MACHINE LEARNING TIME-TO-EVENT MORTALITY PREDICTION IN MIMIC-IV CRITICAL CARE DATABASE

An Undergraduate Research Scholars Thesis

by

JAMES ROYALTY

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Bobak J. Mortazavi

May 2021

Major: Computer Science

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, James Royalty, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Machine Learning Time-to-event Mortality Prediction in MIMIC-IV Critical Care Database

James Royalty
Department of Computer Science & Engineering
Texas A&M University

Research Faculty Advisor: Bobak J. Mortazavi
Department of Computer Science & Engineering
Texas A&M University

The rise in publicly available healthcare databases, such as MIMIC and the eICU, now make it possible to revolutionize medical care when paired with modern machine learning techniques. The MIMIC-IV critical care database allows us to explore these techniques in the ICU setting using data from thousands of patients. One area that can be improved upon in the medical domain is prediction of events in the ICU setting, such as whether a patient will have a heart attack during their stay. Through improved prediction of events, hospitals can be more efficient and better allocate resources to patients who need it most, saving both lives and costs. In the ICU setting, there has been previous work for prediction of events via machine learning classification models. However, we believe time-to-event models may offer more accuracy and interpretability than these classification models. We also believe that current, popular time-to-event models are limited in their scope, either not being able to deal with dynamic data, being too slow to use in real time, or having to make assumptions about the underlying structure of the data. Some models also require restructuring of the data into specific formats which leads to information loss. These kinds of restrictions are not desirable in the ICU, where measurements

come in frequently at irregular intervals and requires fast prediction of events. It follows, then, that we need dynamic, lightweight, time-to-event models for prediction of events that do not make assumptions about the data's structure. In this paper, we use BoXHED, a lightweight, dynamic, boosting, time-to-event model and compare it to other time-varying models in the ICU. To evaluate the different models' performances, we used time series data from the MIMIC-IV database by refactoring code used previously for MIMIC-III preprocessing by Harutyunyan et al. We then compared the different models' accuracy in predicting mortality in the ICU, both as new data measurements became available and using measurements within the first 48 hours of the patients' stays. We then evaluated the models based on an approach inspired by TREWScore where patient risk scores were compared to given thresholds to obtain each models' AUC-ROC scores.

ACKNOWLEDGEMENTS

Contributors

I would like to thank my faculty advisor, Dr. Bobak J. Mortazavi, and my peers Zhale Nowroozi, Arash Pakbin, and Nate Hurley for their guidance and support throughout the course of this research. Specifically, Zhale was able to help with paper writing, code, figures, and was available to help numerous times throughout this project. They along with Dr. Donald Lee of Emory University were instrumental in our problem and task formulation for this thesis which, without them, there would be nothing to write about. I would like to thank them for their guidance throughout this project, as well as helping out when roadblocks inevitably arose.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. Thanks goes out to my parents who have always been a supportive force in my life and have always been there for me.

Funding Sources

This work was supported, in part, by NIH award 1R21EB028486-01.

NOMENCLATURE

ICU	Intensive Care Unit
MIMIC	Medical Information Mart for Intensive Care

1. INTRODUCTION

In recent years, there has been a rise in access to public databases containing anonymized health records, specifically in the intensive care unit [1-2]. With this rise in data, there is now an opportunity to use machine learning to revolutionize clinical procedures. One potential area that can be improved is the prediction of events in the ICU, such as heart attacks and mortality. With improved prediction of events, hospitals can quickly gage whether a patient needs additional attention, better estimate the resources needed, provide more accurate diagnoses, among a host of other uses [3]. With the enormous amount of data available to the public, there is now a tremendous opportunity to improve prediction of events in multiple areas of the clinical community.

However, the pace of machine learning breakthroughs in the clinical care community have been slower to develop than other fields where machine learning has emerged [2], [4]. Recently, teams have developed machine learning benchmarks in clinical databases to help foster competition [3], [5]. Competition between research teams in developing better machine learning models can potentially lead to breakthroughs in the clinical setting, one example being the ImageNet Large Scale Visual Recognition Challenge [6]. However, one potential limiting factor in these recent ICU benchmarks is that the models used were machine learning classification models. We believe that classification models may be limited in their accuracy and interpretability in the ICU setting.

Another problem in the medical domain is that some of the models developed are able to use all available data but are highly restricted in the domain they serve, leading to less practicality [7]. Other types of models developed for practical application in the medical domain tend to be restricted in the types of data they can use, such as having to discretize the data beforehand or force the data into set intervals [3], [8]. It follows that we believe the currently used models in the medical domain are potentially limited in their accuracy in predicting events. It follows that it may be of importance that the models developed and used in practical applications are able to use all available data, no matter the rate it arrives in.

A field of statistics and machine learning that can be particularly useful to the medical domain is survival analysis. Survival analysis, or time-to-event analysis, deals with analyzing the expected amount of time until an event happens, such as how long until a patient in critical care experiences a cardiac event. Compared to typical machine learning models like regression and classification, time-to-event models predict the likelihood an event will occur over a given period of time. An added benefit of these model is that some are able to make predictions as long as they are in the time series format, meaning there is no loss of information. Survival analysis can prove useful to clinicians by offering more interpretability by outputting a series of probabilities over time, rather than a categorical or numerical response. This would allow hospitals to focus on patients who are predicted to be close to an extreme event or are more likely to experience one in the future.

To achieve our goal, we implemented and compared dynamic, time-to-event models in the MIMIC-IV database, an updated version of the MIMIC-III database [1]. The MIMIC-IV database offers a large, data-rich array of patients in the ICU setting. To tailor the data in a format that is suitable for machine learning, we refactored Harutyunyan's benchmarks for the MIMIC-III database to work with the MIMIC-IV database [3]. We then evaluated the accuracy of three dynamic time-to-event models for predicting mortality in the ICU. We had the models predict mortality throughout the patients' stay as well as during the first 48 hours of their stay.

We then evaluated the models by comparing the outputted patient risk scores with different thresholds, similar to the TREWScore work done by Henry et al. [9].

1.1 Problem Description

As with most machine learning applications, the goal is to boost the performance of one's model in solving some type of task, whether that be predicting the price of a house or determining whether a tumor is malignant. The better performance and accuracy we can achieve within our model, the better performance and accuracy we have for the task. In the medical domain, a better working model can mean more lives saved and a better use of resources by hospitals. The problem, then, is that the medical field is still basing their allocation of resources and patient risk assessments based on older techniques. These techniques can be improved by replacing them with newer techniques via machine learning.

A secondary problem is that within the field of survival analysis, current models have many limitations. First, most survival analysis models are parametric or semiparametric models that take time-static covariates as their inputs. These time-static models do not inherently account for the change in a subject's risk scores as their covariates change over time. However, in the critical care domain, how a patient's measurements change over time is of critical importance. Ideally our models should be able to respond to high frequency data at irregular intervals. This kind of data, where repeated observations of the same variable are taken for a single subject, is referred to as longitudinal data. When formatting this kind of data into a static format, information will inherently be lost. This makes time-dynamic, as opposed to time-static, models ideal in the medical domain. Second, most survival analysis models make assumptions about the structure of the data being fed in. For example, the Cox Proportional Hazard model assumes that all patient's risk scores are proportional to one another across time, meaning that if one patient's risk covariates are much different than the other patients, the magnitude of their survival curve changes but the overall shape remains the same [10]. This is important because, in the real world, a change of a patient's measurements might increase their risk while another patient's risk might decrease, but proportional hazard models do not reflect this. Therefore, it is preferable to use dynamic time-to-event models that do not make many, if not any, assumptions about the data's underlying structure.

A third problem is that much of the machine learning work in the medical domain has been with the common classification or regression models. In contrast, time-to-event models can offer more interpretability than typically used classification or regression models. As seen in Figure 1.1, the outputs of survival analysis models can be more interpretable to clinicians than a probability or number outputted by a classification or regression model. A classification model models may state a patient has a 25 percent chance of dying while plots generated by time-to-event models, like Figure 1.1, offer a much more nuanced picture of the patient's risk. For example, Figure 1.1 shows a relatively stable slope for the blue patient's stay during the first ~300 hours before experiencing a steeper drop in survival probability. The green patient has a similar slope to the blue patient at first but declines less rapidly until it remains relatively flat between 1000 and 1500 hours. The orange line shows a much steeper drop in survival probability during the first ~250 hours of their stay. This type of analysis could be another tool in clinical workers' toolboxes, allowing them to pay attention to different patients' survival odds and assess whether these models show results that match their expectations.

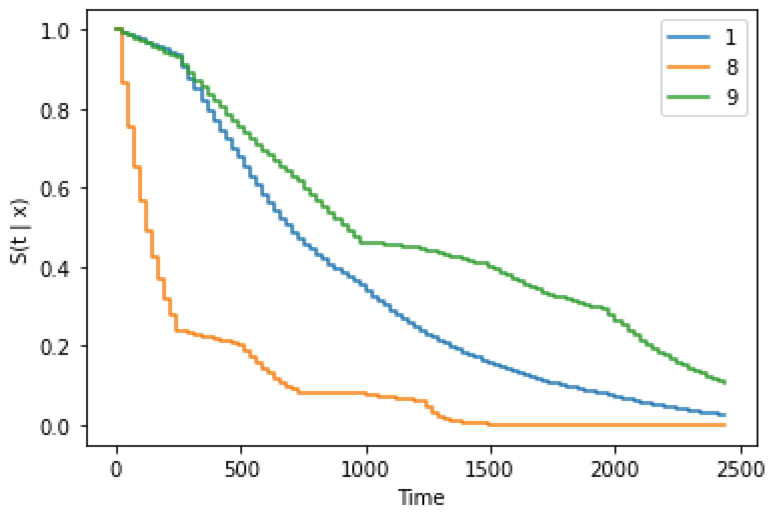

Figure 1.1: A survival function generated for three sample patients. For all patients, as time increase, their likelihood of not experiencing the event decreases.

1.2 Time-to-event Models

1.2.1 Background

One important point to make about survival analysis, also known as time-to-event analysis, is that it does not need to be confined to the topic of death. Survival analysis can be applied to data as simple as a duration of time and whether an event happened or not. In our example of in-hospital mortality, this includes how long the patient was in the ICU and what their mortality outcome was. As a different example, one could record how long a user was subscribed to a service before they left the service. In this case, leaving the service was their "death" event [11].

Survival analysis is also well equipped to deal with data censoring which is whether the events were recorded or not. This could be the result of never receiving information of whether the event occurred or the result of the event having not occurred by the time of analysis. In some cases, the event may never occur. If performing survival analysis on a population while recording their births and deaths, a person who has not died yet is considered right-censored, as their death has not been recorded yet. Rather than ignoring these individuals like other data methods, survival analysis accounts for it, making it a particularly strong method of data analysis. For our purposes of mortality prediction, this is less applicable as we *know* the information of whether a patient died or survived, as it is recorded in their medical records. However, in the cases where we ignored patient measurements past 120 hours (5 days), explained later in the paper, right-censoring comes into play. Because a patient may have experienced the mortality event at time, say, 200 hours, they will be right censored because their mortality recording has been removed.

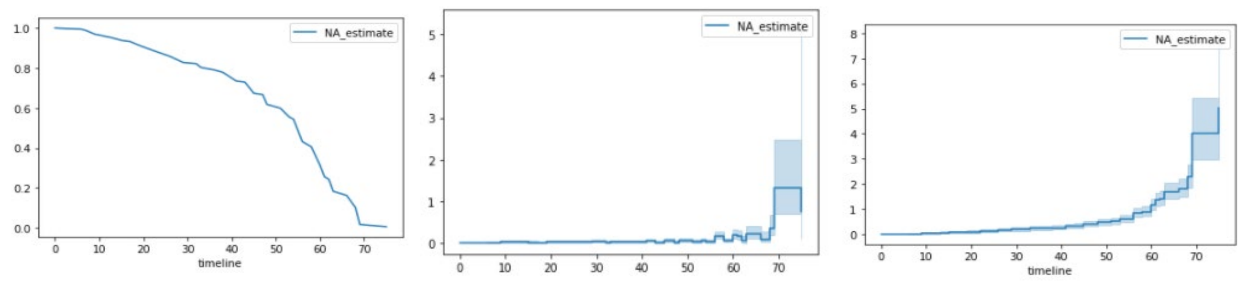

Figure 1.2: Example plots from a time-to-event model. From left to right: a) the survival function, b) the hazard function, and c) the cumulative hazard function.

Typically in survival analysis, there are three fundamental outputs [11]. The first is the survival function, which an example plot can be found in Figure 1.1 and Figure 1.2.a. This function gives the probability the event has not yet occurred at by time *t*. The second output is the hazard function which gives the probability an event occurs at the time *t*. This can be seen in Figure 1.2.b. The third output is the cumulative hazard function which is the integrated form of the hazard function. This can be seen in Figure 1.2.c. The cumulative hazard function is useful because it allows analysts to examine change in slope of the plot to see the changes in risk score for a patient over time. Hazard scores can be volatile, so cumulative hazard functions allow analysts to “zoom out” and get a bigger picture of the patient’s risk.

$$h(t|x) = b_0(t)\ exp\left(\sum_{i=1}^{n} b_i(x_i - \overline{x_i})\right) \tag{1.1}$$

Another important note to make is that many time-to-event models make the proportional hazard assumption. This is the assumption that the patients' individual hazards are proportional to one another. For the Cox proportional hazard, a widely used time-to-event model, the hazard function is simply the multiplication of the baseline hazard with a scaler calculated for the specific individual. This can be seen in Equation 1.1, the hazard function for the Cox proportional hazard model, where $b_0(t)$ is the baseline hazard and x signifies a patient's covariates [10-11]. The baseline hazard is the patients' hazard when the covariates are equal to the mean. This means that a baseline hazard is computed over the whole training population which is then multiplied by a function for the individual's covariates. This means that each patient's resulting hazard function is off by a scalar value compared to any other patient's hazard function. This can potentially lead to problems when modelling non-proportional data, which is the case in the medical domain.

1.2.2 Cox Time-Varying

$$h(t|x) = b_0(t)\ exp\left(\sum_{i=1}^{n} b_i(x_i(t) - \overline{x_i})\right) \tag{1.2}$$

As mentioned earlier in the paper, dynamic, time-to-event models are of particular interest to us. Given the popularity of the Cox proportional hazard model, it follows that a time-varying version of the model would be developed. The CoxTimeVarying model works similarly to the original Cox model but instead uses the updated covariates at time *t* whenever outputting the patient's hazard at that time [10-12]. It works the same way as the original Cox model, yet whenever outputting the hazard at time *t*, rather than using the same covariates for all outputs, it multiplies the baseline hazard by an updated calculation with the covariates at time *t*. This can be seen in the differences between Equations 1.1 and 1.2, as the difference is that the patient's covariates are updated at each time *t* as they may have changed since the last risk output. Although time dynamic, CoxTimeVarying is still limited in its assumption that all risk scores are proportional to one another. But due to its time-varying nature, it is a good baseline to compare to other time-variant models.

1.2.3 Dynamic DeepHit

$$o_\tau := P(T = \tau \mid X) \tag{1.3}$$

To accommodate time-dependent covariates, some deep survival models, specifically Dynamic DeepHit, have specialized in making predictions at discrete time τ rather than making real-time predictions [13]. Although this is not ideal for the medical domain, this model is still one of the few dynamic, time-to-event models in existence and can serve as a good comparison for our other models. For Dynamic DeepHit, event prediction becomes a binary classification prediction at each point in a predefined time grid. The model requires a predefined time τ_{max} by which the event must happen with a probability of one, another limiting assumption. But because of this assumption, the model can use a recurrent neural network with a softmax output layer to estimate Equation 1.3, where $\tau \leq \tau_{max}$, $\sum_{\tau \leq \tau_{max}} o_\tau = 1$, and X is the covariate history up to the time of prediction.

It is also important to note that Dynamic DeepHit was also developed to account for competing risks. However, because we are only attempting to predict mortality in the ICU, we do not use this feature. To use this model to predict mortality risk throughout the stay, we take the risk measure at time *t* to be $o_{\tau(t)}$ which serves as an approximation to the patient's hazard. This is useful, as we will be using patient hazard to output a mortality classification for the patient, as explained in in Section 1.3.2.

1.2.4 BoXHED

$$\mathrm{F_M(t, x) = F_0 - v \sum_{m=0}^{M-1} g_m(t, x)} \tag{1.4}$$

$$\hat{\lambda} = e^{F_M(t,x)} \tag{1.5}$$

The time-to-event model that is of most interest to us is BoXHED. BoXHED is a dynamic, time-to-event model that makes no assumptions about the underlying structure of the data it is trained on, putting it at an advantage compared to CoxTimeVarying [12], [14]. BoXHED is also a boosting model based on XGBoost, allowing it to predict data with high accuracy, while requiring little training time compared to other models [15]. This makes BoXHED very well suited for the medical domain, where measurements are constantly changing and coming in at frequent, but irregular intervals.

As seen in Equation 1.4, BoXHED works by training a log-hazard estimator $F_m(t, x)$, where *v* is the learning rate and $g_m(t, x)$ is the *m*th tree learner used to minimize the negative log-likelihood. BoXHED outputs the hazard estimator via Equation 1.5 which we will use to output the mortality classification, as explained later in the paper.

1.3 Related Work

1.3.1 Harutyunyan et al.

A lot of the work in this paper is based off Harutyunyan et al.'s work [3]. For the purposes of our paper, their work accomplished two major tasks. First, our group's code used for preprocessing the MIMIC-IV database is a modified version of Harutyunyan et al.'s MIMIC-III code on GitHub. Some additions to their original code were needed to be made to deal with the differences between the two databases structures; however, these were not too substantial. This formatted MIMIC's data into a time series format conducive to our time-to-event models.

Secondly, they defined four tasks to be targeted by machine learning. These were in-hospital mortality, decompensation, phenotyping, and length of stay, inspired by discussion by Bates et al. [16]. For the purposes of building time-to-event models, we focused on their decompensation task, as time-to-event models can be well suited for this kind of task. Similar to this task, throughout the patient's stay, our models attempted to predict patient mortality. However instead of making a mortality prediction every hour, like their decompensation task, our model makes a prediction every time a new data measurement is obtained, making it more flexible. Also, our model is attempting to predict mortality for any time in the future, not 24 hours before the event occurs.

The models developed in their paper, however, tends to use classification models, specifically LSTM and logistical regression models. These require either feeding the data into the models as time-static or having to discretize the data, causing a loss of information. The time-dynamic models we ran do not encounter this loss of information, which may lead to better prediction.

1.3.2 TREWScore

Our chosen task is for prediction of mortality in the ICU, yet our models do not output binary, yes-or-no responses like classification models. Because survival analysis outputs risk scores over time, we need a way to transform these models' outputs into a classification for whether patient will experience the mortality event. Fortunately, Henry et al.'s work provides a novel solution [9]. Because the chosen models we are comparing output a risk score for each patient with each new measurement, we can compare the risk scores to a given threshold in order to output a prediction. For example, if our chosen threshold is 1.20 and a given patient has outputted the risk scores 1.01, 1.11, and 1.19 as each of their measurements come in, this *will not* cause the model to output a positive mortality prediction, since no risk score has exceeded the threshold. However, if a new measurement comes in that causes the model to output a risk score of 1.23, the model *will* output a positive mortality prediction, as the risk score of 1.23 is greater than our chosen threshold of 1.20.

Another added benefit of this approach is we can change what triggers a positive prediction. If hazard scores are too volatile, which will trigger positive predictions too frequently, generating false positives, we can use various methods like smooth moving average to reduce this volatility. This allows us to properly predict mortality events where patients' risk scores are frequently above the threshold, rather than being triggered by a mismeasurement or an infrequent spike. Henry et al.'s work did something similar by examining results when the patient's risk score remained above the threshold for at least 8 hours [9]. We also implemented this technique in this paper.

1.3.3 Time-to-event models

With increases in computational power and the rise of popularity in machine learning, there has been increased interest in survival analysis. However, the field is by no means new. In 1958, Kaplan and Meier developed a nonparametric model that calculated survival function via calculating the fraction of patients who had survived up that point in time [17]. However, this model did not account for covariates in any way. Cox later developed the Cox Proportional Hazard model, a semiparametric model using a baseline hazard function assumption with subjects' hazard functions being proportional to one another [10].

More recent models have combined ideas of existing machine learning models with survival analysis. Ishwaran et al. used ensemble learning by developing random survival forests that generated cumulative hazard functions for patients via averaging out outputs from survival decision trees [18]. With the recent improvements in neural networks and deep learning, researchers have also developed neural networks for survival analysis usage [19-21]. However, all these models take static covariates as inputs which does not allow us to take advantage of the longitudinal data that is useful for this type of task. This type of longitudinal data is especially important in the clinical setting where observations of the same type of measurements are taken repeatedly.

Therefore, researchers have looked into how to best handle this longitudinal, or dynamic, data with time-to-event models. Historically, one way researchers have accounted for this is by using joint models that deal with the distribution of the temporal aspect in parallel with the time-to-event aspect [22-24]. Another way researchers have dealt with dynamic data is via landmarking [25]. In this process, data is kept in a time series format until a landmark point. These models have the advantage of being not computationally heavy, yet they do not take advantage of all the longitudinal data available. A limitation of both these joint and landmarking models is that they also based on proportional hazard models.

As a solution, Lee et al. designed Dynamic DeepHit, where they used a recurrent multi-task neural network with an attention mechanism to handle competing risks in dynamic data [13]. Although the ability to handle competing risks is an advantage, the data has to be discretized beforehand to be compatible with the model. This makes it difficult for real-time estimation of survival analysis, especially in the medical domain where time is of the utmost importance. Again, we prefer to have real-time estimation for our time-to-event models.

To the best knowledge of our literature review, Pakbin et al.'s work, BoXHED, is the first model that uses dynamic data to generate acute risk prediction [14]. This is necessary in the clinical domain where new data is frequently obtained and clinicians need to constantly assess the patient's risk. Thus, it seems like the best model with our goal in mind.

1.4 Hypothesis: Improved Mortality Prediction with Dynamic, Time-to-event Models

For purposes of comparison, we decided to compare BoXHED to Dynamic-DeepHit and CoxTimeVarying, as these models are also dynamic time-to-event models. CoxTimeVarying is expected to perform worse than BoXHED as it assumes the patients' risks are proportional to one another. Although maybe not the best suited in the ICU setting, Dynamic-DeepHit is powered by a powerful recurrent neural network which may make it a strong competitor against BoXHED.

As explained above, we needed some way to transform our time-to-event models' output into a classification prediction if we want to predict mortality. TREWScore's method of comparing risk scores to a given threshold works well for our task, as in a real-world setting our model just needs to take in the new data, output a risk score, compare it to the chosen threshold, and output a prediction [9]. Like TREWScore, our first evaluation criterion is once a single risk score is greater than the threshold, the model outputs a positive mortality prediction.

Realizing this may react poorly to potential volatile measurements or risk spikes, we created a second criterion. Thinking that risk spikes would be constrained to a smaller time period, we used a sliding window of 8 hours that makes a positive prediction if the risk score remains above the threshold for the given time. This means for any 8-hour block of time during a patient's stay, their risk score had to be above the given threshold for 4 hours. This was intended to reduce potential risk volatility's effect on the patient's prediction.

Our first task is to treat the data like a patient is in the ICU with data being recorded throughout their stay. In terms of the data, that means leaving it as is. If we imagine our models are reading the data from $t = 0$ to $t = t_{max}$, with each new measurement at time t, our models can output an updated risk score from this new data. The model can then compare the risk score to the chosen threshold and output its mortality prediction. This task is somewhat like Harutyunyan et al.'s decompensation task, except rather than outputting a prediction at every hour, a new prediction is outputted with each new measurement, offering more flexibility for risk prediction in the clinical domain [3].

Combining the TREWScore evaluation criteria along with dynamic, time-to-event models is a novel solution for the prediction of events in the medical domain. TREWScore was originally set up to work with the time-static, proportional-hazard Cox model [9]. However, in our approach, not only did we use time-varying models, but we also used models that make no assumption about the underlying structure of the data. Our prediction is that BoXHED can more accurately predict mortality in the ICU than CoxTimeVarying and Dynamic-DeepHit [12-14].

2. METHODS

2.1 Formatting MIMIC-IV

We set out to create, run, and compare our models on MIMIC-IV because it is a publicly available dataset containing many different types of patients useful for training and testing our models [1]. MIMIC-III was also used for a popular classification benchmark by Harutyunyan et al. from which we developed our preprocessing code [3]. We specifically chose to use data from MIMIC-IV instead of MIMIC-III as it has more patients, is up to date, and has the most recent data available. It is important to note that our time-to-event models use time series of events data and MIMIC-IV is not in an ideal format to use out of the box for this type of data. Fortunately, Harutyunyan et al.'s released their code on GitHub, most importantly the time series preprocessing code [3]. This code takes MIMIC's various tables and merges them so that we can derive each patient's time series data, allowing us to use our chosen models. Because of the structural differences between the MIMIC-III and MIMIC-IV database, we had to refactor their preprocessing code to work on the MIMIC-IV database. After modifying their code to meet the MIMIC-IV's purposes, we had time series data in the same format as Harutyunyan et al.'s work.

After running the previously mentioned preprocessing code, we still had to develop additional scripts to transform the data into a format conducive for our time-to-event models. This consisted in adding additional data columns for the time at which the measurements were recorded, the time to event, and whether the patient experienced the mortality event. Missing data was also filled forward, meaning if at time *t,* one's heart rate was 86, but at time $t + 1$ the patient did not have a heart rate reading, our data tables showed that at time $t + 1$, the patient had a heart rate of 86. In other words, all missing data was filled from the previous patient's measurements. For the CoxTimeVarying model, missing values had to be imputed, so they were imputed according to the values in Table 2.1, also derived from Harutyunyan et al.'s work [3], [12]. These missing values occurred when no measurements of that specific type were recorded since the start of their stay. For the BoXHED and CoxTimeVarying models, the categorical data was one-hot encoded with an additional column to reflect missing values. Dynamic DeepHit dealt with its categorical features via its own preprocessing code [12-14].

Furthermore, data was reduced to be the first 5 days of measurements and measurements recorded for a patient past 300 rows were removed, as we did not want patients with long stays to overload the dataset. Instead, we found it more useful to track more common cases where a patient's stay was within roughly a week.

2.2 Data Description

After running the above scripts, the output was a large csv file containing 31,532 ICU stays with 9 percent of stays experiencing the mortality event. This file contains their recorded medical measurements, the time of their measurements, when they left the ICU, and whether they experienced the mortality event. The medical measurements are the features listed in Table 2.1, based on Table 2 in Harutyunyan et al.'s paper [3]. Fortunately, each feature in their work for MIMIC-III was able to also be produced through the data available in MIMIC-IV. Modifications were needed to derive some features, such as "Glascow coma scale total" which was able to be derived from the other Glascow features shown in Table 2.1.

Table 2.1: The features used for time series data along with where they are found in MIMIC, their impute value, and whether certain models treated the feature as categorical or continuous.

Variable	**Impute Value**	**Modeled as**
Capillary refill rate	0.0	categorical
Diastolic blood pressure	59.0	continuous
Fraction inspired oxygen	0.21	continuous
Glascow coma scale eye opening	4 spontaneously	categorical
Glascow coma scale motor response	6 obeys commands	categorical
Glascow coma scale total	15	categorical
Glascow coma scale verbal response	5 oriented	categorical
Glucose	128.0	continuous
Heart Rate	86	continuous
Height	170.0	continuous
Mean blood pressure	77.0	continuous
Oxygen saturation	98.0	continuous
Respiratory rate	19	continuous
Systolic blood pressure	118.0	continuous
Temperature	36.6	continuous
Weight	81.0	continuous
pH	7.4	continuous

2.3 Running Models

When running the models, it was important that we hyperparameter tuned our model so that each one was outputting the best results. This ensured that we could present the best results, but it also made sure that the models are being compared fairly to one another.

The only hyperparameter that needed to be tuned for CoxTimeVarying's was the coefficient for its L2 penalizer, which was chosen to be 0.1 [12]. The penalizer helped ensure sure the model was converging, while also helping to reduce overfitting and variance. For BoXHED, the hyperparameters we tuned were the number of estimators and the maximum depth of each of the estimators [14]. Because CoxTimeVarying is a regression model and BoXHED is a boosting model, neither needs any further instruction on how to train, such as how many iterations to run. However, because Dynamic DeepHit is a neural network, it has a number of hyperparameters to tune. Dynamic DeepHit, like other neural networks, runs for a specified number of epochs.

To ensure we were hyperparameter tuning properly and would later fairly test our models, we divided our data into training, validation, and test sets. Because Dynamic DeepHit splits the datasets during preprocessing, we used the resulting train, validation, and test data's indices to split the dataset for the other models.

When hyperparameter tuning BoXHED, we cross validated the number of estimators to be 25 through 500 in increments of 25 with a maximum depth of 1 through 6. We monitored the negative log-likelihood to choose the best model. The best BoXHED model ended up having 75 estimators with maximum depth of 2.

Due to the difficulty of hyperparameter tuning neural networks, Dynamic DeepHit was trained using its default parameters. These included a learning rate of 0.0001, an LSTM architecture, a minibatch of 4, an alpha of 1.0, among other hyperparameters that can found on the paper's GitHub [13].

2.4 Evaluating Models

As mentioned earlier in the Related Works section, we aimed to convert out time-to-event models' output into a classification problem for mortality prediction. Similar to Henry et al.'s work TREWScore, where they used a threshold to classify binary predictions via a Cox Proportional Hazards model, we also used thresholds on the time-dynamic models we are comparing [9-10]. Like TREWScore, we implemented an evaluation method where a single risk score above a given threshold indicated a positive prediction. This is represented by the first orange dot in Figure 3.1. This was easy to implement as one just needs to take the maximum risk score before the patient left the hospital and compare it to the given threshold. If the risk score is above the threshold, it is a positive prediction which is then compared to whether the mortality event actually occurred.

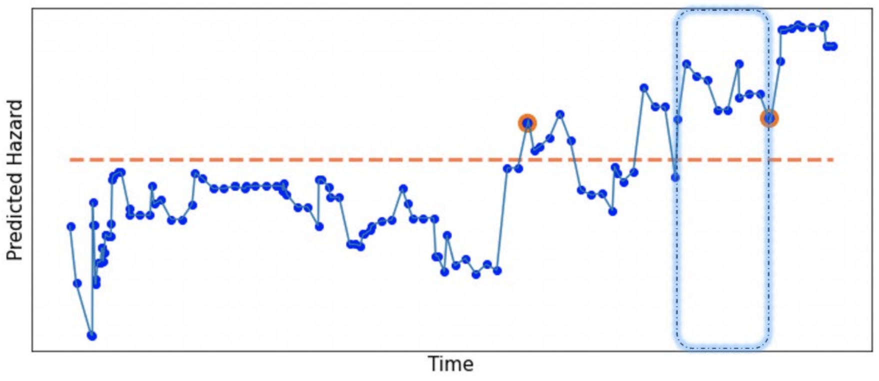

Figure 3.1: Example plot of a patient's risk score (blue). The dashed red line represents the chosen threshold. The first orange dot represents the first risk score to be above the threshold. The second orange dot represents the first time the risk score stayed above the threshold for 8 hours (rectangular box).

We also implemented a second evaluation method where, given a threshold, patients were evaluated on a sliding 8-hour window. If the patient's risk score is above the threshold throughout the entire 8-hour window, the model outputs a positive prediction. This is represented by the second orange dot in Figure 3.1. To explain, each patient's risk scores is calculated by the model as usual. Then, at a given time *t*, the model looks back at its risk scores from 8 hours prior to now. If the risk score is above this threshold for the whole window duration, the model outputs a positive prediction. This window is then "slid" forward to the patient's next series of data measurements in time and this process in repeated.

For both these evaluation methods, given a threshold, we can generate the models' predictions and compare them to the true results. This allows us to generate the ROC and PR curves and their corresponding AUC-ROC and AUC-PR. For a given threshold, we can calculate the specificity, sensitivity, and the confusion matrix.

As is standard practice, after the models were trained on the training set, they were tested on the testing set with data it had never seen before. Furthermore, because we used the same train, validation, and test split as mentioned above, we were able to test all 3 models using the same test set. This ensured all models were being evaluated fairly.

3. RESULTS

3.1 Results

Table 3.1: Comparison of models based on predicted risk score exceeding threshold at any time.

Model	**AUC-ROC**	**AUC-PR**
Baseline	0.50	0.09
Cox Time-Varying	0.74	0.29
Dynamic DeepHit	0.50	0.06
BoXHED	**0.78**	**0.35**

Table 3.2: Comparison of models based on predicted risk score exceeding threshold for 8 hours.

Model	**Window Size**	**AUC-ROC**	**AUC-PR**
Baseline	-	0.50	0.09
Cox Time-Varying	8 hours	0.81	0.36
Dynamic DeepHit	8 hours	0.47	0.05
BoXHED	8 hours	**0.83**	**0.41**

The results for this paper can be seen in Tables 3.1 and 3.2. Table 3.1 corresponds to our experiment where if a single risk score exceeded the given threshold, the model would output a positive mortality prediction. Table 3.2 results were derived from the sliding window method where a risk score had to stay above the threshold for 8 hours to trigger the mortality prediction. It is a good reminder to note that an AUC-ROC of 0.50 corresponds to a random guess and an AUC-PR of 0.09 corresponds to always predicting the positive outcome. This is noted in the baseline row of Tables 3.1 and 3.2. While AUC-ROC is typically used to evaluate classifiers, AUC-PR is also a useful metric due to the imbalance in the number of positive and negative outcomes in our dataset [26].

In Tables 3.1 and 3.2, it can be seen that BoXHED outperforms the other time-to-event models fairly well, especially in the AUC-PR category. CoxTimeVarying gets decently close to BoXHED in terms of AUC-ROC, but BoXHED clearly performs better in AUC-PR. The results for Dynamic DeepHit may seem odd at first, with its predictions performing worse than the previously described baseline. This could be because the model assumes that mortality must occur by some time τ_{max}, which is inherently unsuited for our task of mortality prediction in the ICU where patients may be in the hospital for an unknowable amount of time [13].

4. CONCLUSION

As can be seen in Tables 3.1 and 3.2, BoXHED handily outperformed the other dynamic, time-to-event models it was compared to. This makes it an excellent model choice for anyone wanting to work with high-frequency, irregularly-sampled, time-series data. BoXHED has the additional benefits of requiring very little hyperparameter tuning as well as a short training time when compared to neural networks, like Dynamic DeepHit.

4.1 Other Work

The best model trained was BoXHED with a sliding window criterion of 8 hours, receiving an AUC-ROC and AUC-PR of 0.83 and 0.41, respectively. This is fairly comparable to other work that have been done in the past. In Henry et al.'s TREWScore, their final AUC-ROC score was 0.83 for detecting septic shock [9]. Their task can arguably be easier or harder than ours since their data set was designed to have a specific distribution of sepsis patients, while ours was the resulting distribution generated by the MIMIC-III preprocessing code. This code only removed patients if they had certain information missing, rather than aiming for a specific distribution of patients [3]. In other cases, their task may appear more difficult than ours. For example, we were targeting the general mortality prediction, not specifically targeting what event happened to cause the event, whilst TREWScore was specifically attempting to detect onset septic shock.

In Harutyunyan et al.'s MIMIC-III benchmark paper, their best model on the decompensation task was able to achieve an AUC-ROC of 0.911 [3]. However, this task made a prediction at every hour for whether the patient would die within the next 24 hours. This means for relatively healthy patients the model could predict survival which would generate 24 correct predictions every day the patient remained in the ICU. Whereas our model calculated the AUC-ROC based on whether the model ever flagged the patient for mortality during their stay. In other word, the benchmark paper calculated their AUC-ROC based on every prediction the model made, while our work calculated AUC-ROC on a per-patient basis.

Overall, the results for our work are positive. We were able to define a novel task to test the only dynamic, time-to-event models of which we know of that can generate risk scores without any underlying assumptions about the data's structure.

4.2 Limitations and Future Work

One area to look in to is creating new criteria for flagging patients for mortality predictions. In this paper, we only used the criteria of a single risk score going above the threshold and risk scores staying above the threshold for 8 hours. Another example of a criterion that could be used is a smoothed moving average which may be able to deal with volatility even better than the sliding-window approach. Other possible approached could be inputting the hazard outputs into a time-static model, say XGBoost, via feature engineering and seeing how it affects results [15].

Another area for improvement is having a set amount of time we want to predict the patient at risk before the event happens. For example, trying to predict if the patient is at risk for mortality in the next 6 hours. This would give clinicians time to react to a mortality prediction, say, by requesting certain types of lab work on the patient. As it stands, our model currently flags that patient with no regards to this. The model only flags mortality risk, whether the patient will die, say, within the next 2 hours or the next 2 days.

A third area of improvement would be expanding the scope of our models. The models currently only make predictions from data within the ICU. This could potentially be expanded to include data throughout a patient’s stay at the hospital. There is also a potential for improvement in the models by including other types of data collected throughout a patient’s stay. Another area for improvement is accounting for competing risks from multiple adverse events.

4.3 Closing

The medical domain has been much slower to be improved by the advancements in machine learning when compared to other fields. This is partly due to the fact that it is a very complex domain, so it is important that we use the best models for the job. The ICU specifically gives us access to a rich source of data that is time variant. This is type of data is difficult for most standard, machine learning models to deal with and requires transformations as a result where information in inherently lost. This is where dynamic, time-to-event models can be of use, as they do not require these transformations. In this paper, we compared the available dynamic, time-to-event models and their performance for ICU mortality prediction. We showed that that BoXHED performs very well when compared to these models with an AUC-ROC and AUC-PR of 0.83 and 0.41, respectively. This is because BoXHED offers the functionality to be lightweight and be able to handle dynamic, time-varying data while making very few assumptions in the process. This makes it an ideal model choice for those working with high-frequency, time-varying measurements.

REFERENCES

[1] A. Johnson, L. Bulgarelli, T. Pollard, S. Horng, L. A. Celi, and R. Mark, "MIMIC-IV," *MIMIC-IV v0.4*. https://physionet.org/content/mimiciv/0.4/ (accessed May 3, 2021).

[2] T. J. Pollard, A. E. W. Johnson, J. D. Raffa, L. A. Celi, R. G. Mark, and O. Badawi, "The eICU Collaborative Research Database, a freely available multi-center database for critical care research," *Sci. Data*, vol. 5, no. 1, p. 180178, 2018.

[3] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Sci. Data*, vol. 6, no. 1, p. 96, 2019.

[4] M. Engelhard, A. Navar and M. Pencina, "Incremental Benefits of Machine Learning—When Do We Need a Better Mousetrap?" *JAMA Cardiology*. https://jamanetwork.com/journals/jamacardiology/article-abstract/2777054?casa_token=g-ulfX1adcAAAAAA:ZBLeOj5ranftwmcOZOYUVwJ5SjoXgp2fnZWNnLu4mRVMn4AJKTvmU7_JWRLUv4BrquKvF2Xnh90 (accessed April 12, 2021).

[5] S. Sheikhalishahi, V. Balaraman, and V. Osmani, "Benchmarking machine learning models on multi-centre eICU critical care dataset," *PLoS One*, vol. 15, no. 7, p. e0235424, 2020.

[6] "Image Classification on ImageNet." Papers With Code. https://paperswithcode.com/sota/image-classification-on-imagenet (accessed April 12, 2021).

[7] B. J. Mortazavi et al., "Comparison of Machine Learning Methods With National Cardiovascular Data Registry Models for Prediction of Risk of Bleeding After Percutaneous Coronary Intervention," *JAMA Netw Open,* Jul. 2019, doi: 10.1001/jamanetworkopen.2019.6835.

[8] Z. C. Lipton, D. C. Kale, and R. Wetzel, "Modeling Missing Data in Clinical Time Series with RNNs," *arXiv preprint arXiv:1606.04130*, 2016.

[9] K. E. Henry, D. N. Hager, P. J. Pronovost, and S. Saria, “A targeted real-time early warning score (TREWScore) for septic shock,” *Science Translational Medicine*, vol. 7, no. 299, Aug. 2015, doi: 10.1126/scitranslmed.aab3719.

[10] D. R. Cox, “Regression models and life-tables,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 34, no. 2, pp. 187–202, Mar. 1972.

[11] “Introduction to survival analysis.” Read the Docs: Lifelines. https://lifelines.readthedocs.io/en/latest/index.html (accessed April 12, 2021).

[12] “Time varying survival regression.” Read the Docs: Lifelines. https://lifelines.readthedocs.io/en/latest/Time%20varying%20survival%20regression.html (accessed April 12, 2021).

[13] C. Lee, J. Yoon, and M. Van Der Schaar, “Dynamic-DeepHit: A deep learning approach for dynamic survival analysis with competing risks based on longitudinal data,” *IEEE Transactions on Biomedical Engineering*, vol. 67, no. 1, pp. 122–133, Apr. 2019.

[14] A. Pakbin, X. Wang, B. J. Mortazavi, and D. K. Lee, “BoXHED 2.0: Scalable boosting of functional data in survival analysis,” *arXiv preprint arXiv:2103.12591*, Mar. 2021.

[15] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, Aug. 2016.

[16] D. W. Bates, S. Saria, L. Ohno-Machado, A. Shah, and G. Escobar, “Big data in healthcare: using analytics to identify and manage high-risk and high-cost patients,” *Health Affairs*, vol. 33, no.7, pp. 1123–1131, Jul. 2014, doi: 10.1377/hlthaff.2014.0041.

[17] E. L. Kaplan and P. Meier, “Nonparametric estimation from incomplete observations,” *Journal of the American Statistical Association*, vol. 53, no. 282, pp. 457–481, Jun. 1958.

[18] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, M. S. Laueret, “Random survival forests,” *Annals of Applied Statistics*, vol. 2, no. 3, pp. 841–860, Sep. 2008.

[19] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger, "DeepSurv: personalized treatment recommender system using a cox proportional hazards deep neural network," *BMC Medical Research Methodology*, vol. 18, no. 1, pp. 1–12, Feb. 2018.

[20] C. Nagpal, S. Yadlowsky, N. Rostamzadeh, and K. Heller, "Deep coxmixtures for survival regression," *arXiv preprint arXiv:2101.06536*, Jan. 2021.

[21] C. Lee, W. Zame, J. Yoon, and M. Van Der Schaar, "DeepHit: A Deep Learning Approach to Survival Analysis with Competing Risks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[22] X. Huang, G. Li, and R. M. Elashoff, "A joint model of longitudinal and competing risks survival data with heterogeneous random effects and outlying longitudinal measurements," *Statistics and its interface*, vol. 3, no. 2, p. 185, 2010.

[23] D. Zeng and J. Cai, "Simultaneous modelling of survival and longitudinal data with an application to repeated quality of life measures," *Lifetime Data Analysis*, vol. 11, pp. 151–174, Jun 2005.

[24] J. Barrett and L. Su, "Dynamic predictions using flexible joint models of longitudinal and time-to-event data," *Statistics in medicine*, vol. 36 ,no. 9, pp. 1447–1460, Apr. 2017.

[25] K. T. Tanner, L. D. Sharples, R. M. Daniel, and R. H. Keogh, "Dynamic survival prediction combining landmarking with a machine learning ensemble: Methodology and empirica comparison," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 184, no. 1, pp. 3–30, Jan. 2021.

[26] Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, vol. 10, no. 3, p. e0118432, 2015.