香　港　大　學

THE UNIVERSITY OF HONG KONG

STAT 3612: Statistical Machine Learning

Course Coordinator: Dr. Yu Lequan

Group Project Report

30-day All-Cause Hospital Readmission Prediction

Group Member:

Wang Ziyu 3035777547

Yao Zixuan 3035845148

Fang Jiahe 3035772482

Huang Yining 3035662522

Huang Zixun 3035844522

December 2023

1. Executive Summary

This report presents a comprehensive study on applying statistical machine-learning techniques to predict 30-day all-cause hospital readmission. The project involves data exploration, model training, interpretation of results, and assessing limitations and scope for improvement. Through this work, we will demonstrate the potential of machine learning in healthcare settings and its real-life implications in medical decision-making.

2. Introduction

Hospital readmissions within 30 days of discharge are a significant concern for healthcare systems, as they indicate potential issues in patient care and can result in increased healthcare costs (Leppin et al., 2014). Predicting readmission risk can help healthcare providers make informed decisions regarding patient care and resource allocation (Casalini et al., 2017). This report presents a systematic approach to predicting 30-day all-cause hospital readmission using statistical machine-learning techniques. Our study covers data exploration, feature engineering, model training, and result interpretation. We will also discuss the limitations of our approach and potential improvements.

The increasing availability of electronic health records (EHRs) offers an opportunity to leverage large-scale datasets to develop predictive models for hospital readmission (Johnson et al., 2023). Machine learning techniques have been widely adopted in various healthcare applications, including risk stratification, disease diagnosis, and treatment planning (Alanazi, 2022; Callahan et al., 2017). In this study, we aim to develop a robust and accurate model for predicting 30-day all-cause hospital readmission using a comprehensive set of patient features derived from EHR data.

This report is structured as follows: Section 3 details the data exploration process, including data cleaning, preprocessing, and feature engineering. Section 4 describes the model training process and the various machine learning techniques evaluated, which will also incorporate the model evaluation and selection process. Section 5 provides an interpretation of the results, including feature importance and partial dependence analysis. Section 6 discusses the limitations of our approach and potential areas for improvement. Finally, Section 7 concludes the report and highlights the implications of our findings for medical decision-making.

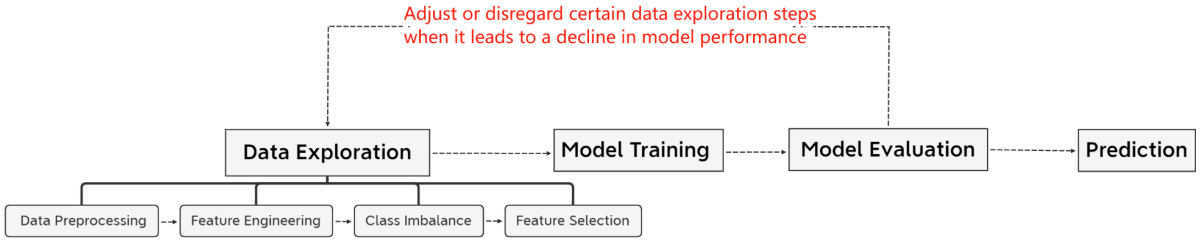
3. Data Exploration

In this section, we will explore the Electronic Health Records (EHRs) data from MIMIC-IV v1.0 (Johnson et.al., 2023). Data exploration aims to understand the data better, identify any patterns or anomalies, and make informed decisions about data preprocessing and feature engineering.

During the process of conducting data exploration, it is crucial to recognize the strong interplay between the data preprocessing steps and the subsequent training process. The preprocessed data quality significantly impacts the trained model's performance. Therefore, it is imperative to iteratively analyze and preprocess the data based on the training and validation performance to optimize the model's overall performance. Datasets that exhibit satisfactory performance will be considered for further use, while those that fail to meet the desired performance criteria will be either improved or discarded after identifying the reasons for their underperformance. Figure 1 illustrates the dynamic interaction between the data exploration and model training steps to provide a clearer understanding of the data processing procedure.

Figure 1

Interaction between the data exploration and model training steps



3.1. Dataset Overview

The EHR includes demographics (age, gender, and ethnicity), comorbidities recorded as ICD-10 codes, laboratory test results, and medications administered during hospitalizations. The dataset comprises 14,532 hospital admissions from 11,664 unique patients. Among these admissions, 2,552 resulted in readmission within 30 days of discharge, including cases where patients passed away during their hospital stay.

3.2. Data Preprocessing

In this section, we will focus on the preprocessing of the data. The data is obtained from a ‘pkl’ file containing 171 features.

To begin with, we sorted the patient data to obtain their latest demographic status. This information is likely to stay the same during the admission. By matching the 'id' in the train, valid, and test files with the keys in the "feat_dict," we can obtain the relevant EHR features for

each admission. These features will be used for further analysis, model training, and prediction tasks related to hospital readmission.

Next, we merged the train and valid datasets into the full training dataset. This step combined the training and validation data for further preprocessing. We then checked for missing values and ensured the completeness of our dataset. We did not handle outliers as we wanted to retain the information they provided.

Data scaling is essential in our preprocessing steps. It involves identifying categorical and numerical features using the "cat_idxs" variable, which contains the dataset's categorical features index. We then performed standardization on the numeric features to ensure that the values of the numeric features have zero mean and unit variance, making them more suitable for further analysis.

3.3. Feature Engineering

In this section, we will describe the feature engineering process applied to the datasets to enhance the predictive power of the models.

3.3.1. Descriptive Statistics as Features

In this step, we applied various transformations and calculations to derive summary statistics as new features (Guo et al., 2020). These statistics help us summarize the attributes of patients' records and gain valuable insights into their health conditions.

We first extracted common descriptive statistics, including mean, mode, last, and standard deviation, from the dataset. The mean represents the average value and provides an understanding of typical values observed. For example, a high mean value for a patient's blood pressure may indicate a potential health concern. The mode identifies the most common value and offers insights into prevailing trends or common attributes among the patients. If the mode value for a specific symptom is "fatigue," it suggests that fatigue is a prevalent symptom among the patients. The standard deviation measures variability or dispersion within the dataset, indicating how much the values deviate from the mean. A higher standard deviation suggests greater variability, which may indicate potential fluctuations or worsening of a patient's condition.

In addition to the primary descriptive statistics mentioned above, other measures such as quartiles (Q1 and Q3), interquartile range (IQR), range, maximum, minimum, kurtosis, and skewness were calculated to gain insights into the spread, shape, and distribution of attribute values. Quartiles and IQR help identify the range of values where most patients fall, with a larger IQR indicating higher variability in patients' conditions. This variability is significant in predicting readmission, as patients with diverse attribute values may have different outcomes.

Range, maximum, and minimum statistics provide information about the overall span of attribute values, helping identify extreme or outlier values impacting readmission risk. Kurtosis and skewness provide information about the shape and symmetry of the data distribution, indicating concentration or skewness. These distribution characteristics can offer insights into the likelihood of readmission, as specific distributions may be associated with certain health conditions or risk factors.

After extracting the descriptive statistics as features, we further investigated their correlation with readmission. This analysis allowed us to identify the relatively significant correlations between the features and the response variable and possibly the feature importance. The results are presented in Table 1 below.

Table 1

Relatively significant correlations between the features and the readmission.

Correlation Range	Variables	
0.3 - 0.2	● Lactate Blood_mean_mode ● pH Blood_mean_mode ● pCO2 Blood_mean_mode ● Lactate Blood_latest	● pCO2 Blood_latest ● Lactate Blood_entropy_std ● pCO2 Blood_entropy_std
0.4 - 0.3	● pO2 Blood_mean_mode ● pH Blood_latest	● pO2 Blood_latest

The correlations between specific features and the readmission risk fall within the ranges of 0.3 - 0.2 and 0.4 - 0.3, indicating moderate and stronger relationships, respectively. Notably, variables like pO2 Blood_mean_mode, pH Blood_latest, and pO2 Blood_latest exhibit a higher correlation with readmission. These results suggest that these features may play a more influential role in predicting the probability of readmission.

By extracting and analyzing these statistics, we can effectively summarize patients' records' attributes and comprehensively understand their health conditions. This information helps us identify patterns and potential areas of concern, ultimately enhancing our ability to make informed predictions.

3.3.2. Polynomial and Interaction Terms

In addition to the previously mentioned feature engineering methods, we also investigated using polynomial and interaction terms to capture non-linear relationships and interactions between predictors. We then added polynomial and interaction terms to both the training and test datasets, specifically focusing on the numerical columns. However, after evaluating the performance of the models with the added polynomial and interaction terms, we found that they did not significantly contribute to the prediction process. As a result, we decided to remove these features from the datasets.

There are a few potential reasons for this outcome. Firstly, the underlying relationships between predictors and the target variable may be primarily linear, rendering the inclusion of non-linear terms unnecessary. Additionally, the lack of improvement from incorporating interaction terms could be attributed to the low correlation between predictors. The interaction terms may fail to capture meaningful relationships or provide additional predictive power in such cases. Therefore, including polynomial and interaction terms may not significantly contribute to the prediction process.

3.3.3. Principal Component Analysis

In the feature engineering process, Principal Component Analysis (PCA) was applied as a dimensionality reduction technique for neural networks. It is commonly used due to its ability to capture the most significant patterns and variations in the data. However, it is essential to note that PCA is primarily applied to neural networks and does not work well with our analysis's regular machine-learning methods and transformers. This is because these algorithms typically have their own feature selection or dimensionality reduction techniques more suitable for their respective models.

Alternative techniques that may be more suitable for other machine learning algorithms are worth exploring. For example, Recursive Feature Elimination and t-SNE have demonstrated promising results in feature selection and dimensionality reduction results. Considering these alternative methods in our future analysis could potentially enhance the overall performance and accuracy of our models.

3.4. Class Imbalance

It is important to observe that there is a significant class imbalance in the target variable, with more instances labeled as 0s than 1s. This class imbalance can potentially lead to a biased model, as the model may be more inclined to predict the majority class accurately while neglecting the minority class (Elreedy & Atiya, 2019). To address this issue, we considered employing techniques such as SMOTE (Synthetic Minority Over-sampling Technique). SMOTE oversamples the minority class by creating synthetic instances, thereby balancing the class distribution in the training data. Another approach is to adjust the weights of the classes during model training, giving more importance to the minority class. These techniques can help mitigate the impact of class imbalance and improve the performance of the model in predicting both classes accurately.

3.5. Feature Selection

In order to enhance the performance and interpretability of the model, we also performed feature selection using ANOVA F-test. It assesses the significance of individual features in explaining the variance in the target variable. By calculating the F-test scores for each feature, we can

determine their importance in predicting the target variable. Based on these scores, a decision can be made whether to include or exclude certain features in the model.

However, it is important to note that the decision to perform feature selection depends on the unique characteristics of the models being used in this analysis. While reducing the number of features can simplify the model and enhance its interpretability, it may also result in a loss of valuable information. Consequently, we approach feature selection on a case-by-case basis, carefully weighing the trade-offs between model complexity, interpretability, and performance. By considering these factors, we can make informed decisions regarding the inclusion or exclusion of features in our models.

4. Model Training

In this section, we will present the model training process, which is divided into two parts: general machine learning models and neural network models.

4.1. General Machine Learning Models

4.1.1. Model Description

We evaluated a total of 13 general machine-learning models in our study. These models include logistic regression, linear and quadratic discriminant analysis, naive Bayes, k-nearest neighbors, support vector machines, classification trees, bagging, random forests, extra trees, XGBoost, gradient boosting, and AdaBoost.

The selection process considers several critical factors, namely the interpretability, complexity, and performance of each model in an intuitive sense. Our primary focus during the subsequent fitting and selection process will be on the AUC (Area Under the Curve) score.

4.1.2. Model Fitting

The model training process involves several steps, including an initial round of model fitting using default parameters, adjustment for class imbalance, best subset feature selection, and tuning hyperparameter.

4.1.2.1. Preliminary Fitting

Preliminary model fitting was conducted using default parameters, with features being the last-day observations. Cross-validated and test AUC scores were used to evaluate model performance. Table 2 below shows the cross-validated AUC and test AUC in the first and second columns, respectively.

Table 2

AUC scores in the preliminary round of model fitting.

Preliminary Model Evaluation		
Model	**AUC (valid)**	**AUC (test)**
Logistic Regression	0.7377	0.7231
LDA	0.7424	0.7418
QDA	0.688	0.7047
Naive Bayes	0.6699	0.6818
KNN	0.6153	0.5988
SVM	0.7108	0.7229
Decision Tree	0.6196	0.6132
Bagging	0.7253	0.7284
Random Forest	0.7674	0.7763
Extra Trees	0.7695	0.7699
XGBoost	0.7549	0.7593
Gradient Boosting	0.7734	0.7578
Transformer	0.7834	0.775
Bidirectional LSTM	0.7614	0.7351

Upon comparing the results, QDA, Naive Bayes, KNN, and Decision Tree models exhibited relatively low AUC scores and possessed limited hyperparameters for tuning. Consequently, these models are excluded from further investigation. The rest of the models proceeded to the next step.

4.1.2.2. Adjusting Class Imbalance

In the second step, we address the issue of imbalanced datasets. Initially, the dataset consists of 9,278 instances with a value of 0 and 2,318 instances with 1, potentially impacting prediction accuracy. To mitigate this imbalance, we experimented with two methods:

- SMOTE (Synthetic Minority Over-sampling Technique): We utilized the SMOTE package and developed our oversampling function.
- Class-weight parameter: We leveraged the class-weight parameter provided by specific classifiers, such as logistic regression, random forest, and extra tree, if available.

We observed that these methods proved effective for some models. For example, in the case of logistic regression, the original AUC was 0.73, which improved to 0.7366 with SMOTE-adjusted data and to 0.7343 when utilizing the class-weight parameter. Similar improvements were observed for random forest and gradient-boosting models. In subsequent practice, we employed both methods to determine which would yield the best results. However, it should be noted that we later discovered that the combination of oversampling and parameter tuning resulted in a test AUC of 0.5 for most of the Tree-based models, indicating the issue of overfitting when applying resampling in decision trees.

Table 3

Performance evaluation for adjusting class imbalance.

SMOTE v.s. Class-weight adjustment			
Model	**AUC (original)**	**AUC (smote adjusted)**	**AUC (class-weight adjusted)**
Logistic Regression	0.7319	0.7366	0.7343
LDA	0.7292	0.7212	N/A
Bagging	0.7323	0.7304	N/A
Random Forest	0.772	0.7786	0.7807
Extra Trees	0.7717	0.7716	0.7713
XGBoost	0.7679	0.7552	N/A
Gradient Boosting	0.7644	0.7709	N/A

4.1.2.3. Best Subset Feature Selection

In step three, we performed the best subset feature selection to determine the subset of features that consistently resulted in the highest AUC score. We considered various combinations of descriptive statistics as features and experimented with the Random Forest model, which showed the most promising result in the preliminary round. After evaluation, we found that the combination of the last day observation (latest), mean, standard deviation (sd), minimum (min), and maximum (max) descriptive statistics yielded the best results consistently. Hence, we applied this feature subset in the subsequent model training process.

Table 4.

AUC scores (test) with different feature subsets in the Random Forest model.

Feature Combinations	AUC (test)
latest	0.7763
latest+mean	0.7836
latest+mean+sd	0.7891
latest+mean+sd+min+max	0.8048
latest+mean+sd+min+max+q1+q3	0.8007
latest+mean+sd+min+max+kurt+skew	0.7949
…	…

4.1.2.4. Tuning Hyperparameter

In the last step, we performed intensive hyperparameter tuning using Bayesian Optimization, Random Search, and Grid Search techniques to locate the best hyperparameters for each model. First, we utilized Bayesian Optimization and Random Search to identify significant hyperparameters and determine their potential optimal ranges preliminarily. The two techniques are computationally efficient and help narrow the hyperparameter search space. Once we identified the models that showed promising results during the preliminary round of parameter tuning, we employed Grid Search to perform a fine-grained exploration of the hyperparameter space. This method systematically tests all possible combinations of hyperparameters within the predefined range to find the optimal values.

4.1.3. Model Comparison

After tuning hyperparameters, Extra Trees and XGBoost emerged as the top-performing models, achieving the highest test AUC scores of 0.8093 and 0.8097, respectively. The tree-based models (excluding AdaBoost) demonstrated remarkably close performance, with test AUC scores ranging between 0.80 and 0.81. On the other hand, the most straightforward and interpretable model, logistic regression, achieved a test AUC of approximately 0.79, which was only slightly lower than the best-performing model. The results suggested that Tree-based models such as Extra Trees and XGBoost were particularly well-suited for this problem due to their ability to capture complex interactions and non-linear relationships within the data. Additionally, these models demonstrated superior performance in terms of prediction accuracy. However, logistic regression offers the advantage of simplicity and interpretability, making it a valuable alternative when these factors are prioritized in practice. Ultimately, the choice of model depends on the specific context and the trade-offs between accuracy and interpretability that are deemed most important for the given application.

In addition to AUC, we evaluated the models using other performance metrics, such as accuracy and F1 score, to provide a more comprehensive understanding of their performance. We noted that models display varying levels of performance across the different metrics. For instance, XGBoost had the highest AUC on the test set (0.81) but had the lowest F1 score (0.62). While Tree-based models achieved significantly higher accuracy than Logistic Regression and Linear Discriminant Analysis, indicating their ability to classify instances correctly, the lower F1 scores suggested difficulties in handling the minority class. This also emphasizes the importance of selecting a model based on the task's specific requirements and the dataset's nature.

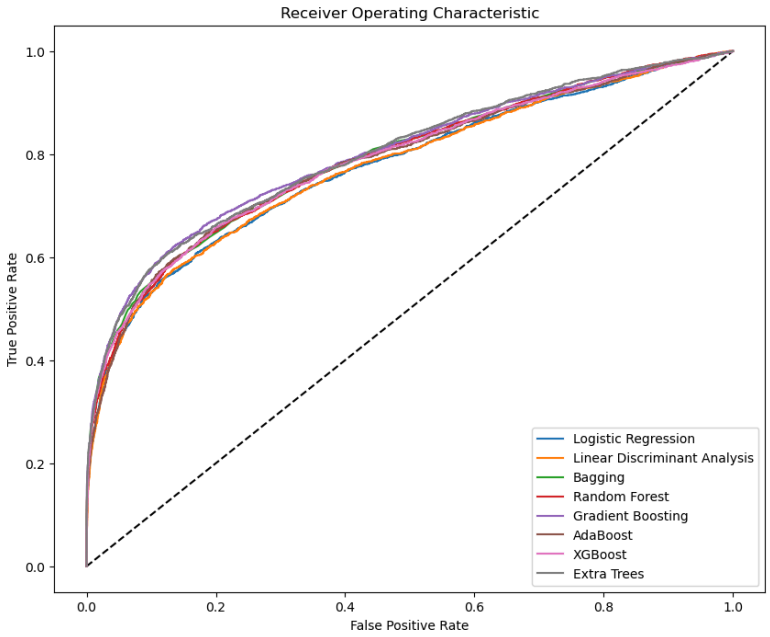
Table 5

Performance metrics for eight selected machine learning models.

Model	Logistic Regression	Linear Discriminant Analysis	Bagging	Random Forest	Extra Trees	Gradient Boosting	AdaBoost	XGBoost
AUC (valid)	0.8029	0.8009	0.7934	0.8000	0.8017	0.8014	0.7865	0.8058
AUC (test)	0.7886	0.7829	0.8003	0.8059	0.8093	0.8019	0.7897	0.8097
Accuracy (valid)	0.7329	0.7302	0.8690	0.8703	0.8735	0.8737	0.8616	0.8575
F1-score (valid)	0.7321	0.7292	0.6798	0.7197	0.7158	0.7163	0.6612	0.6233

Figure 2

ROC-AUC plot for eight selected machine learning models.



4.1.4. Model Ensembling

To further improve the performance of our models, we explored advanced ensemble techniques using the top-performing tree-based models: Extra Trees, Gradient Boosting, XGBoost, and Random Forests. Ensembling involves combining the predictions of multiple models to produce a final prediction, often resulting in improved accuracy and reduced overfitting (Ren et al., 2016).

Initially, we ensembled these models with equal weighting to determine the optimal model combination. Combining Extra Trees and Gradient Boosting yielded the best results when ensembling two models. At the same time, the optimal three-model ensemble consisted of Extra Trees, Gradient Boosting, and XGBoost, achieving cross-validated AUC scores of 0.8072 and 0.8087, respectively (Table 6). Ensembling all four models resulted in a lower AUC score compared to the previous two combinations.

Table 6

AUC scores (valid) for different ensembled models. ET=Extra Trees, GB=Gradient Boosting, XGB=XGBoost, RF=Random Forest.

Ensembled Model (2)	ET + GB	ET + XGB	ET + RF	GB + XGB	GB + RF	XGB + RF
AUC (valid)	0.8072	0.8056	0.8006	0.8044	0.8038	0.7993
Ensembled Model (3)	ET + GB + XGB	ET + GB + RF	ET + XGB + RF	GB + XGB + RF		
AUC (valid)	0.8087	0.8066	0.8024	0.8053		
Ensembled Model (4)	ET + GB + XGB + RF					
AUC (valid)	0.8071					

Next, we adjusted individual model weightings by grid search to optimize the performance of the two ensemble models mentioned above. The optimal weights for the Gradient Boosting + Extra Trees ensemble were 0.5 + 0.5, resulting in AUC scores of 0.808 and 0.813 on the validation and test set, respectively. The Extra Trees, Gradient Boosting, and XGBoost ensemble demonstrated similar performance with optimal weights of 1/9 + 1/9 + 7/9, achieving AUC scores of 0.809 and 0.812 on the validation and test set, respectively. This analysis revealed that model ensembling significantly increased AUC scores (approximately 0.005) on both validation and test sets. In conclusion, we enhanced the performance in predicting hospital readmission by leveraging advanced ensembling techniques and carefully optimizing model weightings. This approach demonstrated the potential of combining multiple models to achieve more accurate and robust predictions in healthcare settings.

Figure 3

AUC scores (valid) for Extra Trees + Gradient Boosting w.r.t different weightings.

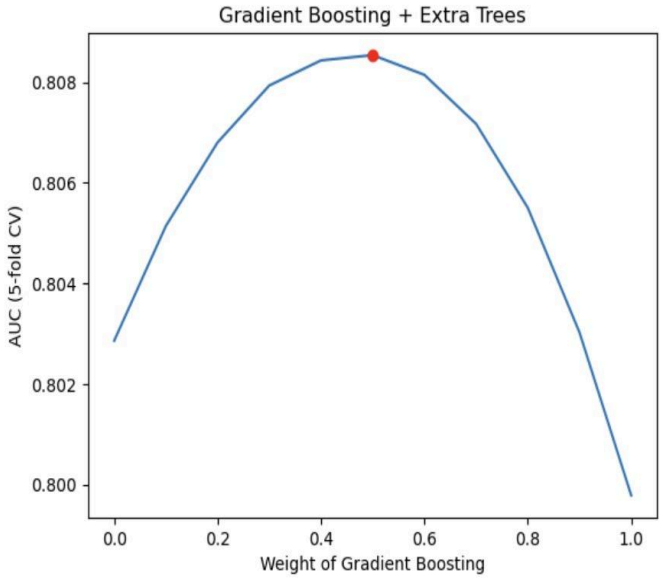
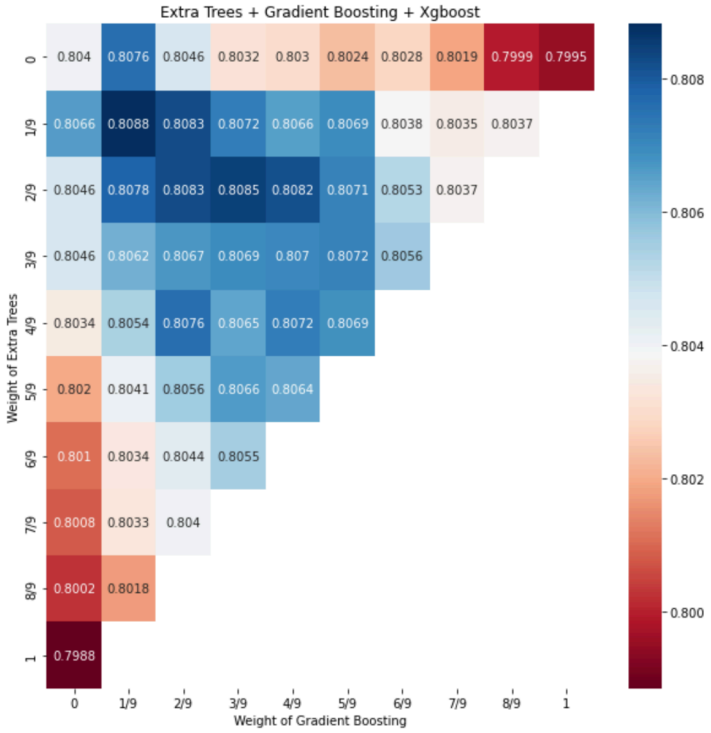


Figure 4

AUC scores (valid) for Extra Trees + Gradient Boosting + XGBoost w.r.t different weightings.



4.2. Deep Learning Models

In this section, we will explore the performance of multiple major deep-learning models on the 30-day all-cause hospital readmission prediction. Deep learning models have great potential to capture intricate patterns in high-dimensional data and surpass traditional machine learning models in many complex tasks (Dargan et al., 2020). We employed five deep learning models, including Feedforward Neural Networks (FNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory Networks (LSTM), and Transformer models.

4.2.1. Model Description

The deep learning models FNN, CNN, RNN, LSTM, and Transformers share a common foundation: they are all built upon layers of artificial neurons or nodes, which mimic the neural structures of the human brain. These neurons process input data and pass information through activation functions to learn hierarchical representations. The depth of these models, often comprising many such layers, enables the extraction of high-level, complex abstractions from data.

- Feedforward Neural Networks (FNN) are the simplest artificial neural network architecture type. In FNN, information moves in only one direction—forward—from the input nodes, through the hidden nodes (if any), and finally to the output nodes. There are no cycles or loops in the network.
- Convolutional Neural Networks (CNN) are specialized for processing data, such as images, with a grid-like topology. CNNs employ a mathematical operation called convolution and have proven to be highly effective in image recognition and classification tasks.
- Recurrent Neural Networks (RNN) are distinguished by their "memory" as they process data sequences. They can use their internal state (memory) to process variable-length sequences of inputs. This makes RNNs ideal for tasks like language modeling and translation.
- Long Short-Term Memory Networks (LSTM) are a type of RNN that are designed to remember information for long periods. LSTMs effectively overcome the vanishing gradient problem, enabling them to capture long-range dependencies in sequence data.
- Transformers rely on an attention mechanism to draw global dependencies between input and output. The Transformer's ability to handle sequential data without requiring sequential operation processing makes it exceptionally efficient for various tasks.

Compared to the traditional machine learning models such as Logistic Regression, LDA, QDA, Naive Bayes, and deep learning models can adapt to a wider range of data structures and use more data types for model training. Deep learning models are likely to perform better for the EHR data having 160 or even more than 400 variables. To conclude, the capability to capture

temporal dynamics, process high-dimensional data, and efficient computation are the motivations behind our choice to use these deep learning methods.

4.2.2. Model Fitting

Based on the feature subset that previously yielded the most promising preliminary results, namely "latest+mean+sd+min+max," the first step in the model-fitting workflow is establishing the model architecture. In this step, we designed the number of neural network layers, featured model architectures, and batch structures. For instance, the Transformer model was constructed with self-attention layers to handle sequential input. At the same time, the LSTM was architected with memory cells to manage long-term dependencies in the data. During this phase, we also explored computational enhancements using CUDA (Compute Unified Device Architecture) for Nvidia GPUs and MPS (Metal Performance Shaders) for Apple's GPUs. These technologies facilitate hardware acceleration on their respective platforms, enabling more efficient model inference and further training through parallel processing.

The subsequent model training phase highlights the tuning of each model's parameters, supplemented by hyperparameter optimization techniques. The goal was to intricately tailor each model to the dataset and unearth the optimal performance for the given task.

After rigorous training, tuning, and evaluation, the performances of various deep-learning models are recorded as follows:

- Transformer: exhibited a credible AUC of 0.7834 on the validation set and an AUC of 0.775 on the test set.
- Bidirectional LSTM: achieved an AUC of 0.7614 on the validation set, with a slightly lower AUC of 0.7351 on the test set.
- Feedforward Neural Network (FNN): demonstrated an AUC of 0.7222 on the test set.
- Recurrent Neural Network (RNN): presented an AUC of 0.7028 on the test set.
- Convolutional Neural Network (CNN): Unfortunately, the CNN model did not converge to a satisfactory AUC score, indicating that the model was not well-suited to the input data structure.

These outcomes highlight the strengths and limitations of each deep learning architecture when applied to healthcare data. The Transformer model emerged as the most proficient, likely due to its capacity to handle long sequences and attention mechanism, which is well-suited for capturing the dependencies and relationships within the data. The LSTM's ability to record information over extended time frames proved valuable, although it trailed slightly behind the Transformer in performance. The FNN and RNN, with their more superficial structures, offered moderate success but were outperformed by the more advanced models. CNN's underwhelming performance underscores the importance of model alignment with data characteristics for successful predictions.

4.2.3. Discussions

In applying deep learning models, exciting patterns emerged in their performance. These models are based on neural network architectures and display commendable predictive capabilities but fall short of the benchmarks set by tree-based models. Notably, when the best-performing Tree-based models were ensembled, their combined predictive power outperformed the individual deep-learning models.

Among the deep learning models assessed, Transformer performs satisfactorily on both the validation and prediction sets. Moreover, the Transformer introduces attention mechanisms and features long-distance context attention compared to other models. Therefore, we introduced the Bidirectional LSTM, which can understand the context data in both directions. Additionally, in the comparison, we found that the short-term dependency of the data sequence is more prominent than the long-term dependency, and the bidirectional model is better than the unidirectional attention model. The bidirectional Transformer, also known as Transformer-XL, gave the best AUC score on the validation set, which confirmed this finding. This result underscores the value of understanding past and future contexts in predicting readmission.

In the comparison between RNN and FNN models, the FNN model surprisingly yielded better results despite the lack of contextual linking. This aligns with the intuitive understanding that each hospital admission record is independent, and the sequence of admissions—what comes before or after—does not influence the likelihood of a 30-day readmission. At times, the FNN even outperformed the RNN under the same batch size and epoch number, suggesting that focusing on the sequence of admissions could potentially harm the model's performance. This aligns with the understanding that hospital admission records are not sequential data like stock prices or temperature readings, from which models can learn valuable patterns.

The fact that deep learning models, renowned for their success in tasks involving complex patterns and sequences, did not yield the best results in our study points to a significant limitation: the lack of interpretability. The inability to decode the reasons behind their predictions makes it challenging to fully trust and understand these models, especially in the high-stakes field of healthcare. This stands in contrast to our initial expectations and serves as a reminder of the importance of model interpretability—mainly when dealing with medical datasets, where understanding the reasoning behind predictions is crucial.

Moving forward, our learnings from this phase of model fitting will guide us in refining our approach, balancing the trade-off between predictive power and interpretability, and aligning model selection with the inherent nature of the healthcare data we seek to analyze.

5. Model Interpretation

By understanding the impact of each feature value on the likelihood of readmission, hospitals can formulate more effective treatment plans, improve the speed of patient recovery, and reduce medical costs (Kroch et al., 2016). This section will use feature importance and partial dependence plots to interpret the Extra Trees and Logistic Regression.

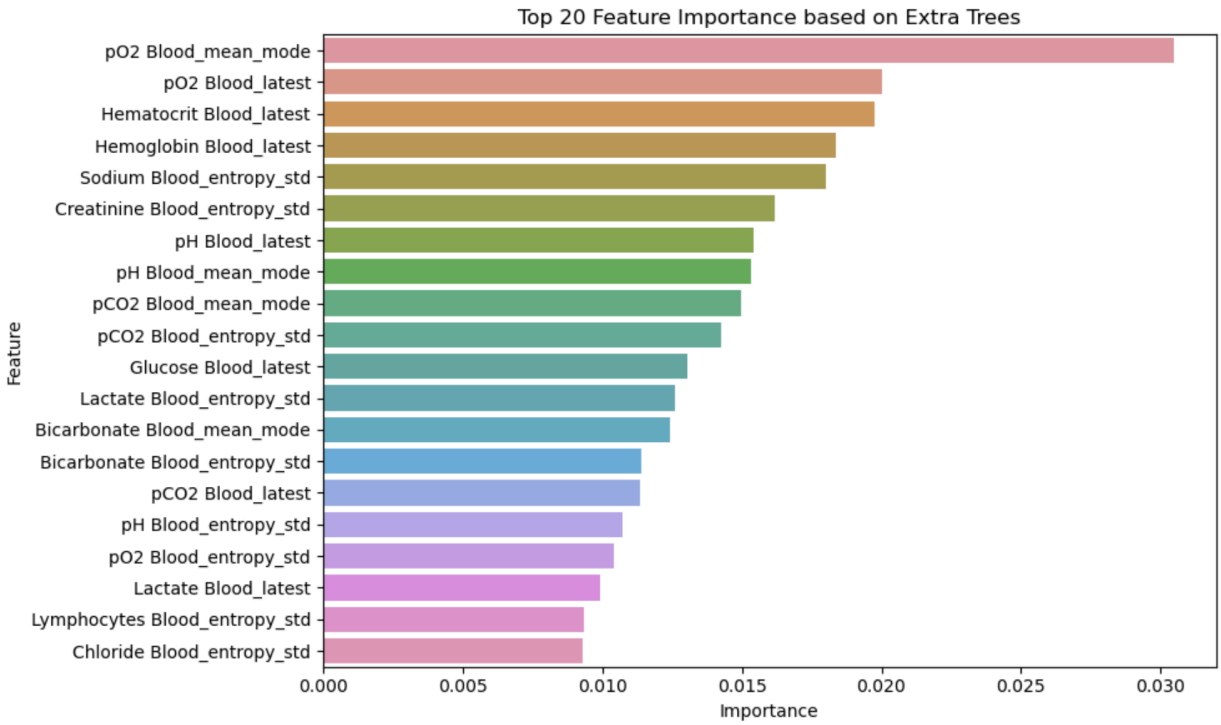
5.1. Feature Importance

5.1.1. Extra Trees

As a Tree-based model, Extra Trees could directly compute future importance through the 'sklearn' library. According to Figure 5, the variable PO2_Blood_mean_mode test result dominated the effect, indicating an outrange value could be relatively more significant while measuring health. The result corresponds to the finding in Table 1 in section 3.1.1. "pO2 Blood_mean_mode exhibits a higher correlation with the readmission". It is followed by Hematocrit and Hemoglobin Blood test results.

Figure 5

Top 20 Feature Importance based on Extra Trees



5.1.2. Logistic Regression

Although the tree-based models exhibited considerably high prediction accuracy, they are relatively weak in interpretability compared to logistic regression. Since the topic aims to predict if readmission would happen and Logistic Regression uses a logistic function to model a binary dependent variable, it is particularly well-suited for binary classification problems. Hence, we also analyzed the results from Logistic Regression for comparison purposes.

The classification report provides key metrics in assessing the performance of Logistic Regression. Table 7 shows the model's accuracy is 0.86, which correctly predicts the class for 86% of instances. However, the macro-average F1-score (which gives equal weight to each class) is 0.69, and the weighted-average F1-score (which gives each class a weight proportional to its size) is 0.84. While the model performs well overall, it needs help with class 1 instances. Thus, the Extra Tree model is chosen rather than Logistic Regression as it performs better.

Table 7

Classification report of Logistic Regression.

	precision	recall	f1-score	Number of occurrences
0	0.8736	0.9772	0.9225	9564
1	0.7572	0.3346	0.4641	2032
accuracy	0.8646	0.8646	0.8646	N/A
macro avg	0.8154	0.6559	0.6933	11596
weighted avg	0.8532	0.8646	0.8421	11596

Table 8 illustrates the coefficients in Logistic Regression, which represent the change in log odds for a one-unit change in the corresponding feature. A positive coefficient indicates that an increase in the feature value increases the odds of a positive outcome. A more significant absolute coefficient value indicates a more substantial effect on the outcome.

Table 8

Top 20 Feature Importance based on Logistic Regression.

Feature	Coefficient	Absolute Coefficient
Lymphocytes Pleural_latest	2.7710	2.7710
Eosinophils Joint Fluid_entropy_std	2.5811	2.5811
G30-G32_latest	-2.5437	2.5437
G30-G32_mean_mode	-2.5437	2.5437
Eosinophils Pleural_mean_mode	-2.2480	2.2480
E50-E64_latest	-1.9631	1.9631
E50-E64_mean_mode	-1.9631	1.9631
Lactate Blood_latest	-1.9199	1.9199
F20-F29_mean_mode	-1.8767	1.8767
F20-F29_latest	-1.8767	1.8767
N17-N19_mean_mode	1.7307	1.7307
N17-N19_latest	1.7307	1.7307
Eosinophils Ascites_latest	1.7153	1.7153
Basophils Blood_mean_mode	1.7135	1.7135
Y90-Y99_mean_mode	-1.6628	1.6628
Y90-Y99_latest	-1.6628	1.6628
Lymphocytes Pleural_entropy_std	1.6500	1.6500
Eosinophils Other Body Fluid_latest	-1.5670	1.5670
G35-G37_latest	-1.5597	1.5597
G35-G37_mean_mode	-1.5597	1.5597

The sign of the coefficient of Logistic Regression could be used to analyze the reliability in reality. Take the features with the highest positive and negative coefficients as examples: Lymphocytes Pleural_latest and G30-G32_latest.

The change in log odds for a one-unit change in Lymphocytes Pleural_latest is 2.7710. According to Mercer et al. (2019), pleural fluid accumulation disrupts the balance between production and reabsorption. In healthy individuals, the pleural cavity contains approximately 0.3 mL/kg of fluid. A pleural effusion occurs either when production exceeds reabsorption or

when the reabsorption mechanisms have been disrupted, the latter being more common. A higher number leads to a more dangerous situation for the patient. It shows that the coefficient result is corresponding to reality.

The change in log odds for a one-unit change in G30-G32 is 2.5437. Neurological conditions were identified through medical history and linkage to data on hospital admissions (ICD-10 code G00–G99). ICD10data (2023) states that G30-G32 refers to other nervous system degenerative diseases. For example, G31.83 refers to neurocognitive disorder with Lewy bodies. A neurodegenerative disease is marked by Lewy body cells in the cerebral cortex and brainstem. Symptoms often include dementia, parkinsonism, and striking fluctuations in cognitive performance. Lewy body disease usually begins between the ages of 50 and 85. The disease gets worse over time, and there is no cure. Treatment focuses on drugs to help symptoms. As the project aims to predict readmission within 30 days, it is relatively short compared to the clinical period of degenerative diseases. Patients tend to be treated with drugs instead of physical treatment in hospitals. Thus, the model result corresponds to the reality that a patient with G30-G32 disease is relatively unlikely to be readmitted to the hospital.

5.1.3. Common Important Features

In machine learning, different models often assign different levels of importance to features when making predictions. Ideally, we expect to see many standard features considered necessary across various models, as this suggests a strong consensus about the predictive power of these features. However, it is common to find that only one or two features are commonly identified as important across different models. This discrepancy can be attributed to the different ways that models handle data and make predictions.

Comparing the importance of extra trees and logistic regression, lactate blood is the common feature. While both methods provide a measure of feature importance, they work in different ways and provide different types of information. In Tree-based models like Extra Trees, feature importance is typically calculated based on the average reduction in the Gini impurity or entropy brought by a feature. Features that tend to split the data into purer nodes have higher importance. This method of determining feature importance is based on the structure of the model itself and does not provide any direction for the relationship. Unlike tree-based models, logistic regression is important if it has a strong linear relationship with the target variable. The coefficients provide the direction of the relationship. Therefore, the difference in standard essential features might be due to the distinct methods to calculate the feature importance in Extra Trees and Logistic Regression by the nature of different models.

Additionally, various versions of the same blood test contribute to the significant difference in feature significance as the training dataset consists of different scales of parameter transformation. For example, PO2_mean_mode and PO2_blood_latest rank first and second,

respectively, according to Figure 5. After removing the suffix of variables, it would be straightforward to see which individual blood test is more influential to the dependent variable. The result indicates that PO2 Blood is a common significant feature of the two models, corresponding to the most crucial feature identified in Figure 5.

5.2. Partial Dependence

Partial dependence is a method for interpreting machine learning models that shows the marginal effect one or two features have on the predicted outcome of a machine learning model. It is based on visualizing the average effect of the values of a particular feature by marginalizing all other features in the feature set. One-way partial dependence (PDP) tells the interaction between the response and a feature, where X is the value of the individual feature, and y represents the model's predicted outcome. These interpretations are marginal, considering a feature at one time. Take the most significant variable, PO2 blood_mean_mode, as an example.

According to Figure 6, the partial dependence plot shows that the feature has a non-linear relationship with the predicted outcome. The sharp increase from 0 to 10 on the x-axis suggests that small increases in the pre-processed PO2 value within this range significantly increase the likelihood of readmission. This could indicate that deficient PO2 levels (well below the normal range) are associated with a high risk of readmission, possibly due to severe health conditions that these low PO2 levels might indicate.

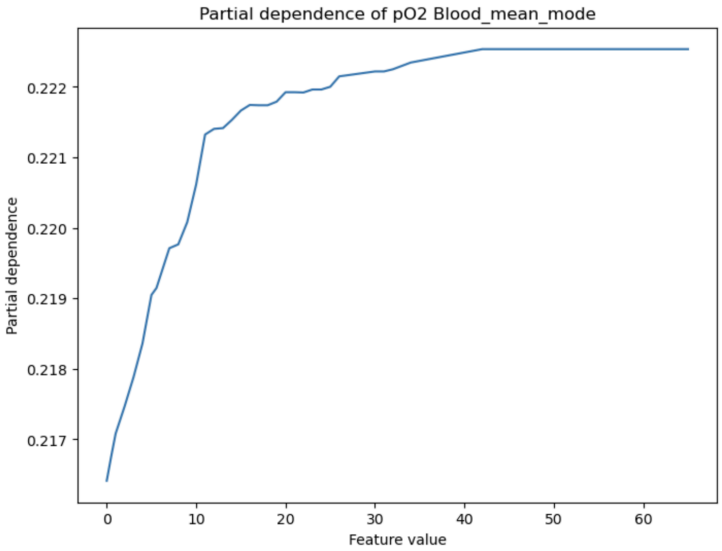
After the feature value of 10, the rate of increase in the predicted outcome slows down. This suggests that further increases in the PO2 value still increase the likelihood of readmission, but this effect is less pronounced. This range might correspond to the lower end of the normal PO2 range.

The line stabilizes around a partial dependence of 0.223 and becomes a straight line towards the end of the x-axis. This indicates that beyond a particular PO2 value, further increases in the PO2 value have little to no effect on the likelihood of readmission. This point might correspond to the upper end of the normal PO2 range. The feature has a strong positive effect on the predicted outcome up to a certain point, after which the effect diminishes. This suggests that there may be a threshold or saturation point beyond which further increases in the feature value do not significantly increase the predicted outcome.

Overall, feature importance is crucial to model interpretation and understanding as it identifies the most influential features in the model's predictions. Different models calculate feature importance differently. It could provide insights into feature selection and the underlying processes that generated the data to understand the corresponding domain.

Figure 6

Partial dependence plot of PO2 blood_mean_mode.



6. Limitations and Scopes of Improvements

This section elaborates on our model training and fitting limitations, especially in deep learning models. We will also give the corresponding t for improvement regarding the limitations.

6.1. Computational Power Limitation in Hyperparameter Tuning

One of our primary limitations in the current study was the need for sufficient computational power to conduct extensive hyperparameter tuning, which includes methods like grid search and Bayesian optimization. For instance, attempting a grid search on RNN models using GPU took 8 hours, and no conclusive results were obtained. The optimal AUC scores we have reported thus far were achieved through manual tuning of crucial parameters, which could be better. The visualization of tuning results for FNN and RNN across three parameters—epoch size, batch size, and learning rate—highlights the nuanced impact of these hyperparameters on model performance (Figure 7 & Figure 8).

To address this, a straightforward solution is to increase our GPU computing power. However, this is a linear solution to an exponentially growing demand for computational resources. A more sustainable approach is to learn how to navigate the hyperparameter space more efficiently. Techniques such as transfer learning, few-shot learning, or meta-learning could be explored to reduce the reliance on brute-force computational strength.

Figure 7

Visualization of FNN model parameter tuning and AUC score

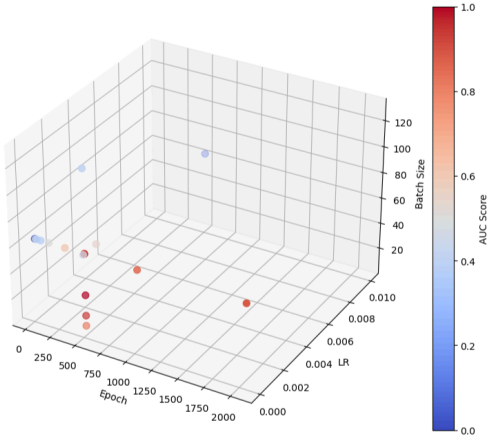
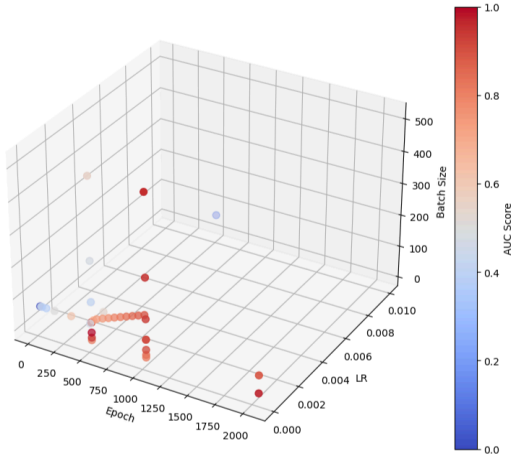


Figure 8

Visualization of RNN model parameter tuning and AUC score



6.2. Non-Convergence of CNN Models

The CNN models in our study did not converge. We could not resolve the issue despite attempts to adjust learning rates and batch sizes. To facilitate the ease of modifying the number of hidden layers in our data grid design, we introduced several columns of NULL data, which may have contributed to the non-convergence. Additionally, the dimensional design of the input, hidden, and output layers could also be a factor in the model's failure to converge.

To solve this problem, future work will involve thoroughly reviewing the data grid design and eliminating NULL placeholders to enhance model stability. Redesigning the architecture to ensure compatibility between the dimensions of different layers is also crucial. Moreover,

applying normalization techniques and exploring advanced initialization methods could mitigate the convergence issues.

6.3. Incorporation of Attention Mechanisms and Advanced Models

Our current models, while proficient in certain areas, could be further improved by incorporating attention mechanisms within the RNN framework. Attention mechanisms can allow a model to focus on specific parts of the input sequence, which is beneficial for capturing relevant patterns.

We can explore integrating attention mechanisms into RNNs to enhance their performance. Furthermore, experimenting with pre-trained models such as BERT, which comes with built-in bidirectional attention mechanisms, could be promising. Although BERT is traditionally used in text processing tasks, its potential for generalization makes it a candidate worth exploring for the classification task of predicting hospital readmissions within 30 days.

In conclusion, while our study has significantly applied machine learning techniques to predict hospital readmissions, these limitations highlight the need for continuous improvement. Advancements in computational strategies, model architecture design, and the exploration of sophisticated pre-trained models are among the avenues through which we can enhance the accuracy and efficiency of our predictive models.

6.4. Additional Predictors

To improve the readmission probability analysis further, we propose including new predictors, such as length of stay and previous number of admissions. The length of stay is calculated by computing the difference between admittance and discharge times, and the number of previous admissions for each patient is obtained by aggregating the readmission occurrences. These predictors could provide valuable insights into the severity of a patient's condition and health history, potentially influencing the likelihood of readmission. By incorporating these predictors or exploring other relevant predictors, it is expected that we can enhance the datasets and enable a more comprehensive analysis of readmission probability.

7. Conclusion

In summary, our project aimed to predict 30-day all-cause hospital readmissions using machine learning models. Through an extensive data exploration and feature engineering process, we identified crucial features and their distribution patterns, tackled class imbalance, and chose the optimal subset of features to enhance our models. We assessed a total of 18 models and determined that XGBoost, Extra Trees, and Gradient Boosting were the most promising in terms of performance metrics, specifically AUC scores. After training individual models, we also showcased the potential of model ensembling to enhance further model performance, where Extra Trees + Gradient Boosting appeared to be the best ensemble. Regarding individual models, the Extra Trees model emerged as the most interpretable and accurate, enabling us to pinpoint the

most vital features contributing to readmission prediction. Furthermore, we delved into the partial dependence of selected features, offering valuable insights into the relationships between individual features and the predicted response. Despite the satisfactory performance of our models, we recognize that continuous improvement in computational strategies, model architecture design, and the incorporation of additional predictors will further enhance the accuracy and efficiency of our predictive models in future work. In conclusion, our project showcased the potential of machine learning techniques in predicting 30-day hospital readmissions, which could have significant implications for medical decision-making and resource allocation in healthcare systems. By identifying patients at a higher risk of readmission, healthcare providers can implement targeted interventions and strategies to improve patient outcomes and alleviate the strain on healthcare systems.

References

Alanazi, A. (2022). Using machine learning for healthcare challenges and opportunities. *Informatics in Medicine Unlocked*, *30*, 100924.

Casalini, F., Salvetti, S., Memmini, S., Lucaccini, E., Massimetti, G., Lopalco, P. L., & Privitera, G. P. (2017). Unplanned readmissions within 30 days after discharge: improving quality through easy prediction. *International Journal for Quality in Health Care*, *29*(2), 256-261.

Callahan, A., & Shah, N. H. (2017). Machine learning in healthcare. In *Key advances in clinical informatics* (pp. 279-291). Academic Press.

Dargan, S., Kumar, M., Ayyagari, M. R., & Kumar, G. (2020). A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, *27*, 1071-1092.

Elreedy, D., & Atiya, A. F. (2019). A comprehensive analysis of synthetic minority oversampling technique (SMOTE) for handling class imbalance. *Information Sciences, 505*, 32-64.

Guo, C., Lu, M., & Chen, J. (2020). An evaluation of time series summary statistics as features for clinical prediction tasks. BMC medical informatics and decision making, 20(1), 1-20.

ICD10Data.com. (2023). G31.83 - Dementia with Lewy bodies. Retrieved from https://www.icd10data.com/ICD10CM/Codes/G00-G99/G30-G32/G31-/G31.83

Johnson, A. E., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., ... & Mark, R. G. (2023). MIMIC-IV, a freely accessible electronic health record dataset. *Scientific data, 10*(1), 1.

Kroch, E., Duan, M., Martin, J., & Bankowitz, R. A. (2016). Patient factors predictive of hospital readmissions within 30 days. *The Journal for Healthcare Quality (JHQ)*, *38*(2), 106-115.

Leppin, A. L., Gionfriddo, M. R., Kessler, M., Brito, J. P., Mair, F. S., Gallacher, K., ... & Montori, V. M. (2014). Preventing 30-day hospital readmissions: a systematic review and meta-analysis of randomized trials. *JAMA internal medicine*, *174*(7), 1095-1107.

Mercer, R. M., Corcoran, A. J. P., Porcel, J. M., Rahman, N. M., & Psallidas, I. (2019). Interpreting pleural fluid results. *Clinical Medicine (London)*, 19(3), 213-217

Ren, Y., Zhang, L., & Suganthan, P. N. (2016). Ensemble classification and regression-recent developments, applications and future directions. IEEE Computational Intelligence Magazine, 11(1), 41-53.