Early Identification of High-Risk ICU Patients Using Machine Learning

Rishabh Sharad Pomaje *Dept. of Electrical Engineering*
Stanford University
rishabhp@stanford.edu Rutanshu Jhaveri *CGOE Student*
rutanshu@stanford.edu Shruthi Shekar *CGOE Student*
scshekar@stanford.edu

***Abstract*—The abstract is optional, depending on your available space. It should consist of 1 paragraph consisting of the motivation for your paper and a high-level explanation of the methodology you used/results obtained.**

I. Introduction

Hospital readmissions, particularly those that occur shortly after discharge, represent a major clinical and economic burden. In the United States, unplanned readmissions cost the Centers for Medicare and Medicaid Services (CMS) an estimated $17-26 billion annually [1], while simultaneously straining limited hospital capacity. For patients with chronic conditions such as heart failure or COPD, frequent readmissions are associated with elevated mortality risk and increased psychological and social stress. Early identification of high-risk patients enables targeted interventions, including individualized discharge planning and timely outpatient follow-up.

With increasing availability of electronic health records (EHRs) and large de-identified datasets such as MIMIC, it is now possible to model heterogeneous patient information spanning demographics, comorbidities, laboratory measurements, vitals, and length of stay. In this work, we experiment with different machine learning models to predict all-cause 30-day readmission using data from a patient's first ICU stay. Our experiments evaluate models of varying complexity, from baseline logistic regression to gradient-boosted decision trees (XGBoost, CatBoost) and Deep Learning models (TabNet). Among these, XGBoost achieves the strongest predictive performance. The input to these models consists of a various laboratory vital test measurements and a few patient details. The models output the probability of readmission which is then used for classification.

II. Related Work

III. Dataset and Features

We use the MIMIC-III Clinical Database (PhysioNet) [2], which contains de-identified EHR data from more than 40,000 ICU patients at a single center. The dataset includes demographics, laboratory measurements, vital signs, diagnosis codes, clinical notes, and additional patient information. For this study, we focus on a selected set of laboratory test parameters (listed in Table I) and construct per-patient feature vectors using the mean, minimum, maximum, and standard deviation of each parameter. These lab values, typically available within 4-12 hours of admission, support early readmission risk assessment. We also incorporate patient age at admission and ICU length of stay during the first visit. The target variable is defined as all-cause ICU readmission within 30 days.

Data extraction from MIMIC-III was a time and compute-intensive components of the project, as the database comprises multiple interlinked tables that must be queried systematically. Following the approach of prior studies [3], [4], we construct a well-defined cohort for model development. Our patient selection criterion are: (i) patients aged 18 years or older; (ii) exclusion of patients who died during their ICU stay; and (iii) exclusion of patients who were eventually readmitted to the ICU but only after a hospital discharge, as these cases do not constitute true readmissions.

TODO: PLEASE EXPLAIN THE REASONING BEHIND THE CRITERION.

These criteria provide a coarsely refined initial dataset, after which additional preprocessing steps were applied depending on the requirements of each model. In general, we employed standard preprocessing techniques, including imputation for missing values (a common issue in clinical datasets), normalization, data balancing trick,and correlation-based dimensionality reduction. To evaluate the effect of preprocessing, we also trained baseline models without any such adjustments (detailed discussion included in results section). We also dropped non-informative identifiers, such as the ICU stay ID, to prevent introducing unintended bias into the models.

The fully processed dataset was divided into training

Anion-Gap	Bicarbonate	Calcium
Chloride	Creatinine	Glucose
Hematocrit	Hemoglobin	MCHC
MCV (Mean Corpuscular Volume)	Magnesium	PT (Prothrombin Time)
Phosphate	Potassium	RDW (Red Cell Distribution Width)
RBC Count	Sodium	Urea-Nitrogen
WBC Count	Age (years)	ICU-time (hrs)

TABLE I
STATIC VITAL PARAMETERS AND PATIENT ATTRIBUTES USED TO GENERATE THE FEATURE VECTOR FOR EACH ICU STAY.

and test subsets using a 70-30 split. When necessary, the training portion was further partitioned into training and validation sets for hyperparameter optimization. Summary statistics for the final cohort are provided in Table II.

In all cases, we used stratified splitting to construct the train, validation, and test sets. Stratification preserves the proportion of readmissions across all subsets, which is essential given that the positive class constitutes only about 10.74% of the cohort.

IV. METHODS

Prior machine learning-based studies on this task use different datasets or apply different cohort selection criterion prohibiting direct comparison. Consequently, we implemented and evaluated multiple ML Models which we now briefly discuss.

Prior machine learning studies addressing readmission prediction typically rely on different datasets or adopt distinct cohort selection criteria, making direct comparisons infeasible. To establish a consistent performance baseline for our cohort, we therefore implemented and evaluated multiple machine learning models, briefly described below.

Suppose a patient is represented by the feature vector $\mathbf{x}$ and y is used to denote the class (readmission/non-readmission) of that patient.

A. Logistic Regression

As a starting point for this binary classification task, we employ logistic regression (LR), a standard linear model widely used in clinical risk prediction ([5] uses LR to define a *Readmission Risk Score*).

Logistic regression models the probability of readmission as

$$P(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b),$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function, $\mathbf{w}$ is the weight vector, and b is the bias term. The model parameters are obtained by minimizing a loss function. Despite its simplicity and linear decision boundary, logistic regression provides a strong interpretability advantage and serves as a useful baseline for evaluating more expressive models.

B. XGBoosted Trees

While the logistic regression provided a solid starting point, its performance on our data suggested more complex relationships in the data. To capture nonlinear relationships and feature interactions that linear models cannot represent, we employ eXtreme Gradient Boosting (XGBoost) [6], a high-performance ensemble method based on gradient-boosted decision trees. XGBoost constructs a model that cumulatively uses the prediction of multiple decision trees. Mathematically,

$$\hat{y} = \sum_{k=1}^{K} f_k(\mathbf{x}), \qquad f_k \in \mathcal{F},$$

where each f_k is a regression tree and $\mathcal{F}$ denotes the space of all possible trees. XGBoost optimizes the following regularized objective:

$$\mathcal{L} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \left(\gamma T_k + \frac{1}{2}\lambda \|\mathbf{w}_k\|^2\right),$$

where $l(\cdot)$ is a differentiable loss function, T_k is the number of leaves in the k-th tree, $\mathbf{w}_k$ are the leaf weights, and γ, λ are regularization parameters that penalize model complexity. This model is particularly well suited for classification tasks on structured, tabular data, such as the clinical variables used in this project. With appropriate hyperparameter tuning, XGBoost often outperforms more complex architectures while remaining computationally efficient.

C. CatBoost

CatBoost [7] is another gradient-boosted decision tree algorithm designed to handle categorical features natively without requiring extensive preprocessing. It uses an ordered boosting scheme to reduce overfitting and employs symmetric trees for computational efficiency. Like XGBoost, CatBoost minimizes a regularized loss function.

	Train Set	Validation Set	Test Set	Total
No. of Entries	14939	6403	9147	30489
No. of Readmissions	1605	688	983	3276
(% Readmissions)	∼10.74	∼10.74	∼10.74	∼10.74

TABLE II
STATISTICS OF OUR COHORT SET

CatBoost is effective for tabular datasets, especially when categorical variables play an important role. It also exhibits superior performance without any parameter tuning, as we observed in our experiments.

D. *TabNet*

TabNet [8] is a deep learning architecture specifically designed for tabular data. Unlike traditional fully connected networks, TabNet uses sequential attention to choose which features to reason from at each decision step, allowing the model to focus on the most relevant information. It combines feature selection and representation learning in a single framework. The model optimizes a standard classification loss (e.g., binary cross-entropy for readmission prediction) and includes sparse regularization on the feature selection masks to improve interpretability and prevent overfitting. TabNet can capture complex, non-linear interactions among features, making it a competitive alternative to gradient-boosted trees for tabular datasets.

V. EXPERIMENTS, RESULTS, AND DISCUSSION

In this section, we discuss and analyze the experiments[1] conducted during this project.

As mentioned previously, the dataset is imbalanced, with only about 10% of patients experiencing readmission. Consequently, accuracy is not an appropriate metric for model performance. Instead, we prioritize AUROC for model evaluation. Table III summarizes the performance of all evaluated models. In addition, we report weighted (class-frequency weighted) and macro (equal-weighted) precision and recall metrics.

A. *Logistic Regression Experiments*

Logistic regression (LogReg) provides a solid baseline for binary classification. Its simplicity allows rapid experimentation with preprocessing techniques. All LogReg experiments were implemented using the `scikit-learn` Python library [9]. Since logistic regression cannot handle missing values, stricter preprocessing was required, reducing the dataset to 9,774 examples ($\sim 13\%$ readmission rate). Despite this limitation, the model remains a useful baseline because the data is drawn from the same distribution.

1) **LogReg (Base)**: The first exploration used default parameters: solver = `lbfgs`, convergence tolerance = 0.0001, regularization = L_2, maximum iterations = 500,000. This baseline model achieved an AUROC of 0.670 and helped us gain initial insights into the dataset.
2) **LogReg (Balanced)**: To address class imbalance, we experimented with (i) minority oversampling and (ii) majority undersampling with loss upweighting. These strategies had minimal effect, yielding an AUROC of 0.668. The downscaling factor of 4 (i.e., $\frac{\texttt{num\_minority}}{\texttt{num\_majority}} = \frac{1}{4}$) was determined through trial and error, with similar performance observed for values between 3 and 5.
3) **LogReg (Imputation)**: Missing data was handled using median imputation. To prevent data leakage, the imputer was fitted only on the training set and then applied to the test set. This approach slightly improved performance, achieving an AUROC of 0.709.

Overall, the performance of the LogReg model was consistent with previous studies. The lack of improvement from data balancing techniques reflects the complex distribution and inherent noise characteristic of medical datasets. Hence, we did not apply synthetic sampling methods to the more complex models. In contrast, imputing missing values led to improved performance, likely due to the increased amount of data available for training and testing. This again points to the intricate relationships present in the dataset, which can be better captured when more complete data is provided to the model.

B. *XGBoost Experiments*

Motivated by the complex relationships present in the data, we turned to a more powerful model known to perform well on tabular datasets: XGBoost. All

[1]The source code of experiments is available at `GitHub`

Model	Precision		Recall		AUROC
	Weighted	Macro	Weighted	Macro	
LogReg (Base)	0.54	0.78	0.50	0.86	0.670 ± 0.025
LogReg (Balanced)	0.56	0.82	0.62	0.66	0.668 ± 0.025
LogReg (Imputation)	0.67	0.85	0.51	0.89	0.709 ± 0.017
XGBoost (Base)	0.60	0.83	0.52	0.89	0.687 ± 0.017
XGBoost (HyperOpt)	0.63	0.85	0.60	0.87	0.745 ± 0.015
XGBoost (Imputed)	0.64	0.85	0.59	0.88	0.745 ± 0.015
CatBoost	0.XX	0.XX	0.XX	0.XX	0.XX
TabNet	0.XX	0.XX	0.XX	0.XX	0.XX

TABLE III
PERFORMANCE METRICS OF DIFFERENT MACHINE LEARNING MODELS FOR ICU READMISSION PREDICTION.

experiments were conducted using the `xgboost` Python library [10]. We performed three sets of experiments:

1) **XGB (Base)**: The initial model used default hyperparameters[2], including learning rate = 0.3 and maximum tree depth = 6. This baseline model achieved an AUROC of 0.687, demonstrating its ability to capture non-linear interactions among features out of the box, while also highlighting the potential benefit of hyperparameter tuning.
2) **XGB (HyperOpt)**: Hyperparameter tuning was performed using the Optuna library, leveraging the validation set to prevent data leakage. The search space was informed by parameter ranges reported effective in prior studies. The Tree-Structured Parzen Estimator (TPE) algorithm was used for optimization, as it has been shown to efficiently explore hyperparameter spaces [11]. The optimal parameters identified were: learning rate = 0.00105, max tree depth = 17, and minimum child weight = 15. A complete list of tuned parameters is available in the associated Jupyter notebook on GitHub. The model resulted in an AUROC score of 0.745 on using scaled weights equal to the fraction of readmissions to handle class imbalance.
3) **XGB (Imputed)**: This experiment was meant to study the impact of imputation with XGBoost model. As before, the simple median based strategy was applied prior hyperparameter optimization (as described in the previous experiment).

Observations: Similar to other tree-based models, XGBoost was relatively insensitive to class imbalance techniques. The ability of XGBoost to internally handle missing values and find the optimal splits factoring the missing data improves the performance. The results also indicate the importance of hyperparameter tuning and careful handling of missing, imbalanced data can meaningfully enhance predictive performance, particularly in complex clinical datasets.

C. *CatBoost and TabNet*

The experiments using CatBoost and TabNet were out of pure curiosity.

VI. CONCLUSION AND FUTURE WORK

VII. AI-USE DISCLOSURE

The use of large language models (LLMs), such as ChatGPT, was **in accordance** with the course policy. These tools were employed solely for querying documentation of specialized libraries and for minor coding assistance. Additionally, LLMs were used to improve the grammar, clarity and presentation quality of this report. No intellectual contribution to the content, analysis, or interpretation of results was derived from AI.

[2]A detailed list of default parameters is available here

VIII. Contributions

- **Rishabh Sharad Pomaje**: Literature review, Fine-grained (minor) data-processing. He designed, implemented and verified LogReg, XGBoost, CatBoost and TabNet experiments. He was responsible for final report writing and presentation.
- **Rutanshu Jhaveri**: Literature review, Study of the database, primary (major) data fetching using Queries and pre-processing, and implemented base XGBoost.
- **Shruthi Shekhar**: Literature review, feature selection and qualitative (high-level) cohort data selection criterion, qualitative field-specific analysis of the results.

References

[1] M. Alvarado, B. Lahijanian, Y. Zhang, and M. Lawley, "Penalty and incentive modeling for hospital readmission reduction," *Operations Research for Health Care*, vol. 36, p. 100376, 2023.

[2] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[3] S. Davis, J. Zhang, I. Lee, M. Rezaei, R. Greiner, F. A. McAlister, and R. Padwal, "Effective hospital readmission prediction models using machine-learned features," *BMC Health Services Research*, vol. 22, no. 1, p. 1415, Nov 2022. [Online]. Available: https://doi.org/10.1186/s12913-022-08748-y

[4] A. Moerschbacher and Z. He, "Building prediction models for 30-day readmissions among icu patients using both structured and unstructured data in electronic health records," in *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2023, pp. 4368–4373.

[5] Y. P. Tabak, X. Sun, C. M. Nunez, V. Gupta, and R. S. Johannes, "Predicting readmission at early hospitalization using electronic clinical data: An early readmission risk score," *Med. Care*, vol. 55, no. 3, pp. 267–275, mar 2017.

[6] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. ACM, Aug. 2016, p. 785–794. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[7] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," 2019. [Online]. Available: https://arxiv.org/abs/1706.09516

[8] S. O. Arik and T. Pfister, "Tabnet: Attentive interpretable tabular learning," 2020. [Online]. Available: https://arxiv.org/abs/1908.07442

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[10] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho, K. Chen, R. Mitchell, I. Cano, T. Zhou, M. Li, J. Xie, M. Lin, Y. Geng, Y. Li, J. Yuan, and D. Cortes, *xgboost: Extreme Gradient Boosting*, 2025, r package version 3.2.0.0. [Online]. Available: https://github.com/dmlc/xgboost

[11] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.