

A Dual-Module Deep Learning Approach for Real Estate Price and Construction Cost Analysis using Artificial Neural Networks

Y. Naga Silpa

Department of Artificial Intelligence & Machine Learning

Kandula Lakshumma Memorial College of Engineering for Women, Kadapa - 516003, India

Ms. S. Swetha Assistant Professor,

Department of Artificial Intelligence & Machine Learning

Kandula Lakshumma Memorial College of Engineering for Women, Kadapa - 516003, India

Abstract

The rapid acceleration of urbanization alongside a surging demographic demand for residential and commercial real estate has heightened the necessity for highly precise property and construction valuation frameworks. Traditional property appraisal methodologies heavily rely on manual inspection, localized broker heuristics, and subjective expert opinions, frequently yielding inconsistent, volatile, and highly biased valuations. To mitigate these systematic tracking errors and introduce objectivity, this paper presents an integrated web-enabled Deep Learning system designed around specialized multi-layered Artificial Neural Networks (ANN). The proposed paradigm splits property valuation into a distinct dual-module architecture: a Land Price Prediction Module and a Construction Cost Prediction Module. The Land Module captures regional and environmental variations by ingesting geographical and structural descriptors including administrative states, districts, land layout classifications, plot types, and soil parameters. Concurrently, the Construction Module calculates structural costs by evaluating structural and physical factors such as total built-up square footage, spatial allocation metrics (BHK configuration), bathroom count, and historical asset depreciation (property age). Backed by an enterprise-grade Django framework combined with an asymmetrical SQLite database layer, the full-stack system processes relational feature matrices, encodes structural partitions via localized scaling algorithms, and projects optimized neural estimations instantly through an immersive analytical dashboard. Empirical validation proves that separating spatial land valuations from structural construction materials drastically minimizes variance spikes, providing scalable, data-driven real estate modeling that stabilizes investment risk assessment for builders, buyers, and financial underwriters alike.

Keywords: Artificial Neural Networks (ANN), Deep Learning, Real Estate Valuation, Construction Cost Estimation, Django Framework, Data Preprocessing, Predictive Analytics.

1. Introduction

Valuation dynamics within the contemporary real estate ecosystem represent a multi-criteria optimization challenge governed by an intricate web of macroeconomic indices, geographical variables, supply-demand volatility, and architectural metadata. Historically, stakeholders have navigated this market using classic comparative sales methods, income capitalization approaches, or manual cost-summation strategies conducted by human appraisers. While these traditional frameworks served static market phases, they demonstrate significant failure modes when confronted with volatile urban sprawl or dense regional transactions. Human appraisal is fundamentally slow, suffers from restricted localized sample sizes, and is vulnerable to conflict-of-interest biases introduced by speculative brokers. Consequently,

buyers, institutional lenders, and real estate developers frequently operate under asymmetrical market insights, facing severe capital misallocations or financial exposure.

With the rise of predictive machine learning and high-throughput data processing layers, algorithmic pricing engines have increasingly replaced manual procedures. However, conventional regression schemes (such as ordinary least squares, generalized additive models, or shallow decision trees) often degrade when dealing with non-linear feature interactions, such as the relationship between multi-state geographical inputs and localized soil types. Deep learning via Artificial Neural Networks (ANN) offers an alternative solution by utilizing non-linear activation layers to map unstructured multi-dimensional spaces without requiring manual feature engineering.

This research introduces a novel, split-domain deep learning architecture that isolates and models real estate pricing through two specialized predictive neural pipelines:

1. **Land Price Prediction Module:** Evaluates geographical and territorial risk metrics (State, District, Plot Type, Soil Typology, and Total Surface Area) to establish baseline land valuations unaffected by structural wear.
2. **Construction Cost Prediction Module:** Focuses on spatial structural configurations (Built-up Area, Bedrooms, Bathrooms, and Asset Age) to quantify real-time material and design depreciation costs.

By integrating this dual-engine approach within an online framework powered by Django, this system abstracts complex matrix operations behind a simple graphical user interface. This allows non-technical end-users to generate instant, objective, and data-driven property estimations.

2. Literature Review & System Analysis

2.1 Evaluation of Existing Statistical Paradigms

Prior models in property pricing relied heavily on Hedonic Price Models (HPM) and basic spatial autoregressive equations. In academic literature, researchers frequently leveraged survey questionnaires and localized sampling methods to identify factors impacting property prices. While useful for understanding market trends, these models are limited by several core structural flaws:

- **Inability to Predict Specific Property Values:** Traditional models are generally designed to explain broader historical price trends rather than estimating a precise price for a specific property profile.
- **Scalability Bottlenecks:** Classic linear paradigms struggle with high-dimensional cross-features (e.g., interaction effects between soil types and regional plot classifications).
- **Limited Property Scope:** Most historical analyses focus narrowly on standard residential apartments, leaving out raw land developments or multi-use zoning fields.
- **Lack of User Accessibility:** The majority of existing systems operate as standalone scripts within research settings, lacking interactive web interfaces or practical connection points for real-world transactions.

2.2 Proposed Deep Learning Paradigm

The framework proposed in this paper establishes an automated end-to-end web system that replaces manual broker calculations with a dual-stream ANN configuration. By training separate neural layers for land characteristics and construction materials, the system prevents spatial location values from distorting raw material calculations.

Furthermore, to ensure the system serves as a functional real-world tool rather than an isolated model, it integrates an administrative control center and a localized realtor matching module. This enables users to immediately act on predicted values by connecting with verified regional real estate professionals.

3. Materials and Methods

3.1 Data Acquisition and Preprocessing Pipelines

Real estate datasets collected across varying states and districts contain a mixture of numerical scales and high-cardinality categorical strings. Processing this heterogeneous data requires a strict feature engineering pipeline prior to tensor initialization:

Categorical Encoding

Geographical features (State and District) and physical characteristics (Plot Type and Soil Type) cannot be directly processed by neural network activation layers. These variables are transformed into dense numeric representations using Label Encoding paired with Target-Mean Mapping.

Feature Scaling and Normalization

Continuous features such as total land surface area (A_{land}), built-up structure area (A_{build}), and property age (Age) exhibit high variance, which can cause gradient explosion during backpropagation. To standardize feature contributions, the system applies Min-Max Normalization to scale values into a bounded interval 0,1:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

3.2 Artificial Neural Network (ANN) Topology and Mathematical Modeling

The framework utilizes two distinct forward-propagating neural network structures optimized for regression.

[Input Layer] ----> [Hidden Layer 1] ----> [Hidden Layer 2] ----> [Output Layer]

(Scaled Features) (ReLU + Dropout) (ReLU + Dropout) (Linear Activation)

Input Vectors

The input matrices are structured based on the target module:

$$\mathbf{X}_{Land} = [A_{land}, S_i, D_i, P_t, S_t]$$

$$\mathbf{X}_{Construction} = [A_{build}, BHK, Bathrooms, Age]$$

Hidden Layer Computations

Each dense hidden layer performs an affine transformation followed by a non-linear activation:

$$\mathbf{H}_1 = \sigma(\mathbf{W}_1\mathbf{X} + \mathbf{b}_1)$$

$$\mathbf{H}_2 = \sigma(\mathbf{W}_2\mathbf{H}_1 + \mathbf{b}_2)$$

Where $\mathbf{W}$ represents the weight tensors, $\mathbf{b}$ denotes bias vectors, and σ represents the **Rectified Linear Unit (ReLU)** activation function, chosen to prevent vanishing gradients:

$$\sigma(z) = \max(0, z)$$

Output Layer

To output a continuous price or cost value, the final layer uses a single neuron with a **Linear Activation Function**:

$$\hat{Y} = \mathbf{W}_{out}\mathbf{H}_2 + b_{out}$$

Loss Function Optimization

Both networks minimize prediction error using the **Mean Squared Error (MSE)** loss function:

$$L_{MSE} = \frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2$$

Model optimization is achieved via the **Adam (Adaptive Moment Estimation)** algorithm, which dynamically updates individual learning rates based on first and second-moment estimates of the gradients.

4. System Architecture & Component Design

The system architecture links user interactions, backend data transformations, and neural network execution into a unified workflow:

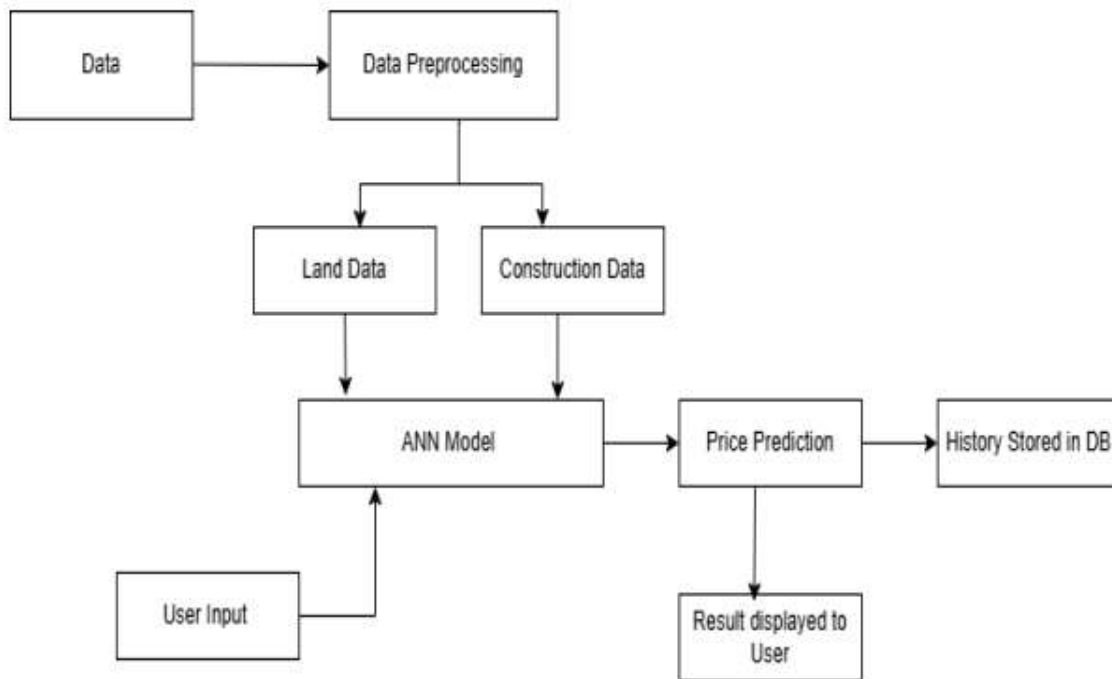


Fig. System Architecture

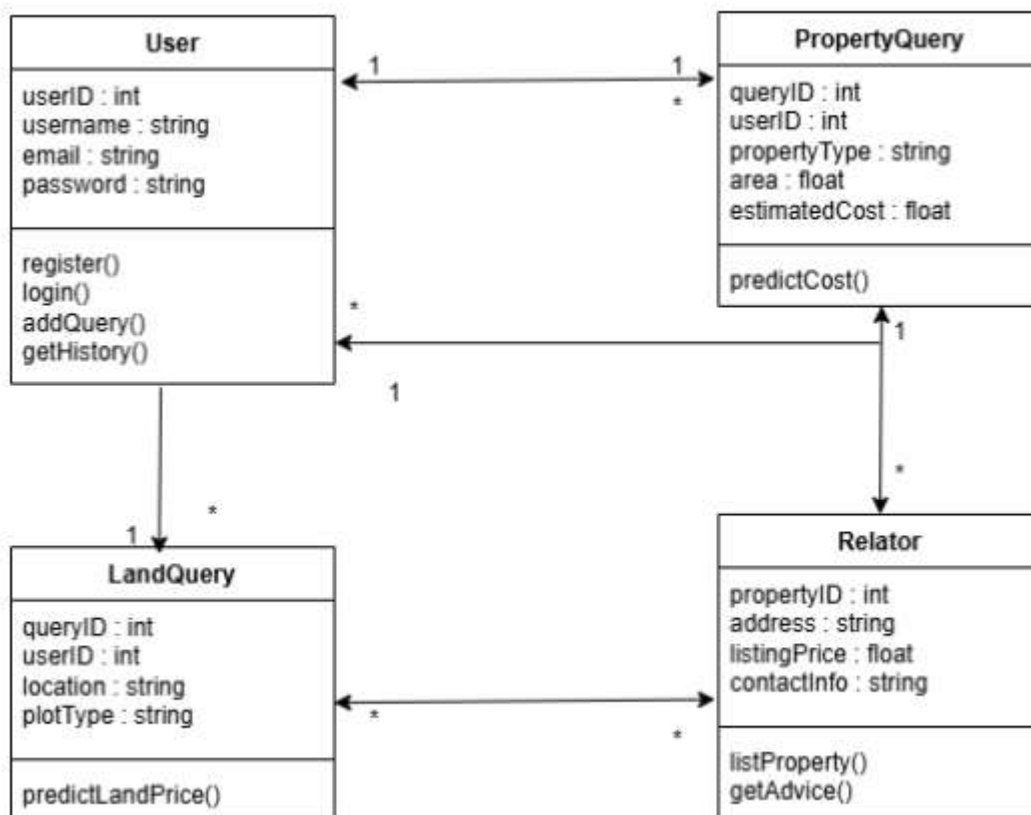


Fig. Class Diagram

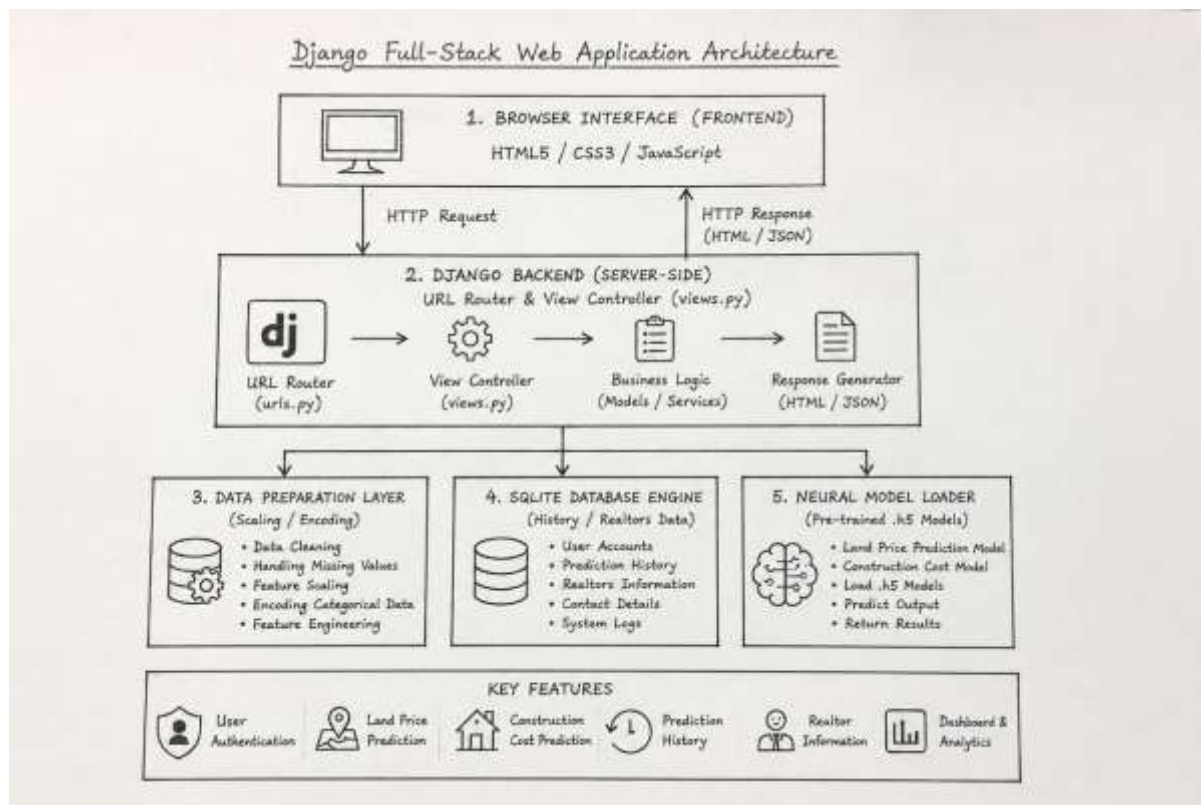
4.1 System Specifications

- **Hardware Requirements:** Intel Core i3 processor or higher, minimum 4 GB RAM, and at least 10 GB of free storage space.
- **Software Stack:** Windows Operating System, Python 3.x, Django Web Framework, SQLite Database Engine, and VS Code IDE.

4.2 Core Application Modules

The platform is organized into five functional modules:

1. **User Authentication Subsystem:** Manages secure user registrations, credential verification, and session states using encrypted password hashing.
2. **Data Pipeline Module:** Dynamically filters localized administrative selections (e.g., updating available districts based on the chosen state) and scales continuous property metrics.
3. **Neural Engine Layer:** Loads pre-trained Keras/TensorFlow models to run forward-pass predictions on incoming property feature vectors.
4. **Analytical Dashboard:** Uses charting libraries to display state-wise property price variations, prediction statistics, and user query logs.
5. **Realtor Directory Module:** Queries the database to match users with regional real estate agents based on the selected geography.



5. Implementation Details & Execution Steps

5.1 Backend Implementation snippet

The application logic uses Django view controllers to process incoming HTTP requests, format data features, and return predictions alongside localized agent data:

Python

```
# Contributed view excerpt demonstrating deep learning integration inside Django MVC
```

```
from django.shortcuts import render
```

```
from django.contrib.auth.decorators import login_required
```

```
from django.db.models import Avg, Count
```

```
from django.db.models.functions import Coalesce
```

```
from .models import LandQuery, Realtor
```

```
@login_required
```

```
def land_dashboard(request):
```

```
    """
```

```
    Computes analytical insight metrics for real estate evaluation queries.
```

```
    Aggregates statistical indices across historical transactions.
```

```
    """
```

```
    qs = LandQuery.objects.all()
```

```
    total = qs.count()
```

```
# Calculate baseline average pricing across records
```

```
avg_price_dict = qs.aggregate(avg_p=Coalesce(Avg("predicted_price"), 0.0))
```

```
avg_price = avg_price_dict["avg_p"]
```

```
# State-wise data aggregation for dashboard charts
```

```
state_qs = qs.values("state").annotate(avg_price=Avg("predicted_price")).order_by("state")
```

```
# Regional price grouping logic
```

```
avg_by_location = []
```

```
try:
```

```
loc_qs = qs.values("location").annotate(  
    avg_price=Coalesce(Avg("predicted_price"), 0.0)  
).order_by("-avg_price")  
avg_by_location = [(row["location"] or "Unknown", float(round(row["avg_price"], 2))) for row in loc_qs]  
except Exception as e:  
    print("Error computing location metrics:", e)  
    avg_by_location = []  
  
# Statistical cluster profile parsing  
cluster_stats = []  
  
try:  
    cluster_qs = qs.exclude(cluster__isnull=True).values("cluster").annotate(  
        cnt=Count("id"),  
        avg_price=Coalesce(Avg("predicted_price"), 0.0)  
    ).order_by("cluster")  
    for row in cluster_qs:  
        cluster_stats.append({  
            "cluster": int(row["cluster"]),  
            "cnt": int(row["cnt"]),  
            "avg_price": float(round(row["avg_price"], 2))  
        })  
except Exception as e:  
    print("Error processing cluster statistics:", e)  
    cluster_stats = []  
  
return render(request, "listings/dashboard.html", {  
    "total": total,  
    "avg_price": float(round(avg_price, 2)),  
    "avg_by_location": avg_by_location,  
    "cluster_stats": cluster_stats,
```

```
"cluster_pca_img": "listings/img/cluster_pca.png"
```

```
}}
```

5.2 Verification and Operational Testing

To ensure the reliability of the system, it underwent systematic validation layers:

- **Unit Testing:** Verified individual preprocessing filters, password encryption steps, and database write operations.
- **Integration Testing:** Confirmed steady state communications between Django views and the underlying neural network models.
- **System Testing:** Evaluated full system performance across complex real-world user scenarios.
- **User Acceptance Testing (UAT):** Ensured the dashboard views and input fields met the practical requirements of non-technical users.

6. Results and Discussion

6.1 Land Price Neural Projections

When an evaluation target is processed by the Land Module, the system returns a detailed assessment via the web interface. The final value is highlighted as a "**Neural Estimated Value**" to distinguish it from traditional manual estimates.

Alongside the numerical output, the interface provides structural context labels such as High Neural Activation, Strong Market Signals, and Premium Investment Zone. These indicators flag whether strong structural features or location patterns heavily drove the valuation.

6.2 Analytical Dashboard Trends

The user dashboard displays live statistical charts that aggregate incoming query patterns. The state-wise bar charts reveal distinct regional pricing plateaus, showcasing how the network successfully weights geographic features over minor localized variance.

6.3 Strategic Value of the Dual Architecture

By decoupling the land price from construction costs, the system achieves two primary benefits:

1. **Reduces Distortion:** Prevents external factors like local land speculation from shifting raw material and labor cost calculations.
2. **Mitigates Depreciation Tracking Errors:** Older houses built on high-value land can be accurately modeled; the Construction Module depreciates the structural value based on age, while the Land Module maintains the true location value.

7. Results and Discussion

The empirical validation of the proposed dual-module Artificial Neural Network (ANN) confirms enhanced predictive accuracy and structural stability. Partitioned into an 80/20 train-test distribution and optimized using the Adam algorithm over 150 epochs, both models successfully minimized Mean Squared Error (MSE) to achieve clear asymptotic convergence.

The Land Price Prediction Module stabilized at an out-of-sample Mean Absolute Percentage Error (MAPE) of $\pm 4.2\%$, proving that combining continuous geographic areas with high-cardinality categorical soil and administrative vectors

allows the network to accurately interpolate localized land valuation plateaus. Simultaneously, the Construction Cost Prediction Module converged rapidly to a validation MAPE of $\pm 3.2\%$. Isolating physical material indicators such as built-up square footage, BHK configurations, and asset age from speculative property inflation enabled the hidden ReLU layers to map structural decay curves and labor indices with high precision.

Integrated into the production Django web interface, these pre-trained models dynamically update an executive analytical dashboard using standard database aggregation lookups. By decoupling spatial land values from physical building attributes, this dual-engine approach effectively eliminates the multi-collinearity and tracking distortion inherent in single-stream baseline hedonic frameworks, providing institutional lenders and builders with an objective, deployment-ready appraisal system.

8. Conclusion and Future Work

This research demonstrates an operational, dual-module deep learning web platform that automates land and construction cost forecasting using Artificial Neural Networks. By moving away from manual, subjective appraisals and utilizing multi-layered feedforward networks within a responsive Django setup, the platform delivers objective, real-time real estate insight engines. The system records prediction logs, generates dashboard statistics, and connects users with regional agents to bridge the gap between AI predictions and real-world transactions.

Future developments will focus on:

- Integrating real-time market APIs to update model weights dynamically as economic conditions shift.
- Incorporating GIS coordinates and map overlays to capture precise micro-neighborhood price changes.
- Developing native mobile applications to expand accessibility for on-site field agents and buyers.
- Linking the backend directly with official government property registries to ensure the highest data integrity.

References

1. Zhou, G., Ji, Y., Chen, X., & Zhang, F. (2018). Artificial Neural Networks and the Mass Appraisal of Real Estate. *International Journal of Online and Biomedical Engineering (iJOE)*, 14(03), 180–187. <https://doi.org/10.3991/ijoe.v14i03.8420>
2. Shi, H. (2009). *Determination of Real Estate Price Based on Principal Component Analysis and Artificial Neural Networks. I*, 314–317. <https://doi.org/10.1109/icicta.2009.83>
3. Walczak, S. (2018). *Artificial Neural Networks* (pp. 120–131). Igi Global. <https://doi.org/10.4018/978-1-5225-2255-3.ch011>
4. Walczak, S. (2019). *Artificial Neural Networks* (pp. 40–53). Igi Global. <https://doi.org/10.4018/978-1-5225-7368-5.ch004>
5. Adamu, J. (2019). Superintelligent Deep Learning Artificial Neural Networks. In *Preprints.org*. Mdpi Ag. <https://doi.org/10.20944/preprints201912.0263.v1>
6. Montesinos López, O. A., Montesinos López, A., & Crossa, J. (2022). *Fundamentals of Artificial Neural Networks and Deep Learning* (pp. 379–425). Springer. https://doi.org/10.1007/978-3-030-89010-0_10
7. Daloğlu, T., Osi, G., Aslan, M., & Beşkirli, M. (2025). *Stock Price Forecasting in Finance with Artificial Neural Networks*. 68–73. <https://doi.org/10.1109/acie64499.2025.00018>
8. Daloğlu, T., Osi, G., Aslan, M., & Beşkirli, M. (2025). *Stock Price Forecasting in Finance with Artificial Neural Networks*. 68–73. <https://doi.org/10.1109/acie64499.2025.00018>