Extreme gradient boosting (XGBoost) method in making forecasting application and analysis of USD exchange rates against rupiah

S F N Islam, A Sholahuddin, and A S Abdullah

Department of Computer Science, Universitas Padjadjaran, Jl. Raya Bandung Sumedang km 21 Jatinangor, Sumedang 45363, Indonesia

*atje.setiawan@unpad.ac.id

Abstract. Economic conditions in Indonesia are still unstable, causing the US dollar exchange rate to increase. This is because most international transactions in Indonesia use US dollars. Prediction or forecasting is chosen as one of the important things in choosing a market to invest in buying and selling. This research will focus on making forecasting applications and analyzing the exchange rate of USD against rupiah based on time series data or temporal datasets from the Investing.com site using machine learning methods, namely Extreme Gradient Boosting (XGBoost). Applications created using the python programming language and streamlit framework. Modeling is carried out using the Knowledge Discovery in Database (KDD) methodology with the stages of dividing the dataset with a 50:50 percentage share into test and train data. The modeling uses hyperparameter tuning values, namely n_estimators = 1000, max_depth = 1, x_colsample_bytree = 0.9894, x_gamma = 0.9989, x_min_child = 1.0, x_reg_lambda = 0.2381, and x_subsample = 0.7063 with best loss or RMSE 451.4151. The values of Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) when making the model were 6.61374% and 3.95485%. Meanwhile, when testing the model, the RMSE is 0.23577% and MAPE is 0.11643%. Keywords: exchange rate; forecasting; Extreme Gradient Boosting (XGBoost); streamlit framework; Knowledge Discovery in Database (KDD); Root Mean Square Error (RMSE); Mean Absolute Percentage Error (MAPE).

1. Introduction

Indonesia is the largest archipelago country in the world and also the largest country in Southeast Asia. Indonesia is a large country that is rich in natural resources, there are many foreign investors from Asia and Europe who take the opportunity to invest in this country [1]. Apart from investing, there are also many large companies in Indonesia that trade with foreign companies. Indonesian companies or investors buying goods or services produced in the United States will be forced to buy US dollars, and vice versa. The drastic change in the foreign exchange rate between the Indonesian rupiah and the US dollar will significantly affect the price of goods. These facts motivate many studies that focus on prediction of exchange rates [2].

Economic conditions in Indonesia are still unstable, causing the US dollar exchange rate to increase. This is because most international transactions in Indonesia use US dollars. Unstable economic conditions can reduce the amount of capital investment by the retrogressive development of foreign investors in Indonesia because foreign investors play a very important role in the rate of economic growth in Indonesia [3].
Prediction is chosen as one of the important things in choosing a market to invest in in terms of buying and selling. Every prediction that comes out accuracy is very calculated. The fact that the number of manual prediction errors is influenced by many factors, one of which is caused by human calculation errors. This of course will be fatal for investors. In addition, manual prediction will be a waste of time, thought, and effort when large data are used. Therefore, we need a method that can solve problems that cannot be solved manually.

This research will focus on predicting and analyzing the exchange rate of USD against rupiah based on time series data or temporal dataset. Periodic data (time series) is a set of observational data obtained from time to time. In general, data collection is carried out within a certain period. Periodic data can be used as a basis for decision making and forecasting [4].

Extreme Gradient Boosting (XGBoost) is a machine learning method. XGBoost can perform regression or classification. XGBoost works on structured data such as data that is already arranged in rows of data, so it doesn't work on unstructured data such as images or videos. The Gradient Boosting Algorithm is a machine learning technique used to build predictive tree-based models. Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) are used to determine how accurate XGBoost is in forecasting the future exchange rate of USD dollars against rupiah.

2. Method
This research begins by examining the topics contained in several recent journals on economics, such as inflation and stocks. The exchange rate is the most important thing in analyzing current information. Apart from that, in investing or investing, the exchange rate is also needed. So that the exchange rate data can be obtained from investment sites. Exchange rate data obtained from the Investing.com website is one of them. Exchange rate data can be downloaded by adjusting the currency to be compared. After that, you can set the range of data to be downloaded and the data obtained is in the form of csv.

This research includes needs analysis, data collection, Knowledge Discovery in Databases (KDD) process, design of application systems using the waterfall method, black box testing of web applications and results and discussion.

2.1 Knowledge Discovery in Databases (KDD)
Knowledge Discovery in Database (KDD) refers to areas of research that use data mining methods from pattern recognition, machine learning, and database techniques in a broad context. The terms data mining and KDD are used as a process of finding hidden information in a data set. KDD refers to a multi-step process which can be very interactive and iterative [5]. KDD has several stages, namely selection, preprocessing, data transformation, data mining and finally interpretation evaluation.

2.2 Extreme Gradient Boosting (XGBoost)
XGBoost stands for Extreme Gradient Boosting. XGBoost is a machine learning method that has been used by many data researchers, especially in various data competitions and machine learning competitions, reflecting superior performance over other methods [6]. XGBoost can perform classification and regression, which has been validated in many cases, such as store sales prediction, customer behavior prediction, ad click prediction, hazard prediction, web text prediction, malware classification.

Boosting is an ensemble technique in which new models are added to correct mistakes made by existing models. Models are added sequentially until no further refinement can be made. The ensemble technique uses the tree ensemble model which is a set of classification and regression trees (CART). The ensemble approach is used because one CART, usually, does not have strong predictive power. By using a CART set (example a tree ensemble model) a number of predictions from several trees are considered.

Gradient boosting is an approach in which a new model is created that predicts the residues or errors of the previous model and then adds them together to make a final prediction. Unlike the decision tree, in gradient boosting, making a tree starts with a leaf. The initial leaf is filled with the average value of the features to be predicted. The following are the stages of gradient boosting [7].
**Input:** Data \(\{(x_i, y_i)\}_{i=1}^{n}\), and differentiable Loss Function \(L(y_i, F(x))\).

\[
L(y_i, F(x)) = \frac{1}{2} (y_i - F(x))^2
\]  

(1)

where,

- \(y_i\) : observational or real data to \(i\)
- \(F(x)\) : prediction data to \(i\).

**Step 1:** Initiate a model with a constant value:

\[
F_0(x) = \text{argmin} \sum_{i=1}^{n} L(y_i, \gamma)
\]  

(2)

where,

- \(\gamma\) : The gamma parameter booster with default value is 0.

**Step 2:** for \(m = 1\) to \(M\) :

a. Count \(r_{im} = - \left[ \frac{\partial (y_i F(x_i))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)}\) for \(i = 1, \ldots, n\)

b. *Fit a regression tree* on the \(r_{im}\) value and create a terminal area \(R_{jm}\) for \(j = 1 \ldots J_m\)

c. For \(j = 1 \ldots J_m\) count \(y_{jm} = \text{argmin} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)\)

d. Update \(F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} y_{jm} I(x \in R_{jm})\)

**Step 3:** Output \(F_M(x)\).

XGBoost is a supervised learning, which uses a train data with several features \(x_i\) to predict a target \((y_i)\). XGBoost has the same steps as gradient boosting but has a unique tree. The difference between regular and extreme gradient boosting in regression is the predictive initial value set. In gradient boosting the initial prediction value is the result of the average real value of one feature that will be predicted as in equation 2. Whereas in XGBoost the initial prediction value is taken randomly, but the default value commonly used is 0.5.

The formation of a tree is based on the calculation of the Loss Function in equation 1 or what is called the calculation of residuals. After calculating the residuals, the next step is to make a tree at XGBoost, calculate the similarity score.

\[
\text{Similarity score} = \frac{(\text{Sum of residuals})^2}{\text{Number of residuals} + \lambda}
\]  

(3)

where,

- \(\lambda\) : Lambda parameter regularization with default value is 0.

The similarity score calculation in equation 3 is used to calculate the gain value. This gain value will be used for tree trimming at XGBoost.

\[
\text{Gain} = \text{Left}_\text{Similarity} + \text{Right}_\text{Similarity} + \text{Root}_\text{Similarity}.
\]  

(4)

Pruning trees at XGBoost uses calculations as in equation 5.

\[
\text{Gain} - \gamma.
\]  

(5)

If the result of equation 5 is positive then the branch is not omitted. But, if the calculation result is negative then the branch is omitted. There is also an exception if the first branch of the calculation is positive and the calculation for root is negative, then the root is not removed. Making the tree ends...
with pruning. After trimming is finished, then the calculation of the predictive output on each leaf is calculated using equation 6.

\[
\text{Output} = \frac{\text{Sum of residuals}}{\text{Number of residuals} + \lambda}.
\]  

(6)

Modelling using the XGBoost method has the following steps [8]:

1. **Feature selection**
   The specific steps of feature selection via XGBoost are data cleaning, data feature extraction, and data feature selection based on the feature importance score.

2. **Modelling training**
   The model is trained based on the features that have been selected with parameters. The stages of making a model are carried out with the stages described.

3. **Parameter optimization**
   Parameter optimization is carried out with the aim of minimizing the error between the predicted value and the actual value. The steps for determining the hyperparameters of the XGBoost model are as follows:
   a. Step one: The estimator count is first tuned to optimize XGBoost when fixing learning rates and other parameters.
   b. Step two: A different combination of max_depth and min_child_weight is set to optimize XGBoost.
   c. Step three: Max delta step and gamma are adjusted to make the model more conservative with the parameters specified in steps 1 and 2.
   d. Step four: A different combination of subsample and col-sample_bytree is set to prevent overfitting.
   e. Step five: The parameter settings were increased to make the model more conservative.
   f. Step six: Learning rate is reduced to prevent overfitting.

   The hyperparameter values in the XGBoost method greatly affect the model created. So that in this method there are several hyperparameters that need to be adjusted to produce a good model for predicting. Table 1 shows the hyperparameters in the XGBoost method.

| Parameter Type       | Parameter                  | Description                                      | Main purpose                                           |
|----------------------|----------------------------|--------------------------------------------------|-------------------------------------------------------|
| Booster parameters   | Max_depth                  | Maximum depth of tree                             | Increasing this value will make the model more complex and more overfit |
|                      | Min_child_weigth           | Minimum sum of weights in a child                 | The bigger the min_child_weight, the more conservative the algorithm will be |
|                      | Gamma                      | Minimum loss reduction                             | The bigger the gamma, the more conservative the algorithm will be |
|                      | Subsample                  | Subsample ratio of the training instances         | This is used in updates to prevent overfitting         |
|                      | Col sample by a tree       | Subsample ratio of columns for each tree          | This is used in updates to prevent overfitting         |
| Regularization       | Lamda                      | Regularization term on weights                    | Increasing this value will make the model more conservative |
| parameters           | Number of estimators       | Number of estimators                              | It is used to determine the number of iterative calculations |

Table 1. Hyperparameters in the XGBoost method
2.3 Mean Absolute Percentage Error (MAPE)
Mean Absolute Percentage Error (MAPE) is a statistical measure of how accurate the forecasting system is. MAPE measures this accuracy as a percentage, and can be calculated as the average absolute percent error for each time period minus the actual value divided by the actual value. The smaller the MAPE gave a better result. MAPE can be calculated by equation 7. Forecasting ability is very good if the MAPE value is less than 10% and has good forecasting ability if the MAPE value is less than 20% [9].

\[
MAPE = \frac{100}{n} \left( \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right| \right) \tag{7}
\]

where,
\(n\) : Test set size
\(y_t\) : actual or original data value
\(\hat{y}_t\) : the value of forecasting data.

2.4 Root Mean Square Error (RMSE)
According to [10] to measure the goodness of a forecasting method, RMSE is needed. In determining the most accurate estimation method, it is necessary to compare the estimation methods used. The best method is determined by the resulting RMSE value. The RMSE formula can be written according to equation 8.

\[
RMSE = \sqrt{\frac{\sum_{t=1}^{n} (y_t - Y_t)^2}{n}} \tag{8}
\]

where,
\(n\) : Number of samples
\(y_t\) : Actual value
\(Y_t\) : Predicted value.

3. Result and Discussion
3.1. Result and Implementation
The system implementation is made on one web page using the Python programming language and streamlit framework. Source code or program line is written in a single file called nilaitukar.py, which is a forecasting application for USD against rupiah using the Extreme Gradient Boosting (XGBoost) method. Source code contains lines of programs to display the user interface and the processes carried out for preprocessing, visualization and forecasting with XGBoost.

The first part is the application title and the section for selecting a dataset which is built using streamlit framework as shown at Figure 1. In this section the user can select a dataset which will then be used for modeling and forecasting. The dataset used for this research is real time data regarding the USD exchange rate against the rupiah originating from the Investing.com website from 19 September 2016 to 07 January 2020. The dataset that can be obtained consists of 862 rows and 2 (two) columns, namely date index and exchange rate data. In addition, this section provides an explanation of the application.
The preprocessing part is done by adding the time series feature to the selected dataset. We built an application program using streamlit framework as shown on figure 2. The first stage is to read the file and change the data type. Then change the date order to the same date format. Finally, the time series feature was added. At the preprocessing stage, a time series feature is made based on the datetime index using the one hot encoding technique. A feature is a known variable that is used to inform a variable estimate for which only historical information is known. The time series features made are date, hour, dayofweek, quarter, month, year, dayofyear, dayof month, and dayofyear. The purpose of making the feature is to forecast using the XGBoost model so that it can do CART.

The third part is the graphic visualization of the distribution of training data and testing data. The distribution is done with a percentage of 50:50. As previously explained, the data training will be used to create models and test data for testing the model.
The hyperparameter tuning part is the process of finding the best hyperparameter value that has the smallest RMSE value. This value search takes a long time is almost 1 hour. The hyperparameter values looking for are n_estimators, n_jobs, max_depth, x_colsample_bytree, x_gamma, x_min_child, x_reg_lamda, and x_subsample. Search is done using python programming with hyperopt library to get the hyperparameter value. After a successful search, max_depth = 1, x_colsample_bytree = 0.9894, x_gamma = 0.9989, x_min_child = 1.0, x_reg_lamda = 0.2381, and x_subsample = 0.7063 with best loss or RMSE 451.4151.

The next section is the information section regarding feature importance. This section provides information to the user in the form of a graph. It is intended that the user can find out which features are very influencing in the modeling process. After getting the results by comparing the F score of each feature, it can be seen that the feature that has the highest importance is the dayofweek with an F score of 304. This shows that the dayofweek feature has a significant influence in shaping the model. In addition, the year feature is in second place with an F score of 96. Then quarter, dayof month, and month.

The visualization of forecasting results is done by making a model first and then forecasting it. Visualization is done after getting the predicted value and visualizing the graph with that value. Figure 4 shows the results of the source code above which displays a visualization of the predicted and actual values in May.

![Figure 3](image.png)  
**Figure 3.** Application of the training testing data sharing visualization section

![Figure 4](image.png)  
**Figure 4.** Application of the forecasting result visualization section

The next section is the information section regarding the RMSE and MAPE values from the results of modeling and testing from the previously split testing data. The RMSE and MAPE values that have been obtained, namely 6.6137 for RMSE and 3.9548 for MAPE.
The last part is testing the previously created models. This section has a similar process, namely bypassing the section on selecting the testing dataset. Once selected, the preprocessing stage of reading files and changing data types is carried out. Then change the date order to the same date format. Finally, the time series feature was added.

The preprocessing stage is complete, then enter the stage of viewing or visualizing feature importance and testing data forecasting. After the predictive value is obtained, then visualization of the predicted value is carried out with the actual value. All stages have been carried out, finally an evaluation was carried out by calculating the RMSE and MAPE values.

Figure 5 shows visualizing graphs of actual predicted values from the test results. Testing this model is important to know and measure the accuracy of the model that has been made. The results in figure 9 show the results of model testing using the new dataset, the most important feature is dayofyear. This is the same as before when using old data. In addition, the RMSE value obtained is 0.2358 and the MAPE is 0.1164.

3.2. Black Box Testing
Application testing with the black box testing method in this forecasting application has two conditions, namely "success (S)" and "failure (G)". Furthermore, the application is distributed to thirty users to test according to the scenario that has been made. Before the user does the test, the user needs to install python and streamlit first. This is done because the application is based on the python language and uses the streamlit framework.

In addition, it is also necessary to install python libraries such as xgboost and hyperopt. This library needs to be installed personally because it is not a built-in library from the python installer package. Placing the dataset also matters, the dataset needs to be placed in the right folder. The dataset used to make the model is stored in the datasets folder and the dataset that will be used as test data is stored in the datatest folder. The following are the results of testing by thirty users in this forecasting application:

| No | Distance (m) | Velocity (ms⁻¹) | Test results |
|----|-------------|----------------|--------------|
|    |             |                | S  | F  |
| 1  | Run the application program | The user successfully enters the application and opens the application page | 100% | 0 |
| 2  | Choose a dataset | The user has successfully | 100% | 0 |
to build a model selected a dataset to create a model

3 Perform a hyperparameter tuning search

The system can perform a hyperparameter tuning search and display its value.

4 Choose a test dataset for model testing

The user can select a dataset to test the model, so that the following functions can run.

Table 2 shows the test results on each part of this forecasting application that requires interaction with the user. Based on the results of application testing in each section on the application page, the average success category is 95% and failure is 5%. This shows that application success has a higher percentage of failures, so that the application can be said to run well.

Possible failures that occur are due to the user not installing the libraries required by the program. Such as the hyperopt library which is used to find hyperparameter tuning. So that failure occurs in that scenario. In addition, due to a failure in the hyperparameter tuning search function, the next function does not work properly.

3.3. Testing with a New Dataset

Testing applications based on datasets can also be said to test models using other datasets. As with the application, this can also be done. The new dataset that is used as the object of testing is a dataset taken from the Investing.com website regarding the exchange rate of USD in rupiah. The new dataset used for the trial is data from February 10, 2020 to April 9, 2020. Table 3 shows the difference in feature importance to the old data used to create the model with the new data used for the trial.

**Table 3. Comparison of old data importance feature graph results with new data**

| Data   | Importance Feature Graph |
|--------|--------------------------|
| Old    | ![Feature Importance Graph](image) |
| New    | ![Feature Importance Graph](image) |

In table 3 it can be seen that both the old data and the new data the most important feature in making predictions or forecasting is day of year. So, it can be concluded that these features are very influential in modeling and forecasting. The most important feature in the second position has a difference, for old data the second most important feature is year, this is because the old data has a more complex time structure and a longer time span. Whereas in the new data, the second position is
day off month. This is because the new data is the complexity of the data contained on the day of the month, besides that because the new data only contains data in the same year.

| Data   | RMSE | MAPE  |
|--------|------|-------|
| Old    | 6.6137 | 3.9548 |
| New    | 0.2358 | 0.1164 |

In table 4, it can be seen that the RMSE and MAPE values of each data. The RMSE value on the old data is 6.61374% and the new data is 0.23577%. So it can be said that the RMSE value gets smaller when the model is used on new data. The MAPE value in the old data is 3.95485% and the new data is 0.11643%. Both MAPE values are below 10%, so it can be said that the model has excellent forecasting capabilities.

4. Conclusion
Based on the research and implementation and discussion that has been done previously, it can be concluded that:
1. The results of data analysis on the USD exchange rate dataset for rupiah taken from the Investing.com website prior to preprocessing have one feature, namely "Last" type data object. This feature contains the exchange rate data that will be forecasted. Then the feature is changed to float data type and preprocessing is carried out. The preprocessing process is carried out by adding the time series feature. So that after going through the preprocessing stage the features add to nine features, namely 'last', dayofweek, quarter, hour, month, year, dayofyear, dayof month, and weekofyear.
2. Forecasting application design can be done using the Unified Modeling Language (UML) by making use case diagrams, sequence diagrams, and activity diagrams. Forecasting applications using the Python programming language using the Numpy, Pandas, Matplotlib, XGBOOST, Sklearn, Math, and Hyperopt libraries. Making the display uses a website-shaped streamlit framework.
3. Hyperparameter tuning in the modeling process is obtained with max_depth = 1, x_colsample_bytree = 0.9894, x_gamma = 0.9989, x_min_child = 1.0, x_reg_lamda = 0.2381, and x_subsample = 0.7063 with best loss or RMSE 451.4151.
4. The RMSE value is getting smaller when the model is used for forecasting the new dataset. So it can be said that the XGBoost method is suitable for forecasting, especially in the case of this exchange rate and both of MAPE values are below 10%, so it can be said that the model has excellent forecasting capabilities.

Acknowledgments
The authors thank Rector Universitas Padjadjaran who provided financial assistance for the dissemination of research through Academic Leadership Grant 2020 with contract number: 1427/UN6.3.1/LT/2020. Also thanks to the reviewers for the valuable review of this paper.

References
[1] Wang T C, Kuo S H, Anh T N, and Li L 2013 Forecast the foreign exchange rate between Rupiah and US dollar by applying grey method, Lecture Notes in Electrical Engineering, 156 LNEE 321–327
[2] Winarko T, Nugroho A S and Widiputra H 2009 Foreign Exchange Rate Prediction between Indonesian Rupiah and US Dollar using Transductive Learning, in Proceedings international conference on rural information and communication technology
[3] Ernita, Amar S and Syofyan E 2013 Analisis Pertumbuhan Ekonomi, Investasi, dan Konsumsi di Indonesia, *Jurnal Kajian Ekonomi* 1 2 176 193

[4] Qonita A, Pertwi A G and Widiyaningtyas T 2017 Prediction of rupiah against us dollar by using arima, in *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)* 746 750

[5] Karim R, Tretten P and Kumar U 2018 eMaintenance (Dd) 559 587

[6] Wang W, Shi Y, Lyu G and Deng W 2017 Electricity Consumption Prediction Using XGBoost Based on Discrete Wavelet Transform, *DEStech Transactions on Computer Science and Engineering* (aiea) 716 729

[7] Abrahan S P, T V 2019 Daily evapotranspiration prediction using gradient boosted regression

[8] Ji S, Wang X, Zhao W and Guo D 2019 An application of a three-stage XGboost-based model to sales forecasting of a cross-border e-commerce enterprise *Mathematical Problems in Engineering* 2019 8503252

[9] Margi S K and Pendawa W S 2015 Analisa dan Penerapan Metode Single Ezponensial Smoothing untuk Prediksi Penjualan Pada Periode Tertentu (Studi Kasus: PT. Media Cemara Kreasi) *Prosiding SNATIF* 259 266

[10] Lestari N and Wahyuningsih N 2012 Peramalan Kunjungan Wisata dengan Pendekatan Model SARIMA (Studi kasus : Kusuma Agrowisata) *Jurnal Sains dan Seni ITS* I 1 *Jurnal Sains dan Seni ITS* A29 A33