Application of Machine learning techniques models for forecasting of Redgram prices of Andhra Pradesh, India

ABSTRACT

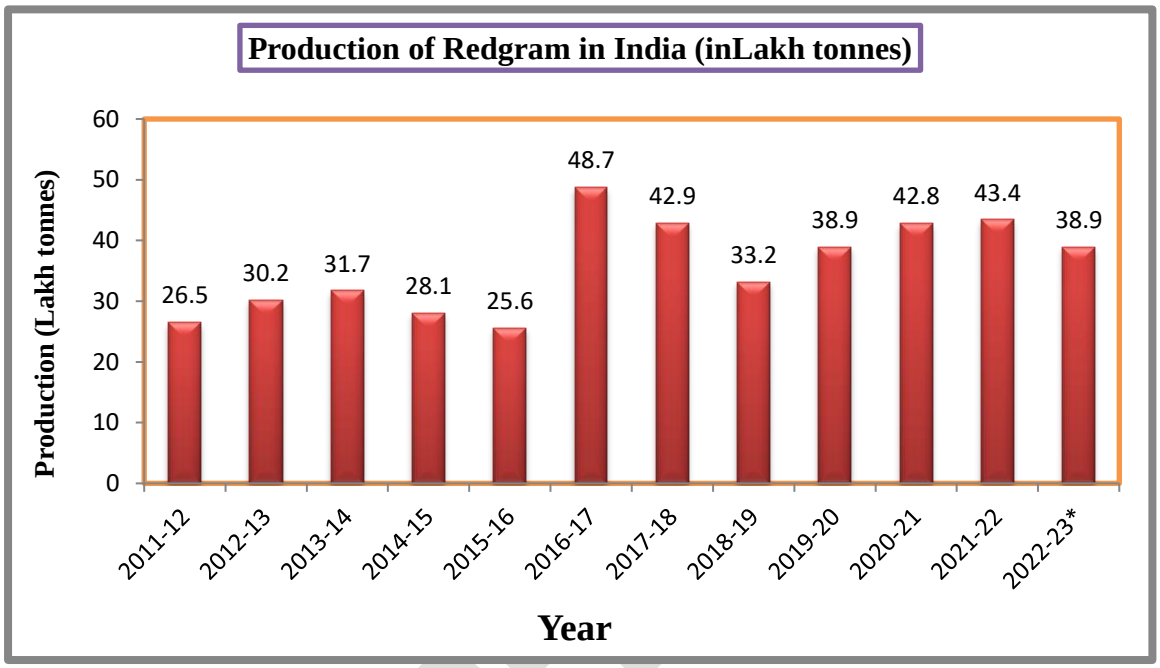
Recent advancements in Machine Learning (ML) had proven highly effective in modeling time series data, consistently outperforming traditional time series models in forecasting accuracy according to empirical studies. However, the application of ML techniques in forecasting agricultural commodity prices in India was remains scarce, despite their demonstrated success in other domains. The present study endeavours to investigate the efficiency of various machine learning (ML) algorithms, including Artificial Neural Network (ANN), Support Vector Regression (SVR) and Random Forest (RF) models, alongside traditional linear time series models such as SARIMA and GARCH models in forecasting of the monthly price series of redgram in Andhra Pradesh, India. The findings of this study indicated that the Random Forest (RF) model exhibited superior performance compared to other machine learning techniques and univariate time series models in forecasting redgram monthly prices in Andhra Pradesh. However, the forecasting accuracies of alternative techniques, including Support Vector Regression (SVR), Artificial Neural Network (ANN), GARCH, and SARIMA models, fell short of expectations. In this research, the superiority of various models was substantiated through accuracy metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). Additionally, the Diebold-Mariano test is conducted to assess significant differences in predictive accuracy among the models. The DM test also concluded that the RF model outperformed than the other models under consideration.

Keywords: ANN, GARCH, Machine learning, Redgram, RF, SARIMA, SVR.

1. INTRODUCTION

Red gram, scientifically known as ***Cajanus cajan***, is a legume crop widely cultivated in tropical and subtropical regions around the world. It's commonly known as pigeon pea or arhar dal in India. Red gram is valued for its high protein content and is a staple food in many parts of the world, particularly in South Asia, Africa, and the Caribbean. It's not only a dietary staple but also plays a crucial role in sustainable agriculture due to its ability to fix nitrogen in the soil, thus enriching it for subsequent crops. Red gram is versatile, used in various culinary dishes such as soups, stews, and curries, and also holds significance in traditional medicine for its potential health benefits. Globally, redgram is grown in an area of 63.57 lakh hectares with a production of 54.75 lakh tonnes and productivity of 861.25 kg/ha (FAO STAT, 2021,26,27,28). India ranks first in redgram production globally with 43.4 lakh

tonnes cultivated in 49.8 lakh hectares with productivity of 871 kg/hectare in 2021-22 (agricoop.nic.in). According to Government 1st advance estimates, all India redgram production in 2022-23 is at 3.89 million tonnes. The production scenario of redgram in India was shown in figure 1. India, Andhra Pradesh contributes an area of 2.42 lakh hectares, production 0.78 lakh tonnes and productivity of 323 kg/ha during 2022-23 (des.ap.gov.in)



*Source: Directorate of Economics and Statistics (DES). *1st Advance estimates*

Figure. 1 Production of Redgram in India (in lakh tonnes)

In agricultural markets, access to accurate prices data significantly enhances the bargaining power of farmers while fostering healthy competition among traders. Armed with pricing information, farmers were empowered to strategically navigate between nearby markets, optimizing the sale of their produce and securing favourable prices. This data-driven approach allows farmers to make informed decisions regarding the timing of their product marketing, thereby mitigating the impact of erratic price fluctuations. By leveraging price information, farmers would able to capitalize on arbitrage opportunities across different times and locations, ultimately reducing the volatility of prices. One notable challenge in analysing price data was pronounced seasonality observed in agricultural markets. This seasonal pattern poses a formidable obstacle to the accurate forecasting of prices. Despite the many models available to capture the intricacies of price behaviour, consensus remains elusive among researchers regarding the most effective approach for forecasting prices. This complexity

underscores the need for continued exploration and refinement of analytical models to capture the agricultural price dynamics.

Various linear and nonlinear methodologies have been developed within the framework of time series analysis to model agricultural commodity prices. These include well-known approaches such as the Autoregressive Integrated Moving Average (ARIMA) model, Seasonal ARIMA (SARIMA), and the Generalized Autoregressive Conditional Heteroscedastic (GARCH) model. Past research endeavours have focused on leveraging these methodologies to forecast agricultural commodity prices, aiming to provide valuable insights into market trends and price dynamics. Recently, Machine Learning (ML) algorithms, developed within the data science paradigm, have risen to prominence in forecasting tasks. This trend extends to the prediction of financial and economic time series, where ML techniques have demonstrated notable efficacy. Empirical studies have consistently revealed the superior performance of ML approaches compared to traditional time series models across various financial assets. Notably, a comprehensive comparative analysis of statistical models and machine learning techniques can be found in the literature. Among the ML methodologies, popular choices include Artificial Neural Networks (ANN), Support Vector Regression (SVR) and Random Forest (RF). These techniques are characterized by their data-driven, nonparametric nature, enabling them to effectively capture the stochastic dependencies present within the data. The primary aim of this paper was to assess and compare the predictive performance of efficient Machine Learning (ML) algorithms-namely, ANN, SVR and RF for forecasting the monthly prices of Redgram in Andhra Pradesh, India.

2. MATERIALS AND METHODS

Data and Source of Study

To achieve the defined objectives, selected redgram monthly prices data was collected from Agricultural Market Intelligence Committee (AMIC), Lam, Guntur for the period from January 1991 to December 2023. The collected data was divided into training and a testing datasets. There are 396 observations of redgram monthly prices series, of which 385 observations were utilized for training dataset for model development and last 12 observations, were used as testing dataset for model validation purpose.

Descriptive Statistics:

The summary statistics viz., mean, median, standard deviation, skewness, kurtosis, minimum and maximum were used to study the behaviour of the monthly prices of redgram in Andhra Pradesh.

ARIMA Model:

The Autoregressive Integrated Moving Average (ARIMA) methodology developed by Box-Jenkins is the most widely used model for analysing time series data. The Box-Jenkins model-building process is used to fit a blended ARIMA model to provided data. The basic purpose of fitting the ARIMA model is to accurately characterise and forecast the time series stochastic process (Box and Jenkins, 1970).

Initially, George Box and Gwilym Jenkins conducted substantial research on ARIMA models, and their names were frequently associated with the broad ARIMA method used in time series analysis, forecasting, and control. The two forms of stochastic processes are stationary and non-stationary. The ARIMA model can only be used with stationary data.

Stationarity and Non-stationarity:

A process that generates data in equilibrium around a constant value and has a constant variance around the mean throughout time is referred to as "stationary." If the means shift over time and the variance is not roughly constant both mean and variance, the series is said to be non-stationary. To build the ARIMA model the series should be stationary in nature. If the original series is not stationary then it has to make stationary by differencing will be done to convert the non-stationary series into stationary series.

Autoregressive Model of order p (AR (p)):

An autoregressive model is one in which Y_t depends only on its past values Y_{t-1}, Y_{t-2}, Y_{t-3}, etc. is called autoregressive of order p and abbreviated as AR (p), where $\emptyset$ is autoregressive coefficient and ε_t is white noise.

$$y_t = \emptyset_1 y_{t-1} + \emptyset_2 y_{t-2} + \cdots + \emptyset_p y_{t-p} + \varepsilon_t \qquad \ldots (1)$$

In general, a variable y_t is said to be autoregressive of order p [AR (p)], if it is a function of its p past values and can be represented as:

$$y_t = \sum_{i-1}^{p} \emptyset_i y_{t-i} + \varepsilon_t \quad \ldots (2)$$

Moving Average Model order q (MA (q)):

Moving Average (MA) is the one where Y_t depends on its lagged forecast errors.

$$y_t = \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} \quad \ldots. (3)$$

The MA term is represented by the order q and abbreviated as MA(q) and θ is MA coefficient.

Autoregressive Moving Average model (ARMA (p, q)):

It is often advantageous to use both autoregressive and moving average processes in order to achieve greater flexibility in fitting of time-series data. This leads to mixed autoregressive-moving average model.

$$y_t = \emptyset_1 y_{t-1} + \emptyset_2 y_{t-2} + \cdots + \emptyset_p y_{t-p} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} \quad \ldots. (4)$$

2.2.5 Autoregressive Integrated Moving Average Model (ARIMA (p, d, q)):

The ARIMA model allows y_t to be explained by its past, or lagged values and stochastic error terms. The models are often referred to as "mixed models." ARIMA models use a combination of autoregressive (AR), integration (I) and moving average (MA). The term integration is referred when a nonstationary series is converted into stationary series by means of differencing. Box and Jenkins propose a practical four stage procedure for finding a good model. The four-stage univariate Box Jenkins procedure is summarized schematically in Figure 2.

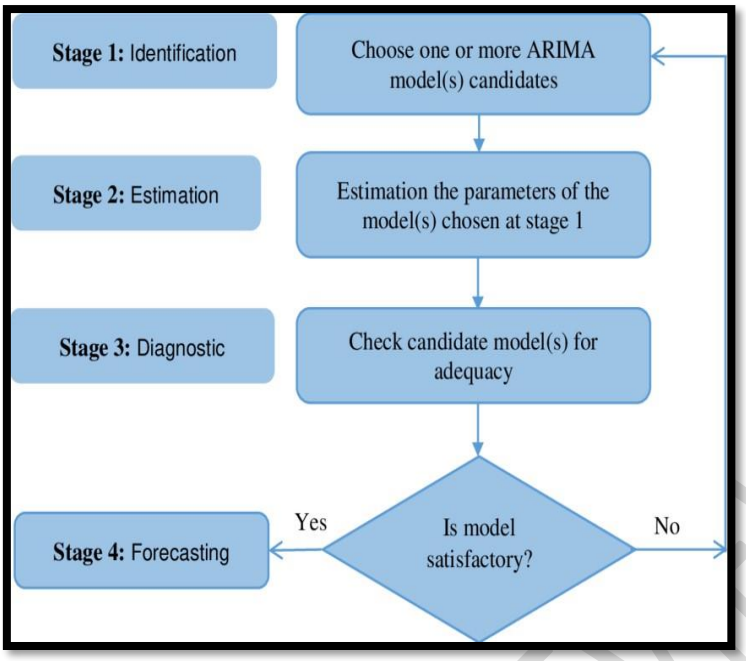


Figure. 2 Flow chart of Box-Jenkins Methodology

The main stages in setting up a Box-Jenkins forecasting model are described below:

1. **Identification:**

The autocorrelation function (ACF) and partial autocorrelation function (PACF) are two graphical devices to measure the correlation between the observations within a single data series and they give an idea about the patterns and relationship in the available data. As the time series under study is a particular realization of the process, the theoretical ACF and PACF must resemble the estimated ACF and PACF of the data.

Table 1. Pattern of ACF and PACF for identification of AR, MA and ARMA process

PROCESS	ACF	PACF
AR	Decays towards zero	Cut off to zero (lag length of last spike is the order of the process)
MA	Cut off to zero (lag length of last spike is the order of the process)	Decays towards zero
ARMA	Tails off towards zero	Tails off towards zero

2. Estimation of parameters

At the estimation stage, coefficients of the identified models are estimated using method of least squares or maximum likelihood estimation methods are used to estimate the parameters. Stationarity and invertibility are checked for the coefficient obtained and at the same time diagnostic checking is done in order to know whether the model fit the data satisfactorily or not. The importance of the estimation coefficients is measured in terms of the statistical significance.

3. Diagnostic Checking

Different models can be obtained for various combinations of AR and MA individually and collectively. The best model is obtained with following diagnostics.

(a) Low Akaike Information Criteria (AIC) / Bayesian Information Criteria (BIC)

AIC is given by (-2 log L + 2m) where m = p + q + P + Q and L is the likelihood function. Since -2log L is approximately equal to {n (1+log 2π) + n log σ^2} where σ^2 is the model MSE. Thus, AIC can be written as AIC = {n (1+log 2π) + n log σ^2 + 2m} and because first term in this equation is constant, it is usually omitted while comparing between models. The model having lowest AIC/BIC is considered as the best model.

(b) Plot of residual ACF

Once the appropriate model has been fitted, the goodness of fit can be examined by plotting the ACF of residuals of the fitted model. If most of the sample autocorrelation coefficients of the residuals are within the limits $\pm$ 1.96 / N where N is the number of observations on which the model is based, then the residuals are white noises indicating that the model is good fit.

(c) Box-Pierce or Ljung-Box texts

Box-Pierce statistic is a test to measure the overall adequacy of the chosen model by examining a quantity Q, whose approximate distribution is Chi-square.

$$Q = n \sum_1^k r_{(j)}^2 \quad \ldots (5)$$

Where *k* as maximum lag considered, and is usually around 20, *n* = number of observations, *r(j)* is the estimated autocorrelation at lag *j*. Chi-square with *(k-m-1)* degrees of freedom where *m-1* is the number of parameters estimated in the model.

A modified Q statistics is the Ljung-box which is given by

$$q = n(n+2) \sum \frac{r_{(j)}^2}{n-j} \quad \ldots (6)$$

The critical value of *Q* statistic is compared with Chi-square *(n-1)* degrees of freedom. Residuals should be uncorrelated and Q should be small if model is correctly specified. A significant value of test statistic indicates the chosen model is not a good fit.

4. Forecasting

The model that satisfies all the diagnostic checks is considered for forecasting. If the model is based on differencing / de-trending transformations, then the model must be represented with relevant expressions of original series. Then only, the forecasts can be made.

Seasonal ARIMA

In the time series analysis, seasonality is defined as the pattern of changes that repeats over *S* time periods, where *S* is the number of time periods between the repeats of the pattern. For quarterly data, *S* = 4 time periods per year and for monthly data *S* = 12 time periods per year are considered. As the regular differencing was applied to the series having non-stationary nature similarly seasonal differencing will be applied to the seasonal non-stationary series. The seasonal Autoregressive (SAR) and Seasonal Moving Average (SMA) are the parameters of seasonal ARIMA. In the seasonal ARIMA model, seasonal AR and MA terms predicts the x_t often with the lags that are multiples of *S*.

Seasonal ARIMA model is denoted by ARIMA (p, d, q) $(P, D, Q)_S$, where *p* represents the number of autoregressive terms, *q* represents the number of moving average terms and *d* denotes order of differencing to induce stationarity, *P* represents the number of seasonal Autoregressive components, *Q* represents the number of seasonal moving average terms and *D* represents the number of seasonal differences required to make the series stationarity. The seasonal ARIMA model expressed as follows;

$$\emptyset(B)\Phi(B)\nabla^d\nabla_s^D r_t = \theta(B)\Theta(B)\varepsilon_t \quad \text{…. (7)}$$

$$w_t = \nabla^d\nabla_s^D r_t \quad \text{…. (8)}$$

$\nabla^d = (1-B)^d$ denotes the number regular differences and $\nabla_s^D = (1-B^s)^D$ denotes number of seasonal differences.

Where, $\emptyset(B)$ is stationary Autoregressive operator, $\theta(B)$ is a stationary moving average operator, ε_t is a white noise (Brockwell and Davis, 1996).

Generalized Autoregressive Conditional Heteroskedasticity (GARCH)

Bollerslev (1986) proposed Generalized ARCH (GARCH) model, in which the conditional variance is also a linear function of its own lags. The conditional variance has the property that the auto correlation function of ε_t^2 can decay slowly.

$$h_t = a_0 + \sum_{i=1}^{q} a_i \varepsilon_{t-i}^2 + \sum_{j=1}^{p} b_j h_{t-j} \quad \text{…. (9)}$$

For the ARCH family, the decay rate is very rapid. The overwhelmingly most popular GARCH model in applications has been the GARCH (1, 1) model. The GARCH (p, q) process is weakly stationary if and only if

$$\sum_{i=1}^{q} a_i + \sum_{j=1}^{p} b_j < 1 \quad \text{…. (10)}$$

The GARCH model can be regarded as an application of ARMA model to the squared series ε_t^2 where a and b are constants.

Estimation

Maximum likelihood method of estimation is used to estimate the parameters of GARCH model.

$$L_t(\theta) = T^{-1}\sum_{t=1}^{T}(\log h_t + \varepsilon_t^2 h_t^{-1}) \quad \text{…. (11)}$$

$$h_t = a_o + \sum_{i=1}^{q} a_i y_{t-i}^2 + \sum_{j=1}^{p} b_j h_{t-j} \quad \text{…. (12)}$$

The log likelihood function of a sample of T observations, apart from constant, is

$$L_t(\theta) = T^{-1} \sum_{t=v}^{T} (\log h_t + \varepsilon_t^2 h_t^{-1}) \quad \text{…. (13)}$$

The Akaike Information Criterion (AIC) and Bayesian information criterion (BIC) values for GARCH model Gaussian distributed errors are computed by

$$AIC = -2\ln(L) + 2k \quad \text{…. (14)}$$

$$BIC = -2\ln(L) + \ln(N)k \quad \text{…. (15)}$$

Where L is the value of likelihood function, Evidently, the likelihood equations are extremely complicated.

Steps in fitting a GARCH model

Step-1: Determine whether the time series is stationary

1. Perform stationarity test- ADF test
2. If required differencing is done

Step-2: Identify the mean model

1. Autocorrelation Function (ACF)
2. Partial Autocorrelation Function (PACF)

Step-3: Estimate the model parameters and diagnostic checking

The parameters are estimated through maximum likelihood function such that overall measure of errors is minimized or likelihood function is maximized. Portmanteau test is used to check if the model assumptions about the errors are satisfied.

Step-4: Select the most suitable ARIMA model

The most suitable ARIMA model is selected using the smallest Akaike Information Criteria (AIC) or Schwarz-Bayesian Criteria (SBC).

Step-5: Determination of residuals and heteroscedasticity test

ACF and PACF values of the '*r*-square' are determined and the lags in which the values are found to be significant are identified. The test for heteroscedasticity is done at identified significant lags. The test employed is the ARCH-LM test.

Step-6: Residuals and Diagnostic Checking

The residuals obtained from the mean model used for fitting the different GARCH models were squared and stored in a new variable called '*e*-square.' The diagnostic tests are employed to check whether the residuals are white noise or not.

ARCH (LM) Test

A time series exhibiting conditional heteroscedasticity or autocorrelation in the squared series is said to have *autoregressive conditional heteroscedastic* (ARCH) effects. ARCH(LM) test is used to test for ARCH effects by regressing the squared errors on its lags. The null hypothesis is that the lagged regression coefficients are zero there are no ARCH effects.

$$\text{var}\frac{y_t}{H_{t-1}} = \text{var}\left(\frac{\varepsilon_t}{H_{t-1}}\right) = E\left(\frac{\varepsilon_{2t}}{H_{t-1}}\right) = \sigma_t^2 \qquad \ldots (16)$$

Where, y_t is the time series, ε_t is an innovation process with mean zero, H_t is the history of the process available at time *t*. σ_t^2 is the conditional variance.

Artificial Neural Networks:

The artificial neural network is a machine learning algorithm which resembles the biological neuron and works based on the learning experience and pattern present in the data sets. The Artificial Neural Network for time series modelling and analysis is termed as Time Delay Neural Network (TDNN) because the network contains time lags or delays in input layer. Generally, ANN has three-layer data structure i.e., input layer, output layer, hidden layer as depicted in Figure.3.

Output of the neuron can be obtained as

$$f(x_j) = f(\alpha_j + \sum_{i=1}^{k} wij\, y_i) \qquad \ldots. (17)$$

The objective of the neural network is to transform the inputs into meaningful outputs.

$$Y = f(\sum_{i=1}^{n} w_i x_i) \quad \text{…. (18)}$$

where *Y* is output, *f* is activation function, w_i is weights of the connections and x_i is Input to the neurons

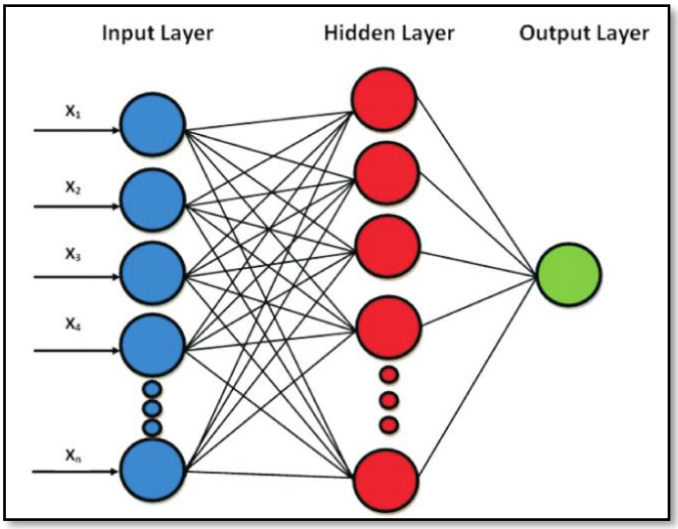


Figure. 3 ANN architecture

Feed Forward neural Network:

In feed forward neural network, the information flow is unidirectional as unit sends information to other unit from which it does not receive any information. There are no feedback loops available in this architecture and they also have fixed input and output neurons. They are used in pattern generation/recognition/classification.

Total number of trainable parameters in a feed-forward neural network with *n* hidden layers can be obtained as;

$$i \times h_1 + \sum_{k=1}^{n-1}(h_k \times h_{k+1}) + h_n \times o + \sum_{k=1}^{n} h_k + o \quad \text{…. (19)}$$

The neural network predicts the weights and biases to calculate feed the inputs forward through the network, the total net input will be for first weight (w_1) and input (i_1).

$$neth_1 = w_1 \times i_1 + w_2 \times i_2 + b_1 \times 1 \quad \text{….(20)}$$

Logistic activation function is used to get the output (o_1) of the network is as follows;

$$outh_1 = \frac{1}{1+e_{h1}^{-net}} \quad \text{… (21)}$$

It Calculates the error for each output neuron using the squared error function and adds them to get the total error:

$$E_{total} = \sum \frac{1}{2}(target - output)^2 \quad \text{…. (22)}$$

The total error for the neural network is the sum of the errors

$$E_{total} \quad = E_{o1} + E_{o2} + E_{o3} + \cdots, + E_{on} \quad \text{…. (23)}$$

2.4.2 The Back propagation Algorithm

Multiple Layer Perceptron (MLP) network is trained using one of the supervised learning algorithms i.e., back propagation algorithm (BPA). The BPA uses data to adjust the network weights and thresholds to minimize the error in its predictions on training set.

$$X_j = \sum y_i \, W_{ij} \quad \text{…. (24)}$$

Where, y_i is the activity level of the j^{th} unit in the previous layer and W_{ij} is the weight of the connection between the i^{th} and the j^{th} unit.

The unit calculates the activity y_j using some function of the total weighted input. Generally, the sigmoid function used as activation function and which is given as follows

$$y_i = [1 + e^{-x_j}]^{-1} \quad \text{….. (25)}$$

After calculating all the output units network computes the error E, and which is expressed by the following equation:

$$E = \frac{1}{2}\sum_j \left(y_i - d_j\right)^2 \quad \text{…. (26)}$$

Where, y_i is the activity level of the j^{th} unit in the top layer and d_j is the desired output of the j^{th} unit.

Further steps in the calculation of back propagation are explained as follows;

Calculate the error changes, as the activity of an output unit is changed. The difference between the actual and the desired activity is the error derivative (EA) and is expressed as follows;

$$EA_j = \frac{\partial E}{\partial y_i} = y_i - d_j \quad \text{…. (27)}$$

Following equations explains how fast the error changes weight on the connection into output unit is changed.

$$E_j^I = \frac{\partial E}{\partial x_i} = \frac{\partial E}{\partial y_i} \times \frac{\partial E}{\partial x_i} = EA_j y_i (1 - y_j) \quad \text{…. (28)}$$

The quantity (EW) is the answer from, equation (29) multiplied by the activity level of the unit from which the connection obtained.

$$EW_{ij} = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial X_i} \times \frac{\partial X_i}{\partial W_{ij}} = E_j^I y_i \quad \text{…. (29)}$$

In this case, if back propagation is applied to multilayer networks, then the activity of a unit in the previous layer changes, it affects the activities of all the output units to which it is connected and we can calculate how quickly the error changes as the activity of a unit in the previous layer is changed. Add each of these individual effects on output units to determine the overall influence on the error. For this, equation (30) is multiplied by the weight on the connection to that output unit.

$$EA_j = \frac{\partial E}{\partial y_i} \sum_j \frac{\partial E}{\partial x_i} \times \frac{\partial x_i}{\partial y_i} = \sum_j E_j^I W_{ij} \quad \text{…. (30)}$$

By using these equations (3.28) and (3.31), can convert the EAs of one layer of units into EAs for the previous layer. This procedure shall be repeated to get the EAs based on the required previous layers. Once the EA of a unit is obtained, then by using (29) and (30) equations can compute the EWs on its incoming connections.

Support Vector Regression (SVR)

Support Vector Regression (SVR), the supervised learning algorithm is used to predict discrete values. SVR's main aim is to locate the line of best fit. The best-fit line in SVR is the hyperplane with the maximum number of points. The SVR, unlike other regression models, aims to fit the best line inside a threshold value, rather than minimizingthe error between the real and projected value. The distance between the hyperplane and the boundary line is the threshold value. The goal of Support vector regression is to develop a function that

approximates mapping from an input domain to real numbers by using training sample data. The key goal here is to choose a decision boundary that is a distance from the original hyperplane and contains data points closest to the hyperplane or support vectors.

Fitting of Support Vector regression $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ can be expressed as

$$\text{Minimimize } \{\Sigma_{i=1}^{n} \max[0, 1 - y_i f(xi)] + \lambda \Sigma_{j=1}^{p} \beta_j^2 \quad \ldots\ldots (31)$$

The principal idea involved in SVR is to transform the original input space into high-dimensional variable space and then build the regression or time series model in a transformed high-dimensional feature space. A vector of data set says $Z = \{xiyi\}_{i=1}^{N}$, where $x_i \in R^n$ is the input vector, y_i is the scalar output, and N is the size of the data set. The general equation SVR can be written as follows:

$$f(x) = W^T \emptyset(x) + b \quad \ldots\ldots (32)$$

where, *W* is the weight vector, *b* is the bias term, and superscript *T* denotes the transpose.

The coefficients *W* and *b* are estimated from data by minimizing the following regularized risk function:

$$R(\theta) = 1/2||w||^2 + C\ [1/N\ \Sigma_{i=1}^{N} L_\varepsilon(y_i, f\ (x_i))\ ] \quad \ldots(33)$$

This regularized risk function minimizes both the empirical error and regularized term simultaneously, which helps in avoiding both under and overfitting of the model. The first term $1/2||\ w\ ||^2$ is called the 'regularized term', which measures the flatness of the function. Minimizing $1/2||\ w\ ||^2$ will make a function as flat as possible.

The second term $1/N\ \Sigma_{i=1}^{N} L_\varepsilon(y_i, f\ (x_i))$ is called the 'empirical error', which was estimated by Vapnik ε-insensitive loss function as follows:

$$L_\varepsilon(y_i, f(x_i)) = f(x) = \{|y_i, f(x_i) - \varepsilon|;\ |y_i - f(x_i)| \geq \varepsilon$$
$$0, \qquad |y_i - f(x_i)| < \varepsilon \quad \ldots(34)$$

where,y_i is actual value and $f(x_i)$ is an estimated value. The most commonly usedkernel function is the radial basis function (RBF) which is given as follows:

$$k(x_i, x_j) = \exp\{-\gamma||x - x_i||^2 \quad \ldots(35)$$

The performance of the RBF kernel function requires optimization of two hyper-parameters:

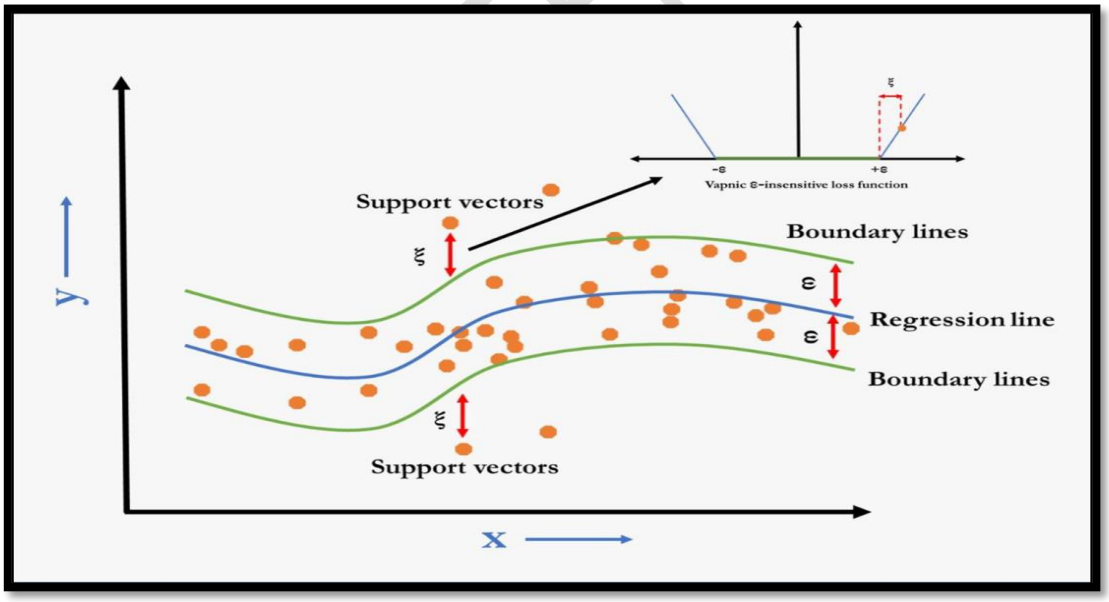
regularization parameter *C*, which balances the complexity and approximation accuracy of the model, and the Kernel bandwidth parameter, which represents the variance of the RBF kernel function γ.

Hyper parameters in SVR

Hyperplane: Hyperplanes are decision boundaries for predicting the continuous output. Support Vectors are the data points on either side of the hyperplane that are closest to the hyperplane. These are used to draw the required line that shows the algorithm's predicted outcome.

Kernel: A kernel is a collection of mathematical functions that take data and change it into the desired form. These are most commonly used to find a hyperplane in higher-dimensional space. Linear, Non-Linear, Polynomial, Radial Basis Function (RBF), and Sigmoid are the most commonly used kernels. RBF is the kernel that is used by default. Depending on the dataset, each of these kernelsis used.

Boundary Lines: These are the two lines that are drawn at a distance of s (epsilon) from the hyperplane. It's used to separate the data points by a margin shown in the Figure.4.



Source: https://doi.org/10.1371/journal.pone.0270553.g003

Figure.4 The algorithm of SVR

Random Forest

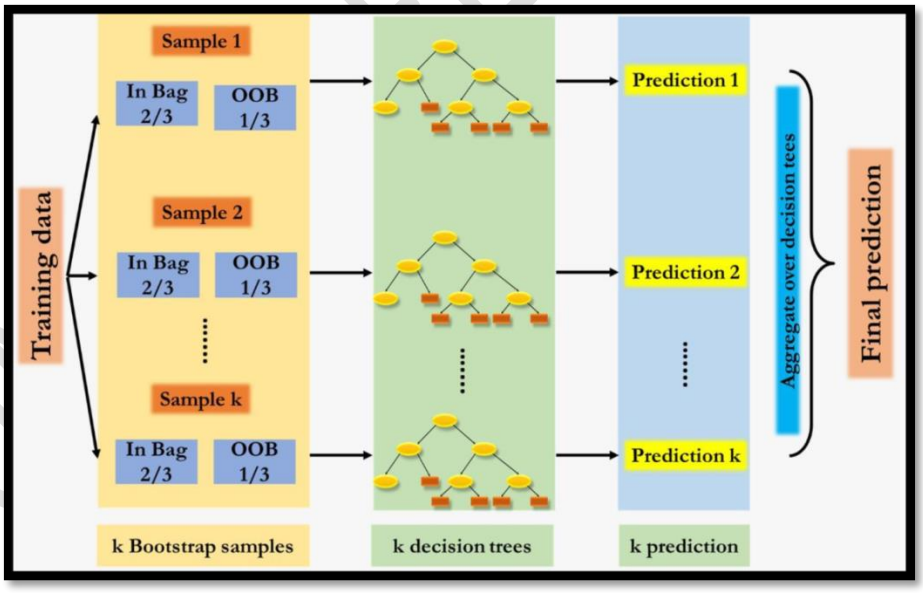
Random forest (RF) is a flexible, easy-to-use machine learning method that, in most cases,

delivers good results even without hyper-parameter tuning. Because of its simplicity and diversity, it is also one of the most often used algorithms. Random forest is a supervised machine learning algorithm. It builds a "forest" out of an ensemble of decision trees, which are generally trained using the "bagging" process. A Random Forestis made up of several trees that are built in a specific "random" manner. Each tree is madeup of a distinct sample of rows, and each node is split up into a different set of features. Each tree has its prediction. After then, the average of these predictions is used to create a single result. The bagging method's general concept is that combining several learning models improves the outcome. The schematic representation of RF model was depicted in Figure 5.

Procedure:

- Randomly select k data points from the training dataset.
- Build a decision tree associated with these k points.
- Choose the number N of trees, you want to build and repeat the above steps.
- Make each of your N-tree trees forecast the value of y for a new data point, and then assign the new data point to the average of all of the predicted y values. Form of the regression trees model

$$f(x) = \Sigma_{m=1}^{M} C_m . 1_{(x \epsilon R_m)} \quad \text{.....(36)}$$



Source: https://doi.org/10.1371/journal.pone.0270553.g003

Figure. 5 The Schematic representation of RF.

3. RESULTS AND DISCUSSIONS:

Secondary data on monthly price series of redgram in Andhra Pradesh were collected from Agricultural Market Intelligence Centre (AMIC) Lam, Guntur from January 1991 to December 2023. There are 396 observations, first 384 observations were used for training data set, used for development of model and last 12 observations were used for validation (testing data set). The actual prices scenario of redgram in Andhra Pradesh was plotted and depicted in Figure 6.

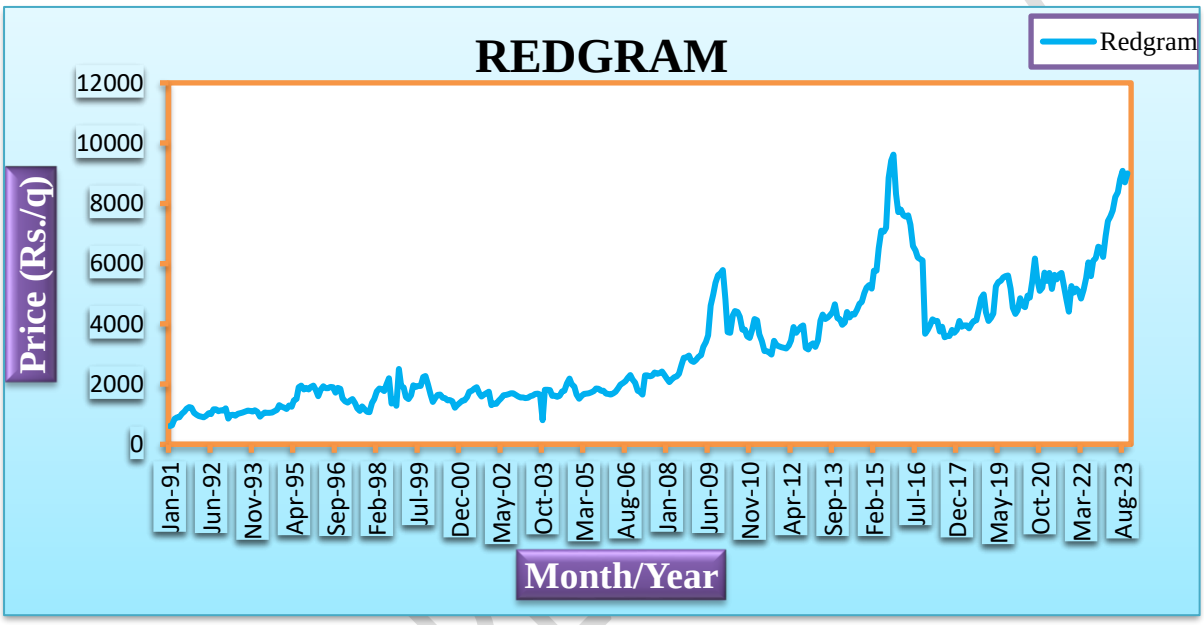


Figure.6 Redgram Actual Prices scenario in Andhra Pradesh during 1991-2023

Descriptive Statistics

Descriptive Statistics were conducted to examine the behaviour of redgram monthly prices of Andhra Pradesh. The findings were depicted in Table 1 provided valuable insights in to the characteristics of the data. It was observed that the prices of redgram during the study period had varied from Rs. 606/q to Rs. 9618.18/q with an average of Rs. 3120.17/q. Standard Deviation was recorded as 1977.93, which indicates that the prices were dispersed highly over the months. It was also revealed that the data was positively skewed and platykurtic in nature. Further lists the summary statistical measures which were self-explanatory. The price series were also verified for the presence of outliers by Grubb's test. It was confirmed that there were no outliers detected from the Grubb's test during the study period.

BDS (Brock - Dechert- Scheinkman) test for non-linearity

To test the linearity characteristics of the price series BDS test was conducted and results of

the test was presented in Table 3. Here, the embedding dimensions were set to 2 and 3. The probability value for both dimensions was 0.00 ($p<0.05$) which indicates that the data under consideration was nonlinear in nature.

Table 2. Summary statistics of Redgram prices of Andhra Pradesh

Statistic	Redgram
No of observations	396
Mean	3120.19
Median	2258.93
Standard Deviation	1977.93
Minimum	606.00
Maximum	9618.18
Skewness	1.00
Kurtosis	0.34
Outliers detected (Grubbs test)	No

Table 3. BDS test for non-linearity in Redgram prices of Andhra Pradesh

Sample	Dimension	Redgram	
		Statistics	Probability
eps (1)	*m*=2	79.16	p<0.0001
	m=3	124.83	p<0.0001
eps (2)	*m*=2	75.36	p<0.0001
	m=3	93.15	p<0.0001
eps (3)	*m*=2	45.31	p<0.0001
	m=3	47.34	p<0.0001
eps (4)	*m*=2	36.18	p<0.0001
	m=3	35.06	p<0.0001

3.3 Autocorrelation (ACF) and Partial Autocorrelation (PACF) plots for Redgram prices of Andhra Pradesh

The Autocorrelation (ACF) and Partial Autocorrelation Function (PACF) plots of redgram price series were depicted in the below figure 3. The figure shown that the prices were autocorrelated, which was supported by Box-Jung test statistic as the probability value was less than 0.05. It indicates that data under consideration was autocorrelated in nature. Once the price series were autocorrelated, the ARIMA model was built for the series. Further,

the redgram prices contains seasonal component which was confirmed by observing ACF and PACF plots in Figure 7. So, SARIMA model was built for the price series.

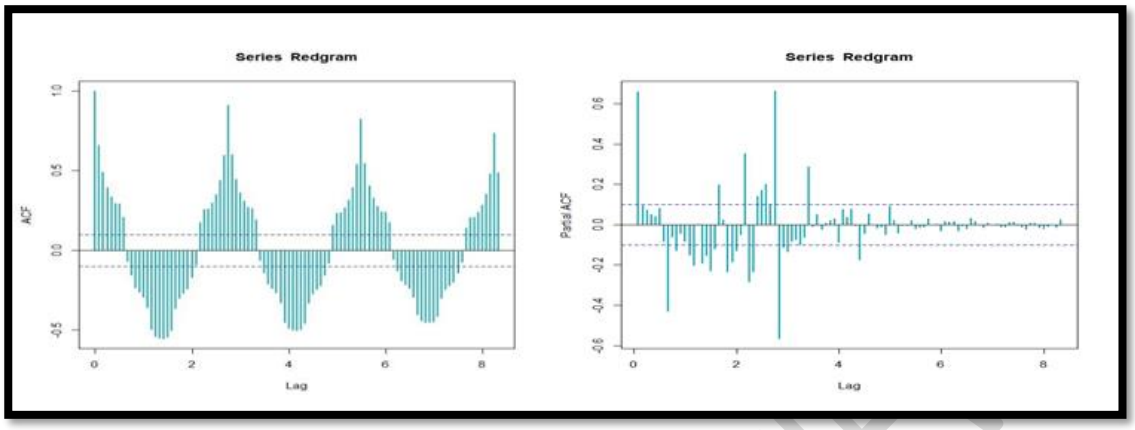


Figure.7. ACF and PACF plots for Redgram prices of Andhra Pradesh

Fitting of SARIMA model

To develop SARIMA mode, first step was testing the stationarity of the data set. Augmented Dickey-Fuller test (ADF) was used to check the stationarity of the data and the results were presented in Table 4. The redgram price series shown the probability value 0.16 ($p>0.05$), confirmed that the data under consideration was non- stationary and became stationary at first difference as probability value was 0.01 ($p<0.05$). Before model estimation to ensure that the data for model consideration was autocorrelated by applying Box- Pierce non-correlation test and it was found significant as the probability value was 0.00 (<0.05) that the data was autocorrelated in nature. All possible SARIMA models were developed and out of the developed models, best performed SARIMA models were presented in Table 5. Among the best performed models, the final SARIMA model order i.e., SARIMA $(2,1,2)(2,0,2)_{12}$ model was selected based on least RMSE, MAE, MAPE and AIC values. The results of selected SARIMA $(2,1,2)(2,0,2)_{12}$ model parameter specification viz., AR, MA, SAR and SMA were presented in Table 6. After determining the SARIMA model order, the model parameters were estimated using maximum likelihood method. After fitting of the model, the diagnostic checking of the residuals by Box- Pierce non-correlation test and it was showed that the residuals were non-autocorrelated in nature as probability value was 0.83 ($p>0.05$). The residuals plot of the best performed SARIMA model was depicted in Figure 8. The modelling and forecasting performance of the training and testing data set were given in Table 13 and Table 14. The research findings of Sanjeev (2022), Biswal and Sahoo (2020), Sabu and Kumar (2020), Mithiya *et al*. (2019) and Venkataviswateja (2018) were also confirmed that SARIMA models were used for forecasting of agricultural commodity prices.

Table 4. ADF test for stationarity of Redgram prices of Andhra Pradesh

	Data type	ADF Statistic	P-value	Decision
Redgram	ADF at level	-2.99	0.16	Non-Stationary
	ADF at 1st Difference	-6.93	0.01	Stationary

Table 5. SARIMA models and Error values for Redgram prices of Andhra Pradesh.

SARIMA model	RMSE	MAE	MAPE	AIC
$SARIMA(1,1,2)(1,0,1)_{12}$	339.43	194.97	6.75	5563.76
$SARIMA(1,1,2)(1,0,2)_{12}$	343.28	192.06	6.65	5574.3
$SARIMA(1,1,2)(1,0,3)_{12}$	339.45	194.98	6.95	5570.25
$SARIMA(2,1,2)(1,0,2)_{12}$	339.78	195.02	6.91	5569.98
$SARIMA(2,1,2)(2,0,2)_{12}$	339.36	194.68	6.90	5569.65
$SARIMA(2,1,2)(3,0,1)_{12}$	339.93	195.63	6.85	5575.26
$SARIMA(2,1,2)(3,0,2)_{12}$	339.98	194.92	6.98	5574.96
$SARIMA(2,1,2)(2,0,3)_{12}$	339.85	195.85	6.89	5571.52
$SARIMA(3,1,2)(1,0,3)_{12}$	340.23	195.28	6.54	5574.26
$SARIMA(3,1,2)(1,0,2)_{12}$	339.63	194.78	6.94	5576.36
$SARIMA(3,1,3)(2,0,3)_{12}$	339.74	194.91	6.92	5570.78
$SARIMA(3,1,3)(2,1,3)_{12}$	339.54	194.85	6.91	5569.89

Table 6. Parameter estimation of SARIMA model for Redgram prices of Andhra Pradesh

Model	Parameters	Estimation	S.E.	Z value	Probability	Model fitting	
SARIMA $(2,1,2)(2,0,2)_{12}$	AR1	-0.44	0.11	-4.21	p<0.0001	Log likelihood	-2775.82
	AR2	-0.6	0.13	-4.53	p<0.0001		
	MA1	0.49	0.15	3.33	p<0.0001		
	MA2	0.69	0.12	5.79	p<0.0001		
	SAR1	-0.44	0.11	-4.21	p<0.0001	AIC	5569.65
	SAR2	-0.6	0.13	-4.53	p<0.0001		
	SMA1	0.49	0.15	3.33	p<0.0001		
	SMA2	0.69	0.12	5.79	p<0.0001		
Box-Pierce test for non-correlations				**Original**	**χ^2 =375.27, p<0.0001**		
				Residual	**χ^2 = 0.047, p= 0.83 (>0.05)**		

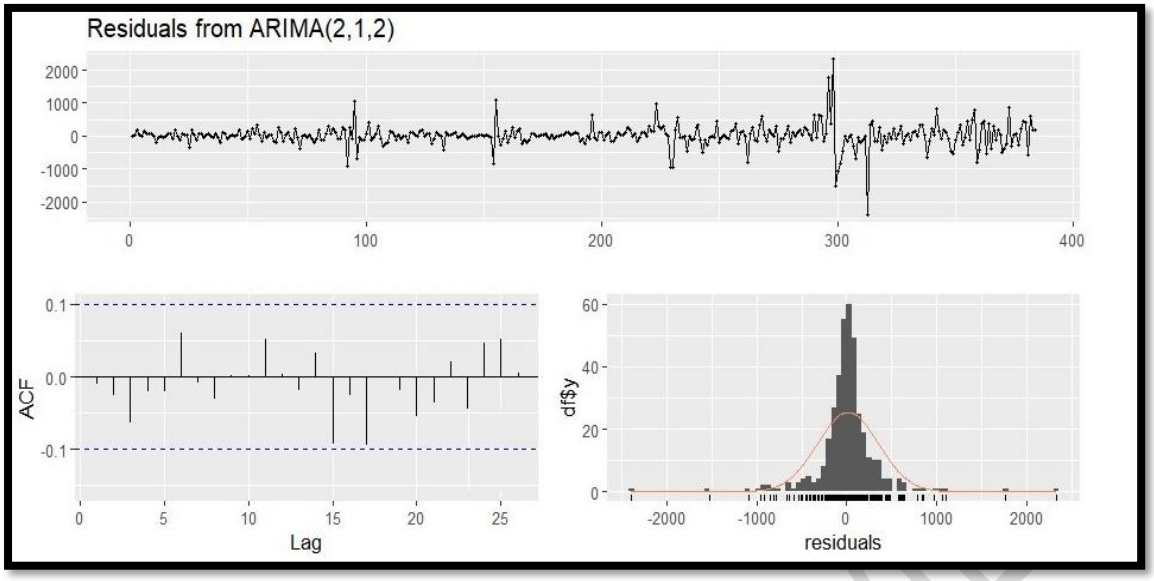


Figure 8. Residual plot of SARIMA model for Redgram prices of Andhra Pradesh

Fitting of GARCH model

To develop GARCH model the first step was selecting suitable SARIMA model. The suitable SARIMA model was selected based on lowest AIC value. Significant lags were identified from ACF and PACF values. ARCH-LM test was used test the heteroscedasticity of the identified significant lags and found that the data was heteroscedasticity nature as probability value was less than 0.05. The results of the ARCH-LM test for redgram price series were depicted in Table 7. The residuals obtained from the fitted SARIMA model was used to fit the GARCH model. After determining the GARCH model order, the model parameters were estimated using maximum likelihood estimation method. The ARMA(1,1),GARCH(1,1) model order was identified as appropriate model for the data under consideration. The results of parameter specification of the redgram price series were outlined in Table 8. After fitting of the model, the diagnostic checking of the residuals by Box- Pierce non-correlation test and it was revealed that the residuals were non-autocorrelated in nature as probability value was 0.07 ($p>0.05$). The residuals plot of the best performed GARCH model was depicted in Figure 9. The modelling and forecasting performance of the training and testing data set were given in Table 13 and Table 14 respectively. Similar results were found by Agbo (2022), Bisht and Kumar (2019), Rojalin *et al.* (2019) and Chi (2018) in their studies GARCH (1,1) model was better performed in forecasting of agricultural commodity prices compared with ARIMA model.

Table 7. ARCH-LM Heteroscedasticity test for residuals of Redgram prices of Andhra Pradesh

Order	LM	p-value
4	487.00	p<0.0001
8	226.90	p<0.0001
12	147.00	p<0.0001
16	58.90	p<0.0001
20	38.10	p<0.0001
24	30.90	0.12

Table 8. Parameter estimation of GARCH for Redgram prices of Andhra Pradesh

Model	Parameters	Estimation	S.E.	t- value	Probability	Model fitting	
ARMA (1, 1) GARCH(1,1)	μ	69.47	12.00	5.79	p<0.0001	Log-likelihood	-2365.09
	AR1	0.98	0.01	173.57	p<0.0001		
	MA1	-0.06	0.01	-4.89	p<0.0001		
	ω	3.55	0.66	5.41	p<0.0001	AIC	16.53
	α	0.06	0.01	6.53	p<0.0001		
	β	0.92	0.01	92.81	p<0.0001		
Box-Pierce test for non-correlations: χ^2= 3.55, p=0.07(>0.05)							

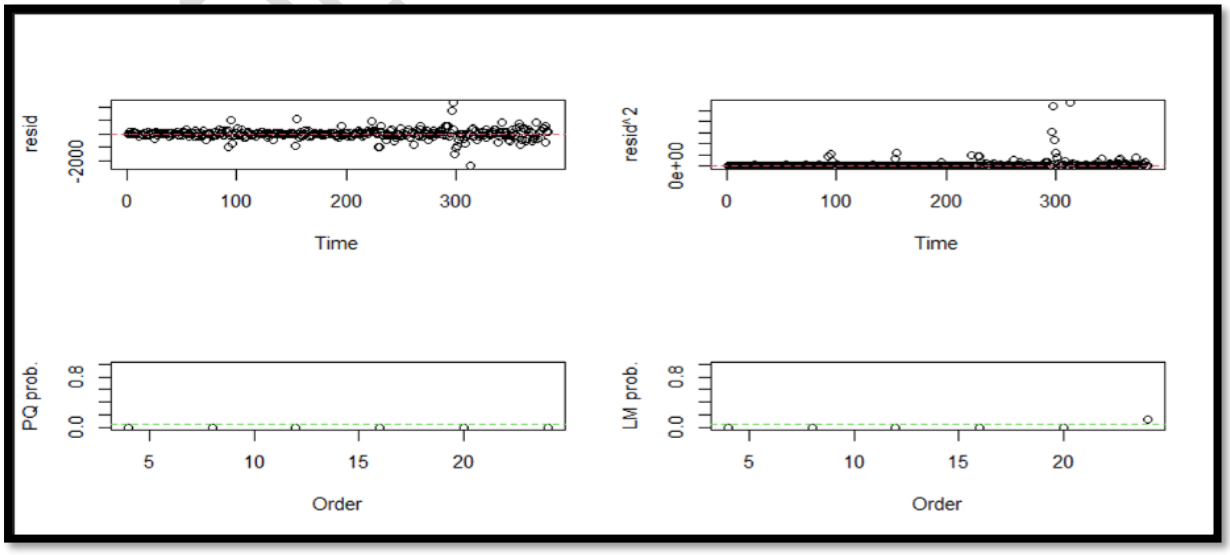


Fig 9. Residual plot of GARCH (1,1) model for Redgram prices of Andhra Pradesh

Fitting of ANNs model

The feed forward network architecture utilized the sigmoidal activation function from input to hidden layers and a linear identity function from hidden layers to the output layer was used to fit redgram monthly prices data set. Prior to model estimation, diagnostic checking of the residuals through the Box-Pierce non-correlation test, it was revealed that the autocorrelations in the residuals as the probability value was 0.00 ($p<0.05$). Various network topologies were trained by increasing the number of hidden nodes from 1 to 25 with sigmoid function as the activation function in the hidden layer. The tested models were detailed in Table 9. Furthermore, the parameter estimation of the best performed ANN model was depicted in Table 10.

The model selected was NNAR (3,12), employed three tapped time delays and 12 hidden nodes (3:12S:1L). This configuration comprised an average of three networks, each being a 3-12-1 network with a total of 61 weights and the selection of the model was based on the lowest RMSE, MAE and MAPE values. After the model development, diagnostic checking of the residuals by employing Box-Pierce non-correlation test indicated that the residuals exhibited non-correlated behaviour with the probability value of 0.42 ($p >0.05$). The residuals plot of the best performed model was depicted in Figure 10. Table 13 and Table 14 provided an overview of the modeling and forecasting performance for both the training and testing datasets. The similar results were found by Sai (2023) in his research NNAR (3-12-1) model performed better for forecasting the productivity of redgram in Karnataka State. Further, several earlier studies conducted by Suthar *et al*. (2023), Paul and Garai (2021) and Srikala (2020) were also consistently affirmed that the superior performance of ANN model for forecasting of agricultural commodity prices compared to univariate time series models.

Table 9. ANN models and their Error values for Redgram prices of Andhra Pradesh

Structure	RMSE	MAE	MAPE
3-1-1	337.92	188.96	6.66
3-2-1	312.87	18393	6.56
3-3-1	291.88	174.49	6.35
3-4-1	286.57	172.96	6.35
3-5-1	280.86	170.64	6.34
3-6-1	271.96	165.36	6.11
3-7-1	264.98	163.42	6.14

3-9-1	**264.61**	**161.12**	**6.02**
3-10-1	260.7	160.64	6.02
3-11-1	257.9	158.72	5.97
3-12-1	254.35	158.22	5.95
3-13-1	255.25	158.34	5.95

Table 10. Parameter estimation of ANN model for Redgram prices of Andhra Pradesh

Parameter	Specification
Input lag	3
Output variable/dependent	1
Hidden nodes	12
Hidden layers	1
Model	3:12S:1L
Total number of parameters	61
Network type	Feed Forward
Activation function (I:H)	Sigmoidal
Activation function(H:O)	Identity
Box-Pierce test for non-correlations : $\chi 2$ =0.64 , p = 0.42	

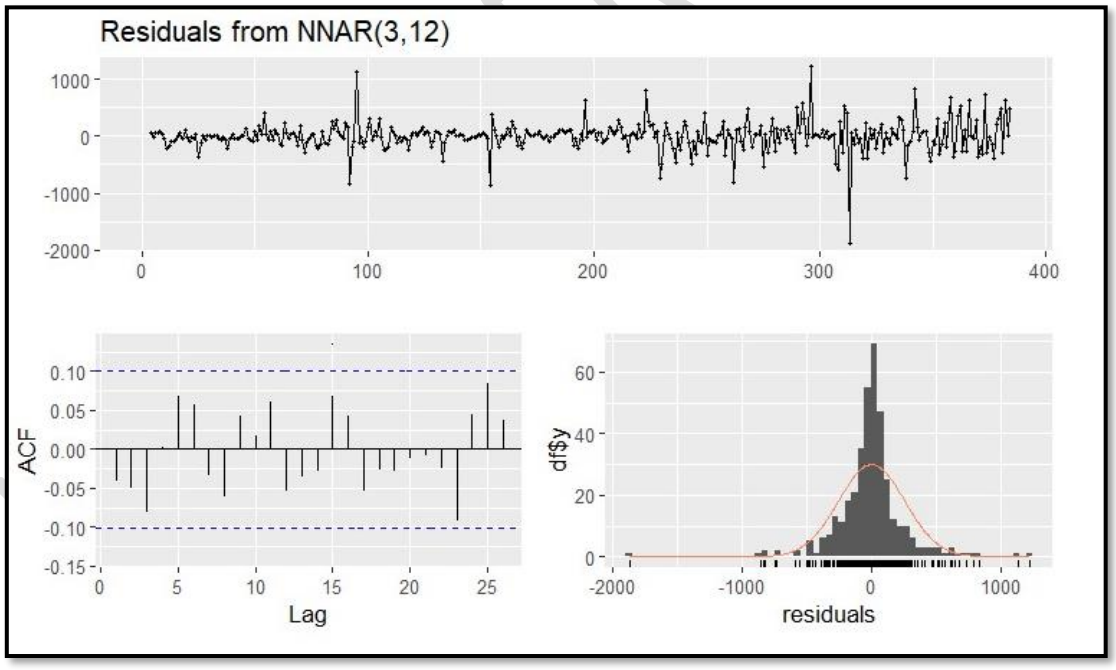


Fig 10. Residual plot of ANN model for Redgram prices of Andhra Pradesh

Fitting of SVR model

The nonlinear Support Vector Regression (SVR) model stands out as a potent machine learning algorithm employed for modelling and forecasting of redgram price series. Radial Bias kernel Function (RBF) was used in this study, as it is suitable for modelling non-linear

problems and can handle complex data distributions. Before model estimation, it was imperative to verify the autocorrelation of the data using the Box-Pierce non-correlation test. The test yielded significant result that the probability value of the actual price series was 0.00 ($p<0.05$), indicated the data was indeed autocorrelated in nature. The parameter specification of the SVR model was depicted in Table 11. Furthermore, the residual test of the fitted model yielded a non-significant result as probability value was 0.62 ($p>0.05$). The residuals plot of the SVR model was depicted in Figure 11. Table 13 and Table 14 showed the SVR model performance metrics of both the training and testing datasets. Similar performance was observed by Minruhi (2023), SVRX model performed better than ANNX and INGARCH models in her research for forecasting of leaf minor population in Groundnut crop. Further, the results were also confirmed by the studies of Saha *et al*. (2021) for forecasting cotton production in India, SVR model performed better than ARIMA model and Rathod *et al.* (2018), SVR model outperformed ARIMA and TDNN models for forecasting of oil seed production in India.

Table 11. Model estimation of SVR for Redgram prices of Andhra Pradesh

Parameters	Specification
Kernel function	RBF
No. of Support Vectors	381
Cost	20
Gamma	1.5
Epsilon	0.001
R^2	0.95
Box-Pierce test for non-correlations: χ^2= 0.24, p=0.62(>0.05)	

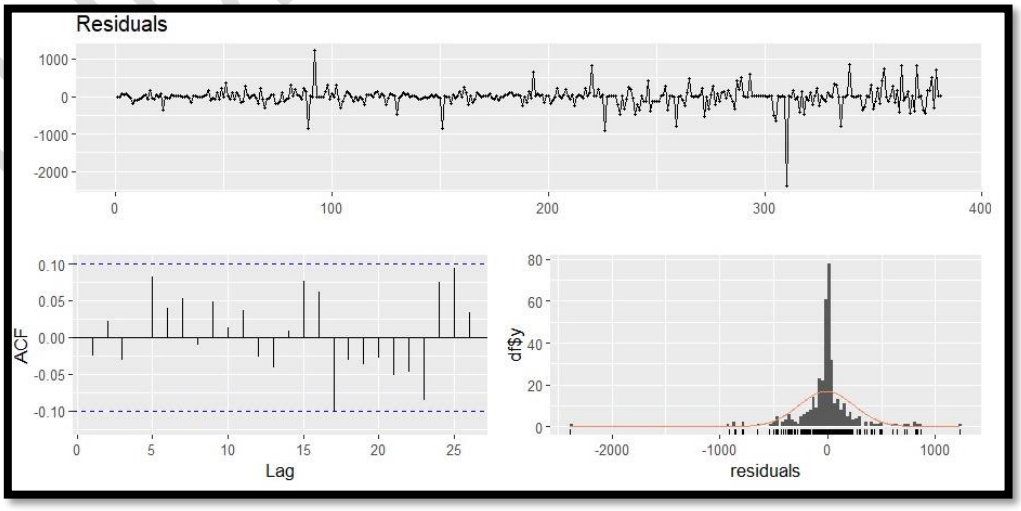


Figure 11. Residual plot of SVR model for Redgram prices of Andhra Pradesh

Fitting of Random Forest:

Random Forest (RF) model was used to fit redgram price series, before model estimation, to ensure that the data under consideration was autocorrelated by applying Box-Pierce non-correlation test, it was found to be significant as the probability value of the actual price series was 0.00 ($p<0.05$) and concluded that the data was autocorrelated in nature. For the development of random forest regression model, random forest algorithm was used. The parameter specification of the best performed RF model was depicted in Table 12. The loss of error per tree was plotted in Figure 12, which means the model stored the loss after each tree. From the Figure 12, it was shown that loss was decreasing at each tree. After model development, diagnostic checking of the residuals by Box- Pierce non-correlation test, it was showed that residuals were non-correlated in nature as probability value was 0.29 **(**$p>0.05$).The modelling and forecasting performance of the training and testing data sets were given in Table 13 and Table 14, respectively. Notably, in this study, the RF model outperformed SVR, ANN, GARCH, and SARIMA models. These findings were in congruent with Paul *et al.* (2022) who revealed that the superior forecasting performance RF and over ARIMA, SVR and GBM models for forecasting of wholesale price of Brinjal in four markets viz., Athagarh, Betnoti, Boudh and Khunthabandha markets of Odisha, India.

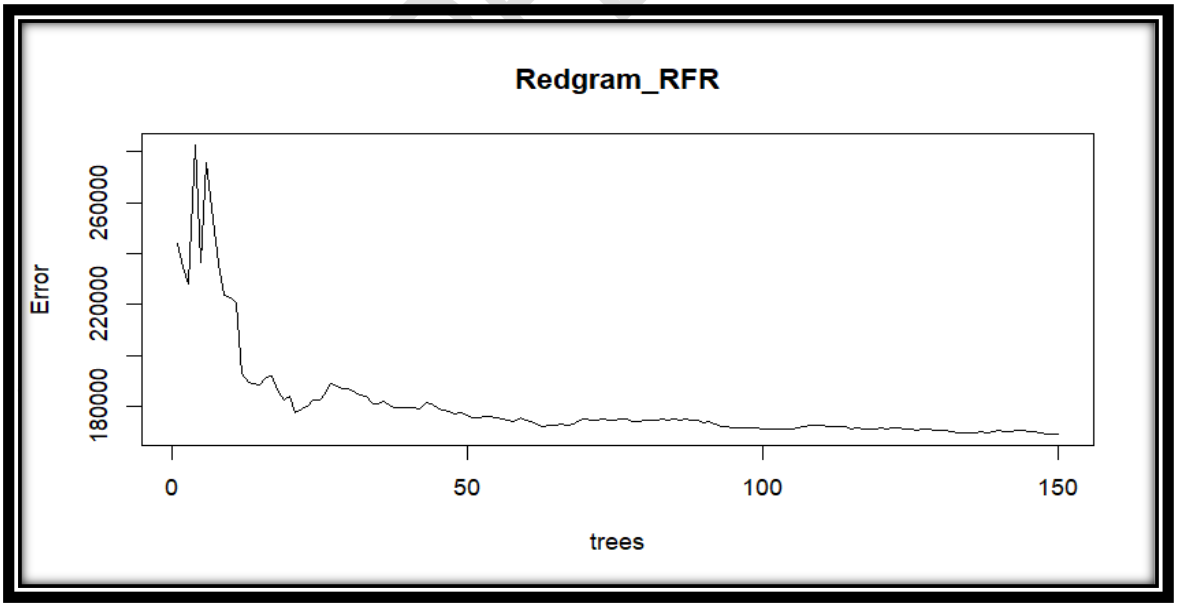


Figure 12. Loss of Error per tree in RF model for Redgram prices of Andhra Pradesh

Table 12. Model specification of RF for Redgram prices of Andhra Pradesh

Parameters	Specification
No. of trees	150
No of Variables taken at each split	3
Depth of the tree	10
R^2	0.95
Box- Pierce non-correlation test	**χ^2 = 0.06, p =0.29(>0.05)**

Table 13. Performance metrics of Training set for Redgram prices of Andhra Pradesh

Training set	SARIMA	GARCH	ANN	SVR	RF
RMSE	339.24	325.42	254.35	245.04	226.54
MSE	115083.78	105898.18	64693.92	60044.6	51320.37

Table 14. Performance metrics of Testing set for Redgram prices of Andhra Pradesh

Testing set	Actual	SARIMA	GARCH	ANN	SVR	RF
Jan-23	6515	6581.92	6598.36	7896.32	7415.98	6770.17
Feb-23	6217	6529.35	6695.32	7706.32	6974.43	6782.15
Mar-23	6919	6503.75	6897.56	7836.79	6188.02	6237.21
Apr-23	7414	6536.68	6897.64	7953.23	8576.24	7514.43
May-23	7567	6568.15	6986.56	7981.36	7143.19	7847.20
Jun-23	7767	6579.63	6984.24	7985.51	7641.16	7560.30
Jul-23	8205	6588.98	7084.23	7963.85	8117.60	7336.19
Aug-23	8373	6604.82	7171.01	8347.69	8969.96	7343.13
Sep-23	8795	6622.22	7225.23	8272.43	9681.97	8313.36
Oct-23	9081	6637.64	7285.69	8674.81	9972.82	9325.84
Nov-23	8703	6652.27	7756.98	8635.56	9894.21	9742.18
Dec-23	8995	6667.46	7896.41	8278.18	8669.19	9421.11
RMSE		**1568.03**	**997.60**	**735.64**	**760.33**	**602.05**
MSE		**2458718.08**	**995205.76**	**541166.21**	**578101.71**	**362464.20**

Model performance in terms of MSE and RMSE for Training and Testingdata sets of Redgram prices in Andhra Pradesh

The prediction accuracy of all the models under consideration for both the training and testing data sets was measured in terms of MSE and RMSE. For forecasting the redgram

prices data of Andhra Pradesh, the models like SARIMA, GARCH, ANN, SVR and RF have been investigated. The above-mentioned models have been evaluated in terms of their prediction capacity as measured by model errors under both training and testing data sets. RF model was performed better than other models in both training and testing sets of redgram prices data of Andhra Pradesh as it yielded lowest MSE and RMSE values. The artificial intelligence-based models outperformed from univariate SARIMA and GARCH models. The performance hierarchy of the model for redgram price series in training dataset were RF>SVR >ANN>GARCH>SARIMA.

The criteria for comparison MSE and RMSE merely showed the observed difference between the predicted values of the models. As a result, the statistical significance difference between the models utilized in the study was determined using Diebold-Mariano test statistic (DM test). In comparison to the RF (M5) model, the SVR (M4) ANN (M3), GARCH (M2) and SARIMA (M1) models were significantly different. It means that RF model outperformed remaining models significantly. The ability of RF model to outperform SVR, ANN, GARCH and SARIMA in the training set was due to its superior capacity of the model and the non-linear nature of the time series data of redgram prices data under consideration. Inter combinational significance of training set were presented in Table 15.

Table 15. DM test for Redgram prices series of Andhra Pradesh

Data type	M1, M2	M1, M3	M1, M4	M1, M5	M2,M3
Training set	-0.87 (0.37)	2.83 (p<0.004)	2.80 (p<0.005)	-2.37 (p<0.05)	2.47 (p<0.05)
Data type	**M2, M4**	**M2, M5**	**M3, M4**	**M3, M5**	**M4,M5**
Training set	3.68 (p<0.001)	-4.69 (p<0.005)	0.0009 (p<0.15)	2.65 (p<0.05)	2.02 (p<0.05)

3. Conclusions:

The redgram monthly price series of Andhra Pradesh, India was analysed by SARIMA,GARCH, ANN, SVR and RF algorithms. The results revealed that RF model was outperformed ARIMA, GARCH, ANN and SVR models in both training and testing data sets. In both training and testing sets in the performance hierarchy of the models were given as follows. The performance hierarchy of the model for redgram monthly prices of training

data set was RF>SVR>ANN>GARCH > SARIMA. The performance hierarchy of the testing data set was RF>ANN>SVR>GARCH>SARIMA.

The criteria's MSE and RMSE were used for comparison merely showed the observed difference between the predicted values of the models. The Diebold-Mariano test statistic (DM test) was used to know the statistical significance difference between the performance of models utilized in the study and the results revealed that the RF model outperforms univariate and Machine learing models like ANN and SVR. The ability of RF model outperforms other models in both training and testing sets was due to its superior capacity to the model and non-linear nature of the time series data under consideration. Furthermore, the diagnostic checking of the residuals obtained by the SARIMA, GARCH, ANN and SVR models were also non-autocorrelated and non-random in nature, which indicate that models under consideration were adequate.

Disclaimer (Artificial intelligence)

Author(s) hereby declare that NO generative AI technologies such as Large Language Models (ChatGPT, COPILOT, etc) and text-to-image generators have been used during writing or editing of manuscripts.

References:

1. Agbo, H.M.S. 2022. Forecasting agricultural price volatility of some export crops in Egypt using ARIMA-GARCH model. *Review of Economics and Political Science*. 8(1). doi:10.1108/REPS-06-2022-0035.
2. Alam, W., Ray, M., Kumar, R.R., Sinha, K., Rathod, S and Singh, N.K. 2018. Improved ARIMAX modal based on ANN and SVM approaches for forecasting rice yield using weather variables. *Indian Journal of Agricultural Sciences.* 88(12): 1909-13.
3. Areef, M. 2020. An application of GARCH and ANN models for potato price forecasting: A case study of Bangalore market. Karnataka State. *Indian Journal of Agricultural Marketing*. 34 (3): 12-19.
4. Breiman, L. 2001. Random Forests. *Machine Learning*. 45: 5-32.
5. Bisht, A and Kumar, A. 2019. Estimating volatility in prices of pulses in India: An Application of Garch model. *Economic Affairs*. 64(3): 513-516.
6. Biswal, A.K and Sahoo, A. 2020. Agricultural Product Price Forecasting using

ARIMA Model. *International Journal of Recent Technology and Engineering*. 8(5): 2277-3878.

7. Bollerslev, T. 1986. Generalized autoregressive conditional heteroscedasticity. *Journal of Econometrics.* 31: 307-327.
8. Box, G.E.P and Jenkins, G.M. 2015. *Time Series Analysis: Forecasting and Control*. Second Edition, Holden Day.
9. Box, G.E.P and Pierce, D.A. 1970. Distribution of Residual Autocorrelations in Autoregressive-Integrated Moving Average Time Series Models. *Journal of the American Statistical Association*. 65 (332): 1509–1526.
10. Chi, W. 2018. Stock Price Short-term Forecasting Based on GARCH Model. *Advances in Engineering Research.* 149 (1): 9-12.
11. Choudhary, K., Jha, G.K., Kumar, R and Mishra, D. 2019. Agricultural commodity price analysis using ensemble empirical mode decomposition: A case study of daily potato price series. *Indian Journal of Agricultural Sciences.* 89(5): 882–886.
12. Cortes, C. Vapnik, V. 1995. Support-Vector Networks. *Machine Learning*. 20: 273-297.
13. Engle, R.F. 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of U.K. inflation. *Econometrica.* 50: 987-1008.
14. Garai, S and Paul, R.K. 2023. Development of MCS based-ensemble models using CEEMDAN decomposition and machine intelligence. *Intelligent Systems with Applications.* 18: 200202. ISSN 2667-3053.
15. Kumari, R.V., Venkatesh, P., Ramakrishna, G and Sreenivas, A. 2019. Chilli price forecasting using auto regressive integrated moving average (ARIMA). International Research Journal of Agricultural Economics and Statistics. 10(2): 290-295.
16. Mallikarjuna, H.B., Paul, A., Paul, A., Noel, A.S and Sudheendra, M. 2019. Forecasting of black pepper price in Karnataka state: An application of ARIMA and ARCH models. *International Journal of Current Microbiology Agricultural Science*. 8 (1): 1486-1496.
17. Minruhi, P. 2023. Development of weather based statistical and machine learning forewarning models for major Pests in groundnut crop. *M.Sc. (Ag.) Thesis*. Acharya N.G. Ranga Agricultural University. Lam, Guntur.

18. Paul, R. K., Yeasin, M., Kumar, P., Kumar, P., Balasubramanian, M and Roy, H. S. 2022. Machine learning techniques for forecasting agricultural prices: A case of brinjal in Odisha, India. PLoS ONE.17(7): e0270553. doi.org/10.1371/journal.pone.0270553.
19. Rathod, S., Singh, N.K., Patil, G.S., Naik, H.R., Ray, M and Meena, S.V. 2018. Modelingand forecasting of oilseed production of India through artificial intelligence techniques. *Indian Journal of Agricultural Sciences.* 88(1): 22-28.
20. Rojalin, P. A. N. I., Biswal, S.K and Mishra, U. S. 2019. Green gram weekly price forecasting using time series model. *Revista Espacious*. 40(7): 15.
21. Sai, K.S., Sreenivasulu, K. N., Srinivasa Rao, V and Sitarambabu, V. 2023. Forecasting Models for red gram area, production and productivity in Karnataka. *The Andhra Agricultural Journal*. 70 (2): 233-245.
22. Saha, A., Singh, K. N., Roy, M., Rathod, S and Choudhury, S. 2021. Modelling and forecasting cotton production using tuned-support vector regression. *Current Science*. 121(8): 1090-1098.
23. Sanjeev, Kundu, R., Sharma, A and Preeti. 2022. Development of Seasonal ARIMA Model to Predict Wholesale Price of Rice in Delhi Market. *Current Journal of Applied Science and Technology*. 41(48): 155–161. doi.org/10.9734/cjast/2022/v41i484050.
24. Suthar, B., Pundir, R.S and Popat, R. 2023. Groundnut price forecasting in Gondal market of Gujarat: A comparison of arima and ann models. *Indian Journal of Agricultural Marketing*. 37 (1): 51-60.
25. Venkataviswateja, B., Srinivasa Rao, V., Umar, S.N and Reddy, M.C.S. 2018. A study on arrivals and prices of red Chillies in Guntur market yard- A Time Series approach. *International Journal of Current Research in Multidisciplinary (IJCRM)*. 3(5): 06-10.
26. Mithiya, Debasis, Lakshmikanta Datta, and Kumarjit Mandal. 2019. "Time Series Analysis and Forecasting of Oilseeds Production in India: Using Autoregressive Integrated Moving Average and Group Method of Data Handling – Neural Network". Asian Journal of Agricultural Extension, Economics & Sociology 30 (2):1-14. https://doi.org/10.9734/ajaees/2019/v30i230106.
27. De Gooijer JG, Hyndman RJ. 25 years of time series forecasting. International journal of forecasting. 2006 Jan 1;22(3):443-73.

28. Wu H, Wu H, Zhu M, Chen W, Chen W. A new method of large-scale short-term forecasting of agricultural commodity prices: illustrated by the case of agricultural markets in Beijing. Journal of Big Data. 2017 Dec;4:1-22.