Climate-Informed Cocoa Price Forecasting: A Comparative Analysis of Time Series and Machine Learning Methods

Yuqi An Tommy Fu Jumbo Jiang Steven Li

April 4, 2025

This study examines the forecasting of cocoa futures prices by integrating climate data from Ghana with various time series and machine learning approaches to aid stakeholders' decision-making. We analyze daily price data from 1994-2025 alongside climate variables using multiple modeling techniques, including Linear Regression, ARIMA variants, and machine learning methods with cross-validated 90-day forecasting horizons. Our findings reveal that simpler Linear Regression models consistently outperformed more complex approaches across all metrics, achieving approximately 50% lower error rates (RMSE = 0.0534) than the worst-performing models after appropriate logarithmic transformation and differencing. These results suggest that preprocessing techniques and model parsimony may be more crucial for accurate cocoa price forecasting than algorithmic complexity, offering practical implications for commodity traders and policymakers seeking reliable forecasting tools amid increasing market volatility.

Table of contents

1 Introduction

Cocoa is a fundamental component of the global chocolate industry, shaping both national farming incomes and international commodity markets. Recent research has stressed the significant challenges relating to the prediction of cocoa prices; prices are impacted by inherent seasonal fluctuations, general economic conditions, and, more recently, climate change risks (Zhang et al. 2020). Environmental factors, such as fluctuations in temperature and precipitation patterns, have been shown to impact cocoa yields, further affecting futures markets and creating high levels of uncertainty for producers, traders, and policymakers (Sukiyono et al. 2018).

Against this background, the present research considers cocoa price fluctuation forecasting and assessment by merging climatic information that involves variations in mean and extreme temperatures and rainfall severity with mainstream time series and machine learning methods. A critical component of the present work is the necessity to provide precise short and medium range forecasts that can serve as decision-support tools, from farming planting dates to policy-setting by governments and business buying decisions (Kumar et al. 2022). Precise forecasting against heightened marketplace volatility and climatic uncertainty reduces monetary risk as well as facilitates long-term agricultural and economic planning by economies that depend on cocoa output.

Therefore, our current study seeks to:

1. Examine major external determinants — namely, temperature, rainfall, and other climatological conditions influencing the variation in cocoa prices.
2. Compare the forecasting performance of different predictive approaches, including classical time series models (e.g., SARIMA and linear regression with autocorrelated errors) and machine learning methods.
3. Assess the model's robustness in real-world data settings with a focus on how climatic information can aid in enhancing or refining predictive accuracy.

Despite its practical importance, forecasting cocoa prices presents notable challenges. The problem of data non-stationarity requires the use of transformations or differencing methods in order to satisfy critical model conditions, while the existence of missing values in daily climate data prevents the proper integration of data and reduces the sample average. In addition, the need to allow for short-run autocorrelation and occasional disturbances makes the model more complicated. By addressing these problems and increasingly adding exogenous variables, this study makes cocoa price behavior more understandable, hence aiding the industry and policymakers in the formation of more effective and well-informed policies.

2 Literature Review

Time series forecasting and commodity price modeling have evolved significantly over recent decades, with researchers exploring various methodologies to improve forecasting accuracy. This section summarizes three published research on time series forecasting methods applied to agricultural commodities, with particular focus on cocoa price modeling.

2.1 Time Series Forecasting for Commodity Prices

Agricultural commodity price forecasting has evolved from traditional statistical methods to sophisticated approaches with artificial intelligence capabilities. Zhang et al. (2020) note that while ARIMA, exponential smoothing, and multivariate regression remain common, increasing price volatility has prompted development of self-learning models like neural networks, support vector regression, and extreme learning machines that often outperform traditional approaches. They emphasize the "no free lunch" theory, that no single model suits all commodities—and propose a selection framework incorporating time series features and forecast horizons based on 29 features of periodicity, nonlinearity, and complexity, which outperforms both individual models and simple averaging techniques.

2.2 Cocoa Price Forecasting Studies

Research specific to cocoa price forecasting has demonstrated the effectiveness of time series approaches. Sukiyono et al. (2018) analyzed monthly cocoa price data (2008-2016) using ARIMA, Exponential Smoothing, and Decomposition models, evaluating them with MAD, MAPE, and MSD metrics. Their findings showed ARIMA as most appropriate for both world and domestic markets, consistent with earlier studies on agricultural commodities. Kumar et al. (2022) expanded this work by developing econometric models for different cocoa bean types, using ARIMA and VAR to analyze monthly prices (2009-2020). After stationarity testing and model selection via AIC and SBIC, they identified ARIMA(1,1,0) for dry cocoa beans and ARIMA(1,1,2) for wet beans. Their VAR(1) model revealed significant influence of US and London futures prices on Indian dry cocoa prices, establishing connections between international futures and domestic markets.

Study	Time Period	Methodology	Key Findings
Zhang et al. (2020)	Various Datasets	Model selection framework with RF, SVM, ANN, SVR, and ELM as candidate models	Model selection framework outperforms both individual models and simple averaging; forecast horizon significantly impacts model performance
Kumar et al. (2022)	2008-2016	ARIMA, Exponential Smoothing	ARIMA is the most appropriate method for both world and domestic cocoa prices
Sukiyono et al. (2018)	2009-2020	ARIMA, VAR	ARIMA(1,1,0) is optimal for dry cocoa beans; ARIMA(1,1,2) for wet cocoa beans; ICE futures prices influence domestic dry cocoa prices

2.3 Gaps in Existing Literature

The literature shows a clear progression from classical time series methods to hybrid approaches combining traditional models with machine learning techniques. Zhang et al. (2020) demonstrates how classical forecasting methods can be enhanced through machine learning for model selection, while Kumar et al. (2022) bridges classical and modern approaches by combining ARIMA with VAR analysis to establish market linkages.

Despite these advancements, several gaps remain:

1. Zhang et al. (2020) highlights that forecast models perform differently across horizons, yet this factor is seldom considered in studies.

2. Kumar et al. (2022) notes limited research on forecasting models for specific cocoa varieties and a lack of commodity-specific futures in Indian exchanges.

3. Sukiyono et al. (2018) identify a need for more comprehensive comparative studies evaluating multiple forecasting methodologies for cocoa prices.

2.4 Building Upon Previous Methods

Our research extends previous studies by incorporating multiple forecast horizons as recommended by Zhang et al. (2020), evaluating both classical time series and machine learning approaches to assess whether sophisticated techniques improve cocoa price forecasting accuracy (Sukiyono et al. 2018).

We apply model selection frameworks to agricultural commodity data, addressing a gap identified by Zhang et al. (2020), while incorporating external climate data from Ghana to account for Kumar et al. (2022)'s finding that external factors significantly influence cocoa prices.

Building on Sukiyono et al. (2018)'s evaluation approach, we employ multiple performance metrics such as Root Mean Squared Error (RMSE), Mean Average Error (MAE), and Mean Average Percentage Error (MAPE) for comprehensive accuracy assessment across model types. This integrated approach aims to develop a forecasting framework specifically tailored to global cocoa price dynamics.

3 Methodology

This section outlines our approach to modeling and forecasting cocoa futures prices, detailing the selection rationale, data preparation techniques, model specifications, and validation methodology.

3.1 Model Selection Rationale

Building on findings from Zhang et al. (2020) that no single model is universally optimal for commodity price forecasting, we implemented a diverse modeling framework incorporating:

1. **Linear Regression with Exogenous Variables**: As a baseline approach that captures linear relationships between cocoa prices and various predictors, including lagged price values and climate variables.
2. **Machine Learning Models** (XGBoost, Random Forest, SVR): To capture potential nonlinear relationships in the data that traditional time series models might miss. These models align with Zhang et al. (2020) observation that advanced self-learning algorithms often outperform traditional statistical methods when modeling complex commodity price behaviors.
3. **ARIMAX and SARIMAX**: Extending the traditional ARIMA models (identified by Sukiyono et al. (2018) as particularly effective for cocoa price forecasting) to incorporate exogenous climate variables and seasonal components.

This multi-model approach allows us to assess which methodology best captures the complex dynamics of cocoa prices while addressing the "no free lunch" principle highlighted by Zhang et al. (2020).

3.2 Preprocessing Steps

Our preprocessing workflow includes essential data preparation for robust time series modeling. We apply logarithmic transformation to cocoa prices to stabilize variance and normalize distributions, followed by differencing of log-transformed prices for ARIMAX and SARIMAX models to address non-stationarity confirmed by augmented Dickey-Fuller tests. For feature engineering, we create 7-day lagged price features to capture short-term autocorrelation patterns and incorporate lagged climate variables to account for delayed effects of weather conditions on cocoa production and market expectations. These steps ensure our models receive appropriately structured inputs while satisfying time series analysis requirements.

3.3 Model Specifications

Our modeling approach employs multiple techniques with consistent configuration: Linear Regression using climate variables and 7-day lagged prices to predict log-transformed prices; Machine Learning models including XGBoost (gradient-boosted trees with moderate complexity), Random Forest (ensemble with default parameters), and SVR (radial basis kernel) all using identical features; and time series models including ARIMAX with exogenous climate predictors targeting log-transformed or differenced prices as needed, and SARIMAX configured to capture annual seasonal patterns with the same exogenous variables. This framework enables comparative evaluation across modeling paradigms while maintaining consistent feature inputs.

3.4 Training and Validation

Addressing Kumar et al. (2022) methodology for robust model evaluation, we will implement:

1. **Rolling Window Cross-Validation**:
    - Training window: 360 days of historical data
    - Forecasting horizon: 90 days forward
    - Incremental window movement: 90 days
    - This approach simulates real-world forecasting scenarios where only past information is available for model training
2. **Performance Metrics**:

- **Root Mean Square Error (RMSE)**: Emphasizes larger prediction errors
- **Mean Absolute Error (MAE)**: Measures average absolute deviation
- **Mean Absolute Percentage Error (MAPE)**: Provides relative error measurement
- This comprehensive set of metrics aligns with Sukiyono et al. (2018) recommendation for multi-faceted model evaluation

3.5 Model Refinement

Following the initial evaluation of all models, we will go through a refinement phase aimed to enhance predictive performance. Our tuning methodology focuses on exploring a targeted range of hyperparameters for the selected model types. For traditional time series models, we will investigate various ARIMA orders, limiting components to low values to prevent overfitting. For machine learning approaches, we examined the impact of parameters controlling model complexity, learning dynamics, and feature utilization.

Throughout the refinement process, we will maintain rolling window validation framework to ensure consistent evaluation across parameter combinations. Performance metrics will be aggregated across all forecast windows, providing a robust assessment of each configuration's forecasting capability under various market conditions. This methodical approach to model refinement ensures that our final selected models balance complexity with predictive accuracy while addressing the specific characteristics of cocoa price data.

4 Data

4.1 Overview

The analysis utilizes two primary datasets: daily cocoa futures prices from International Cocoa Organization (2025) and climate data from Ghana provided by National Centers for Environmental Information (2025).

Our analysis utilizes two complementary datasets: primary daily cocoa futures prices from the International Cocoa Organization (ICCO) spanning March 1994 to February 2025, providing comprehensive historical price movements across multiple market cycles; and secondary climate data from Ghana including daily precipitation and temperature metrics (average, maximum, minimum) from multiple weather stations. After cleaning to remove invalid entries and handle missing values, these datasets offer crucial market and environmental context, considering cocoa cultivation's high sensitivity to temperature and rainfall conditions that directly impact yield, quality, and ultimately market prices.

4.2 Data Cleaning

Our data preparation process involved several steps to ensure data quality and compatibility for time series modelling:

1. **Date Standardization**: We converted date fields in both datasets to ensure consistent data type and arranged records chronologically.
2. **Data Validation**: We filtered out invalid records, including rows with missing dates or non-positive prices from the cocoa price dataset.
3. **Climate Data Aggregation**: For dates with multiple weather station readings, we calculated the average values for precipitation and temperature variables to create a single daily climate profile.
4. **Dataset Integration**: We performed an inner join between the cocoa prices and Ghana's climate data using dates as the key. This approach ensured that our analysis only included days where both price and climate data were available.
5. **Missing Value Treatment**: For occasional gaps in climate variables, we applied forward and backward filling using `tidyr::fill(..., .direction = "downup")` to maintain continuity in the time series.

A detailed implementation of these steps can be found in Appendix Section B.

4.3 Log Transformations and Differencing

As shown in Figure 1, the raw cocoa price data exhibits substantial volatility and a strong upward trend, particularly in recent years. This non-stationary behavior is typical of commodity price series and requires transformation before applying time series modeling techniques.

To address these characteristics, we implemented two key transformations:

1. **Logarithmic transformation**: We converted the raw prices to natural logarithm scale which helps normalize the variance across the entire series. As evident in Figure 2, this transformation preserves the overall pattern while reducing the disproportionate impact of recent price surges, creating a more consistent variance structure throughout the time series.
2. **First differencing**: To remove the persistent trend component and achieve stationarity, we calculated the first-order differences of the log-transformed prices. Figure 3 demonstrates how this transformation effectively stabilizes the series around a zero mean with consistent variance across time, except for several volatility clusters that correspond to significant market events.

Figure 1: Daily cocoa futures prices (USD/tonne) from 1994 to 2025

Figure 2: Logarithmically transformed cocoa prices

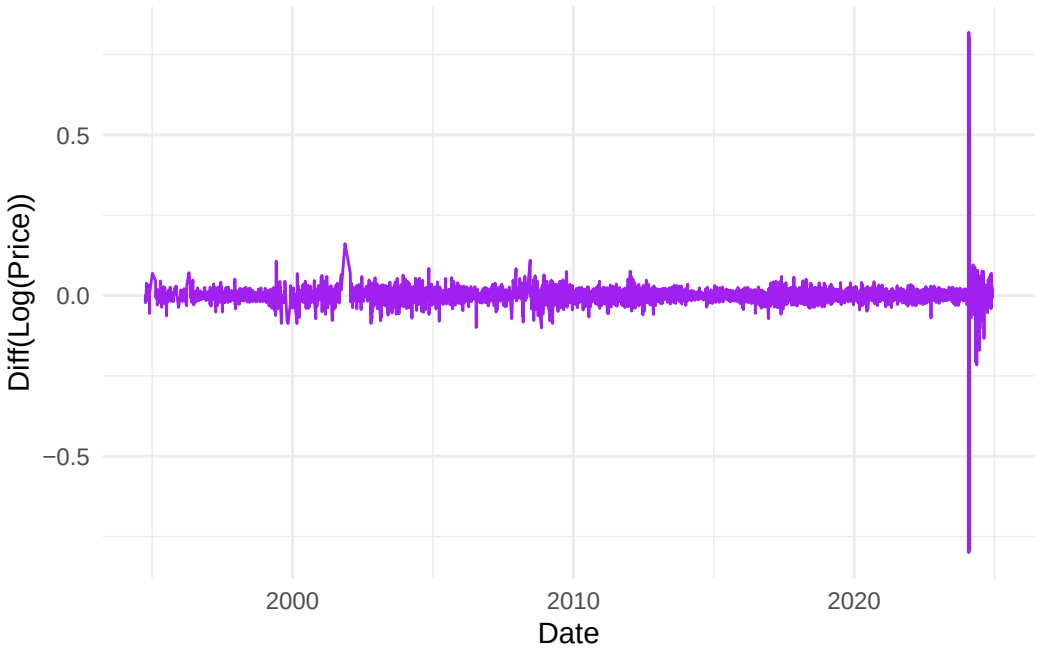

Figure 3: First differences of logarithmically transformed cocoa prices

These transformations are critical preprocessing steps that address key assumptions of time series modeling, namely stationarity and homoscedasticity. The differenced log series offers a more suitable foundation for our forecasting models, as confirmed by subsequent stationarity tests.

4.4 Stationarity and Exploratory Checks

```
ADF Test on log_price:


    Augmented Dickey-Fuller Test

data:  cocoa_data$log_price
Dickey-Fuller = -1.5883, Lag order = 18, p-value = 0.7527
alternative hypothesis: stationary


ADF Test on 1st diff_log_price:
```

```
    Augmented Dickey-Fuller Test

data:  cocoa_data$diff_log_price
Dickey-Fuller = -19.822, Lag order = 18, p-value = 0.01
alternative hypothesis: stationary
```

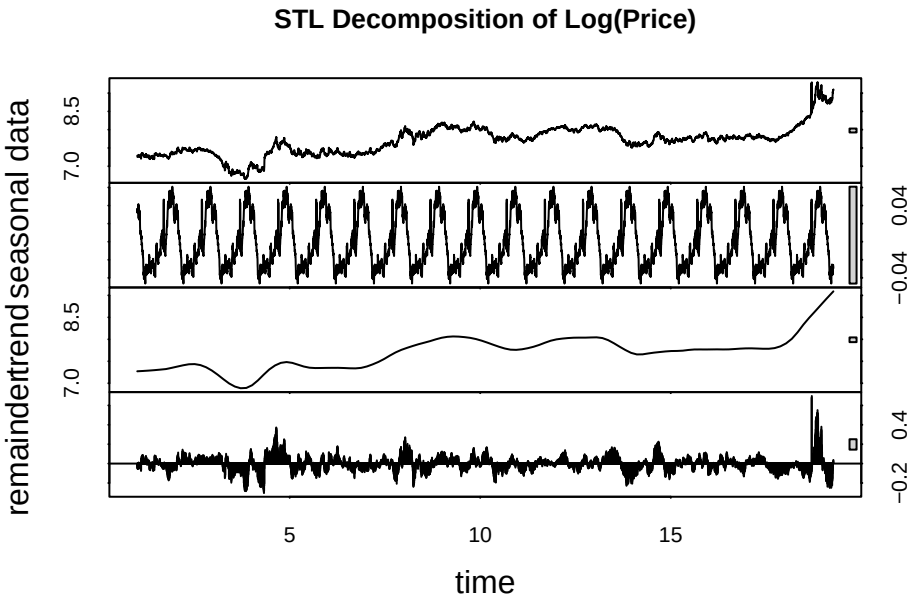

Figure 4: STL decomposition of log-transformed cocoa prices showing the original series (top), seasonal patterns (second panel), underlying trend (third panel), and residual variations (bottom panel)

To assess the statistical properties of our cocoa price series, we performed diagnostics focusing on component decomposition and stationarity testing:

1. **STL Decomposition**: Figure 4 displays the Seasonal-Trend decomposition using LOESS for the log-transformed price series. This visualization clearly separates the time series into four components:

   - The original log-price data (top panel)
   - A distinct seasonal pattern with consistent annual cycles (second panel)
   - The underlying long-term trend showing major market shifts (third panel)
   - The residual component capturing market noise and irregular events (bottom panel)

   The decomposition reveals a pronounced seasonal pattern in cocoa prices that repeats annually, likely reflecting harvest cycles in major producing regions. The trend component closely mirrors our earlier observations of price evolution, while the residual component appears largely stationary with occasional volatility clusters.

2. **Augmented Dickey-Fuller Tests**: We conducted formal stationarity tests to validate our transformation strategy:

    - The log-transformed price series yielded a test statistic of -1.5883 with a p-value of 0.7527, failing to reject the null hypothesis of non-stationarity
    - In contrast, the first-differenced log prices produced a test statistic of -19.822 with a p-value of 0.01, strongly rejecting the null hypothesis

    These results provide statistical confirmation that while the original log prices exhibit unit root behavior characteristic of financial time series, the first-differenced series achieves the stationarity required for valid time series modeling.

The combined evidence from both visual decomposition and formal hypothesis testing provides a strong foundation for our modeling approach, confirming that differencing was an appropriate and necessary transformation for this dataset.

4.5 Feature Engineering

To enhance our models' predictive power, we created additional features:

1. **Lagged Price Variables**: We incorporated 7-day lagged values of log-transformed prices to capture autocorrelation patterns.
2. **Lagged Climate Variables**: Similarly, we created 7-day lags for precipitation and temperature variables to account for delayed effects of weather conditions on market expectations and cocoa production.

These engineered features provide our models with temporal context and potential predictors that reflect the complex relationships between climate factors, historical prices, and future price movements.

Implementation of the log-transformation, differencing, stationarity check, and feature engineering steps can be found in Appendix Section C.

5 Results

This section presents the findings from our model implementation and refinement processes, focusing on performance metrics across multiple forecasting approaches. The full model implementation scripts can be found in Appendix Section D.

5.1 Model Implementation

Following our methodology outlined in Section 3, we implemented six distinct modeling approaches for predicting cocoa futures prices: Linear Regression, XGBoost, Random Forest, SVR, ARIMAX, and SARIMAX. Table 2 presents the cross-validated performance metrics for these baseline models.

Table 2: Baseline Model Performance Metrics

Model	Avg_RMSE	Avg_MAE	Avg_MAPE
LinearRegression	0.0534483	0.0425047	0.5503867
SARIMAX	0.0825830	0.0688042	0.8847440
XGBoost	0.0852723	0.0703289	0.9070454
ARIMAX	0.1003774	0.0837200	1.0833020
SVR	0.1085447	0.0912827	1.1745636
RandomForest	0.1166918	0.1027238	1.3238287

Linear Regression exhibited the lowest error rates across all metrics (RMSE = 0.0534, MAE = 0.0425, MAPE = 0.5504%), substantially outperforming more complex models. This represents approximately half the error rate of the worst-performing model (Random Forest). Among the more sophisticated approaches, SARIMAX demonstrated the second-best performance (RMSE = 0.0826), followed closely by XGBoost (RMSE = 0.0853). The traditional time series models and machine learning approaches showed considerably higher error rates, with RMSE values exceeding 0.10.

5.2 Selected Model Parameter Tuning

Based on the baseline performance, we selected the two most promising complex models, SARIMAX and XGBoost, for hyperparameter optimization to determine whether refined configurations could surpass the Linear Regression benchmark.

5.2.1 SARIMAX Tuning

We explored various SARIMAX orders by testing combinations of autoregressive (p), differencing (d), moving average (q), seasonal autoregressive (P), seasonal differencing (D), and seasonal moving average (Q) parameters. Table 3 in the Appendix shows the performance of these parameter configurations.

The best-performing SARIMAX configuration was SARIMAX(1,1,1,0,0,0) with RMSE = 0.0910, MAE = 0.0764, and MAPE = 0.9878%. While this represents an improvement over some baseline models, it still underperforms compared to the Linear Regression benchmark.

Notably, all tested SARIMAX configurations showed similar performance (RMSE ranging from 0.0910 to 0.0952), suggesting that model order selection had limited impact on forecast accuracy within our testing framework.

5.2.2 XGBoost Tuning

Grid search optimization of XGBoost hyperparameters identified an optimal configuration (max_depth = 3, eta = 0.10, colsample_bt = 1.0) that achieved RMSE = 0.0812, representing a 5% improvement over the baseline XGBoost model but still underperforming compared to Linear Regression. Tuning revealed that simpler configurations with shallower trees, moderate learning rates, and full feature utilization produced better forecasts than more complex structures, suggesting that more generalizable models better suited our forecasting task than configurations prone to overfitting.

5.3 Forecast Visualization and Interpretation

Visualization of model forecasts across representative time periods reveals significant differences in predictive behavior. As illustrated in Figure 5, Linear Regression successfully captures the downward price trend despite some delay and response attenuation, maintaining correct directional movement and gradually converging toward actual values. In contrast, the XGBoost model demonstrates minimal variation and fails to adapt to the evident price decline, suggesting overfitting to historical patterns. This visual comparison confirms our quantitative findings that despite its simplicity, Linear Regression better captures essential market dynamics than more complex models that struggle with emerging trends.

6 Discussion

6.1 Interpretation of Model Performance

The consistent superiority of Linear Regression across all evaluation metrics represents a noteworthy finding that warrants careful interpretation. Drawing on insights from our literature review and modeling results, we propose several explanations for this result.

6.1.1 Linear Relationships in Transformed Data

The logarithmic transformation and first differencing applied to cocoa prices may have effectively linearized the underlying relationships between predictors and target variables. As demonstrated in our data exploration (Figure 4), these transformations successfully removed non-stationary components and stabilized variance. Sukiyono et al. (2018) similarly found

that appropriate preprocessing enhanced the performance of simpler models for cocoa price forecasting.

Once these transformations were applied, the 7-day lagged price variables likely captured most of the short-term price dynamics through linear autocorrelation structures. This aligns with Kumar et al. (2022)'s finding that first-order ARIMA models (essentially linear models with differencing) performed well for cocoa bean price prediction.

A particularly notable finding is that while time series data typically benefit from models specifically designed to handle autocorrelated residuals (such as SARIMAX), our preprocessing steps appears to have linearized much of the price dynamics. The combination of logarithmic transformation, differencing, and engineered lag features effectively captured the autocorrelation structure that would otherwise require more complex modeling approaches. Consequently, the simpler linear regression model, incorporating lagged prices and climate variables, not only matched but outperformed SARIMAX in both accuracy and generalization capability. This supports the principle of parsimony in statistical modeling, suggesting that under appropriate preprocessing conditions, model simplicity can yield superior results compared to algorithmic complexity. This finding aligns with Zhang et al. (2020) observation that model selection should be context-dependent rather than defaulting to more complex approaches.

6.1.2 Model Complexity and Data Constraints

The "no free lunch" theorem highlighted by Zhang et al. (2020) suggests that model performance is context-dependent. In our specific forecasting context (90-day horizons with daily data and limited external predictors) the data may not contain enough complex nonlinear patterns to justify sophisticated models. The more complex models may be attempting to capture nonlinear relationships that either:

1. Don't exist in the data (leading to worse performance through overfitting)
2. Exist but require additional external features to properly model

This interpretation is supported by the observation that tuned XGBoost models with simpler structures (shallower trees, moderate learning rates) outperformed more complex configurations.

6.1.3 Forecast Horizon Effects

The 90-day forecast horizon used in our rolling window validation framework may favor simpler models. Zhang et al. (2020) specifically noted that forecast horizon significantly impacts model selection, with different model types excelling at different time scales. It's possible that Linear Regression is particularly well-suited for medium-term (90-day) cocoa price forecasting, while more complex models might demonstrate advantages at shorter or longer horizons.

6.2 Limitations and Future Research

Despite the strong performance of our Linear Regression model, several limitations of our approach suggest directions for future research:

6.2.1 Feature Engineering Opportunities

Our current feature set primarily leverages price lags and basic climate variables from Ghana. Future studies could incorporate:

1. **Comprehensive climate indicators**: Including data from other major cocoa-producing regions such as Ivory Coast and Ecuador
2. **Macroeconomic factors**: Currency exchange rates, particularly of major cocoa-producing nations, as suggested by Kumar et al. (2022)
3. **Market sentiment indicators**: Futures market positioning data and commodity index fund flows
4. **Expanded lagging structure**: Investigating optimal lag lengths and potentially incorporating autoregressive distributed lag (ARDL) frameworks

6.2.2 Methodological Extensions

Several methodological improvements could enhance future forecasting accuracy:

1. **Ensemble approaches**: Creating weighted combinations of models rather than selecting a single "best" model, potentially addressing the limitations highlighted by Zhang et al. (2020)
2. **Alternative seasonal modeling**: Exploring Fourier terms for seasonal modeling rather than relying on seasonal ARIMA components, which may be computationally challenging with high-frequency data

6.2.3 Expanded Validation Framework

While our rolling window cross-validation approach provided robust performance estimates, future work could:

1. **Test multiple forecast horizons**: Evaluating model performance across different time spans (e.g., 30, 90, 180 days) as suggested by Zhang et al. (2020)
2. **Stress test during volatile periods**: Specifically analyzing forecast accuracy during major market disruptions or supply shocks

6.3 Practical Implications

From a business perspective, our findings offer valuable insights for participants in the cocoa market:

1. **Model parsimony**: The superior performance of Linear Regression suggests that commodity trading firms might benefit from simpler, more maintainable forecasting models rather than investing in highly complex approaches
2. **Data transformation importance**: The significant performance improvements achieved through appropriate logarithmic transformation and differencing highlight the critical role of preprocessing in agricultural commodity forecasting
3. **Feature relevance**: Our results suggest that historical price patterns and basic climate variables provide substantial predictive power, potentially allowing market participants to develop effective forecasting systems with accessible data sources

These practical implications align with the objectives outlined in our introduction, offering actionable insights for stakeholders involved in cocoa production, trading, and procurement strategies.

Appendix

A Additional Figures & Tables

Table 3: Tuned SARIMAX model performance metrics

p	d	q	P	D	Q	Avg_RMSE	Avg_MAE	Avg_MAPE
1	1	1	0	0	0	0.0909823	0.0763811	0.9877723
0	1	1	0	0	0	0.0913524	0.0767327	0.9920160
0	1	0	0	0	0	0.0950122	0.0799437	1.0337701
1	1	0	0	0	0	0.0951541	0.0803000	1.0393090

Table 4: Tuned XGBoost model performance metrics

max_depth	eta	colsample_bt	Avg_RMSE	Avg_MAE	Avg_MAPE
3	0.10	1.0	0.0811709	0.0676657	0.8700254
3	0.10	0.8	0.0828984	0.0695876	0.8932140
6	0.10	1.0	0.0838684	0.0699736	0.8996944
6	0.10	0.8	0.0855260	0.0719558	0.9229097
3	0.05	1.0	0.1027522	0.0896109	1.1464265
6	0.05	1.0	0.1043475	0.0908809	1.1629961
3	0.05	0.8	0.1052632	0.0923151	1.1800027
6	0.05	0.8	0.1068959	0.0937761	1.1984652

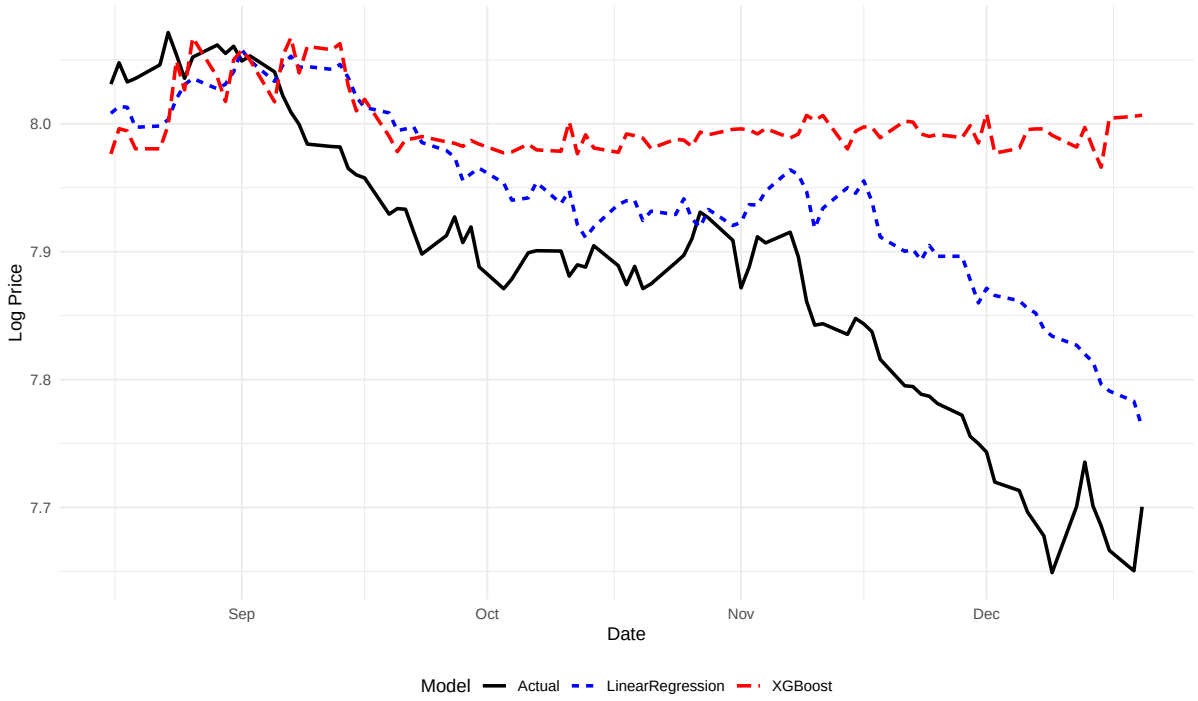

Figure 5: Comparison of actual vs. predicted log-prices for Linear Regression and XGBoost models on a representative cross-validation fold (35)

B Data Cleaning Scripts

```
 # Load libraries
library(tidyverse)
library(lubridate)
library(kableExtra)
library(dplyr)
library(knitr)
library(kableExtra)
library(forecast)
library(tseries)
library(xgboost)
library(randomForest)
library(e1071)

# Load Cocoa Price Data
cocoa_prices <- read.csv(here::here("data/raw_data/Daily Prices_ICCO.csv"), stringsAsFactors
  mutate(
    Date  = as.Date(Date, format = '%d/%m/%Y'),
    Price = as.numeric(gsub(",", "", ICCO.daily.price..US..tonne.))
  ) %>%
  select(Date, Price) %>%
  arrange(Date) %>%
  # Remove any rows with invalid dates or non-positive prices
  filter(!is.na(Date) & !is.na(Price) & Price > 0)

# Load Ghana Weather Data
ghana_weather <- read.csv(here::here("data/raw_data/Ghana_data.csv"), stringsAsFactors = FALS
  mutate(DATE = as.Date(DATE)) %>%
  # Average weather variables for each date
  group_by(DATE) %>%
  summarise(
    PRCP = mean(PRCP, na.rm = TRUE),
    TAVG = mean(TAVG, na.rm = TRUE),
    TMAX = mean(TMAX, na.rm = TRUE),
    TMIN = mean(TMIN, na.rm = TRUE)
  ) %>%
  ungroup() %>%
  arrange(DATE)

# Inner join Cocoa Prices and Weather Data
```

```r
cocoa_data <- inner_join(
  cocoa_prices,
  ghana_weather,
  by = c("Date" = "DATE")
)

# Fill missing weather values with nearest known values
cocoa_data <- cocoa_data %>%
  tidyr::fill(PRCP, TAVG, TMAX, TMIN, .direction = "downup") %>%
  drop_na()
```

C Data Preprocessing Scripts

C.1 Log Transformations and Differencing

```r
 # Log transformation, differencing
cocoa_data <- cocoa_data %>%
  mutate(
    log_price = log(Price),
    diff_log_price = c(NA, diff(log_price))
  ) %>%
  drop_na()
```

C.2 Stationarity and Exploratory Checks

```r
 # STL decomposition on log price
ts_log_price <- ts(cocoa_data$log_price, frequency = 365)
decomp <- stl(ts_log_price, s.window = "periodic")

### Commented out to allow plots to show in main report section
# plot(decomp, main="STL Decomposition of Log(Price)")
#
# # Augmented Dickey-Fuller Tests
# cat("ADF Test on log_price:\n")
# print(adf.test(cocoa_data$log_price))
# cat("\nADF Test on 1st diff_log_price:\n")
# print(adf.test(cocoa_data$diff_log_price))
```

C.3 Feature Engineering

```
 # Load libraries
library(tidyverse)
library(lubridate)
library(kableExtra)
library(dplyr)
library(knitr)
library(kableExtra)
library(forecast)
library(tseries)
library(xgboost)
library(randomForest)
library(e1071)

# Load Cocoa Price Data
cocoa_prices <- read.csv(here::here("data/raw_data/Daily Prices_ICCO.csv"), stringsAsFactors
  mutate(
    Date  = as.Date(Date, format = '%d/%m/%Y'),
    Price = as.numeric(gsub(",", "", ICCO.daily.price..US..tonne.))
  ) %>%
  select(Date, Price) %>%
  arrange(Date) %>%
  # Remove any rows with invalid dates or non-positive prices
  filter(!is.na(Date) & !is.na(Price) & Price > 0)

# Load Ghana Weather Data
ghana_weather <- read.csv(here::here("data/raw_data/Ghana_data.csv"), stringsAsFactors = FALS
  mutate(DATE = as.Date(DATE)) %>%
  # Average weather variables for each date
  group_by(DATE) %>%
  summarise(
    PRCP = mean(PRCP, na.rm = TRUE),
    TAVG = mean(TAVG, na.rm = TRUE),
    TMAX = mean(TMAX, na.rm = TRUE),
    TMIN = mean(TMIN, na.rm = TRUE)
  ) %>%
  ungroup() %>%
  arrange(DATE)

# Inner join Cocoa Prices and Weather Data
cocoa_data <- inner_join(
```

```
  cocoa_prices,
  ghana_weather,
  by = c("Date" = "DATE")
)

# Fill missing weather values with nearest known values
cocoa_data <- cocoa_data %>%
  tidyr::fill(PRCP, TAVG, TMAX, TMIN, .direction = "downup") %>%
  drop_na()
```

D Model Scripts

D.1 Time Series Cross-validation

```
 cv_folds <- list()
start_idx <- 1
step <- 90
train_window <- 360
test_window <- 90
n <- nrow(cocoa_data_lagged)

while ((start_idx + train_window + test_window - 1) <= n) {
  end_train_idx <- start_idx + train_window - 1
  end_test_idx <- end_train_idx + test_window
  cv_folds[[length(cv_folds) + 1]] <- list(
    train_indices = start_idx:end_train_idx,
    test_indices = (end_train_idx + 1):end_test_idx
  )
  start_idx <- start_idx + step
}
```

D.2 Baseline Model Implementation

```
 # Evaluation metrics
rmse_func <- function(actual, predicted) sqrt(mean((actual - predicted)^2))
mae_func  <- function(actual, predicted) mean(abs(actual - predicted))
mape_func <- function(actual, predicted) {
  mean(abs((actual - predicted)/actual)) * 100
```

```
}

results <- data.frame(
  Fold = integer(),
  Model = character(),
  RMSE = numeric(),
  MAE = numeric(),
  MAPE = numeric(),
  stringsAsFactors = FALSE
)

for (fold_i in seq_along(cv_folds)) {
  train_idx <- cv_folds[[fold_i]]$train_indices
  test_idx  <- cv_folds[[fold_i]]$test_indices

  train_fold <- cocoa_data_lagged[train_idx, ]
  test_fold  <- cocoa_data_lagged[test_idx, ]

  # Features: lags + climate variables
  x_cols <- c(grep("^lag_", names(train_fold), value=TRUE),
              "PRCP", "TAVG", "TMAX", "TMIN")

  train_x <- train_fold[, x_cols]
  test_x  <- test_fold[, x_cols]
  train_y <- train_fold$log_price
  test_y  <- test_fold$log_price

  # 1) Linear Regression
  lm_model <- lm(train_y ~ ., data=train_x)
  lm_pred_log <- predict(lm_model, newdata=test_x)
  lm_rmse  <- rmse_func(test_y, lm_pred_log)
  lm_mae   <- mae_func(test_y, lm_pred_log)
  lm_mape  <- mape_func(test_y, lm_pred_log)

  results <- rbind(results, data.frame(
    Fold=fold_i, Model="LinearRegression", RMSE=lm_rmse, MAE=lm_mae, MAPE=lm_mape
  ))

  # 2) XGBoost
  dtrain <- xgb.DMatrix(as.matrix(train_x), label=train_y)
  dtest  <- xgb.DMatrix(as.matrix(test_x))
```

```r
xgb_fit <- xgboost(data=dtrain, objective="reg:squarederror",
                   nrounds=50, verbose=0)

xgb_pred_log <- predict(xgb_fit, dtest)
xgb_rmse <- rmse_func(test_y, xgb_pred_log)
xgb_mae  <- mae_func(test_y, xgb_pred_log)
xgb_mape <- mape_func(test_y, xgb_pred_log)

results <- rbind(results, data.frame(
  Fold=fold_i, Model="XGBoost", RMSE=xgb_rmse, MAE=xgb_mae, MAPE=xgb_mape
))

# 3) Random Forest
rf_fit <- randomForest(x=train_x, y=train_y)
rf_pred_log <- predict(rf_fit, newdata=test_x)
rf_rmse <- rmse_func(test_y, rf_pred_log)
rf_mae  <- mae_func(test_y, rf_pred_log)
rf_mape <- mape_func(test_y, rf_pred_log)

results <- rbind(results, data.frame(
  Fold=fold_i, Model="RandomForest", RMSE=rf_rmse, MAE=rf_mae, MAPE=rf_mape
))

# 4) SVR
svr_fit <- svm(train_x, train_y, type="eps-regression", kernel="radial")
svr_pred_log <- predict(svr_fit, newdata=test_x)
svr_rmse <- rmse_func(test_y, svr_pred_log)
svr_mae  <- mae_func(test_y, svr_pred_log)
svr_mape <- mape_func(test_y, svr_pred_log)

results <- rbind(results, data.frame(
  Fold=fold_i, Model="SVR", RMSE=svr_rmse, MAE=svr_mae, MAPE=svr_mape
))

# 5) ARIMAX
# Build time series from the training portion
ts_train <- ts(train_fold$log_price, frequency=365)
xreg_train <- as.matrix(train_fold %>% select(PRCP, TAVG, TMAX, TMIN))

# Fit ARIMAX
arimax_fit <- auto.arima(ts_train, xreg=xreg_train, seasonal=FALSE)
```

```r
# Forecast
xreg_test <- as.matrix(test_fold %>% select(PRCP, TAVG, TMAX, TMIN))
arimax_fc <- forecast(arimax_fit, xreg=xreg_test, h=nrow(test_fold))
arimax_pred_log <- as.numeric(arimax_fc$mean)

arimax_rmse <- rmse_func(test_y, arimax_pred_log)
arimax_mae  <- mae_func(test_y, arimax_pred_log)
arimax_mape <- mape_func(test_y, arimax_pred_log)

results <- rbind(results, data.frame(
  Fold=fold_i, Model="ARIMAX", RMSE=arimax_rmse, MAE=arimax_mae, MAPE=arimax_mape
))


# 6) SARIMAX
ts_train_seas <- ts(train_fold$log_price, frequency = 365)

# Create exogenous matrix for training
xreg_cols <- c(grep("^PRCP|^TAVG|^TMAX|^TMIN", names(train_fold), value = TRUE),
               grep("^lag_", names(train_fold), value = TRUE))
xreg_train_seas <- as.matrix(train_fold[, xreg_cols])

# Fit a Seasonal ARIMAX model
sarimax_fit <- auto.arima(
  ts_train_seas,
  xreg = xreg_train_seas,
  seasonal = TRUE
)

# Forecast
xreg_test_seas <- as.matrix(test_fold[, xreg_cols])
sarimax_fc <- forecast(sarimax_fit, xreg = xreg_test_seas, h = nrow(test_fold))

sarimax_pred_log <- as.numeric(sarimax_fc$mean)
test_y <- test_fold$log_price

sarimax_rmse <- rmse_func(test_y, sarimax_pred_log)
sarimax_mae  <- mae_func(test_y, sarimax_pred_log)
sarimax_mape <- mape_func(test_y, sarimax_pred_log)

results <- rbind(results, data.frame(
  Fold = fold_i, Model = "SARIMAX", RMSE = sarimax_rmse,
```

```
    MAE = sarimax_mae, MAPE= sarimax_mape
  ))
}

results_summary <- results %>%
  group_by(Model) %>%
  summarise(
    Avg_RMSE = mean(RMSE),
    Avg_MAE  = mean(MAE),
    Avg_MAPE = mean(MAPE)
  ) %>%
  arrange(Avg_RMSE)
```

D.3 Model Refinement Implementation

```
 sarimax_grid <- expand.grid(
  p = c(0,1),
  d = c(1),
  q = c(0,1),
  P = c(0,1),
  D = c(0),
  Q = c(0,1)
)

results_sarimax <- data.frame(
  p = integer(), d = integer(), q = integer(),
  P = integer(), D = integer(), Q = integer(),
  Avg_RMSE = numeric(), Avg_MAE = numeric(), Avg_MAPE = numeric(),
  stringsAsFactors = FALSE
)

for (i in seq_len(nrow(sarimax_grid))) {
  # Extract parameters from grid
  p_ <- sarimax_grid$p[i]
  d_ <- sarimax_grid$d[i]
  q_ <- sarimax_grid$q[i]
  P_ <- sarimax_grid$P[i]
  D_ <- sarimax_grid$D[i]
  Q_ <- sarimax_grid$Q[i]
```

```r
fold_errors <- data.frame(RMSE=numeric(), MAE=numeric(), MAPE=numeric())

# Walk-forward CV
for (fold_i in seq_along(cv_folds)) {
  train_idx <- cv_folds[[fold_i]]$train_indices
  test_idx  <- cv_folds[[fold_i]]$test_indices

  train_fold <- cocoa_data_lagged[train_idx, ]
  test_fold  <- cocoa_data_lagged[test_idx, ]

  # Create time series from training portion
  ts_train <- ts(train_fold$log_price, frequency=365)

  xreg_cols  <- setdiff(names(train_fold), c("Date","Price","log_price"))
  xreg_train <- as.matrix(train_fold[, xreg_cols])
  xreg_test  <- as.matrix(test_fold[, xreg_cols])

  # Fit ARIMA in a tryCatch to handle occasional errors
  fit_result <- tryCatch({
    Arima(ts_train,
          order = c(p_, d_, q_),
          seasonal = list(order = c(P_, D_, Q_), period = 365),
          xreg = xreg_train)
  }, error = function(e) {
    NA
  })

  # If model fitting failed, skip this combo
  if (all(is.na(fit_result))) {
    fold_errors <- rbind(fold_errors, data.frame(RMSE=NA, MAE=NA, MAPE=NA))
  } else {
    fc <- forecast(fit_result, xreg = xreg_test, h=nrow(test_fold))
    pred_log <- as.numeric(fc$mean)
    actual_log <- test_fold$log_price

    fold_errors <- rbind(fold_errors, data.frame(
      RMSE = rmse_func(actual_log, pred_log),
      MAE  = mae_func(actual_log, pred_log),
      MAPE = mape_func(actual_log, pred_log)
    ))
  }
}
```

```
  # If the entire set of folds was NA, skip storing
  if (!all(is.na(fold_errors$RMSE))) {
    results_sarimax <- rbind(results_sarimax, data.frame(
      p=p_, d=d_, q=q_, P=P_, D=D_, Q=Q_,
      Avg_RMSE = mean(fold_errors$RMSE, na.rm=TRUE),
      Avg_MAE  = mean(fold_errors$MAE,  na.rm=TRUE),
      Avg_MAPE = mean(fold_errors$MAPE, na.rm=TRUE)
    ))
  }
}

results_sarimax <- results_sarimax %>% arrange(Avg_RMSE)
```

```
 xgb_grid <- expand.grid(
  max_depth    = c(3, 6),
  eta          = c(0.05, 0.1),
  colsample_bt = c(1.0, 0.8)
)

results_xgb <- data.frame(
  max_depth = integer(), eta = numeric(), colsample_bt = numeric(),
  Avg_RMSE = numeric(), Avg_MAE = numeric(), Avg_MAPE = numeric(),
  stringsAsFactors = FALSE
)

for (i in seq_len(nrow(xgb_grid))) {
  md  <- xgb_grid$max_depth[i]
  lr  <- xgb_grid$eta[i]
  cs  <- xgb_grid$colsample_bt[i]

  fold_errors <- data.frame(RMSE=numeric(), MAE=numeric(), MAPE=numeric())

  for (fold_i in seq_along(cv_folds)) {
    train_idx <- cv_folds[[fold_i]]$train_indices
    test_idx  <- cv_folds[[fold_i]]$test_indices

    train_fold <- cocoa_data_lagged[train_idx, ]
    test_fold  <- cocoa_data_lagged[test_idx, ]

    # Features (excluding Date, Price, log_price)
    xreg_cols <- setdiff(names(train_fold), c("Date","Price","log_price"))
```

```
    train_x <- as.matrix(train_fold[, xreg_cols])
    train_y <- train_fold$log_price
    test_x  <- as.matrix(test_fold[, xreg_cols])
    test_y  <- test_fold$log_price

    dtrain <- xgb.DMatrix(train_x, label=train_y)
    dtest  <- xgb.DMatrix(test_x)

    xgb_params <- list(
      objective        = "reg:squarederror",
      max_depth        = md,
      eta              = lr,
      colsample_bytree = cs
    )

    # Fit XGB with small nrounds for demonstration
    xgb_fit <- xgb.train(
      params  = xgb_params,
      data    = dtrain,
      nrounds = 100,
      verbose = 0
    )

    pred_log <- predict(xgb_fit, dtest)

    fold_errors <- rbind(fold_errors, data.frame(
      RMSE = rmse_func(test_y, pred_log),
      MAE  = mae_func(test_y, pred_log),
      MAPE = mape_func(test_y, pred_log)
    ))
  }

  results_xgb <- rbind(results_xgb, data.frame(
    max_depth = md,
    eta       = lr,
    colsample_bt = cs,
    Avg_RMSE  = mean(fold_errors$RMSE),
    Avg_MAE   = mean(fold_errors$MAE),
    Avg_MAPE  = mean(fold_errors$MAPE)
  ))
}
```

```
results_xgb <- results_xgb %>% arrange(Avg_RMSE)
```

References

International Cocoa Organization. 2025. "Daily Cocoa Futures Prices." https://www.icco.org.

Kumar, Kepulaje Abhaya, Cristi Spulbar, Prakash Pinto, Iqbal Thonse Hawaldar, Ramona Birau, and Jyeshtaraja Joisa. 2022. "Using Econometric Models to Manage the Price Risk of Cocoa Beans: A Case from India." *Risks* 10 (6): 115. https://doi.org/10.3390/risks10060115.

National Centers for Environmental Information. 2025. "Ghana Daily Climate Data." https://www.ncei.noaa.gov.

Sukiyono, Ketut, Musriyadi Nabiu, Bambang Sumantri, R. R. Novanda, Nyayu Neti Arianti, Sriyoto, M. Zulkarnain Yuliarso, Redy Badrudin, M. Mustopa Romdhon, and H. Mustamam. 2018. "Selecting an Accurate Cacao Price Forecasting Model" 1114 (1): 012116. https://doi.org/10.1088/1742-6596/1114/1/012116.

Zhang, Dabin, Shanying Chen, Liwen Ling, and Qiang Xia. 2020. "Forecasting Agricultural Commodity Prices Using Model Selection Framework with Time Series Features and Forecast Horizons." *IEEE Access*. https://doi.org/10.1109/ACCESS.2020.2971591.