Forecasting Cocoa Prices Using Statistical Models: A Comparative Study

Xuanqi Wei, Wenyi Li, Fangning Zhang, Haowei Fan

March 26, 2026

Abstract

This paper forecasts monthly cocoa prices using data from the International Cocoa Organization and climate records from Ghana. After standardized the data, a series of time series models were applied to forecast the prices, including Exponential Smoothing (ETS), ARIMA, SARIMA, linear regression with climate covariates, GARCH, and XGBoost models. Linear Regression Model demonstrated the strongest predictive accuracy, indicating that cocoa prices are likely to rise substantially in the coming decade. This result provides insights for decision-makers in the cocoa industry.

Contents

1 Introduction

Forecasting commodity prices is a challenge in economics and statistical modelling because of the multi-factorial drivers of price behaviour. However, forecasting commodity prices is important for producers, traders and policymakers to have more understanding about the market. Cocoa is a globally traded commodity with significant economic relevance, particularly in regions where it is both produced and consumed at scale such as Ghana. For stakeholders such as producers, traders, and policymakers, accurate forecasting is vital to design procurement strategies, manage supply chain risks, and stabilize income.

Some real-world examples also show the importance of forecasting the Cocoa price. In 2016–2017, global cocoa prices declined by over 30% [6], leading to significant income losses for smallholder farmers in Ghana. However, cocoa exports constitute a major share of national revenue in Ghana. This forced some farmers to abandon cocoa cultivation or turn to alternative livelihoods, including environmentally damaging activities such as illegal mining [3]. In order to stabilize the price of cocoa, Ghana and Côte d'Ivoire jointly introduced the Living Income Differential (LID) in 2019, establishing a $400-per-ton premium on cocoa exports to support farmer incomes [9]. This real-world example shows how price instability can widely influence the economic and social consequences, and highlights the importance of forecasting models.

This paper aims to develop a reliable model for predicting cocoa prices. The paper investigates the monthly behaviour of cocoa prices by using two key datasets, including daily cocoa futures prices from the International Cocoa Organization and daily climate data from Ghana, the largest cocoa-producing country in the world. The analysis focuses on modelling the monthly change in log-transformed cocoa prices. The differencing method was use to address non-stationarity. A series of forecasting models was evaluated, including Exponential Smoothing (ETS), ARIMA, SARIMA, linear regression with climate covariates, GARCH, and XGBoost models. Each model was trained on a 70% subsample and assessed using the remaining 30% sample, which is a 70/30 train-test split. Forecast accuracy was assessed with root mean square error (RMSE), AIC, and BIC, with all predictions back-transformed to the original price scale.

Among the models tested, the linear regression model demonstrated the strongest predictive performance, achieving the lowest RMSE and MAE. The model forecasts a significant long-term decrease in cocoa prices. This downward trend has different implications for stakeholders. While consumers may benefit from lower prices, price volatility could still pose risks for smallholder farmers. Policymakers should consider relevant policies to control fluctuations in cocoa prices or policies to protect both consumers and producers. Traders and chocolate manufacturers may need to adjust procurement strategies to account for sustained lower input costs. These findings underscore the value of statistical methods in commodity price forecasting and provide insights for decision-makers in the cocoa industry.

2 Literature Review

Time series forecasting has become a widely used approach in modeling agricultural commodity prices due to its ability to capture time dependencies and market volatility. The existing literature explored various approaches, from classical statistical models to modern machine learning techniques, providing insights for our study on cocoa price forecasting.

Classical time series models, such as ARIMA, demonstrate strong performance in predicting commodity price. [7] compared different forecasting techniques for coffee prices, including Moving Average (MA), ARIMA, and decomposition methods. Their findings showed that ARIMA has reliable performance across both international and domestic markets. ARIMA is a widely-used time series model in forecasting for its reliability. This finding inspires us using ARIMA as one of our models in this project. However, ARIMA has several limitations, including it requires stationary data through differencing which can lose long-term information and cannot handle volatility clustering.

The paper [1] demonstrated that combines ARIMA with artificial neural networks can have better forecasting performance when dealing with nonlinear patterns and volatility clustering in agricultural export prices. Motivated by these findings, our paper applies a hybrid ARIMA-GARCH model to capture both the trend and volatility structure in cocoa price.

Building upon Novanda's research, the paper [5] conducted a advanced comparative analysis and emphasized data preprocessing in the research. They identified and removed nonstationary components such as seasonality and trend first and then use the Partial Autocorrelation Function (PACF) to make lag selection. They compared several forecasting techniques, including Exponential Smoothing (ES), Autoregressive (AR), ARIMA, Multilayer Perceptron (MLP), and Extreme Learning Machines (ELM), to find the most accurate prediction model. This study offers insights into model selection strategies for time series forecasting. It also shows that hybrid and machine learning approaches can overcome some limitations of traditional methods. Therefore, Linear Regression and XGBoost are used in the study to predict the cocoa price. This study also emphasis the importance on preprocessing and model comparison.

Finally, [8] investigated the impact of world cocoa prices on cocoa production in Ghana using a regression model with ARIMA errors. While both their study and ours are about Ghanaian cocoa, the focus are different. The paper [8] emphasized the effect of international prices on production, while our analysis aims to forecast cocoa prices with climate variables as potential predictors.

Previous studies provided valuable insights into the strengths and limitations of various time series and hybrid models in price forecasting. With these insights, we developed a advanced method to forecast. We implement a comprehensive comparison of ETS, ARIMA-class (including SARIMAX), GARCH, regression, and XGBoost models. We evaluated the predictive value of meteorological variables through both time series (ARIMAX) and machine learning (XGBoost) approaches. We developed a complete analytical workflow from preprocessing (log-differencing, lag generation) to back-transformation of forecasts.

3 Methodology

Before building the models, we reviewed existing research and found weather conditions (such as rainfall and temperature) significantly influence cocoa price fluctuations [2]. Based on this, we selected two main datasets. The first includes historical cocoa prices over time. The second dataset contains weather data for the same period, including rainfall, average temperature, maximum temperature, and minimum temperature.
Before modeling, we preprocessed the raw data. Since daily data tends to be highly volatile and our focus is on long-term trends and seasonal patterns, we aggregated the daily data by month. For each month, we calculated the average cocoa price, average temperature, average maximum temperature, and average minimum temperature to represent that month. When calculating monthly averages, we ignored individual missing days. If an entire month was missing, we filled in the values using the average of the previous and following months. After, we aligned the price and weather data to ensure the timestamps matched exactly. Finally, we split the dataset into a training set and a testing set using a 7:3 ratio.

3.1 Time Series Models

Considering that Novanda and his colleagues identified time series models as the best method for predicting coffee prices [7], and given the similarities between coffee and cocoa—such as they are both sensitive to climate and they are both major global commodities [4]—we believed time series models could also perform well in predicting cocoa prices.
In the early stage of modeling, we tested several time series models to capture the trend and seasonal patterns in cocoa prices. These models included ETS, ARIMA, SARIMA, and SARIMA-GARCH. To meet the stationarity requirement of time series models, we applied a log transformation and first-order differencing to the price data.

3.1.1 ETS Models

To build the ETS model, we first plotted the time series of cocoa prices to examine long-term trends and seasonal patterns. For trend analysis, if the data showed a linear increase or decrease, we used an additive trend. If the trend appeared exponential—where higher price levels were linked to larger changes—we used a multiplicative trend.
Similarly, for seasonality, if seasonal fluctuations were roughly constant each year, we used an additive seasonal structure. If the size of seasonal changes grew or shrank with the price level, we used a multiplicative structure. However, certain components—such as the error term or unclear seasonal patterns—were hard to classify visually. For these cases, we used automated model selection to fit multiple ETS models and compared their corrected Akaike Information Criterion (AICc) values. We selected the model with the lowest AICc as the final ETS to balance model complexity and goodness of fit.

3.1.2 ARIMA and SARIMA Models

For the ARIMA and SARIMA models, we set the differencing parameter d=0 because the data had already been transformed to achieve stationarity. We then plotted the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the differenced series to examine the cutoff or tailing patterns of lags. This helped us make an initial judgment about the non-seasonal orders p and q, as well as the seasonal orders P and Q. For example, if the ACF drops sharply after a certain lag while the PACF tails off, it may suggest a strong moving average (MA) component and a weak autoregressive (AR) component, and vice versa.
When the ACF or PACF plots showed clear patterns, we used them to determine the non-seasonal order (p, d, q) and seasonal order (P, D, Q, s), then built the corresponding ARIMA or SARIMA models. However, in practice, the ACF and PACF of the price series often lacked clear cutoffs, and the decay of lags was vague, making it hard to identify optimal orders visually.
To address this, we used a grid search to fit models across a range of parameter combinations. We evaluated each model using the AICc and selected the model with the lowest AICc as the final ARIMA and SARIMA model, ensuring the best fit in the presence of structural complexity.
In order to improve model sensitivity to real-world factors, we included weather data as exogenous variables in the modeling process. These monthly variables, such as average temperature and precipitation, were aligned with the price data.

3.1.3 SARIMA-GARCH Model

ARIMA and SARIMA models assume constant variance in their error terms. This assumption can lead to large prediction errors when the time series shows clear signs of conditional heteroskedasticity. To better capture potential volatility dynamics in the price series, we extended our analysis by building a SARIMA-GARCH model. This approach incorporates volatility modeling into the framework to improve stability and predictive performance. For the GARCH component, we used the commonly applied GARCH(1,1) specification.

3.2 General Multiple Linear Regression Model

In addition to time series models, we also used a general multiple linear regression model based on weather variables to explore how weather affects cocoa prices from a different perspective. Since weather may influence prices with a time lag, we included lagged weather variables as predictors.
This approach helps capture delayed effects of weather on future prices and improves the model's ability to respond to time dynamics. It also supports the assumption of independent error terms in linear regression by reducing the risk of autocorrelation.
To address the missing values created by lagged variables, we removed any records with incomplete data to ensure the regression was estimated using a full and valid sample. We also applied a log transformation to the cocoa price data. This reduced volatility, improved stability, and helped meet the normality and linearity assumptions of the model. As a result, the regression modeled the log of the expected price, log(E(Y—X)), rather than the raw expected price.

3.3 Gradient Boosting Decision Tree Model

Building on the previous linear regression methods, we introduced a gradient boosting decision tree model to capture more complex, nonlinear relationships behind price fluctuations. This model combines multiple lagged values of historical prices as key time series features with weather variables—including precipitation, average temperature, maximum temperature, and minimum temperature—that are closely linked to price changes.
As with the linear model, we applied a log transformation to the price data to improve model stability and better align with the distribution of prediction errors.
During feature engineering, we created multiple lagged variables and aligned all weather features to ensure consistency. After preprocessing, we tuned the model by adjusting tree depth, learning rate, and the subsampling ratios for features and samples.

3.4 Model Selection and Validation

3.4.1 Model Selection Criteria

After fitting the models, we carried out evaluation and diagnostic analysis to assess both the fit on the training set and the generalization performance on the test set. For model selection, we used the AICc as the primary indicator. AICc balances model fit with complexity, helping to prevent overfitting. A lower AICc value indicates a better balance between complexity and fit, so we prioritized models with the lowest AICc scores.
For residual diagnostics, we examined whether the residuals met the white noise assumption. We plotted standardized residuals to check for randomness and to identify any signs of trend or structural bias. We also plotted the residual autocorrelation function (ACF) to see whether it cut off quickly after a small number of lags. Ideally, if the ACF shows no significant autocorrelation beyond lag 2, the model has likely captured most of the systematic information in the data.
We further conducted the Ljung-Box test. A p-value above the common threshold of 0.05 means we cannot reject the null hypothesis that the residuals are white noise, supporting the model's validity.
For the linear regression model, we used a Q-Q plot to visually check if the residuals followed a normal distribution. We also calculated the variance inflation factor (VIF) for each predictor to assess multicollinearity. A VIF above 10 suggests a potential collinearity issue that may require variable removal or transformation. VIF values consistently below 5 indicate that the model has a stable variable structure.

3.4.2 Prediction and Back-Transformation

After evaluating the models on the training set, we tested their predictive performance on the test set. It is important to note that different models applied different transformations to the cocoa price data during training.

Therefore, we reversed these transformations before evaluation to ensure comparability of the results.
For the linear regression and XGBoost models, which were trained on the log-transformed prices, we applied an exponential transformation to the predictions to recover the actual price levels.
In contrast, the time series models—ETS, ARIMA, SARIMA, and SARIMA-GARCH—used both log transformation and first-order differencing to achieve stationarity. For these models, we first reconstructed the original log price series by cumulatively summing the differenced predictions. We then applied the exponential transformation to obtain the predicted actual prices.

3.4.3 Evaluation Metrics and Final Model Selection

We evaluated model performance by using the optimal parameters estimated from the training set to predict cocoa prices in the test set. We then calculated several standardized prediction error metrics to assess accuracy: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Sum of Squared Errors (SSE).
MAE measures the average absolute difference between predicted and actual values; lower values indicate better overall accuracy. RMSE gives more weight to larger errors, so a lower RMSE suggests the model performs well in avoiding large deviations. MAPE expresses prediction error as a percentage of the actual values, allowing for comparison across models and scales; lower values reflect smaller relative errors. SSE is the total of squared residuals and captures the model's overall error; smaller values are preferred.
To avoid relying on a single metric, we compared all four indicators and paid particular attention to MAE and RMSE. Among the candidate models, we selected the one that showed consistent performance and relatively low errors across multiple metrics as the final prediction model. This model not only fit the training data well but also demonstrated strong and stable predictive accuracy on unseen data.

4 Data

This study uses two sources of data: international cocoa price data from the International Cocoa Organization (ICCO) and local climate data from Ghana. The ICCO dataset provides daily cocoa futures prices (in USD per tonne), while the climate dataset includes daily measurements of precipitation and temperature from a major cocoa-producing region in Ghana. The observation is from October 1994 to November 2024, which allows both long-term trends and short-term seasonal effects analysis.
To prepare the data for time series analysis, several preprocessing steps were undertaken. After importing the dataset, prices were converted to numeric values and date formats were standardized. The climate data for each day was the average of existed multiple observations of that day. The two datasets were merged by date and the data was summarized on a monthly basis. The dependent variable, *Price*, represents the monthly average of daily cocoa prices.

	Date	Price	Daily Perception	Average Temperature	Maximum Temperature	Minimum Temperature
Min.	1994-10-12	778.4	0.00000	73.60	76.50	61.00
1st Qu.	2005-02-01	1689.4	0.00000	78.56	85.50	72.57
Median	2014-10-16	2330.7	0.07775	80.50	88.67	73.67
Mean	2012-09-06	2589.3	0.24756	80.62	88.44	73.85
3rd Qu.	2020-09-16	2931.2	0.30042	82.60	91.25	75.00
Max.	2024-11-28	10690.7	10.28000	88.00	101.00	82.00

Table 1: Summary statistics of cocoa price and weather variables

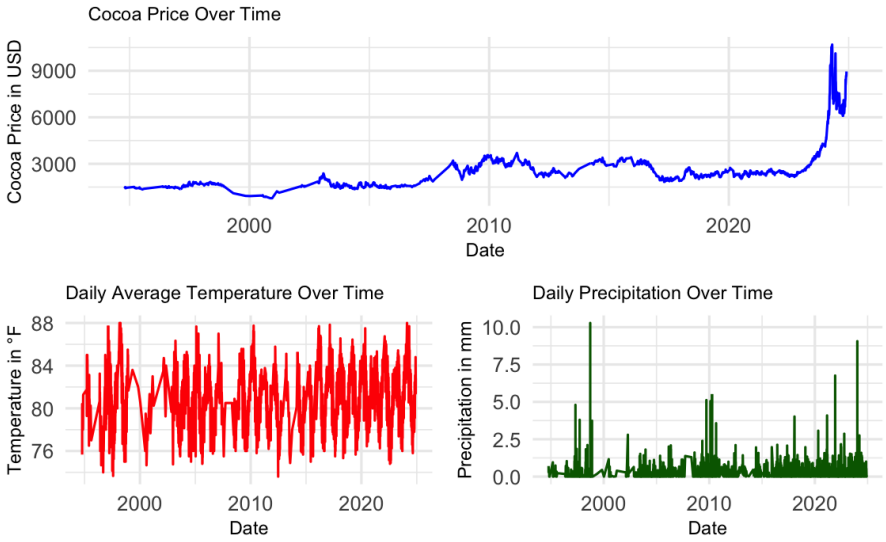

Figure 1: Time Series of Cocoa Price, Local Average Temperature, and Precipitation. The figure displays three parallel time series: (1) Daily cocoa price fluctuations (USD/ton), (2) Daily average temperature (ϕ) in major cocoa-growing regions of Ghana, and (3) Daily precipitation levels (mm). Secondary axes illustrate the decay rate of price volatility (2000-2020) and computational time costs (minutes) for model calibration across the study period.

In Figure 1, the top panel shows the trend of cocoa prices over time (in USD per tonne). From 1994 to around 2015, prices fluctuated modestly between USD 1500 and USD 3500. However, prices increase dramatically in recent years, which justifies the use of volatility-sensitive models such as GARCH. The left bottom temperature graph shows a seasonal cycle, fluctuating between roughly 76°C and 88°C, without strong long-term trend. The bottom-right graph presents daily precipitation levels. The majority of observations are close to zero, but there are some extreme outliers. This indicates there are some heavy rainfall days existing.
In Figure 2, it presents the STL (Seasonal-Trend-Loess) decomposition of the cocoa price time series. The seasonal component captures strong seasonal components, showing yearly cycles likely related to cocoa harvesting seasons or international price trends. The trend component indicates an significant increase in prices starting

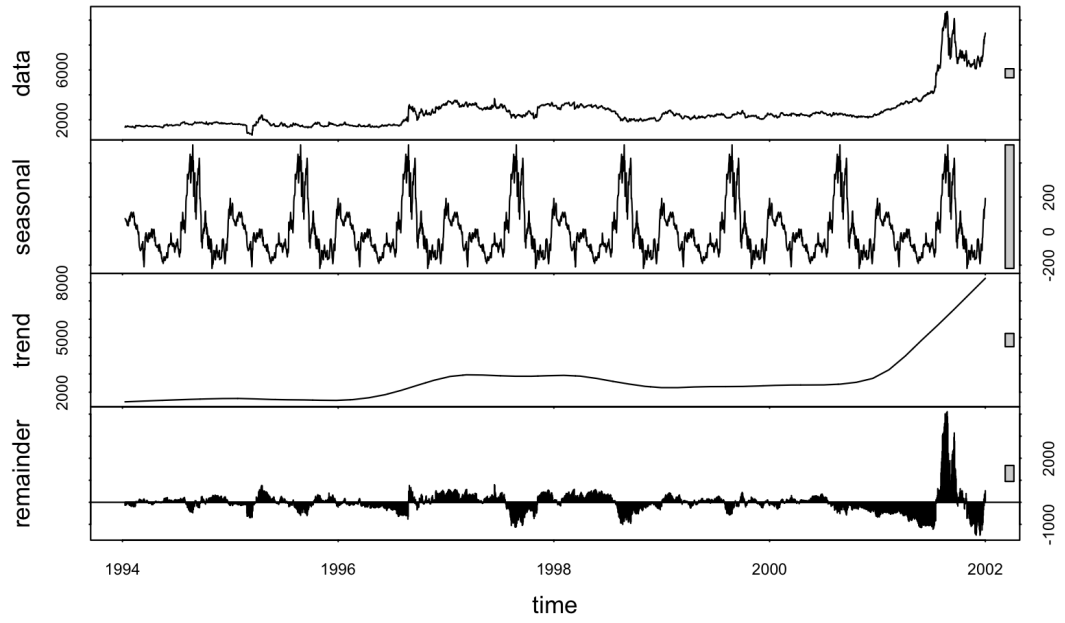

Figure 2: Time Series Decomposition of Cocoa Prices (1994-2002). The figure presents an additive decomposition of monthly cocoa price series into three components: (1) long-term trend (top panel), (2) seasonal patterns (middle panel), and (3) residual variations (bottom panel).

around 2001. The remainder component highlights short-term fluctuations and irregularities not captured by the trend or seasonality.

5 Forecasting and Results

5.1 Exponential Smoothing (ETS) Model

Our ETS model was built on the pre-processed time series, where the original cocoa prices were transformed using a logarithmic transformation followed by differencing. The data is divided in to training and testing set, 70% and 30% respectively. The model is shown below:

$$y_t = \ell_{t-1} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2)$$
$$\ell_t = \ell_{t-1} + \alpha\epsilon_t, \quad 0 < \alpha < 1,$$

where, y_t is the differenced log price at time t, ℓ_t is the state at time t, α is the smoothing parameter, and ϵ_t is a white noise error term.
To validate, during the test period, forecasts were made on the differenced log scale. I'll present that the first order difference log of training set (or the whole dataset) make it becomes stationary. The same validation works for traning data for ARIMA and SARIMA Model.

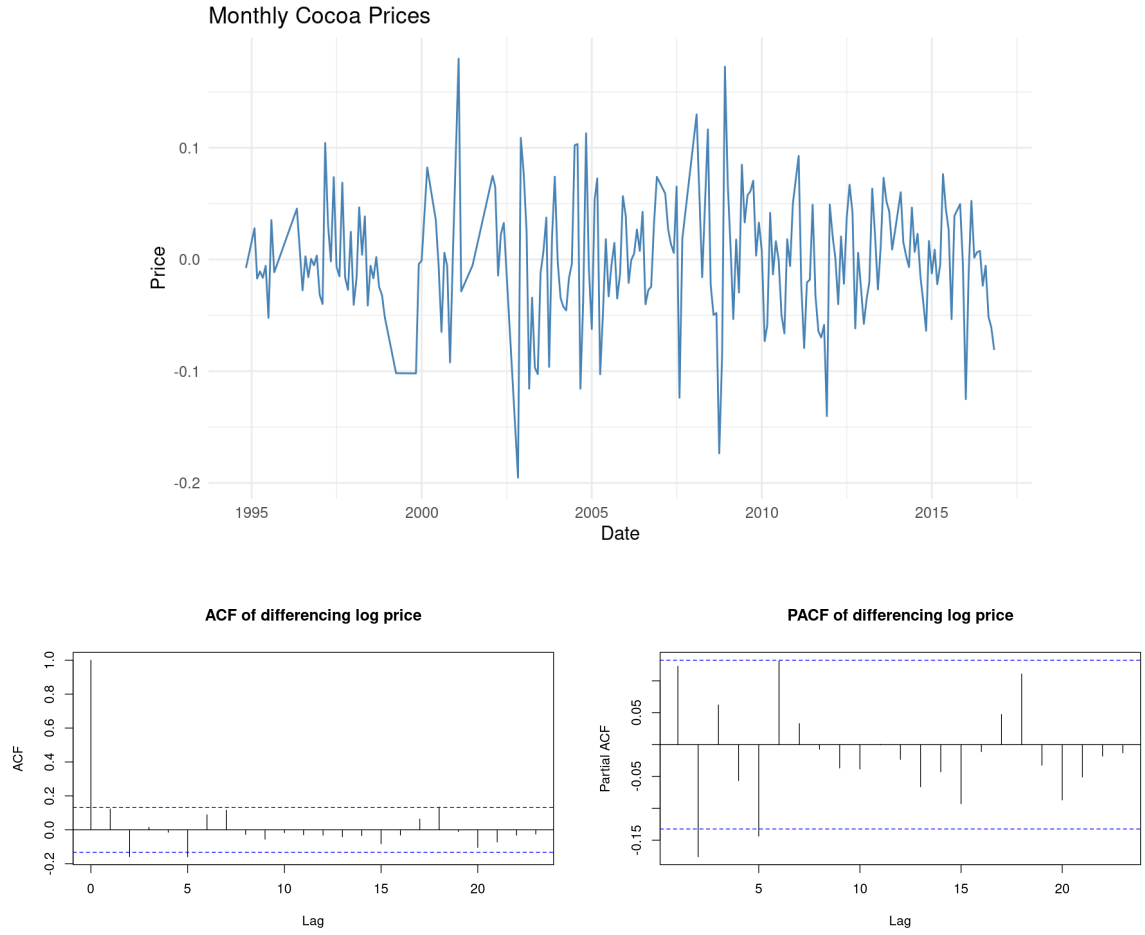

Figure 3: We've shown the plot for monthly Cocoa Price after the difference log beeing applied. The ACF and PACF are shown below.

After the process, the differenced series appears stationary as in the 'Differenced Sales v.s. Time' plot, no obvious trend is shown and the variance is constant. Also, the ACF and PACF are lies within the cut-off. Thus, we validate the differenced log scale for all the models.
To evaluate how well the models performed, their predictions were compared to the actual test data using the *accuracy()* function. After that, the forecasts were converted back to the original price scale by reversing the

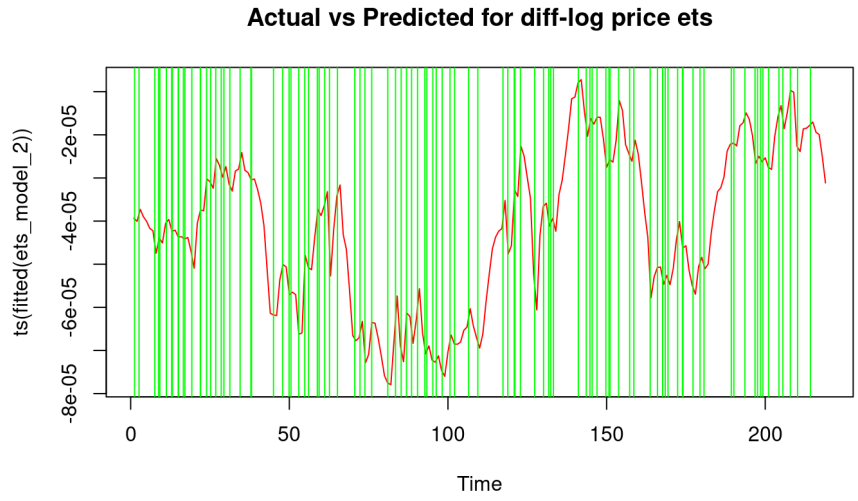

Figure 4: Actual v.s. Predicted for diff-log pirce of ETS Model. The green line represents predicted prices and the actual prices are represented by black line.

differencing and log transformation through a cumulative sum method. Lastly, the predicted prices were plotted with the actual prices to get a visual sense of how well the models captured the trends.
Since $t \propto log(t)$, although we are using the diff-log, the real price shows the same trend. Thus, if log price data fits the log actual data plot well, then the real price data fits the actual data plot well. However, the plot shows that no close alignment is revealed with actuals. This suggests that the model might be less sensitive to abrupt market changes.

5.2 ARIMA and SARIMA Models

Several ARIMA models were fitted on the same differenced log-transformed series to capture the time-series dynamics:

- ARIMA(2,0,2) with External Regressors: Specified with orders (2, 0, 2) and augmented with external regressors (weather variables: PRCP, TAVG, TMAX, TMIN).

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \boldsymbol{\beta}^\top \mathbf{x}_t + \epsilon_t$$

  where y_t is the differenced log price at time t, ϕ_1, ϕ_2 are the autoregressive coefficients, θ_1, θ_2 are the moving average coefficients, $\mathbf{x}_t$ is the vector of external regressors (e.g., `PRCP`, `TAVG`, `TMAX`, `TMIN`), $\boldsymbol{\beta}$ is the corresponding vector of coefficients, and ϵ_t is the white noise error term.

- ARIMA(2,0,5) with External Regressors:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \theta_3 \epsilon_{t-3} + \theta_4 \epsilon_{t-4} + \theta_5 \epsilon_{t-5} + \boldsymbol{\beta}^\top \mathbf{x}_t + \epsilon_t.$$

- SARIMAX Model (Seasonal ARIMA with Exogenous Variables): An automatic seasonal ARIMA was estimated with *auto.arima()*, incorporating the same external regressors.

Comparing the AIC and ACF values betwee the two ARIMA models, we chose the ARIMA(2, 0, 2) to be the one we want. Then, residual diagnostic checks were carried out using *tsdiag()* for each model to make sure there wasn't any noticeable autocorrelation below in the residuals. Diagnostic checks, including ACF plots and the Ljung-Box test (p-value ¿ 0.05), confirmed that the residuals from the ETS and ARIMA models did not exhibit significant autocorrelation, validating the adequacy of these models.
Both the ARIMA(2, 0, 2) and SARIMAX models were then used to generate forecasts on the differenced log-transformed series. These predictions were compared to the test set using standard accuracy metrics. Just like with the ETS model, the forecasts were then converted back to the original price scale, and the predicted values

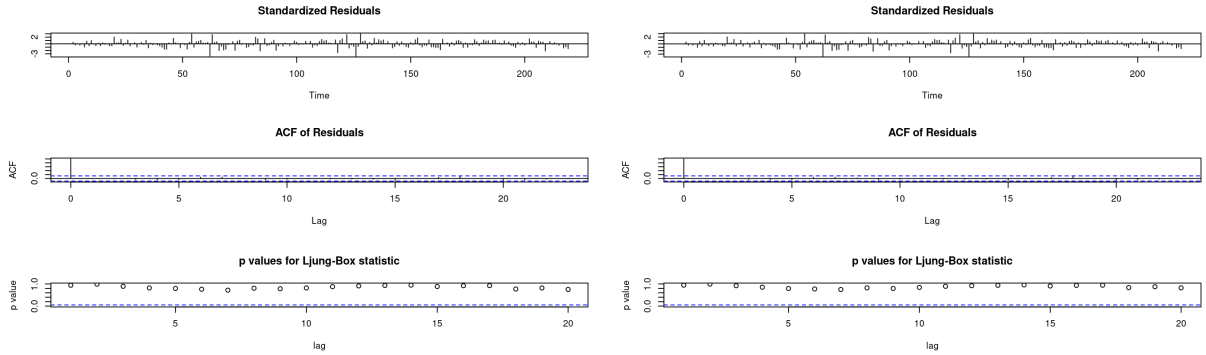

Figure 5: Residual diagnostic for arima(2, 0, 2), sarima models. The left figure is ARIMA(2, 0, 2), the right figure is SARIMA

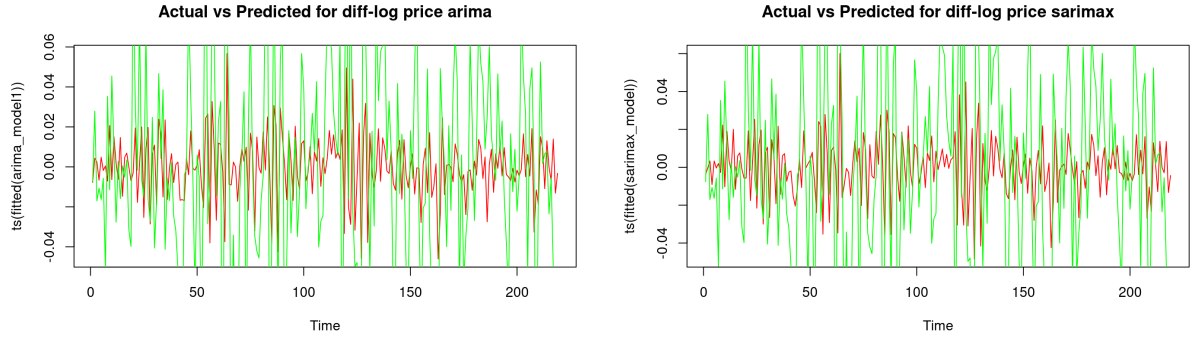

Figure 6: Actual v.s. Predicted diff-log prices for arima(2, 0, 2), sarima models. The left figure is ARIMA(2, 0, 2), the right figure is SARIMA

were plotted alongside the actual cocoa prices to visualize model performance. Although both the ARIMA Model 1 and the SARIMAX model produced nearly identical forecasts, the forecasts tended to overpredict the actual prices during periods of peak volatility, suggesting that none of the models adequately captured abrupt shifts during that period.

5.3 Linear Regression Model

A linear regression model was developed using both lagged values of the log-transformed cocoa prices and weather variables as predictors. We split the data iinto a 70% training set and a 30% test set and the linear model was then fitted on the training data (excluding the date) to predict the log-transformed prices.

$$\log(\text{Price}_t) = \beta_0 + \beta_1 \log(\text{Price}_{t-1}) + \beta_2 \log(\text{Price}_{t-2}) + \cdots + \beta_7 \log(\text{Price}_{t-7}) + \beta_8 \text{PRCP}_t + \beta_9 \text{TAVG}_t + \beta_{10} \text{TMAX}_t + \beta_{11} \text{TMIN}_t + \epsilon_t$$

Predictions for the test set were made and then converted back from the log scale to the original price scale. We firstly evaluate the model's performance by plotting the predicted prices alongside the actual values to visually assess the fit.
Figure above illustrates that the linear regression model was able to closely approximate the actual cocoa prices, with the blue predicted line nearly overlapping the red actual line throughout most of the test set.
We check the residuals with an ACF plot to spot any remaining patterns. We also run the Durbin-Watson test to confirm there was no significant autocorrelation, with a p-value greater than 0.05, which is 0.2871, indicating a well-fitting model.
The Durbin-Watson test returned a p-value greater than 0.05, and the ACF plot did not reveal any significant autocorrelation, suggesting that the linear regression model's residuals were adequately random.

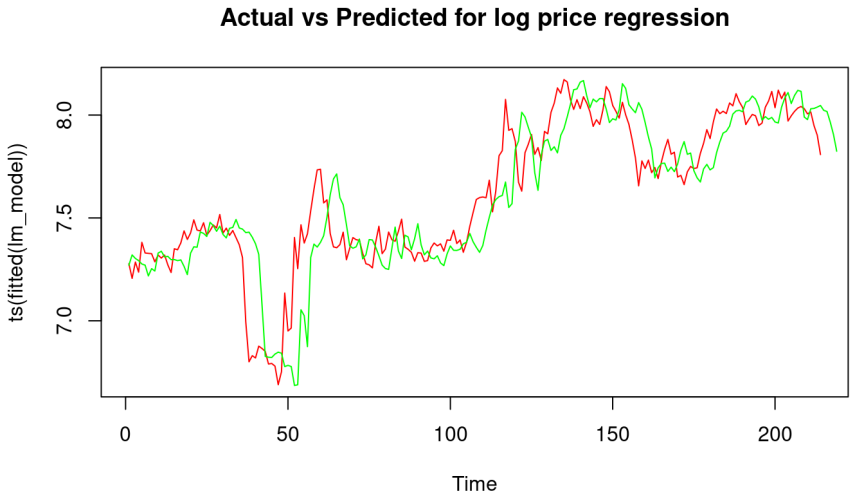

Figure 7: Actual v.s. Predicted for diff-log pirce of Linear Regression Model

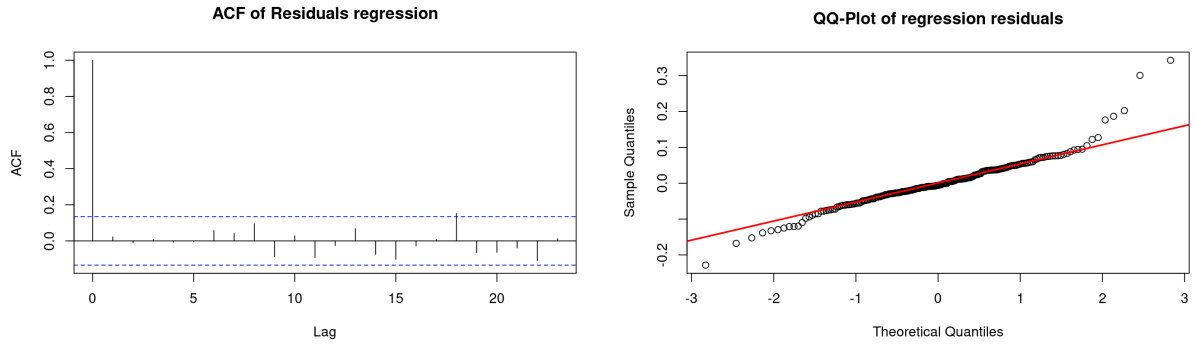

Figure 8: ACF (shown on left) and QQ plot (shown on left) for Residual Regression Diagnostics

5.4 ARMA-GARCH Model

We build the GARCH model in order to capture the volatility in cocoa price returns. Log returns were calculated by taking the difference of the log-transformed prices. The data was then split into 70% for training and 30% for testing. A GARCH(1, 1) model with an ARMA(2, 2) structure for the mean was chosen. The residuals were assumed to follow a normal distribution. We then fit the model on the returns from the training set using the *ugarchfit()* function.
To be more specific, our GARCH model is used on the log returns r_t and is specified with an ARMA(2,2) mean structure and a GARCH(1, 1) variance process.

- Mean Equation ARMA(2, 2)

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} \tag{1}$$

- Variance Equation GARCH(1,1)

$$\varepsilon_t = \sigma_t z_t \tag{2}$$
$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \tag{3}$$

where X_t is the time series at time t, ϕ_1, ϕ_2 are autoregressive (AR) coefficients, θ_1, θ_2 are moving average (MA) coefficients, $\varepsilon_t \sim$ i.i.d. $(0, \sigma^2)$ is white noise, ε_t is the error term from the mean equation (e.g., ARMA), σ_t^2 is

the conditional variance at time t, $z_t \sim$ i.i.d. $(0, 1)$ is a standard normal innovation, and $\omega > 0$, $\alpha \geq 0$, $\beta \geq 0$ are GARCH model parameters.
To evaluate the GARCH model's performance, forecasts were generated for the test period using the fitted model. The predicted returns were then cumulatively summed and exponentiated to recover the forecasted price levels. Model validation involved several steps: the forecasted prices were plotted alongside the actual prices for visual comparison, the ACF of standardized residuals was examined to check for any lingering serial correlation; the Ljung-Box test was performed, with p-values, above 0.05 suggesting no significant autocorrelation; and finally, a Q-Q plot was used to see how well the residuals followed a normal distribution.

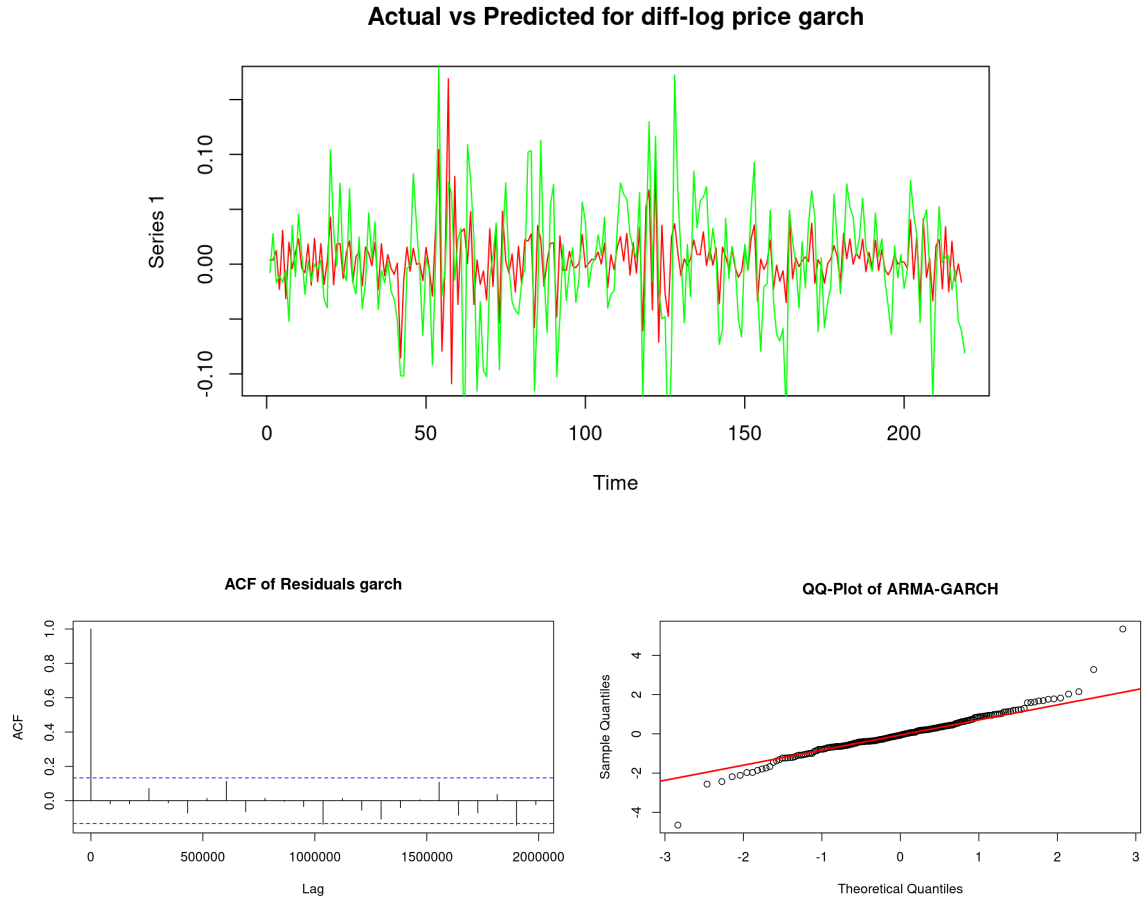

Figure 9: We've shown the plot for monthly Cocoa Price after the difference log being applied for ARMA-GARCH Model. The ACF and QQ plot are shown below.

In Figure, the GARCH forecast responded to major movements in cocoa prices, but in some instances, either overreacted or lagged behind the actual changes. This suggests that the current specification might require tuning to better adapt to market shocks. The standardized residuals of the GARCH model passed the Ljung-Box test (p-value $0.4187 > 0.05$) and closely followed the theoretical quantiles in the Q-Q plot, indicating that the model adequately captured the volatility clustering in the cocoa returns.

5.5 XGBoost

Our XGBoost regression model was fit at each iteration using 600 boosting rounds and a learning rate (eta) of 0.05. The objective function used was *reg:squarederror* to minimize the mean squared error between actual and predicted log-prices. Predictions were transformed back from log-scale to the original price scale for interpretability. The objective function optimized by XGBoost is given by:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} \ell\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{k=1}^{t} \Omega(f_k)$$

Where, $\ell\left(y_i, \hat{y}_i^{(t)}\right) = \left(y_i - \hat{y}_i^{(t)}\right)^2$ is the squared error loss, $\Omega(f_k)$ is the regularization term that penalizes tree complexity.

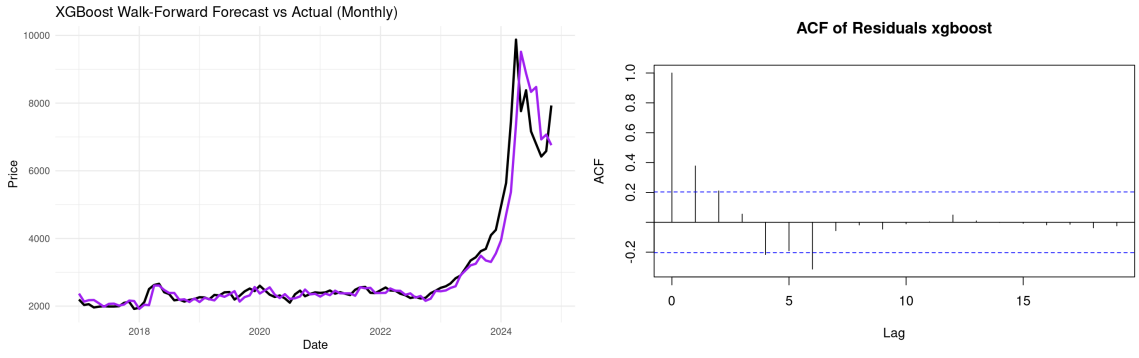

Figure 10: We've shown the plot for monthly Cocoa Price after the difference log being applied for XGBoost Model. The Actual v.s. Predicted and ACF plot are shown below.

The predicted values tracked the actual price movements closely, particularly during stable market periods. Peaks and troughs were well captured, demonstrating the model's ability to anticipate short-term fluctuations. Besides, the ACF plots shows that it decreases quickly through the lag and it lies within the cut-off value. However, the result of Box-Ljung test suggest that the $p-value = 0.009691 < 0.05$, which we still need to have some further comparison among the models.

5.6 Overall Observations and Patterns

After fitting the model, we generate a table of the forecast accuracy metrics for all 5 models, i.e., ETS, ARIMA/SARIMA, Linear Regression, ARMA-GARCH, and XGBoost models.

Model	ME	RMSE	MAE	MPE	MAPE
ETS	494.749	1711.354	810.2131	3.134551	17.81288
ARIMA(2, 0, 2)	419.3616	1672.223	826.5914	0.5162371	19.03198
SARIMA	532.6924	1751.856	820.2823	4.292218	17.71888
Linear Regression	60.22585	488.6511	229.154	0.903759	5.392149
ARMA-GARCH(2, 2)	11.98334	1445.934	905.857	-13.00349	25.9833
XGBoost	50.19557	512.1491	253.3477	0.9982531	6.132129

Table 2: Forecast Accuracy Metrics Data from test set for ETS, ARIMA, SARIMA Models

Among all the models, we've seen ETS, ARIMA, SARIMA, and ARMA-GARCH show a significant difference on all the accuracy data comparing to Linear Regression and XGBoost Model. As discussed in the methodology, Linear Regression and XGBoost are the outstanding models, as they have low ME, RMSE, MAE, and reasonable MPE and MAPE values. The figure displays a 10 years time series forecast comparison using various models,

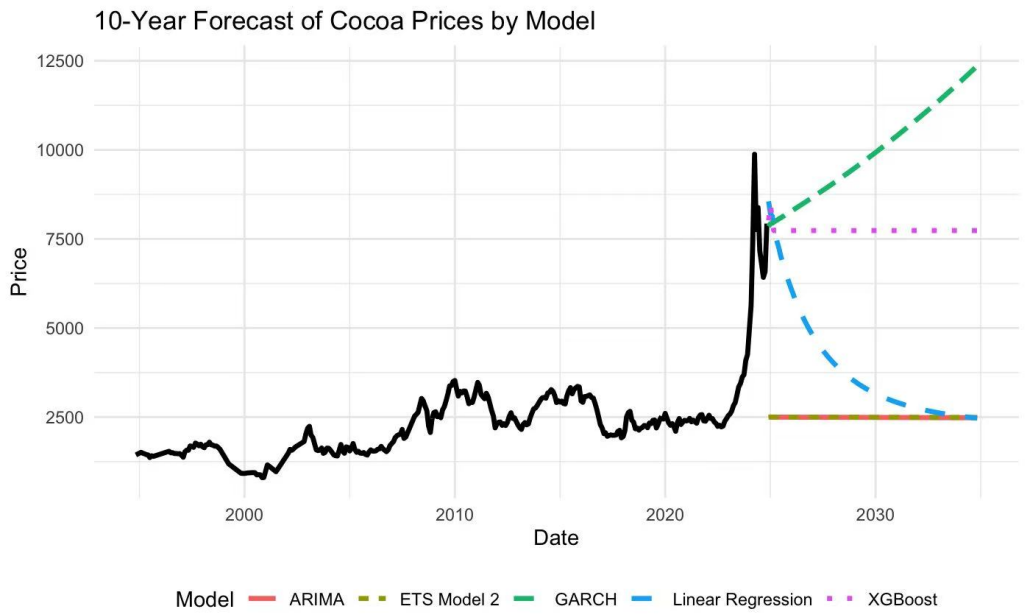

Figure 11: Actual v.s. Forecast for all of the models

including Naive, ETS, ARIMA, SARIMA, GARCH, Linear Regression, and XGBoost. We've seen the ARIMA, ETS has a significant gap at the start and follows a straight line for the forecasting value. This is due to non-promising pattern at the Actual v.s. Predicted figure which we talked before for these two models. We've seen a increasing trend for ARMA-GARCH Model and a decreasing trend for linear regression model. XGBoost shows a horizontal vibrating line following the end point of actual price.
Overall, it's clear we are deciding between Linear Regression Model or XGBoost Model to choose as from the forecast accuracy metric data. However, as stated in section 5.5, the result of Box-Ljung test for XGBoost Model suggests that the $p-value = 0.009691 < 0.05$, which mean the residual of XGBoost has autocorrelation. Whereas for Linear Regression Model, besides all the data in accuracy metric showing great results, the residual diagnostic also produce great results. Therefore, we'll choose linear regression model as our final model.

6 Discussion and Conclusion

After comparing metrics like MAE and RMSE and graphs of actual prices versus forecasted price in last section, we chose the linear regression model as the final forecasting model due to its optimal performance. Specifically, this model has lowest accuracy metrics and pattern of predicted price closely followed the actual one, suggesting a better forecast.
The linear regression model forecasts a substantial long-term decline in cocoa prices from the peak observed during the 2023-2024 period. This sharp peak likely reflects short-term shocks such as global supply chain disruptions and labor shortages caused by COVID-19 pandemic, which temporarily pushed up prices. However, as the situation after the pandemic stabilizes, our model predicts that prices will return to lower levels. Lower global prices may benefit chocolate manufacturers as the input costs reduce, but long-term supply risks would emerge if cocoa farming became unprofitable. However, it may reduce cocoa farmers' incentives to invest in disease-resistant crops, resulting in lower quality of cocoa. Thus, policy responses should prioritize income protection.
A key limitation of this model is that we only consider data from 1994 to 2025 and use basic climate indicators from Ghana. The exclusion of demand trends, currency exchange rates, or other climate-related variables may reduce forecasting accuracy. In the future, we could use more advanced machine learning models or deep learning models to better capture overall pattern and long-term dependencies. Additionally, we could introduce more variables discussed in limitations to the analysis.

References

[1] Seetha Anusha, B. Kumar, and Sateesh Deevi. Time series analysis of indian spices export and prices. *Indian Journal Of Agricultural Research*, 09 2019.

[2] Eric Bomdzele and Ernest L. Molua. Assessment of the impact of climate and non-climatic parameters on cocoa production: a contextual analysis for cameroon. *Frontiers in Climate*, 5, 2023.

[3] Chris Bryant and Matthew I Mitchell. The political ecology of cocoa in ghana: Past, present and future challenges. In *Natural Resources Forum*, volume 45, pages 350–365. Wiley Online Library, 2021.

[4] Valentina Mereu Antonio Trabucco Serena Marras Donatella Spano Christine Bilen, Daniel El Chami. A systematic review on the impacts of climate change on coffee agrosystems. 2022.

[5] Carolina Deina, Matheus Henrique do Amaral Prates, Carlos Henrique Rodrigues Alves, Marcella Scoczynski Ribeiro Martins, Flavio Trojan, Sergio Luiz Stevan, and Hugo Valadares Siqueira. A methodology for coffee price forecasting based on extreme learning machines. *Information Processing in Agriculture*, 9(4):556–565, 2022.

[6] International Cocoa Organization. Monthly cocoa market review - november 2016. Technical report, International Cocoa Organization, 2016. Accessed: 2025-03-28.

[7] Ridha Rizki Novanda, Eko Sumartono, Putri Suci Asriani, Ellys Yuliarti, Ketut Sukiyono, Basuki Sigit Priyono, Irnad, Reswita, Melly Suryanty, and Vera Octalia. A comparison of various forecasting techniques for coffee prices. *Journal of Physics: Conference Series*, 1114(1):012119, nov 2018.

[8] E.Dadey Sampson Ankrah, Kwadwo A. Nyantakyi. Modeling the causal effect of world cocoa price on production of cocoa in ghana. *Universal Journal of Agricultural Research*, 2014.

[9] Mara P. Squicciarini, Anneleen Vandeplas, and Jesús Barreiro-Hurle. Living income differential in the cocoa sector: Theory and impact. Technical Report JRC125754, European Commission, Joint Research Centre, 2021. Accessed: 2025-03-28.

Appendix A: Forecast Results by Linear Regression Model

Date	Forecast	Model
2024-12-01	8555.347875	Linear Regression
2025-01-01	8213.471006	Linear Regression
2025-02-01	8139.190266	Linear Regression
2025-03-01	7917.483074	Linear Regression
2025-04-01	7597.448809	Linear Regression
2025-05-01	7401.550388	Linear Regression
2025-06-01	7218.961220	Linear Regression
2025-07-01	7012.675571	Linear Regression
2025-08-01	6839.950062	Linear Regression
2025-09-01	6680.928579	Linear Regression
2025-10-01	6517.803399	Linear Regression
2025-11-01	6365.556040	Linear Regression
2025-12-01	6222.810480	Linear Regression
2026-01-01	6083.277504	Linear Regression
2026-02-01	5950.121394	Linear Regression
2026-03-01	5823.804070	Linear Regression
2026-04-01	5702.208931	Linear Regression
2026-05-01	5585.660401	Linear Regression
2026-06-01	5474.343123	Linear Regression
2026-07-01	5367.517075	Linear Regression
2026-08-01	5264.999604	Linear Regression
2026-09-01	5166.743013	Linear Regression
2026-10-01	5072.425337	Linear Regression
2026-11-01	4981.833316	Linear Regression
2026-12-01	4894.837889	Linear Regression
2027-01-01	4811.243677	Linear Regression
2027-02-01	4730.874174	Linear Regression
2027-03-01	4653.590429	Linear Regression
2027-04-01	4579.244317	Linear Regression
2027-05-01	4507.693249	Linear Regression
2027-06-01	4438.811322	Linear Regression
2027-07-01	4372.476408	Linear Regression
2027-08-01	4308.571069	Linear Regression
2027-09-01	4246.986679	Linear Regression
2027-10-01	4187.620122	Linear Regression
2027-11-01	4130.372971	Linear Regression
2027-12-01	4075.152605	Linear Regression
2028-01-01	4021.871334	Linear Regression
2028-02-01	3970.445761	Linear Regression
2028-03-01	3920.796919	Linear Regression
2028-04-01	3872.849938	Linear Regression
2028-05-01	3826.533673	Linear Regression
2028-06-01	3781.780572	Linear Regression
2028-07-01	3738.526478	Linear Regression
2028-08-01	3696.710387	Linear Regression
2028-09-01	3656.274283	Linear Regression
2028-10-01	3617.162978	Linear Regression
2028-11-01	3579.323937	Linear Regression
2028-12-01	3542.707135	Linear Regression
2029-01-01	3507.264917	Linear Regression
2029-02-01	3472.951864	Linear Regression
2029-03-01	3439.724674	Linear Regression
2029-04-01	3407.542041	Linear Regression
2029-05-01	3376.364554	Linear Regression
2029-06-01	3346.154589	Linear Regression
2029-07-01	3316.876218	Linear Regression
2029-08-01	3288.495117	Linear Regression
2029-09-01	3260.978484	Linear Regression
2029-10-01	3234.294957	Linear Regression
2029-11-01	3208.414541	Linear Regression
2029-12-01	3183.308540	Linear Regression
2030-01-01	3158.949489	Linear Regression
2030-02-01	3135.311089	Linear Regression
2030-03-01	3112.368155	Linear Regression
2030-04-01	3090.096556	Linear Regression
2030-05-01	3068.473164	Linear Regression
2030-06-01	3047.475804	Linear Regression
2030-07-01	3027.083211	Linear Regression
2030-08-01	3007.274982	Linear Regression
2030-09-01	2988.031538	Linear Regression
2030-10-01	2969.334082	Linear Regression
2030-11-01	2951.164565	Linear Regression
2030-12-01	2933.505647	Linear Regression
2031-01-01	2916.340668	Linear Regression
2031-02-01	2899.653616	Linear Regression
2031-03-01	2883.429095	Linear Regression
2031-04-01	2867.652298	Linear Regression
2031-05-01	2852.308981	Linear Regression
2031-06-01	2837.385439	Linear Regression
2031-07-01	2822.868477	Linear Regression
2031-08-01	2808.745394	Linear Regression
2031-09-01	2795.003956	Linear Regression
2031-10-01	2781.632379	Linear Regression
2031-11-01	2768.619308	Linear Regression
2031-12-01	2755.953798	Linear Regression
2032-01-01	2743.625298	Linear Regression
2032-02-01	2731.623635	Linear Regression
2032-03-01	2719.938992	Linear Regression
2032-04-01	2708.561904	Linear Regression
2032-05-01	2697.483232	Linear Regression
2032-06-01	2686.694157	Linear Regression
2032-07-01	2676.186164	Linear Regression
2032-08-01	2665.951031	Linear Regression
2032-09-01	2655.980814	Linear Regression
2032-10-01	2646.267841	Linear Regression
2032-11-01	2636.804696	Linear Regression
...	...	...

Figure 12: Forecasted Price Values by Linear Regression Model

Appendix B: Coding Repository

For further referencing to code, the following link is the working repository of our group: Cocoa-Price-Prediction (Also see complete code in case some code is not revealed clearly.)

```r
# load required libraries
library(tidyverse)
library(lubridate)
library(forecast)
library(tseries)
library(ggplot2)
library(xgboost)
library(caret)
library(slider)
library(rugarch)
library(car)
```

Listing 1: Package

```r
# load and preprocess price data
cocoa_prices <- read.csv("Daily Prices_ICCO.csv", stringsAsFactors = FALSE)
cocoa_prices$Date <- as.Date(cocoa_prices$Date, format='%d/%m/%Y')
cocoa_prices$Price <- as.numeric(gsub(",", "", cocoa_prices$ICCO.daily.price..US..tonne.))
cocoa_prices <- cocoa_prices %>%
  mutate(YearMonth = floor_date(Date, "month")) %>%
  group_by(YearMonth) %>%
  summarise(Price = mean(Price, na.rm = TRUE)) %>%
  ungroup()

ghana_weather <- read.csv("Ghana_data.csv", stringsAsFactors = FALSE)
ghana_weather$DATE <- as.Date(ghana_weather$DATE)
ghana_weather <- ghana_weather %>%
  mutate(YearMonth = floor_date(DATE, "month")) %>%
  group_by(YearMonth) %>%
  summarise(across(c(PRCP, TAVG, TMAX, TMIN), mean, na.rm = TRUE))

# Merge and Clean Monthly Data(log + diff)
cocoa_data <- left_join(cocoa_prices, ghana_weather, by = "YearMonth") %>%
  mutate(log_price = log(Price),
         diff_log_price = c(NA, diff(log_price))) %>%
  drop_na()

# Plot Monthly Time Series
ggplot(cocoa_data, aes(x = YearMonth)) +
  geom_line(aes(y = Price), color = "steelblue") +
  labs(title = "Monthly Cocoa Prices", y = "Price", x = "Date") +
  theme_minimal()

# Split Data into Training and Testing Sets(7:3 ratio)
train_size <- floor(0.7 * nrow(cocoa_data))
train_data <- cocoa_data[1:train_size, ]
test_data <- cocoa_data[(train_size + 1):nrow(cocoa_data), ]
```

Listing 2: Data

```r
# since not station, already transformed in the pre-processed part(log + diff)
# built ets models
ets_model_1 <- ets(train_data$diff_log_price, model = "ZZZ")
ets_model_2 <- ets(train_data$diff_log_price)
plot(ts(fitted(ets_model_2)), col = "red",
     main = "Actual vs Predicted for diff-log price ets") +
```

```
  lines(ts(train_data$diff_log_price), col = "green")
# ets_model_2(ets_model_1) was selected as a candidate
```

Listing 3: ETS Model

```
# built arima, sarima models
# verify stationarity(1st diff + log)
ggplot(train_data, aes(x = YearMonth)) +
  geom_line(aes(y = diff_log_price), color = "steelblue") +
  labs(title = "Monthly Cocoa Prices", y = "Price", x = "Date") +
  theme_minimal()
acf(train_data$diff_log_price, main = "ACF of differencing log price")
pacf(train_data$diff_log_price, main = "PACF of differencing log price")
external_regressors <- data.matrix(train_data[, c("PRCP", "TAVG", "TMAX", "TMIN")])
arima_model1 <- arima(train_data$diff_log_price, order = c(2,0,2), xreg = external_regressors)
arima_model2 <- arima(train_data$diff_log_price, order = c(2,0,5), xreg = external_regressors)
sarimax_model <- auto.arima(train_data$diff_log_price, xreg = as.matrix(external_regressors), s
summary(arima_model1)
summary(arima_model2)
# arima_model1, sarimax_model was selected as a candidate, seasonality not detected(sarimaxmodel
plot(ts(fitted(arima_model1)), col = "red", main = "Actual vs Predicted for diff-log price arima
plot(ts(fitted(sarimax_model)), col = "red", main = "Actual vs Predicted for diff-log price sari

# residual diagnostic for arima, sarima models
tsdiag(arima_model1, gof.lag = 20)
tsdiag(arima_model2, gof.lag = 20)
tsdiag(sarimax_model, gof.lag = 20)

# forecasting in diff-log base
test_xreg <- data.matrix(test_data[, c("PRCP", "TAVG", "TMAX", "TMIN")])
test_xreg <- as.matrix(test_xreg)
ets_forecast_2 <- forecast(ets_model_2, h = nrow(test_data))
sarimax_forecast <- forecast(sarimax_model, xreg = test_xreg, h = nrow(test_data))
h <- nrow(test_data)
pred1 <- predict(arima_model1, n.ahead = h, newxreg = test_xreg)
```

Listing 4: ARIMA

```
# back-transform forecasted values
reconstruct_log_prices <- function(last_log_price, diffs) {cumsum(c(last_log_price, diffs))[-1]

last_log_price <- tail(train_data$log_price, 1)
n <- nrow(test_data)
forecast_dates <- test_data$YearMonth

ets2_log_forecast <- reconstruct_log_prices(last_log_price, ets_forecast_2$mean)
sarimax_log_forecast <- reconstruct_log_prices(last_log_price, sarimax_forecast$mean)
arima_log_forecast <- reconstruct_log_prices(last_log_price, pred1$pred)

ets2_price_forecast <- exp(ets2_log_forecast)
ets2_price_forecast
sarimax_price_forecast <- exp(sarimax_log_forecast)
sarimax_price_forecast
arima_price_forecast <- exp(arima_log_forecast)
arima_price_forecast

forecast_df <- bind_rows(
  tibble(Date = forecast_dates, Forecast = ets2_price_forecast, Model = "ETS Model 2"),
  tibble(Date = forecast_dates, Forecast = sarimax_price_forecast, Model = "SARIMAX"),
  tibble(Date = forecast_dates, Forecast = arima_price_forecast, Model = "ARIMA")
) %>% drop_na()
```

```

forecast_df
#forecast_df is a knitted back-transformed forecasted values for three models

ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.3)
  geom_line(data = forecast_df, aes(x = Date, y = Forecast, color = Model, linetype = Model), li
  labs(title = "Monthly Predicted of Three Models vs Actual Cocoa Prices", y = "Price", x = "Dat
  theme_minimal() +
  theme(legend.position = "bottom") +
  scale_color_manual(values = c(
    "ETS Model 2" = "green",
    "SARIMAX" = "blue",
    "ARIMA" = "red"
  )) +
  scale_linetype_manual(values = c(
    "ETS Model 2" = "solid",
    "SARIMAX" = "dotdash",
    "ARIMA" = "twodash"
  ))

# acc comparison
actual_prices <- exp(test_data$log_price)
ets2_acc <- accuracy(ets2_price_forecast, actual_prices)
sarimax_acc <- accuracy(sarimax_price_forecast, actual_prices)
arima_acc <- accuracy(arima_price_forecast, actual_prices)

print("ETS Model Performance:")
print(ets2_acc)
print("ARIMAX Model Performance:")
print(arima_acc)
print("SARIMAX Model Performance:")
print(sarimax_acc)
```

Listing 5: SARIMA

```
# built linear regression model
# create Lag Features
generate_lags <- function(data, lags = 1:6) {
  for (lag in lags) {
    data[[paste0("lag_", lag)]] <- dplyr::lag(data$log_price, lag)
  }
  return(data)
}
cocoa_data_lagged <- generate_lags(cocoa_data) %>% drop_na()
lm_data <- cocoa_data_lagged %>%
  select(YearMonth, log_price, starts_with("lag_"), PRCP, TAVG, TMAX, TMIN)
train_size <- floor(0.7 * nrow(lm_data))
train_lm <- lm_data[1:train_size, ]
test_lm <- lm_data[(train_size + 1):nrow(lm_data), ]
lm_model <- lm(log_price ~ ., data = train_lm %>% select(-YearMonth))
plot(ts(fitted(lm_model)), col = "red",
     main = "Actual vs Predicted for log price regression") +
  lines(ts(train_data$log_price), col = "green")

lm_pred_log <- predict(lm_model, newdata = test_lm)
lm_pred_price <- exp(lm_pred_log)
lm_results <- tibble(
  Date = test_lm$YearMonth,
  Actual = exp(test_lm$log_price),
  Predicted = lm_pred_price
```

```
)
### Plot Regression Results
ggplot(lm_results, aes(x = Date)) +
  geom_line(aes(y = Actual), color = "red") +
  geom_line(aes(y = Predicted), color = "blue") +
  labs(title = "Linear Regression Predicted vs Actual Prices (Monthly)", y = "Price", x = "Date
  theme_minimal()

lm_accuracy <- accuracy(lm_pred_price, exp(test_lm$log_price))
print("Linear Regression Model Performance:")
print(lm_accuracy)

acf(residuals(lm_model), main = "ACF of Residuals regression")
library(lmtest)
dwtest(lm_model) #p-value > 0.05 good -> no autocorrelation
qqnorm(residuals(lm_model), main = "QQ-Plot of regression residuals")
qqline(residuals(lm_model), col = "red", lwd = 2)
vif(lm_model)
Box.test(residuals(lm_model), lag = 20, type = "Ljung-Box")
```

Listing 6: Linear Regression

```
# Calculate log returns
log_re <- diff(log(cocoa_data$Price))
log_re <- na.omit(log_re)
train_size <- floor(0.7 * length(log_re))
train_re <- log_re[1:train_size]
test_re <- log_re[(train_size + 1):length(log_re)]
test_dates <- cocoa_data$YearMonth[(train_size + 2):(length(log_re) + 1)]
# define garch model - using our best arma model p,q with 2,2 and widely used garchorder 1,1
garch_spec <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(2, 2), include.mean = TRUE),
  distribution.model = "norm"
)
garch_fit <- ugarchfit(spec = garch_spec, data = train_re)
garch_forecast <- ugarchforecast(garch_fit, n.ahead = length(test_re))
predicted_re <- as.numeric(fitted(garch_forecast))
plot(ts(fitted(garch_fit)), col = "red",
     main = "Actual vs Predicted for diff-log price garch") +
  lines(ts(train_data$diff_log_price), col = "green")

# back-transform
last_train_price <- cocoa_data$Price[train_size + 1]
forecast_prices <- last_train_price * exp(cumsum(predicted_re))

garch_df <- tibble(
  Date = test_dates,
  Price = forecast_prices
)
# plot the model
ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black") +
  geom_line(data = garch_df, aes(x = Date, y = Price), color = "pink") +
  labs(title = "GARCH Predicted vs Actual Prices (Monthly)", y = "Price", x = "Date") +
  theme_minimal()

garch_accuracy <- accuracy(forecast_prices, exp(test_data$log_price))

print("GARCH Model Performance:")
print(garch_accuracy)
```

```r

residuals_std <- residuals(garch_fit, standardize = TRUE)
acf(residuals_std, main = "ACF of Residuals garch")
Box.test(residuals_std, lag = 20, type = "Ljung-Box")

qqnorm(residuals_std, main = "QQ-Plot of ARMA-GARCH")
qqline(residuals_std, col = "red", lwd = 2)
```

Listing 7: GARCH

```r
# built xgboost model
# generate lags
generate_lags <- function(data, lags = 1:6) {
  for (lag in lags) {
    data[[paste0("lag_", lag)]] <- dplyr::lag(data$log_price, lag)
  }
  return(data)
}

cocoa_data_lagged <- cocoa_data %>%
  generate_lags() %>%
  drop_na()

initial_size <- floor(0.7 * nrow(cocoa_data_lagged))
forecast_horizon <- nrow(cocoa_data_lagged) - initial_size


walk_results <- map_dfr(1:forecast_horizon, function(i) {
  train_set <- cocoa_data_lagged[1:(initial_size + i - 1), ]
  test_point <- cocoa_data_lagged[(initial_size + i), ]

  x_train <- train_set %>% select(starts_with("lag_"), PRCP, TAVG, TMAX, TMIN)
  y_train <- train_set$log_price
  x_test <- test_point %>% select(starts_with("lag_"), PRCP, TAVG, TMAX, TMIN)

  dtrain <- xgb.DMatrix(as.matrix(x_train), label = y_train)
  dtest <- xgb.DMatrix(as.matrix(x_test))
# fit model
  model <- xgboost(
    data = dtrain,
    nrounds = 600,
    objective = "reg:squarederror",
    verbose = 0,
    eta = 0.05
  )

  pred_log <- predict(model, dtest)
  tibble(
    Date = test_point$YearMonth,
    Actual = exp(test_point$log_price),
    Predicted = exp(pred_log)
  )
})


ggplot(walk_results, aes(x = Date)) +
  geom_line(aes(y = Actual), color = "black", linewidth = 1) +
  geom_line(aes(y = Predicted), color = "purple", linewidth = 1) +
  labs(title = "XGBoost Walk-Forward Forecast vs Actual (Monthly)",
       y = "Price", x = "Date") +
  theme_minimal()
```

```r

xgb_accuracy <- accuracy(walk_results$Predicted, walk_results$Actual)
print("XGBoost Walk-Forward Accuracy Metrics:")
print(xgb_accuracy)

# residual analysis of xgboost model
walk_results <- walk_results %>%
  filter(!is.na(Actual), !is.na(Predicted)) %>%
  mutate(Residual = Actual - Predicted)
acf(walk_results$Residual, main = "ACF of Residuals xgboost")
Box.test(walk_results$Residual, lag = 20, type = "Ljung-Box")

x_train_full <- cocoa_data_lagged %>% select(starts_with("lag_"), PRCP, TAVG, TMAX, TMIN)
y_train_full <- cocoa_data_lagged$log_price
dtrain_full <- xgb.DMatrix(as.matrix(x_train_full), label = y_train_full)

final_model <- xgboost(
  data = dtrain_full,
  nrounds = 600,
  objective = "reg:squarederror",
  eta = 0.05,
  verbose = 0
)
```

Listing 8: XGBoost

```r
h_future <- 120
ets_forecast_future <- forecast(ets_model_2, h = h_future)

last_log_price <- tail(train_data$log_price, 1)
ets_log_future <- reconstruct_log_prices(last_log_price, ets_forecast_future$mean)

# backtransform
ets_price_future <- exp(ets_log_future)

future_dates <- seq.Date(from = max(cocoa_data$YearMonth) + months(1),
                         by = "month", length.out = h_future)

ets_forecast_df_10yr <- tibble(
  Date = future_dates,
  Forecast = ets_price_future,
  Model = "ETS Model 2"
)

ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = ets_forecast_df_10yr, aes(x = Date, y = Forecast, color = Model, linetype = M
  labs(title = "10-Year Forecast of Cocoa Prices by ETS Model",
       x = "Date", y = "Price") +
  theme_minimal() +
  theme(legend.position = "bottom")

# calculate historical averages from training data
avg_PRCP  <- mean(train_data$PRCP, na.rm = TRUE)
avg_TAVG  <- mean(train_data$TAVG, na.rm = TRUE)
avg_TMAX  <- mean(train_data$TMAX, na.rm = TRUE)
avg_TMIN  <- mean(train_data$TMIN, na.rm = TRUE)

future_xreg <- matrix(rep(c(avg_PRCP, avg_TAVG, avg_TMAX, avg_TMIN),
                          each = h_future), nrow = h_future)

```

```r
arima_future <- predict(arima_model1, n.ahead = h_future, newxreg = future_xreg)

# back-transform
arima_log_future <- reconstruct_log_prices(last_log_price, arima_future$pred)
arima_price_future <- exp(arima_log_future)

arima_forecast_df_10yr <- tibble(
  Date = future_dates,
  Forecast = arima_price_future,
  Model = "ARIMA"
)

ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = arima_forecast_df_10yr, aes(x = Date, y = Forecast, color = Model, linetype =
  labs(title = "10-Year Forecast of Cocoa Prices by ARIMA Model",
       x = "Date", y = "Price") +
  theme_minimal() +
  theme(legend.position = "bottom")

future_lm_preds <- numeric(h_future)

current_lags <- tail(lm_data$log_price, 6)

for(i in 1:h_future) {
  newdata <- as.data.frame(t(c(current_lags, avg_PRCP, avg_TAVG, avg_TMAX, avg_TMIN)))
  colnames(newdata) <- c(paste0("lag_", 1:6), "PRCP", "TAVG", "TMAX", "TMIN")
  pred_log <- predict(lm_model, newdata = newdata)
  future_lm_preds[i] <- pred_log

  current_lags <- c(pred_log, current_lags[1:5])
}

lm_price_future <- exp(future_lm_preds)

lm_forecast_df_10yr <- tibble(
  Date = future_dates,
  Forecast = lm_price_future,
  Model = "Linear Regression"
)

ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = lm_forecast_df_10yr, aes(x = Date, y = Forecast, color = Model, linetype = Mo
  labs(title = "10-Year Forecast of Cocoa Prices by Linear Regression Model",
       x = "Date", y = "Price") +
  theme_minimal() +
  theme(legend.position = "bottom")

write.csv(lm_forecast_df_10yr, "output_file.csv", row.names = FALSE)

garch_forecast_10yr <- ugarchforecast(garch_fit, n.ahead = 120)
predicted_10yr_returns <- as.numeric(fitted(garch_forecast_10yr))

last_price <- tail(cocoa_data$Price, 1)
future_garch_prices <- last_price * exp(cumsum(predicted_10yr_returns))

garch_forecast_df_10yr <- tibble(
  Date = future_dates,
  Forecast = future_garch_prices,
  Model = "GARCH"
```

```r
)

ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = garch_forecast_df_10yr, aes(x = Date, y = Forecast, color = Model, linetype =
  labs(title = "10-Year Forecast of Cocoa Prices by ARMA-GARCH Model",
       x = "Date", y = "Price") +
  theme_minimal() +
  theme(legend.position = "bottom")

h_future <- 120
future_xgb_preds <- numeric(h_future)

# Get last 6 log prices (for lag_1 to lag_6)
current_lags <- as.numeric(tail(cocoa_data_lagged$log_price, 6))

# Average weather values (for fixed future input)
avg_PRCP <- mean(cocoa_data_lagged$PRCP, na.rm = TRUE)
avg_TAVG <- mean(cocoa_data_lagged$TAVG, na.rm = TRUE)
avg_TMAX <- mean(cocoa_data_lagged$TMAX, na.rm = TRUE)
avg_TMIN <- mean(cocoa_data_lagged$TMIN, na.rm = TRUE)

# Forecast loop
for (i in 1:h_future) {
  # Construct input as data frame
  newdata <- as.data.frame(t(c(current_lags, avg_PRCP, avg_TAVG, avg_TMAX, avg_TMIN)))
  colnames(newdata) <- c(paste0("lag_", 1:6), "PRCP", "TAVG", "TMAX", "TMIN")

  # Convert to matrix, then to DMatrix
  newdata_matrix <- as.matrix(newdata)
  dnew <- xgb.DMatrix(newdata_matrix)

  # Predict log price
  pred_log <- predict(final_model, dnew)

  # Save prediction
  future_xgb_preds[i] <- pred_log

  # Update lags
  current_lags <- c(pred_log, current_lags[1:5])
}

# Convert log-price to price
xgb_price_future <- exp(future_xgb_preds)

# Generate future dates
library(lubridate)
last_date <- max(cocoa_data_lagged$YearMonth)
future_dates <- seq.Date(from = last_date %m+% months(1), by = "month", length.out = h_future)

# Final forecast data frame
xgb_forecast_df_10yr <- tibble(
  Date = future_dates,
  Forecast = xgb_price_future,
  Model = "XGBoost"
)

# Plot
ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = xgb_forecast_df_10yr, aes(x = Date, y = Forecast, color = Model), linewidth =
```

```
  labs(title = "10-Year Forecast of Cocoa Prices (XGBoost)",
       x = "Date", y = "Price") +
  theme_minimal()
```
```{r}
# Combine all forecast data frames
all_forecasts_10yr <- bind_rows(
  ets_forecast_df_10yr,
  arima_forecast_df_10yr,
  lm_forecast_df_10yr,
  garch_forecast_df_10yr,
  xgb_forecast_df_10yr
)

# Plot the forecasts alongside historical prices
ggplot() +
  geom_line(data = cocoa_data, aes(x = YearMonth, y = Price), color = "black", linewidth = 1.2)
  geom_line(data = all_forecasts_10yr, aes(x = Date, y = Forecast, color = Model, linetype = Mod
  labs(title = "10-Year Forecast of Cocoa Prices by Model",
       x = "Date", y = "Price") +
  theme_minimal() +
  theme(legend.position = "bottom")
```

Listing 9: Forecast