Workshop IV

Universidad Distrital Francisco José de Caldas

Faculty of Engineering

Authors

Juan David Zárate Moya 20222020184
Jesus Mateo Munevar Mendez 2023202042
Kevin David Rincon Valencia 20232020356
Juan David Romero Morales 20222020102

Teacher
Carlos Andrés Sierra Virguez

November 2025
Bogotá D.C

Contents

1 Systemic Classification of the Dataset Variables

The variables from the `train.csv` file were organized according to their role within the Ames real estate system. This classification allowed us to understand the structural, spatial, and functional dimensions of the system before designing predictive models or causal diagrams.

1.1 1. Structural Variables (Physical characteristics of the property)

- **LotArea**: Total lot area.
- **LotFrontage**: Lot frontage.
- **OverallQual**: Overall material quality.
- **OverallCond**: Overall condition.
- **YearBuilt**: Year of construction.
- **YearRemodAdd**: Year of remodeling.
- **MasVnrArea**: Masonry veneer area.
- **BsmtFinSF1, BsmtFinSF2**: Finished basement areas.
- **BsmtUnfSF**: Unfinished basement area.
- **TotalBsmtSF**: Total basement area.
- **1stFlrSF, 2ndFlrSF**: Floor surface areas.
- **GrLivArea**: Above-ground living area.
- **LowQualFinSF**: Low-quality finished area.
- **Bedroom**: Number of bedrooms.
- **Kitchen**: Number of kitchens.
- **TotRmsAbvGrd**: Total rooms above ground.
- **Fireplaces**: Number of fireplaces.
- **GarageArea**: Garage area.
- **GarageCars**: Garage capacity.
- **WoodDeckSF, OpenPorchSF, EnclosedPorch, ScreenPorch**: Exterior areas.
- **PoolArea**: Pool area.

1.2 2. Spatial Variables

- **Neighborhood**: Neighborhood.
- **Condition1, Condition2**: Proximity to roads.
- **LotShape**: Lot shape.
- **LandContour**: Topography.
- **LandSlope**: Slope.
- **LotConfig**: Configuration.

1.3 3. Functional Variables

- **Functional**: Overall functionality.
- **Heating, HeatingQC**: Type and quality of heating.
- **Electrical**: Electrical system.
- **CentralAir**: Air conditioning.
- **KitchenQual**: Kitchen quality.
- **FireplaceQu**: Fireplace quality.
- **GarageQual, GarageCond**: Garage quality and condition.
- **ExterQual, ExterCond**: Exterior quality and condition.
- **BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2**: Basement details.

1.4 4. Construction Variables

- **BldgType**: Building type.
- **HouseStyle**: House style.
- **RoofStyle, RoofMatl**: Roof style and material.
- **Exterior1st, Exterior2nd**: Exterior materials.
- **MasVnrType**: Veneer type.
- **Foundation**: Foundation type.
- **GarageType**: Garage location.
- **GarageFinish**: Garage finish.

1.5 5. Temporal Variables

- **MoSold**: Month sold.
- **YrSold**: Year sold.

1.6 6. Administrative Variables

- **MSSubClass**: Building class.
- **MSZoning**: Zoning classification.
- **Street**: Street type.
- **Alley**: Alley.
- **Utilities**: Utilities.
- **SaleType**: Sales type.
- **SaleCondition**: Sales condition.

1.7 7. Target Variable

- **SalePrice**: Final price.

2 Dataset Cleaning and Preparation

Before training the models, we reviewed the structure of `train.csv`, identified missing values, noisy variables, and inconsistent categories, and based on this we made justified decisions to ensure a coherent, usable, and stable dataset.

2.1 Variable Selection

We began with all available variables, but chose to work only with those that truly provided useful information to the model or that had a clear role within the real estate system. The selection was based on visual inspection, correlations, domain knowledge, and preliminary tests. The final subset included:

- **Structural**: `GrLivArea`, `TotalBsmtSF`, `GarageCars`, `GarageArea`, `LotArea`.
- **Quality**: `OverallQual`, `OverallCond`, `KitchenQual`, `ExterQual`.
- **Temporal**: `YearBuilt`, `YearRemodAdd`.
- **Habitability**: `FullBath`, `TotRmsAbvGrd`.
- **Key categorical**: `Neighborhood`.

2.2 Cleaning Process

During the initial review we found missing values in several numerical and categorical variables. To avoid deleting complete rows—which would have excessively reduced the sample—we applied the following strategies:

- **Numerical variables**: Imputed using the **median**. This decision was made because many of these variables are skewed and contain outliers, so the mean would have distorted the data.
- **Categorical variables**: Imputed using the **mode**, as it reflects the most common category and maintains consistency with the typical behavior of most houses.

2.3 Variable Encoding

To allow the models to use categorical variables, we applied two types of transformations:

- **Ordinal variables** such as `KitchenQual` and `ExterQual` were mapped to numerical values following a logical quality order:

  $$\text{Po} = 1,\ \text{Fa} = 2,\ \text{TA} = 3,\ \text{Gd} = 4,\ \text{Ex} = 5$$

  This preserves the interpretation from "worst to best".
- For **Neighborhood**, which has no natural order, we used *one-hot encoding*. This ensures that the model does not assume relationships that do not exist among neighborhoods.

2.4 Creation of Derived Variables

To capture deeper system characteristics, we added three variables that facilitate both simulation and interpretation of causal loops:

- `HouseAge = 2010 - YearBuilt` — approximate age of the house.
- `YearsSinceRemodel = 2010 - YearRemodAdd` — time since the last remodeling.
- `TotalBathrooms` — consolidated count of relevant bathrooms.

These variables were useful to model wear, aging, and temporal effects on price.

2.5 Final Result

After applying cleaning, imputation, encoding, and creation of new variables, we obtained the final version of the dataset used to train all models and for systemic analysis. This file was exported as:

`dataset_limpio_equipo1.csv`

3 Data-Driven Simulation (Machine Learning)

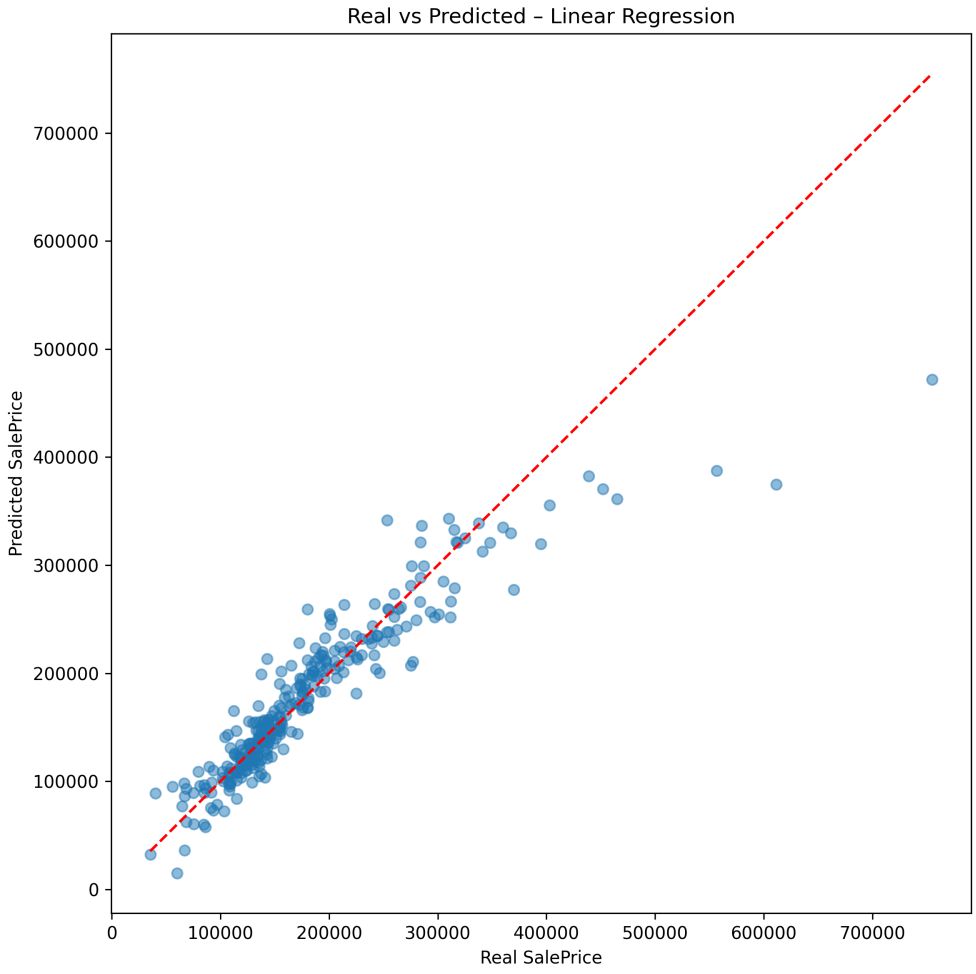
For this part of the workshop, we implemented and compared two predictive models to analyze the behavior of the real estate system from a quantitative perspective. The trained models were a **Linear Regression** and a **Random Forest Regressor**. The objective was to evaluate how well each model could explain the variability of *SalePrice* and, at the same time, identify which variables have the greatest influence in the system.

3.1 Linear Regression Results

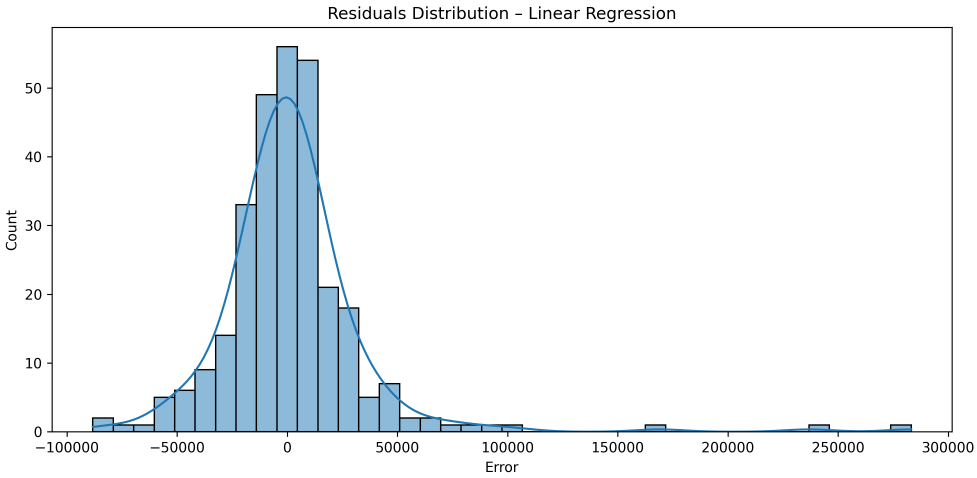
Linear regression served as a baseline model, as it clearly interprets the global relationship between the selected variables and the sale price.

- The obtained metrics (RMSE and R^2) showed acceptable performance for a linear model.
- In the *Real vs Predicted* plot, the points align reasonably well with the ideal diagonal, indicating that the model captures an important part of the general behavior.
- The residual distribution remained centered around zero, although with longer tails, which is common in real estate data due to extreme values.

3.1.1 Figure 1: Real vs. Predicted — Linear Regression



3.1.2 Figure 2: Residual Distribution — Linear Regression

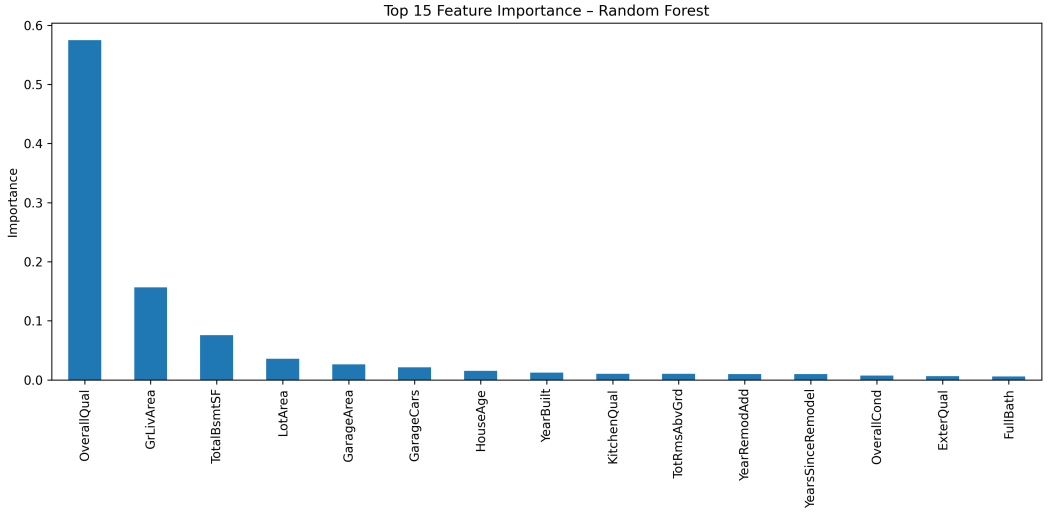


3.2 Random Forest Results

The *Random Forest* model offered better overall performance than linear regression. This was expected due to its ability to capture nonlinear relationships and interaction effects between variables.

- The RMSE decreased significantly compared to the linear model.
- The R^2 value increased, indicating greater explanatory power.
- The variable importance analysis showed that structural and quality characteristics exert the most influence on price.

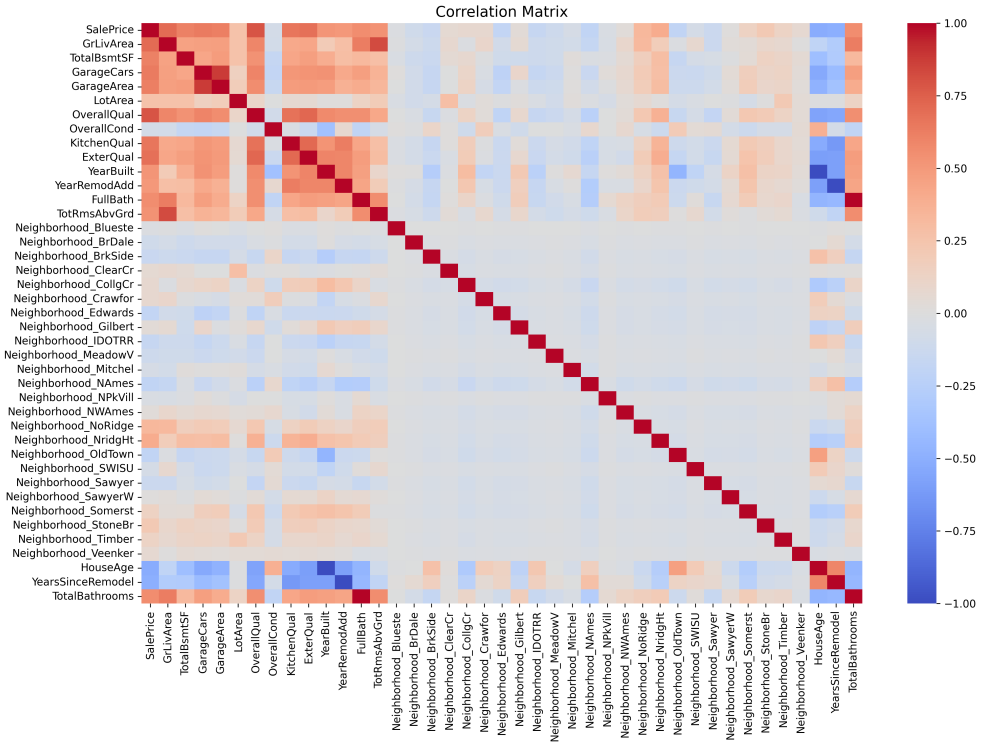
3.2.1 Figure 3: Feature Importance — Random Forest



3.3 Correlation Analysis

The correlation matrix allowed us to identify strong relationships between variables before model training. This helped us understand why certain attributes were highlighted by the Random Forest model.

3.3.1 Figure 4: Correlation Matrix



4 Connection with Causal Loops and the Systemic Approach

Machine Learning analysis not only allowed us to predict housing prices but also helped identify structural patterns within the real estate system. Based on the strongest relationships found by the models—especially by Random Forest—we proposed causal loops that explain reinforcing behaviors present in the market.

4.1 Relationship Between Simulation and Causal Loops

The quantitative results revealed dynamics that can be interpreted as positive feedback loops within the system. Among the most relevant:

- **OverallQual → SalePrice → Incentive for remodeling → OverallQual**: As overall quality increases, the sale price increases. A higher price incentivizes future remodeling, which improves quality again. This forms a classic *reinforcing loop* of continuous improvement.
- **GrLivArea → SalePrice → Incentive for expansions → GrLivArea**: Homes with larger living areas tend to achieve higher prices. That appreciation generates incentives to build expansions, which again increases the area—another clear reinforcing loop.

- **GarageCars / TotalBsmtSF → Comfort → Perceived value → GarageCars / TotalBsmtSF**: Larger garage or basement spaces increase perceived utility, raising value and motivating owners or builders to invest in these areas.

4.2 Scenario Exploration ("playing with the data")

To validate these systemic relationships, we performed experiments by manually adjusting key variables in the dataset. The goal was to observe how the models responded to changes in specific system conditions.

- Controlled increases in the **OverallQual** variable.
- Increases in lot area or living area.
- Simulation of aging through higher **HouseAge** and **YearsSinceRemodel**.

These scenarios allowed us to estimate the sensitivity of the system, identify leverage points, and verify whether the proposed relationships produced the expected effects.

5 Scenario 2: Cellular Automata Simulation — Analysis and Interpretation

5.1 Overview

In this simulation we implemented a **Cellular Automata (CA)** model to explore spatial dynamics within a simplified representation of a housing market. The automaton operates on a **20×20 grid**, where each cell represents a local housing unit or micro-region. Each cell evolves according to simple **local interaction rules**, leading to complex global behavior, including pattern formation, clustering, contagion, and recovery.

The CA does not use real dataset values; instead, it models **conceptual behavior** that complements the ML model from Scenario 1 by simulating emergent spatial interactions within the system.

5.2 States Definition

Each cell can take one of three possible states:

- **State 0 – Stable:**
  Represents regions with normal conditions and low influence from neighboring disturbances.
- **State 1 – Risky:**
  Represents areas showing signs of instability, affected by local contagion or transitions.

- **State 2 – Noise/Chaos:**
  Represents high-volatility regions that produce unpredictable changes in surrounding cells.

These states reflect the system's conceptual dynamics rather than real housing attributes.

5.3 Transition Rules

The evolution of the system was governed by simple yet expressive rules:

1. **Risk Contagion:**
   If a cell has more than three risky neighbors, it becomes risky.
2. **Noise Influence:**
   If any neighbor is in the noise state, the cell may transition with a probability of 25%, modeling chaotic perturbations.
3. **Recovery:**
   Risky cells surrounded by stable neighbors tend to return to the stable state.
4. **External Shock (Global Event):**
   Every 10 iterations, 3% of the grid is randomly converted to the noise state, mimicking a macro-level disturbance.

Despite being simple, these rules produce non-linear behavior.

5.4 Simulation Behavior and Emergent Phenomena

Across the 40 iterations, the automaton produced rich dynamic patterns illustrating how local interactions can scale into complex global behavior.

5.4.1 Clustering

Cells with similar states formed **spatial clusters**—stable patches, risky corridors, and chaotic hotspots. These clusters **emerged naturally**, even though no global structure was explicitly programmed.

5.4.2 Expansion and Retraction

Risky zones expanded toward stable regions but also shrank when surrounded by stability. This dynamic "breathing" pattern reflects **feedback interactions** within the system.

5.4.3 Sensitivity to Noise

Noise cells introduced irregular, unpredictable transitions. Small perturbations in noisy areas produced disproportionately large effects downstream—demonstrating **chaos-like behavior** and sensitivity to initial conditions.

5.4.4 Reaction to Shocks

The periodic global events temporarily increased instability, but the system partially recovered afterward, exhibiting **resilience**.

Together, these dynamics show how local rules can produce **macro-level behavior that was not predetermined**, creating a strong argument for emergent behavior within complex systems.

The results of the iterations here:

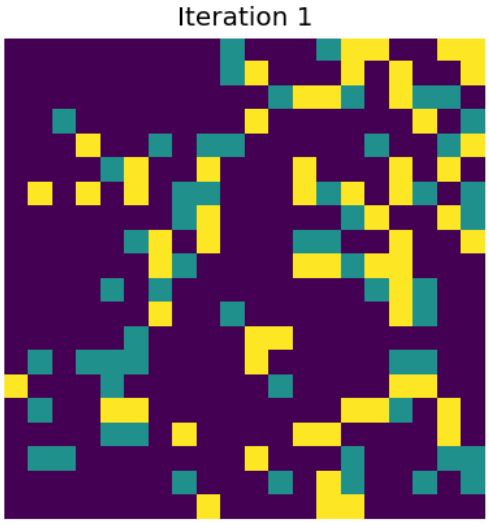


Figure 1: **Cellular Automata State Distribution at Iteration 1.**

This figure shows the initial spatial configuration of the Cellular Automata model. Although states are randomly assigned based on predefined probabilities, small clusters of 'risky' and 'noise' cells begin to appear naturally, forming the seeds of later emergent behavior.

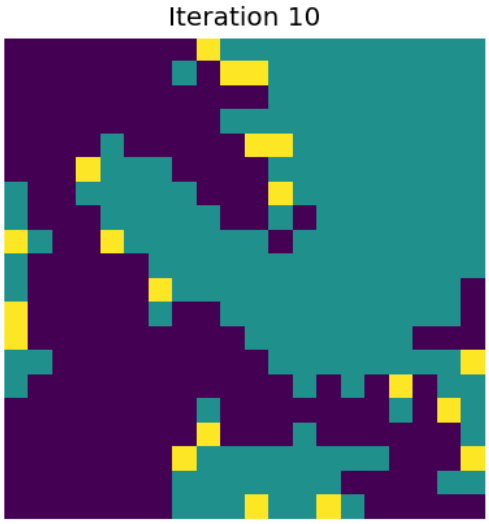


Figure 2: **Cellular Automata Evolution at Iteration 10.**

By iteration 10, the first global shock has occurred, introducing random noise perturbations throughout the grid. Local contagion rules begin to take effect, with risky cells spreading to adjacent stable regions. Early clustering patterns become more pronounced as the system responds to both neighbor interactions and external disturbances.

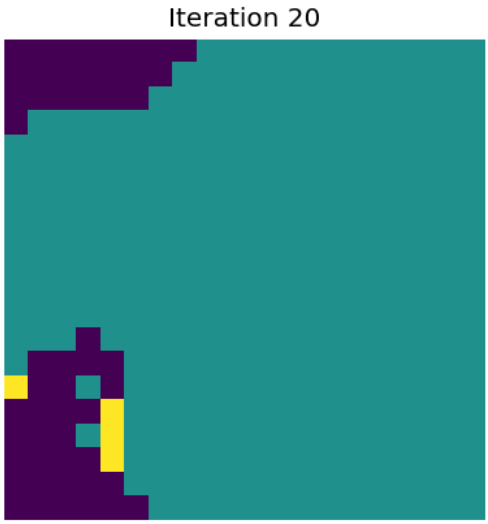


Figure 3: **Cellular Automata Evolution at Iteration 20.**

At iteration 20, distinct spatial patterns have emerged. Risky regions have spread due to local contagion effects, while noise cells introduce irregular disruptions. Stable regions begin

to consolidate, revealing early evidence of clustering and non-linear transitions.

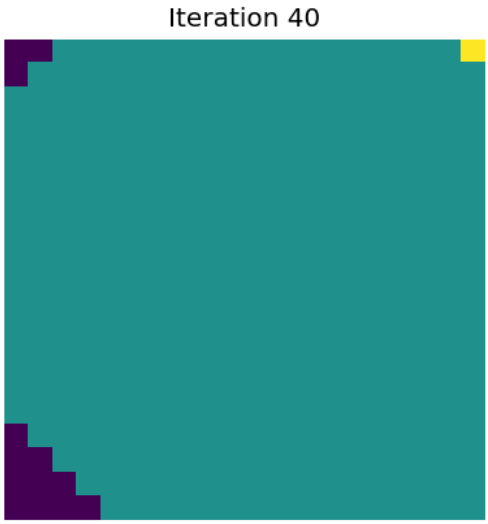


Figure 4: **Cellular Automata Final Configuration at Iteration 40.**

The final iteration presents a complex mixture of stable zones, concentrated risky clusters, and intermittent noise-induced perturbations. The system exhibits sustained dynamism rather than convergence, demonstrating sensitivity to local interactions and global shocks.

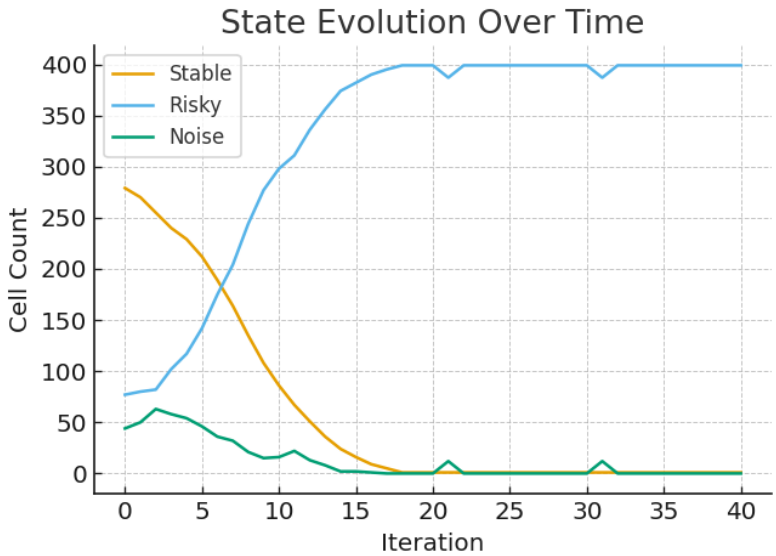


Figure 5: **State Evolution Over Time.**

This line plot summarizes the temporal dynamics of the three states across 40 iterations. The stable state dominates early but fluctuates as risky regions grow and shrink. Noise

spikes are clearly visible following periodic shock events, illustrating system-level instability and recovery patterns.

5.5 Quantitative Evolution

A CSV file containing the count of stable, risky, and noise states per iteration allows tracking system-level tendencies. Typical observations include:

- The number of risky cells fluctuates rather than converging.
- Noise tends to spike after shocks and then decrease.
- Stability dominates early but is continually threatened by local contagion.

This behavior mirrors real-world systems where **micro-level interactions produce unpredictable macro-level patterns over time**.

5.6 Conclusion

The Cellular Automata simulation successfully demonstrates:

- **Emergence:** Complex patterns form from simple rules.
- **Local Interaction Effects:** Neighbor relationships strongly influence regional behavior.
- **Sensitivity to Perturbations:** Noise and shocks generate non-linear propagation.
- **System-Level Dynamics:** The model evolves in a non-stationary, self-organizing manner.

This event-based simulation complements the machine learning model by providing a conceptual understanding of how local interactions within a system architecture can generate global behavior—validating the broader System Design explored in Workshop 2.