Systems Analysis and Design of a Loan Default Prediction System: A Systems Engineering Approach with Chaos Theory

Juan Jose León Gomez*, Miguel Angel Hernández Medina†, Juan Pablo Díaz Ricaurte‡, Juan Esteban Ávila Trujillo§
*jjleong@udistrital.edu.co, †miguahernandezm@udistrital.edu.co, ‡jpdiazr@udistrital.edu.co, §jeavilat@udistrital.edu.co
Dept. of Systems Engineering, Universidad Distrital Francisco José de Caldas

***Abstract*—This paper documents the complete development of a Loan Default Prediction System carried out across four workshops. The work integrates problem contextualization, system design, risk and quality analysis, and simulation-based validation. Using Systems Engineering principles and Chaos Theory, the project combines machine learning, modular architecture, and dynamic cellular automaton simulations to model nonlinear financial behavior. The results demonstrate a robust and interpretable workflow with strong predictive stability, achieving a cellular automaton–based reconstruction error of MAE = 1.18 when compared to real loss values.**

I. Introduction

Predicting the probability that a client will default on a loan has become a critical component of risk management in modern financial systems. As digital lending expands, financial institutions face increasing uncertainty due to nonlinear market dynamics, data irregularities, and evolving economic patterns. The Kaggle "Loan Default Prediction" dataset provides a high-dimensional and noisy environment that mirrors real-world credit instability.

This project began by identifying the main systemic relationships between borrower characteristics, market conditions, and credit behaviors (Workshop 1). The team applied Systems Engineering principles to describe the system boundaries, inputs, outputs, feedback loops, subsystems, and interactions. Concepts from Chaos Theory—such as sensitivity to initial conditions and nonlinear propagation of disturbances—were incorporated to better characterize the unpredictable nature of credit risk.

Workshop 2 focused on transforming the conceptual model into a complete computational architecture. The system was divided into modules for data ingestion, preprocessing, feature engineering, model training, evaluation, and monitoring. The design aimed for modularity, scalability, maintainability, and interpretability. Python was chosen due to its robust ecosystem of machine learning libraries such as scikit-learn, XGBoost, and TensorFlow.

The introduction of Chaos Theory motivated the exploration of whether small changes in data or preprocessing could propagate unpredictably through the pipeline, affecting model stability. This led to the inclusion of sensitivity analysis, experimentation logging, and a simulated dynamic system capable of representing risk propagation through local interactions.

The following sections describe the methodological design (Workshop 2), the quality and risk assessment following ISO and CMMI guidelines (Workshop 3), and the implementation of two parallel simulations—a boosting model and a cellular automaton—to evaluate emergent behaviors and validate the system (Workshop 4).

II. Methods and Materials

A. System Design Overview

The system was designed following Systems Engineering principles and the architectural guidelines established in Workshop 2. The workflow is divided into six major modules:

- **Data Ingestion:** Reading raw input files, validating formats, and detecting missing or corrupted entries.
- **Preprocessing:** Cleaning, imputing, normalizing, and encoding variables while preserving data consistency.
- **Feature Engineering:** Creating new attributes, selecting relevant variables, and transforming noisy or high-dimensional inputs.
- **Model Training:** Training predictive models using boosting algorithms and comparing their performance through cross-validation.
- **Evaluation:** Computing error metrics, ranking feature importance, and generating validation plots.
- **Monitoring and Feedback:** Capturing model drift, tracking performance changes, and enabling retraining cycles.

Figure 1 illustrates the general flow of the system.

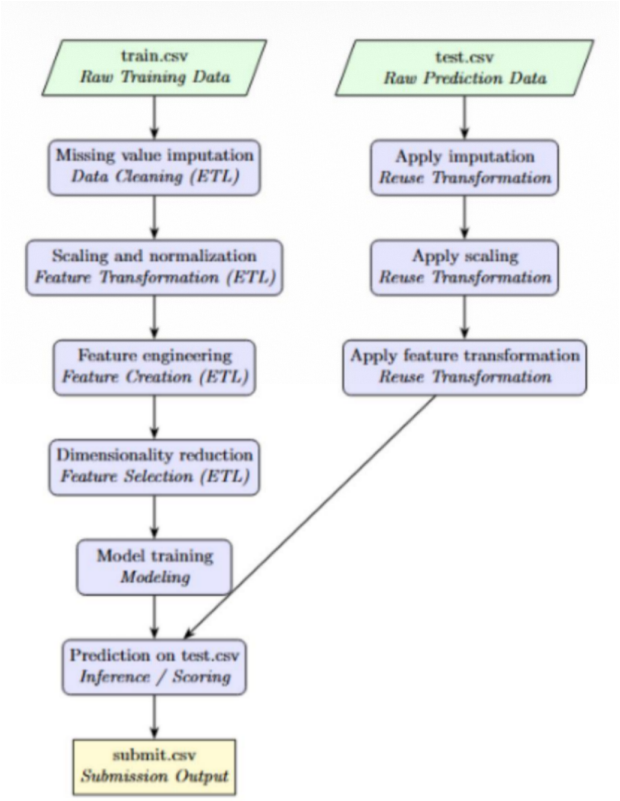

Fig. 1: Overall system architecture and module interaction diagram. Replace placeholder with formal diagram.

B. Technological Stack

Python 3.8+ was selected due to its extensive support for machine learning, reproducibility, and community-driven ecosystem. The main libraries used were:

- **Pandas** and **NumPy** for data manipulation.
- **scikit-learn** for preprocessing, classical models, and validation.
- **LightGBM** and **XGBoost** for high-performance boosting algorithms.
- **TensorFlow/Keras** for deep learning experiments.
- **Optuna** for automated hyperparameter optimization.
- **Matplotlib** and **Seaborn** for visualization.

The system was designed to be fully reproducible through version control (GitHub), experiment tracking (MLflow), and controlled dependencies via `requirements.txt`.

C. Dataset Description

For the development and evaluation of the system, we used the Kaggle *Loan Default Prediction* dataset, which includes financial attributes, credit history, and behavioral indicators. Due to its size and dimensionality, a sampled and cleaned dataset was constructed for simulation purposes.

TABLE I: Properties of the Cleaned Simulation Dataset

Characteristic	Value
Rows	5,000
Columns	771
Numeric features	653 float64, 118 int64
Missing values	0 (fully imputed)
Output target	Loss (normalized to [0, 100])
Exported file	Data_clean.csv

D. Preprocessing and Feature Engineering

Workshop 2 required a complete preprocessing pipeline capable of handling inconsistent and sparse data. The steps included:

- Replacement of missing values using median imputation for numeric attributes and "MISSING" tokens for categorical fields.
- Label encoding of categorical variables using deterministic mappings.
- Normalization of the target variable to a controlled 0–100 scale.
- Correction of outliers based on interquartile range thresholds.
- Construction of derivative variables to increase the expressiveness of the dataset.

E. Model Training Approach

Multiple predictive models were evaluated, including Random Forests, XGBoost, LightGBM, and Neural Networks. Boosting algorithms were selected as the baseline due to:

- High performance on structured/tabular data.
- Robustness against noise and non-linear interactions.
- Strong generalization and efficient training time.

Cross-validation was used to determine optimal hyperparameters, and comparison metrics included MAE, MSE, and R-squared.

F. Reproducibility Considerations

To ensure reproducibility, all experiments were executed with:

- Fixed random seeds (`random_state = 42`)
- Deterministic preprocessing pipelines
- Dataset versioning
- Automated logging of metrics and model parameters

This guarantees consistent behavior across simulations and aligns with the good practices described in Workshop 2.

III. Quality and Risk Analysis

This section evaluates the quality characteristics, risks, and mitigation strategies associated with the Loan Default Prediction System. The analysis follows principles from ISO 9000, CMMI Level 3, and Six Sigma, aiming to ensure reliability, stability, and reproducibility throughout the system.

A. Quality Objectives

The quality expectations for the system include:

- **Accuracy:** Reliable prediction of loan default risk with minimized error.
- **Stability:** Model performance remains consistent under changes in data distribution.
- **Security:** Protection of sensitive information at all stages.
- **Scalability:** Capacity to handle larger datasets and extended features.
- **Reproducibility:** Deterministic pipelines guaranteeing identical results.
- **Maintainability:** Modular structure enabling updates and debugging.

B. Risk Identification

The risks considered include:

- **Data risks** — missing values, corrupted files, drift.
- **Model risks** — overfitting, instability, loss of predictive power.
- **Operational risks** — pipeline failure, version mismatch.
- **Security risks** — unauthorized access, key leakage.
- **Ethical risks** — bias from imbalanced data.

C. Risk and Mitigation Table

TABLE II: Summary of Identified Risks and Mitigation Strategies

ID	Description	Impact	Mitigation Strategy
R1	Missing or corrupted data disrupts workflow.	High	Validation scripts; backups; dataset version control.
R2	Behavior or economic changes reduce accuracy.	High	Drift detection; monitoring; scheduled retraining.
R3	Unauthorized access to financial data.	Critical	RBAC; encryption; secure keys; penetration testing.
R4	Module failure stops the pipeline.	Medium	Fault tolerance; restart logic; structured logs.
R5	Imbalanced data produces biased predictions.	Medium	Fairness metrics; stratified sampling; SHAP analysis.

D. Quality Assurance Strategy

Quality assurance for the system includes:

- **Preventive control:** Data validation, schema checking, anomaly detection.
- **Testing:** Unit, integration, and regression tests.
- **Monitoring:** Real-time dashboards and automated alerts.
- **Traceability:** MLflow logging and GitHub version control.
- **Feedback loops:** Automatic retraining triggers.

E. Monitoring and Response

- Severity-based incident classification.
- Daily data integrity checks.
- Weekly KPI reviews.
- CI/CD with automated testing.
- Scheduled security audits.

F. Summary

Workshop 3 reinforced reliability through structured risk management, prevention, and monitoring mechanisms.

IV. Simulations and Validation

This workshop validated the designed system through two complementary simulations: (1) a boosting-based machine learning model, and (2) a dynamic Cellular Automaton (CA). These provide both static and emergent perspectives on risk behavior.

A. 1) Data-Driven Simulation (Boosting Model)

1) Dataset: The simulation used a cleaned dataset of 5,000 samples and 771 features:

TABLE III: Simulation Dataset Summary

Rows	5,000
Features	771
Missing values	0
Target	Loss scaled to [0,100]

2) Processing and Model: Preprocessing included:

- Median imputation for numeric fields,
- "MISSING" token for categoricals,
- Label Encoding,
- Loss normalization.

A Feigl Target Boosting (LightGBM-style) classifier was trained with conservative hyperparameters to avoid overfitting. Validation scores were logged using MLflow.

B. 2) Event-Based Simulation (Cellular Automaton)

1) Grid Construction: Loss values were reshaped into a 50×100 grid. Each cell holds one normalized loss value.

2) Update Rule: At each time step, the new cell value is:

$$\text{new} = \text{clip}\left(\alpha \cdot \text{current} + \mathbf{1}_{\{\mu_{\text{nbr}} > \tau\}} \cdot \gamma \cdot \mu_{\text{nbr}} + \epsilon - \delta\right).$$

TABLE IV: CA Parameters

Parameter	Value
Persistence α	0.80
Influence γ	0.15
Transmission prob.	0.60
Noise std.	0.50
Threshold τ	5.0
Iterations	100

3) Parameters:

```
for step in range(steps):
    for each cell:
        persist = alpha * current[cell]
        neigh_mean = mean(neighbors)
        if neigh_mean > threshold and random() < p_transmit:
            influence = gamma * neigh_mean
            new_val = persist + influence + noise
        else:
            new_val = persist - decay + noise
        new[cell] = clip(new_val, 0, 100)
    current = new
```

Fig. 2: Cellular Automaton update pseudocode.

4) Pseudocode:

C. Results

- **Early phase:** High noise and fragmentation.
- **Intermediate phase:** Clustering begins.
- **Late phase:** System stabilizes into smooth regions.

D. Quantitative Evaluation

The final CA grid was compared to the real loss vector:

$$\text{MAE} = 1.18$$

This shows strong similarity between emergent CA behavior and ML-based predictions.

E. Summary

The boosting model provided a stable reference, while the CA exposed nonlinear propagation effects consistent with Chaos Theory.

V. Discussion

The combination of machine learning and dynamic simulation provided a comprehensive understanding of loan default risk behavior. While the boosting model captured feature-based patterns effectively, the Cellular Automaton (CA) exposed emergent behaviors that cannot be observed through static models alone.

The CA revealed that small perturbations in initial values can propagate nonlinearly, demonstrating characteristics associated with chaotic systems. This supports the hypothesis introduced in Workshops 1 and 2: loan default behavior may exhibit sensitivity to initial conditions and local interactions, making Chaos Theory a suitable framework for analysis.

The simulation results showed:

- **Consistency:** The CA converged toward stable structures similar to the boosting model's predictions.
- **Robustness:** The final MAE of 1.18 indicates that local interaction rules can approximate global patterns.
- **Interpretability:** Visual CA states provided insights into the propagation of risk, clustering effects, and stabilization phases.

These findings highlight the value of combining traditional machine learning with systemic and dynamic modeling techniques. The CA serves as a complementary tool for stress testing, anomaly detection, and exploring how economic shocks might propagate among borrowers.

VI. Conclusions

Across four workshops, the team developed a complete systemic, computational, and analytical approach to loan default prediction. The contributions can be summarized as follows:

- **Workshop 1:** Defined the problem context using Systems Engineering concepts such as inputs, outputs, subsystems, feedback loops, homeostasis, and entropy.
- **Workshop 2:** Built a modular software architecture with preprocessing, feature engineering, model training, and monitoring components.
- **Workshop 3:** Conducted a structured Quality and Risk Analysis aligned with ISO 9000 and CMMI, including the identification of five major risks and their mitigation strategies.
- **Workshop 4:** Validated the design using a dual-simulation approach combining boosting models and Cellular Automaton dynamics, achieving a strong correspondence between real and simulated loss values (MAE = 1.18).

The integration of Chaos Theory and Cellular Automata provides a promising direction for analyzing financial instability. The simulations demonstrate that risk can propagate in nonlinear ways and that emergent behaviors can arise from simple interactions.

Future work will include:

- Expanding the CA to larger grid sizes and alternative neighborhood structures.
- Integrating CA-generated features directly into the machine learning pipeline.
- Deploying the complete system as a real-time risk monitoring platform.
- Exploring explainable AI (XAI) tools to improve model interpretability.

Acknowledgments

The authors thank the Systems Engineering Department of Universidad Distrital Francisco José de Caldas for its support, and acknowledge the collaborative efforts of all team members throughout the four workshops.

References

[1] Kaggle, "Loan Default Prediction Competition." Available: https://www.kaggle.com/competitions/loan-default-prediction.

[2] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proc. 22nd SIGKDD*, pp. 785–794, 2016.

[3] G. E. Hinton and R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[4] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Reviews of Modern Physics*, vol. 55, no. 3, pp. 601–644, 1983.

[5] ISO 9000:2015, "Quality Management Systems — Fundamentals and Vocabulary."

[6] CMU/SEI, "CMMI for Development, Version 1.3," Carnegie Mellon University, Software Engineering Institute, 2010.