Machine Learning-Based Prediction of Patient Survival Outcomes

Xiwen Feng, Claire Kong and Leyuan Qian

2022-12-17

Abstract

Objective Predicting the probability of patient survival in intensive care units (ICU) can effectively improve the utilization of ICU and help to prioritize hospital admission for patients with more urgent needs. Thus, we attempted to find an optimal model to predict the probability of survival of these patients. **Method** Missing data was imputed using the mean of each variable, and categorical variables were transformed into dummy variables using one-hot method. Then, three machine learning models, Random Forest, K-Nearest Neighbors (kNN), and Extreme Gradient Boosting (XGBoost) were trained and tested for predictive purposes. We used five diagnostics to evaluate these models, including accuracy, recall, precision, F1 score, and AUC-ROC. **Results** We found that amongst the three imbalanced implementations, XGBoost performed a lot better than the rest. However, balanced random forest did generate the most accurate model, with a balanced accuracy of 0.87 and a recall of 0.818.

Introduction

In ICU, doctors would need to pre-classify patient background information and judge whether a patient will survive based on their experience [5]. However, incompletion of patient data adds up to the difficulty in predicting ICU mortality efficiently and accurately. Previous studies have explored machine learning approaches to analyze clinical data and predict patient survival outcome [10]. Our study aims to apply different machine learning algorithms on real-world clinical data to help healthcare professionals optimize their patient management strategies.

Data Description

The data investigated in this study is a subset of two large multicenter datasets from Australian and New Zealand Intensive Care Society Adult Patient Database and the United States' eICU Collaborative Research Database, containing information of 249,229 and 131,051 patients, respectively.

This dataset contains 91,713 observations and 186 variables. The outcome variable we worked with is hospital death, a binary variable indicating whether a patient died during hospitalization. Excluding this response variable, IDs and the all-zero variable readmission_status, we have 158 continuous and 21 categorical variables. These variables come from 10 major categories, ranging from demographics, vital signs, lab values, to comorbidities. Overall, the data is very messy because 75 of the continuous variables exhibited severe multicollinearity by having correlations above 0.9. Moreover, our outcome class is imbalanced given its survival-death ratio being 91:9, meaning that those who survive appear 9 times more often than those who do not.

Methods

Data Processing

First, we used the one-hot encoding method to transform the categorical features. Then, all-zero and irrelevant variables, such as patient IDs and hospital IDs, were removed. Given the large amount of missing values, we decided to impute continuous variables using their means. In the end, we were left with 232 variables. The processed dataset was then split into two parts, with 75% being the training data and the rest 25% being the test data.

Statistical Analysis

We used three different algorithms for prediction: kNN, XGBoost, and Random Forest. The first two were implemented in R and the last in Python. After model tuning, we used five diagnostic measures to compare results from the three methods: recall, F1 score, AUC-ROC, precision, and accuracy. F1 score is one of our key indicators, as this evaluation metric elegantly summarizes the predictive performance of a model by combining two competing metrics - precision and recall.

K-Nearest Neighbors

The k-nearest neighbors (kNN) algorithm is a widely used classification algorithm in machine learning [7]. It is recognized for its simplicity, as it only has few hyperparameters and is based on a general assumption that observations with similar features are likely to have similar responses [6]. The implementation of the kNN algorithm in this study follows a conclusive instruction made by Zhang [9]. First, the data was normalized using min-max scaling. Subsequently, we used a range of k values to train the model, and k = 7 outputted the best accuracy, specificity, and other diagnostic scores. Using the kNN model with k = 7, the outcome of the test dataset was predicted and compared with the actual outcome.

XGBoost

Extreme Gradient Boosting provides one of the fastest implementations of gradient boosting trees. It generates a prediction model in the form of an ensemble of weak prediction models, or in this case, decision trees [1]. Additionally, XGBoost is able to perform cross-validation as well as provide estimates of variable importance from the trained model [8].

With some trial and error, we chose to do a 5-fold cross validation, setting the maximum tree depth to 6 and number of rounds to 20, in an attempt to obtain a more robust model. In fact, the training and testing errors continued to decrease as we increased the number of rounds. However, there wasn't enough of a difference to justify the lengthy computational time given the size of our dataset.

We also examined the features of importance that XGBoost outputted (plot can be found in the Figures folder on Github). Since there were over 200 variables, only ones with more importance were featured in the plot. In particular, we observed that d1_lactate_min, or "the lowest lactate concentration for the patient in their serum or plasma during the first 24 hours of their unit stay", was significantly more important than all other predictors. Moreover, d1_sysbp_min, or "the patient's lowest systolic blood pressure during the first 24 hours of their unit stay, either non-invasively or invasively measured", was also one of the predominant predictors.

Random Forest

Random forest is a machine learning method that exploits ensemble learning principles. In classification mode, it trains many decision trees and uses voting to provide a final decision, where for each decision tree, it will generate decisions by enumerating conditions related to the decision on the trees. The benefit of random forest is that it does not overfit [2]. Furthermore, ensemble learning can make several weak classifiers collaborate into a strong classifier.

We also examined the feature's importance and plotted the decision tree. Please refer to the submitted code for figures.

Results

Measures	Imbalanced KNN	Imbalanced XGBoost	Imbalanced Random Forest	Balanced Random Forest
Recall	0.0512	0.329	0.025	0.818
F1 Score	0.0932	0.444	0.049	0.518
AUC-ROC	0.523	0.657	0.513	0.846
Precision	0.521	0.683	1.0	0.379
Accuracy	0.915	0.930	0.917	NA
Balanced Accuracy	0.523	0.657	NA	0.870

Along with Accuracy, there is a Balanced Accuracy measure for the first two columns above. This is included because it is a better metric to refer to than just accuracy when dealing with imbalanced outcome classes, as is the case for our dataset [3]. We note that balanced accuracy values were lower than their corresponding accuracy; however, they do manage to better capture model performances.

The performance of kNN analysis is not ideal according to the metrics, especially the F1 score. The survival cases were well predicted, which accounts for the high accuracy. In contrast, the recall (sensitivity) is relatively low, reflecting a poor performance on identifying death cases from the population who indeed died during the ICU stay. This result is attributed to the imbalance of our data. Manually balancing the data to make the ratio of survival and death about 1:1 would allow the model to have a more balanced sensitivity and specificity.

In comparison to the imbalanced kNN and random forest implementations, we remark that XGBoost performed drastically better in almost all diagnostics. This could be explained because as the most complex decision tree algorithm, it has optimized features in parallelized tree building, regularization through LASSO and Ridge to avoid overfitting, and built-in cross validation capabilities [4]. Nevertheless, it is clear that the balanced random forest outputted the much better recall, F1 score, AUC and balanced accuracy values.

Next, we know that random forest models are built on decision trees, and decision trees are sensitive to class imbalance. In our imbalanced random forest model, we noticed that the precision is 1, which is problematic. This could be due to the fact that each tree can be biased in the same direction and magnitude (on average) as a result of class imbalance. Consequently, we implemented weighted bootstrap to address the issue. We found that the balanced random forest results showed a significant improvement in most diagnostics compared to the imbalanced random forest and other methods.

Conclusion & Limitations

In the end, XGBoost performed better than the other two imbalanced machine learning algorithms we carried out. However, the balanced random forest procedure produced much better diagnostics all around, except for precision. This suggests that balancing our binary outcome class would, in general, enhance the accuracy of resulting models and make more correct predictions.

In fact, the kNN algorithm did not fit our data well because of the imbalance of data and "the curse of dimensionality." Class balancing techniques help to solve the problem of imbalanced data, but we should be cautious, as artificial deletion or addition of the observations in class balancing techniques may not generalize to the original real-world dataset. As well, with access to a more powerful computer in the future, we would run many more rounds when performing cross validation in XGBoost to obtain the lowest training and testing errors possible.

It is also feasible that our imputation method by mean value significantly impacted these algorithm's ability to fit a more predictive model. This calls for comparing and contrasting applying other imputation methods in the future, such as using the built-in data imputation features of various machine learning algorithms. The downside to this is that although resulting models may be more accurate, we may not be able to compare across methods because the algorithms are using self-generated imputation models. Finally, since we weren't able to find out how our specific dataset was extracted from the larger dataset, we acknowledge that there could be certain selection biases that may impact our subsequent analyses.

Github:

https://github.com/clairekwl/625FinalProject

Author Contributions

- Claire Kong: XGBoost
- Xiwen Feng: Random Forest
- Leyuan Qian: kNN

Works Cited

[1] Banerjee, Prashant. “XGBoost + k-fold CV + Feature Importance.” Kaggle. Accessed 15 December 2022.

[2] Breiman, Leo. “Random forests.” Machine learning 45.1 (2001): 5-32.

[3] Hale, Jeff. “The 3 Most Important Composite Classification Metrics | by Jeff Hale.” Towards Data Science, 20 May 2020. Accessed 15 December 2022.

[4] Khah, Leo Soheily. “Why does XGBoost work so well? | Data Science and Machine Learning.” Kaggle. Accessed 17 December 2022.

[5] Li, Zhuoying. “Study of ICU Mortality Prediction and Analysis based on Random Forest.” Proceedings of the 7th International Conference on Cyber Security and Information Engineering. 2022.

[6] Richman, Joshua S. “Chapter Thirteen - Multivariate Neighborhood Sample Entropy: A Method for Data Reduction and Prediction of Complex Data.” METHODS IN ENZYMOLOGY, Elsevier, 2011, pp. 397-408. Elsevier.

[7] Sun, Bo, Junping Du, and Tian Gao. “Study on the improvement of K-nearest-neighbor algorithm.” 2009 International Conference on Artificial Intelligence and Computational Intelligence. Vol. 4. IEEE, 2009.

[8] “What is XGBoost? Why is it so Powerful in Machine Learning.” Abzooba. Accessed 15 December 2022.

[9] Zhang, Zhongheng. “Introduction to machine learning: k-nearest neighbors.” Annals of Translational Medicine, vol. 4, no. 11, 2016, p. 218. PMC.

[10] Zhang, Wenbin, Jian Tang, and Nuo Wang. “Using the machine learning approach to predict patient survival from high-dimensional survival data.” 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2016.