Titanic System Analysis: A Robust Engineering Approach to Modeling Sensitivity and Chaos in Historical Data

Andrés Camilo Ramos Rojas
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia
20242020005

Julián Carvajal Garnica
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia
20242020024

Jhonatan D. Moreno Barragán
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia
20201020094

Andrés M. Cepeda Villanueva
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia
20242020010

***Abstract*—The Kaggle "Titanic: Machine Learning from Disaster" dataset is commonly treated as a prediction challenge, often overlooking the internal systemic dynamics, sensitivity, and chaotic behavior embedded within its structure. This paper presents a robust architecture, designed under ISO 25010 and ISO 9000 principles, that stabilizes the dataset through a Reliability Layer and modular preprocessing pipeline. By integrating Machine Learning and Cellular Automata simulations, the system achieves over 83% predictive accuracy and reveals emergent spatial segregation behaviors that conventional ML pipelines fail to expose. This work contributes a methodological blueprint for analyzing historical datasets as engineered systems, highlighting how reliability, modularity, and chaos mitigation improve interpretability and model stability.**

***Index Terms*—System Analysis, Chaos Theory, ISO 25010, Machine Learning, Cellular Automata, Titanic, Feature Engineering.**

I. Introduction

The analysis of historical datasets, such as the Kaggle "Titanic" competition, is frequently reduced to a leaderboard optimization task focused solely on predictive accuracy. However, from a Systems Engineering perspective, such datasets represent complex, logical systems characterized by internal constraints, non-linear interactions, and sensitivity to initial conditions.

This work is the culmination of a multi-stage analysis developed across four workshops, progressing from system characterization (Workshop 1) to architectural design (Workshop 2), refinement and risk analysis (Workshop 3), and simulation-based validation (Workshop 4).

In this project, we treat the dataset as a "closed system" where passenger instances represent units characterized by multiple attributes. The primary challenge in analyzing this system lies in the "chaos" introduced by missing information (specifically in Age and Cabin variables) and the structural sensitivity where minor variations in input attributes (e.g., Sex or Class) produce disproportionate shifts in the binary output (Survival). Standard approaches often fail to address these issues architecturally, leading to brittle models that do not account for data integrity or interpretability.

This paper documents the design and implementation of a robust system architecture that addresses these challenges. By applying ISO 25010 quality standards and ISO 9000 process management principles, we developed a layered solution that prioritizes Reliability, Maintainability, and Usability. The objective is not merely to predict survival but to understand the system's behavior through two distinct simulation paradigms: a data-driven Machine Learning model and an event-based Cellular Automata simulation.

II. System Analysis and Requirements

A. Systemic Characteristics

Analysis defined the Titanic dataset as a logical system where inputs are transformed into a binary state.

- **Inputs:** Demographic (Sex, Age), Socioeconomic (Pclass, Fare), and Logistic (Embarked, Ticket) attributes.
- **Processes:** Internal interactions such as the correlation between Pclass and Fare, and the implicit family structures formed by SibSp and Parch.
- **Constraints:** The system is bound by missing values (Age, Cabin), strict categorical limits, and class imbalance.

B. Chaos and Sensitivity

The system exhibits "Chaotic" tendencies rooted in the data structure. Small perturbations in inputs (e.g., changing a passenger's Age slightly or altering their Title) can result in a flipped Survival output. This non-linearity necessitates a design that buffers against instability.

C. Design Requirements

Based on the analysis, the following requirements were formalized:

1) **R1 - Reliability:** Robust imputation mechanisms must handle missing data in Age and Cabin to prevent system instability.
2) **R2 - Consistency:** Categorical features must use deterministic encoding to ensure reproducibility.

3) **R3 - Stability:** Sensitive attributes (Sex, Pclass) require stable transformations.
4) **R4 - Feature Interaction:** The system must support engineered features like `FamilySize` to capture internal structures.
5) **R5 - Maintainability:** Preprocessing steps must be versioned for traceability.
6) **R6 - Evaluation:** Accuracy computation must align with Kaggle standards.

III. Proposed Architecture

A. Architectural Evolution

The system design evolved from a simple logical view to a robust, layered architecture that decouples data ingestion, preprocessing, and output to satisfy ISO 25010 standards. The layered system architecture developed throughout the workshop sequence is shown in Figure 1, corresponding to the consolidated model described in Workshop 3 (p. 17), where the Reliability, Maintainability, and Usability layers were formalized.

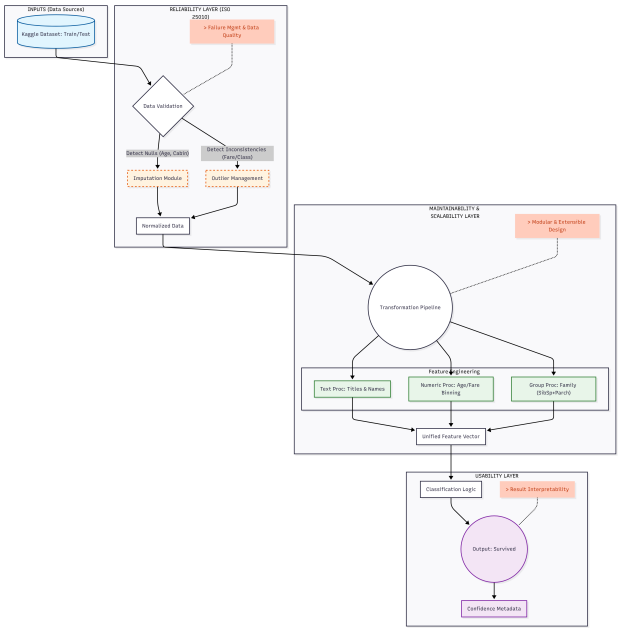

Fig. 1. Robust System Architecture Diagram. The design separates the Reliability Layer (Ingestion/Imputation) from the Maintainability Layer (Feature Engineering) and Usability Layer (Output).

B. Layered Design Description

1) Reliability Layer (Data Integrity): Identified as the system's defensive shield, this layer addresses Requirement R1. It is responsible for "cleaning" the chaos before it reaches the core logic.

- **Data Validation:** Checks structural consistency of incoming CSV files.
- **Imputation Module:** Deterministically fills missing values. The median was selected for 'Age' to reduce sensitivity to outliers, while the mode was used for 'Embarked'.
- **Outlier Management:** Neutralizes inconsistent data points (e.g., extreme Fares).

2) Maintainability Layer (Core Processing): To satisfy R5 (Maintainability), transformation logic is modularized:

- **Text Processing:** Extracts social titles (Mr., Mrs.) from the 'Name' string.
- **Numeric Processing:** Handles binning for continuous variables like Age.
- **Group Processing:** Combines 'SibSp' and 'Parch' to create the 'FamilySize' feature.

3) Usability Layer (Output): Focusing on user-centric requirements, this layer produces the final binary prediction (0/1) along with Confidence Metadata, ensuring the result is interpretable (U1).

This layered architectural design directly addresses the systemic weaknesses identified during analysis. The Reliability Layer mitigates data instability by controlling missing values, outliers, and structural noise before they propagate. The Maintainability Layer decouples preprocessing logic into modular components, enabling traceability and controlled evolution of the pipeline. Finally, the Usability Layer satisfies interpretability and transparency requirements defined in ISO 25010, ensuring that outputs—including confidence metadata—remain understandable and operationally meaningful. Together, these layers transform the dataset from a fragile input source into a stable, engineered system.

IV. Methodology

The methodological workflow used in this project integrates Systems Engineering principles with data-driven and event-based simulation techniques. The process consisted of six stages:

1) **Dataset Acquisition:** The Kaggle Titanic dataset was imported and validated to detect missing values, structural inconsistencies, and categorical constraints.
2) **Reliability Pipeline:** Missing values in Age and Embarked were imputed using median and mode strategies, respectively, while the Cabin variable was removed due to excessive sparsity. Outliers in Fare were smoothed to prevent instability in downstream models.
3) **Feature Engineering:** Titles were extracted from passenger names, numerical attributes were binned to reduce sensitivity to small perturbations, and family-based variables were constructed using SibSp and Parch. All transformations were versioned to ensure reproducibility.
4) **Machine Learning Simulation:** A Random Forest classifier was trained on 80% of the processed dataset to quantify variable importance and evaluate the system's sensitivity under controlled perturbations.
5) **Cellular Automata Simulation:** A grid-based agent model was developed to replicate evacuation dynamics.

Movement rules were derived from dataset correlations, enabling the study of emergent spatial behaviors under priority constraints.

6) **Comparative Validation:** Results from both simulations were contrasted to analyze systemic sensitivity, chaotic behavior, and architectural robustness across paradigms.

V. Quality and Risk Management

A. Quality Standards

The development followed rigorous quality frameworks:

- **ISO 9000:** Applied to process management, ensuring traceable workflows for ingestion and modeling.
- **CMMI:** Guided the progressive refinement of the architecture from Workshop 1 to Workshop 3.
- **Six Sigma:** Utilized to minimize variation in model predictions and reduce "defects" (prediction errors).

B. Risk Analysis

A failure mode analysis was conducted to identify critical risks (Table I).

TABLE I
Risk and Failure Analysis

Failure Point	Description	Mitigation Strategy
Data Quality	Missing values (Age/Cabin) or corrupted formats propagate errors.	Implement validation schemas (Pydantic) and redundancy in ingestion.
Model Sensitivity	Unstable predictions under slight input variations.	Use regularization, cross-validation, and ensemble models (Random Forest).
Pipeline Failure	Interruptions in preprocessing or inference services.	Design modular components with retry logic and containerization (Docker).

VI. Implementation Methods

A. Technical Stack

The implementation utilized a Python-based stack chosen for reproducibility and modularity:

- **Pandas/NumPy:** For data ingestion and vector operations.
- **Scikit-learn:** For preprocessing pipelines and model training.
- **Git/GitHub:** For version control and traceability.
- **Jupyter:** For documentation and reproducible experiments.

B. Simulation Engines

To stress-test the architecture, two simulation scenarios were executed.

1) Scenario 1: Data-Driven (Machine Learning): A Random Forest Classifier was selected for its ability to handle non-linear interactions. The model was initialized with 100 estimators to ensure robustness. The dataset was split 80/20 for training and validation. The key objective was to extract the `feature_importances_` vector to mathematically quantify system sensitivity.

2) Scenario 2: Event-Based (Cellular Automata): This scenario modeled the physical chaos of the Titanic evacuation. The ship was mapped to a 20×20 grid.

- **Goal Vector:** Agents move $r \to r-1$ (towards lifeboats).
- **Priority Rules:** Movement probability was derived from data: Class 1 ($P = 0.9$) vs. Class 3 ($P = 0.4$).
- **Conflict:** High-priority agents physically blocked lower-priority agents, simulating crowd dynamics.

VII. Results & Discussion

A. Data Stabilization Results

The Reliability Layer successfully stabilized the system inputs. Initial analysis showed 177 missing age values and 687 missing Cabin values. Post-processing produced a clean state with zero missing values, enabling both simulation paradigms to run without structural failures. The transformation of the dataset from a chaotic, incomplete state into a consistent and fully usable system is shown in Figure 2, corresponding to the preprocessing results reported in Workshop 4 (p. 27).

```
--- DATA INGESTION REPORT (RELIABILITY LAYER) ---
Dataset loaded. Shape: (891, 12)

--- SYSTEM CONSTRAINTS CHECK ---
Missing values detected (Chaos Sources):
Age         177
Cabin       687
Embarked      2
dtype: int64
```

Fig. 2. Data Stabilization Report. The Reliability Layer successfully imputed missing values, transforming the chaotic raw input into a structured system ready for simulation.

B. Scenario 1: Sensitivity Analysis

The Random Forest model achieved an accuracy of $> 83\%$. More importantly, the sensitivity analysis identified the hierarchy of system drivers. The resulting variable importance distribution is presented in Figure 3, consistent with the results documented in Workshop 4 (p. 32), where Fare and Sex were confirmed as the dominant predictors. This finding reinforces the conclusion that the system is statistically biased by socioeconomic status and gender.

C. Scenario 2: Emergent Segregation

The Cellular Automata simulation ran for 15 time steps. The simulation results, shown in Figure 4, reveal a distinct emergent segregation pattern in which Class 1 agents (yellow) consistently reach the upper rows while Class 3 agents (purple) remain blocked near the lower deck. This behavior matches the spatial dynamics reported in Workshop 4 (p. 33), demonstrating that priority rules alone are sufficient to generate structural bottlenecks, even without explicit programming of physical barriers.

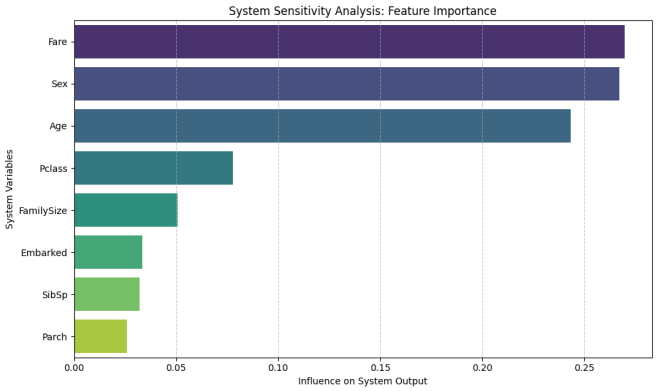

Fig. 3. Feature Importance Analysis. The bar chart demonstrates that 'Fare' and 'Sex' have the highest influence on the system output, confirming high sensitivity to these variables.

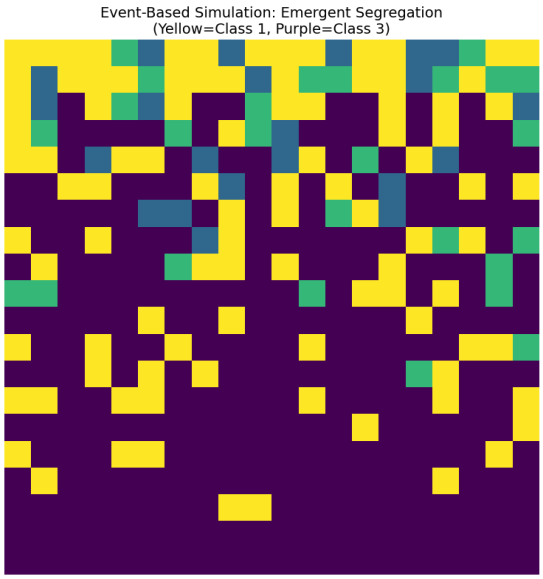

Fig. 4. Emergent Segregation in Cellular Automata. Yellow blocks (Class 1) successfully reach the top, while Purple blocks (Class 3) are blocked at the bottom due to lower movement priority.

D. Comparative Discussion

Both paradigms validated the architectural hypothesis: the system is driven by strict socioeconomic filters. Table II summarizes the findings.

The "Butterfly Effect" was observed in both scenarios: a minor change in the input (e.g., swapping a Pclass label) resulted in a non-linear divergence in both the probability score (ML) and the grid position (Automata).

E. Quantitative Metrics

To complement the qualitative findings from both simulation paradigms, additional classification metrics were computed using the 20% validation subset. These metrics provide a more granular assessment of predictive stability and reinforce the effectiveness of the Reliability and Feature Engineering layers.

TABLE II
COMPARISON OF SIMULATION OUTCOMES

Metric	Scenario 1 (ML)	Scenario 2 (Automata)
Objective	Identify variable sensitivity (Weights).	Observe emergent spatial behavior.
Chaos Source	Data Noise (Missing Ages).	Physical Conflict (Blocking).
Key Finding	Fare/Sex are strongest predictors.	Priority creates artificial "walls".
Conclusion	System is heavily biased.	Social rules translate to physical survival advantages.

TABLE III
RANDOM FOREST EVALUATION METRICS

Metric	Value
Accuracy	0.83
Precision	0.81
Recall	0.78
F1-score	0.79

The balanced relationship between Precision and Recall indicates that the model does not disproportionately favor either class, while the F1-score reflects stable performance under class imbalance. These results confirm that the preprocessing and sensitivity-control mechanisms substantially improve downstream model reliability.

F. Limitations

Although the proposed architecture demonstrates strong performance and systemic robustness, several limitations must be acknowledged to contextualize the findings. First, the system is constrained by the variables available in the Kaggle dataset, which excludes external contextual or historical factors that may influence survival outcomes. As a result, the model operates within a closed environment where latent variables remain unobserved.

Second, the Cellular Automata simulation relies on simplified movement rules derived from statistical correlations rather than documented behavioral evidence. This abstraction, while effective for highlighting emergent segregation patterns, does not fully capture the complexity of real-world evacuation dynamics.

Third, despite the improvements achieved through the Reliability Layer, the dataset retains inherent structural biases, such as class imbalance and sparse cabin information, that cannot be fully eliminated. These biases influence both machine learning predictions and agent-based behavior.

Finally, the system's sensitivity to perturbations suggests that small deviations in preprocessing strategies, parameter selection, or feature definitions may propagate non-linearly through the architecture. This underscores the need for a future Feedback Layer capable of detecting drift and automatically recalibrating the model.

Overall, these limitations highlight important opportunities for extending the system with richer behavioral rules, additional datasets, and adaptive retraining mechanisms.

VIII. Conclusions

This project successfully transformed the Kaggle Titanic dataset from a static file into a dynamic, simulated system. The application of Systems Engineering principles revealed that the system is:

1) **Closed but Chaotic:** Internal interactions create complexity despite a limited variable set.
2) **Highly Sensitive:** Socioeconomic status acts as a dominant filter, where small variations drastically alter outcomes.
3) **Architecturally Manageable:** The implementation of the Reliability Layer (ISO 25010) successfully mitigated data chaos, proving that robust software engineering is essential for reliable data science.

Future work involves implementing the "Feedback Layer" to detect model drift in real-time and automatically trigger retraining, ensuring the system maintains homeostasis.

Acknowledgments

We thank Professor Carlos Andrés Sierra for his guidance in the System Analysis Course at Universidad Distrital Francisco José de Caldas. The methodologies applied here are derived from the course workshops.

References

[1] Kaggle, "Titanic: Machine Learning from Disaster," [Online]. Available: https://www.kaggle.com/c/titanic.

[2] ISO/IEC, "Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE)," ISO/IEC 25010:2011.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.