Quality Prediction in Process of Flotation Plant with Gradient Boosting and Long Short-Term Memory Model

Khaleeq Ahmed[1] and Yashar Pourrahimian[2]

[1,2] School of Mining and Petroleum Engineering, University of Alberta, Edmonton, Canada

ABSTRACT

Machine learning has demonstrated its efficacy in various database domains, indicating interest in its application in mining processes. Popular models, such as support vector machines, and random forests, deep learning models, including convolutional neural networks (CNN), recurrent neural networks (RNN), and generative adversarial networks (GAN), are emerging as optimal tools for handling large-scale datasets. Predicting the output quality in mining processes, such as flotation plants, is crucial for optimization. This paper employs two powerful methods, the Extreme Gradient Boost Regressor (XGBR) and Long Short-Term Memory (LSTM) networks, to predict the concentration purities for iron and waste silica. For this purpose, 23 variables reflecting a floatation plant were monitored and collected every two minutes over a half-year span. This data is then processed, split and restructured for the model use. The solved problem demonstrates the effectiveness of XGBR and LSTM in predicting ore concentrate quality and waste. Compared with LSTM, XGBR exhibits better predictive capabilities for iron and silica on test and validation datasets. This leads to the foundation for realizing automation control of the floatation plant, and this study should encourage traditional machine learning as well as deep learning in mineral processing engineering.

1. Introduction

In mineral processing, froth floatation is an extensively used physiochemical separation phenomenon for valuable and unwanted minerals. Froth floatation employs the physiochemical surface characteristics of the materials, either hydrophilic or hydrophobic [1, 2]. Since they are critical to process management and optimization, processing engineers must promptly and effectively monitor concentrate grade and recovery in froth flotation plants. Implementing online monitoring systems typically requires costly and sophisticated equipment. In order to estimate concentrate grade and recovery using processing parameters, predictive models must be created. Conventional knowledge-based models, which usually rely on explicit calculations for the flotation process, are less feasible to develop due to issues like the difficulty of formulating and solving mathematical equations for a dynamic boundary (the interface between the collection and froth zone), and the limited understanding of the physicochemical principles governing flotation subprocesses [3, 4]

Machine learning (ML), serving as a key component of data-driven strategies, has proven its effectiveness in various data-intensive domains like spam filtering, machine translation, and natural language processing. Consequently, it has sparked the curiosity of engineers involved in mineral processing [5]. Several machine learning models, such as multilayer perception [6], support vector machine [7], and random forest [8] have been developed to model flotation processes. These models have shown success in modelling laboratory flotation processes, which typically involve limited and straightforward process data. However, when faced with tasks encompassing extensive and complex data structures, such as high-dimensional, autocorrelated, and temporal data, shallow machine learning models or statistical learning models have proven ineffective. This implies that the previously mentioned shallow machine learning models are not suitable for effectively modelling engineering flotation processes.

In contrast, deep learning models, exemplified by the convolutional neural network (CNN), the recurrent neural network (RNN), and the generative adversarial network (GAN), have emerged as optimal tools for handling large-scale and intricate datasets due to their deep, intricate, and adaptable architectures [10]. Presently, there are only a limited number of reported successful cases involving the application of deep learning in mineral processing, with a particular scarcity of instances pertaining to manufacturing froth flotation [11]. Technical background and database details

In froth floatation, reverse cationic floatation is the most commonly used process to separate the iron particles from undesired particles, such as silica, the dominant gangue mineral in iron ores. During this process, silica particles cling to air bubbles and float to the surface, while iron minerals stay in the water and produce sediment with a high concentration of iron.

The data used in this paper is from a real manufacturing floatation plant, where silica is separated from the iron particles using revere cationic floatation [12]. The flowsheet for the iron ore flotation process is shown in Figure 1 with a zoom on a single flotation cell in our target flotation plant. This flotation facility feeds iron pulp into three parallel-operating flotation cells (flotation cells 1, 2, and 3 in Figure 1) at a rate of approximately 400 t/h. Silica-rich froth is continuously removed from the surface. The process is repeated in the subsequent cell (flotation cell 4, 5, 6 in Figure 1) as depressed sediment with high iron content flows into it. The last flotation (flotation cell 7 in Figure 1) receives an input stream from the output streams of flotation cells 4, 5, and 6.

Data from the real manufacturing plant, collected every 20 seconds for a span of almost half year, was aggregated to a two-minute interval for analysis. Each data sample has 23 measurable values that fall into one of four categories: environment variables (rows 3 to 7), process variables (rows 8 to 21), processed materials (rows 22 to 23), and raw materials (rows 1 to 2), which relate to the purities of iron and silica fed into the plant. Our objective in this paper is to forecast, based on the measurable values, the purity of processed materials at time t (2 minutes) as presented in Table 1.

1.1. Data Processing

Data collected from the database is raw data. The data is then analyzed and further structured to meet the model's requirement. Outliers in the collected data are identified and removed using an outlier detection formula in which IQR is Inter Quartile range, Q1 is the first quartile and Q3 is the third quartile

$$IQR = Q3 - Q1 \tag{1}$$

$$lower_{bound} = Q1 - 1.5 * IQR \tag{2}$$

$$upper_{bound} = Q3 + 1.5 * IQR \tag{3}$$

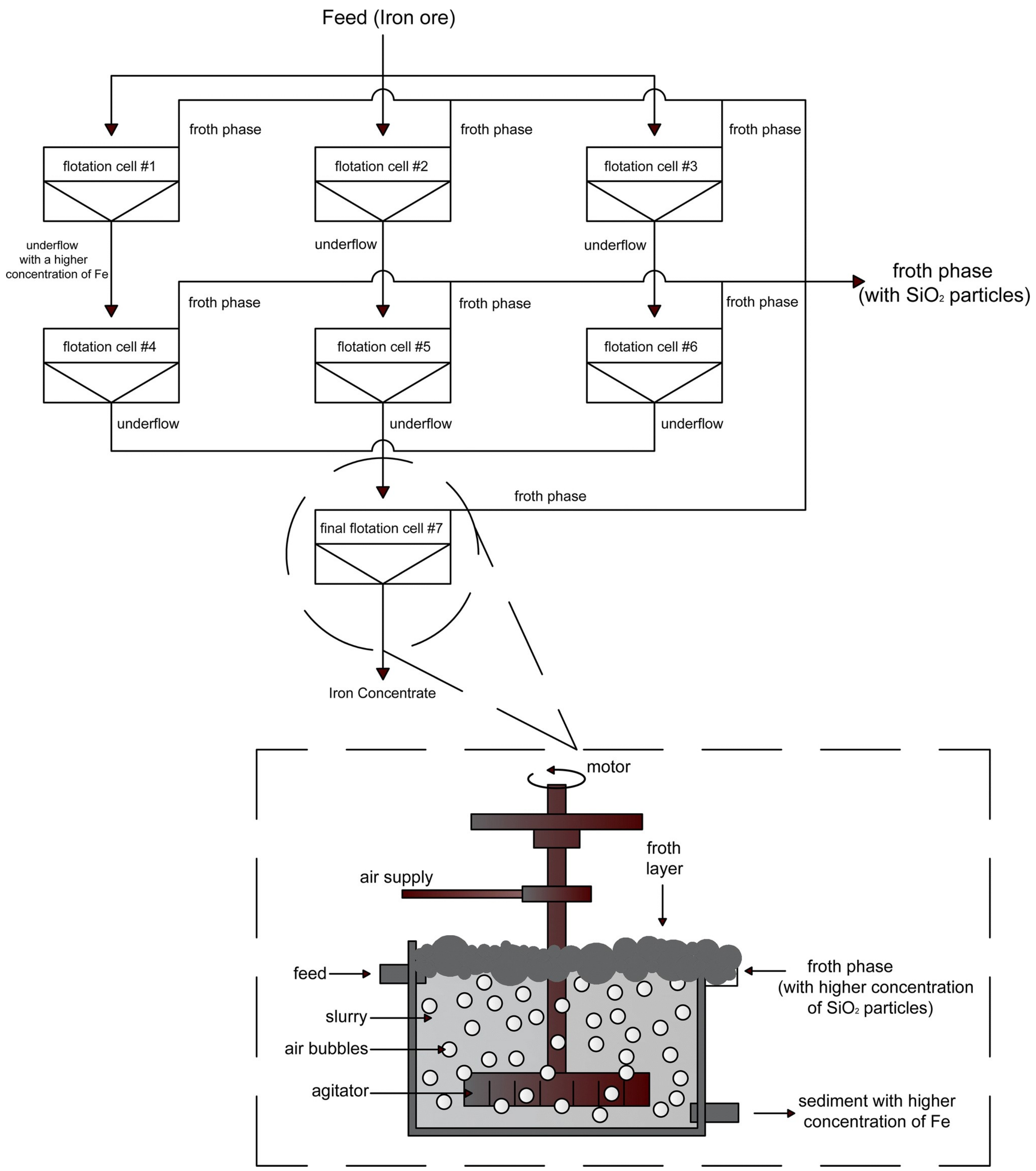

Figure 1. Iron ore flotation process flowsheet featuring a single flotation cell magnified [9].

Table 1. A brief overview of the gathered engineering database.

Timestamp	**3/10/2017 12:00:00 AM**	**3/10/2017 00:02:00 AM**	**3/10/2017 00:04:00 AM**	**3/10/2017 00:06:00 AM**	**3/10/2017 00:08:00 AM**
percent_iron_feed	55.2	55.2	55.2	55.2	55.2
percent_silica_feed	16.98	16.98	16.98	16.98	16.98
starch_flow	3079.1	3150.39	3784.96	3523.24	3570.12
amina_flow	564.697	558.472	557.983	563.416	571.533
ore_pulp_flow	396.533	397.852	394.834	394.453	401.572
ore_pulp_ph	10.0705	10.0755	10.0804	10.0854	10.0936
ore_pulp_density	1.74	1.74	1.74	1.74	1.74
flotation_column_01_air_flow	250.73	249.17	250.049	250.181	250.291
flotation_column_02_air_flow	248.906	249.829	246.533	247.456	251.653
flotation_column_03_air_flow	249.521	251.147	249.829	249.324	251.279
flotation_column_04_air_flow	295.096	295.096	295.096	295.096	295.096
flotation_column_05_air_flow	306.4	306.4	306.4	306.4	306.4
flotation_column_06_air_flow	250.356	250.928	250.378	249.434	252.114
flotation_column_07_air_flow	251.873	246.533	248.643	248.181	250.115
flotation_column_01_level	444.384	461.45	451.991	453.542	448.438
flotation_column_02_level	443.269	421.41	426.335	500.023	425.826
flotation_column_03_level	460.449	467.79	443.024	454.624	461.59
flotation_column_04_level	439.92	458.59	429.998	458.008	485.898
flotation_column_05_level	451.588	453.84	452.629	414.796	482.759
flotation_column_06_level	433.539	448.05	440.77	460.818	466.418
flotation_column_07_level	425.458	451.34	463.421	403.059	479.645
percent_iron_concentrate	66.91	66.91	66.91	66.91	66.91
percent_silica_concentrate	1.31	1.31	1.31	1.31	1.31

After removing outliers, the data is then plotted as a correlation heatmap to check the correlation between different features, as shown in Figure 2. After analyzing the heatmap, it was observed that some of the features have a high correlation (either positive or negative). To check the distribution of our target variables, which are percent_silica_concentrate and percent_iron_concentrate, histograms are plotted. The distribution explains these variables' characteristics, which can help with data preprocessing, model selection, and decision-making. The first target variable shows a positively skewed distribution, while the second target variable shows a slightly negatively skewed distribution, as shown in Figure 3.

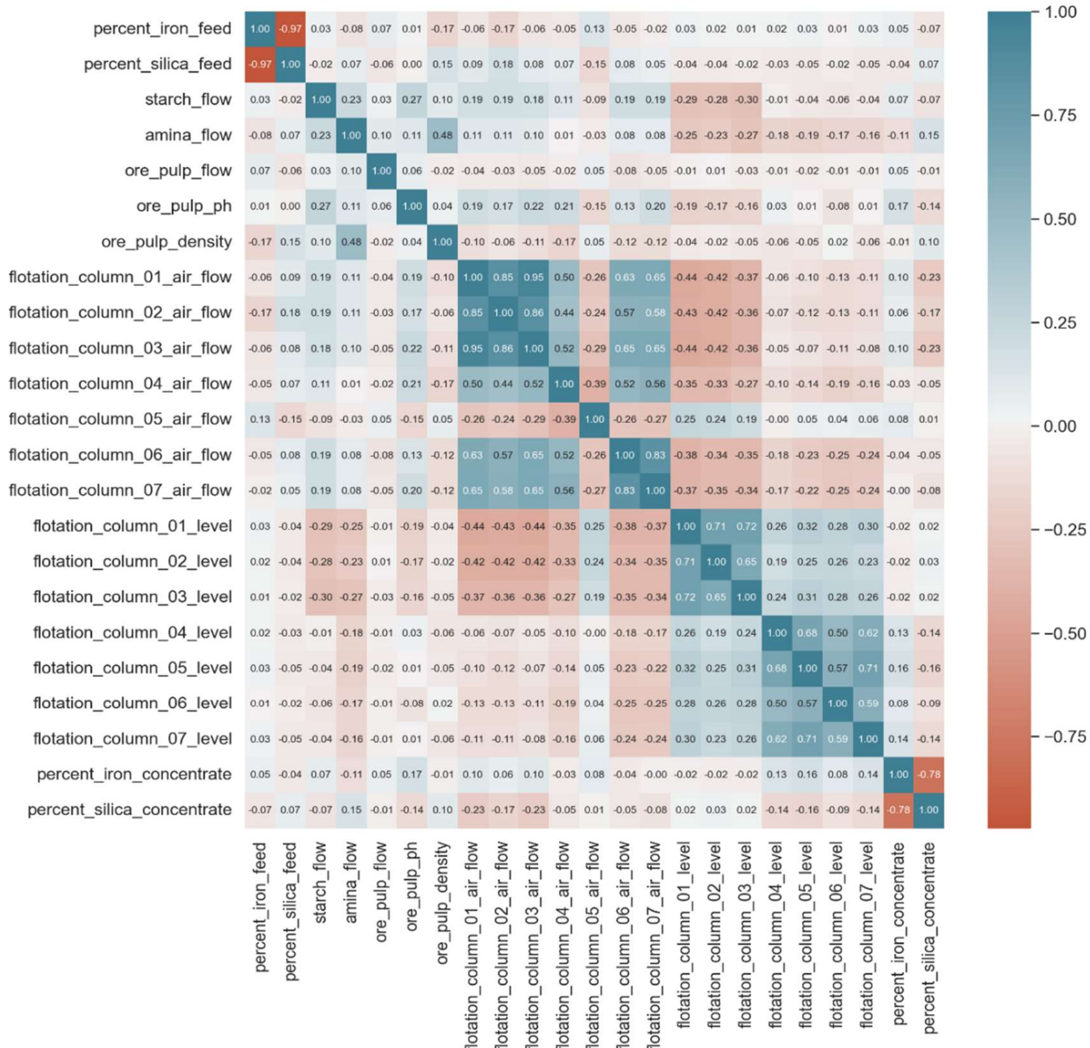

Figure 2. Correlation heatmap of the collected data.

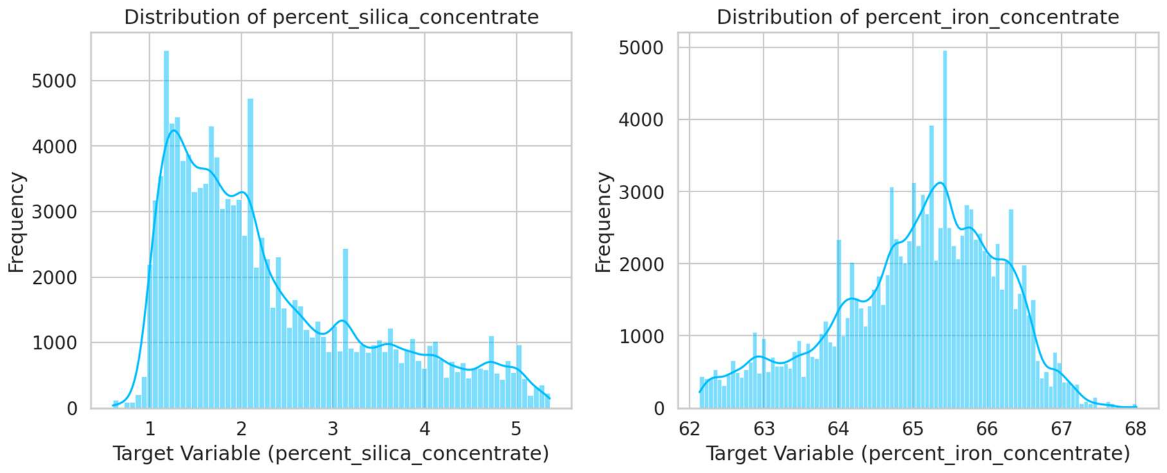

Figure 3. Distributions of the target variables.

As mentioned above, the correlation heatmap shows some highly correlated features. The highly correlated features are averaged to minimize the number of features, and a new data set is created.

1.2. Seasonality

Periodic fluctuations that repeat at certain intervals over time are referred to as seasonality in data. Many datasets, especially those pertaining to time series, exhibit this recognizable pattern. Seasonal elements like the season, month, week, or even day influence this pattern. Seasonality is an important factor to consider and examine in various fields, including retail, finance, environmental studies, and more. The current data set also has a time series relation with the output, so the seasonality in the data set was checked for months, days and hours. Seasonality can be accounted for addressed by various statistical and ML models, such as deep learning's LSTM (Long Short-Term Memory) networks and statistics' ARIMA (Auto-Regressive Integrated Moving Average). This paper employs LSTM and extreme gradient boosting techniques necessitating feature engineering to account for seasonality. After incorporating the seasonality, the correlation heatmap is plotted to check the collinearity of the data, as shown in Figure 4.

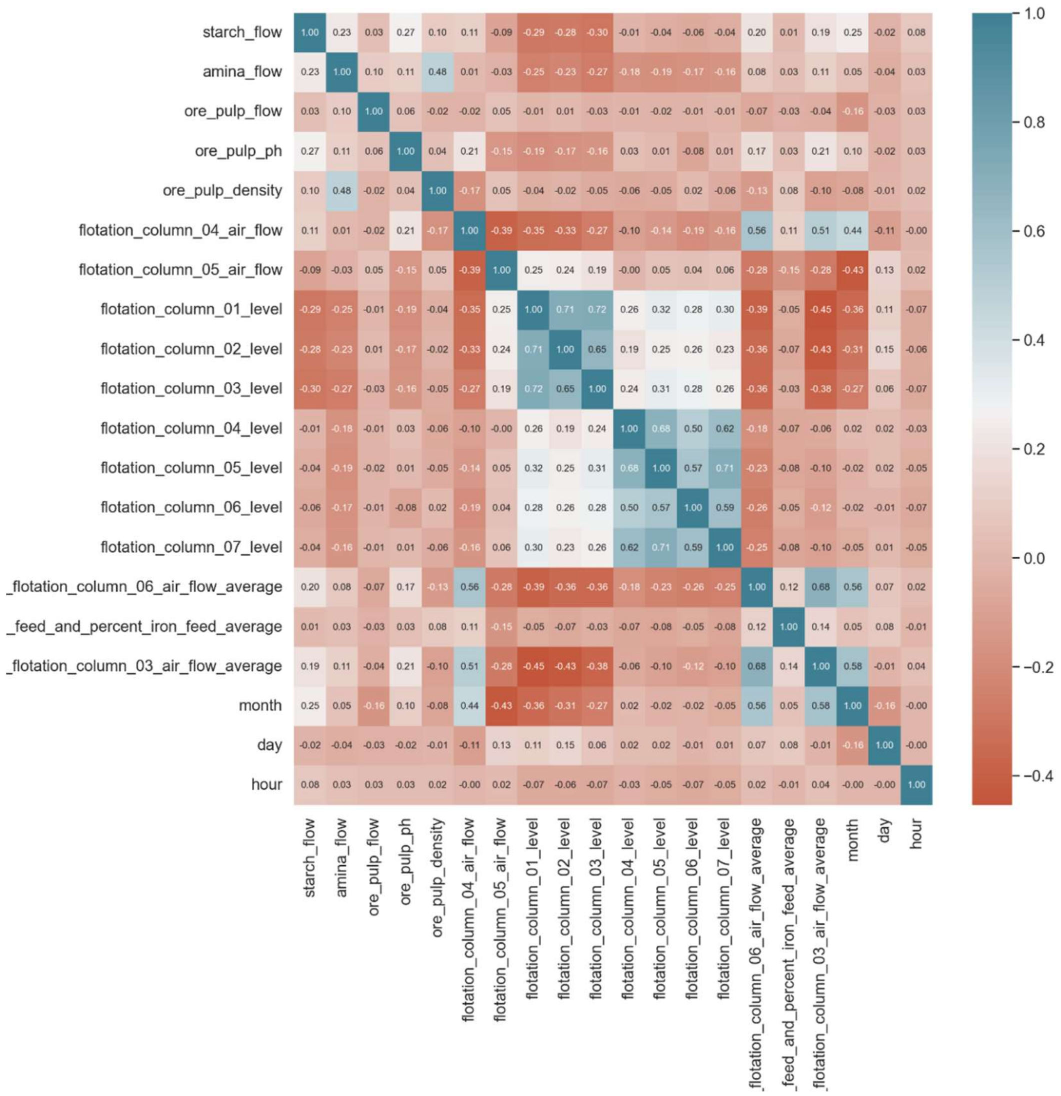

Figure 4. Correlation map of the data after incorporating seasonality.

2. XGB Regressor Model Construction

2.1. Introduction to eXtreme Gradient Boosting Regressor (XGBR)

Gradient boosting, a popular machine learning algorithm, has a highly effective and scalable implementation known as XGBR, or eXtreme Gradient Boosting. The remarkable performance of XGBR regression on a variety of predictive tasks has garnered a lot of attention. It is especially well-known for its speed and accuracy in handling complicated and sizable datasets. XGBR is designed to maximize memory consumption and computational efficiency. It is able to efficiently handle large datasets thanks to its tree construction algorithms and system optimization. XGBR's objective function is a combination of a loss function and a regularization term:

$$\mathrm{Obj}(\Theta)=\mathrm{L}(\Theta)+\Omega(\Theta) \tag{4}$$

Where, Θ represents the parameters of the models, L(Θ) is the loss function that measures how well the model's predictions match the actual data, and Ω(Θ) is the regularization term that penalizes the complexity of the model to avoid overfitting.

2.2. Training Data Preparation for XGBR

XGBR predicts one target variable at a time because it is a single-output model by design. Nevertheless, XGBR can still be used for multi-output regression tasks, whichinvolve predicting multiple continuous target variables. One XGBR model is trained internally for each target by the wrapper using scikit-learn's 'MultiOutputRegressor'. The wrapper abstracts away the process, enabling a more efficient implementation, but the formula for each target would remain the same as with the independent model's approach.

The data is split down to training and testing sets, with 70% allocated for training and 30% for testing. The test data is further divided into test data and validation data, each comprising 15% of the total data. After splitting, the data were scaled in the range [0,1] using Eq. (5):

$$z=\frac{x-\mu}{\sigma} \tag{5}$$

Where, z is the standardized value, x is the original feature value, μ is the mean of the feature across all data points, and σ is standard deviation of the feature across all data points.

2.3. Model Building and Training for XGB Regressor

XGBR model building and training entails careful data preparation, model type selection, parameter tuning, and assessment. Table 2 shows the parameters, their description and the values used to train this model. The depth of each tree is 20, and the learning rate used in the model training is 0.01. Different values for these parameters were analyzed for tuning purposes to check the model's accuracy, and the optimized values are presented in Table 2.

Table 2. Parameters for XGB Regressor.

Parameter	Value	Description
n_estimators	250	Number of boosting rounds (trees).
max_depth	20	Maximum depth of each tree.
learning_rate	0.01	Step size shrinkage used to prevent overfitting.
random_state	21	Seed for random number generator for reproducibility.
verbosity	0	Verbosity of output messages (0 means silent).
tree_method	hist	Tree construction algorithm (histogram-based).
gpu	cuda	Suggests GPU usage for acceleration (use 'gpu_hist' for actual GPU acceleration).

3. LSTM Model Construction

3.1. Introduction to Long-Short Term Memory Model (LSTM)

A particular kind of recurrent neural network (RNN) called a Long Short-Term Memory (LSTM) network was created to get around conventional RNNs' drawbacks in identifying long-range dependencies and learning from data sequences. LSTMs were first presented by Sepp Hochreiter and Jürgen Schmidhuber in 1997 [13]. Since then, they have been used extensively in many applications, including time series analysis, speech recognition, natural language processing, and more. The following are the main attributes of LSTMs:

Memory Cell: LSTMs have a memory cell that can store information over long sequences. The cell state can be seen as a conveyor belt that runs through the entire sequence, allowing information to be added or removed.

Three Gates: LSTMs have three gates, as shown in Figure 5.

- Forget Gate: Decides what information from the cell state should be thrown away.
- Input Gate: Updates the cell state with new information.
- Output Gate: Produces the output based on the updated cell state

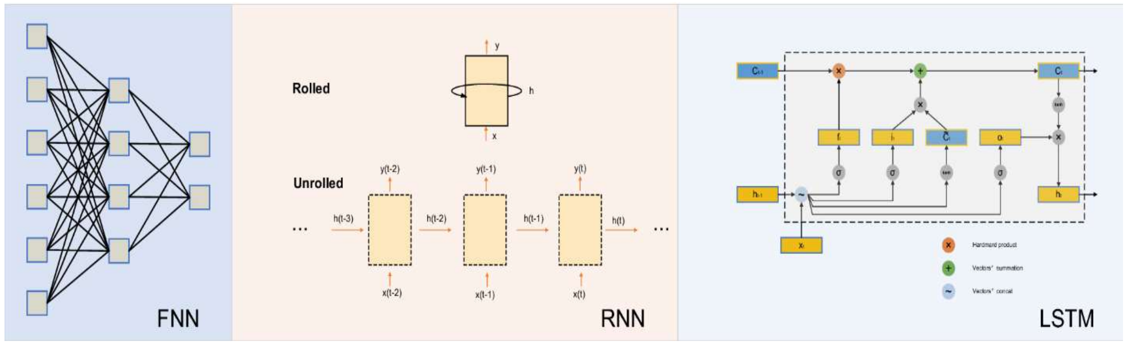

Figure 5. Basic architectures for FNN, RNN and LSTM [9].

Sigmoid and Tanh Activation Functions: Sigmoid activation functions are used in the gates to control the flow of information, with values ranging between 0 and 1. Tanh activation functions are used to regulate the values in the cell state, mapping them between -1 and 1.

3.2. Model Building and Training for LSTM

Forget Gate (f_t):

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \quad (6)$$

Where, σ is sigmoid function, W_f is weight matrix for the forget gate, $[h_{t-1}, x_t]$ is concatenation of the previous hidden state and the current input, and b_f is bias state.

Input Gate i_t and candidate Cell state $\widetilde{C_t}$:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (7)$$

$$\widetilde{C_t} = \tanh(W_c . [h_{t-1}, x_t] + b_c)$$

Where W_i and W_c are the weight matrices for the input gate and candidate cell state respectively, $b_i and$ b_c are the biases and tanh is the hyperbolic tangent activation function.

Cell State Update (C_t):

$$C_t = f_t * C_{t-1} + i_t * \widetilde{\boldsymbol{C_t}} \quad (8)$$

LSTM model and training parameters are presented in

Table 3.

Table 3. LSTM Model Configuration and Training Parameters.

Parameter/Step	Description / Value
Data Preprocessing	
Data Resampling	Median over each group of 6 entries
Feature Selection	Dropped '% Iron Concentrate', used '% Silica Concentrate'
Data Split	70% training, 30% testing
Further Data Split	Further split training into 85% training, 15% validation
Scaling	MinMaxScaler applied to features
Model Architecture	
Input Shape	(1, number of features)
First LSTM Layer	50 units, returns sequences
Second LSTM Layer	50 units
Output Layer	1 unit (Dense layer)
Training Parameters	
Optimizer	Adam
Loss Function	Mean Squared Error
Epochs	100
Batch Size	32
Validation Data	Used during training
Model Evaluation	
Test Data	Used to evaluate the model after training

4. Results and Discussion

4.1. For XGBR Regression

XGBR is trained on the parameters mentioned in the Table 2. The model is trained on 86,036 samples, comprising 70% of the data. The remaining 30% of data is divided into a test set and validation set comprising 9218 samples for each. After training the model, the feature importance plot is generated, as shown in Figure 6. Feature importance plot.

. The feature importance plot shows the importance of each feature in the modified data. The most essential feature in this plot is the ‘month’ feature, followed by the ‘day’ feature.

The 'month' feature's high significance indicates a significant seasonal component in the data. This indicates that the time of year significantly impacts the predicted target variables (iron concentrate and silica concentrate). In our data, different months are probably linked to different patterns or trends. The model is picking up on variations that occur in different months. This could be because the feed's percentage of iron and silica changes with time. In addition, the scatter plot was produced to investigate the predicted results further. Figure 7 shows the scatter plot for iron concentrate on the test data set. It shows a strong correlation between the true and predicted values, with an R-square value of 0.9. This suggests that the model fits the test data very well on the first target variable, iron concentrate.

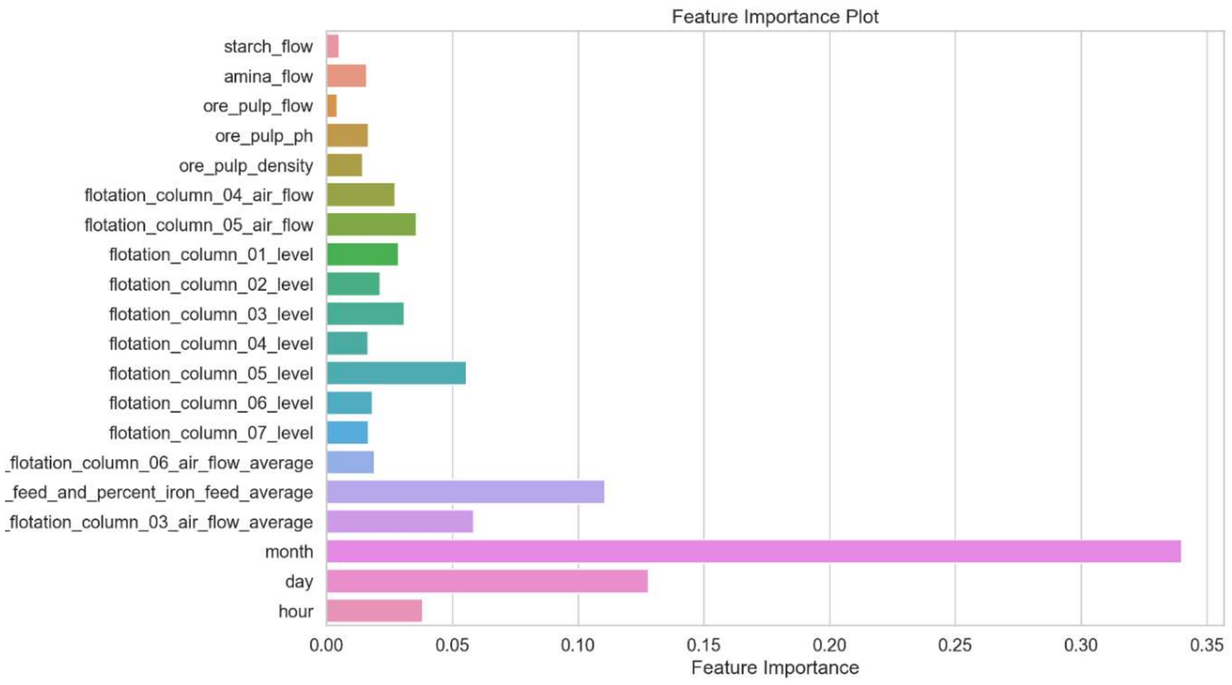

Figure 6. Feature importance plot.

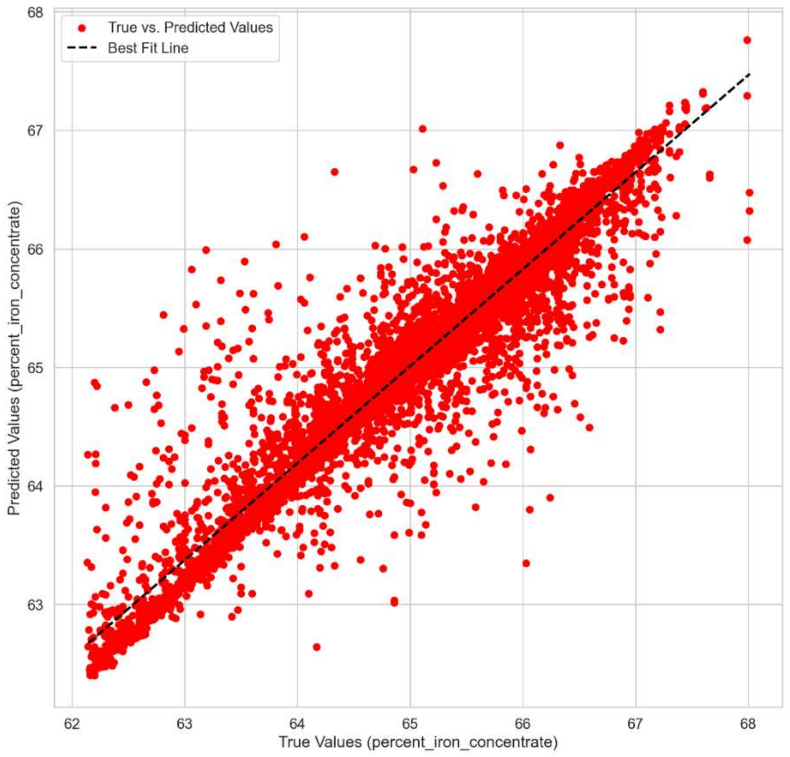

Figure 7. Scatter plot for Iron concentrate on test data set

Similarly, the scatter plot for the second target variable, silica concentrate, is shown in Figure 8. This also indicates a strong positive correlation between the true values and the predicted values of the 2nd target variable on the test data set, having an R-square score of 0.91.

The greater number of training samples helps enhance the model's interpretability, especially given the substantial number of training features (23 in our task). The model can be further tuned by changing the hyperparameters, including the number of estimators and the maximum depth of the estimator. In this study, the model is tuned to the different hyperparameter values, and the best values are chosen for the model after inspecting the outputs of the models.

As the test data was divided into test and validation data sets, the scatter plots for validation data set are also included. Validation is crucial for tuning the parameters of the model to improve the model's performance. In between training and testing, the validation set acts as a compromise. During the training phase, it helps make decisions about the model without jeopardizing the integrity of the test set, which is meant to remain an entirely impartial evaluation of the model's performance. A model that is accurate on known data and performs well on new, unseen data can be developed with the aid of this tripartite data division.

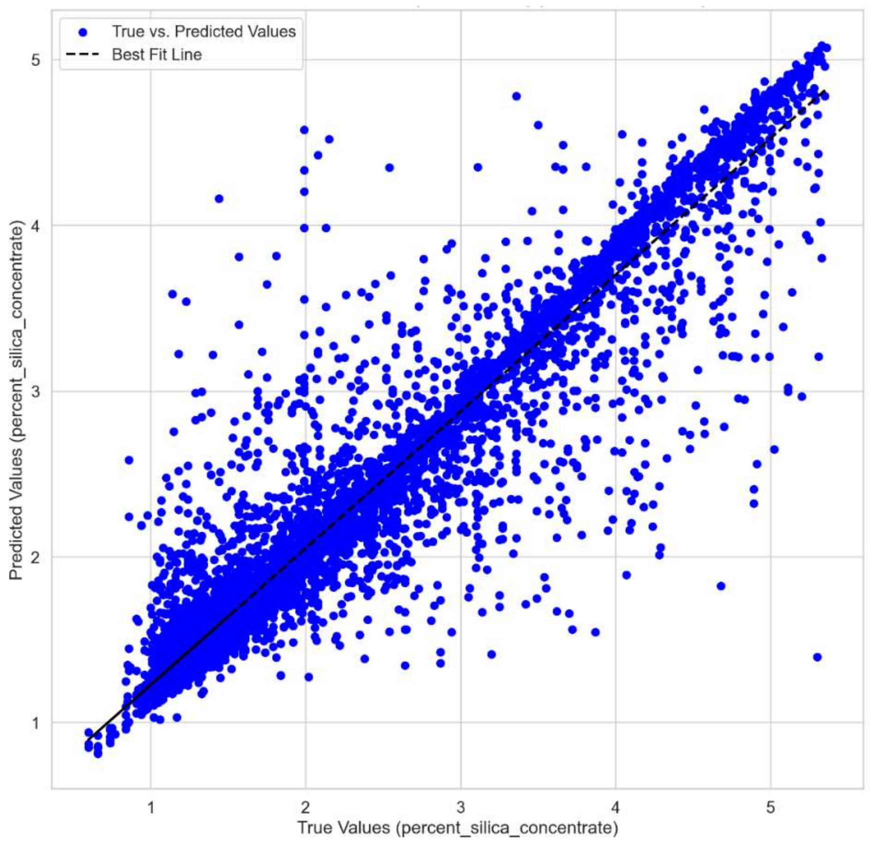

Figure 8. Scatter plot for silica concentrate on test data set.

Error! Reference source not found. Error! Reference source not found. shows the scatter plot of the validation set on the iron concentrate. This plot shows a strong positive correlation between the true and predicted values on the validation data set, having R-square score of 0.9. Similarly, Figure 9 depicts the scatterplot for the silica concentrate. Also, it shows a positive correlation between the true and the predicted values on the validation data set. Table 4 shows the model performance metrics for both the training set and validation set for both variables. Figure 9 represents the scatter plot for the second output variable, silica concentrate on the validation data set.

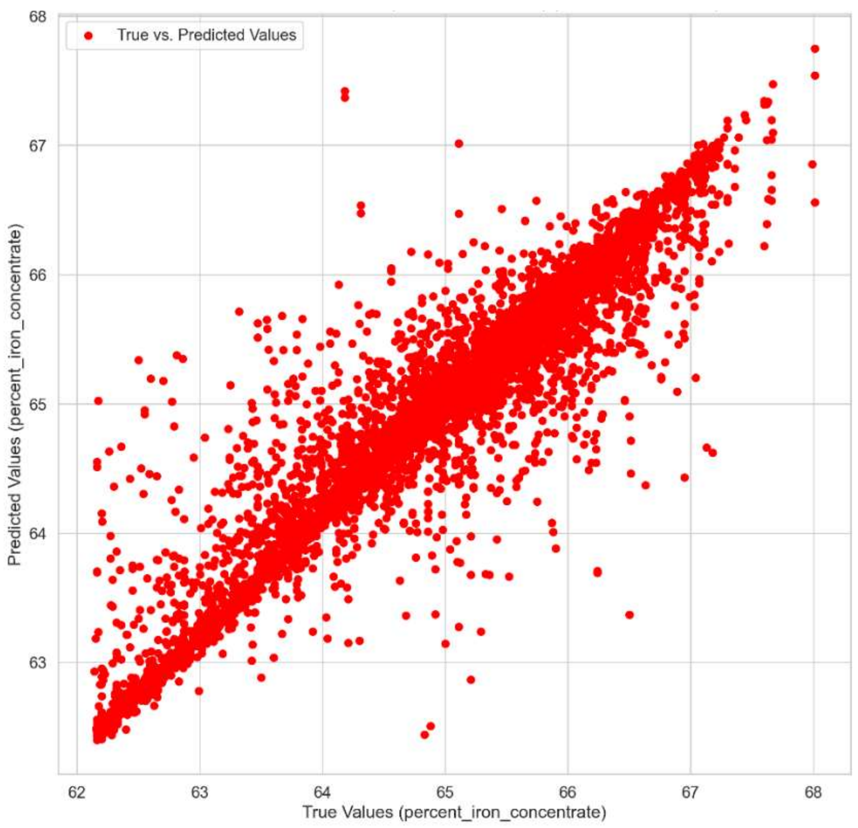

Figure 9. Scatter plot for iron concentrate on validation data set.

Table 4. Model performance metrics.

Model Performance Metrics	Training set		Validation Set	
	Iron Concentrate	Silica concentrate	Iron Concentrate	Silica Concentrate
R-square	0.904	0.904	0.894	0.894
MSE	0.1169	0.1169	0.128	0.128
RMSE	0.341	0.341	0.358	0.358

Such high R-square values demonstrate the model's ability to identify the underlying relationships and patterns in the data. This is especially important considering how difficult it is usually to predict the output depending on 23 features having temporal changes as well. The model's accuracy and effectiveness can lead to lower costs, higher-quality outputs, and improved decision-making. A major factor in attaining these outcomes was the XGBR algorithm's resilience, which is well-known for its effectiveness, scalability, and capacity to handle a range of data types. Key features such as tree pruning techniques, regularization to prevent overfitting, and the ability to handle missing values enhance the model's high predictive power.

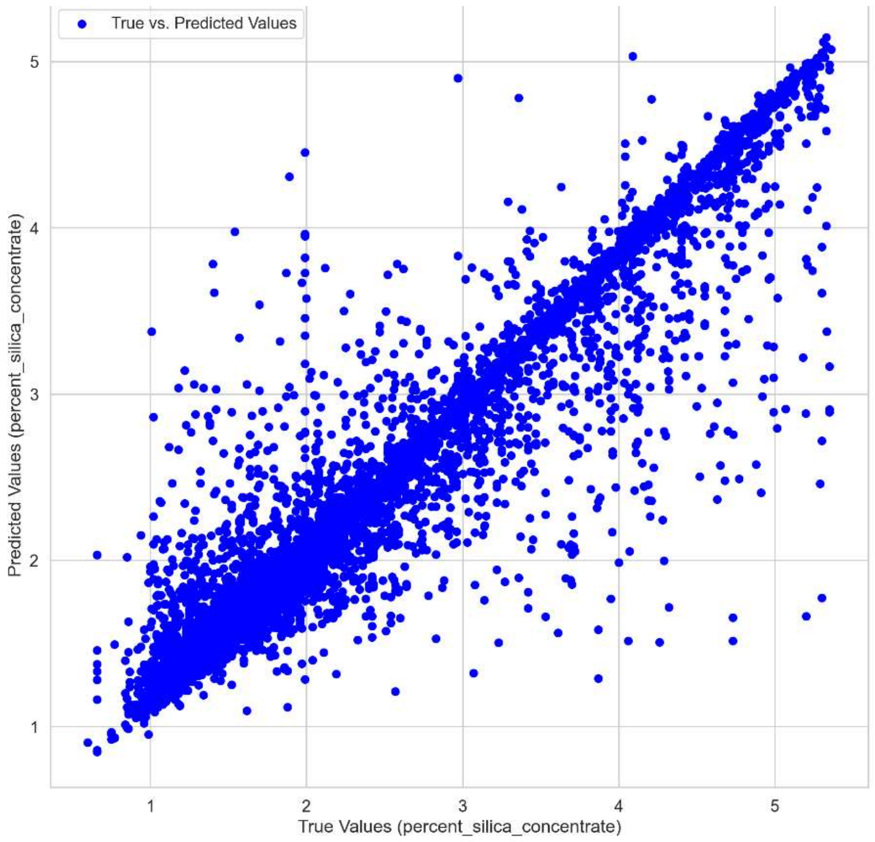

Figure 9. Scatter plot for silica concentrates on the validation data set.

4.2. For LSTM model

The LSTM model for both outputs was trained on the parameters described in

Table 3. Due to limited computation power and large dataset, it was not feasible to run this model on a greater number of layers. Consequently, the model was trained separately with 32 and 64 layers for each output.

A loss curve was generated for train and validation loss in model training as shown in

Figure 10. In the start, the validation loss is lower than the training loss, which is typical. However, after approximately 40 epochs, the validation loss started to rise, and then training loss suggested that the model was overfitting. In the start, it is still learning generalized patterns, but after 40 epochs, the model is not performing better on unseen data despite its improvements on training data, as evidenced by the widening gap where the validation loss surpasses the training loss. Similarly, the model shows the same trend for the other output variable, and even increasing the number of layers did not significantly improve the model's ability to capture patterns and relationships in the data.

Scatterplots for the test set and validation set are shown in Figure 11. It can be seen that the spread of the data is much greater in the validation set and in the test set. Also, the R-square value of test and validation set is 0.574 and 0.566, respectively.

While these values indicate that the model captures some variance in the target variable, it also shows where there is still space for development. This score suggests that the model only explains approximately 57.4% of the variance in our target, leaving a sizable portion unaccounted for. This observation, along with the loss curve analysis, indicates that further optimization could improve our current model's predictive accuracy and generalization ability, despite it captures some underlying patterns.

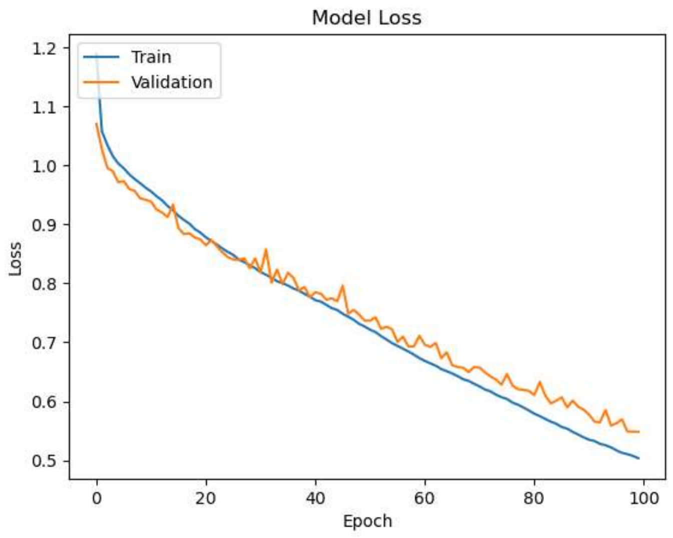

Figure 10. Train and Validation Loss in model training for Silica Concentrate.

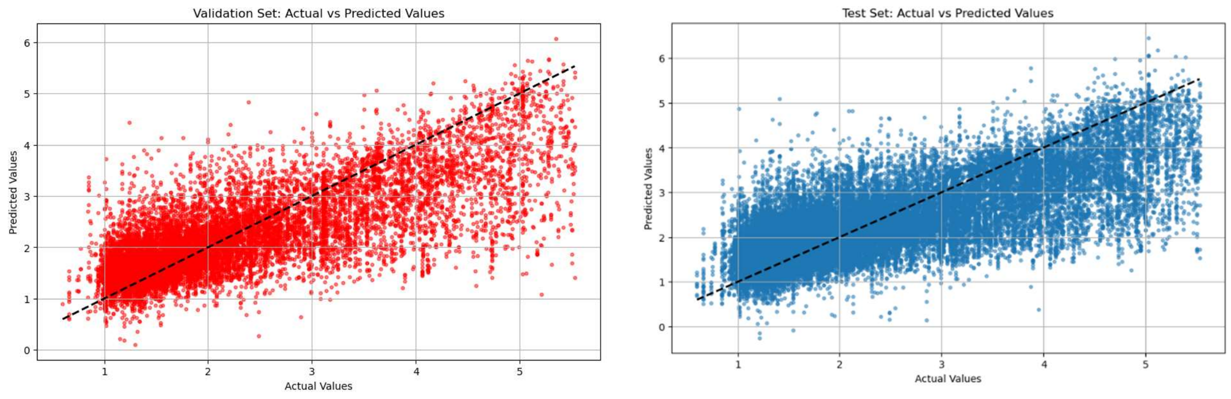

Figure 11. Scatter plot for LSTM model on test and Validation data set for Silica Concentrate.

Similarly, LSTM model is applied to the other output variable and scatter plots are shown in Figure 12. The model could also not capture the patterns and relations in the other output variables.

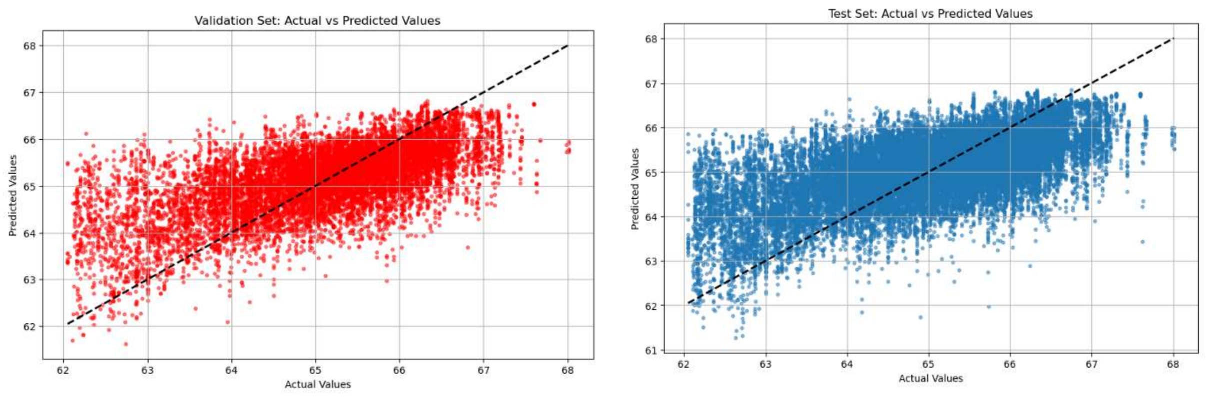

Figure 12. Scatter plot for LSTM model on test and Validation data set for Iron Concentrate.

5. Conclusion

This project has effectively illustrated how machine learning can significantly improve mining processes' efficiency, especially in flotation plants. This study has produced insightful predictions on concentrate purities for iron and waste silica by utilizing two sophisticated predictive techniques: Gradient Boost Regressor (XGBR) and Long Short-Term Memory (LSTM) networks. The use of an extensive dataset—23 variables total—that was gathered every two minutes for a period of six months highlights the stability and dependability of our findings.

XGBR performs better than LSTM in predicting the quality of iron and silica concentrates on both test and validation datasets, according to a comparative evaluation of the two models. This discovery is crucial because it not only confirms that XGBR is efficient at handling the kind of large-scale, complicated datasets that are commonly used in mining operations, but also highlights its superior performance over LSTM in this particular application.

These results have effects that go beyond the specific parameters of this project. Because of XGBR's superior predictive capabilities, flotation plants could benefit from automate control processes, potentially revolutionizing the sector by increasing productivity, cutting costs, and having a minimal negative environmental impact. Furthermore, this project's success in applying both conventional and deep learning machine learning techniques highlights the usefulness and efficiency of these methods in mineral processing engineering.

The study's findings support the mining industry's continued investigation and adoption of machine learning techniques. Subsequent studies might concentrate on improving these models, examining their scalability, and evaluating how well they work in other crucial areas of mineral processing. The ultimate objective would be to fully utilize these technologies in order to achieve more profitable, sustainable, and efficient mining operations .

6. Recommendation

Due to limited computation resources, the LSTM model was not fully tested with changing parameters. Further work can be done to explore the LSTM model's potential in predicting the quality of a floatation process. XGBR model shows significant predictive capabilities in predicting the floatation process quality in silica and iron concentrate.

7. References

[1] Jahedsaravani, A., M., Marhaban, M., Massinaei, M., Saripan, and S.M., Noor, *Froth-based modeling and control of a batch flotation process.* International Journal of Mineral Processing, 2016. **146**, pp. 90-96.

[2] Wills, B.A., and J., Finch, Wills' Mineral Processing Technology: An Introduction to the Practical Aspects of Ore Treatment and Mineral Recovery. 2015.

[3] Nakhaei, F., M., Mosavi, A., Sam, and Y., Vaghei, *Recovery and grade accurate prediction of pilot plant flotation column concentrate: Neural network and statistical techniques.* International Journal of Mineral Processing, 2012. **110**, pp. 140-154.

[4] Vieira, S., J., Sousa, and F., Durão, *Fuzzy modelling strategies applied to a column flotation process.* Minerals Engineering, 2005. **18**(7), pp. 725-729.

[5] McCoy, J.T., and L., Auret, *Machine learning applications in minerals processing: A review.* Minerals Engineering, 2019. **132**, pp. 95-109.

[6] Chelgani, S.C., B., Shahbazi, and B., Rezai, *Estimation of froth flotation recovery and collision probability based on operational parameters using an artificial neural network.* International Journal of Minerals, Metallurgy, and Materials, 2010. **17**, pp. 526-534.

[7] Yang, C., H., Ren, C., Xu, and W., Gui, *Soft sensor of key index for flotation process based on sparse multiple kernels least squares support vector machines.* The Chinese Journal of Nonferrous Metals, 2011. **21**(12), pp. 3149-3154.

[8] Shahbazi, B., S.C., Chelgani, and S., Matin, *Prediction of froth flotation responses based on various conditioning parameters by random forest method.* Colloids and Surfaces A: Physicochemical and Engineering Aspects, 2017. **529**, pp. 936-941.

[9] Pu, Y., A., Szmigiel, and D.B., Apel, *Purities prediction in a manufacturing froth flotation plant: The deep learning techniques.* Neural Computing and Applications, 2020. **32**(17), pp. 13639-13649.

[10] LeCun, Y., Y., Bengio, and G., Hinton, *Deep learning.* Nature, **521**(7553), pp. 436-444.

[11] Miriyala, S.S., and K., Mitra, *Deep learning based system identification of industrial integrated grinding circuits.* Powder Technology, 2020. **360**, pp. 921-936.

[12] Oliveira, E.M., *Quality prediction in a mining process.* Kaggle, 2017.

[13] Hochreiter, S., and J., Schmidhuber, *Long short-term memory*. Neural Computation, 1997. **9**(8), pp. 1735-1780.