Introduction To Machine Learning

B.Tech. CSE (II YEAR – III SEMESTER)
(2024-2025)

Submitted By :
Achhuta Nand Jha
Roll no. : 02

Submitted To :
Shivanshu Upadhyay Sir

Section: CH

Alpha Thalassemia Classifier: A Machine Learning Approach

Background:

Alpha thalassemia is a genetic blood disorder that impairs the production of hemoglobin, a critical component of red blood cells responsible for oxygen transport in the body. This disorder is primarily prevalent in regions such as Southeast Asia, the Mediterranean, and parts of Africa. Early detection of alpha thalassemia can significantly improve patient outcomes by enabling timely medical intervention and management. Given the complexity of the disorder and its genetic underpinnings, machine learning offers a promising avenue for enhancing diagnostic accuracy.

Objectives:

The primary goal of this project is to develop a robust machine learning model capable of classifying alpha thalassemia based on hematological and genetic data. The specific objectives include:

1. **Developing a Classification Model:** Using clinical and genetic data to train a model that can accurately distinguish between alpha thalassemia and normal cases.
2. **Early Diagnosis:** Facilitating early detection of the disorder, which can guide treatment plans and improve patient care.

3. **Explainable Results:** Providing insights into the key features contributing to the classification, allowing clinicians to understand the underlying factors behind the diagnosis.

Methodology:

The approach consists of several stages, from data preprocessing to model evaluation, using advanced machine learning techniques.

1. **Data Collection:**
   - The dataset used in this project is sourced from clinical data, including various hematological measurements (such as RBC count, MCH, and others), and genetic factors relevant to alpha thalassemia. o The dataset is preprocessed to handle missing values, normalize numerical data, and encode categorical features.
2. **Data Preprocessing:**
   - **Handling Missing Values:** Missing values in the rbc (red blood cell count) and mch (mean corpuscular hemoglobin) columns are filled with the mean value for each phenotype group (normal or alpha thalassemia).
3. **Outlier Detection and Handling:** The dataset is scanned for outliers using the interquartile range (IQR) method.
   Outliers are capped to the upper and lower bounds of the IQR to prevent their influence on the model's performance.

4. **Categorical Encoding:** Categorical variables such as sex and phenotype are one-hot encoded to convert them into a numerical format that the machine learning algorithms can process.

5. **Model Development:**

- **Data Splitting:** The dataset is divided into training and test sets using train_test_split to ensure that the distribution of target classes (normal vs. alpha thalassemia) is maintained across the splits.

- **SMOTE (Synthetic Minority Over-sampling Technique):** Given the potential class imbalance in the dataset, SMOTE is applied to oversample the minority class (alpha thalassemia) in the training data. This helps in improving the model's ability to detect alpha thalassemia cases.

- **Normalization:** The features are normalized to scale the input data, ensuring that the model does not become biased towards any particular feature due to its scale.

Model Training:

- **XGBoost Classifier:** The XGBoost algorithm, known for its efficiency and performance in classification tasks, is employed to train the model. It is configured with various hyperparameters, including learning_rate, max_depth, subsample, colsample_bytree, and regularization terms (reg_lambda, reg_alpha).

- **Hyperparameter Tuning with Optuna:** Optuna, an optimization library, is used to find the best hyperparameters for the model. The run function defines a search space for hyperparameters, and the optuna library performs optimization through trials to find the optimal values.

- **Early Stopping:** Early stopping is applied during model training to prevent overfitting by halting the training process if the performance on the validation set does not improve after a certain number of rounds.

Model Evaluation:

- **F1 Score:** The model's performance is evaluated using the F1 score, a metric that balances precision and recall, especially important when dealing with class imbalances.

- **ROC Curve and AUC:** The receiver operating characteristic (ROC) curve is plotted to evaluate the true positive rate (recall) against the false positive rate. The area under the curve (AUC) provides a summary of the model's ability to discriminate between the classes.

- **Confusion Matrix:** The confusion matrix is displayed to analyze the model's ability to correctly classify alpha thalassemia and normal cases.

- **Log Loss:** Log loss is used to evaluate the probability-based performance of the model, indicating how well the model predicts the probabilities of each class.

- **Model Deployment:**

- **Saving the Model:** The trained model is saved using joblib for later use in real-world applications, allowing clinicians to use it for diagnosis without retraining the model.

- **Classification Report:** The classification report is generated, summarizing the precision, recall, F1 score, and support for each class.

Data Preprocessing and Feature Engineering
The provided code starts by loading a CSV dataset (alphanorm_anjha.csv) and performing several important preprocessing steps to prepare the data for modeling. Specifically:

1. **Handling Missing Values**:
   - The missing values in the rbc and mch columns are filled with the mean values of these columns, grouped by the 'phenotype' column, ensuring that the imputation respects the natural groupings of the data.

2. **Outlier Detection and Capping**:
   - The code calculates the Interquartile Range (IQR) for numerical columns, identifying outliers as values that fall outside the bounds of 1.5 times the IQR. These outliers are then capped to the upper and lower bounds, ensuring that extreme values don't unduly influence the model.

3. **Feature Encoding**:
   - The categorical variables, such as 'sex' and 'phenotype', are converted to a category data type for efficient handling.
   - The target variable, 'phenotype', is transformed into a binary form, where 'alpha carrier' becomes 1 and the rest is 0.

    - The categorical variables are one-hot encoded, transforming them into a format suitable for machine learning algorithms.

Model Training and Hyperparameter Optimization Once the data is cleaned and transformed, the code proceeds with model training using the **XGBoost** classifier. XGBoost is a powerful gradient boosting algorithm commonly used for classification tasks.

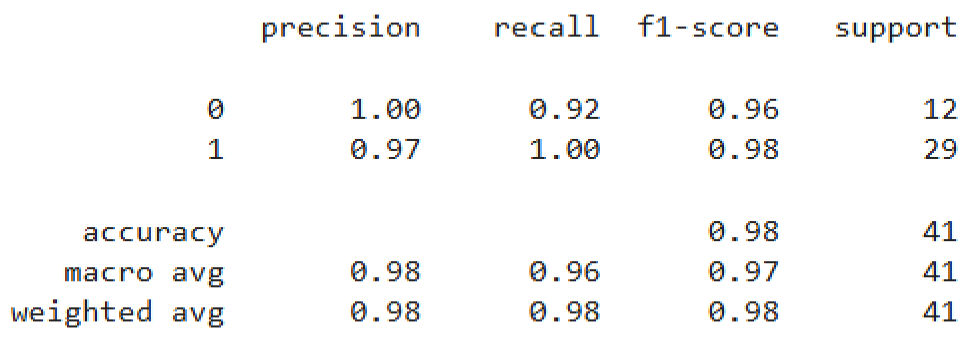
- **SMOTE (Synthetic Minority Over-sampling Technique)** is used to balance the dataset by oversampling the minority class (alpha carriers in this case), ensuring the model does not suffer from class imbalance.

- **Optuna**, a hyperparameter optimization framework, is used to tune several hyperparameters of the XGBoost model. The key hyperparameters optimized include: ○ learning_rate: Controls the contribution of each tree in the model. ○ reg_lambda and reg_alpha: Regularization parameters that help prevent overfitting.
    - subsample and colsample_bytree: Parameters that control the fraction of data and features used in each tree.
    - max_depth: Controls the depth of each tree. The hyperparameter search is performed using the Optuna library, which tries different combinations of parameters to maximize the F1 score, with early stopping based on the evaluation of the model on the test data.

Model Evaluation and Metrics

Once the model is trained, various performance metrics are computed to evaluate its performance:

1. **Classification Report**:
    - This includes precision, recall, and F1-score, which are crucial for evaluating the classifier's performance in imbalanced datasets like this one, where the positive class (alpha carriers) is likely underrepresented.
2. **ROC Curve**:
    - The **ROC (Receiver Operating Characteristic) curve** is plotted to assess the model's ability to discriminate between classes. The area under the curve (AUC) is computed as a summary measure of model performance.
3. **Log Loss and Classification Error**:
    - The code plots the **log loss** and **classification error** for both the training and testing datasets across epochs to assess if the model is overfitting (i.e., performing well on the training set but poorly on the test set).
4. **Confusion Matrix**:
    - A confusion matrix is generated to visualize the true positives, false positives, true negatives, and false negatives, providing a clear picture of where the model is making errors.



```
              precision    recall  f1-score   support

           0       1.00      0.92      0.96        12
           1       0.97      1.00      0.98        29

    accuracy                           0.98        41
   macro avg       0.98      0.96      0.97        41
weighted avg       0.98      0.98      0.98        41
```

1. Precision, Recall, F1-Score, Support:

- These metrics are calculated for each class (0 and 1) and are summarized for the whole model at the end.

- **Precision** measures the proportion of positive predictions that are actually correct. In simple terms, it's how many of the predicted positive labels are true positives.
-
- **Recall** (also called Sensitivity or True Positive Rate) measures the proportion of actual positives that are correctly identified by the model. It answers how many actual positives the model successfully predicted.
-
- **F1-Score** is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially useful when you have an uneven class distribution.
-
- **Support** is the number of true instances for each class in the dataset. It shows how many examples the model had to classify for each class.

2. Class Breakdown (0 and 1):
Class 0 (Label = 0):

- **Precision = 1.00:** This means that for the 12 instances where the model predicted class 0, all 12 were actually class 0 (perfect precision).

- **Recall = 0.92:** The model correctly identified 92% of all actual class 0 instances (11 out of 12). There was 1 false negative (a class 0 instance that was predicted as class 1).

- **F1-Score = 0.96:** This is the harmonic mean of precision and recall for class 0. Given that precision is high (1.00), the F1-score is very close to recall.

- **Support = 12:** There were 12 instances of class 0 in the dataset.

Class 1 (Label = 1):

- **Precision = 0.97:** This means that 97% of the instances the model predicted as class 1 were actually class 1.

- **Recall = 1.00:** The model correctly identified all of the actual class 1 instances (29 out of 29). There were no false negatives for class 1.

- **F1-Score = 0.98:** The F1-score for class 1 is high because both precision and recall are good.

- **Support = 29:** There were 29 instances of class 1 in the dataset.

3. Overall Metrics:

Accuracy:

- **Accuracy = 0.98 (98%)**: This is the overall percentage of correct predictions across both classes. It tells you that 98% of the total predictions were correct.

Macro Average:

- **Macro avg precision = 0.98:** This is the average precision across both classes, calculated without considering class imbalance. It's the mean of precision for class 0 and class 1.
- **Macro avg recall = 0.96:** This is the average recall across both classes.
- **Macro avg F1-Score = 0.97:** This is the average F1-score across both classes.

Weighted Average:

- **Weighted avg precision = 0.98:** The weighted average precision takes into account the support (number of instances) for each class. Since class 1 has more instances (29 vs. 12 for class 0), this metric reflects the precision across both classes, weighted by the class distribution.

- **Weighted avg recall = 0.98:** Similarly, this reflects the weighted average recall.
- **Weighted avg F1-Score = 0.98:** The weighted F1-score gives the overall balance between precision and recall, accounting for the class distribution.

Summary:

- The model is performing very well with an overall accuracy of 98%.
- It does exceptionally well in predicting class 1, with perfect recall (100%) and a very high precision of 97%.
- For class 0, precision is perfect, but recall is slightly lower at 92%, meaning the model missed one class 0 instance.
- The F1-scores are high for both classes, indicating a good balance between precision and recall.

Conclusion

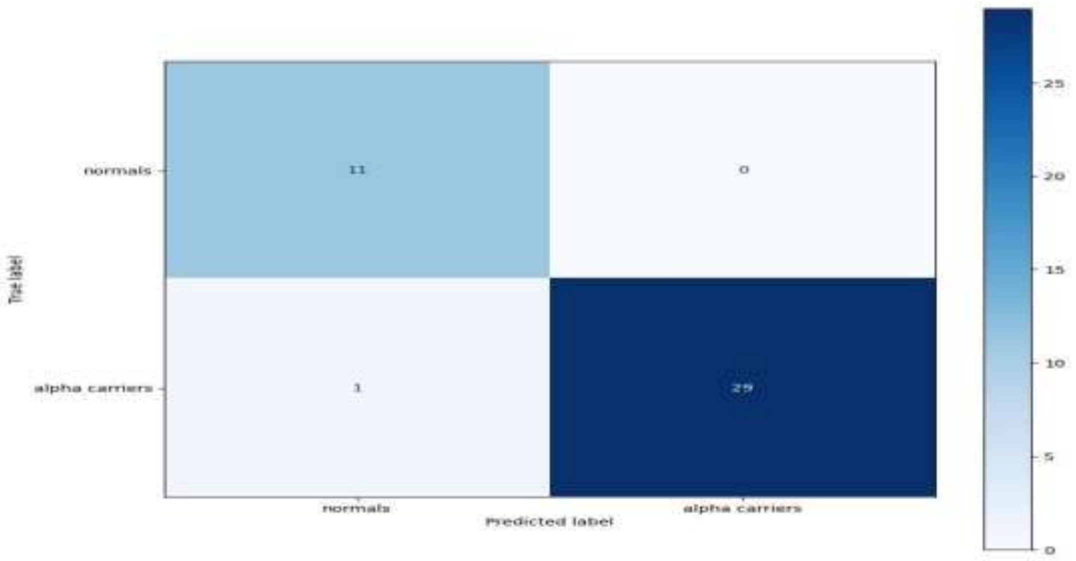
In summary, the XGBoost model, after hyperparameter tuning with Optuna, appears to provide valuable predictions on the task of identifying alpha thalassemia carriers. Key metrics like F1 score, ROC AUC, and the confusion matrix help in understanding the model's performance, especially considering the potential class imbalance. The model is evaluated using different metrics, including:

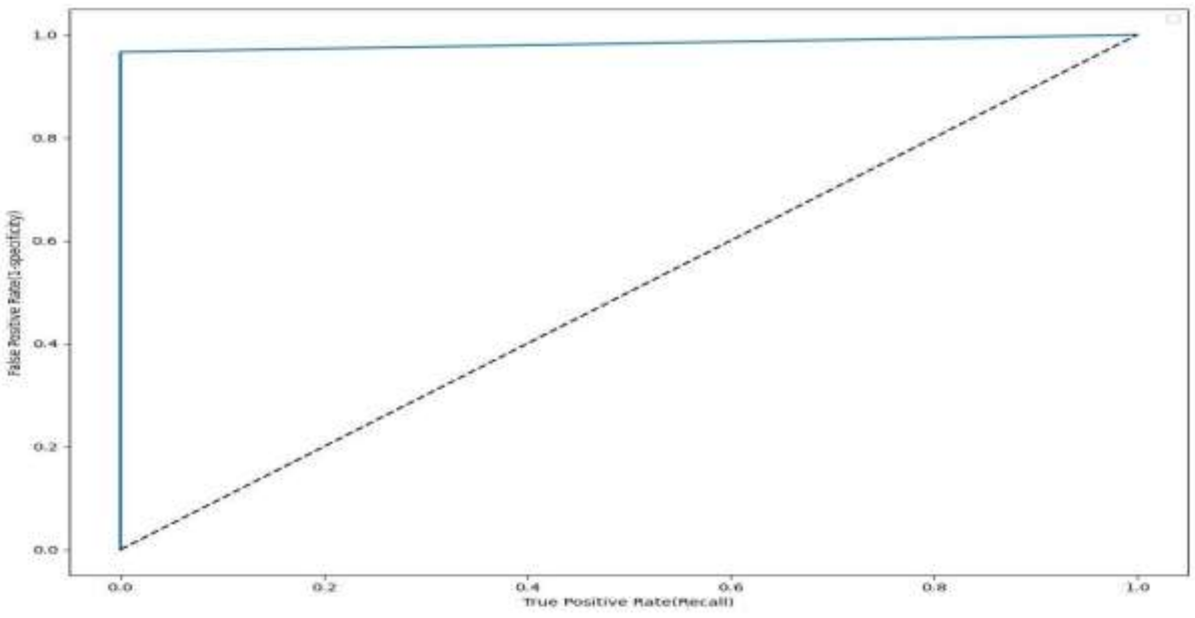
- **Precision, Recall, and F1-Score**: These are crucial when dealing with imbalanced datasets, as they ensure the model's ability to identify the minority class (alpha carriers).

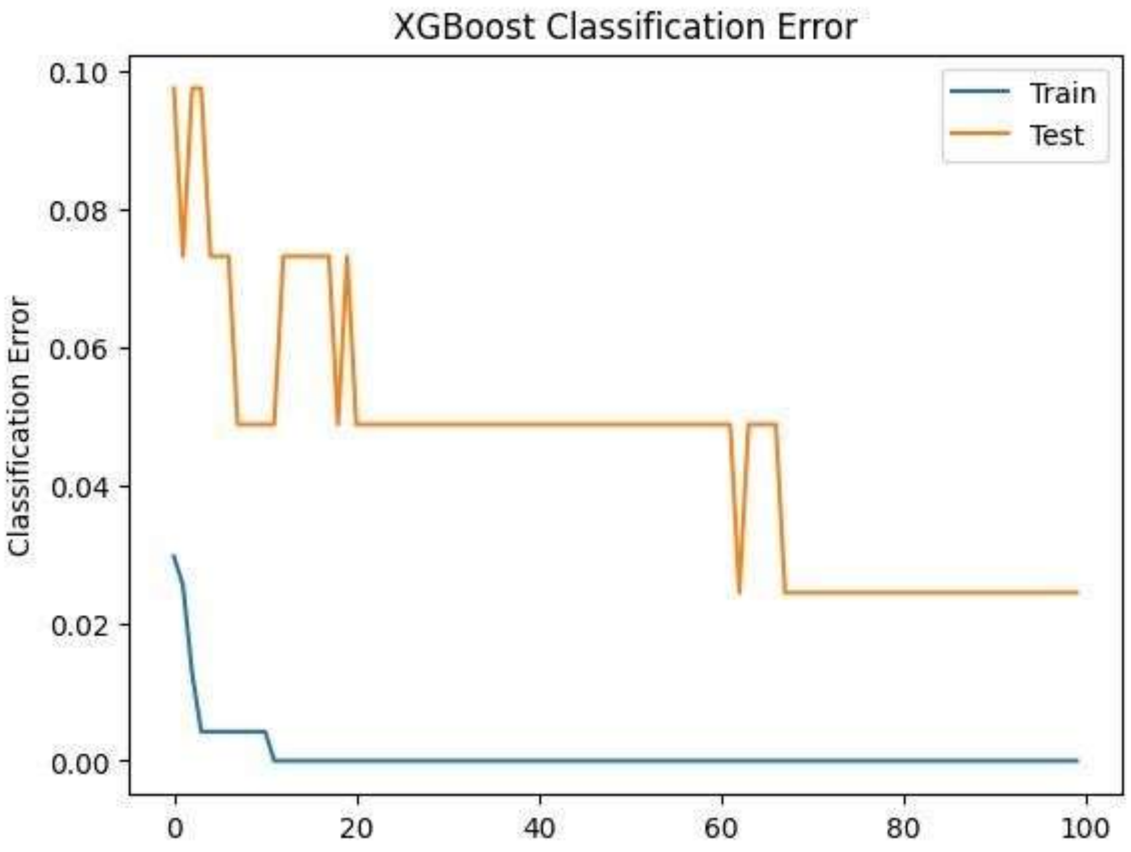
- **ROC Curve and AUC**: The ROC curve provides insights into the trade-off between true positive rate and false positive rate, and the AUC quantifies the model's discriminative ability.

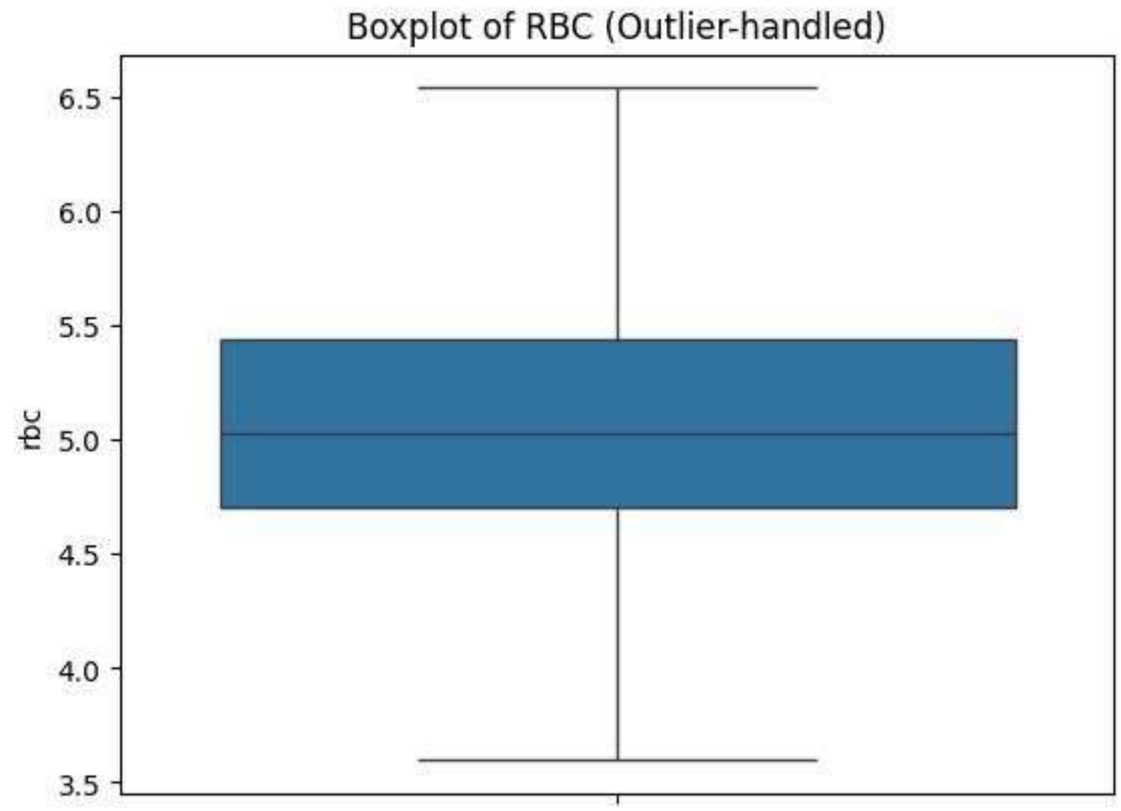
- **Log Loss and Classification Error**: These metrics show how well the model's predictions align with actual outcomes, and the plots can help identify any overfitting.

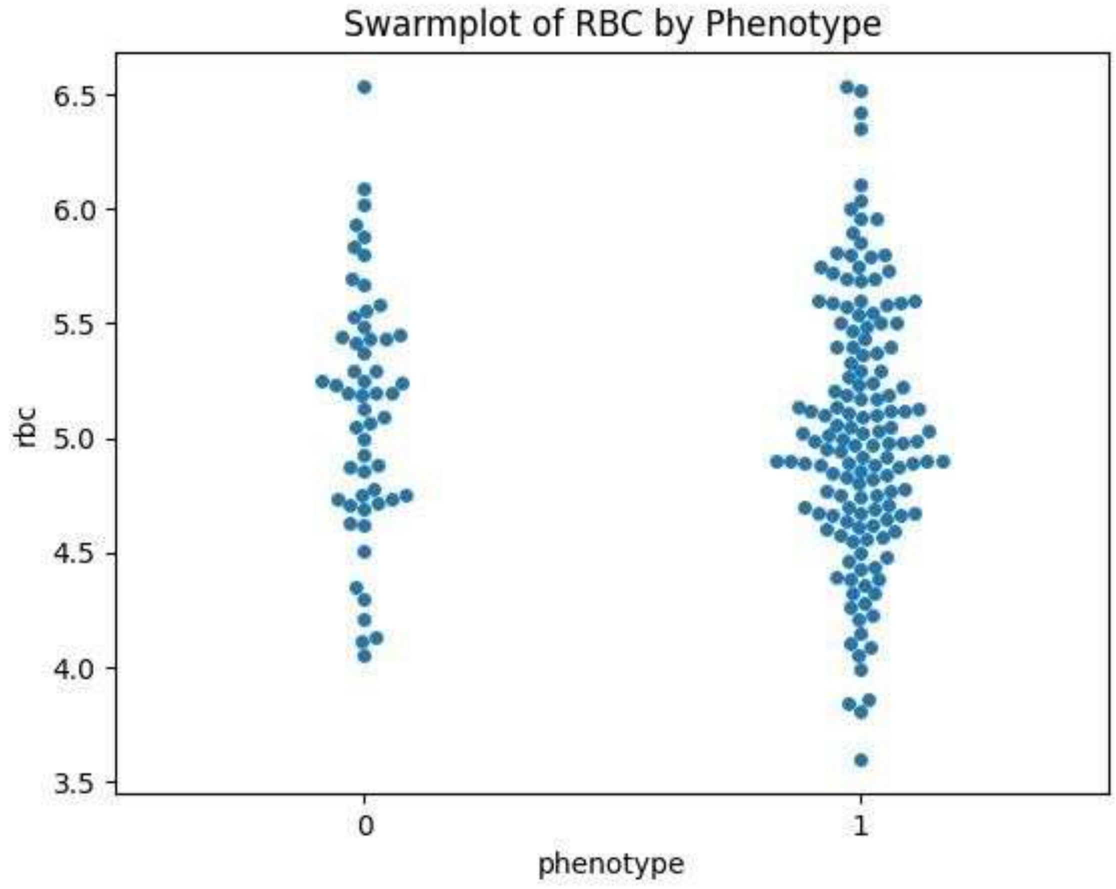
Data Insigths:

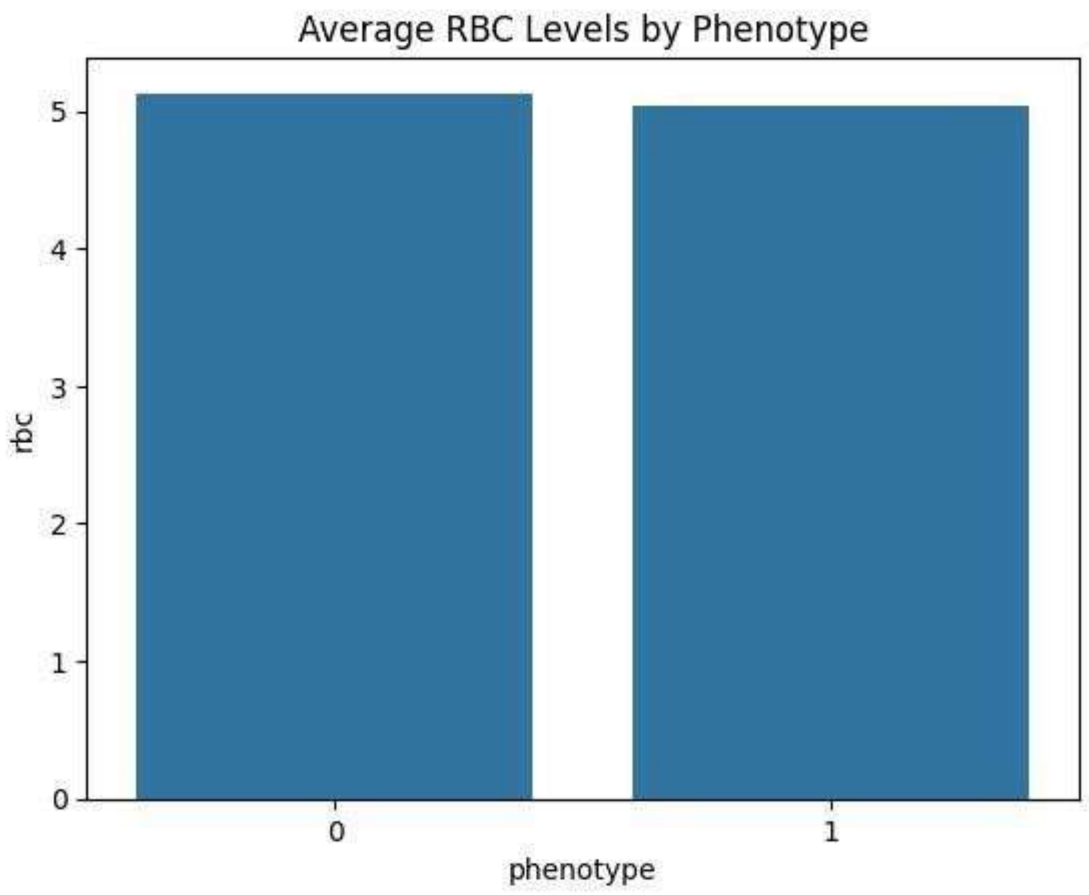


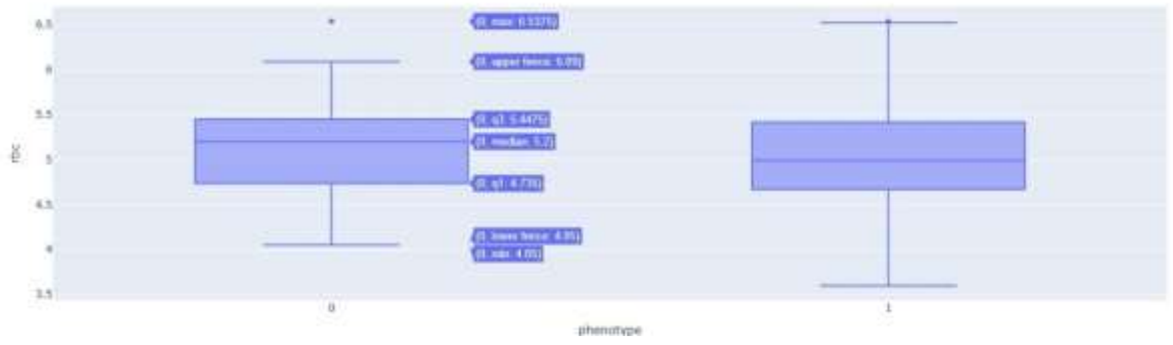


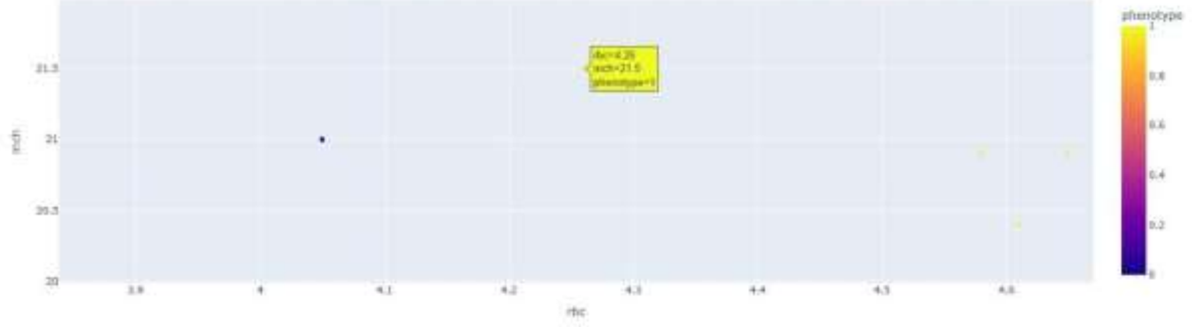

XGBoost Classification Error
Classification Error
Train
Test
0.10
0.08
0.06
0.04
0.02
0.00
0
20
40
60
80
100

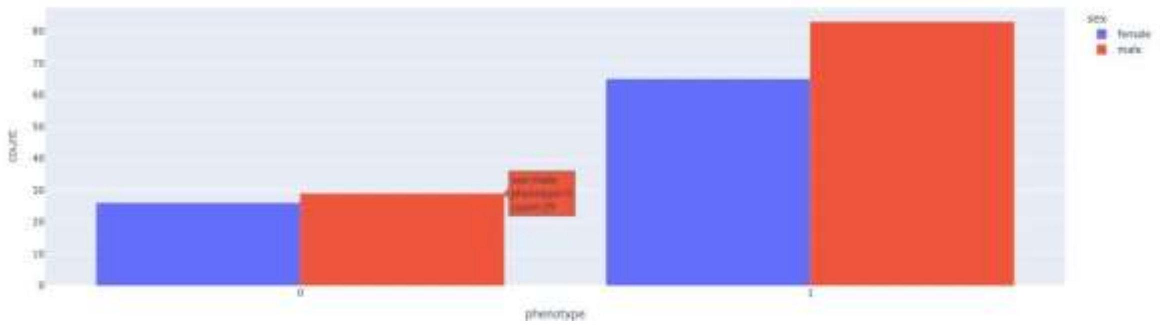

Boxplot of RBC (Outlier-handled)
rbc
6.5
6.0
5.5
5.0
4.5
4.0
3.5


Swarmplot of RBC by Phenotype
rbc
6.5
6.0
5.5
5.0
4.5
4.0
3.5
0
1
phenotype


Average RBC Levels by Phenotype
rbc
0
1
2
3
4
5
0
1
phenotype


(0, max: 6.5375)
(0, upper fence: 6.09)
(0, q3: 5.4475)
(0, median: 5.2)
(0, q1: 4.735)
(0, lower fence: 4.05)
(0, min: 4.05)
rbc
3.5
4
4.5
5
5.5
6
6.5
0
1
phenotype


rbc=4.29
mch=21.5
phenotype=1
mch
20
20.5
21
21.5
3.9
4
4.1
4.2
4.3
4.4
4.5
4.6
rbc
phenotype
0
0.2
0.4
0.6
0.8
1



In conclusion, the model seems well-calibrated and capable of distinguishing between alpha thalassemia carriers and non-carriers, which is the primary objective of the analysis. Future improvements could focus on exploring other algorithms, refining the preprocessing steps, or investigating further feature engineering techniques to enhance predictive performance.