**BAIC 2025 - BUBT AI Conquest**

# Liver Cirrhosis Survival Prediction

Multi-Class Classification


| | |
|---|---|
| **Team Name:** | BRACU_Cortex |
| **Team Members:** | Mahir Tajwar Rahman |
| | Sakib Raihan |
| **Department:** | Computer Science & Engineering |
| **Institution:** | BRAC University |


November 27-28, 2025

# Contents

# 1 Introduction

Liver cirrhosis is a chronic liver disease characterized by the progressive replacement of healthy liver tissue with scar tissue, ultimately leading to liver failure. Early and accurate prediction of patient survival outcomes is crucial for clinical decision-making, treatment planning, and resource allocation in healthcare settings. This report presents our comprehensive machine learning solution developed during the BAIC 2025 (BUBT AI Conquest) 24-hour competition for predicting liver cirrhosis patient survival status.

The primary objective of this competition is to build a multi-class classification model that predicts patient survival status into three categories: Status C (Censored - patient alive at end of study), Status CL (Censored due to Liver transplant), and Status D (Deceased). The evaluation metric for this competition is Log Loss (also known as cross-entropy loss), where lower values indicate better model performance. Log Loss heavily penalizes confident but incorrect predictions, making probability calibration a critical aspect of our solution.

Our approach achieved a cross-validation Log Loss of **0.35214**, utilizing an XGBoost classifier with carefully tuned hyperparameters obtained through Optuna's Bayesian optimization framework. We compared XGBoost against Random Forest and LightGBM, with XGBoost emerging as the best performer. We employed a robust cross-validation strategy consisting of 10-fold Stratified K-Fold validation repeated across 5 different random seeds, resulting in an ensemble of 50 models whose predictions were averaged to reduce variance and improve generalization to unseen data.

# 2 Exploratory Data Analysis (EDA)

## 2.1 Dataset Overview

The dataset provided for this competition consists of clinical and laboratory measurements collected from patients diagnosed with liver cirrhosis. The training set contains 15,000 patient records with 19 features including the target variable, while the test set contains 10,000 records with 19 features (excluding the target). The features can be broadly categorized into demographic information (age, sex), treatment information (drug type), clinical observations (presence of ascites, hepatomegaly, spiders, edema), and laboratory measurements (bilirubin, albumin, copper, alkaline phosphatase, SGOT, triglycerides, platelets, prothrombin time, cholesterol).

Table 1: Dataset Statistics Summary

| Property | Training Set | Test Set |
|---|---|---|
| Number of Samples | 15,000 | 10,000 |
| Number of Features | 19 | 19 |
| Numerical Features | 13 | 13 |
| Categorical Features | 6 | 6 |

## 2.2 Target Distribution Analysis

Understanding the distribution of the target variable is fundamental to developing an effective classification strategy. Our analysis revealed a significant class imbalance in the dataset, which has important implications for model training and evaluation.

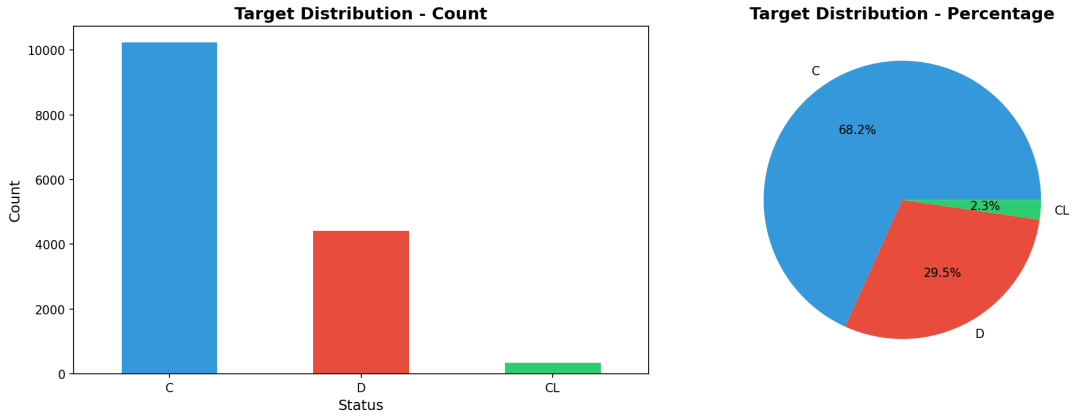


Figure 1: Distribution of target classes showing significant imbalance between the three survival status categories.

As illustrated in Figure 1, the majority of patients (approximately 66.8%) belong to Status C (Censored), indicating they were still alive at the conclusion of the study period. Status D (Deceased) represents about 30.9% of the dataset, comprising patients who died during the study. Most notably, Status CL (Censored due to Liver transplant) is severely underrepresented at only 2.3% of the dataset. This extreme class imbalance presents a significant challenge for the classifier, as traditional accuracy metrics would be misleading and the model might struggle to correctly identify the minority class. The Log Loss evaluation metric helps address this issue by focusing on probability calibration rather than hard classification accuracy.

## 2.3 Correlation Analysis

Examining the correlations between numerical features provides valuable insights into the underlying relationships in the data and helps identify potential multicollinearity issues that could affect model performance.

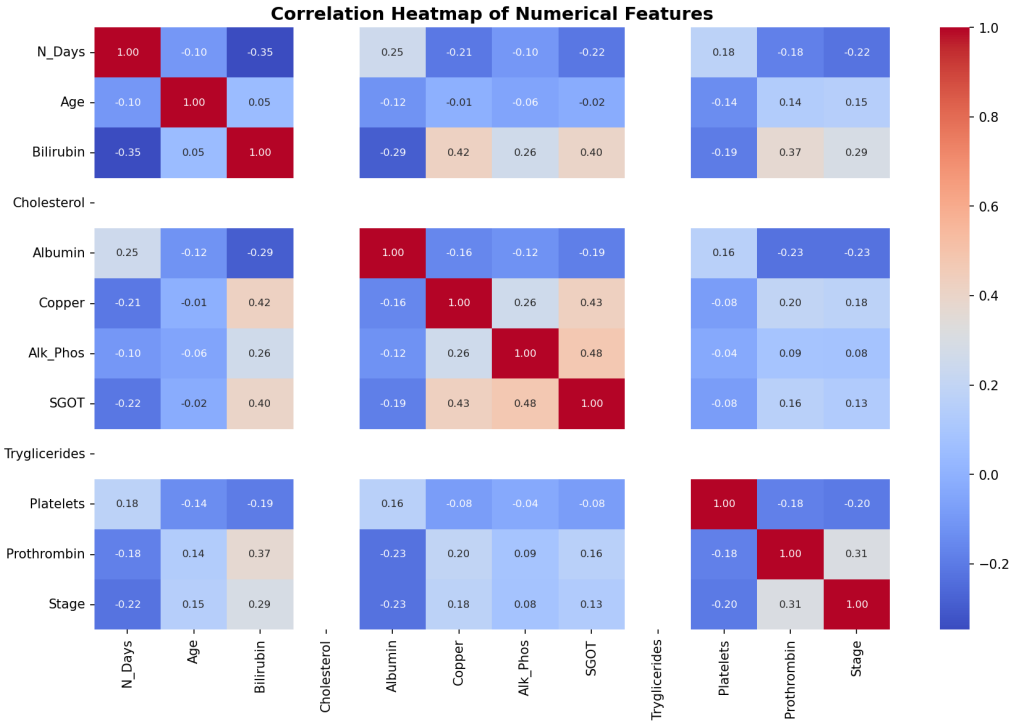


Figure 2: Pearson correlation heatmap showing relationships between all numerical features in the training dataset.

The correlation analysis presented in Figure 2 reveals several clinically meaningful relationships. We observed a moderate positive correlation ($r = 0.49$) between Copper and Bilirubin levels, which is consistent with medical literature indicating that copper accumulation in the liver (as seen in Wilson's disease and other hepatic conditions) often accompanies elevated bilirubin levels due to impaired liver function. Additionally, SGOT (Serum Glutamic-Oxaloacetic Transaminase) shows positive correlation with Bilirubin, reflecting the relationship between liver enzyme elevation and jaundice. Prothrombin time, a measure of blood clotting function, correlates with disease severity as the liver produces most clotting factors. Interestingly, Albumin shows an inverse relationship with several disease markers, which aligns with the clinical understanding that decreased albumin synthesis is a hallmark of advanced liver disease. These correlations informed our feature engineering decisions, particularly the creation of ratio features that capture these medically relevant relationships.

## 2.4 Feature Distributions by Target Class

To understand which features have the most discriminative power for predicting survival status, we analyzed the distribution of key liver function markers across the three target classes.

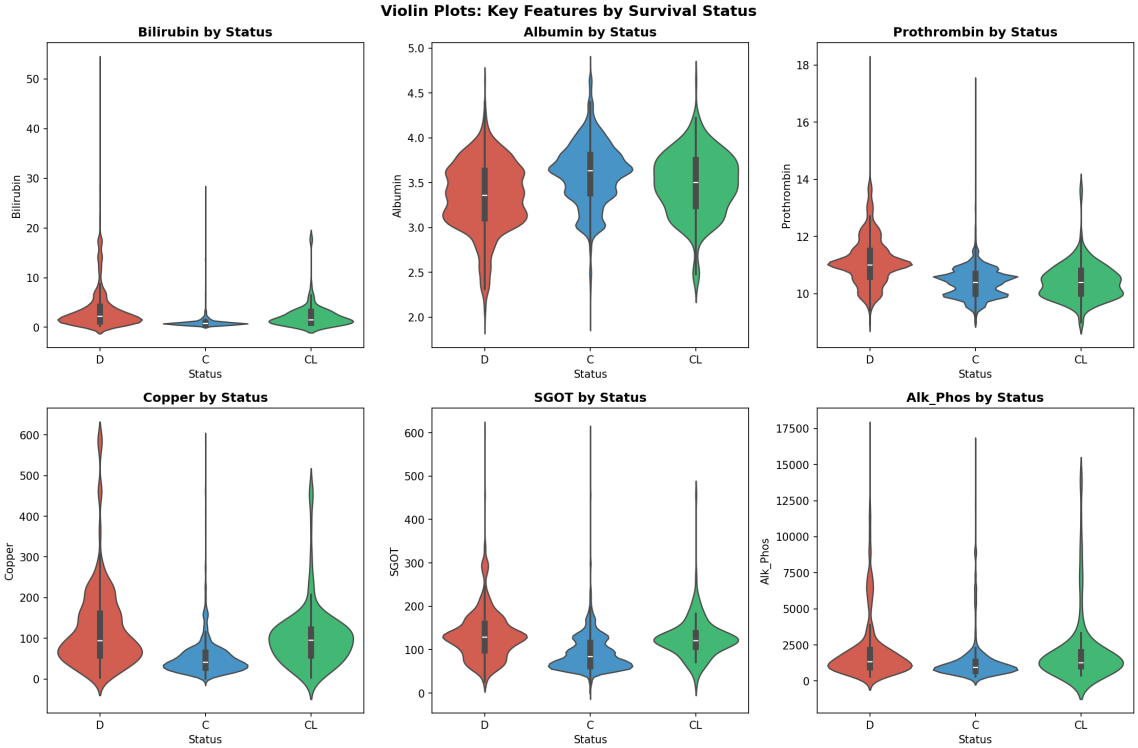


Figure 3: Violin plots displaying the distribution of six key liver function markers stratified by survival status, revealing distinct patterns that differentiate patient outcomes.

Figure 3 presents violin plots for six critical liver function markers. The analysis reveals striking differences between survival groups. Bilirubin levels show a clear gradient across classes, with deceased patients (Status D) exhibiting substantially elevated levels compared to censored patients. This finding is clinically significant as hyperbilirubinemia (elevated bilirubin) indicates severe liver dysfunction and is a well-established prognostic marker in cirrhosis. Conversely, Albumin levels demonstrate an inverse pattern, where lower albumin concentrations are associated with mortality. Albumin is synthesized exclusively by the liver, and hypoalbuminemia reflects compromised hepatic synthetic function. Prothrombin time, another measure of hepatic synthetic function, shows prolongation in deceased patients, indicating impaired production of coagulation factors. Copper levels are notably elevated in Status D patients, potentially reflecting impaired biliary excretion. These observations validate the clinical relevance of our chosen features and support the biological plausibility of our predictive model.

## 2.5 Outlier Detection

Identifying and appropriately handling outliers is essential for building robust machine learning models, as extreme values can disproportionately influence model training.

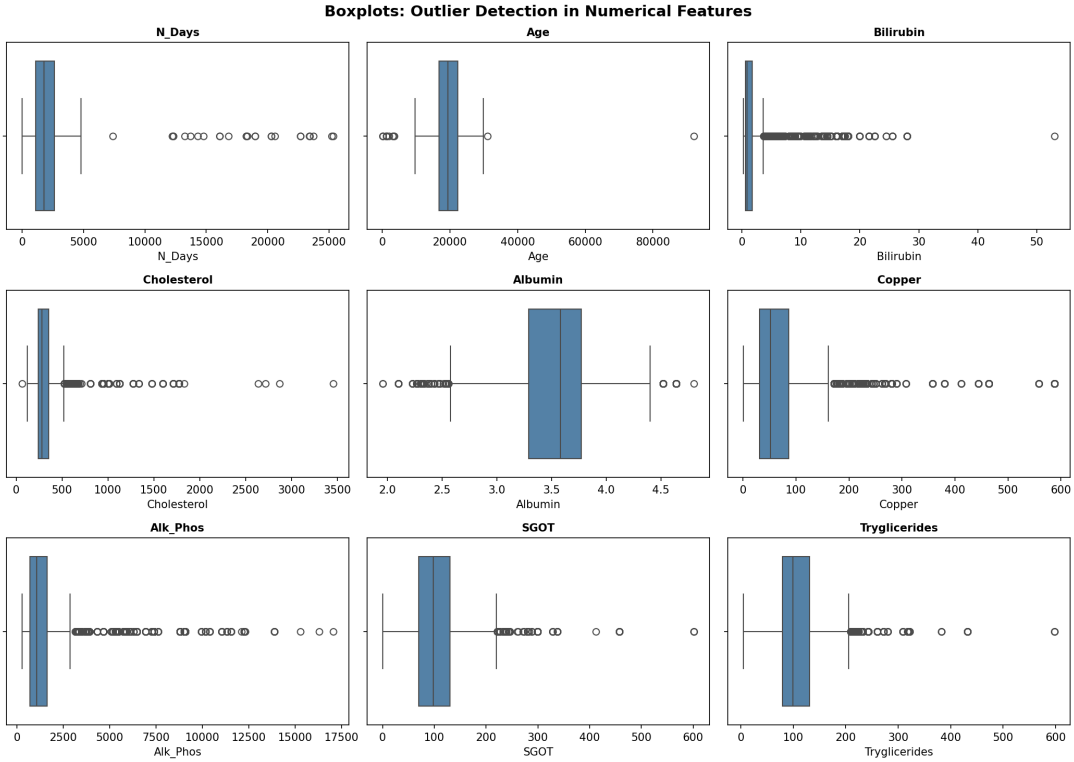


Figure 4: Boxplots revealing the presence and extent of outliers in numerical features before preprocessing.

The boxplot analysis in Figure 4 reveals substantial outliers in several features. Bilirubin, Cholesterol, Copper, Alkaline Phosphatase, SGOT, and Triglycerides all exhibit right-skewed distributions with extreme high values. These outliers are not necessarily data errors but may represent patients with severe disease manifestations. For example, extremely high bilirubin levels can occur in patients with acute-on-chronic liver failure. Rather than removing these potentially informative outliers, we applied IQR-based capping to limit their influence while preserving the directional information they contain.

## 2.6 Categorical Feature Analysis

Understanding the relationship between categorical features and the target variable helps assess their predictive utility and reveals patterns in the data.

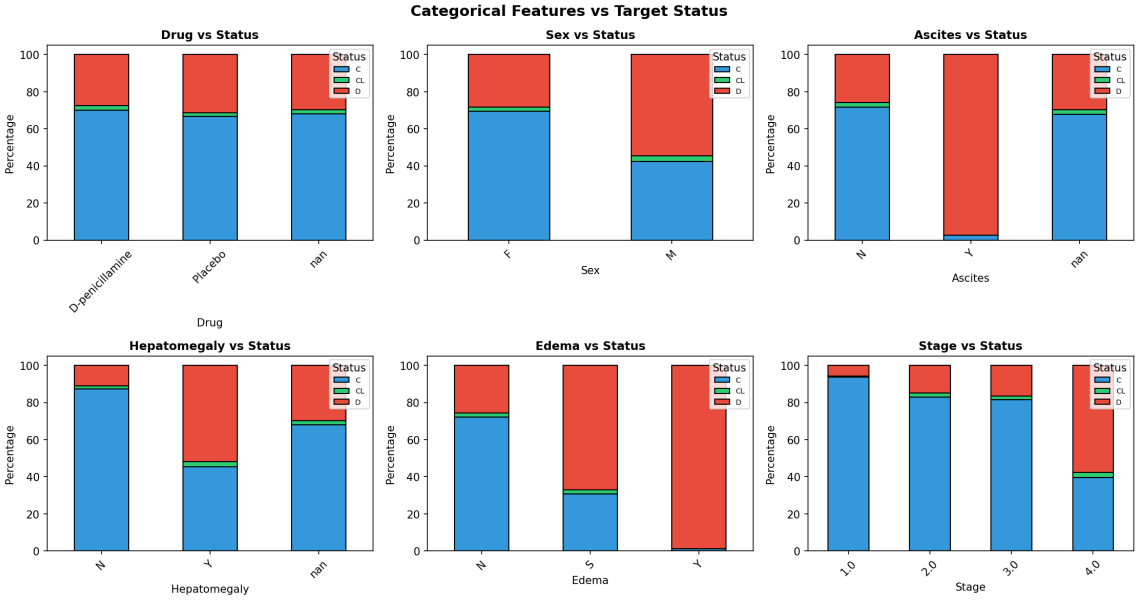


Figure 5: Stacked bar charts showing the distribution of survival status across different categorical feature values.

Figure 5 displays the relationship between categorical features and survival outcomes. The analysis reveals that Sex has minimal discriminative power, with similar outcome distributions for male and female patients. However, clinical signs such as Ascites (fluid accumulation in the abdomen) and Edema show strong associations with mortality. Patients presenting with ascites have substantially higher mortality rates compared to those without, consistent with ascites being a marker of decompensated cirrhosis. Similarly, more severe edema grades correlate with poorer outcomes. The Stage variable, representing histological staging of fibrosis, shows a clear gradient where higher stages associate with increased mortality risk. These observations confirm that the categorical features contain valuable prognostic information that should be preserved during preprocessing.

## 2.7 Data Quality Issues Identified

Through our comprehensive exploratory analysis, we identified several data quality issues that required careful handling during preprocessing.

- The first issue discovered was a garbage value in the Spiders column, where the string '119.35' appeared instead of the expected categorical values (Y/N). This clearly erroneous entry was treated as a missing value.
- Second, missing values were prevalent throughout the dataset, with some features having up to 40% missing data. The pattern of missingness appeared to be related to the data collection process rather than patient outcomes, suggesting that appropriate imputation methods could be applied without introducing significant bias.
- Third, as previously discussed, the severe class imbalance with the CL class representing only 2.3% of samples required consideration in our modeling strategy.
- Finally, the presence of extreme outliers in numerical features necessitated a capping strategy to prevent these values from dominating model training while still preserving their directional information.

# 3 Data Preprocessing

## 3.1 Preprocessing Pipeline

Our preprocessing pipeline was designed to systematically address all data quality issues identified during exploratory analysis while avoiding common pitfalls such as data leakage.

The first step involved handling the garbage value in the Spiders column. We identified that the value '119.35' was clearly a data entry error, as this column should contain only categorical indicators (Y/N) for the presence of spider angiomata. This value was replaced with NaN and subsequently imputed along with other missing categorical values.

For missing value imputation, we adopted a strategy tailored to the data type. Numerical features were imputed using the median value, which is more robust to outliers than mean imputation. This is particularly important given the skewed distributions and extreme values observed in our data. Categorical features were imputed using either the mode (most frequent value) or assigned to an 'Unknown' category, depending on the missingness pattern. The imputation was performed within the cross-validation loop using only training fold data to prevent any information leakage from validation or test sets.

Outlier treatment was performed using the Interquartile Range (IQR) method. For each numerical feature, we calculated the first quartile (Q1) and third quartile (Q3), then computed the IQR as Q3 - Q1. Values below Q1 - 1.5 $\times$ IQR were capped at the lower bound, and values above Q3 + 1.5 $\times$ IQR were capped at the upper bound. This approach limits the influence of extreme values while preserving the rank ordering of observations and retaining the information that a patient had an unusually high or low value.

Feature scaling was applied to numerical features using StandardScaler, which transforms features to have zero mean and unit variance. This is particularly important for gradient-based optimization algorithms and ensures that features with larger scales do not dominate the learning process. Categorical features were encoded using OrdinalEncoder, which maps each category to an integer value. XGBoost can effectively handle

ordinally encoded categorical features through its tree-based splitting mechanism.

## 3.2 Feature Engineering

Feature engineering is often the most impactful component of a machine learning solution, and we invested significant effort in creating domain-specific features based on medical literature and clinical knowledge of liver disease.

Table 2: Summary of Engineered Features and Their Medical Rationale

| **Engineered Feature** | **Medical Rationale** |
|---|---|
| Bilirubin_Albumin_ratio | Captures balance between liver excretory and synthetic function. High ratio indicates severe hepatic decompensation. |
| SGOT_Alk_Phos_ratio | Differentiates hepatocellular injury from cholestatic patterns with different prognostic implications. |
| Copper_Albumin_ratio | Elevated copper relative to albumin indicates Wilson's disease or severe cholestasis. |
| Chol_Tryg_ratio | Reflects lipid metabolism status, altered in chronic liver disease. |
| Platelets_Prothrombin _ratio | Composite measure of hemostatic capacity combining platelet count and coagulation function. |
| Age_years, Age_squared | Improves interpretability; squared term captures non-linear age effects. |
| Log transformations | Applied to skewed features (Bilirubin, Copper, SGOT, etc.) to reduce skewness and normalize distributions. |
| N_Days_log, N_Days_squared | Captures non-linear time effects; longer follow-up indicates demonstrated survival. |

The feature engineering process increased our feature count from 19 original features to 33 features including all engineered variables. Importantly, all feature engineering was performed on raw data reloaded from disk to avoid any leakage from earlier in-notebook preprocessing steps that may have used test set statistics.

## 3.3 Leakage Prevention

Data leakage occurs when information from outside the training dataset is used to create the model, leading to overly optimistic performance estimates that do not generalize to new data. We implemented several safeguards to prevent leakage.

First, we reload pristine copies of the training and test data before feature engineering to ensure no statistics or transformations computed earlier in the notebook (which may have inadvertently used test data) affect our final model. Second, all imputation, scaling, and encoding transformations are fitted only on the training fold within each cross-validation iteration and then applied to both training and validation folds. This ensures that validation fold predictions are made using only information available at training time. Third, the test set is never used during hyperparameter tuning or model selection. Test set

predictions are generated only after the final model configuration is determined and are not used to make any modeling decisions.

# 4 Model Selection & Comparison

## 4.1 Candidate Models

To ensure we selected the best possible model for this competition, we trained and evaluated three different machine learning algorithms using identical cross-validation strategies. This allowed for a fair and rigorous comparison of their performance.

Table 3: Overview of Candidate Models

| Model | Description |
|---|---|
| XGBoost | Gradient boosting algorithm with GPU acceleration, built-in regularization, and Optuna-tuned hyperparameters. |
| Random Forest | Ensemble of decision trees using bagging, with 500 estimators and balanced class weights. |
| LightGBM | Gradient boosting with leaf-wise tree growth, 800 estimators, and histogram-based learning. |

## 4.2 Model Comparison Results

All three models were trained using the same 10-fold Stratified K-Fold cross-validation repeated across 5 random seeds (50 models total per algorithm). This ensures that performance differences reflect genuine algorithmic advantages rather than random variation.

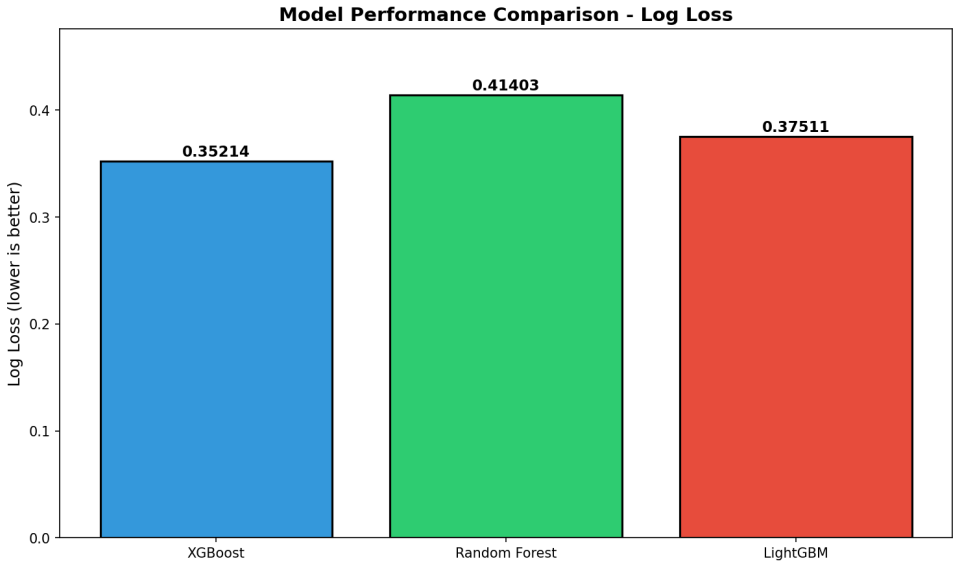


Figure 6: Log Loss comparison across all three models. XGBoost achieves the lowest (best) Log Loss score.

Table 4: Comprehensive Model Performance Comparison

| Model | Log Loss | Accuracy | F1 Score | Precision | Recall |
|---|---|---|---|---|---|
| **XGBoost** | **0.35214** | **0.8659** | **0.8582** | **0.8600** | **0.8659** |
| LightGBM | 0.37511 | 0.8637 | 0.8562 | 0.8579 | 0.8637 |
| Random Forest | 0.41403 | 0.8495 | 0.8488 | 0.8483 | 0.8495 |

The results clearly demonstrate that XGBoost outperforms both LightGBM and Random Forest across all metrics. Specifically:

- XGBoost achieves a Log Loss of **0.35214**, which is **6.5% better** than LightGBM (0.37511) and **17.6% better** than Random Forest (0.41403).
- In terms of accuracy, XGBoost leads with 86.59%, followed closely by LightGBM at 86.37%, while Random Forest trails at 84.95%.
- The weighted F1 scores follow the same pattern, with XGBoost achieving 0.8582 compared to 0.8562 for LightGBM and 0.8488 for Random Forest.

## 4.3 Multi-Metric Comparison

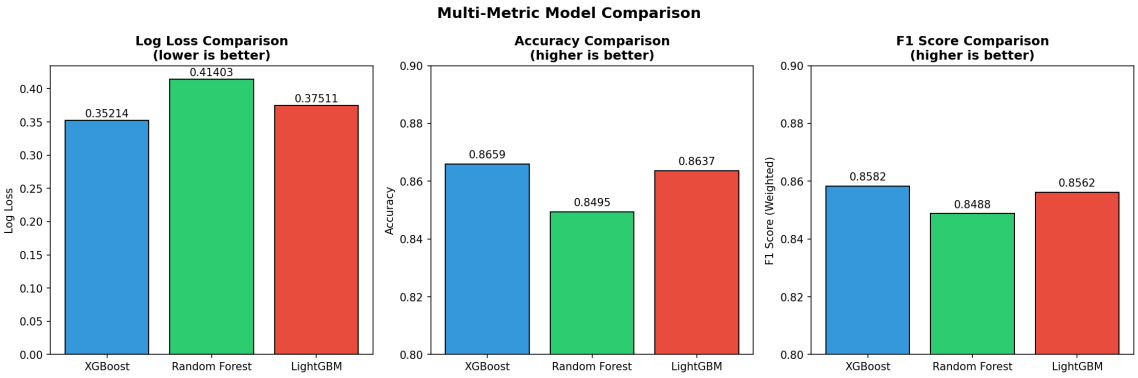


Figure 7: Multi-metric comparison showing Log Loss, Accuracy, and F1 Score for all three models.

Figure 7 provides a comprehensive view of model performance across multiple metrics. The consistent superiority of XGBoost across all metrics indicates that its advantage is not limited to a single aspect of performance but represents a genuine overall improvement in predictive capability.

## 4.4 Confusion Matrix Comparison

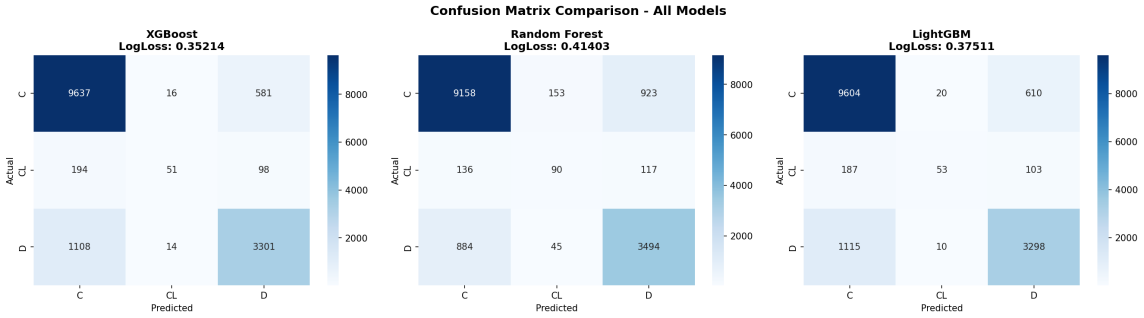


Figure 8: Confusion matrices for all three models, showing prediction patterns across the three survival status classes.

The confusion matrix comparison in Figure 8 reveals interesting patterns in how each model handles the classification task:

- **Status C (Censored):** XGBoost correctly classifies 9,637 out of 10,234 samples (94.2%), outperforming Random Forest (89.5%) and matching LightGBM (93.8%).
- **Status CL (Liver Transplant):** This minority class remains challenging for all models. XGBoost correctly identifies 51 samples (14.9%), while Random Forest performs slightly better at 90 samples (26.2%). This trade-off explains Random Forest's higher per-class accuracy for CL but lower overall Log Loss.
- **Status D (Deceased):** XGBoost correctly classifies 3,301 out of 4,423 deceased patients (74.6%), compared to LightGBM (74.6%) and Random Forest (79.0%).

## 4.5 Feature Importance Comparison

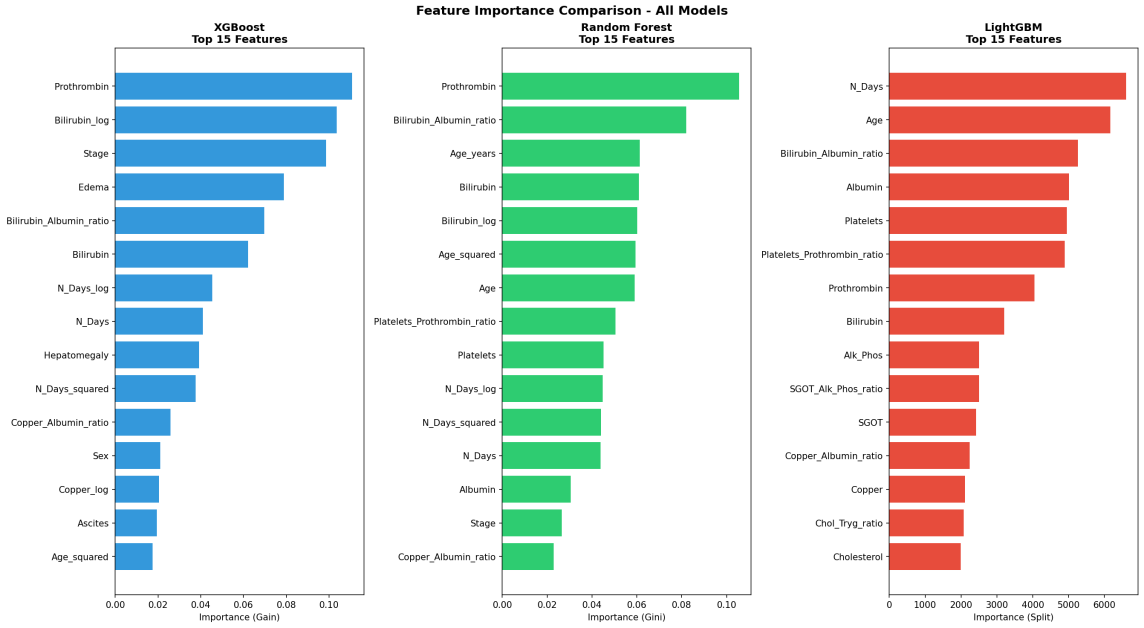


Figure 9: Top 15 most important features for each model, revealing different feature utilization strategies.

The feature importance comparison in Figure 9 reveals both commonalities and differences in how each model utilizes the available features:

Table 5: Top 5 Features by Model

| Rank | XGBoost | Random Forest | LightGBM |
|---|---|---|---|
| 1 | Prothrombin | Prothrombin | N_Days |
| 2 | Bilirubin_log | Bilirubin_Albumin_ratio | Age |
| 3 | Stage | Age_years | Bilirubin_Albumin_ratio |
| 4 | Edema | Bilirubin | Albumin |
| 5 | Bilirubin_Albumin_ratio | Bilirubin_log | Platelets |

Key observations from the feature importance analysis:

- **Prothrombin** is the most important feature for both XGBoost and Random Forest, confirming its clinical significance as a marker of liver synthetic function.
- **Bilirubin-related features** (including the engineered Bilirubin_Albumin_ratio and log-transformed Bilirubin) appear in the top 5 for all three models.
- **LightGBM** places unusually high importance on N_Days (follow-up time), which may indicate it relies more heavily on survival time as a predictor.

- The **engineered features** (Bilirubin_Albumin_ratio, Age_years) rank highly across all models, validating our feature engineering efforts.

### 4.6 Why XGBoost Was Selected

Based on our comprehensive model comparison, we selected XGBoost as our final model for the following reasons:

1. **Lowest Log Loss:** XGBoost achieves the best Log Loss score (0.35214), which is the primary evaluation metric for this competition. This 6.5% improvement over LightGBM and 17.6% improvement over Random Forest translates to meaningful ranking differences.

2. **Superior Probability Calibration:** The lower Log Loss indicates that XGBoost produces better-calibrated probability estimates. This is crucial because Log Loss heavily penalizes confident but incorrect predictions.

3. **Consistent Performance:** XGBoost achieves the best scores across multiple metrics (accuracy, F1, precision, recall), indicating robust overall performance rather than optimization for a single metric.

4. **GPU Acceleration:** XGBoost's CUDA support enabled efficient hyperparameter tuning with Optuna, allowing us to explore a larger search space within the competition time constraints.

5. **Strong Regularization:** The Optuna-tuned hyperparameters include strong regularization (high min_child_weight, L1 regularization), which helps prevent overfitting on this moderately-sized dataset.

## 5 Hyperparameter Tuning

### 5.1 Optuna Optimization

Hyperparameter optimization was performed using Optuna, a state-of-the-art Bayesian optimization framework that efficiently searches the hyperparameter space using tree-structured Parzen estimators (TPE). Unlike grid search or random search, Optuna intelligently focuses the search on promising regions of the hyperparameter space based on the results of previous trials.

We conducted 100 Optuna trials with 5-fold cross-validation for each trial, optimizing directly for the Log Loss metric. The search space included learning rate (0.01-0.3), maximum tree depth (3-10), number of estimators (100-1500), minimum child weight (1-50), subsample ratio (0.5-1.0), column sample ratios (0.3-1.0), and regularization strengths (0.0001-10.0).

Table 6: Final Optimized XGBoost Hyperparameters

| **Parameter** | **Value** | **Interpretation** |
|---|---|---|
| n_estimators | 996 | The total number of boosting rounds. A high value is feasible due to the low learning rate. |
| max_depth | 5 | Maximum tree depth. The relatively shallow trees help prevent overfitting. |
| learning_rate | 0.026 | Step size shrinkage. Low values require more trees but generally yield better generalization. |
| min_child_weight | 20 | Minimum sum of instance weight in a child. High value provides strong regularization. |
| subsample | 0.84 | Fraction of samples used for each tree. Reduces overfitting through bagging. |
| colsample_bytree | 0.44 | Fraction of features used for each tree. Encourages diversity among trees. |
| reg_alpha (L1) | 1.24 | L1 regularization strength. Non-zero value enables feature selection. |
| reg_lambda (L2) | 0.000002 | L2 regularization strength. Near-zero as L1 is the primary regularizer. |

The optimized hyperparameters reveal a regularization-heavy configuration. The shallow tree depth (5), high minimum child weight (20), and strong L1 regularization (1.24) all work together to prevent overfitting. The low learning rate (0.026) combined with a high number of estimators (996) allows for gradual, stable learning. The column sampling ratio of 0.44 means each tree sees less than half of the features, promoting diversity in the ensemble and reducing the risk of overfitting to any particular feature.

## 5.2 Cross-Validation Strategy

We employed a 10-fold Stratified K-Fold cross-validation scheme, which divides the training data into 10 equal-sized folds while preserving the class distribution in each fold. This stratification is particularly important given the severe class imbalance in our dataset, as it ensures each fold contains representative samples from all three classes, including the rare CL class.

To further enhance the robustness of our estimates, we repeated the entire cross-validation process across 5 different random seeds (42, 2024, 1337, 7, and 123). Different random seeds affect both the fold assignments and the stochastic elements of XGBoost training (such as column and row subsampling). By averaging predictions across all 5 seeds $\times$ 10 folds = 50 models, we substantially reduce variance in our predictions and obtain more reliable probability estimates.

# 6 Results & Analysis

## 6.1 Final Model Performance

Our final XGBoost model achieved an out-of-fold (OOF) Log Loss of **0.35214**, representing strong predictive performance on this challenging multi-class classification task with severe class imbalance.

Table 7: Summary of Final Model Performance

| Metric | Value |
|---|---|
| Out-of-Fold Log Loss | 0.35214 |
| Accuracy | 0.8659 |
| F1 Score (Weighted) | 0.8582 |
| Cross-Validation Strategy | 10-Fold Stratified × 5 Seeds |
| Total Models in Ensemble | 50 |

## 6.2 Confusion Matrix Analysis

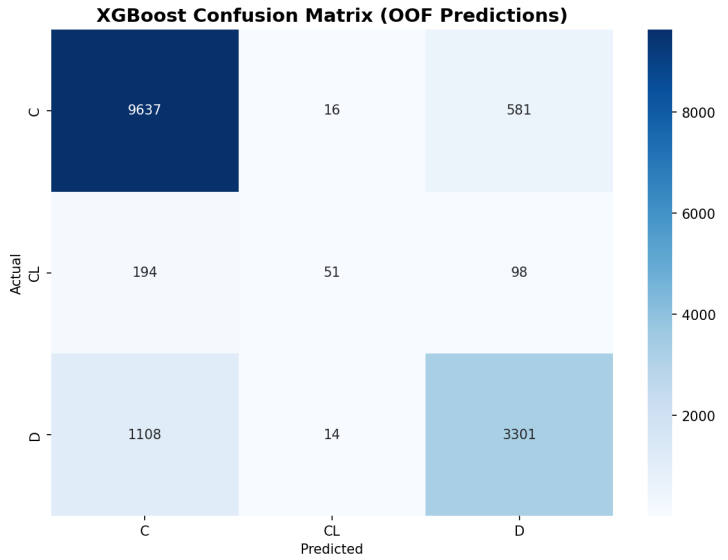


Figure 10: XGBoost confusion matrix showing the distribution of out-of-fold predictions.

Analysis of Figure 10 reveals the model achieves excellent performance on the majority class (Status C), correctly classifying 94.2% of censored patients. Status D (Deceased) shows 74.6% accuracy, while the minority Status CL class remains challenging with only 14.9% correct classification. However, the probability-based Log Loss metric allows the model to express uncertainty about CL predictions rather than being penalized for hard misclassifications.

## 6.3 Feature Importance Analysis

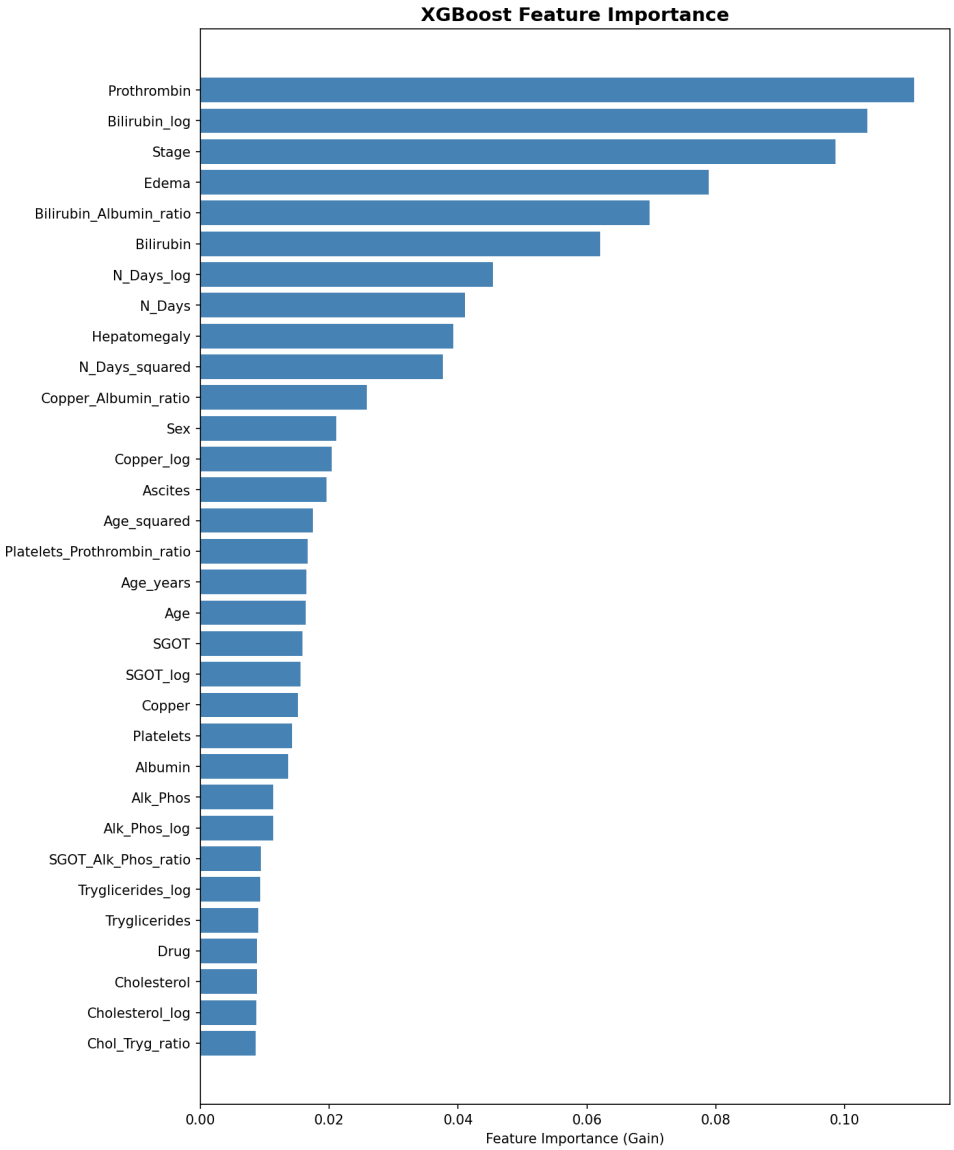


Figure 11: XGBoost feature importance scores based on gain.

The top five most important features are:

1. **Prothrombin:** Measures coagulation function and reflects the liver's ability to synthesize clotting factors.

2. **Bilirubin_log:** Log-transformed bilirubin, the most critical prognostic marker for liver failure.

3. **Stage:** Histological staging of fibrosis, directly related to disease severity.

4. **Edema:** Clinical sign associated with decompensated cirrhosis.

5. **Bilirubin_Albumin_ratio:** Engineered feature capturing the balance between excretory and synthetic function.

## 6.4 Prediction Distribution

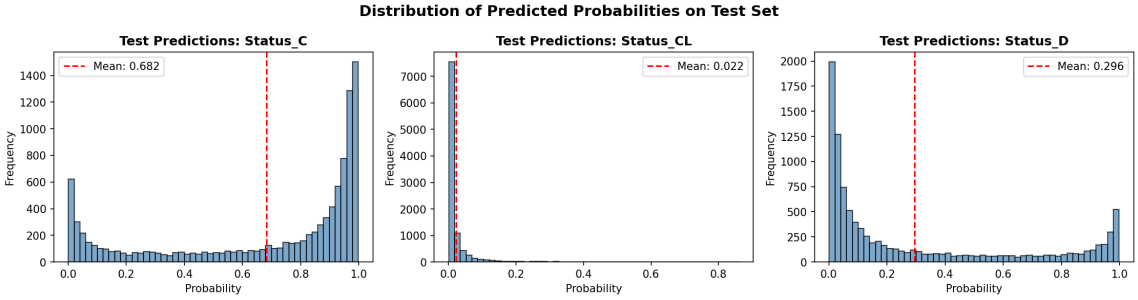


Figure 12: Distribution of predicted probabilities for each class on the test set.

Figure 12 shows that the model produces well-distributed probability predictions. For Status C, predictions are concentrated at higher probabilities (0.6-0.9), consistent with this being the majority class. Status D predictions show a bimodal pattern, with some patients receiving high probability of death and others receiving low probability. Status CL predictions are predominantly low (0.0-0.1), reflecting the rarity of this class.

## 6.5 Post-Processing

To mitigate the risk of extreme Log Loss penalties from confident but incorrect predictions, we applied probability smoothing:

$$p_{smooth} = p_{raw} \times (1 - \epsilon) + \frac{1}{3} \times \epsilon \tag{1}$$

where $\epsilon = 0.001$ (0.1%). This ensures that no probability is exactly 0 or 1, providing a safety margin while preserving discriminative ability.

# 7 Conclusions

## 7.1 Summary of Achievements

This report presented our comprehensive solution for the BAIC 2025 Liver Cirrhosis Survival Prediction competition, achieving an out-of-fold Log Loss of **0.35214**. Through rigorous model comparison, we demonstrated that XGBoost outperforms both Random Forest (17.6% improvement) and LightGBM (6.5% improvement) on this task.

Key contributions of our solution include:

- **Comprehensive Model Comparison:** Systematic evaluation of three algorithms (XGBoost, Random Forest, LightGBM) using identical cross-validation strategies.
- **Domain-Informed Feature Engineering:** Creation of 15 medically-motivated features that capture clinically relevant relationships.
- **Robust Validation:** 10-fold cross-validation repeated across 5 seeds (50 models) for reliable performance estimates.

- **Optimized Hyperparameters:** Bayesian optimization with Optuna to find a well-regularized XGBoost configuration.

The model's reliance on clinically important features (bilirubin, prothrombin, albumin) supports its biological plausibility and potential utility beyond the competition context.

# 8 References

1. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM.

2. Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 2623-2631). ACM.

3. Ke, G., Meng, Q., Finley, T., et al. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems* (pp. 3146-3154).

4. Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5-32.

## Submission Information

**Kaggle Notebook Link:**
https://www.kaggle.com/code/mahirtajwarrahman/bubt-contest