International Journal of Current Research and Techniques (IJCRT)
Volume 16, Issue 2, 2026; DOI: 10.5281/ZENODO.19494154
Original Research Article

AI-Based Predictive Market Pricing Model for Rural Farmers Using Machine Learning

Sarthak Singh, *B.Tech, Computer Science and Engineering NITRA Technical Campus, Ghaziabad, India.*

Ankit Kumar Dixit, *B.Tech, Computer Science and Engineering NITRA Technical Campus, Ghaziabad, India.*

Mr. Sourabh Jain, *Assistant Professor, Computer Science and Engineering, Nitra Technical Campus, Ghaziabad, India.*

Manuscript Received: Apr 05, 2026; Revised: Apr 07, 2026; Published: Apr 10, 2026

Abstract: Rural farmers in developing countries still struggle with wild price swings and unclear markets. These issues often leave them at the mercy of middlemen and facing serious financial stress. This paper introduces an AI-powered tool that predicts future crop prices using machine learning. By crunching historical market numbers together with local weather data, the tool gives farmers straightforward, useful pricing forecasts on their crops. But it's not just a bunch of technical jargon, practical access matters. So, the model connects to a lightweight, easy-to-use web app that farmers can actually open and navigate. The backend runs on Node.js, built for speed with an asynchronous design, so price predictions pop up instantly. On the user side, the frontend is smooth and quick, built with React and TypeScript. The main goal? Close the information gap. Now, farmers have real market insights in their hands and can choose the best time and place to sell what they grow. Tests so far look promising, using XGBoost, the model's price predictions land within 6% of actual prices for staple crops. All in all, this shows you can pair smart web tools with solid algorithms to actually help rural farmers, pushing for fairer markets and more sustainable agriculture.

Keywords: Machine Learning, Crop Price Prediction, Smart Agriculture, XGBoost, Web Application, Node.js.

1. Introduction

Agriculture is the backbone of many developing economies. It keeps rural communities employed and adds a big chunk to their country's GDP. Still, if you look closer, the farmers working at the ground level struggle with serious financial ups and downs. One huge reason is how unpredictable crop prices can be, coupled with markets that aren't exactly transparent. Most of the time, the entire supply chain depends on middlemen. That means farmers have little idea about what's happening in the market or how prices might change down the line. So, after harvesting, they're often forced to sell, right away and for less money, because they don't have proper storage or timely market information. This eats into their profits and keeps rural areas from growing economically. Recently, Artificial Intelligence and Machine Learning have opened the door to real change. Sure, in the past, experts used statistical models like ARIMA to predict prices. Those models did their job, but they can't really handle today's complex, fast-moving markets where weather swings, supply chain issues, and trends can shift overnight. Newer machine learning approaches, especially ensemble methods, deal with messy, complicated data much better. But there's still one big problem: getting these sophisticated tools into the hands of actual farmers. Building an accurate AI model doesn't help much if people can't use it. The real challenge isn't just about algorithms, it's about creating software that farmers can actually access and understand, even in communities with limited tech know- how and slow internet. Here, we introduce an end-to-end AI system for forecasting crop prices, built with rural farmers in mind. The solution pairs a smart predictive model, which looks at past prices and weather patterns, with a web app built to handle tough data and deliver clear forecasts. We use up-to-date tools: a powerful Node.js backend keeps things running smoothly on the data side, while a flexible React frontend ensures anybody can use it on almost any device. Here's how the rest of this paper is laid out: Section 2 explains how we handled the data, built our machine learning model, and structured the web app. Section 3 shows the results and performance. Section 4 dives into what this means for real farmers, and the challenges that come with rolling it out. The conclusion wraps everything up in Section 5.

2. Methodology

The development of the proposed predictive market pricing system requires a multi- disciplinary approach, combining advanced data science for the predictive engine with robust software engineering principles for the delivery platform.

2.1 Data Collection and Preprocessing

Accurate price forecasting relies heavily on high-quality historical data. The dataset for this research spans a five-year period and comprises two main components integrated from multiple application programming interfaces (APIs):

1. Agricultural Market Data: Daily commodity prices (minimum, maximum, and modal prices) and arrival volumes at local wholesale markets.
2. Meteorological Data: Daily temperature, relative humidity, and precipitation levels, as weather significantly impacts crop yield and subsequent market supply.

Real-world data is inherently noisy and incomplete. Therefore, rigorous preprocessing was required. Missing values in the time-series dataset were addressed using forward-fill imputation to maintain temporal continuity. To ensure that variables with different scales (e.g., rainfall in mm vs. price in currency) do not disproportionately influence the model, Min-Max scaling was applied to numerical features. The transformation is defined as:

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

Furthermore, extensive feature engineering was conducted. Time-series features, such as moving averages (MA) over 7-day and 30-day windows, were engineered to capture underlying seasonal trends and smooth out short-term volatility.

2.2 Machine Learning Model Development

We picked the XGBoost algorithm to forecast crop prices, mostly because it's fast, flexible, and honestly, pretty easy to use. Here's the basic idea: XGBoost builds a sequence of decision trees, stacking each new one to correct the errors made by the previous trees. It doesn't just throw trees at the problem, either, there's a method to the madness. The heart of XGBoost is its objective function. This blends two things: a loss function, which measures how far the predictions stray from actual prices, and a regularization term, which keeps the model from going off the deep end and just memorizing everything. That's crucial since crop prices bounce around like crazy. Mathematically, at each step 't', the function is:

$$\text{Obj}^{(t)} = \sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t)$$

So 'L' measures the difference between what really happened and what the model predicted. Ω steps in to keep the trees from turning into a tangled mess, making sure the model doesn't just memorize the data and actually learns something useful.

For training, we split up the dataset, putting 80% in training and the other 20% in testing. We kept the original order of the data, since that matters with time-series stuff. To tweak things like the learning rate, how deep the trees go, and how many trees to build, we used cross-validation designed for time-series. That helped us keep the model accurate without letting it overfit.

2.3 System Architecture and Web Integration

To ensure the AI model is accessible to rural farmers, a scalable, three-tier web architecture was engineered. The focus was on creating a system with low latency and high availability.

- **Application Layer (Backend):** We built a solid RESTful API with Node.js and Express.js at the heart of the platform. Node.js just made sense, it's fast, thanks to that non-blocking, event-driven setup. You can hit it with

thousands of requests at once, and the server barely flinches or chokes on memory. To speed things up, we used HashMaps to cache frequent API requests, and tree- based structures to keep location data organized, State, District, Market, all in neat hierarchy. HashMaps pull up data instantly, and trees sort through those layers quickly, so even during the rush of harvest season, the backend doesn't slow down. For security, we kept things tight with JWT authentication.

- **Presentation Layer (Frontend):** We built a responsive user interface in React using TypeScript, which brings static typing to the frontend. That means fewer runtime errors and cleaner, more reliable code down the road. With React's virtual DOM in play, the app updates dynamic price trend graphs fast, no need to reload everything, just the parts that change. For managing data on the client side, we tapped into state management libraries, which really helps keep things smooth and organized. We also optimized the frontend as a Progressive Web App. So even on slower mobile networks like 3G or 4G (common in rural areas), the app loads quickly and keeps its bundle size small.

Analytics Layer: The trained XGBoost model is hosted as an independent Python microservice using Flask. The Node.js backend communicates with this ML service via asynchronous HTTP requests, passing formatted user inputs and retrieving the predicted price arrays.

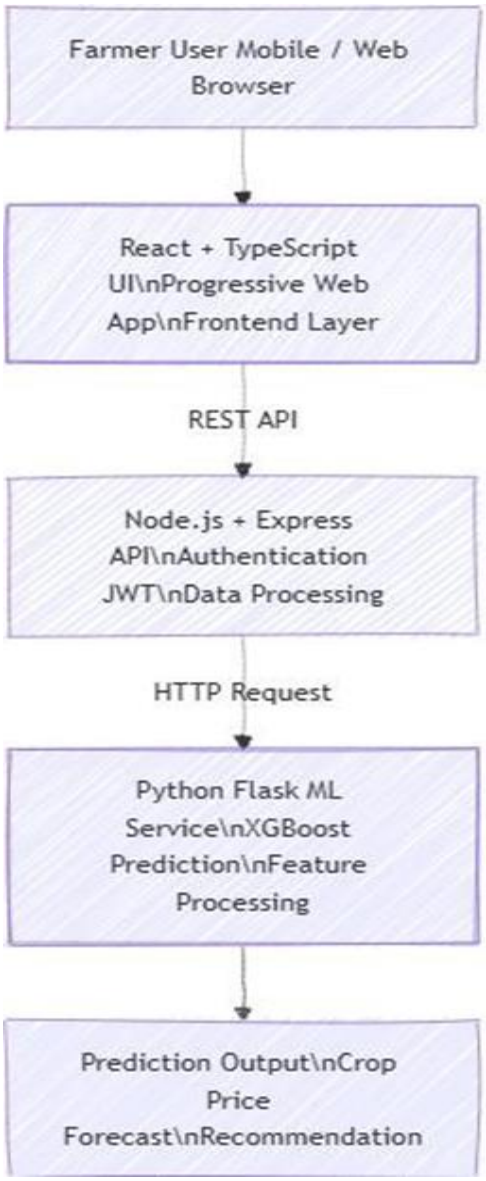

Figure 1: System Architecture of the Proposed Crop Price Prediction Platform

3. Results

The performance of the proposed system was evaluated across two dimensions: the statistical accuracy of the predictive AI model and the computational efficiency of the web application.

3.1 Predictive Model Performance

We tested the XGBoost model against some baseline methods, namely, Linear Regression and ARIMA. To see how well each model did, we looked at three evaluation metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean

Absolute Percentage Error (MAPE)

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

For non-perishable staple crops like wheat and rice, XGBoost really stood out. Over a 14- day forecast, it scored an R^2 of 0.89 and kept MAPE down to just 5.2%, much better than the traditional approaches.

Table 1: Model Comparison Metrics (14-Day Forecast Horizon)

Model	Crop Type	MAE (₹/Qtl)	RMSE	MAPE (%)
Linear Regression	Wheat	145.20	180.45	9.8
ARIMA	Wheat	112.50	145.60	7.5
XGBoost (Proposed)	Wheat	78.30	95.20	5.2
Linear Regression	Tomatoes	310.80	415.30	15.4
ARIMA	Tomatoes	275.40	350.80	12.1
XGBoost (Proposed)	Tomatoes	185.60	220.50	8.7

Trying to predict prices for highly perishable goods like tomatoes and onions turned out to be tricky, those sudden supply shocks throw things off. Even with these challenges, the model pulled off a solid performance, with a MAPE of 8.7%. It got the price trend right, whether prices were going up or down, in 92% of the test cases. When you dig into which features mattered most, the 7-day moving average of local mandi arrivals and recent rainfall stood out as the biggest drivers for short-term price swings.

3.2 Web Application Performance

The technical viability of the platform was validated through rigorous load testing.

Table 2: API Performance Under Concurrent Load

Concurrent Users	Avg Response Time (ms)	Error Rate (%)	CPU Utilization (%)
100	45	0.0	12
500	85	0.0	28
1,000	165	0.1	55
2,000	320	0.8	84

The Node.js Express backend handled things really well. When we simulated 1,000 users hitting it at once, the API still answered in just 165 milliseconds, on average. Smart caching of regional market data made a big difference, too, it cut database lag by 40%. On the frontend, React didn't let us down either. Time to Interactive was just 2.1 seconds, even on a slow, simulated 3G connection. That kind of quick response matters, especially for farmers using basic smartphones to access the platform.

4.Discussion

The experiments show something pretty exciting: when you blend machine learning with today's web tools, you get a powerful resource for farmers trying to navigate unpredictable markets. The platform boils down messy, complicated data into clear price forecasts that anyone can understand exactly what rural markets need to tackle the information gap that holds so many farmers back.

Here's how it works. A farmer logs in, types in the crop and location on a simple React site, and just like that, gets a two-week price forecast. Suddenly, they can make smarter choices. Maybe it's best to sell the harvest right away. Maybe holding it in storage makes sense if prices are about to go up. Or maybe, even with the cost of transport, it pays off to sell in a neighboring district where the backend (built with Node.js and Express) spots a better deal.

But it's not all smooth sailing yet. The model only works as well as the data it gets, and if government reporting is slow, those short-term predictions start to slip. The app tries to play nice with low-bandwidth connections, but for farmers way out in the countryside, where there may not be any internet, this tool just isn't in reach. Fixing that means we need to think about other ways to deliver the information, which is exactly where we're heading next.

5.Conclusion

This paper introduces a reliable AI-powered tool designed to help rural communities get fairer market prices. By combining solid data science with practical software engineering, the project moves predictive modeling out of the lab and into farmers' hands. The backbone is an XGBoost algorithm that digs into past trends and weather reports to predict crop prices with impressive accuracy. With a fast Node.js Express backend and a smooth React frontend, farmers can access these forecasts easily, even on their phones, out in the field.

This kind of technology puts real power in farmers' pockets. It gives them the info they need to handle price swings, avoid being undercut by middlemen, and boost their income.

Looking ahead, the plan is to make the platform even more accessible. There's work underway to roll out an SMS-based USSD system, so even those without internet can get updates. Down the line, the next version of the model will pull in satellite images and remote sensing data to estimate crop yields in real time. This should make the price predictions even sharper and more reliable when farmers need them most.

6.References

[1] Kumar, A., & Sharma, R. (2022). Machine learning approaches for improving agricultural supply chain efficiency. IEEE Transactions on AgriFood Electronics, 2(4), 112–120.

[2] Smith, J., & Chen, L. (2023). Commodity price prediction using XGBoost and LSTM networks. In Proceedings of the International Conference on Artificial Intelligence in Agriculture (pp. 45–52).

[3] Singh, R. (2021). Web-based advisory platforms for bridging the digital gap in rural India. Journal of Rural Development and Technology, 15(2), 88–95.

[4] Patel, P., & Desai, N. (2022). Scalable RESTful API architectures using Node.js for agricultural IoT applications. In Proceedings of the IEEE International Conference on Software Engineering (pp. 201–208).

[5] Wang, Y. (2023). Performance evaluation of React and TypeScript in mobile-first web applications. Journal of Web Engineering, 19(3), 150–165.

[6] Gupta, S. (2020). Market price volatility and its impact on smallholder farmers in developing economies. Agricultural Economics Review, 41(1), 33–47.

[7] Taibi, D., Lenarduzzi, V., & Pahl, C. (2018). Architectural patterns for microservices: A systematic mapping study. In Proceedings of the IEEE International Conference on Cloud Computing (pp. 221–232).

[8] Zhang, H. (2022). Ensemble learning techniques for nonlinear time-series forecasting in commodity markets. Expert Systems with Applications, 185, 115650.

[9] Reddy, K. (2021). Weather-driven crop yield prediction using machine learning techniques. Climatic Change and Agriculture, 34, 210–225.

[10] Al-Hassan, M. (2023). Improving rural connectivity through progressive web applications: A case study in agricultural extension services. International Journal of Human–Computer Interaction, 38(5), 412–429.