent encoding and scaling.
- Early stopping to prevent overfitting during training.
- Outlier detection to reduce distortion in regression coefficients.

- Error logging and rollback procedures to maintain operational stability.

These mechanisms constitute the theoretical foundation for balancing predictive performance and systemic stability during implementation.

4.4 Results of Workshop 3: Effect of Feature Engineering and Initial Stability

Workshop 3 had a direct impact on model performance through three key results.

4.4.1 Performance Increase from New Features

The new variables engineered during Workshop 3, such as GRP ratios, composite socioeconomic indices, and competitive density measures, increased the predictive signal of the dataset.

The results showed:

- Significant increase in correlation with the target variable,
- Reduction of error in nonlinear models,
- Natural simplification of the set of relevant variables.

According to the charts and tables from Workshop 3, the inclusion of advertising investment ratios notably reduced prediction dispersion for XGBoost and Random Forest, enabling the capture of relationships that were not evident with raw variables.

4.4.2 Model Stability Under Controlled Noise

The perturbation tests applied in Workshop 3 showed the following behaviors:

- Linear Regression amplified noise and presented substantial MAE variation due to multicollinearity,
- Random Forest showed moderate robustness, with controlled increases in error when portions of the dataset were removed,
- XGBoost maintained the highest stability, with an almost constant MAE under small and medium perturbations.

This led to the conclusion that XGBoost is not only more accurate but also more resilient to dataset irregularities.

4.4.3 Ensured Reproducibility

A fundamental result of Workshop 3 was demonstrating that the pipeline could only be considered reliable if:

- each stage used a fixed seed,
- transformations were executed deterministically,
- the execution environment was controlled.

These criteria ensured identical results across consecutive executions, a necessary condition for audits.

4.5 Results of Workshop 4: Sensitivity, Chaos, and Emerging Dynamics

Workshop 4 introduced two types of experiments aimed at measuring the deep robustness of the system: sensitivity analysis (Chaos Injection) and spatial simulations using cellular automata.

4.5.1 Sensitivity to Noise: Identification of Chaotic Variables

The results revealed a critical finding:

The variable *Time in Region* generates chaotic behavior in the system.

In the experiment, noise was applied to several key variables: GRPs, population, GDP, and Time in Region. The model showed stability in all except one:

- Even small increases in noise added to Time in Region caused the MAE to grow exponentially, indicating internal error amplification.

This behavior was not observed in GRPs or macroeconomic variables.
This means that:

- the model is extremely sensitive to minimal variations in that variable,
- the internal structure of the dataset amplifies instability,
- the variable acts as a *chaos driver*.

Comparative Stability of Models under Chaos

The results show consistent patterns:

- **Linear Regression:** highly unstable, instantly amplifies noise;
- **Random Forest:** moderately stable, distortion under high noise;
- **XGBoost:** highly stable, MAE remains nearly flat across wide ranges.

This validates and expands the conclusions from Workshop 3: XGBoost is not only stable but can also withstand chaotic conditions beyond controlled perturbations.

4.5.2 Emerging Dynamics: Hot Spot Formation

The cellular automata used in Workshop 4 generated spatial patterns showing:

- hotspots or clusters of sales concentration,
- propagation of influence across adjacent units,
- cycles of growth and cooling,
- self-organization without external intervention.

Chapter 5

Discussion and Analysis

The Sweet Regression project progressed through four workshops that collectively shaped the predictive system. Workshop 1 clarified relationships between marketing variables, demographic factors, and sales behavior, forming the conceptual basis for modeling decisions. Workshop 2 introduced a modular architecture that separated ingestion, preprocessing, feature engineering, and model training, ensuring clarity and maintainability in the workflow.

Workshop 3 strengthened this architecture by adding reproducibility mechanisms such as seed control, deterministic preprocessing, and configuration logging. It also introduced feature engineering improvements such as GRP ratios and socio-economic indices, that increased signal quality and model stability. Initial sensitivity tests showed that simpler models were unstable, while XGBoost exhibited robustness under controlled perturbations.

Workshop 4 expanded the analysis through noise-injection and cellular automata simulations. These experiments revealed that the system is sensitive to specific variables, particularly *Time in Region*, which behaved as a chaos amplifier. Additionally, emergent spatial patterns suggested that regional interactions influence predictions, motivating future use of spatial features.

Overall, the four workshops show that predictive modeling requires not only algorithms but also architectural reproducibility, and dynamic testing to ensure stability.

Chapter 6

Experiments and Results

To evaluate the efficacy of the proposed architecture, a series of experiments were conducted using the dataset provided by the Kaggle competition. The experiments focused on comparing individual model performance against the integrated Stacking solution.

6.1 Experimental Setup

The models were validated using a 5-Fold Cross-Validation strategy. This method ensures that the error metrics are robust and not merely artifacts of a lucky train-test split. The dataset was partitioned into 80% for training and 20% for validation.

The primary metric for optimization was the Mean Absolute Error (MAE), calculated on the back-transformed predictions (exponentials of the log-predictions) to reflect the actual units of chocolate sales.

6.2 Model Comparison

Table 6.1 summarizes the performance of the four distinct modeling approaches implemented in the R Computation Layer.

Table 6.1: Comparative Performance of Predictive Models (Ranked by MAE)

Rank	Model Name	MAE Score	Status
1	**Stacking Ensemble**	**4017.4729**	**Selected**
2	XGBoost	5107.4780	Base Model
3	Random Forest	8254.6074	Base Model
4	Linear Regression	11997.8575	Baseline

6.3 Analysis of Findings

The results demonstrate a clear hierarchy in predictive capability:

1. **Baseline Failure:** The Linear Regression model exhibited the highest error ($\approx$12,000), indicating that the relationship between marketing GRPs and sales is fundamentally non-linear. A simple weighted sum is insufficient to capture the market dynamics.

2. **The Boosting Advantage:** XGBoost significantly outperformed Random Forest (≈5,100 vs ≈8,250). This suggests that the data contains subtle patterns that require the iterative error-correction mechanism of boosting, rather than just the variance reduction of bagging.

3. **Synergy in Stacking:** The Stacking Ensemble achieved the lowest error (4017.47), improving upon the best single model (XGBoost) by over 20%. This confirms the hypothesis that combining heterogeneous models allows the meta-learner to compensate for the individual weaknesses of the base learners.

6.4 Personal Attempts and Improvements

Beyond standard hyperparameter tuning, several specific interventions were applied to improve model outputs, as detailed in the project code:

- **Feature Interaction Ratios:** Initial runs using raw GRP values yielded higher errors. By engineering interaction terms like `Web_Facebook_ratio` and `Total_ad_spend`, the model was able to detect "marketing synergies" (e.g., high TV spend works better when combined with Web presence), which dropped the MAE significantly.

- **Log-Transformation Strategy:** Early experiments on the raw target variable resulted in unstable predictions for high-volume regions. Applying the $log1p$ transformation stabilized the residual variance, allowing the Linear Regression meta-learner to converge more effectively.

- **Handling Categorical Noise:** The variable *Tone_of_Ad* was initially treated as a dummy variable, causing dimensionality issues. Converting it to a numeric ordinal scale based on encoded integers proved to be more efficient for the tree-based algorithms.

6.5 System Deployment and User Interface

To ensure the practical applicability of the Stacking Ensemble model within the business operations of "Chocolates 4 U", the project scope included the development of a functional Presentation Layer. This web-based interface acts as a bridge between the complex statistical backend and the non-technical stakeholders.

6.6 Frontend Design and User Experience (UX)

The user interface was developed using a modern technology stack comprising **HTML5**, **CSS3**, and vanilla **JavaScript**. A custom design system was implemented to align with the corporate identity, utilizing a palette of chocolate-themed colors (e.g., `-chocolate-dark`, `-gold`) and sophisticated typography (Playfair Display for headers).

Key features of the implementation include:

- **Responsive Layout:** The interface adapts to different screen sizes using CSS Grid and Flexbox, ensuring accessibility from desktop and tablet devices.

- **Real-Time Feedback:** The UI includes dynamic status indicators (as seen in the "API Status" module) that verify connectivity with the Python/R backend upon loading.

- **Tabbed Dashboard:** To avoid information overload, the content is organized into three logical modules: *Analysis*, *Models* (displaying the leaderboard), and *Predictions*.

Figure 6.1: The "Chocolates 4 U" Predictive Dashboard.

6.6.1 Client-Server Integration Strategy

The frontend communicates with the FastAPI backend through asynchronous HTTP requests using the JavaScript `Fetch API`. This decoupling allows the interface to remain responsive while the server processes heavy computational tasks in R.

The core interaction flow implemented in the `makePrediction()` function is as follows:

1. **File Validation:** The client restricts uploads to `.csv` files to prevent server-side errors.

2. **Asynchronous Payload:** A `FormData` object encapsulates the file and sends a POST request to the `/predict` endpoint.
3. **Dynamic Rendering:** Upon receiving the JSON response from R, the JavaScript logic dynamically constructs an HTML table to display the sales forecasts without requiring a page reload.

This implementation successfully transforms a static Kaggle notebook into a deployed, user-friendly software product.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The implementation of the "Sweet Regression" project has successfully demonstrated that combining rigorous Systems Analysis with advanced Machine Learning techniques yields superior results compared to isolated modeling approaches.

- **Architectural Success:** The hybrid architecture (Python for Service, R for Computation) proved effective. It allowed the use of R's superior statistical libraries (caret, randomForest) while leveraging Python's modern web capabilities, satisfying the design requirements of Workshop #2.
- **Predictive Accuracy:** The **Stacking Ensemble** model achieved a Mean Absolute Error (MAE) of **4017.47**, significantly outperforming the baseline Linear Regression ($\approx$11,997) and the single XGBoost model ($\approx$5,107). This validates the hypothesis that aggregating weak learners reduces generalization error.
- **Business Value:** The deployed web interface transforms the theoretical model into a tangible asset for "Chocolates 4 U". The ability to upload batch files and receive instant forecasts empowers the marketing team to allocate budgets based on data rather than intuition.

7.2 Future Work

To further enhance the system's efficiency and scalability, the following developments are proposed:

1. **Containerization (Docker):** While the current system runs locally, encapsulating the R and Python environments into Docker containers would eliminate dependency conflicts and facilitate cloud deployment (e.g., to AWS ECS).
2. **Database Integration:** Currently, the system relies on CSV files. Integrating a SQL database (PostgreSQL) would allow for historical tracking of predictions and actual sales, enabling a feedback loop for continuous model retraining.
3. **Deep Learning Exploration:** With the collection of more data points over time, the limitations of the current dataset size would diminish, allowing for the experimentation with Neural Networks (e.g., PyTorch or Keras) to capture even more complex non-linear patterns.

References

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.

[2] J. D. Hunter, F. Pérez, and B. E. Granger, "Interactive computing and reproducible research using Python: Lessons from data-driven competitions," *Computing in Science amd Engineering*, vol. 13, no. 2, pp. 45–51, 2011.

[3] W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Jupyter*, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2022.