IDS 705 Final Report

Early Detection of Sepsis in ICU Patients

Team Members:
Gray, Mac
Lu, Dylan
Parker, Jack

Abstract

Our group set out with a goal of detecting sepsis early in ICU patients through the use of machine learning. In 2019, a competition was held which challenged contestants to accomplish this exact task. Sepsis was such an interesting topic to help fix because of how deadly, costly, and common it is. Initially, our team explored multiple models that were established for large amounts of tabular data. We landed on XGBoost (the technique favored by the top four finishers in the competition) and a new, much less well-known library called CatBoost. We aimed to see which of these models could perform better. With a difficult-to-work-with dataset, we ran into a lot of issues which ultimately required creative solutions. CatBoost emerged as the stronger of the two models for predicting sepsis and for providing tools for reporting feature importance, which is highly valuable in clinical settings.

Introduction

Sepsis is **very common**; each year it affects about 30 million people globally. Sepsis is also **very costly**; the U.S. spends approximately $24 billion annually on sepsis treatments (Reyna et al., 2020).

Thankfully, sepsis is **much less deadly and much less costly if detected early**. Significantly fewer hospital resources are needed to treat sepsis if it is caught in its early stages. Each hour of delay in detecting sepsis is associated with a 4-8% increase in the chance of death (Reyna et al., 2020).

Background

The 2019 PhysioNet/Computing in Cardiology challenge asked teams to develop computational approaches for detecting sepsis early (Reyna et al., 2020). Teams were challenged to detect sepsis six hours prior to when it was detected by hospital staff. For this project, we decided to take on this challenge ourselves.

Yang et al. (2020) won first place. They used XGBoost and tuned model hyperparameters with a Bayesian optimizer (Agnihotri & Batra, 2020). Inspired by Lundberg et al. (2020), Yang et al. placed a strong emphasis on building a model that could be easily interpreted by clinicians. To this end, they figured out which features were most important to their model when making classification decisions. This team achieved an AUC of 0.85, a sensitivity of 0.90, and a specificity of 0.64 when their model was applied to a held-out test set.

According to Lundberg et al. (2020), tree-based machine learning models are often deployed in medical contexts, and there is a strong appetite in the medical field for these models to be made more interpretable. Complex models such as ensemble models or deep learning approaches tend to achieve higher prediction accuracy compared to simpler models, but these simpler models are often favored by clinicians over the more accurate, more complex models (Lundberg et al., 2020).

One major reason for this phenomenon is that in order for clinicians to create a bespoke treatment plan for a particular patient after a model classifies the patient as at-risk for a particular health complication, it is often imperative that clinicians know which features influenced the model's decision most strongly (Lundberg et al., 2018). Simpler models tend to be more interpretable, and clinicians are often willing to

sacrifice predictive power for the ability to develop more tailor-made treatment plans for specific patients in specific circumstances.

Thankfully, it is often the case that prediction accuracy does not have to be weighed against interpretability. For instance, Letham et al. (2015) developed a very simple, interpretable model called Bayesian Rule Lists that achieves a prediction accuracy similar to the most powerful complex models when it comes to predicting strokes in hospital patients. We were strongly influenced by the approach of Yang et al. as well as the wider literature on clinicians' desire for feature importance information.

Other Notable Work

The second, third, and fourth place teams in the PhysioNet competition all used some form of feature engineering and gradient boosting. Morrill et al. (2019) used a mathematical transformation called a "signature" to perform feature engineering before feeding those features through a gradient boosting algorithm. Du et al. (2019), used a combination of a log transform and normalization to preprocess the data before boosting. Zabihi et al. (2019) extracted 407 features from the raw training data, used a feature selection algorithm to narrow down that set of features, and then ran the selected features through an ensemble of five XGBoost models. The fact that the top four finishers all used gradient boosting was a major clue to us that we should do the same.

Other groups have taken on projects very similar to the PhysioNet challenge and have also emphasized model interpretability. Nemati et al. (2018) used an easy-to-interpret model called the Weibull-Cox model to achieve a very similar AUC score to Yang et al. for a different but very similar dataset. Rosnati and Fortuin (2021) developed an attention-based (Vaswani et al., 2017) deep learning model for predicting sepsis on a very similar dataset to that used by Yang et al. and Nemati et al. The use of an attention mechanism allowed Rosnati and Fortuin (2021) to make their model more interpretable. They recovered the attention weights from the model and used them to infer which points in time and which features most informed the model's predictions.

Data

The PhysioNet challenge provided patient data from two hospital systems: Beth Israel Deaconess Medical Center (n = 20,336 patients) and Emory University Hospital (n = 20,000 patients). Each dataset contains forty features (Table 1). No names or other identifying information were provided.

***Table 1**: Descriptions of all the features provided in both the Beth Israel and Emory datasets. Official feature descriptions can be found here.*

Feature Shorthand	***Category***	***Description***
HR	Vital signs	Heart rate
O2Sat	Vital signs	Pulse oximetry
Temp	Vital signs	Temperature
SBP	Vital signs	Systolic blood pressure
MAP	Vital signs	Mean arterial pressure

DBP	Vital signs	Diastolic blood pressure
Resp	Vital signs	Respiration rate
EtCO2	Vital signs	End tidal carbon dioxide
BaseExcess	Lab test results	Excess bicarbonate
HCO3	Lab test results	Bicarbonate
FiO2	Lab test results	Fraction of expired oxygen
pH	Lab test results	N/A
PaCO2	Lab test results	Partial pressure of carbon dioxide from arterial blood
SaO2	Lab test results	Oxygen saturation from arterial blood
AST	Lab test results	Aspartate transaminase
BUN	Lab test results	Blood urea nitrogen
Alkalinephos	Lab test results	Alkaline phosphatase
Calcium	Lab test results	N/A
Chloride	Lab test results	N/A
Creatinine	Lab test results	N/A
Bilirubin_direct	Lab test results	conjugated bilirubin
Glucose	Lab test results	Serum glucose
Lactate	Lab test results	Lactic acid
Magnesium	Lab test results	N/A
Phosphate	Lab test results	N/A
Potassium	Lab test results	N/A
Bilirubin_total	Lab test results	Sum of direct and indirect bilirubin
TroponinI	Lab test results	N/A
Hct	Lab test results	Hematocrit
Hgb	Lab test results	Hemoglobin
PTT	Lab test results	Partial thromboplastin time
WBC	Lab test results	Leukocyte count
Fibrinogen	Lab test results	N/A

Platelets	Lab test results	N/A
Age	Demographics	N/A
Gender	Demographics	Female (0), Male (1)
Unit1	Demographics	Identifier for ICU unit
Unit2	Demographics	Identifier for ICU unit
HospAdmTime	Demographics	Hours between hospital admit time and ICU admit time
ICULOS	Demographics	Hours since ICU admission
SepsisLabel	Outcome	0 if sepsis not detected by hospital staff 6 hours later 1 if sepsis was detected by hospital staff 6 hours later

For each patient, observations of all forty features are provided for every hour the patient was in the ICU (so each patient corresponds to a table of data). Each hour's worth of data is labeled (0 for no sepsis detected, 1 for sepsis detected). Since the challenge was to detect sepsis six hours before it was detected by hospital staff, the competition organizers shifted the labels six rows up the table for each patient. Table 2 shows five hours of data collected from a single patient.

***Table 2**: Five hours of data collected from a single patient in the Emory hospital. Each hour, observations are provided for all forty features as well as a label indicating whether or not sepsis was detected by hospital staff six hours later.*

	HR	O2Sat	Temp	SBP	MAP	DBP	Resp	EtCO2	BaseExcess	HCO3	...	WBC	Fibrinogen	Platelets	Age	Gender	Unit1	Unit2	HospAdmTime	ICULOS	SepsisLabel
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	59.0	1.0	1.0	0.0	-6.01	1.0	0.0
1	102.0	100.0	NaN	NaN	NaN	NaN	22.0	NaN	NaN	NaN	...	NaN	NaN	NaN	59.0	1.0	1.0	0.0	-6.01	2.0	0.0
2	102.0	100.0	NaN	99.0	84.0	76.0	18.5	NaN	NaN	NaN	...	NaN	NaN	NaN	59.0	1.0	1.0	0.0	-6.01	3.0	0.0
3	124.0	100.0	NaN	97.0	70.0	55.0	16.0	NaN	NaN	NaN	...	NaN	NaN	NaN	59.0	1.0	1.0	0.0	-6.01	4.0	0.0
4	98.0	100.0	NaN	95.0	73.0	62.0	18.0	NaN	NaN	NaN	...	6.8	NaN	276.0	59.0	1.0	1.0	0.0	-6.01	5.0	0.0

Challenges of Working with this Dataset

Throughout this project, we grappled with three challenges, shown in Table 3.

***Table 3:** The dataset is very large, is highly imbalanced, and contains lots of missing values*

Number of Rows	***% Rows Labeled 1***	***% Missing Values***
1,552,210	1.8%	68.4%

Exploratory Data Analysis

The findings presented in the histograms per variables indicate that several variables associated with lab results are skewed, including Lactate, Magnesium, Glucose, and Age. Notably, age is observed to be right-skewed, indicating a higher proportion of elderly patients admitted to the ICU compared to youths and middle-aged patients. The gender proportions for males and females, however, show similar distributions. In addition, the variable ICULOS (i.e., hours since ICU admission) is left-skewed,

with a higher proportion of data recorded in the first several hours of ICU admission. The variable HospAdmTime (i.e., hours between hospital admit time and ICU admit time) also reveals that a higher proportion of patients are admitted to the ICU shortly after their hospital admission.

Moving on to Figure 2, the top 5 pairs of variables with high correlation, defined as having an absolute correlation greater than 0.7, are identified. This high correlation can be attributed to the fact that the variables are related measurements of similar physical conditions, such as blood pressure. No variables are dropped as our tree-based model is not significantly affected by multicollinearity. Finally, Figure 3 demonstrates that the two groups are not linearly separable based on Principal Component Analysis (PCA).

Top Figure 1: *The correlation heatmap for variables with the absolute correlations greater than .7*

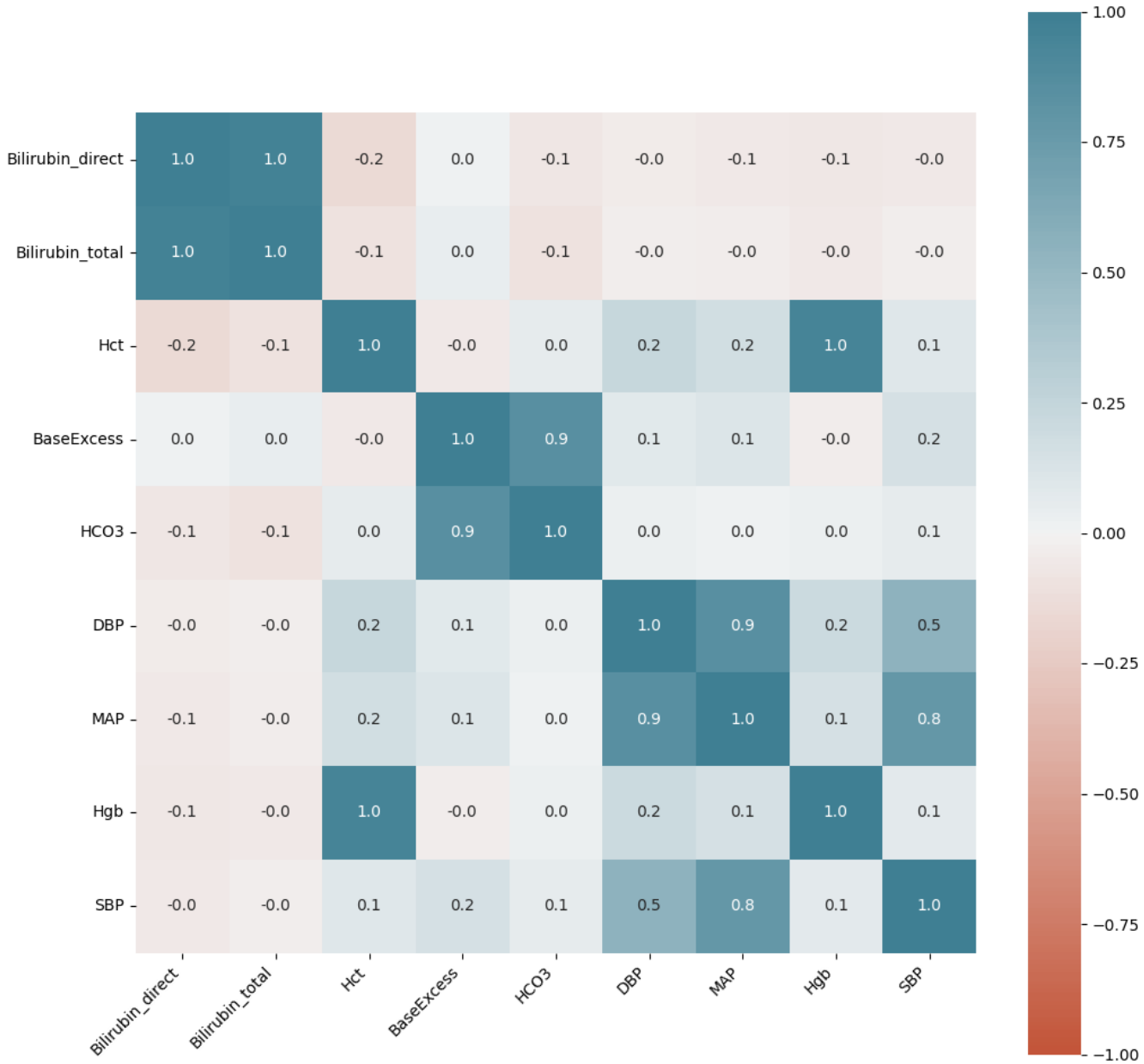
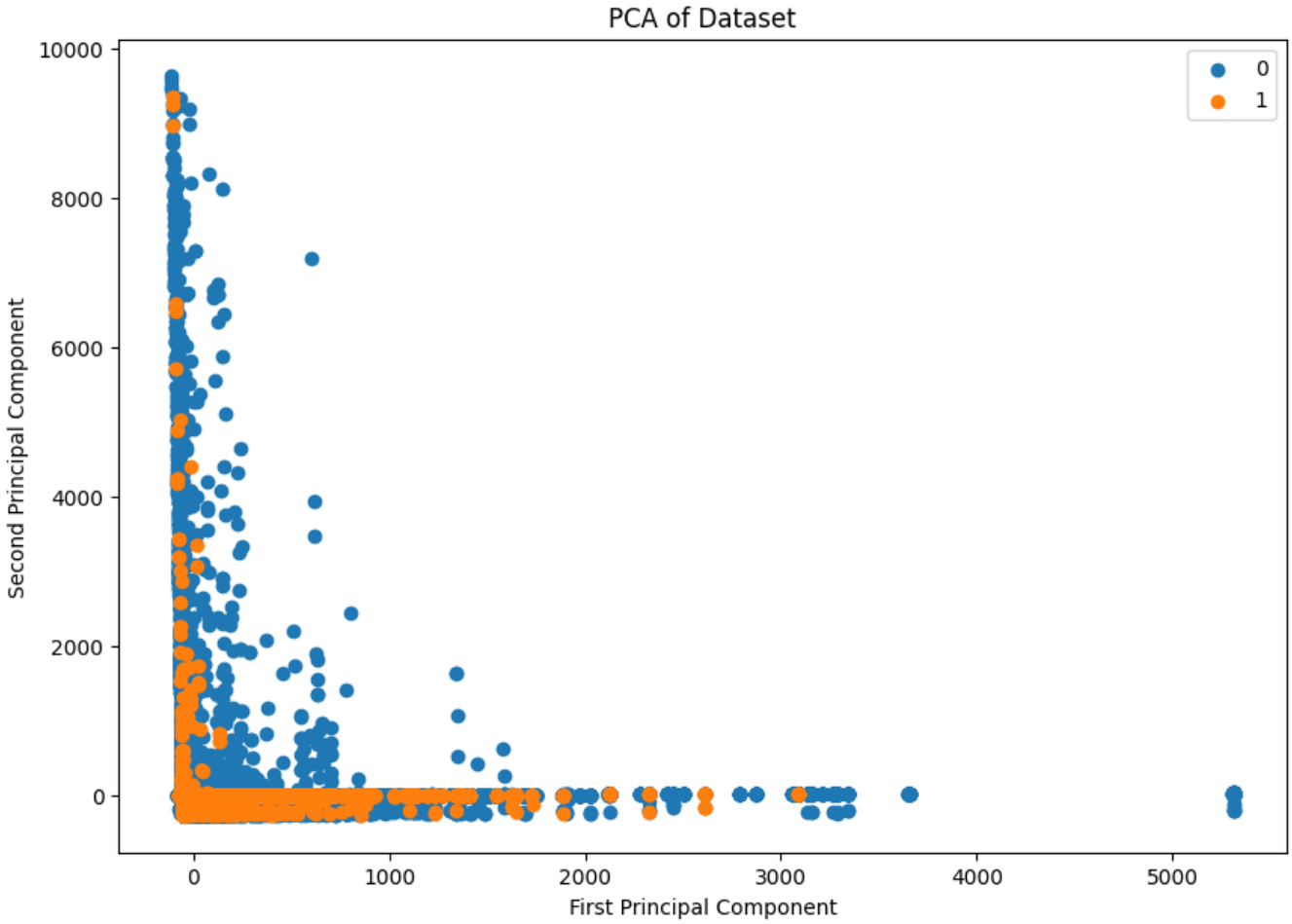


Figure 1 Table: *The correlation table for variables with the absolute correlations greater than .7*

Feature 1 Description	Feature 2 Description	Correlation
Conjugated Bilirubin	Total Billrubin	.96
Hematocrit level	Hemoglobin level	.95
Excess bicarbonate	Bicarbonate level	.85
Mean arterial pressure	Diastolic blood pressure	.85
Systolic blood pressure	Mean arterial pressure	.78

Figure 2: *PCA of the training dataset with mean imputation*



Experiments

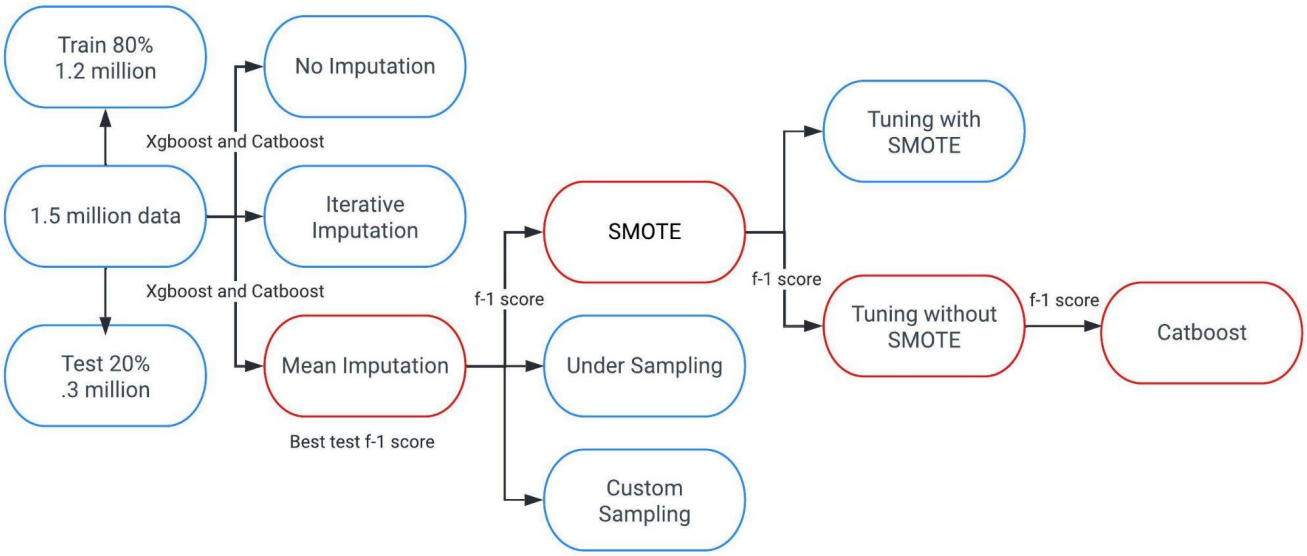
Since the first through fourth place teams in the PhysioNet competition all used some form of XGBoost, this was a natural choice of algorithm for us to experiment with. XGBoost tends to be the dominant algorithm in Kaggle competitions involving tabular data, and it's also quite computationally efficient (Brownlee, 2021).

Recently, a new gradient boosting library called CatBoost was introduced to compete with XGBoost (PyData, 2019). CatBoost has several major advantages over XGBoost: (1) It tends to be slightly more accurate, (2) It is much faster to train on large datasets, and (3) The CatBoost library comes with easy-to-use methods for gathering information about feature importance (PyData, 2019).

Experimental Design

Teams participating in the competition trained their algorithms on both the Beth Israel and Emory datasets, and their final algorithms were evaluated on a held-out test set from a third, unidentified hospital system. We did not have access to this third dataset, so we proceeded as follows: (1) Combine patient data from Beth Israel and Emory into one big dataset (1,552,210 rows), and (2) Perform an 80/20 train/test split to obtain training and test sets. (3) Three phases comparison of different combination of imputation, re-sampling and hyperparameter tuning methods as shown in Figure 3.

Figure 3: *The flow chart of the experiment design*



Phase 1: Imputing Missing Values

We experimented with three different imputation techniques to find which one led to the best model performance (Table 4).

***Table 4**: Comparison of different imputation methods. Each of these runs used the default versions of CatBoost and XGBoost (no hyperparameter tuning).*

	CatBoost	***XGBoost***
No imputation	AUC: **.857** Recall: **.070** Precision: **.827** F1-score: **.129**	AUC: **.867** Recall: **.038** Precision: **.704** F1-score: **.073**
Impute with mean	AUC: **.851** Recall: **.057** Precision: **.812** F1-score: **.107**	AUC: **.864** Recall: **.037** Precision: **.683** F1-score: **.070**
Impute with iterative imputer	AUC: **.849** Recall: **.054** Precision: **.754** F1-score: **.101**	AUC: **.848** Recall: **.023** Precision: **.625** F1-score: **.045**

We can glean several pieces of important information from the experiments in Phase 1:

a) **Not imputing at all was slightly better than both of the imputation methods** for both algorithms and for all four performance metrics. Clearly, the default methods that these models use to deal with missing values perform very well for this dataset.

b) With the exception of AUC with no imputation and AUC when imputing with the mean, **CatBoost outperformed XGBoost in all cases**. In the next two phases we will investigate whether or not this phenomenon persists once sampling techniques are applied and hyperparameters are tuned.

c) In particular, **CatBoost achieved much better recall than XGBoost, and recall is the most important performance metric for sepsis prediction**. Failing to identify a positive sepsis case can very easily lead to death, while a false positive costs valuable hospital resources but does not lead to death.

As mentioned above, we first combined all patient data into one big table and then applied each imputation method. An extension to this project could investigate whether or not performance is improved through the use of the following strategy: impute missing values on the data *from each patient separately*, then combine all patient data into one big table.

Phase 2: Dealing with Imbalanced Data

Since the data are highly imbalanced, sampling techniques were expected to significantly improve model performance, especially recall (since there are far fewer samples labeled 1 compared to samples labeled 0 in this dataset). In Phase 2, we compared a few sampling techniques to see which one led to the best model performance (Table 5). Note that these sampling techniques all require that there be no missing values in the dataset, so we imputed with the mean (the best-performing imputation technique from Phase 1).

***Table 6**: Comparing the optimal versions of CatBoost and XGBoost.*

	Optimal number of trees	***Optimal learning rate***	***Optimal maximum depth***	Performance
CatBoost	2,000	.288	9	AUC: **.821** Recall: **.141** Precision: **.437** F1-score: **.213**
Catboost Without SMOTE	1,000	.216	9	AUC: .**908** Recall: .**147** Precision: .**893** F1-score:. .**253**
XGBoost	100	.010	5	AUC: **.743** Recall: **.436** Precision: **.077** F1-score: **.132**
XGBoost Without Smote	200	.100	8	AUC: **.879** Recall: **.038** Precision: **.870** F1-score: **.073**

Our results demonstrate that CatBoost outperforms XGBoost in the analyzed dataset. Mean-imputation was found to be the optimal imputation method, while SMOTE was deemed ineffective when dealing with highly imbalanced data, characterized by a 50:1 ratio.

These below two figures show that while there was great ROC results, this does not tell the entire story. Our group found that looking at the PR curve for our specific problem of sepsis detection was much more intersting because it showed that recall was such an important metric, and that it ws very hard optimize for. Sepsis detection cares most about recall because patients with the disease should never not be caught, whereas overdetecting is not as bad of an issue. Overall, CatBoost outperformed XGBoost in both the PR and ROC curves.

Figure 4: *ROC Curve for tuned XGBoost and CatBoost without SMOTE*

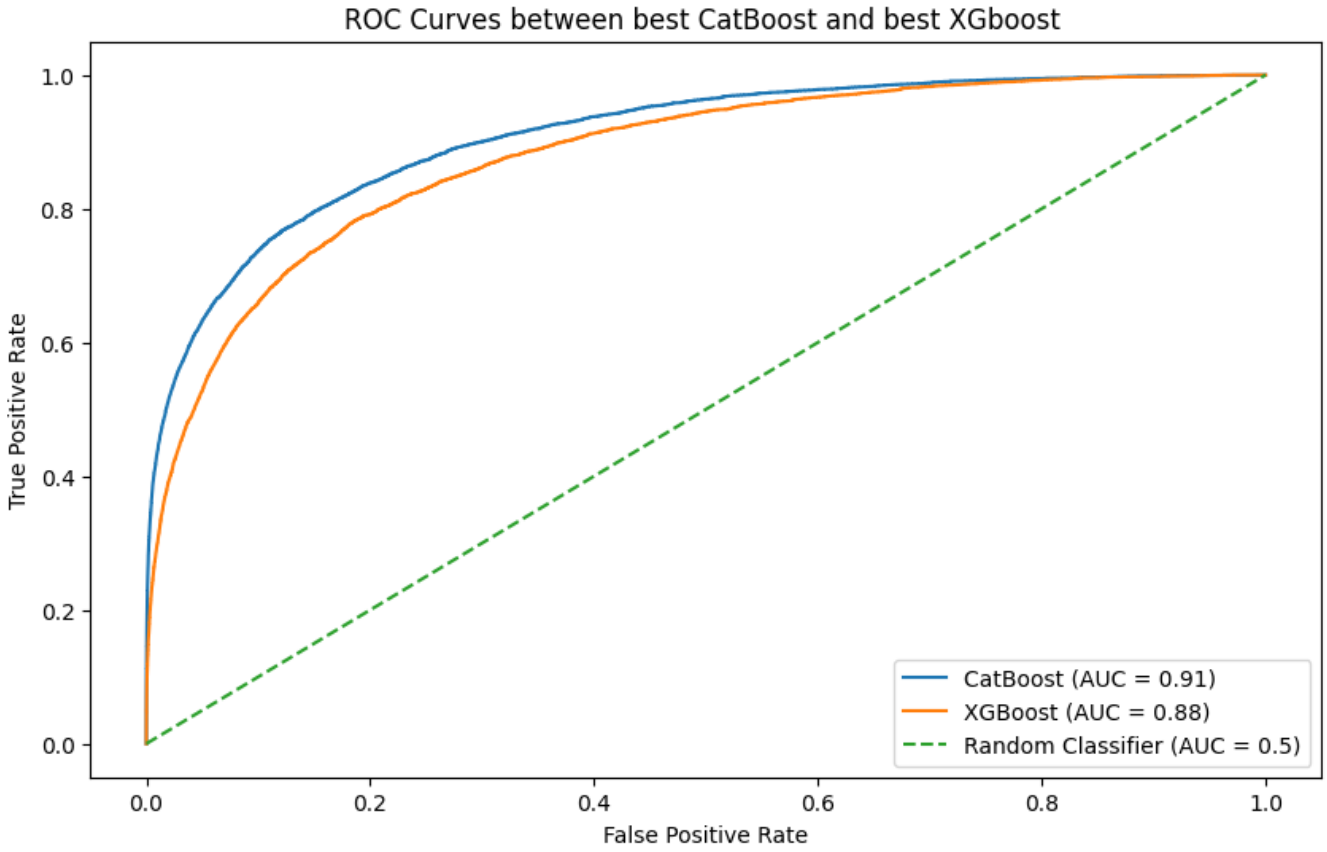


Figure 5: *Precision and Recall Curve for tuned XGBoost and CatBoost without SMOTE*

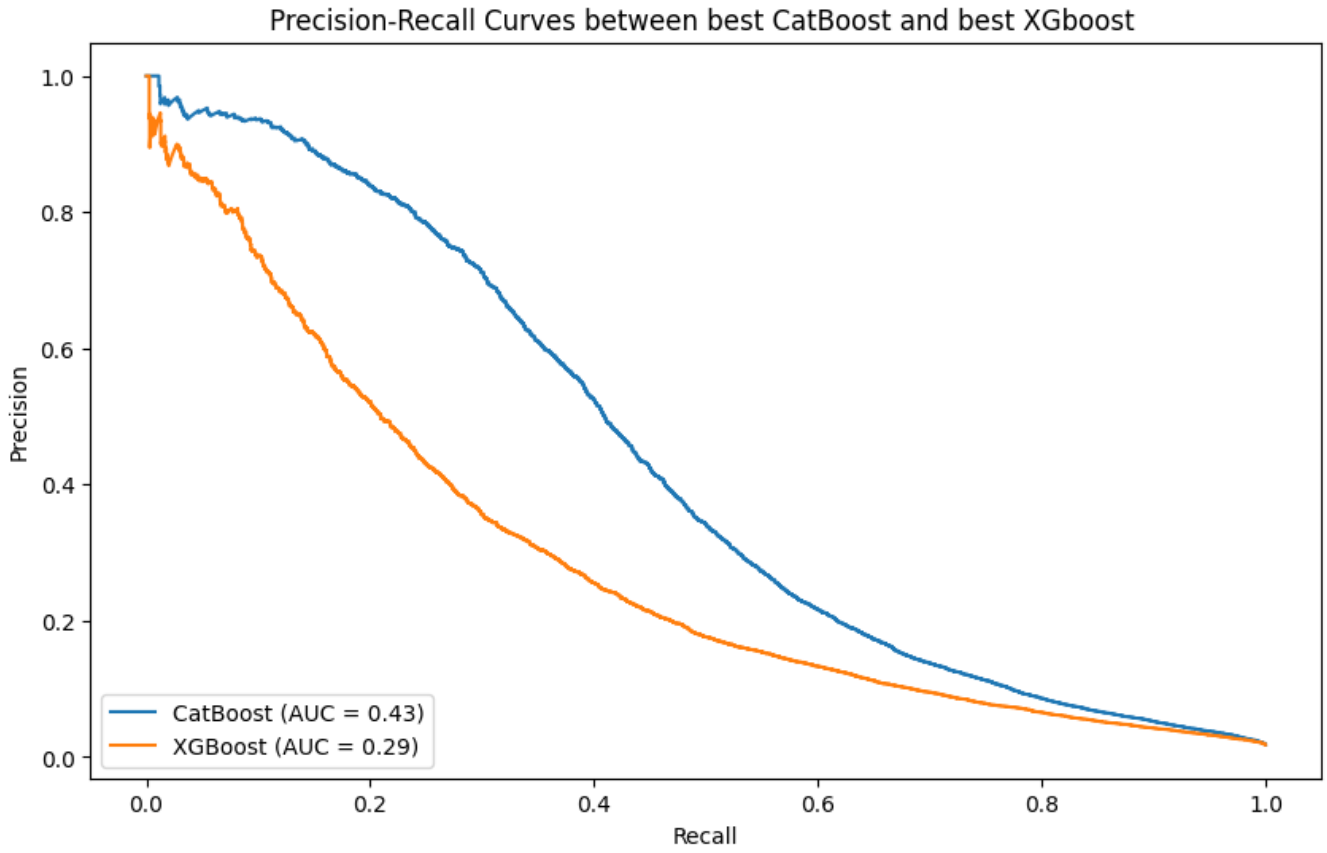
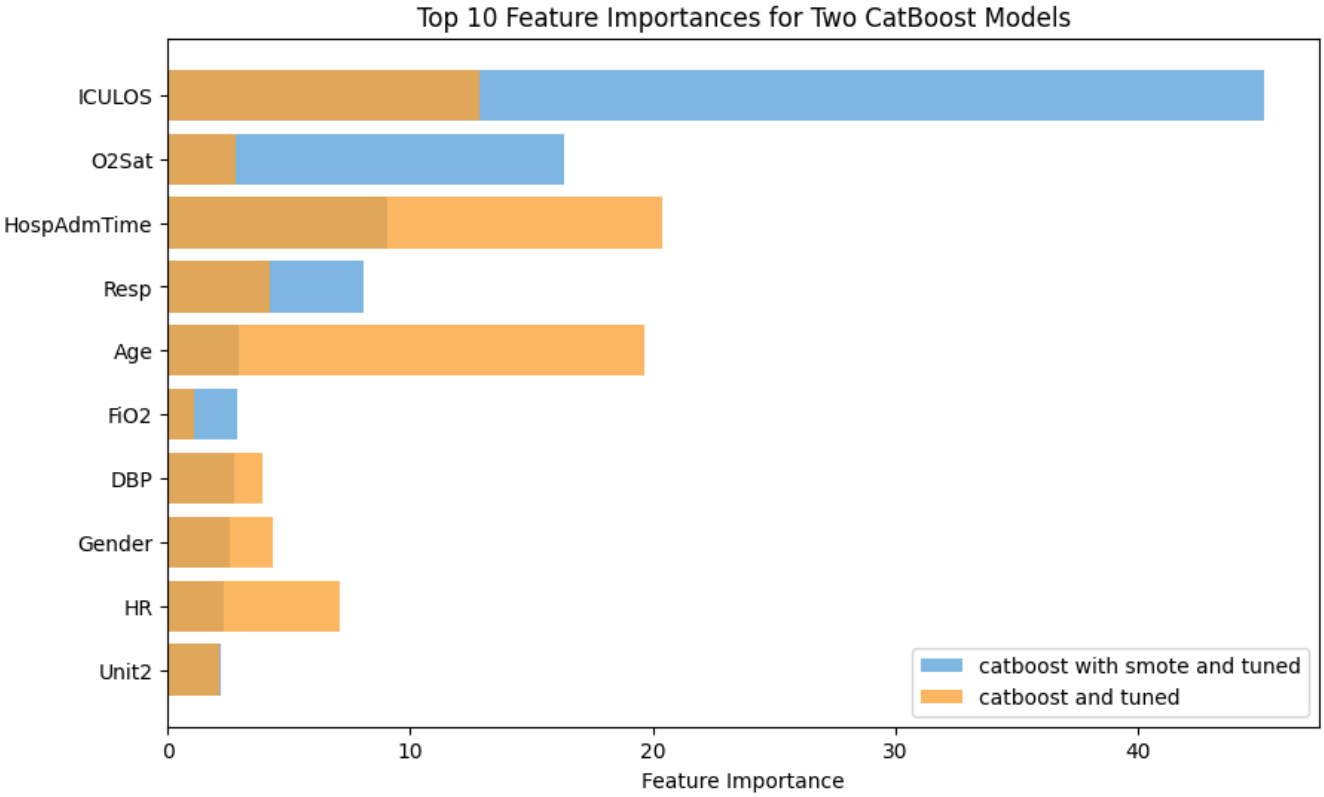


Figure 6: *The feature importance of best Catboost model with mean imputation*



Notably, it was observed that decision trees and feature importance varied significantly depending on the use of SMOTE. Specifically, in the presence of SMOTE, Pulse oximetry (O2SAT) emerged as the second important feature, whereas in its absence, this was no longer the case and age become the second important feature. These results suggest that caution should be exercised when employing SMOTE, as it can modify the distribution of data and lead to overfitting. It is essential to consider these findings in practical applications to improve the accuracy and generalization of predictive models.

Conclusions

Throughout all three phases of our experimentation, we found that CatBoost was consistently beating out XGBoost. This is fascinating because none of the teams that did well in the competition tried using this technique. This is likely because CatBoost was not extremely common at the time (it had just been released). Our team learned that it paid off to be open minded when selecting models.

Future work should experiment further with the custom upsampler, try out other machine learning models, and try to do more feature engineering. Another major struggle that we had was issues with imputing the data, and we had some good ideas for ways to do this better. We would want to impute patient by patient rather than overall. Another natural extension to the project would be to investigate the time series component of the data by using, for example, recurrent neural networks. The extension could also investigate methods for making neural networks interpretable.

Roles

Gray, Mac
I create a custom up-sampler that uses both mean and median of the minority class to create synthetic minority class data.

I ran experiments that added to phase two and phase three.

I wrote the abstract and conclusion.

I did hyper parameter tuning for XGBoost.

I created PR and ROC curves for our best performing XGBoost variant.

I wrote up the conclusion for phase two.

Lu, Dylan
I wrote the EDA, hyperparameter tunning results, and feature importance interpretation.

I implemented the EDA, hyperparameter tuned catboost with smote and without, and compared down-sampling and over-sampling for catboost and xgboost. Finally I created figures for model performance and visualization (e.g. flowchat, ROC, PR, and feature importance plot).

I combined and cleaned all the codes and uploaded them as a single file to the gitlab. I also wrote the README file in the gitlab.

Parker, Jack
I did the research for the Introduction, Background, and Data sections and wrote those three sections.

I researched ways to deal with very large, very imbalanced datasets and discovered the CatBoost library. Since CatBoost is faster than XGBoost for large datasets, tends to outperform XGBoost, and comes with easy-to-use feature importance tools, I concluded that CatBoost would be the perfect algorithm for us to focus on.

I wrote the intro to the Experiments section as well as the section for Phase 1 under Experiments, and I ran all of the experiments detailed in Table 4 (imputing missing values).

I planned out our video.

References

Agnihotri, A & Batra, N. (2020). Exploring Bayesian optimization. *Distill*.

Brownlee, J. (2021, February 17). A gentle introduction to XGBoost for applied machine learning. *Machine Learning Mastery*.

Du, J. A., Sadr N., & de Chazal, P. (2019). Automated prediction of sepsis onset using gradient Boosted decision trees. *Computing in Cardiology, 46.*

Jain, A. (2023, April 11). Mastering XGBoost parameter tuning: A complete guide with Python codes. *Analytics Vidhya*.

Letham, B., Rudin, C., McCormick, T. H., & Madigan, D. (2015). Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics, 9*(3), 1350-1371.

Lundberg, S., Nair, B., Vavilala, M. S., Horibe, M. Eisses, M. J., Adams, T., Liston, D. E., Low, D. K. W., Newman, S. F., Kim, J, & Lee, S. I. (2018). Explainable machine learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering, 2,* 749-760.

Lundberg, S. M., Erion, G., & Chen, H. et al. (2020) From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence 2*, 56–67.

Morrill, J. et al. (2019). The Signature-Based Model for Early Detection of Sepsis from Electronic Health Records in the Intensive Care Unit. *2019 Computing in Cardiology Conference.*

Nemati, S., Holder, A., Razmi, F., Stanley, M. D., Clifford, G. D., & Buchman, T. G. (2018). An interpretable machine learning model for accurate prediction of sepsis in the ICU. *Critical Care Medicine, 46*(4), 547-553.

PyData (2020, July 18). *Anna Veronika Dorogush: Mastering gradient boosting with CatBoost | PyData London 2019* [Video]. YouTube. https://www.youtube.com/watch?v=usdEWSDisS0&t=4690s

Rhee, C. et al. (2019). Prevalence, underlying causes, and preventability of sepsis-associated mortality in US acute care hospitals. *JAMA Netw Open, 2*(2).

Reyna, M. A., Josef, C. S., Jeter, R., Shashikumar, S. P., Westover, M. B., Nemati, S., Clifford, G. D., & Sharma, A. (2020). Early prediction of sepsis from clinical data: The PhysioNet/Computing in cardiology challenge 2019. *Critical Care Medicine, 48*(2), 210-217.

Rosnati, M., & Fortuin, V. (2021). MGP-AttTCN: An interpretable machine learning model for the prediction of sepsis. *PLOS ONE, 16*(5).

Vaswani et al. (2017). Attention is all you need. *31st Conference on Neural Information Processing Systems.*

Yang, M., Liu, C., Wang, X., Li, Y., Gao, H., Liu, X., & Li, J (2020). An explainable artificial intelligence predictor for the early detection of sepsis. *Critical Care Medicine, 48*(11), 1091-1096.

Zabihi et al. (2019). Sepsis prediction in intensive care unit using ensemble of XGboost models. *Computing in Cardiology, 46.*