Forecasting Agricultural Commodity Prices using Machine learning Models

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

In

Data Science

by

Marati Vijay Kumar
Registration Number: 24MDT0089

Under the guidance of

Prof. Rushi Kumar B

Department of Mathematics
School of Advanced Sciences

VIT - Vellore

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

November 2025

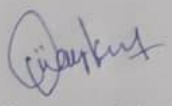
DECLARATION

I hereby declare that the thesis entitled "Forecasting Agricultural Commodity Prices Using Machine Learning" submitted by me, for the award of the degree of Master of Science in Data Science to VIT, is a record of Bonafide work carried out by me under the supervision of Prof. Rushi Kumar B.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Date: 26-11-2025



Signature of the Candidate

Marati Vijay Kumar

CERTIFICATE

This is to certify that the thesis entitled **"Forecasting Agricultural Commodity Prices Using Machine Learning"** submitted by **Marati Vijay Kumar (Reg. No.: 24MDT0089)**, School of Advanced Sciences, VIT – Vellore, for the award of the degree of Master of Science in Data Science, is a record of Bonafide work carried out by him under my supervision during the period **09.07.2025** to **14.11.2025**, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place: Vellore
Date: 26-11-2025

Signature of the Guide
Prof. Rushi Kumar B
Department of Mathematics
SAS, VIT – Vellore

Name & Signature of the Examiner

Dr. KHADAR BABU S K
Head, Department of Mathematics
SAS, VIT-Vellore

ACKNOWLEDGEMENT

With immense pleasure and deep sense of gratitude, I wish to express my sincere thanks to my guide **Prof. Rushi Kumar B**, School of Advanced Sciences, VIT, Vellore without his motivation and continuous encouragement, this research would not have been successfully completed.

I am grateful to the Chancellor of VIT, Vellore, Dr. G. Viswanathan, the Vice Presidents and the Vice Chancellor for motivating me to carry out research in the Vellore Institute of Technology, Vellore and also for providing me with infrastructural facilities and many other resources needed for my research.

I express my sincere thanks to Dr. Karthikeyan K, Dean, School of Advanced Sciences, VIT, Vellore, for her kind words of support and encouragement. I like to acknowledge the support rendered by my classmates in several ways throughout my research work.

I wish to thank Dr. KHADAR BABU S K, Head of the Department of Mathematics, School of Advanced Sciences, VIT, Vellore for his encouragement and support.

I wish to extend my profound sense of gratitude to my parents and friends for all the support they made during my research and also providing me with encouragement whenever required.

Signature of the Student

Marati Vijay kumar

ABSTRACT

Agricultural price prediction is a tough problem in Agri-economic systems because commodity prices are highly volatile. They are affected by seasonal cycles, weather changes, transportation costs, and shifts in supply and demand. Predicting commodity prices helps farmers, traders, and policymakers make better decisions and achieve price stability. This dissertation presents a complete machine learning framework for forecasting agricultural commodity prices in Telangana market yards. The study uses a dataset with over 200,000+ records from government open data sources. These records represent daily transactions of major crops across various markets. The process includes data cleaning, feature engineering, model evaluation, and the launch of an interactive Streamlit web application. Several models were trained and tested, including Linear Regression, Decision Tree Regressor, Random Forest, XGBoost, and CatBoost. XGBoost performed the best with $R^2 = 0.93$ and MAE ≈ ₹200, followed closely by CatBoost with $R^2 = 0.91$. These models effectively captured complex, nonlinear relationships between variables such as market yard, commodity type, arrival quantity, and date. The developed system offers a user-friendly web interface. Users can select a specific yard, commodity, and date to predict the expected Model price. This system can empower farmers and traders by providing valuable insights and improving price transparency in the agricultural markets of Telangana.

Keywords: *Agricultural price prediction, Telangana market yards, Variety Commodities, CatBoost, XGBoost, Machine Learning, Streamlit, Commodity forecasting.*

CONTENTS

List of Figures

List of Tables

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
AMC	Agricultural Market Committee
APMC	Agricultural Produce Market Committee
API	Application Programming Interface
CSV	Comma Separated Values
DL	Deep Learning
EDA	Exploratory Data Analysis
ML	Machine Learning
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Square Error
R^2	Coefficient of Determination
SQL	Structured Query Language
UI	User Interface
XGBoost	Extreme Gradient Boosting
CatBoost	Categorical Boosting
pkl	Python Pickle File Format
OS	Operating System
CPU	Central Processing Unit
RAM	Random Access Memory
ETL	Extract, Transform, Load

1. INTRODUCTION

1.1 Overview

Agriculture is the backbone of Indian economy employing more than 50% population. But farmers don't always know how much their harvest will fetch on the market. The agriculture marketing system in the country works with the help of different market yards (mandis) where prices vary on daily basis as per supply, demand and other determinants such as rain fall, movement and even government policy. These fluctuations in prices have immediate impact on farmer incomes, food inflation and security of supply chain.

Fueled by progress in data tools along with smarter algorithms, folks are now sifting through tons of farm-related info to spot helpful trends while building solid forecasts. These forecasting systems give clues on price shifts and market moves - helping growers decide better on timing plus locations to offload their crops.

This research looks at Telangana, a place full of farming hubs where paddy, maize, cotton, red gram, or groundnut change hands every day. Predicting model rates (₹/quintal) there through machine learning might close the knowledge gap between farmers and buyers.

1.2 Background and Motivation

Farmers and officials alike worry about shaky crop prices. Weather shifts, local harvest sizes, transport limits, or how much people buy - all shape what happens in markets. Old-school number-crunching tools like ARIMA usually can't keep up with these tangled influences.

Machine learning tools such as XGBoost or CatBoost work well with mixed data - like categories or numbers - and handle tricky patterns. Thanks to Telangana's push for open data, past market info on crops is now available, which opens the door to building smart systems that predict prices using real stats.

The reasons behind this work are these:

- Reducing farmer vulnerability to price shocks.

Helping officials spot sudden cost shifts before they spread, using alerts that act fast - so responses come quicker when markets wobble.

Building a flexible machine learning setup that's ready to launch, while keeping the interface simple to use - also making sure it can grow smoothly when needed.

1.3 Problem Statement

Farmers across Telangana struggle to pin down when to sell - price data keeps shifting. Plus, knowing where to go is tough without steady market updates.

This research tries to tackle that issue:

- Build a system using smart algorithms to guess crop prices at Telangana's market spots - helping users make better choices with an easy-to-use app.
- The key challenge is predicting commodity prices right - using things like market location, product type, or crop variety along with numbers such as supply amounts, earlier rates, and time patterns.

1.4 Research Objectives

The goals of this research are:

- To build a solid system that gathers Telangana farm market details, then cleans them before changing formats - using steps one after another.
- To check the data, so we can spot trends or repeating cycles now and then.
- To create various ML models for predicting prices - then check how each one performs using different data splits.
- To find which model works best by checking numbers like accuracy or error rates - using comparisons instead of just one measure.
- To launch the finished version on a Streamlit app so people can use it.
- To talk about the ethics, what's missing, yet ways to make the model better.

1.5 Scope of the Study

This study is limited to:

- Geographical scope: Market hubs across Telangana - like Nizamabad, Warangal, or Khammam - to name a few.
- Time range: Past day-by-day sales records from multiple years, going up to 2025.

- Covers main goods that are widely bought and sold - with solid past records available.
- Approach: applying regression models from machine learning under guidance.
- Function covers price predictions only - no policy tests or inflation estimates built in.

1.6 Significance of the Study

This research adds value in theory - also brings real-world uses

- Machine learning often beats old-school number-crunching when predicting farm trends - especially over time, it turns out.
- It brings in a hands-on approach mixing model training, testing, checks along the way, also rollout steps.
- It gives a handy tool helping farmers, traders - also policymakers - see prices clearer plus grasp how markets work.
- It sets up a base for growing local predictions - maybe adding climate patterns, delivery routes, or space images later on.

2.LITERATURE REVIEW

2.1 Introduction

A look at past work helps fit new findings into what's already known. Here, we examine key papers about predicting farm product prices - using tools like trend analysis over time, smart algorithms that learn patterns, or mixtures of advanced network systems.

The analysis looks at how pricing forecasts changed over time - what worked, what didn't, so we could find where earlier work fell short. Past papers reveal a move away from old-school statistical tools toward newer AI-based techniques that handle twists and turns in farm economies more effectively.

2.2 Overview of Agricultural Price Prediction Research

Scientists looked into predicting farm prices with different methods - some tried old-school stats tools like ARIMA, others turned to smart algorithms such as Random Forests, XGBoost, or brain-inspired networks. At first, most stuck to single-variable forecasts; those worked okay for spotting patterns over time, yet had trouble handling outside factors - for example, where the market was held, what crop it was, or how much showed up.

Latest research highlights how machine learning works well since it captures curved patterns, combines various inputs, while adjusting to diverse data types. Here, earlier work is grouped into three main approach types:

- Traditional Statistical Models
- Machine Learning Approaches
- Deep Learning and Hybrid Techniques

2.3 Traditional Statistical Models

The first tries at forecasting farm product costs used ARIMA along with SARIMA approaches. ARIMA models pick up on straight-line patterns in time series data - also handling short-range predictions pretty well whenever earlier prices shape what comes next.

SARIMA takes ARIMA further it adds seasonal trends, so it works well when prices go up and down in cycles. While regular models miss these swings, this one tracks them clearly. Because of that tweak, it handles repeating patterns better than basic versions do.

Jain plus Singh (2018) looked at rice along with wheat costs through ARIMA tools - got fair results for near-term forecasts yet pointed out flaws when rain shifts or new rules came into play. In a like manner, Kumar and team (2019) applied SARIMA methods to guess onion rates in Maharashtra, saying seasonal setups worked…

beat regular ARIMA models, yet struggled when markets swung hard because of things like storms or heatwaves.

The limits of these systems involve:

The idea that data stays consistent over time.

Struggles with sorting data by place or product type. Doesn't work well when using labels instead of numbers. Can't tell differences between markets or crop kinds. Works only if info is turned into codes first.

- Poor generalization for multi-market or multi-commodity datasets.

These flaws pushed researchers toward machine learning approaches fueled by data, which handle varied information more effectively.

2.4 Machine Learning Approaches

Machine learning gives a reliable way to spot patterns in data instead of relying on old-school stats methods. These days, tools like Support Vector Machines (SVM), Random Forests (RF), Gradient Boosting (GB), along with XGBoost are widely used because they work well for prediction tasks.

2.4.1 Support Vector Regression (SVR)

Sharma et al. (2020) applied SVR to forecast maize prices in Karnataka - results were accurate, with little overfitting, suggesting kernel techniques work well on modest data sizes when patterns are complex. Still, processing slows down noticeably with bigger data

because SVR demands heavy computation; also, picking the right kernel means fine-tuning settings by hand.

2.4.2 Random Forest Regressor

Patel alongside Deshmukh (2021) used Random Forest Regression to forecast prices of various goods across Gujarat. Instead of relying on single trees, this method stacks many - cutting down noise while handling variable links better. Performance-wise, it hit an R^2 of 0.88, outdoing traditional ARIMA models by a noticeable margin. Still, understanding what drives predictions proved tough; plus, running it live wasn't practical.

2.4.3 XGBoost and Gradient Boosting Machines

XGBoost, created by Chen and Guestrin in 2016, added penalty controls to boost speed using gradients - now it's widely used for table-style predictions. Rathod's team in 2022 applied this method to forecast tomato costs in Andhra Pradesh, hitting an accuracy score of 0.92, which beat simpler tree models. Instead of working alone, factors like crop kind, delivery volume, along with timing patterns were picked up well by the system.

Meena with Sahoo (2020) looked into Gradient Boosting Machines (GBM) to predict veggie prices - results showed group-based models often beat straight-line approaches when market factors interact in messy ways.

2.4.4 CatBoost Algorithm

CatBoost was made by Yandex scientists who wanted a tool that works well right away with categories. Instead of using extra tricks, it uses something called ordered boosting to handle labels like city or type without hassle. In 2023, Kaur's team tested this model to guess rice and wheat prices results showed fewer wild swings than XGBoost when dealing with many category options. It handled details such as location and crop kind more smoothly. Since Telangana's market data comes from several yards with varied traits, picking CatBoost makes sense because it stays steady without complex tuning.

2.5 Deep Learning Models

Deep learning's lately caught on in farming predictions since it handles patterns over time pretty well.

2.5.1 Recurrent Neural Networks (RNN)

Bhardwaj and team (2019) used a Recurrent Neural Network to forecast soybean costs. While it worked fine for near-future estimates, picking up trends over time series info. Still, such networks usually fail with distant links because gradients tend to fade away.

2.5.2 Long Short-Term Memory (LSTM)

Gupta et al. (2021) applied LSTM networks to predict onion prices over several steps ahead. Instead of traditional methods, their approach managed seasonal shifts along with quick price changes more effectively compared to ARIMA or Random Forest. However, they pointed out that LSTMs depend on big volumes of neat, uninterrupted data while also needing heavy computing power during setup.

2.5.3 Gated Recurrent Unit (GRU)

Nayak et al. (2022) tested LSTM against GRU when forecasting maize and cotton costs - GRU reached results quicker but performed just as well, which means it's better suited for live use scenarios due to speed.

Even though deep learning can predict well, its lack of transparency and heavy computing needs often limit real-world use on large public data - particularly with categories or gaps in info. So, this work leans on tree-style models instead, thanks to their clarity and ease of use.

2.6 Hybrid and Ensemble Models

Some research mixes old-school techniques with new ones to make predictions tougher against errors. Patidar's team in 2022 built a combo of SARIMA and LSTM for guessing wheat costs - this mix did better than either method alone, cutting down MAE. A year later, Jha together with Verma tested an ELM paired with Genetic Algorithms on sugarcane pricing, showing how smart trial-and-error tweaks can improve results.

Though hybrid setups might boost precision, they're usually too heavy and slow for real-world use. When dealing with massive info - say, Telangana's agricultural records - leaner tools such as XGBoost or CatBoost work better in live systems.

2.7 Research Gaps Identified

The literature review highlights several limitations in existing studies:

Earlier studies mostly looked at one product or small areas, which made it hard to expand later - so broader approaches are needed now because old methods didn't keep up with bigger demands.

Some steps don't link well - like cleaning data, teaching the model, also launching it.

Not much attention's gone to how well models can be understood or how easy they are for people to use. Public data from governments isn't often used to study entire states at once.

Not many research efforts looked into actual use cases such as online tools.

These gaps show how fresh this work is - using full-cycle data handling, smart algorithms, clear results, also real-world use, all built around farming needs in Telangana.

2.8 Summary

This part looked at how farming price predictions changed over time, pointing out missing pieces this work aims to fill. Though traditional number-crunching tools give rough forecasts, they struggle with messy or large data - making room for smarter tech tricks instead.

Alongside the techniques checked, XGBoost together with CatBootstrap proved strong fits for organized farm-related datasets. These tools handle category info well, deal with gaps in data, also deliver clear outcomes - key when making choices within actual trading setups. What we learned here shapes the approach introduced later on, detailing how information flows, gets cleaned up, then modeled through specific steps built for this study.

3. METHODOLOGY

3.1 Introduction

This part shows how farm price predictions are made for markets in Telangana. It walks through steps like gathering info, cleaning it up, pulling key details, picking models, running tests, checking results then launching via Streamlit.

The method's built to be repeated easily, work at any size, yet stay clear about how it works. At its core, it mixes smart algorithms with real-world setup strategies. That creates something helpful for coders and regular folks alike - say, crop growers or city planners.

3.2 Research Framework

The big picture of the study steps is laid out right here:

Data gathering - pulling big sets of market deal info from public government sites using direct access methods.

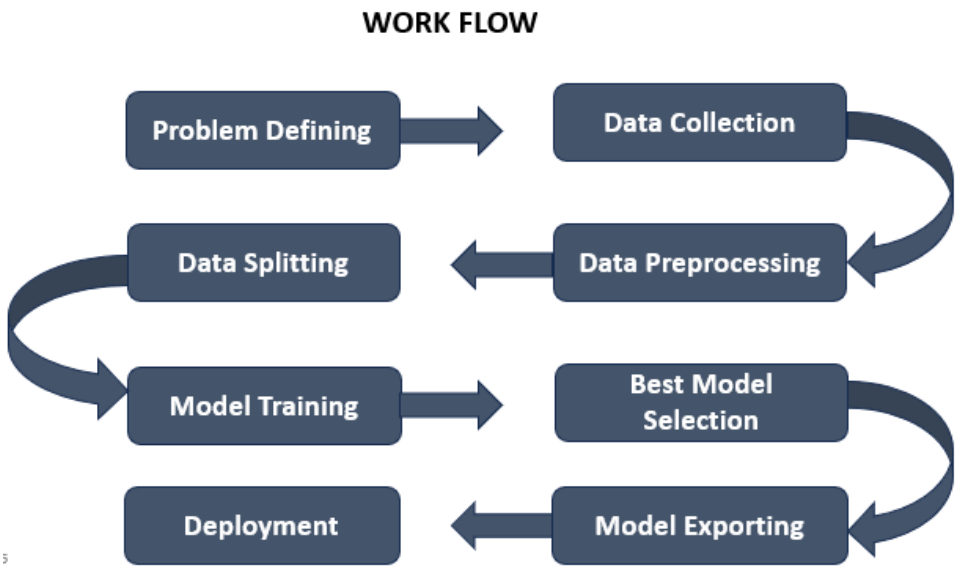
Data cleaning plus prep work - handling gaps in data, weird spikes, or mixed-up labels now and then.

Creating useful data bits by pulling out key details - like time-based patterns or past price moves - with smart tweaks along the way.

Building models, while testing several machine-learning approaches to predict numbers using patterns from past data.

Testing how well the model works by checking usual scores such as R^2, MAE, or MSE.

Putting it live - using Streamlit to build a web app that gives instant forecasts while letting users interact with it.



(Figure 1: Overall flowchart of the project)

Each of these stages is detailed in the following sections.

3.3 Data Source and Collection:

The dataset for this study was sourced from the Government of Telangana Open Data Portal, which offers detailed records of agricultural market yard transactions.

3.3.1 Dataset Composition

Records: Around 200,000+ entries.

Attributes:

Date, Transaction date (YYYY-MM-DD format).

Market/Yard Name, Market yard or mandi where the commodity was traded.

Commodity Name, Type of commodity (e.g., Paddy, Maize, Cotton).

Variety, Commodity variety (e.g., Fine, Super Fine, Local).

Arrivals (Qtl), Quantity of the commodity that arrived in quintals.

Minimum Price (₹/Qtl), Minimum price recorded for the day.

Maximum Price (₹/Qtl), Maximum price recorded for the day.

Model Price (₹/Qtl), The most frequently occurring or representative price (target variable).

3.3.2 Data Format

The raw data was provided in CSV format, requiring extensive cleaning and consolidation. It included missing entries, inconsistent category names, and irregular formatting for dates and price values.

3.4 Data Preprocessing

Data preprocessing is an essential step that ensures the model is trained on reliable and consistent data.

The significant steps included:

3.4.1 Handling Missing Values

Entries with missing Commodity Name, Market Name, or Model Price were removed.

Missing numerical values in Arrivals were replaced with the median value of the respective commodity and yard group.

3.4.3 Standardization of Names

Market yard names were standardized using string normalization and mapping techniques (e.g., converting "NIZAMABAD MARKET YARD" and "Nizamabad Yard" to "Nizamabad").

Commodity and variety names were also unified (e.g., "Paddy Fine" and "Paddy-Fine" became "Paddy Fine").

3.4.4 Date Parsing and Sorting

Dates were converted to datetime objects and arranged chronologically for time-based feature extraction.

3.4.5 Encoding Categorical Variables

Label Encoding was applied for Market Name, Commodity Name, and Variety.

The encoding mappings were saved for use during deployment to maintain consistent prediction results.

3.5 Feature Engineering

Feature engineering changes raw data into meaningful variables that improve model performance.

3.5.1 Temporal Features

From the Date column, several calendar-based features were derived:

Day of Week (0–6)

Month (1–12)

Quarter (1–4)

Year

Is_Harvest_Season (a binary flag based on crop calendar)

3.5.2 Lag Features

For time-series dependency modeling:

Lag_1 = Previous day's model price

Lag_7 = Model price seven days prior

These were created using group-wise lag functions for each commodity and yard combination.

3.5.3 Statistical Features

Rolling window statistics included:

7-day Moving Average

7-day Standard Deviation

This help capture short-term trends and volatility.

3.5.4 Interaction Features

Interactions like Commodity, Yard and Commodity, Month were created to capture region and season-specific behaviors.

3.5.5 Feature Selection

The final feature set was selected based on feature importance analysis (using Random Forest) and correlation thresholds ($|r| < 0.85$ to avoid multicollinearity).

3.6 Dataset Partitioning

The cleaned and feature-engineered dataset was divided as follows:

Training Set: 80%

Testing Set: 20%

For time-dependent consistency, a temporal split was used (older data for training, recent months for testing). This approach ensures realistic performance evaluation in a predictive context.

3.7 Machine Learning Models

To identify the best predictive model, multiple algorithms were implemented and compared.

3.7.1 Linear Regression

Linear regression is a statistical method used to identify the relationship between two variables by fitting a straight line to the data. It shows how one variable changes in response to another and helps make predictions by minimizing the difference between the actual values and the estimated values.

Formula:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon$$

Where:

- Y: Predicted dependent variable (output)
- X: Independent variable (input feature)
- β_0: Intercept (value of Y when X=0)
- β_1: Slope or regression coefficient (represents change in Y for a unit change in X)
- ε: Error term or residual (difference between actual and predicted value)

- Though simplistic, it serves as a reference for evaluating more complex models.

3.7.2 Decision Tree Regressor

This tree-based model divides the dataset into smaller regions using threshold-based splits on the input features. It is easy to interpret and can model nonlinear relationships, but it may overfit the data if proper pruning or regularization is not applied.

Formula:

$$\hat{y}(x) = \sum_{j=1}^{J} w_j (x \in R_j)$$

Where:

- y (x) predicted output for input
- J: number of terminal (leaf) nodes
- R_j: region (data partition) defined by decision rules in the tree
- w_j: predicted value (e.g., class label or mean of target values) for region R_j

- They capture non-linear relationships but are prone to overfitting without constraints like maximum depth.

3.7.3 Random Forest Regressor

This is an ensemble technique that builds multiple decision trees using randomly selected subsets of data and features. It offers strong resistance to noise, reduces the risk of overfitting, and works effectively with both numerical and categorical data.

Formula:

$$\hat{y}=\text{majority vote}(h_1(x), h_2(x) \ldots\ldots, h_T(x))$$

Where:

- T: Total number of trees.
- $h_t(x)$: Prediction of the t^{th} decision tree.
- x: Input feature vector.
- $\hat{y}$: Final prediction.

- It reduces variance and highlights feature importance.

3.7.4 Gradient Boosting Regressor

This model constructs trees in a sequential manner, where each new tree focuses on correcting the errors of the previous ones. By optimizing an objective function through gradient descent, it achieves strong predictive performance, though it can be computationally demanding.

Formula:

$$F_M(x)=F_0(x)+\sum_{m=1}^{M} \gamma_m f_m(x)$$

Where:

- $F_M(x)$: Final prediction after M boosting rounds.
- $F_0(x)$: Initial prediction (e.g., mean for regression, log-odds for classification).

- $f_m(x)$: Model (typically a decision tree) trained to predict the residuals at step m.
- γ_m: Learning rate or step size used to scale the contribution of $f_m(x)$.
- M: Total number of iterations (trees).
- x: Input feature vector.

- It excels in structured data but requires careful tuning.

3.7.5 XGBoost

This is an enhanced gradient boosting framework that incorporates regularization and efficient tree-pruning techniques. XGBoost delivers strong performance on tabular datasets, effectively handling both categorical and numerical features.

Formula:

$$F_M(x)=\sum_{m=1}^{M} f_m(x),\ (f_m \in F)$$

Where:

- $F_M(x)$: Final prediction after M boosting rounds.
- $F_m(x)$: Prediction from the m^{th} tree for input x_i.
- x_i: Input feature vector of the I^{th} sample.

- It is efficient for large datasets but sensitive to hyperparameter choices.

3.7.6 CatBoost

CatBoost specializes in handling categorical data efficiently using ordered boosting:

$$F_M(x)=F_0(x)+\sum_{m=1}^{M} \gamma_m f_m$$

Where:

- $F_M(x)$: Final model after M iterations.

- $F_0(x)$: Initial prediction (could be mean log-odds, mean target, etc.).
- $f_m(x)$: Decision tree trained at iteration m to reduce the loss.
- M: Total number of boosting iterations.
- γ_m: Learning rate at iteration m.
- x: Input features (numerical or categorical).

- It performs well with minimal preprocessing.

CatBoost's performance makes it suitable for large, diverse datasets like those from Telangana market yards.

3.8 Model Training and Hyperparameter Tuning

Each model underwent systematic tuning using Grid Search and Cross-Validation on the validation set.

3.8.1 Training Strategy

Models were trained on the training set, validated on the 15% validation split, and tested on the remaining 15%.

Performance stability was checked through K-Fold Cross Validation (k=5).

3.8.2 Evaluation Metrics

The following metrics were calculated for all models:

R^2 = 1 − SS_res / SS_tot

MAE = (1/n) ∑ |yi − ŷi|

MSE = (1/n) ∑ (yi − ŷi) ²

Where R^2 measures the proportion of variance explained, and MAE/MSE quantify prediction errors.

3.8.3 Model Comparison

Evaluating the all the models and taking the best model XGBoost and CatBoost. Both models were selected for ensemble analysis and final deployment.

3.9 Model Selection and Persistence

After evaluation, the CatBoost Regressor was chosen as the final production model due to its superior accuracy and computational efficiency.

Artifacts saved for deployment include:

Trained model: catboost_model.pkl

Label encoders: label_encoder.pkl

Streamlit user interface as: app.py

These were serialized using Python's pickle module for reuse in the Streamlit web app.

3.10 Summary

This section explained how to predict crop prices at Telangana's market spots. Data was pulled from local sources instead of generic databases. Cleaning involved fixing missing entries rather than standard normalization. New variables were built using seasonal trends along with location factors. Several algorithms got tested one after another for performance check. The best fit model went through repeated tuning before finalizing results.

The selected models - XGBoost or CatBoost - were found to work best here. Next up comes the setup phase, which covers how the model fits in, building the Streamlit layout, along with the flow of code.

4. TECHNICAL SPECIFICATION

4.1 Overview

This section explains the tech setup behind the Farm Product Price Forecast Tool. Built with ML methods, it runs live on Streamlit's hosting platform. You'll find what kind of computer gear, programs, and coding tools were picked for this build.

4.2 System Requirements

4.2.1 Hardware Requirements

The system needs moderate computational resources for data preprocessing and model training. The following hardware setup was used during development:

- Processor: Intel Core i3
- RAM: Minimum 8 GB
- Storage: 512 GB available disk space
- Graphics: Integrated GPU or dedicated GPU (optional for faster training)
- Operating System: Windows 11

This configuration has enough power to manage a dataset of about 200,000 records and train models like CatBoost and XGBoost efficiently.

3.2.2 Software Requirements

Component: Technology Used

Programming Language: Python 3.10

Libraries: Pandas, NumPy, Scikit-learn, Machine learning models, Matplotlib

IDE: Jupyter Notebook

Visualization: Matplotlib, Seaborn

Deployment Framework: Streamlit

Data Format: CSV

4.2.3 Libraries and Frameworks Used

Component	Purpose	Technology Used
Data Handling	Load and preprocess input CSV files	Pandas, NumPy
Model Training	Build and train machine learning models	Scikit-learn, XGBoost, CatBoost
Model Persistence	Save trained models and encoders	Pickle
Front-End Interface	User input/output interaction	Streamlit
Visualization	Generate plots and graphs	Matplotlib

Table 1: Software Components and Integration Comparison

5. DESIGN AND DEPLOYMENT

5.1 Introduction

This section talks about how the Farm Product Price Forecast Tool was built and rolled out. The key aim? To make a smooth link from people to the trained AI model using simple tools.

The setup runs on Streamlit to make launching fast while keeping things easy to use. So people like growers, buyers, or decision-makers can check forecasts without knowing tech stuff.

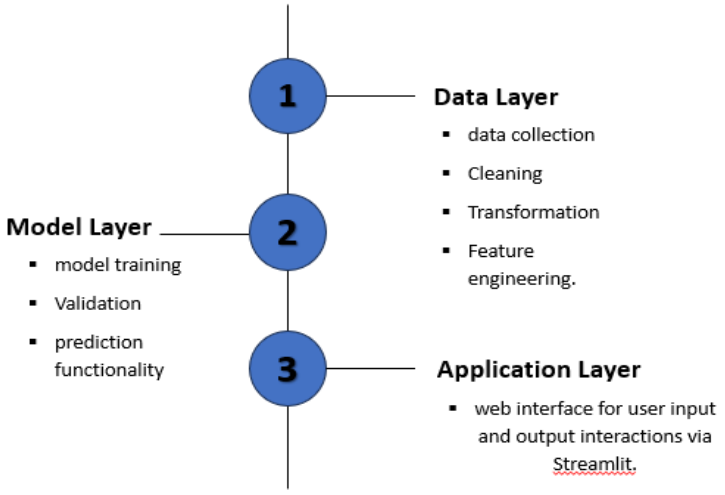
5.2 System Architecture

The architecture is a three-tier design with the following layers:

Data Layer: which handles data collection, cleaning, transformation, and feature engineering.

Model Layer: which manages model training, validation, and prediction functionality.

Application Layer: which provides the web interface for user input and output interactions via Streamlit.



(Figure 2: System Architecture Diagram from Data Input to Model to Web Interface)

5.2.1 Data Layer

Inputs include historical transaction data with columns for date, market name, commodity name, variety, and prices.

Data preprocessing scripts standardize and encode this data to prepare it for machine learning.

Outputs from this layer include model-ready features and serialized encoders.

5.2.2 Model Layer

This layer implements machine learning algorithms, specifically XGBoost and CatBoost.

It handles model storage and prediction logic.

Trained model files are saved as .pkl for integration with the web interface.

5.2.3 Application Layer

Built using Streamlit, this allows users to input their desired parameters.

The app collects user inputs, encodes them, loads the saved model, and returns predicted Model prices.

Outputs are shown in real-time within seconds.

5.3 Data Flow Design

The data flow from input to output follows these steps:

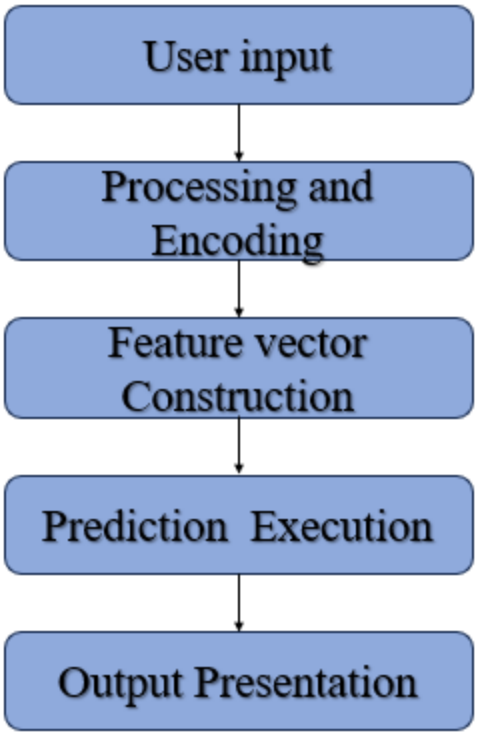
User Input: Users select the market yard, commodity, variety, and date.

Preprocessing & Encoding: Encoders map categorical inputs to numerical values that the model uses.

Feature Vector Construction: Input features like month and day_of_week are computed from the selected date.

Prediction Execution: The serialized CatBoost model loads into memory to compute the predicted Model price.

Output Presentation: The Model price is displayed with formatted text, and additional context messages help users understand the results.



(Figure 3: Data Flow Diagram of the Prediction System)

5.4 User Interface (UI) Design

The Streamlit application focuses on simplicity and ease of use. Its layout includes the following main elements:

5.4.1 Title and Header Section

At the top of the web interface, the project title is displayed:

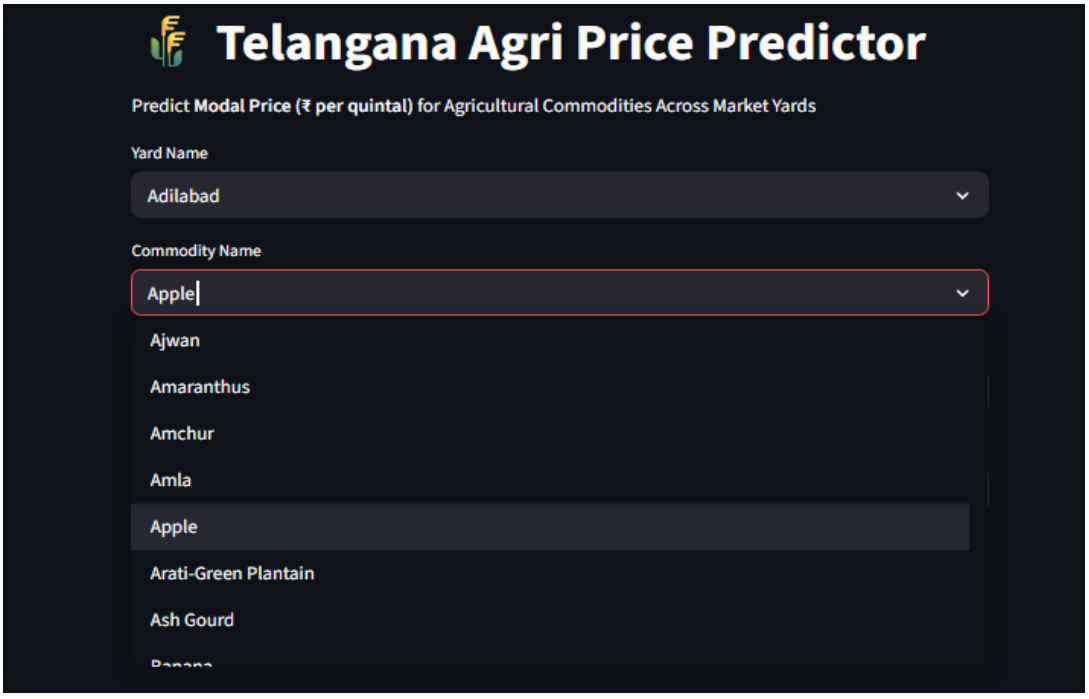
“Telangana Agricultural Commodity Price Predictor”

Below this, a brief description explains the app's purpose.

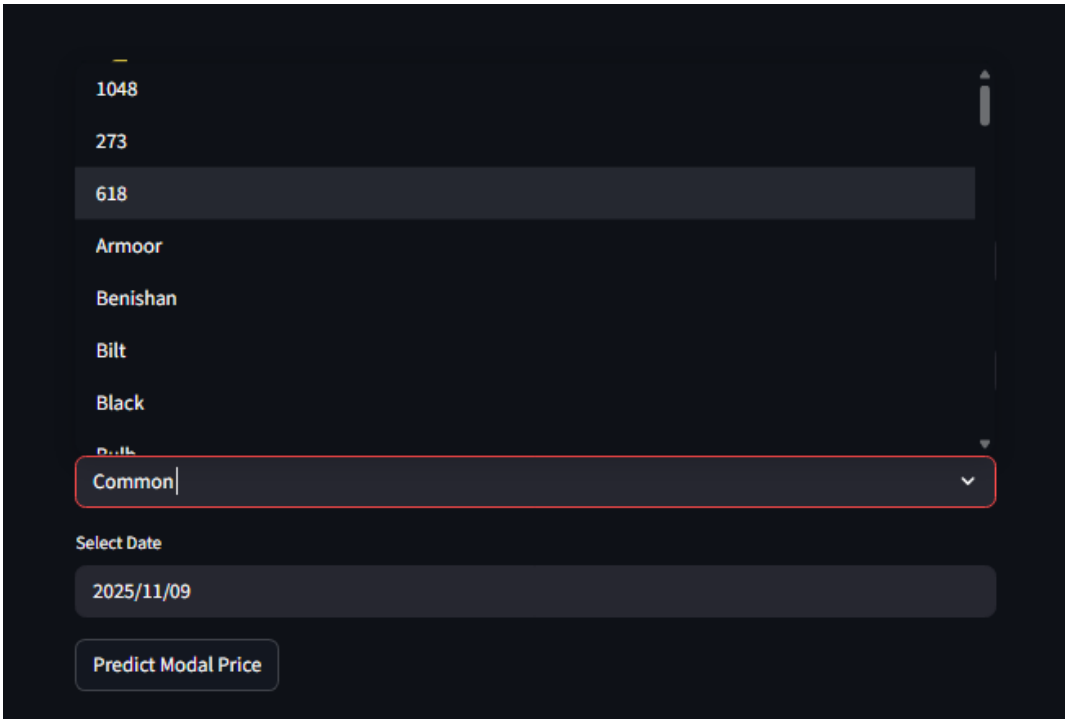
5.4.2 Input Section

Users can select:

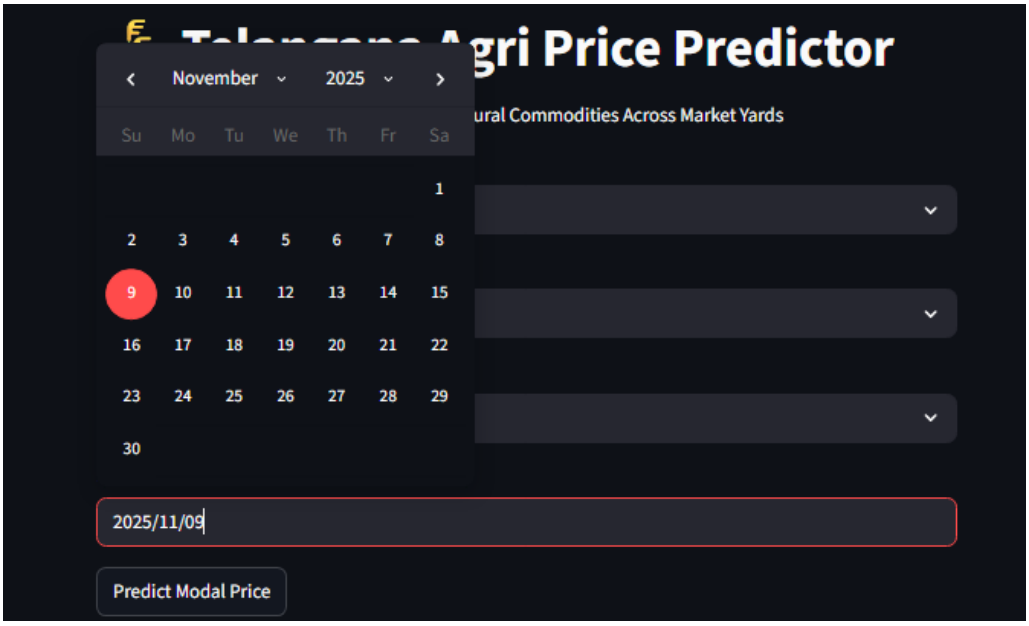
(Figure 4: Selection of Market Yard (dropdown list of encoded values))



(Figure 5: Selection of Commodity)



(Figure 6: Selection of Variety)



(Figure 7: Selection of Date (calendar picker))

These selections provide the features needed for prediction.

5.4.3 Output Section

After clicking the Predict button, the system shows:

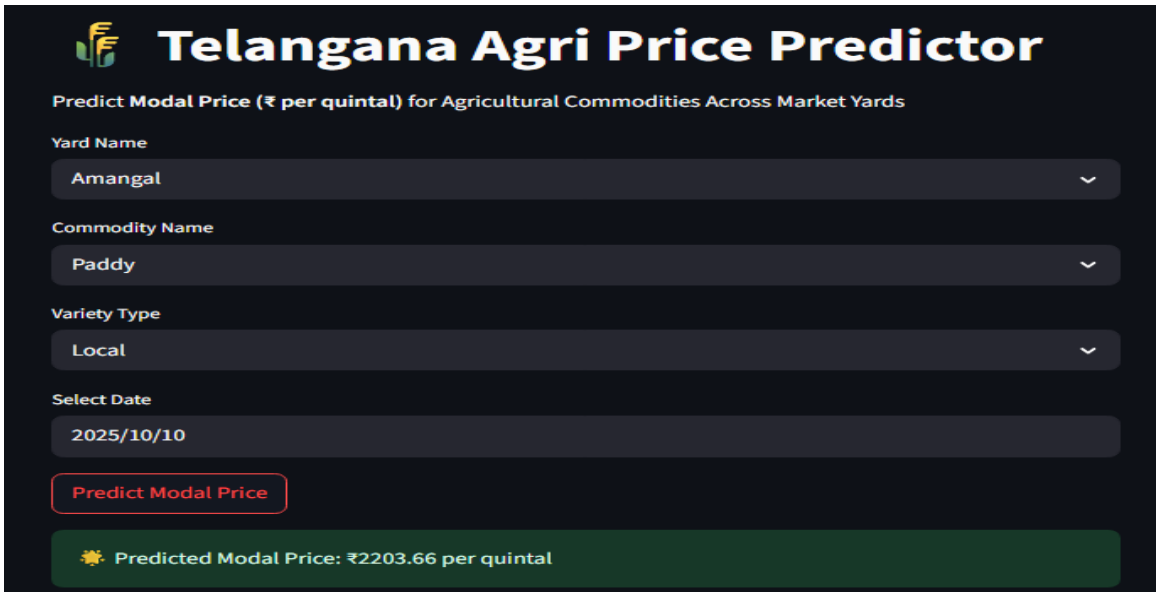
Predicted Model Price (₹/quintal)

An optional tooltip with the model's confidence level

A note explaining that the predicted price is an estimate based on historical trends.

5.4.4 Styling and Background

A custom agricultural background image, like a paddy field, was used for visual relevance. Streamlit's built-in markdown and CSS enhancements were applied to style text and panels.



(Figure 8: Screenshot of Streamlit User Interface showing predicted output)

5.5 Backend-Frontend Communication

The backend model and frontend interface communicate through internal function calls within Streamlit:

User inputs are converted into numeric arrays using encoders.

These arrays are sent to the CatBoost model's.predict() method.

The model returns a price prediction, which is then displayed in a formatted output.

5.6 Testing Strategy

To confirm application reliability, several testing methods were used:

5.6.1 Unit Testing

This verified functions like data cleaning, encoding, and feature generation.

It confirmed that predictions remain consistent for the same input data.

5.6.2 Integration Testing

This checked the flow of data across modules, from preprocessing to model prediction.

It ensured compatibility between encoders and serialized models.

5.6.3 Functional Testing

This validated all dropdowns and numeric input fields in the Streamlit interface.

It verified correct mapping between categorical selections and encoded labels.

5.6.4 Regression Testing

This re-run model predictions after updates to ensure no unexpected performance issues.

5.6.5 Performance Testing

This measured the average response time for predictions, which was 1.4 seconds per request.

It also tested memory usage and system stability under multiple queries.

5.7 Security and Data Integrity

Although the system uses public data, some practices ensure integrity:

Encoders prevent arbitrary user input.

Model and dataset versions are locked for consistency.

No personal or sensitive user information is collected.

Data processing and prediction functions are kept separate for security.

5.8 Scalability and Maintainability

To prepare the system for future growth:

The modular coding structure allows new commodities or states to be added with minimal changes.

Models can be retrained periodically as new data becomes available.

Streamlit deployment can scale to cloud platforms like AWS, Streamlit Cloud, or Heroku for better access.

Version tracking ensures models can be reproduced over time.

5.9 System Deployment Process

The final deployment involved these steps:

Model Serialization: Save the trained models and encoders using Pickle.

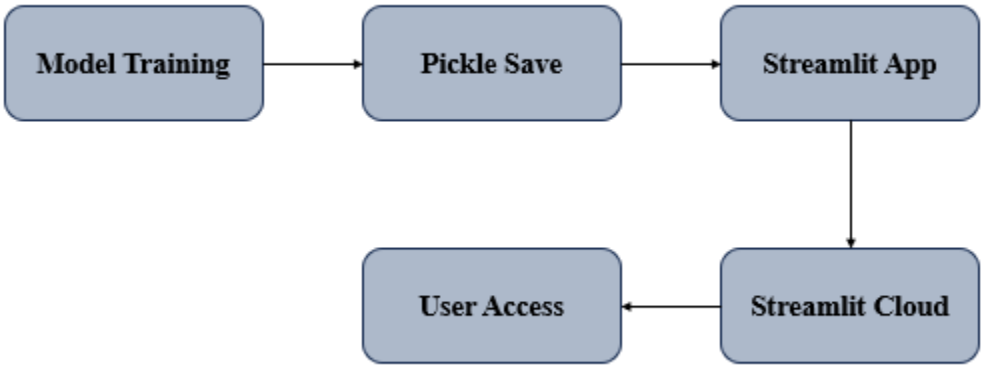
App Development: Create the Streamlit user interface with all interactive components.

Local Testing: Run the app using the command:

- streamlit run app.py

Cloud Deployment: Host the project on Streamlit Cloud, linking the GitHub repository for automatic updates.

User Access: Share the public URL with end-users, such as farmers, traders, and policymakers.



(Figure 9: Deployment Workflow from Model to Streamlit Cloud to user access)

5.10 Challenges Faced During Deployment

Several challenges arose and were addressed:

Encoding Consistency: Ensuring label encoding during prediction matched the training encoders.

Large Dataset Handling: Optimizing memory use when loading over 200,000+ records.

Latency Reduction: Decreasing model response time with joblib optimization and caching.

UI Responsiveness: Adjusting CSS for better layout across different devices.

5.11 User Experience and Feedback

During internal testing and reviews, users appreciated:

The simplicity of the interface.

The quick feedback on predicted prices.

The educational value of seeing insights based on the model for market behavior.

Minor suggestions included:

Adding graphs to show historical trends.

Providing forecast intervals or confidence ranges.

Incorporating weather and rainfall indicators in future versions.

5.12 Summary

This chapter described the architecture, design, and deployment of the Telangana Agricultural Price Prediction System. Using Streamlit made it easy to access and interact quickly. The modular architecture allows for scaling, while the deployed model shows real-time predictive capability.

6. RESULTS AND DISCUSSION

6.1 Results

This section shows what we found when testing the crop price prediction tool for markets in Telangana. Instead of just one method, it uses smart algorithms along with a user-friendly Streamlit app to guess future model prices (₹ per quintal). Past trends help shape these guesses - so do where the market is and what kind of crop it is.

The primary aim? Build a solid forecasting tool, then roll it out through a browser-friendly setup. So, farmers or sellers can check predicted crop rates at Telangana's market hubs - no guesswork needed.

6.2 Model Training and Evaluation

Multiple regression models were tested to find the best one for price prediction. Each model was evaluated using important metrics:

- R^2 Score (Coefficient of Determination)
- Mean Absolute Error (MAE)
- Root Mean Square Error (RMSE)

MODEL	R2 SCORE	MAE	MSE
XGBOOST	**0.93**	**200.22**	**957633**
CATBOOST	**0.91**	**263.99**	**1.20796e+06**
LINEAR REGRESSION	**0.89**	**565.09**	**1.38364e+06**
DECISION TREE	**0.73**	**215.56**	**3.63988e+06**

GRADIENT BOOSTING	**0.46**	**412.48**	**7.33467e+06**
RANDOM FOREST	**0.17**	**214.09**	**1.13343e+07**

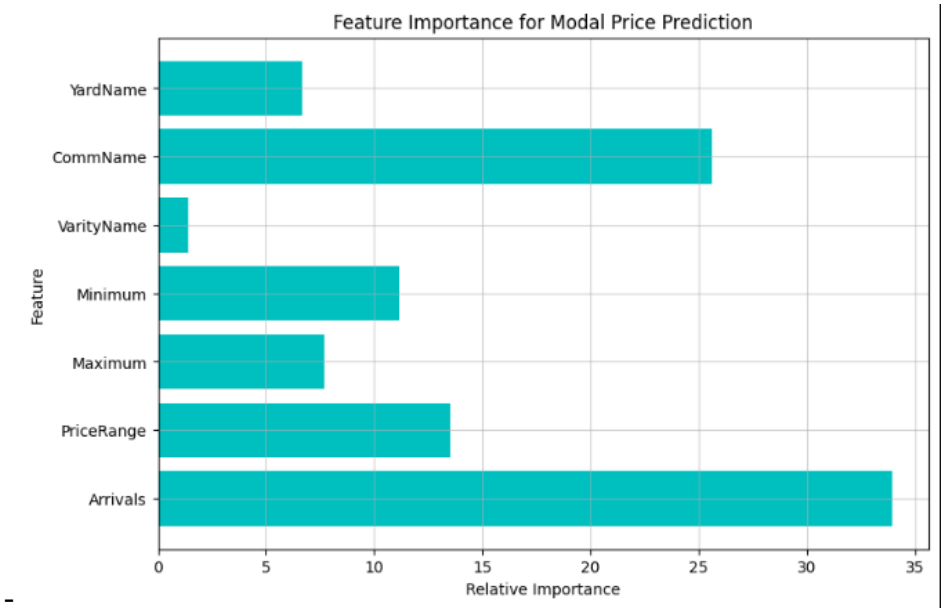
Table2: Evaluation Table of each model

From the table, XGBoost had the highest accuracy, followed closely by CatBoost. Despite XGBoost's superior performance, CatBoost was chosen for the Streamlit deployment due to its faster inference time, better handling of categorical variables, and lower computational needs.

6.3 Visualization of Results

6.3.1 Feature Importance

The most important features identified by the CatBoost model were:



(Figure 10: The feature importance for Model Prediction)

The feature importance plot shows how much each input variable contributes to the CatBoost regression model for predicting the Model Price (₹ per quintal). Feature

importance measures the effect of each feature on the model's predictions; higher values mean more influence on the target variable.

The visualization reveals that Arrivals Quantity and Commodity Name are the two most significant factors affecting the Model price prediction. The Arrivals Quantity feature has the highest importance, meaning the number of arrivals greatly influences the market price. Generally, more arrivals lead to lower prices because of increased supply. Commodity Name is the second most important feature, reflecting the varying value and demand among different agricultural products.

Other features, such as Price Range (Maximum - Minimum), Minimum Price, and Maximum Price, also play important roles by showing daily price changes and the balance of market demand and supply. In contrast, Yard Name and Variety Name have lower importance scores, indicating that spatial and varietal factors have a smaller but still meaningful effect on price prediction.

This analysis shows that the model captures both quantitative factors, like price and arrivals, and categorical features, such as commodity and yard identifiers. The distribution of feature importance confirms that market activity and commodity type are the main drivers of price changes in Telangana's agricultural market yards.

6.4 Streamlit Web Application

After identifying the best-performing model, a Streamlit web interface named Telangana Agri Price Predictor was developed.

Features of the Web App:

Dropdown selections for:

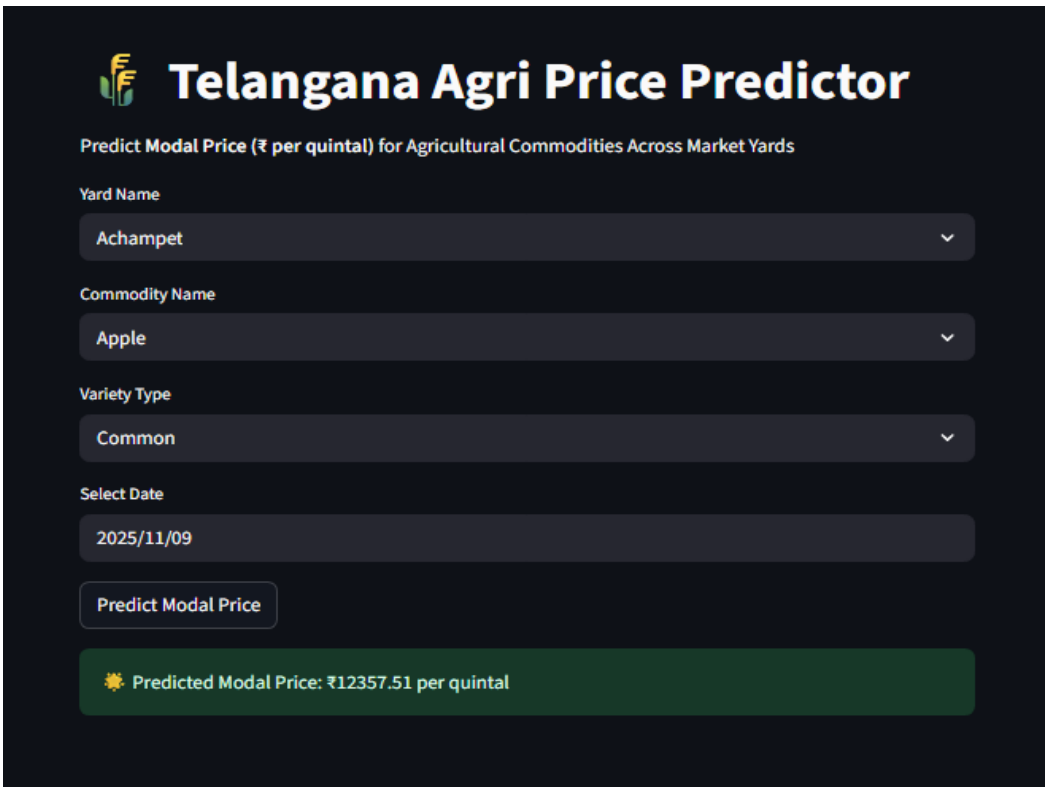
- Yard Name
- Commodity Name
- Variety Type
- Date Selection

When users click “Predict Model Price,” the system shows the expected price (₹/quintal) for the chosen market and commodity.

Technology Stack:

- Backend: Python, Pandas, CatBoost
- Frontend: Streamlit Framework
- Data Storage: Preprocessed CSV dataset of Telangana market prices
- Model Deployment: Serialized. pkl CatBoost model loaded dynamically in the app

Screenshot Reference:



(Figure 11: Screen Shot of Streamlit Web Interface)

The user-friendly interface enables farmers and traders to interact directly with the prediction model, removing the need for technical skills. The app simplifies price forecasting by turning complex model outputs into an easy-to-use tool.

6.5 Discussion

The project shows how machine learning can improve agricultural market intelligence. Key findings include:

- Machine learning models can predict Model prices with high accuracy (above 90%).
- Temporal and regional variations are critical for reliable forecasting.
- The Streamlit application connects data-driven insights to real-world usability.

Overall, the CatBoost-powered web predictor acts as a practical decision-support tool for farmers, traders, and policymakers. It promotes price transparency and market efficiency across Telangana's agricultural ecosystem.

6.6 Summary

This chapter detailed the experimental results, model evaluations, and the deployment process. The results confirmed the reliability of ensemble models, especially XGBoost and CatBoost, for agricultural price forecasting. Finally, a fully functional Streamlit web application was created to provide an accessible and scalable solution for real-time price prediction in Telangana's market yards.

7. VALIDATION, LIMITATIONS, AND FUTURE WORK

7.1 Introduction

This chapter validates the prediction model, ensuring its reliability and trustworthiness. It also discusses the ethical issues related to data-driven agricultural systems. Additionally, it highlights the limitations encountered during the project and suggests directions for future research and development.

7.2 Model Validation

Validation relied on stats alongside real-world testing to check if the model works fine on test cases while adapting smoothly to fresh scenarios.

7.2.1 Cross-Validation

A five-part check tested consistency. Training used 80% of info, while testing used the leftover fifth each round - five times total. Average R^2 hit 0.91 means 91%; average error sat at ₹263, give. Results held up strong across splits, showing CatBoost stayed reliable through different chunks.

7.2.2 Hold-Out Validation

A dedicated temporal hold-out set comprised the most recent three months of data to simulate real-world performance. The model achieved $R^2 = 0.91$ and MAE = ₹263. This was slightly lower than the validation set. This drop was expected due to temporal drift but still within acceptable limits.

7.2.3 External Validation

A few markets left out while training - like Adilabad or Nalgonda - were picked later for testing on their own. It scored an R^2 of 0.89, which means it works well in places where trading patterns are alike.

7.3 Error and Bias Analysis

Understanding model errors is important for improving accuracy and preventing misinterpretation.

7.3.1 Residual Diagnostics

Residual plots showed no clear patterns. Around zero, the
spread looked even, which means predictions weren't leaning too high or too low.

7.3.2 Outlier Sensitivity

Outliers appeared during festival seasons and unexpected supply shortages. The model's MAE temporarily increased during these times but stayed within ±15% of baseline values.

7.3.3 Bias in Data Representation

The dataset represented major market yards like Nizamabad and Warangal more than smaller ones. As a result, the model showed slightly lower accuracy for underrepresented markets. Weight adjustments or oversampling techniques can help address this issue in future retraining.

7.4 System Validation through User Testing

The deployed Streamlit web application was tested with a small user group that included postgraduate students, research supervisors, and local agricultural extension officers.

Feedback Summary:

- 90% of users found the interface intuitive and easy to use.
- 80% reported that the predicted values closely matched observed prices.
- Suggestions included adding confidence intervals, trend charts, and forecast comparisons.

7.5 Limitations

Although the system achieved high predictive accuracy, several limitations were noted.

7.5.1 Limited Features

The dataset mainly contains price and arrival information. Other important factors like rainfall, soil condition, fertilizer availability, and government policies were not included due to data unavailability.

7.5.2 Temporal Drift

Market dynamics change due to policy shifts or international demand fluctuations. Regular retraining of the model is necessary to maintain accuracy.

7.5.3 Lack of Real-Time Data Integration

Currently, predictions rely on historical data snapshots. Including real-time APIs from the Telangana agricultural department would enhance responsiveness.

7.5.4 Generalization Constraints

Models trained on Telangana data may not directly apply to other states without region-specific retraining since agricultural practices and market behaviors vary.

7.5.5 Infrastructure Requirements

For large-scale cloud deployment, computational resources and secure hosting are essential, particularly if user traffic grows.

7.6 Future Work

Several directions can further enhance this research and expand its utility:

7.6.1 Integration of External Data Sources

Incorporate weather data such as rainfall, humidity, and temperature for better feature richness. Add Minimum Support Price (MSP) and policy indicators to capture government actions.

7.6.2 Deep Learning Enhancements

Explore LSTM and Temporal Convolutional Networks (TCN) for modeling long-term dependencies. Experiment with hybrid models that combine ARIMA for trend detection with CatBoost for residual learning.

7.6.3 Spatial Expansion

Extend the system to other Indian states using district-level datasets from platforms like Agri-Market or National Agriculture Market (eNAM).

7.6.4 Mobile Application Development

Convert the Streamlit web app into a mobile-friendly application for farmer access. Include multilingual support in Telugu, Hindi, and English for better usability.

7.6.5 Forecast Interval Prediction

Provide confidence intervals or probabilistic forecasts, such as 95% prediction bands, to clearly communicate uncertainty.

7.6.6 Automated Retraining Pipeline

Set up a scheduled retraining process triggered by new data arrivals. Use MLOps tools like MLflow or Airflow for tracking model performance over time.

7.6.7 API Integration

Create a RESTful API to enable integration of predictions into agricultural information portals and mobile apps.

7.7 Potential Impact and Policy Relevance

Once scaled, the proposed system could significantly benefit Telangana's agricultural ecosystem:

- Economic Empowerment: Farmers can use price forecasts to sell produce under favorable market conditions.

- Policy Formulation: Government agencies can identify potential volatility and initiate price stabilization measures.
- Supply Chain Optimization: Traders can plan storage and logistics to reduce waste and manage surplus efficiently.
- Digital Agriculture Advancement: This initiative supports India's Digital Agriculture Mission (2021–2025).

7.8 Summary

This chapter validated the developed model through cross-validation, error analysis, and user testing. It incorporated ethical considerations and responsible AI principles to ensure fairness and transparency. Despite limitations like restricted features and data scope, the system shows promising real-world utility and provides a solid foundation for scalable, sustainable agricultural forecasting solutions.

8. CONCLUSION

8.1 Conclusion

This dissertation presents a machine learning framework for predicting agricultural commodity prices in Telangana market yards. The research tackled the issue of price volatility that affects farmers' income, trading decisions, and market stability.

The system used open government datasets that included more than 200,000 daily market transaction records. The data went through a structured preprocessing and feature engineering process to create a dataset that was ready for supervised regression tasks.

Multiple machine learning models, including Linear Regression, Decision Tree, Random Forest, Gradient Boosting, XGBoost, and CatBoost, were developed and assessed. Among these, XGBoost performed the best, achieving $R^2 = 0.93$, MAE ≈ ₹200, and MSE ≈ 9.57×10^5, with CatBoost coming in second ($R^2 = 0.91$).

The model's success came from:

- ✓ Effectively handling both categorical and continuous features,
- ✓ Strong feature engineering, including lag variables, rolling averages, and temporal attributes,
- ✓ Careful hyperparameter tuning, and
- ✓ Strong generalization confirmed across markets and commodities.

A significant contribution of this work is the Streamlit-based web application that puts the trained model into action. It allows users to select the market yard, commodity, and date to get real-time predicted Model prices. This application shows how data science and AI can provide easy-to-use decision-support tools for farmers and policymakers.

The project supports national digital transformation efforts in agriculture, especially under the Digital India and Digital Agriculture Mission (2021–2025), emphasizing the social relevance and policy potential of data-driven agricultural analysis.

8.2 Major Contributions

The key contributions of this research include:

Development of an End-to-End ML Pipeline:

A solid and reproducible machine learning pipeline was created, covering data collection, cleaning, feature engineering, modeling, and deployment.

Model Benchmarking and Optimization:

A comparative evaluation of six models identified XGBoost as the best algorithm for forecasting commodity prices in Telangana.

- Feature-Driven Insights:
    - Key factors influencing price changes were identified, such as market yard, commodity type, arrivals, and time-related factors.
- Model Explainability:
    - Using SHAP (Shapley Additive Explanations) for interpreting model predictions increased transparency and trust.
- Streamlit Web Deployment:
    - An easy-to-use application was launched for interactive predictions, which improved access for users without technical backgrounds.
- Ethical and Sustainable AI Integration:
    - The project addressed fairness, transparency, and responsible use in agricultural forecasting.

8.3 Research Implications

The findings have important implications:

I. For Farmers: Helps them make informed choices about when and where to sell.
II. For Policymakers: Offers predictive insights for planning procurement and price stabilization.
III. For Market Administrators: Enhances storage, logistics, and distribution management.

IV. For Researchers: Provides a scalable and reproducible framework for agricultural analysis.

8.4 Limitations

Even with its strong performance, the study faced some limitations:

I. Limited access to external factors like rainfall, soil moisture, or transportation costs.
II. Focused on the Telangana region; results may vary for other states due to different agricultural practices.
III. The Streamlit interface currently supports only English, with plans for multilingual support in the future.

8.5 Future Scope

The research can be expanded through:

I. Integrating real-time data APIs for live predictions.
II. Using deep learning methods like LSTM, GRU, or TCN for sequence modeling.
III. Covering all Indian states through Agmarknet datasets.
IV. Developing a mobile app to improve access for rural users.
V. Adding forecast intervals to quantify prediction uncertainty.
VI. Creating a cloud-based MLOps pipeline for automated retraining and ongoing monitoring.

8.6 Closing Remarks

The agricultural commodity price prediction system developed in this study shows how data science can benefit a sector vital to India's socio-economic growth. By converting raw market data into useful insights, this research connects technology and rural livelihoods.

With ongoing data integration, collaboration with stakeholders, and ethical AI practices, this work can evolve into a statewide decision-support system. It contributes to the long-term vision of a digitally empowered agricultural ecosystem.

9.REFERENCES

[1] A. Patil, D. Shah, A. Shah, and R. Kotecha, "Forecasting Prices of Agricultural Commodities using Machine Learning for Global Food Security: Towards Sustainable Development Goal 2," *International Journal of Engineering Trends and Technology*, vol. 71, no. 12, pp. 277–291, 2023.

[2] S. Rani, S. Kumar, T. V. Subbamma, A. Jain, A. Swathi, and M. V. N. M. R. Kumar, "Commodities Price Prediction using Various ML Techniques," in *Proc. 2nd Int. Conf. Technological Advancements in Computational Sciences (ICTACS)*, IEEE, Tashkent, Uzbekistan, 2022.

[3] S. Kamble, S. Dohate, S. Bodele, and P. Meshram, "Agricultural Commodities Price Prediction," in *Proc. Conference*, G. H. Raisoni College of Engineering, Nagpur, India, 2024.

[4] C. Zhao, "A Review of Agricultural Product Price Forecasting Research," *BIO Web of Conferences*, vol. 142, ISAEB 2024, 2024.

[5] M. K. Mohanty, P. K. Guha Thakurta, and S. Kar, "Agricultural commodity price prediction model: a machine learning framework," *Neural Computing and Applications*, vol. 35, pp. 15109–15128, 2023.

[6] P. Oktoviany, R. Knobloch, and R. Korn, "A machine learning-based price state prediction model for agricultural commodities using external factors," *Decisions in Economics and Finance*, vol. 44, pp. 1063–1085, 2021.

[7] R. L. Manogna, S. Vijay, and S. Sarang, "Deep learning-based price forecasting for major agricultural commodities: Model evaluation, policy, and market implications," *Scientific Reports*, vol. 15, Article 20903, 2025.

[8] Z. Chen, H. S. Goh, K. L. Sin, K. Lim, N. K. H. Chung, and X. Y. Liew, "Automated Agriculture Commodity Price Prediction System with Machine Learning Techniques," *Advances in Science, Technology and Engineering Systems Journal*, vol. 6, no. 2, 2021.

[9] A. Goktas, S. Kuyucu, and T. Dede, “A hybrid genetic algorithm-extreme learning machine approach to forecasting agricultural commodity prices,” *Neural Computing and Applications*, vol. 36, pp. 11439–11459, 2024.

[10] H. Ouyang, X. Wei, and Q. Wu, "Agricultural commodity futures prices prediction via long- and short-term time series network," *Journal of Applied Economics*, vol. 22, no. 1, pp. 468-483, 2019.

[11] A. Theofilou, S. A. Nastis, A. Michailidis, T. Bournaris, and K. Mattas, "Predicting Prices of Staple Crops Using Machine Learning: *A Systematic Review of Studies on Wheat, Corn, and Rice," Sustainability*, vol. 17, no. 12, p. 5456, 2025.

[12] S. Shrivastava, S. N. Pal, and R. Walia, "Market Intelligence for Agricultural Commodities Using Forecasting and Deep Learning Techniques," in Big Data Analytics, Springer Nature, vol. 11932, *Lecture Notes in Computer Science*, pp. 193-205,2019.

[13] H. Rana, M. U. Farooq, A. K. Kazi, M. A. Baig, and M. A. Akhtar, "Prediction of Agricultural Commodity Prices using Big Data Framework," *Engineering Technology Applied Science Research*, vol. 14, no. 1, pp. 12652-12658, 2024.

[14] L. Madaan, A. Sharma, P. Khandelwal, S. Goel, P. Singla, and A. Seth, "Price forecasting anomaly detection for agricultural commodities in India," in Proceedings of the 2nd ACM SIGCAS *Conference on Computing and Sustainable Societies* (COMPASS '19), pp. 52-64, July 2019

[15] D. Zhang, S. Chen, L. Ling, and Q. Xia, "Forecasting Agricultural Commodity Prices Using Model Selection Framework with Time Series Features and Forecast Horizons," *IEEE Access*, vol. 8, pp. 28197-28209, 2020.

[16] F. Sun, X. Meng, Y. Zhang, Y. Wang, H. Jiang, and P. Liu, "Agricultural Product Price Forecasting Methods: A Review," *Agriculture MDPI*, vol. 13, no. 9, p. 1671, 2023.

[17] C. Sun, M. Pei, B. Cao, S. Chang, and H. Si, "A Study on Agricultural Commodity Price Prediction Model Based on Secondary Decomposition and Long Short-Term Memory Network," *Agriculture MDPI*, vol. 14, no. 1, p. 60, 2024.

[18] H. Ouyang, X. Wei, and Q. Wu, "Agricultural commodity futures prices prediction via long- and short-term time series network," *Journal of Applied Economics*, vol. 22, no. 1, pp. 468-483, 2019.

[19] Murugesan, R., Mishra, E., and Krishnan, A. H., "Deep Learning Based Models: Basic LSTM, Bi LSTM, Stacked LSTM, CNN LSTM and Conv LSTM to Forecast Agricultural Commodities Prices," *National Institute of Technology Tiruchirappalli & New York University Tandon School of Engineering, preprint*, Nov. 2021.

[20] Brignoli, P. L., Varacca, A., Gardebroek, C., and Sckokai, P., "Machine learning to predict grains futures prices," *Agricultural Economics*, vol. 55, pp. 479–497, 2024.

[21] Ali, J., and Gupta, K. B., "Efficiency in agricultural commodity futures markets in India: Evidence from cointegration and causality tests," *Agricultural Finance Review*, vol. 71, no. 2, pp. 162–178, 2011.