Performance Analysis of LSTM and XGBoost Models Optimization in Forecasting Crude Palm Oil (CPO) Production at Palm Oil Mill (POM)

Kevien Aqbar
Information System Management Department,
Faculty of Technology and Engineering,
Gunadarma University, Jakarta

Riza Adrianti Supono
Information System Management Department,
Faculty of Technology and Engineering,
Gunadarma University, Jakarta

ABSTRACT
This research aims to test and compare the performance of LSTM (Long Short-Term Memory) and XGBoost (eXtreme Gradient Boosting) prediction models in forecasting the amount of crude palm oil (CPO) production in supporting production planning, stock management, and CPO sales. The background of this research was conducted because of the importance of accurate predictions in overcoming the instability of palm oil production in the future. Various prediction methods use univariate and multivariate data, and produce selected models such as ARIMA, SVR, Prophet, XGBoost, and LSTM. However, this research focuses on evaluating the performance of LSTM and XGBoost models by performing hyperparameter tuning optimization using multivariate data to find the most optimal model in forecasting CPO production with the smallest error rate. The results showed that after hyperparameter tuning, the LSTM model produced better prediction results with an accuracy rate of 93.7% and RMSE of 21.04. The XGBoost model also experienced improved performance after tuning with an RMSE of 22.17 and an accuracy rate of 92.8%. Although XGBoost initially provided superior prediction results closer to the actual data, the LSTM model became the best choice after passing the tuning process. This LSTM model can be used by POM management in production planning, tank stock management, and CPO sales. The results of this research are expected to help improve the efficiency and sustainability of the palm oil industry, as well as provide valuable information for stakeholders in making the right decisions.

Keywords
Time series, Forecasting, LSTM, XGBoost, Crude Palm Oil, Multivariate, Hyperparameter Tuning, RMSE, MAPE.

1. INTRODUCTION
Oil palm (Elaeis guineensis Jacq) is one of the potential oil-producing plants. One of the processed products of oil palm plantations is Crude Palm Oil (CPO), which is crude or unrefined palm oil. Palm oil can be processed by the Palm Oil Mill (POM) into food ingredients such as butter, cooking fat (shortening), chocolate additives, ice cream raw materials, making fatty acids, vanaspati, raw materials for various industries and animal feed [20].

Indonesia is the world's largest producer of crude palm oil (CPO). The high demand for palm oil from emerging economies in Asia such as India and China and the high level of domestic consumption are the main driving forces behind the growth of the palm oil industry in Indonesia. Indonesia's palm oil production in 2019 reached 51.8 million tons of CPO. This number increased by around 9 percent from 2018's production of 47.43 million tons, thus making a significant contribution to the national economy [14].

Forecasting the amount of CPO palm oil production is needed by POM management. The unstable amount of CPO oil production makes it difficult for companies to determine policies. The instability of palm oil production is caused by several factors such as natural or climatic factors. To overcome the problem of instability in production, it is necessary to predict and estimate daily production so that the company knows the amount of CPO production in the future which makes it easier for the company to make decisions.

This research aims to forecast the amount of palm oil (CPO) production in the future using the LSTM and XGBoost models by optimizing the hyperparameters. Previously, research has been conducted comparing various prediction methods using univariate and multivariate data, and the results show that LSTM and XGBoost models have good performance in predicting fluctuations in agricultural commodity prices in Malaysia through two experiments using univariate and multivariate data [2] and milk-tea sales volume in Beijing using multivariate data [22]. Another research also stated that XGBoost is better than LSTM in predicting software sales volume in one of the Russian startup companies using multivariate data [19].

In addition, several other studies compared the results of several prediction methods but by tuning the model hyperparameters first. It was found that the LSTM model produced better predictions than the ARIMA method in predicting agricultural production [10]. For the XGBoost model, it can be superior in short-term prediction and execution time in prediction research on electricity load usage in smart buildings [7]. Research on palm oil (CPO) production forecasting has been done before but using one ARIMA statistical method [12]. This research wants to test the performance of the two models in forecasting palm oil (CPO) production by optimizing the hyperparameters. The results of the analysis of the best model are expected to help company management in planning production, managing stock, and increasing CPO sales.

2. LITERATURE REVIEW
Some previous related research on time series forecasting has been carried out by several previous researchers which researchers summarize in Table 1 below.

Table 1. Summary of Related Research Review

Researcher	Research Subject	Research Methods	Advantages	Disadvantages
Chen et al., 2021 [2]	Agricultural commodity price predictions	LSTM, XGBoost, SVR, ARIMA, and Prophet. Two experiments were conducted (univariate data and multivariate data).	The LSTM model showed better accuracy than other methods when the number of datasets and complexity were increased in the second experiment (using multidimensional data) with the addition of features.	The process of finding suitable model hyperparameters is still a big challenge for global and non-linear solutions. The researcher did not perform hyperparameter optimization.
Zhang et all., 2021 [22]	Predicted product sales volume	LSTM, XGBoost, GBDT, ARIMA, and Prophet. Multivariate data. Hyperparameter optimization is performed.	The XGBoost model requires fewer iterations of boosting rounds to produce better predictions than LSTM, hence shortening the training time. XGBoost has the ability to generalize the model, prevent overfitting, and control model complexity compared to GBDT models.	There is no description of the hyperparameter configuration used for the LSTM model.
Swami et all., 2020 [19]	Predicted product sales volume	LSTM, XGBoost, and ARIMA. Multivariate data. Hyperparameter optimization is performed.	Experimental optimization of various hyperparameters for model optimization was carried out. It was found that the selection of influential hyperparameters for the LSTM model can make a difference in accuracy performance results. It was found that ARIMA only works well for univariate time series data, i.e. data that has a single variable.	The number of features used in multivariate data is not specified.
Mukhlis et all., 2021 [10]	Predicted production of agricultural products	LSTM and ARIMA. Univariate data. Hyperparameter optimization is performed.	Build the LSTM model by finding the best hyperparameters to get the optimal model.	The results of the initial RMSE metric before the tuning process are not explained.
Hadri et all., 2019 [7]	Predicted electricity load usage	LSTM, XGBoost, ARIMA, SARIMA. and Random Forest Univariate data.	The XGBoost model outperforms the other methods in terms of short-term prediction accuracy and execution time.	The form and number of datasets used are not specified. The tuning configuration and hyperparameter values used by each method are not described.

2.1 Forecasting

Prediction is a person's effort in guessing something that will happen in the future based on related information based on existing history or data that has been obtained in the past by using scientific methods in the prediction process [5]. Predictions can be divided into two types, namely qualitative predictions and quantitative predictions.

Qualitative predictions are not based on numbers or data, but on the opinion and intuition of the maker. While quantitative predictions can be based on numbers or data obtained from the past.

In forecasting data analysis, data can be divided into univariate and multivariate. Univariate data contains only one variable in each data sample, while multivariate data contains more than one variable. Examples of univariate data are height, age, weight, and temperature. Univariate data is analyzed using descriptive statistical analysis techniques. Multivariate data contains multiple variables, such as age, height, weight, gender, and medical history. Multivariate analysis is used to find relationships between variables and understand the complexity of data in a multidimensional space.

2.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a development of the Recurrent Neural Network (RNN) architecture. The LSTM method was first introduced by Hochreiter & Schmidhuber in 1997 as a solution to the dissatisfaction with RNN's ability to process long-term sequential data. One of the drawbacks of RNNs is the presence of vanishing gradients when using the backpropagation algorithm. The LSTM architecture can be seen in Figure 1.

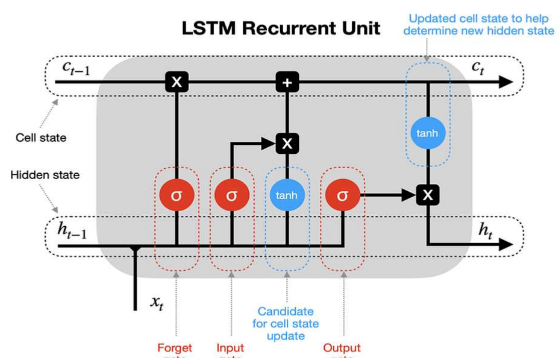


Fig 1: Long Short-Term Memory Architecture [4].

RNN is a type of Artificial Neural Network specifically designed to process sequential data. RNN can process information from the past for the learning process, and can be used in managing time series data. The RNN structure consists of an input layer, hidden layer, and output layer. The flow of information in RNN is one-way, from the input layer to the hidden layer, and from the previous hidden layer to the current hidden layer. The output of the hidden layer becomes the input for the next process. In its prediction, RNN uses current input data and input from previous data. The relationship between these inputs is useful for providing information to all hidden layers. Thus, the RNN has a memory that contains previous recorded information. Theoretically, RNNs should be able to handle long-term dependencies. However, in practice, RNNs are not effective in handling long-term dependencies due to the vanishing gradient problem. To overcome the problem, Hochreiter & Schmidhuber proposed a special type of RNN known as LSTM in 1997. LSTM is a development of RNN with a similar structure, which consists of an input layer, hidden layer, and output layer. The difference lies in the arrangement of the network in the hidden layer.

2.3 eXtreme Gradient Boosting (XGBoost)

eXtreme Gradient Boosting (XGBoost) is a machine learning algorithm that is an extension of the gradient boosting algorithm. XGBoost uses ensemble learning techniques, where multiple models are combined to improve prediction accuracy. XGBoost uses the decision trees method as the base learner, where each tree is generated iteratively by adding one tree at a time to the model. At each iteration, XGBoost calculates the residual error of the previous model and attempts to improve the prediction in the next iteration. To improve model quality, XGBoost uses regularization and pruning techniques. Regularization serves to control model complexity and avoid overfitting, while pruning aims to avoid the formation of tree branches that are not significant in predicting the target variable. XGBoost also uses gradient descent in the process of adjusting (optimizing) model parameters. This algorithm calculates the gradient value of the loss function at each iteration and moves the model parameters in the direction that reduces the loss value. Some of the advantages of XGBoost include its ability to handle large data efficiently, the ability to estimate the importance of features, and a high level of prediction accuracy. However, when using XGBoost, it is necessary to pay attention to several things, such as tuning the hyperparameters so that the model produces optimal predictions, considering the possibility of overfitting, and paying attention to the interpretation of the prediction results produced by the model. Figure 2 shows the architecture of XGBoost.

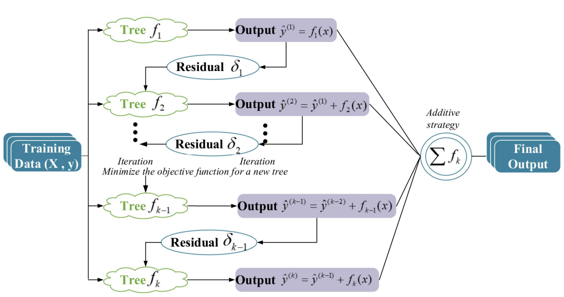


Fig 2: eXtreme Gradient Boosting Architecture [3].

2.4 Metric Score

Measurement of the error value of prediction data in time series forecasting can be measured using RMSE and MAPE by comparing it with actual data.

2.4.1 RMSE

Root Mean Square Error (RMSE) is a commonly used evaluation metric to measure the accuracy of a model in predicting continuous values. RMSE measures how close the predicted value is to the actual value of the predicted target. The RMSE calculation is based on the difference between the predicted value and the actual value, known as the residual or error. The residual is calculated as the difference between the actual value and the predicted value, and then the average of this squared difference is calculated. The RMSE calculation can be seen in formula 1.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{1}$$

RMSE produces a score that is measured in the same units as the target variable. The smaller the RMSE value, the more accurate the model is in predicting the target value.

2.4.2 MAPE

Mean Absolute Percentage Error (MAPE) is an evaluation metric used to measure the relative error of a forecasting model in predicting data in percentage units. MAPE is often used in time series forecasting models. The MAPE calculation can be seen in formula 2.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{2}$$

The smaller MAPE value indicates that the model has better performance in making predictions.

$$Persentase\ Akurasi = (1 - MAPE) * 100\ \% \tag{3}$$

To calculate the percentage level of accuracy, you can use formula 3.

2.5 Hyperparameter Tuning

Hyperparameter tuning is the process of finding the best combination of parameters in a model to achieve optimal performance. This is done by conducting trials and experiments using various hyperparameter values to find the combination that yields the best performance on the data used. Methods such as Grid Search and Random Search can be used to automatically search for the best combination in an efficient manner. Once all combinations are tested, the performance results are compared to select the best hyperparameter combination based on the evaluation metrics used, such as accuracy or RMSE. The main goal of hyperparameter tuning is to optimize model performance and speed up the model development process.

2.6 Knime Analytics Platform

Knime Analytics Platform is an open-source software used for data analysis, data integration, and predictive modeling. KNIME is based on the concept of workflow processing, where workflows are organized in the form of nodes that represent data analysis functions and paired with connections that describe the workflow.

3. RESEARCH METHODOLOGY

All The general research methodology is as shown in Figure 3. This research begins by collecting daily CPO production data that has been recorded previously, then the input data is processed with the preprocessing stage. Followed by feature selection of dependent and independent variables for multivariate data forecasting.

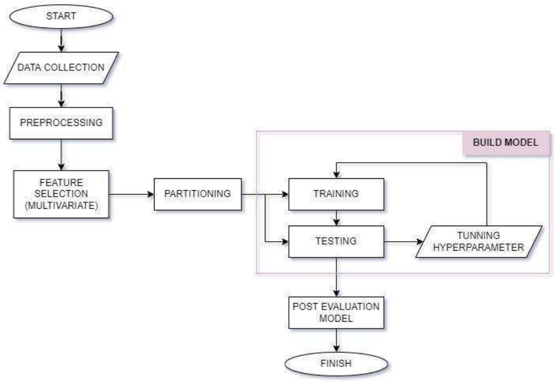
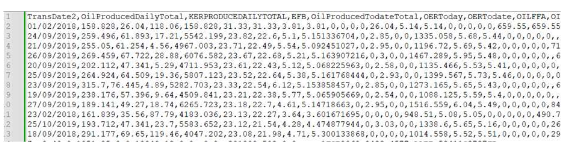


Fig 3: Flowchart of Research Methods

Then the source data is divided into training and testing data, followed by the construction of LSTM and XGBoost models using the default hyperparameters. After that, hyperparameter tuning is done to find the lowest prediction error. Finally, analysis and measurement of error and evaluation of model performance in forecasting production are carried out. The selected model will be saved to be used in the next prediction.

3.1 Data Collection

Dataset collection is a very important stage to ensure the quality and accuracy of the prediction model to be built. This research uses data extracted from the Finished Goods System application database belonging to one of the palm oil mill companies in Belitung, Indonesia. The data taken is the daily CPO production report. After exploration and extraction, the production data is exported in comma separated value (.csv) format. Then the result of this extraction file is called the research source dataset. The initial dataset consists of 1853 records with 98 data fields. An example of the dataset in the CSV file used can be seen in Figure 4.

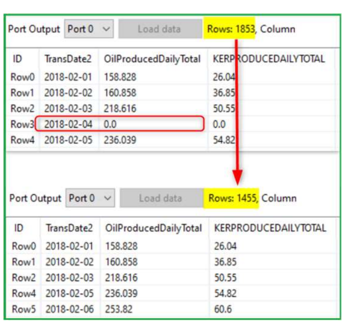


```
TransDate2,OilProducedDailyTotal,KERPRODUCEDAILYTOTAL,EFB,OilProducedTodateTotal,OERToday,OERTodate,OILFFA,OI
01/02/2018,158.828,26.04,118.06,158.828,31.33,31.33,3.81,3.81,0,0,0,0,26.04,5.14,5.14,0,0,0,0,0,659.55,659.55
24/09/2019,259.496,61.893,17.21,5542.199,23.82,22.6,5.1,5.151336704,0,2.85,0,0,1335.058,5.68,5.44,0,0,0,0,0,,
21/09/2019,255.05,61.254,4.56,4967.003,23.71,22.49,5.54,5.092451027,0,2.95,0,0,1196.72,5.69,5.42,0,0,0,0,0,71
26/09/2019,269.459,67.722,28.88,6076.582,23.67,22.68,5.21,5.163907216,0,3,0,0,1467.289,5.95,5.48,0,0,0,0,0,,6
20/09/2019,202.112,47.341,5.29,4711.953,23.61,22.43,5.12,5.068225963,0,2.58,0,0,1135.466,5.53,5.41,0,0,0,0,0,
25/09/2019,264.924,64.509,19.36,5807.123,23.52,22.64,5.38,5.161768444,0,2.93,0,0,1399.567,5.73,5.46,0,0,0,0,0
23/09/2019,315.7,76.445,4.89,5282.703,23.33,22.54,6.12,5.153858457,0,2.85,0,0,1273.165,5.65,5.43,0,0,0,0,0,,6
19/09/2019,238.176,57.396,9.64,4509.841,23.21,22.38,5.77,5.065905669,0,2.54,0,0,1088.125,5.59,5.4,0,0,0,0,0,,
27/09/2019,189.141,49.27,18.74,6265.723,23.18,22.7,4.61,5.14718663,0,2.95,0,0,1516.559,6.04,5.49,0,0,0,0,0,84
23/02/2018,161.839,35.56,87.79,4183.036,23.13,22.27,3.64,3.601671695,0,0,0,0,948.51,5.08,5.05,0,0,0,0,0,490.7
25/10/2019,193.712,47.341,23.7,5583.652,23.12,21.54,4.28,4.474877944,0,3.03,0,0,1338.6,5.65,5.16,0,0,0,0,0,26
18/09/2018,291.177,69.65,119.46,4047.202,23.08,21.98,4.71,5.300133868,0,0,0,0,1014.558,5.52,5.51,0,0,0,0,0,29
```

Fig 4: CSV Source Dataset View

3.2 Preprocessing

In the preprocessing stage, dataset cleaning is performed to clean the data from invalid values or outliers. These invalid values in the source dataset can come from data input errors and can affect model performance if not removed. In data preprocessing, it is necessary to remove missing values and duplicates to ensure the accuracy of the analysis results. In addition, columns containing target (dependent) variables that have null values should also be removed from the dataset so as not to affect the prediction results.



Port Output | Port 0 | Load data | Rows: 1853, Column

ID	TransDate2	OilProducedDailyTotal	KERPRODUCEDAILYTOTAL
Row0	2018-02-01	158.828	26.04
Row1	2018-02-02	160.858	36.85
Row2	2018-02-03	218.616	50.55
Row3	2018-02-04	0.0	0.0
Row4	2018-02-05	236.039	54.82

Port Output | Port 0 | Load data | Rows: 1455, Column

ID	TransDate2	OilProducedDailyTotal	KERPRODUCEDAILYTOTAL
Row0	2018-02-01	158.828	26.04
Row1	2018-02-02	160.858	36.85
Row2	2018-02-03	218.616	50.55
Row4	2018-02-05	236.039	54.82
Row5	2018-02-06	253.82	60.6

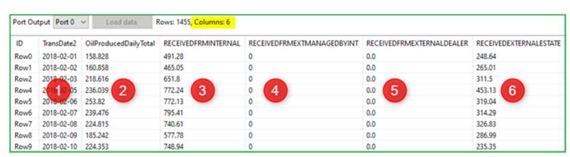
Fig 5: Missing Value Removal Result

Missing values are in the form of zero production values on certain dates. This happens because the factory delays production on that day due to several factors such as lack of palm fruit supply on that day, or holidays, factory machinery repairs, etc. so it is categorized as No Production Day. This zero value does not need to be replaced with an interpolated value or an average number of production before and after, because the factory only postpones the production process to the next day so that the amount of fruit received the day after is the accumulated value of the total fruit received on the previous No Production Day. Therefore, the deletion of this zero value will not have an impact on the accuracy of the model because the data will still be calculated on the next day. Figure 5 shows the result of deleting zero records in the target variable. The initial data consists of 1853 rows to 1455 rows of data.

3.3 Feature Selection (Multivariate)

Feature selection in multivariate time series forecasting is the process of selecting and determining the variables that have the most influence on the target variable. The prediction target (dependent) variable in the research is the amount of CPO production. Proper variable selection can improve model performance and avoid overfitting. The target variable whose value will be predicted is in the "OilProducedDailyToday" column variable, this data column is referred to as the selected dependent variable. Then in the selection of variables supporting predictions (independent) refer to previous research that discusses "CPO productivity is influenced by several factors including: FFB Quality, Harvesting Labor, Loses, Stagnation and Transportation of FFB in Palm Oil Mill processing" [8] which explains that there are several factors that most influence the amount of CPO oil production, such as the quality of FFB based on the management of suppliers. Supplier classification can be a benchmark for assessing FFB quality based on its management. Thus, insignificant variables can be eliminated to improve the performance and complexity of the model.

From the column dataset, it is known that there are 4 groups of FFB fruit suppliers. The four column variables are used as prediction support variables (independent) that can affect the target variable (dependent), namely the amount of CPO oil production. The selected independent column variables are "ReceivedFromInternal", "ReceivedFrmExtManagedByInt", "ReceivedExternalEstate", and "ReceivedFrmExternalDealer". Once determined, other variables that do not significantly affect the target variable are eliminated at this stage. From the results of the feature selection process, a total of 6 columns of data are left consisting of 1 timestep, 1 dependent variable, and 4 independent variables in Figure 6 below.



Port Output | Port 0 | Load data | Rows: 1455, Columns: 6

ID	TransDate2	OilProducedDailyTotal	RECEIVEDFRMINTERNAL	RECEIVEDFRMEXTMANAGEDBYINT	RECEIVEDFRMEXTERNALDEALER	RECEIVEDEXTERNALESTATE
Row0	2018-02-01	158.828	491.28	0	0.0	248.64
Row1	2018-02-02	160.858	465.05	0	0.0	265.01
Row2	2018-02-03	218.616	651.8	0	0.0	311.5
Row4	[illegible]	236.039	772.24	0	0.0	453.13
Row5	[illegible]	253.82	772.13	0	0.0	319.04
Row6	2018-02-07	239.476	795.41	0	0.0	314.29
Row7	2018-02-08	224.815	740.61	0	0.0	326.83
Row8	2018-02-09	185.242	577.78	0	0.0	286.99
Row9	2018-02-10	224.353	748.94	0	0.0	235.35

Fig 6: Result of Feature Selection Columns

3.4 Partitioning

The data partitioning process is the stage of dividing the dataset into training data and testing data. Data used for future prediction is usually based on past data or historical data. Therefore, the data must be divided into two parts, namely training data and testing data to test the performance of the model. Data sharing is done with a 70:30 ratio, where 70% is used as training data and 30% is used as testing data. This ratio is used because the number of datasets available is small, so the amount of testing data must be relatively more to provide representative results [6]. The division of datasets for the training and testing process can be seen in Table 2.

Table 2. Division of Training and Testing Data

Data subset	Training Data	Data Testing
Ratio (percentage)	70%	30%
Many rows of data	1018	437
Date range	Feb 1, 2018 - Aug 28, 2021	Aug 30, 2021 - Feb 28, 2023

3.5 Build Model-Tuning Hyperparameters

In the model building stage, there are three main processes, namely training, testing, and hyperparameter tuning. In the training process, the standard hyperparameter configuration is used to create a model with default input parameters provided by the KNIME application. Next, in the testing stage, the resulting model is tested by comparing the prediction results with the actual ground truth data value to measure how much error. Each change in the hyperparameter input value is iterated including the training and retesting process. The purpose of this iteration is to obtain the optimal value of hyperparameter input in order to produce predictions that are more accurate and suitable for making predictions in the future. From the evaluation process, the best hyperparameter combination with the lowest RMSE error rate is selected. In the initial stage, it is determined that the ground truth value or actual data value can actually be obtained from historical data taken from the results of the data partitioning stage for testing data needs. The ground truth value is obtained from testing data consisting of 437 timestep rows with data columns "TransDate" of datetime type and "OilProducedDailyToday" of decimal type.

3.5.1 LSTM

LSTM models based on deep learning are used because of their ability to overcome long-term dependency problems that often occur in time series. In multivariate LSTM models, lag columns are not required because LSTM models are able to learn the time dependency between independent input variables automatically through the gates mechanism in the LSTM network.

The first stage in the LSTM training process is to prepare a format that suits the needs of the LSTM model, which is in the form of three dimensions with the first dimension as the amount of data, the second dimension as the length of the time step, and the third dimension as the number of variables. The next step is to determine the architecture of the LSTM model to be used. The LSTM model architecture consists of several LSTM layers and other layers such as Dense and Dropout layers that are used to optimize model performance. At this stage, hyperparameter values such as the number of layers, the number of neurons in each layer, and the learning rate are also determined.

Table 3. LSTM Parameter Configuration

No.	Configuration	Standard Value	Option Value
1	Batch size value	32	16/25/32/64 *(*selected based on best tuning results)*
2	Epoch value	100	*25 / 50 / 100 / 200 (*selected based on best tuning results)*
3	Learning rate value	0.001	0.001
5	Number of *input* shape units (layers)	-	[?,4]
6	Number of hidden units (neurons)	-	100
7	Number of unit *outputs* (scalar)	1	1
9	Optimizer	Adam	Adam

After the model architecture is determined, the next step is to perform the training process. After the training process is complete, the LSTM model will produce a predicted value for the testing data. The predicted value is then compared with the actual value to measure model performance using the Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) metrics. If the model performance is not as expected, the hyperparameter tuning stage is carried out again by changing the hyperparameter values such as the number of neurons or the learning rate. After the optimal hyperparameter value is found, the training stage is carried out again on all data and the LSTM model is ready to be used to make predictions on future data. Table 3 shows the LSTM hyperparameter configuration.

3.5.2 XGBoost

The XGBoost model is a machine learning-based model that is often used to predict target variables in forecasting multivariate time series data. This model is one of the ensemble learning methods that combines several weak learner models to produce a more accurate and robust model.

The formation of the XGBoost model begins with inputting the features that will be used in the model, then conducting the training and testing process until the hyperparameter tuning process to find the hyperparameter combination that produces the best model performance. XGBoost uses the concept of ensemble learning, where a small number of simpler predictive models (decision trees) are combined to form a more powerful model. XGBoost builds a series of decision trees sequentially, where each tree attempts to correct the prediction errors made by the previous tree. At each iteration, the model attempts to minimize the prediction error by optimizing the MSE objective function. During training, model parameters such as tree depth, number of trees, and learning rate can be adjusted to improve model performance.

Table 4. XGBoost Parameter Configuration

No.	Configuration	Standard Value	Option Value
1	ETA (learning rate)	0.3	*0.01 / 0.1 / 0.3 / 0.5 (*selected based on best tuning results)*
2	Boosting rounds (n_estimators)	100	*50/100/500/1000 (*selected based on best tuning results)*
3	Maximum depth	6	6
4	Minimum Child Weight	1	1

In the training stage, the XGBoost model will learn from previously separated training data using the gradient boosting algorithm, where each iteration will optimize the residual error in the training data. After the training process is complete, the model testing process is carried out using testing data to test the performance of the model. Model error measurement uses Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) metrics. If the model performance is still considered inadequate, another iteration can be done at the hyperparameter tuning stage. Table 4 shows the XGBoost parameter configuration.

3.6 Post Evaluation Model

During the testing process, the model error is measured using the Root Mean Squared Error (RMSE) evaluation metric. RMSE is one of the evaluation metrics used to measure how accurate a model is in making predictions on unprecedented data. RMSE measures the average of the squared difference between the predicted value and the actual value. A smaller RMSE value indicates that the model has better performance in making predictions. RMSE is calculated using formula 1.

To find the most optimal hyperparameter input in both LSTM and XGBoost models, experiments were conducted by combining hyperparameter values and testing the resulting output RMSE results. In the LSTM model, a combination of epoch and batch inputs was tested during the data training process. The standard epoch value is 100 and then optimized by changing it to 25, 50, 100, and 200. Each change is also combined with a change in the standard batch size from a value of 32 to 16, 25, 32, and 64. The results of this trial are recorded in Table 5 which shows the measurements based on the hyperparameter changes of the LSTM model during the tuning process.

Table 5. Experimental Tuning Results of LSTM Model Hyperparameter Combination

LSTM Model Selection	Hyperparameters		RMSE Score
	Epoch	Batch Size	
A1	50	16	22.99697684
A2	50	32	21.65533268
A3	50	64	27.77631393
A4	50	128	230.8384973
A5	100	16	21.63178887
A6	**100** (default)	**32** (default)	**25.12878067**
A7	100	64	22.66962264
A8	100	128	24.0643584
A9	**200** (best)	**16** (best)	**21.04007526**
A10	200	32	21.70610187
A11	200	64	23.66222402
A12	200	128	26.18869054
A13	400	16	23.90643547
A14	400	32	22.46230656
A15	400	64	230.8384973
A16	400	128	23.29463239

Furthermore, to find the most optimal hyperparameter in the XGBoost model, an experiment was conducted by combining the Boosting Rounds and ETA values during the training process. The standard Boosting Rounds value is 100 then change it to 50, 100, 500, and 1000. While the standard ETA value is 0.3 then optimized by changing it to 0.01, 0.1, 0.3, and 0.5. The results of this trial are recorded in Table 6 which shows the measurements based on the changes in the hyperparameters of the XGBoost model during the tuning process.

Table 6. Experimental Tuning Results of XGBoost Model Hyperparameter Combinations

XGBoost Model Options	Hyperparameters		RMSE Score
	Boosting Rounds	ETA	
B1	50	0.01	139.1989942
B2	100	0.01	84.93552845
B3	500	0.01	22.18953444
B4	1000	0.01	22.42534886
B5	**50** (best)	**0.1** (best)	**22.17147585**
B6	100	0.1	22.53402925
B7	500	0.1	25.5606617
B8	1000	0.1	27.50669944
B9	50	0.3	23.20820116
B10	**100** (default)	**0.3** (default)	**24.47011576**
B11	500	0.3	28.8475257
B12	1000	0.3	29.8412094
B13	50	0.5	24.66703016
B14	100	0.5	26.35674767
B15	500	0.5	29.74048242
B16	1000	0.5	29.98399413

4. RESULT AND ANALYSIS

A time series forecasting model is built to predict the daily production amount of CPO palm oil using multivariate variables. The LSTM and XGBoost models are optimized by finding the best hyperparameters with the smallest error rate. Furthermore, model performance metrics are measured to evaluate the quality of the resulting model. This research uses KNIME Analytics Platform with Keras Deep Learning integration to build the model. The results of the research workflow design using the KNIME application can be seen in Figure 7.

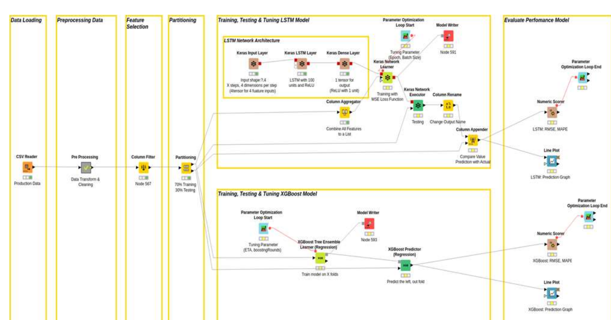


Fig 7: Model Building and Evaluation

The score evaluation results of both LSTM and XGBoost models using default standard hyperparameters measured by RMSE and MAPE metrics can be seen in Table 7 below. The XGBoost model is superior to LSTM because it has a lower RMSE value than LSTM.

Table 7. Model Error Measurement Score Results Using Default Parameters

Model Type	Standard Hyperparameter Input (Default)	RMSE Score	MAPE Score
LSTM	Epoch = 100, Batch Size = 32	25.13	0.081
XGBoost	BoostingRounds = 100, ETA = 0.3	24.47	0.079

Then from the results of hyperparameter tuning experiments in tables 5 and 6, it can be seen that model A9 obtained the lowest RMSE for the LSTM model, namely 21.04 when compared to using the standard hyperparameter in model A6 with an RMSE of 26.68. This proves that after tuning, there is an increase in model performance in predicting data with a percentage increase in improvement of 16.3%. While in the XGBoost model, the lowest RMSE value is 22.17 in model B5. When compared to the standard hyperparameter, it indicates an improvement of 9.4% from the value of 24.47 before tuning. The final results of the best hyperparameters selected for model building can be seen in Table 8.

Table 8. Preferred Hyperparameters (Best)

Model Type	Model	Preferred Hyperparameter Input (Best)	RMSE Score (Best)	Percentage of Improvement
LSTM	A9	Epoch = 200, Batch Size = 16	21.04	+ 16.3%
XGBoost	B5	BoostingRounds = 50, ETA = 0.1	22.17	+ 9.4%

Analysis of the comparison from Tables 7 and 8 shows that in the LSTM model, the selected epoch value is larger than the default epoch while the selected batch size value is smaller than the default batch size. With a larger number of epochs the model has more iterations to adjust weights and learn complex patterns and is able to generalise well to data that has never been seen before, and can increase tolerance to noise. Whereas a smaller batch size can quickly identify and adapt to more specific and complex patterns.

In the XGBoost model, reducing the BoostingRounds and ETA values can help reduce the risk of overfitting especially in less complex data patterns. With a smaller learning rate, the model is more likely to learn more general patterns and avoid patterns that are too specific so that it can produce a more stable and reliable model in producing more accurate predictions on new data. Reducing the value of BoostingRounds will limit the number of trees in the final model, thereby reducing model complexity and the possibility of overfitting, especially suitable for relatively small training datasets.

From the calculation of the evaluation metric score of the two models after the hyperparameter tuning process, it is found that the LSTM model can produce a lower RMSE error rate than the XGBoost model. The RMSE for LSTM after tuning is 21.04 and XGBoost is 22.07. From the data in Table 9, it can be concluded that LSTM after tuning is superior to XGBoost. This is inversely proportional to the results of the calculation of the metric score before the tuning process, where XGBoost is superior because it produces a smaller RMSE error rate. This is also in line with the results of the calculation of the MAPE error metric score where the LSTM model after tuning is superior because it has a lower MAPE score than the XGBoost model, where the MAPE for LSTM is 0.063 while XGBoost is 0.072. So if the percentage of accuracy level is calculated using formula 3, the accuracy level of the LSTM model is 93.7% and the XGBoost model is 92.8%.

Table 9. Model Comparison Error Score Final Results

Model Type	Preferred Hyperparameter Input (Best)	RMSE Score	MAPE Score	Final Accuracy Level
LSTM	Epoch = 200, Batch Size = 16	21.04	0.063	93.7%
XGBoost	BoostingRounds = 50, ETA = 0.1	22.17	0.072	92.8%

Figures 8 and 9 show the visualization results in the form of graphs comparing the predicted value of the model with the actual data for the LSTM and XGBoost models after the best hyperparameter tunning is obtained.

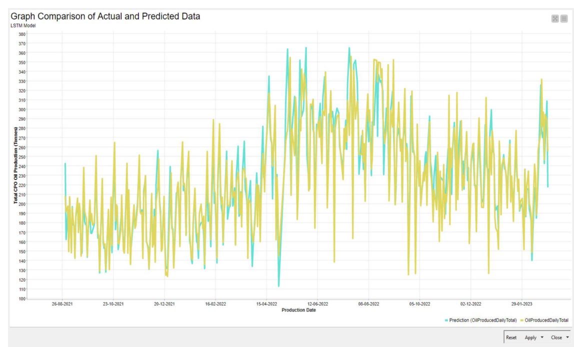


Fig 8: Graph of LSTM Model Prediction Results compared to Actual Values

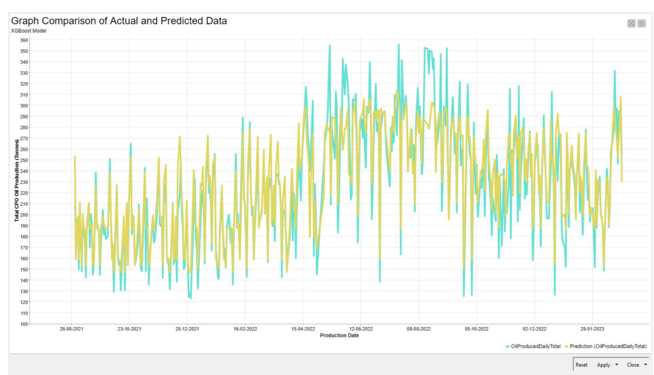


Fig 9: Graph of XGBoost Model Prediction Results compared to Actual Values

5. CONCLUSION

The conclusion of this research is to test and compare the performance of LSTM and XGBoost prediction models in forecasting the amount of CPO oil production. Initially, XGBoost provided prediction results that were close to the actual data, but after tuning the hyperparameters, the LSTM model became the best choice. Evaluation after tuning shows that the LSTM model has an RMSE of 21.04, a MAPE of 0.063, and an accuracy rate of 93.7%. This shows a significant performance improvement compared to the prediction before tuning. Meanwhile, the XGBoost model has an RMSE of 22.17, a MAPE of 0.072, and an accuracy rate of 92.8% after tuning, showing improved performance compared to before tuning. Although XGBoost was initially better than LSTM, these results show that after hyperparameter tuning, the LSTM model is superior in predicting CPO oil production data. Hyperparameter tuning is performed with the Grid Search method to select the best combination. In the LSTM model, the hyperparameters tuned are epoch and batch size, while in the XGBoost model, the hyperparameters tuned are boosting rounds and eta. The results of this research resulted in the LSTM model as the best model in predicting CPO oil production, which can be used to support production planning, tank stock management, and CPO sales. The selection of the best model is based on the smallest RMSE metric, as this is more relevant for management who want to minimize the forecasting error in the same unit of measure.

6. ACKNOWLEDGMENT

The author would like to thank Gunadarma University and all lecturers, especially Dr. Riza Andrianti Supomo and especially my family who have supported this research.

7. REFERENCES

[1] Adlin U. Lubis. (2008), Kelapa Sawit (Elaeis Guineensis Jacq) di Indonesia. Edisi kedua, Penerbit Pusat Penelitian Kelapa Sawit

[2] Chen, Z., Goh, H. S., Sin, K. L., Lim, K., Chung, N. K., & Liew, X. Y. (2021). Automated agriculture commodity price prediction system with machine learning techniques. *Advances in Science, Technology and Engineering Systems Journal*, *6*(4), 376-384. https://doi.org/10.25046/aj060442

[3] Cheng, F., Yang, C., Zhou, C., Lan, L., Zhu, H., & Li, Y. (2020). Simultaneous determination of metal ions in zinc sulfate solution using UV–VIS spectrometry and SPSE-xgboost method. *Sensors (Switzerland)*, *20*(17), 1–14. https://doi.org/10.3390/s20174936

[4] Dobilas, S. (2022). LSTM recurrent neural Networks — How to teach a network to remember the past. Medium. https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e

[5] Fardhani, A. A., Simanjuntak, D. I., & Wanto, A. (2018). Prediksi Harga Eceran Beras Di Pasar Tradisional Di 33 Kota Di Indonesia Menggunakan Algoritma Backpropagation. *Jurnal Infomedia*, *3*(1), 25–30.

[6] Gholamy, Afshin; Kreinovich, Vladik; and Kosheleva, Olga. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation. *Departmental Technical Reports (Computer Science)*. *2*(1209). https://scholarworks.utep.edu/cs_techrep/1209

[7] Hadri, S., Naitmalek, Y., Najib, M., Bakhouya, M., Fakhri, Y., & Elaroussi, M. (2019). A comparative study of predictive approaches for load forecasting in smart buildings. *Procedia Computer Science*, *160*, 173–180. https://doi.org/10.1016/j.procs.2019.09.458

[8] Junaidi, Koko. (2012). Analisis Faktor-Faktor Yang Mempengaruhi Produktivitas CPO PKS Sei Meranti PT. Perkebunan Nusantara Iii (Persero). Medan: Universitas Medan Area.

[9] Luo, J., Zhang, Z., Fu, Y., & Rao, F. (2021). Time series prediction of COVID-19 transmission in America using LSTM and XGBoost algorithms. *Elsevier B.V*, *27*. https://doi.org/10.1016/j.rinp.2021.104462

[10] Mukhlis, M., Kustiyo, A., & Suharso, A. (2021). Peramalan Produksi Pertanian Menggunakan model long short-term memory. *BINA INSANI ICT JOURNAL*, *8*(1), 22. https://doi.org/10.51211/biict.v8i1.1492

[11] Nyoman, N., Pinata, P., Sukarsa, M., Kadek, N., & Rusjayanthi, D. (2020). Prediksi Kecelakaan Lalu Lintas di Bali dengan XGBoost pada Python. *JURNAL ILMIAH MERPATI*, *8*(3), 188–196.

[12] Oktarina, T., & Rasmila, D. (2018). PERAMALAN PRODUKSI CRUDE PALM OIL (CPO) MENGGUNAKAN METODE ARIMA PADA PT. SAMPOERNA AGRO TBK. In *Seminar Nasional Sistem Informasi Indonesia*.

[13] Oktarina, T., & Rasmila, D. (2018). PERAMALAN PRODUKSI CRUDE PALM OIL (CPO) MENGGUNAKAN METODE ARIMA PADA PT. SAMPOERNA AGRO TBK. *SESINDO*, 251–260.

[14] *Produksi Sawit 2019 Capai 51,8 Juta ton |Republika online*. (2020, February 4). Republika Online. https://ekonomi.republika.co.id/berita/q54sje370/produksi-sawit-2019-capai-518-juta-ton

[15] Rizki, M., Basuki, S., & Azhar, Y. (2020). Implementasi Deep Learning Menggunakan Arsitektur Long Short Term Memory Untuk Prediksi Curah Hujan Kota Malang. *REPOSITOR*, *2*(3), 331–338.

[16] Saragih, V. D., Melaca, K. M., Darmawan, R., & Hendrianie, N. (2018). Pra Desain Pabrik CPO (Crude PalmOil) dan PKO (Palm Kernel Oil) Dari Buah Kelapa Sawit. *JURNAL TEKIK ITS*, *7*(1), 181–183.

[17] Silipo, R., & Palacio, V. (2022). *KNIME Advanced Luck: A Guide for Advanced Users*. Switzerland: KNIME Press.

[18] Supriyanto, Y., Ilhamsyah, M., & Enri, U. (2022). Prediksi Harga Minyak Kelapa Sawit Menggunakan Linear Regression Dan Random Forest. *Jurnal Ilmiah Wahana Pendidikan*, *8*(7). https://doi.org/10.5281/zenodo.6559603

[19] Swami, D., Shah, A. D., & Ray, S. K. B. (2020). *Predicting Future Sales of Retail Products using Machine Learning*. http://arxiv.org/abs/2008.07779

[20] Ulfiah, K., Hakim, A., al Hakim, L., Ilham, D., Dimas Ilham, M., Muliyanto, M., Julianti, S., Sri Julianti, N., Ariyanti, N., Ramadhanti, N., Astuti, P., Puji Astuti, R., Nurfaizah, R., Giwangkara, R., & Suryani, R. (2018). Economic Value of Palm Oil (Elaeis guinensis) for Indonesian People. *Munich Personal RePEc Archive*, *90215*. https://mpra.ub.uni-muenchen.de/90215/

[21] Wiranda, L., & Sadikin, M. (2019). PENERAPAN LONG SHORT TERM MEMORY PADA DATA TIME SERIES UNTUK MEMPREDIKSI PENJUALAN PRODUK PT. METISKA FARMA. *Jurnal Nasional Pendidikan Teknik Informatika*, *8*(3), 184–196.

[22] Zhang, L., Bian, W., Qu, W., Tuo, L., & Wang, Y. (2021). Time series forecast of sales volume based on XGBoost. *Journal of Physics: Conference Series*, *1873*(1). https://doi.org/10.1088/1742-6596/1873/1/012067