UPPSALA
UNIVERSITET

UPTEC STS 26025

Degree project 30 credits

June, 2026

# Survival Analysis of Radio Replacements

## A Case Study of Radio Units in Telecom Networks

Elias Ihrefjord
Karl Johansson

Master's Programme in Sociotechnical Systems Engineering

UPPSALA
UNIVERSITET

Survival Analysis of Radio Replacements

Elias Ihrefjord
Karl Johansson

## Abstract

This paper has explored survival analysis models and their ability to predict customer driven replacements of radio units using customer data from Ericsson. Survival Analysis examines how long until an event of interest occurs. While predictive performance is important, it does not provide insight as to how features affect the model predictions. Therefore, this study has also focused on interpreting feature-risk relationships using external explanation methods as well as model coefficients when possible. This paper implemented four different models to compare; Weibull Accelerated Failure Time, Random Survival Forest, DeepHit and Logistic Hazard. A Kaplan-Meier model was implemented as a baseline for comparison. The data used consists of counters, alarms and restarts. The counters were sampled from a single weekday every week for 2025 while alarms and restarts were summed over each week. The models were evaluated using four different metrics: time-dependent concordance index, integrated brier score, time dynamic AUC and $\text{precision}_k$. All models performed substantially better than the baseline. Random Survival Forest was the best performing model in all metrics except $\text{precision}_k$, while Logistic Hazard was the best at $\text{precision}_k$. This suggests that Logistic Hazard is better at ranking short term risk while Random Survival Forest is better at long term risk ranking. The interpretability was evaluated using SHAP, LOFO and when possible the model's coefficients. The most stable SHAP values were found for Weibull Accelerated Failure Time and decreasing in the order Random Survival Forest, DeepHit and Logistic Hazard. The models show consistency across the most important features, however they vary in how the features influence predictions. This study finds that a simpler model, such as the regression based Weibull Accelerated Failure Time model, provides better interpretability of results, with similar predictive performance. In contrast, combining multiple models and explanation methods provides a fuller understanding of underlying data patterns not captured by standard regression approaches.

**Faculty of Science and Technology**

**Uppsala University, Place of publication Uppsala**

Supervisor: Erik Sand  Subject reader: Dave Zachariah

Examiner: Elísabet Andrésdóttir

# Populärvetenskaplig Sammanfattning

I mobilnät krävs tusentals radioutrustningar ute i fält för att vi ska kunna ringa, strömma video och använda internet. När en radionhet börjar bete sig onormalt måste en operatör till slut fatta beslut om den ska bytas ut. Ett byte kan vara dyrt och tidskrävande, men att vänta för länge kan ge sämre täckning och i värsta fall avbrott. Samtidigt är det inte alltid tydligt varför vissa enheter byts tidigare än andra, eftersom beslutet ofta bygger på en kombination av mätvärden, larm och mänskliga bedömningar.

I det här arbetet undersöks om överlevnadsanalys kan användas för att förutsäga när radioutrustning kommer att bytas ut. Överlevnadsanalys är en familj av statistiska och maskininlärningsmetoder som modellerar tid till en händelse, till exempel tid till att en komponent går sönder eller, som i detta fall, tid till att en radioenhet ersätts. En viktig detalj är att många enheter inte byts ut under den tidsperiod som studeras. Då vet man bara att de är kvar i nätet vid den sista observationen. Det kallas censurering och är något som överlevnadsanalys är särskilt bra på att hantera.

Studien bygger på driftdata från Ericsson från en kund och en produkt i 5G Radio Access Network (RAN) portföljen. Datamaterialet består av tre huvudkällor. Räknare och KPI:er; mätvärden som beskriver radio- och nätbeteende, till exempel trafik och signalmått, uppmätta var 15:e minut. Larm; händelser som indikerar avvikelser i hårdvara eller mjukvara. Omstarter; både automatiska och manuella som kan indikera fel som inte ses i övrig data.

För att kunna träna modellerna på tillgänglig hårdvara inom rimlig tid valdes ett representativt dygn per vecka att läsas in med hög upplösning. Genom en inledande analys valdes onsdagar som den veckodag vars mönster (medelvärden och variation) låg närmast helhetsmönstret för en testperiod. Larm och omstarter är mer sällsynta, så de summerades per vecka. Efter rensning av starkt korrelerade eller nästan konstanta variabler återstod 42 förklarande variabler.

En extra utmaning är att enhetsbyte inte alltid sammanfaller med verkligt fel. I datan syntes att många enheter var i praktiken nere (nära 100% otillgänglighet) flera veckor innan de faktiskt registrerades som utbytta, exempelvis på grund av väntetid på reservdelar eller planering av fältarbete. För att modellerna inte skulle lära sig ”byt först när den redan är död” flyttades händelsetidpunkten bakåt för dessa fall till strax innan enheten blev icke-operativ.

Fyra stycken överlevnadsmodeller jämfördes i studien. Weibull Accelerated Failure Time, en regressionsmodell som antar en viss matematisk form för hur risken utvecklas över tid. Random Survival Forest, en samling beslutsträd anpassade för censurerad data som producerar överlevnadsfunktioner över tid. Samt DeepHit och Logistic Hazard, neurala nätverk som delar in tiden i diskreta intervall och lär sig distributionen av risk.

Modellerna utvärderades både på hur bra de rankar vilka enheter som löper störst risk och hur bra de kalibrerar sannolikheter över tid, det vill säga om förutsagda risker stämmer med observerade utfall. Resultaten visar att alla modeller lär sig tydliga mönster och presterar avsevärt bättre än en enkel populationsbaserad referens. Övergripande var Random Survival Forest starkast, särskilt för stabila och långsiktiga prediktioner. De neurala nätverken var däremot bättre på att fånga kortsiktiga varningssignaler och hade högst $\text{precision}_k$ när man tittar på de 20 mest riskfyllda enheterna för en kort tidshorisont.

För att förstå varför modellerna flaggar vissa enheter användes förklaringsmetoder. Weibull Accelerated Failure Time är i sig mer lättolkad eftersom den ger koefficienter för varje variabel. För de mer komplexa modellerna användes bland annat SHAP, som kan ses som ett sätt att uppskatta hur mycket varje variabel bidrar till en enskild prediktion. En viktig observation är att modellerna i stor utsträckning är överens om vilka variabler som är viktiga, men inte alltid ger lika tydliga och konsekventa svar om hur en variabel påverkar risken. Den tydligaste och mest återkommande signalen i samtliga modeller var en viss MAC-relaterad räknare, tillsammans med vissa larmkategorier och mått kopplade till tillgänglighet.

Sammanfattningsvis visar arbetet att överlevnadsanalys är ett lovande angreppssätt för att modellera kunddrivna utbyten av radioutrustning. Metoderna hanterar naturligt att många enheter inte hinner bytas under studieperioden, och de kan ge en prioriteringslista över vilka enheter som bör granskas först. Samtidigt blir tolkbarheten en praktisk avvägning, enklare modeller är lättare att förklara, medan mer flexibla modeller kan fånga komplexa mönster men kräver extra analys för att ge begripliga insikter. I praktiken talar resultaten för att kombinera flera modeller och flera förklaringsmetoder för att få både god prediktiv förmåga och mer robusta slutsatser om vilka signaler som driver beslut om utbyte.

# Acknowledgements

This master thesis was carried out in collaboration with Ericsson's Radio Data Analytics team. We would like to thank all members of the team for their shown interest and support. Especially, Erik Sand and Rikard Bauer for their continuous guidance throughout the project.

Finally, we would like to thank our subject reviewer Dave Zachariah for his support with the thesis.

*- Elias Ihrefjord & Karl Johansson*

# Contents

# 1 Introduction

Survival analysis is the study of outcomes over time, or time-to-event, often used in healthcare studies to analyze the effects of treatments or the outcomes of patient groups. These models are unique, as they are particularly suited for handling time-dependent outcomes and right-censored data, where the outcome of certain individuals is not yet known. Traditionally, survival analysis has been done using statistical methods such as Kaplan–Meier estimators [1] and Cox regression [2]. As applications have expanded, more flexible machine learning approaches have been introduced, such as random survival forests [3] and neural network-based models [4], enabling the modeling of more complex relationships with fewer statistical constraints.

In contrast to many classical applications of survival analysis, where the event of interest corresponds to a well-defined physical outcome such as failure or death, this study considers a setting in which the observed event is driven by human decision-making. The study revolves around predicting the replacements of in-field radio units, which are maintained by network operators who decide which units should be replaced. These decisions carry significant operational and economic consequences, as premature replacements lead to unnecessary costs, while delayed replacements may result in service degradation or critical downtime. While these decisions are informed by measurable indicators of hardware and performance degradation, they are not solely determined by the intrinsic state of the unit. Replacements may occur due to external factors or from errors in the units reporting and logging system, making the observed event an imperfect proxy for true failure. As a result, the modeling task involves not only capturing underlying failure risk, but also implicitly learning the decision patterns responsible for the unit's outcome.

The time-varying nature of the data introduces further challenges, since the operational state of a unit can change over short time intervals. While there exists survival models that explicitly handle these time-varying features, they are not necessarily suitable for this application. Since a unit state can change rapidly, past behavior can be misleading as to the current state of a unit or completely uninformative. Furthermore, these models would require much larger amounts of data for prediction, making it operationally demanding and requiring larger amounts of computational resources. This instead motivates approaches that utilize the most recent observations to accurately rank units according to their risk of replacement, supporting effective prioritization of maintenance decisions. This thesis introduces a dynamic survival modeling framework where each observation is treated as a conditional risk assessment based on the current unit state.

Given the event setting, where replacements can happen due to a wide range of degradations or failure, getting insight into not only when but why they happen becomes a point of interest. By utilizing the models as a method of case interpretability, the study aims to investigate the conclusions that can be drawn from intrinsic model interpretability and explainability techniques such as SHAP analysis.

# 2 Purpose, Aims and Motivation

The aim of this study is to evaluate survival analysis models for modeling customer-driven radio replacements with respect to predictive performance and interpretability. Radio unit replacements are costly operations, often requiring on-site maintenance in remote and elevated locations. Despite their operational impact there is a blind spot regarding which factors are the most influential in driving replacement decisions. This lack of insight limits proactive maintenance planning and product development from a reliability point of view.

Replacements represent a relatively small subset of units and occur at varying points in time, creating challenges in how the data should be represented and modeled. Commonly used classification models are limited in this setting, as they disregard the temporal aspect of replacements and are sensitive to severe class imbalance. Survival analysis models instead provide a framework for estimating and ranking replacement risk over time. Selecting an appropriate survival modeling approach is therefore important, as it influences how replacement events are represented and which assumptions are made about the relationships between features and replacement risk. For that reason, the first research question proposed is:

(1) How do different survival analysis models compare in their ability to predict customer-driven replacements, evaluated in terms of both replacement probability estimation and unit risk ranking?

While evaluating the predictive performance can provide some insight based on the model assumptions, it does not provide insights as to how certain features affect the model prediction. To complement model performance, the study also examines the models' interpretability, to analyze if the models provide understandable insights into the factors driving their predictions. For that reason, the second research questions proposed is:

(2) How consistent and informative are the feature-risk relationships derived from the models for explaining customer-driven replacements?

The combination of these research questions aims to explore which modeling framework not only produces the best performance, but also provides transparent insights into the factors influencing those decisions. This twofold focus enables the study to move beyond black-box forecasting, exploring the use of these models as a method of gaining case specific insight.

## 2.1 Delimitations

This paper will focus on a set number of models, to limit the scope such that the models can be evaluated to the fullest degree. While there may exist further models which could perform better for this use case, the paper focuses more on the viability of the method, and why certain models perform better or worse for the replacement scenario. Consequently, the paper will only utilize existing survival analysis machine learning models.

As the focus of the paper is on the method of survival analysis machine learning, specific

technical insights regarding hardware or software of the analyzed radio units will not be presented. Consequently, features have been anonymized for confidentiality.

# 3 Background

In this section, a short description is presented for Ericsson and their telecommunication radio units, the methodology behind Survival Analysis and the subject of Reliability Engineering.

## 3.1 Ericsson and 5G RAN

Ericsson is a telecommunications company founded by Lars Magnus Ericsson in 1876, which quickly grew into Sweden's largest telephone manufacturer with over 500 employees by the end of the century [5]. The company has continued to grow into the global Radio Access Network supplier it is today, serving customers worldwide and leading the supply of 3G technology throughout the early 2000s [6]. Today the company is focused on 5G and the eventual evolution of 6G, with the intention of enabling future technologies through innovative and high-performing networks [7].

The Radio Access Network is part of the wireless cellular technology and infrastructure that enables users in society to stay connected to each other and the internet wherever they go [8]. The network consists of multiple devices and functionalities that work together to form a connection between user devices and the internet, namely antennas, radio units and baseband units. This paper focuses on the radios which convert digital data into signals and the antennas which transmit the signals, and vice versa. RAN technology has evolved since its introduction from the first generation known as 1G to the current generation known as 5G. For each generation, new functionalities have been introduced which has evolved the capabilities from simple phone calls to live video streaming [9, p. 1-5]. As part of the 5G Radio Access Network evolution, new radio technologies have emerged at Ericsson. One of the main evolutions has been in antenna integrated radios with massive multi-input multi-output (Massive MIMO) configurations. These radio units have a much higher amount of ports than a regular remote radio, 4-20 times the amount, which achieves better coverage, capacity and user performance [10]. Massive MIMO units are also capable of beamforming due to the high amount of ports, which can amplify or mute signals in certain directions or patterns to boost performance in the network. This thesis analyzes a product from this portfolio.

## 3.2 Survival Analysis

Survival Analysis refers to a group of statistical methods that focus on timing of and duration until events of interest occur. They have a lot in common with regression models, but they use different likelihood estimators [11, p. 14-15]. The models examine how long until an event of interest occurs, commonly called survival time, using conditional probability. The models estimate the survival function $S(t|\mathbf{x})$ over time $t$ utilizing covariates $\mathbf{x}$,

which is the probability to survive to a time $T > t$ [11, p. 22], see Equation 1.

$$S(t|\mathbf{x}) = P(T > t|\mathbf{x}) \tag{1}$$

The nature of event can vary widely, infection and patient deaths are typical examples. Survival analysis is the preferred choice when posing research questions concerning timing and duration [11, p. 14-15].

The conditional rate that an event occurs at a particular time $t$ is called the hazard rate $h(t|\mathbf{x})$. The model's goal is not only to examine the effects on the time until an event occurs, but also to assess the relationship between survival time and explanatory variables. The explanatory variables estimate the impact of certain characteristics on the hazard rate. The estimates are case specific but could be anything from receiving treatment to the size of a tumor or level of education depending on the study. The variables can be fixed, unchanged across time such as place of birth or time-varying such as age. Time is at the essence of survival analysis, therefore models implementing survival analysis needs to measure time in units, the granularity can vary from seconds to days to months and so on. Time can be measured continuously or discreetly depending on data availability and modeling approach. [11, p. 16-17]

A unit failure is related to the survival function $S(t|\mathbf{x})$ through the probability density function $f(t|\mathbf{x})$ which describes the distribution of event times. This relationship can be expressed through the hazard rate $h(t|\mathbf{x})$, see Equation 2. The hazard rate indicates the rate at which the unit experiences an event at $t$, given covariates $\mathbf{x}$ and that it has survived up to $t$. [11, p. 21-23]

$$h(t|\mathbf{x}) = \frac{f(t|\mathbf{x})}{S(t|\mathbf{x})} \tag{2}$$

Similarly, if the hazard function $h(t|\mathbf{x})$ is known, the survival function can be derived through the cumulative hazard function. The cumulative hazard up to time $t$ is defined as the accumulated instantaneous risk over time, see Equation 3. The survival probability can then be expressed as the probability of surviving all time steps up to $t$, see Equation 4. Survival thus decreases relative to the total accumulated hazard over time. [11, p. 21-23]

$$H(t|\mathbf{x}) = \int_0^t h(u|\mathbf{x})du \tag{3}$$

$$S(t|\mathbf{x}) = \exp(-H(t|\mathbf{x})) \tag{4}$$

Another appeal of survival analysis is that it takes censoring into account. There are different kinds of censoring but the most common one is right-censoring. Censoring can be defined as knowledge about an individuals survival time but not knowing the exact survival time. An uncensored case contains information about both the starting and ending times of events whereas right-censoring means that the event has not been experience by the last observation. When right-censoring is a result of a non-random process such as the study ending it is referred to as Type 1 censoring. The advantage of survival analysis

is that it accounts for censoring by treating it as a gain in knowledge regarding survival times instead of treating censoring as a loss of information. However, this assumes that censoring is random and that the underlying mechanisms of censoring and occurrences of events are independent of one another. [11, p. 18-19]

Truncation is similar to censoring but refers to a complete lack of information about events. The most common type of truncations is left-truncation, meaning that there is no information about the object before the beginning of the study. Other common truncations are interval-truncation where an individual leaves the study only to return at a later time. The data points from that individual is missing and can't be used to calculate hazard rate during their disappearance from the study. [11, p. 19]

Survival analysis models are divided into 3 groups: non-parametric, semi-parametric and parametric. Non-parametric models do not assume the shape of the hazard function or how covariates may affect that shape. Semi-parametric, just as non-parametric makes no assumption about the shape of the hazard function however it does make assumption about covariates affecting the shape of the hazard function between groups over time. In contrary to non-parametric models, the semi-parametric models allows for the inclusion of multiple covariates and multivariate analysis. Parametric models are similar to semi-parametric but they require the shape of the hazard function to be specified as well as how covariates might impact the function. This allows for more precise estimations but requires a solid understanding of the shape of the hazard function. [11, p. 25-27]

The proportional hazards assumption is widely used in survival analysis and forms the basis of several standard modeling approaches, in particular the Cox proportional hazards model. This assumption states that covariates have a multiplicative effect on the hazard that is constant over time [2]. In other words, the ratio of hazards between two individuals with different covariate values does not change throughout the follow-up period. As a result, effects can be interpreted through constant hazard ratios, simplifying interpretation and comparison across individuals. This assumption is not required in all survival models; alternative approaches allow covariate effects to vary with time or to act in a non-proportional manner.

### 3.2.1 Kaplan-Meier Survival Analysis

Kaplan-Meier is a non-parametric method of estimating survival probability in the presence of censored cases. The model estimates conditional probabilities at each time point when an event occurs, and calculates the product limit of those probabilities to estimate a survival rate of each point in time [12]. The model thus bases the survival of each subject on the overall amount of events thus far in time and the size of the active population. Given all event times $t_e$, the probability of survival at the end of time $T$ [12] is calculated as:

$$\hat{S}_{km}(T) = \prod_{t_e \leq T} p(t_e),$$
$$\text{where} \quad p(T) = 1 - \frac{\text{\#Events at T}}{\text{\#Subjects recorded at start of T}}. \tag{5}$$

The model thus only predicts the survival based on each event time $t_e$, leading to the survival probability only changing when an event occurs. As only the recorded subjects contribute to the survival probability at each time step, censored subjects do not influence the probability but are assumed to have the same survival probability as the rest of the population [12]. Using this model, a null hypothesis regarding survivability of different groups can be tested simply by calculating the Kaplan-Meier probability for each group separately. The model is used in this study as a non-parametric baseline for comparison, a so called null model.

## 3.3 Reliability Engineering

While Survival Analysis has historically been used primarily for medicinal studies, the concept and models are commonly utilized in other areas. Utilizing the time-to-event nature of survival analysis models, applications have been developed such as estimating lead times for production [13], modeling the rate of exoneration for prisoners on death-row [14] and most similar to this work is reliability engineering. It revolves around analyzing product and system performance, often minimizing probabilities of different types of failure [15]. In this type of study, time is more flexible, either indicating a number of operations or classical calendar time. It models reliability similar to survival, where the reliability function of a time point $t$ is the probability that the unit does not fail at any previous time steps before $t$. Reliability engineering encapsulates more than the maintenance phase, it stretches to production, shipping, testing and repair. The goal of reliability engineering is often to get insight into how to manufacture better and more reliable products, which is different from the analysis in this study. Therefore this study utilizes models and methods more commonly related to classical survival analysis, since the goal aligns more closely with modeling when individual units fail instead of gaining insight into the products manufactured reliability.

# 4 Data

The data used in this study is timeseries data consisting of counters, alarms and restarts. It was gathered from all radio units in the network of the chosen customer and product. The choice of product and customer was made based on the set scope of Massive MIMO, data availability and number of units deployed in field, to ensure the best result in model training.

## 4.1 Counters

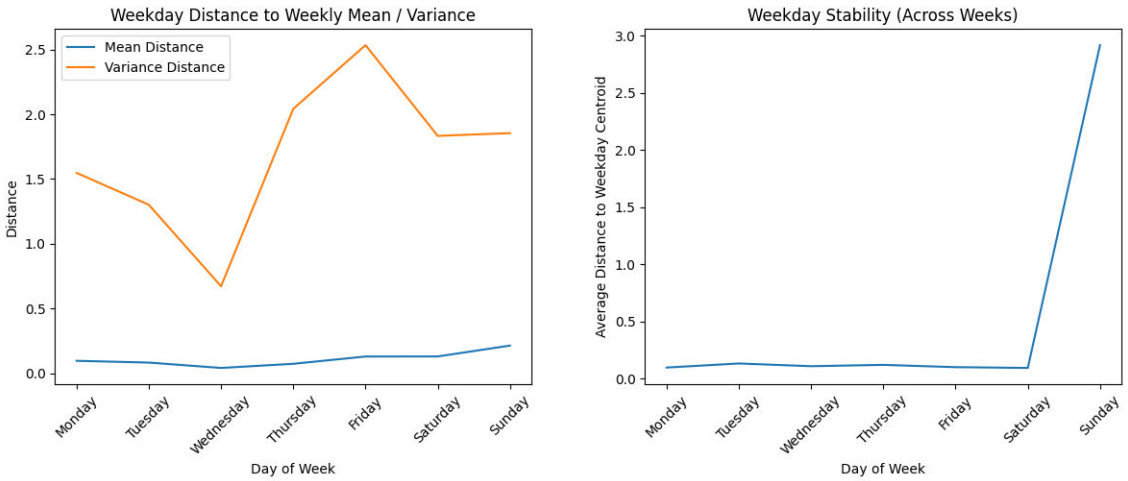
The main data source, from here on called counters, are hardware and software measurements from the radio units which are collected over 15 minute periods, with either counts, binary or mean values depending on the type of measurement. For example, a counter can be an indicator that a certain event happened (binary), a measurement of temperature (mean), or a measurement of traffic (sum). Since the aim of the study is to investigate

why customers chose to replace units, a qualitative approach was used for counter selection. To capture the customer's perception of radio performance, higher-level software measurements were prioritized, as these more directly reflect the impact experienced by end users. As such, more hardware adjacent features for instance power consumption and temperature were discarded in favor of traffic and signal health metrics. The final data set consists of counters that are both direct measurements and Key Performance Indicators (KPIs) that are derived with formulas from multiple counters, see Table 1.

Table 1: Counter groups and features per group

| **Category** | **Number of features** |
|---|---|
| Traffic | 5 |
| Resource | 7 |
| Accessibility | 4 |
| Mobility | 3 |
| Retainability | 3 |
| MAC | 6 |
| Availability | 2 |

Full 15-min granularity over one year of counter data for all available units was not feasible to either ingest or train the model on due to hardware limitations, a representative sample had to be used instead. The data was sampled in two ways; across time and a sampling of units. For the sampling across time, an initial exploratory sample of three weeks of fully granular counter data was ingested to find the best weekday to sample for the full year time span. To compare how weekdays differed, the mean and variance of each counter was calculated in two different ways. First, over the full time period of three weeks. Then only across single weekdays, for example the mean and variance across three consecutive Mondays. The euclidean distance between the full time period and the weekday specific mean and variance thus represent how similar they are. Wednesdays were closest to the weekly mean and the weekly variance across all counters, see Figure 1a. Stability was then calculated by deriving the mean for all counters for each weekday and week, and then comparing each week of data to see how much each weekday differed across weeks. Saturdays had the highest stability meaning it had the most similar values across the three weeks, see Figure 1b. However, stability was very similar for all weekdays except Sundays, and thus Wednesdays were chosen as a representative weekday to ingest counter data for the range of one year. While this three week sample is not fully representable over how user and radio behavior differ across weekdays for a whole year, this quantitative approach should give better results than choosing a weekday at random.




(a) Distance to overall mean and variance centroid per weekday, lower is better

(b) Stability across multiple weeks per weekday, lower is better

Figure 1: Plots showcasing the metrics used for choosing a representative weekday

For the unit sampling, the negative class, i.e. non replaced units, was under-sampled. While this inflates the replacement rate in the data, it is more important to capture all available patterns and behaviors that lead to a replacement than to adhere to the class ratios of the real world data.

## 4.2 Alarms & Restarts

Ericsson utilizes a system, referred to as Radio Fault Management, to monitor the software and hardware of their units. The radio software supervises itself and when it detects persistent problems it tries a local recovery first. If this fails, the fault is sent to the radio fault manager and recorded in the logs. The manager does its own processing, fault prioritization and records what the fault is related to. From there it requests a fault indication signal sent to a radio server that carries the fault indication to the associated clients. The clients transform the indications to customer visible alarms with additional information. The alarms are standardized across products and monitor abnormal behavior in specific components or functionalities, such as power, temperature or software. Different alarms require different conditions to trigger actions, some alarms are raised on immediate detection others require conditions such as the alarm triggering twice during a set time period. Over 70 unique alarms exists, and to map all of them using one-hot encoding would add 70 low variance features. To mitigate this, the alarms were mapped and encoded to 18 categories, where the categories represent types of alarms according to affected components or functionalities. Grouping the alarms allows for dimensionality reduction without significant reduction in granularity. Alarms are infrequent and limiting the alarm data based on the sampled day results in a substantial loss of information. Instead the alarm count is summed over each week, so that for every 15 minute entry in a day the alarm features are static and represents the unit's total number of alarms of that category for that week.

Radio units are capable of restarts, that can either be manually triggered or triggered

automatically by the radio software. These restarts are important to capture as they often happen due to abnormal radio behavior, for example if the radio looses connections it may restart to fix it. Restarts can also happen as an attempted solution for alarms, if the alarm does not trigger post-restart it is not captured in data collection. These restarts are thus included as a complement to alarms, as they may capture events that alarms do not. Since restarts can be manually triggered by the customers network operators it may also indicate that they are observing abnormal radio behavior. Restart data includes reason and explanation, however it is not feasible to split all restart types into different class features as it would add over 20 new low variance features. Restarts were due to this categorized similarly to alarms. Just like with alarms, ingesting all restart data for the time span of one year was feasible and it was summed up over weeks similarly to the alarms data.

## 4.3 Final Data Selection

After the initial data selection, data pre-processing and cleaning was conducted. Since the features consists of both direct measurements and KPIs derived from measurements, many features had high correlation, for example downlink traffic in raw bits existed as a measurement and downlink traffic in gigabytes was a derived KPI. Pair-wise correlation was calculated for all features, and after dropping redundant features based on this list less than 50% of features remained. The criteria for high correlation was 90%, this somewhat high criteria was chosen since many counters can be affected by the same type of issues but capture distinct outcomes. Thus a correlation of under 90% is deemed as insufficient for removal as the information gain beyond the correlation is important to capture. The variance for each feature was also calculated, where features with zero or near-zero variance was dropped, here mainly alarm categories were removed. After these steps, the dimensionality of the data was heavily reduced and redundancy was lowered, which both increases model performance and reduces training time.

Finally, the replacement event was needed to train the survival analysis models. For all replaced units the event was set to 1, everywhere else event was set to 0. Some of the replaced units were repaired and installed in the network again during the observation period of the study. To handle these units in the best way, a suffix on the serial number key column was added to separate these units as unique, since repaired units should not show the same faults as they did before replacement. To capture the time from observation until event/censoring, the duration was calculated as the delta between the observation timestamp and the last observed date in full weeks. The final dataset consists of 42 features and five metadata columns, see Appendix A.

Important to note is that the data likely violates the proportional hazards assumption, as the effect of several predictors changes over time. Certain alarms indicate high immediate replacement risk, but if no replacement occurs shortly after, the alarm may instead represent a false positive and the associated risk decreases. Similarly, high unit downtime may reflect either actual unit failure or non-failure events such as manual shutdowns or system updates, resulting in non-constant risk effects. Additionally, the data is strongly influenced by user behavior, where low traffic may indicate either unit degradation or simply reduced user activity. Consequently, hazard ratios are unlikely to remain constant over time.

## 4.4 Event Time Shifting

The event that the models are trained to predict is replacements, which are caused by observed unit degradation. This can either be due to drops in performance or overall unit failure. When a unit fails, meaning it is no longer operational, it is captured in the availability counters which show the overall down time of a unit. For this dataset, it was observed that close to half of all replaced units had a downtime higher than 95% for 1-9 weeks before replacement. This is interpreted as a late replacement, where the unit was non-operational for some time before being replaced due to delays in replacement unit delivery or unavailability of service staff. This had to be managed since non-operational units show no informative value in any other feature, leading to the models over-indexing on availability counters and making them poor at predicting any other type of unit degradation. It would also misalign with the overarching goal of using replacements as a proxy for unit failure, since the unit had failed long before the replacement event. The solution implemented was to shift the event time for these units to just before they went non-operational, subsequently dropping data where the down time was too high before a replacement. Overall, these units had their event time shifted by an average of 3.7 weeks.

# 5 Method

This paper implements a survival analysis approach for modeling the time to replacement of in-field radio units. Since the proportional hazards assumption does not hold in this study, see Section 4, flexible machine learning models that avoids this are used instead. The paper evaluates non, semi and fully parametric models, with the non-parametric Kaplan-Meier model as a baseline. In this section, the models that are trained and the chosen metrics for evaluation are presented.

These models utilizes the feature vector of an input $i$ denoted as $\mathbf{x}_i$, as covariates to produce the output which is the estimated survival function $\hat{S}(t|\mathbf{x}_i)$. The models are trained using the observed survival time $y$ and event of replaced or censored.

## 5.1 Dynamic Survival Modeling

The models used in this study require input in a duration–event format, where each observation consists of a feature vector, the time until event or censoring occurs (duration), and the outcome (event). However, the dataset in this study contains time-varying co-variates, where each unit is observed repeatedly over time. To align the data with the model assumptions, a dynamic survival format was applied. For each unit, the measurement from a sampled day is seen as an input record, treating the state at that time as the baseline condition. The model is then trained to predict the remaining survival time conditional on the observed state. This results in multiple inputs per unit, each representing a different point in the longitudinal trajectory. This transformation is justified by two key considerations. First, the operational state of a unit evolves over time, with co-variate values that can change substantially between observations, making a single baseline representation insufficient. Second, replacement decisions are primarily based on the current observed

state rather than the full historical trajectory, motivating a formulation that conditions predictions on the most recent information. By reformulating each observation as a conditional survival problem, the model learns the risk associated with the current state rather than relying on historical trajectories. For hyperparameter tuning and test results, the data was split based on unique units. This prevents data leakage since each unit contributes multiple inputs in this format.

## 5.2 Random Survival Forests

Random Forests are an ensemble learning method introduced by Breiman [16]. The model consists of multiple decision trees that aggregate their predictions to determine the final output, a technique known as bagging.

Each tree is grown by drawing a sample of the data using bootstrapping, where about one third of the original data is left out of this sample. At each node a split is determined by selecting $m$ variables from random out of $M$ input variables and ranking the splits. The value of $m$ is constant during forest growing and has to satisfy the condition $m << M$. After each tree is constructed the left out data is used to get the Out Of Bag (OOB) error estimate, ensuring an unbiased estimate of the test set error. Then all data is sifted down the tree and proximities are calculated. Proximities are used to replace missing data and locate outliers by increasing the proximity of two cases if they are in the same terminal node. Lastly the proximities are normalized by dividing with the number of trees. [17]

Random Survival Forests (RSF), originally implemented by Ishwaran et al., extends Random Forests by handling right censored data [3]. It utilizes the outcome of the unit, survival time and censoring information, when growing the forests. The splitting criterion uses survival time and information about censoring to create child nodes. Typically survival analysis models use restrictive assumptions, such as proportional hazards. They are often parametric and require transformations or basis functions to model non-linear effects. Identifying non-linear interactions require brute-force comparison between multiple variables or specialized domain knowledge to narrow the search. Random Survival Forest, however, has inherent solutions to these problems [3]. As a data driven model it avoids assumptions such as proportional hazard and naturally captures non-linear effects and interactions. In high dimensional data, as in this case, Random Survival Forest avoids overfitting, unreliable estimation of coefficients and convergence issues while providing a framework robust against outliers [18].

The Random Survival Forest algorithm is similar to Random Forest but incorporates survival analysis. It works by drawing a bootstrap sample from the original dataset, where approximately one-third of the observations are left out as Out-Of-Bag (OOB) samples. A survival tree is grown for each sample. In every node of the tree, $m$ candidate variables are randomly selected. The node is split using the variable and split point that maximizes the survival difference between child nodes [3]. The splitting is recursively performed until a stopping criterion is met that requires a minimum number of unique events per node. The leaves are called terminal nodes.

In this work the model was implemented using scikit-survival [19], a Python library designed for survival analysis built on top of scikit-learn. In this implementation, the model

output is derived utilizing the built-in function `predict_survival_function` which utilizes the Kaplan-Meier estimator, it returns a list of step functions corresponding to the survival function of the input data. For each tree, a Kaplan-Meier curve is derived in each terminal node, computed from all the outcomes of observations in the bootstrap sample that belong in that node. The ensemble survival function $\hat{S}(t|\mathbf{x}_i)$ for an input $\mathbf{x}_i$ is the average value of the Kaplan-Meier estimator across all trees in the ensemble, based on the terminal nodes in which $\mathbf{x}_i$ belongs [19].

The Random Survival Forests were trained using different hyperparameters based on parameter grid searches. The grid search evaluated `depth`, `n_estimators`, `min_samples_leaf` and `max_features`. For all parameters except `max_depth`, three different values were evaluated, while `max_depth` evaluated 4 different values, see Table 2 for the values used. The candidate models were chosen by searching for the highest concordance index in different depth groups. Max depth was chosen as the unique identifier parameter as it made the biggest difference in model performance. The specific parameters for each candidate model can be seen in Table 8 in Appendix B. Since Random Survival Forest only supports one value per feature and the features have 96 values per weekly observation sample, the data needed to be aggregated. This was done by calculating the mean. While other values, such as minimum, maximum and variation can also be calculated, it would create multiple new features which would be hard to analyze and compare in the evaluation of feature importance.

Table 2: RSF model grid-searched values.

| Parameter | Value 1 | Value 2 | Value 3 | Value 4 |
|---|---|---|---|---|
| max_depth | 3 | 5 | 10 | none |
| n_estimators | 100 | 200 | 300 | - |
| min_samples_leaf | 5 | 10 | 20 | - |
| max_features | sqrt | log2 | 0.5 | - |

## 5.3 Weibull Accelerated Failure Time

The Weibull Accelerated Failure Time model is part of a family of regression models called Accelerated Failure Time (AFT) models, which was first proposed by Newby [20]. The model asserts a direct relationship between failure time and covariates [21, p. 219]. The assertion specifies that AFT models use covariates to act multiplicatively on failure time $T$ or additively as $Y = log\ T$ [21, p. 218]. AFT models propose that two populations $A$ and $B$ have different survival functions, $S_A(t)$ and $S_B(t)$, which are related by some accelerated failure factor $\lambda$, which is a function of covariates [22], see Equation 6.

$$S_A(t) = S_B\left(\frac{t}{\lambda}\right), \quad \text{where } \lambda(\mathbf{x}) = \exp(\beta_0 + \mathbf{x}^T\boldsymbol{\beta}) \tag{6}$$

The model can therefore accelerate or decelerate failure times depending on the subjects' covariates [22]. The Weibull AFT model is based on the Weibull distribution. It is well suited for use in AFT models since it can include non-constant failure rate functions, allowing it to account for both accelerating and decelerating failure rate functions [15].

The probability density function for the Weibull distribution can be seen in Equation 7 [15] .

$$\hat{f}(t;\lambda(\mathbf{x}),\rho) = (\frac{\rho}{\lambda(\mathbf{x})})(\frac{t}{\lambda(\mathbf{x})})^{\rho-1} \quad \exp\left(-(\frac{t}{\lambda(\mathbf{x})})^{\rho}\right) \quad \text{where} \quad t \geq 0; \lambda(\mathbf{x}), \rho > 0 \tag{7}$$

The shape parameter $\rho$ controls the shape of the distribution while the scale parameter $\lambda(\mathbf{x})$ scales the distribution. The hazard function determines the behavior of the hazard rate [15], which is calculated as seen in Equation 8.

$$\hat{h}(t;\lambda(\mathbf{x}),\rho) = (\frac{\rho}{\lambda(\mathbf{x})})(\frac{t}{\lambda(\mathbf{x})})^{\rho-1} \tag{8}$$

If $\rho < 1$ the hazard is decreasing over time, implying higher risk at earlier time steps. Meanwhile, if $\rho > 1$ the hazard is increasing over time and indicating that failure is associated more with deterioration.

In this work the Weibull AFT model was implemented using the Python package Lifelines [22]. This model uses the covariates in the scaling parameter, $\lambda(\mathbf{x}_i)$, corresponding to multiplicative effects on the time scale to accelerate or decelerate time. The survival function for the Weibull distribution [15] can be seen in Equation 9.

$$\hat{S}(t|\mathbf{x}) = \exp\left(-(\frac{t}{\lambda(\mathbf{x}_i)})^{\rho}\right) \tag{9}$$

The lifelines implementation requires the inputs to be represented by a static feature vector, since time-varying covariates are not supported. The features were thus aggregated to the mean of each observation sample, as done for Random Survival Forest. The package contains a built-in method to calculate the survival functions called `predict_survival_function`. The function calculates $\lambda(\mathbf{x}_i)$ by inserting feature values and then inserting $\lambda(\mathbf{x}_i)$ into Equation 9 [22]. The Weibull AFT model was trained with differing strengths of the L2 penalizer, also known as Ridge regression [22]. It ranged from 0.001, 0.01, 0.1 to 1.0. As with other regression models, the Weibull Accelerated Failure Time model may be sensitive to low variance covariates as they may affect numerical stability while training the model. Due to this feature selection varied depending on data split to ensure stable model estimation. Depending on the dataset used during training, between 3 and 8 features were removed.

## 5.4 Neural Network Survival Models

As machine learning methods for survival analysis have gained increasing attention, there has been a growing interest in applying neural networks to time-to-event data. Neural networks offer advantages such as the ability to capture complex non-linear relationships and handle high-dimensional covariates. However, their application to survival analysis is not straightforward, primarily due to the presence of right-censored data and the need

to model time-to-event outcomes. To address these challenges, several strategies have been proposed. One common approach is to discretize the time axis into a finite number of intervals, enabling the model to predict outcomes over a fixed set of time bins. This formulation allows the use of standard neural network training techniques while naturally accommodating censoring. Models based on this idea are often referred to as discrete-time survival models. In this study, two such models are evaluated; Logistic Hazard and DeepHit. [23]

### 5.4.1 Logistic Hazard

Logistic Hazard is a discrete-time hazard model originally proposed by Gensheimer and Narasimhan [24]. The model was proposed to enable a neural network survival analysis model that can be trained fast with mini-batch gradient descent, and avoids the proportional-hazards assumption required by Cox based models. The model outputs the estimated conditional hazard probability $\hat{h}(t|\mathbf{x}_i)$ which is the probability that an individual $i$ with feature vector $\mathbf{x}_i$ fails in time interval $t$ given that it has survived up until that interval. From that the estimated survival for each interval can be derived, see Equation 10.

$$\hat{S}(t|\mathbf{x}_i) = \prod_{j=1}^{t} 1 - \hat{h}(j|\mathbf{x}_i) \tag{10}$$

The loss function is the sum of the negative log likelihood, which can be divided either by time intervals or individuals. As the model is trained in mini-batches, the model divides the likelihood over individuals [24], see Equation 11.

$$\begin{aligned} \log L_{\text{event}}(\mathbf{x}_i) &= \ln(\hat{h}(t|\mathbf{x}_i)) + \sum_{j=1}^{t-1} \ln(1 - \hat{h}(j|\mathbf{x}_i)) \\ \log L_{\text{censored}}(\mathbf{x}_i) &= \sum_{j=1}^{t-1} \ln(1 - \hat{h}(j|\mathbf{x}_i)) \end{aligned} \tag{11}$$

Here, the likelihood is separated for censored and uncensored observations at time $t$. The probability of an event at time $t$ is given by surviving up to $t-1$ and then experiencing the event at $t$, whereas the likelihood for censored observations corresponds to surviving up to $t-1$. The full loss function is defined as the negative log-likelihood over all individuals [24], see Equation 12, which is minimized using gradient descent

$$\log L = -\sum_{i=1}^{N} (\delta_i \log L_{\text{event}}(\mathbf{x}_i) + (1 - \delta_i) \log L_{\text{censored}}(\mathbf{x}_i)) \tag{12}$$

This study uses the pycox Python package [25], where the Logistic Hazard model is implemented using a binary cross-entropy–based formulation of the negative log-likelihood, enabling mini-batch training. The framework is built on PyTorch and allows for flexible

specification of neural network architectures, making it adaptable to different types of input data.

### 5.4.2 DeepHit

DeepHit is a survival analysis model proposed by Lee et al. [4] that implements deep neural networks to directly learn the distribution of survival times. The model makes no parametric assumptions about the distributional form of survival times, just like Logistic Hazard. Unlike Logistic Hazard, which models the hazard function directly, DeepHit does not explicitly parameterize hazards but instead learns the full distribution of event times. The model outputs a discrete probability mass function $\hat{p}(t|\mathbf{x}_i)$ given an input $\mathbf{x}_i$ for each specified discrete time bin $t$ directly. From this the cumulative event probability $\hat{F}(t|\mathbf{x}_i)$ can be derived, see Equation 13. The estimated survival function can then be derived as seen in Equation 14.

$$\hat{F}(t|\mathbf{x}_i) = \sum_{k=1}^{t} \hat{p}(k|\mathbf{x}_i) \tag{13}$$

$$\hat{S}(t|\mathbf{x}_i) = 1 - \hat{F}(t|\mathbf{x}_i) \tag{14}$$

The model is trained using a specific loss function which combines two different types of losses. The first loss $L_1$ aims to make the model learn the distribution of the time to event utilizing both censored and uncensored inputs, see Equation 15. The second loss $L_2$ utilizes ranking loss based on concordance, meaning inputs with event time $t$ should have a higher predicted cumulative risk at $t$ than an input which survived past $t$, see Equation 16.

$$L_1 = -\sum_{i=1}^{N} \left[ \mathbf{1}(\delta_i = 1) \, \log \hat{p}(s_i|\mathbf{x}_i) + \mathbf{1}(\delta_i = 0) \, \log \left( 1 - \hat{F}(s_i \mid \mathbf{x}_i) \right) \right] \tag{15}$$

$$\begin{gathered} L_2 = \alpha \sum_{i \neq j} A_{ij} \, \eta\Big( \hat{F}(s_i \mid \mathbf{x}_i), \hat{F}(s_i \mid \mathbf{x}_j) \Big) , \\ \text{where} \quad A_{ij} = \mathbf{1}\big(\delta_i = 1, \; s_i < s_j\big) , \qquad \eta(a, b) = \exp\left( -\frac{a - b}{\sigma} \right) . \end{gathered} \tag{16}$$

In the loss functions, $\mathbf{1}(\cdot)$ denotes an indicator function, where $\delta_i = 1$ indicates that an event was observed and $\delta_i = 0$ indicates censoring. Furthermore, $s_i$ denotes the observed survival time for individual $i$, which corresponds either to the event time or the time of censorship. The model utilizes the hyperparameters $\alpha$ and $\sigma$, where $\alpha$ controls how much weight is put on $L_2$ in the total loss and $\sigma$ controls how sensitive $L_2$ is to differences in predicted risk.

DeepHit is implemented in the pycox python library [25], specifically DeepHitSingle which handles only one event and is used in this study. It allows for the same network flexibility using PyTorch as the Logistic Hazard implementation.

### 5.4.3 Neural Network Architecture

For Logistic Hazard and DeepHit, the neural networks were implemented in PyTorch. Since the daily data consists of multiple measurements over time, see Section 4, two main types of layers were used: one-dimensional convolutional layers and long short-term memory (LSTM) layers. These were employed in the feature extraction component of the network, while fully connected dense layers were used in the prediction component. Furthermore, batch normalization and dropout layers were incorporated to stabilize training and reduce overfitting. Dilation of convolutional layers were also implemented to see their impact on performance. Given the wide range of possible neural network architectures, this study evaluates four distinct network architectures with different structural properties, as illustrated in Figure 2.

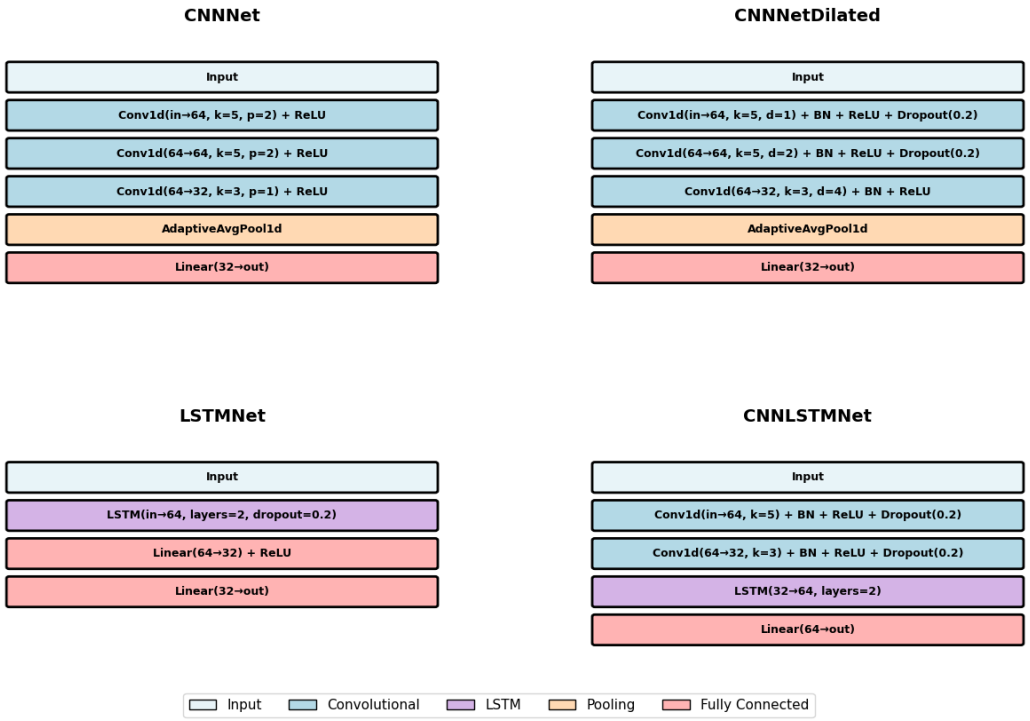



Figure 2: The four different network architectures implemented for DeepHit and Logistic Hazard

Each network architecture for both DeepHit and Logistic Hazard was tuned with respect to the concordance index, using learning rate and batch size as hyperparameters, see Table 9 in Appendix C. To ensure comparability between architectures, the DeepHit loss-specific hyperparameters $\alpha$ and $\sigma$ were tuned on CNNNet and kept fixed across all experiments. While this may limit the optimal performance of other architectures, it reduces differences in results that arise due to objective-specific tuning of the loss function and allows the comparison to focus more directly on architectural differences.

## 5.5 Model Evaluation

In this section the metrics that are used to evaluate the model performance are presented. The models were evaluated over 5-fold cross validation, to ensure that performance dif-

ferences are not based on data splits. Since survival analysis does not simply predict a class probability but instead a survival probability in time steps, classification metrics are not directly applicable. The metrics presented utilize the models predicted survival probability curve over time for evaluation, when risk score is necessary for ranking metrics it is computed as $\hat{f}(t|\mathbf{x}_i) = 1 - \hat{S}(t|\mathbf{x}_i)$.

### 5.5.1 Concordance Index

A commonly used metric for Survival Analysis models is the concordance index, introduced by Harrell et al 1982 [26]. The metric compares all available subjects pairs using the models predicted risk score and the observed survival time. Subjects that are censored can not be compared to each other, similarly subjects that experienced the event at the same time can not be compared either. A pair is deemed concordant if the subject with a higher predicted risk has a shorter observed survival time. The metric computes the following for all available pairs:

- If the pair is concordant a 1 is added to the count
- If the pair is not concordant a 0 is added to the count
- If the pair is non-comparable it is excluded

The final concordance index value is the mean of the total counts, with a 0.5 indicating the model is as good as random guessing and a 1 indicating that the model ranks all subjects correctly. Harell's concordance index evaluates discrimination based on a single risk score per subject and therefore does not explicitly account for how risk evolves over time. As a result, it provides a global measure of ranking performance but ignores potential time-dependent effects. This limitation is particularly relevant in settings where the relative risk between individuals may change over time, such as in models that do not satisfy the proportional hazards assumption. In such cases, a single static ranking may not adequately capture the model's performance, motivating the use of time-dependent evaluation metrics.

Antolini et al. [27] introduced a time-dependent concordance index that evaluates the discriminative ability of survival models using predicted survival functions over time. The metric is based on pairwise comparisons of individuals at observed event times $t$. For each event time, the method compares predicted survival probabilities derived from the model's survival curves, evaluating if individuals with earlier events are assigned higher risk than those surviving beyond the same time point. This formulation allows model discrimination to be evaluated across different time horizons, making it more suitable for settings where risk rankings may change over time and where the proportional hazards assumption may not hold. In this work, the implementation provided in the pycox framework [25] is used, which follows the Antolini-style time-dependent concordance index. This implementation includes modified handling of ties in event times and predicted survival probabilities to improve numerical stability. From here on in this paper, the concordance index will refer to the pycox implementation of Antolini's time-dependent concordance index.

### 5.5.2 Time dependent Dynamic AUC

The time dependent dynamic area under the Receiver Operating Characteristics (ROC) curve is an extension of the ROC curve commonly used in binary classification. The binary classification ROC curve measures the performance of a classifier by plotting the true positive rate (y-axis) and the false positive rate (x-axis) across a range of classification thresholds [28]. The area under the curve represents the probability of the model correctly ranking a positive example over a negative example, with a perfect score being 1.0 meaning it correctly classifies every input. For the time dependent dynamic AUC the metric is extended for survival analysis by defining inputs based on a given time point $t$ [29], see Equation 17. Cases are subjects that experience an event prior to or on time $t$, while the rest are controls. Dynamic controls are subjects who experience the event post time $t$ while the rest of the controls are censored.

$$\hat{AUC}(t) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} I(y_j > t)I(y_i \leq t)\omega_i I((\hat{f}(t|\mathbf{x}_j) \leq \hat{f}(t|\mathbf{x}_i))}{(\sum_{i=1}^{n} I(y_i > t))(\sum_{i=1}^{n} I(y_i \leq t)\omega_i)} \tag{17}$$

$$I(condition) = \begin{cases} 1 & \text{if true} \\ 0 & \text{if false} \end{cases} \tag{18}$$

[29]

Here $\hat{f}(x_i)$ is the estimator of the individual's risk score, $\omega_i$ is the inverse probability of censoring weights derived using Kaplan-Meier estimation and $y_i$ is the observed survival time for the individual. The metric exists as part of the scikit-survival python package [29], where the mean of multiple AUC scores across a specified range is also calculated, weighted by the estimated Kaplan-Meier survival function $\hat{S}_{km}$, see Equation 19. AUC was computed for 10 time steps equally spaced from the 0th to the 90th percentile of the event times in the test set.

$$\overline{AUC}(\tau_1, \tau_2) = \frac{1}{\hat{S}_{km}(\tau_1) - \hat{S}_{km}(\tau_2)} \int_{\tau_1}^{\tau_2} \hat{AUC}(t) d\hat{S}_{km}(t) \tag{19}$$

### 5.5.3 Inverse Probability of Censoring Weights

The Inverse Probability of Censoring Weights (IPCW) originally proposed by Robins et al 1995 [30] is an estimator that can compensate for censoring in observational studies. The estimator assumes that once a subject is censored it will not reappear in the study, and censoring may be dependent on historical explanatory variables. These variables have a correlation with the outcome and the censoring probability, for example sicker patients are more likely to die but also to drop out of the study. This would bias the model such that, in the case of healthcare studies, observing the effects of certain treatments would not reflect reality as mostly healthier patients remain in the study. The key idea is that uncensored subjects at a time $t$ are weighted to represent themselves and similar subjects who were censored before $t$. The probability that a subject $i$ remains in the study at time $t$, $P(R_{it} = 1)$, can be calculated using the known explanatory variables, outcomes, and features from the previous timestep. The IPCW is then estimated by assuming that

this probability is lower bounded by zero and calculating the inverse, see Equation 20. The IPCW is commonly used in evaluation metrics for Survival Analysis to account for censoring, where it is most often derived using the Kaplan-Meier estimator.

$$\omega_i = \frac{1}{P(R_{it} = 1)} \tag{20}$$

### 5.5.4 Brier Score

The Brier score, named after and proposed by G,W Brier (1950) [31], is an evaluation metric originally used to verify the accuracy of weather forecasts. He proposed that for forecasts expressed in probability statements, a verification score can be defined based on the assumption that for $n$ occasions, one of $r$ possible mutually exclusive events occur. Given a forecast probability $f_{ij}$ for occasion $i$ and event $j$ (where $\sum_{j=1}^{r} f_{ij} = 1$), and an observed outcome $E_{ij} \in \{0, 1\}$ based on if the event $j$ occurred at $i$, the Brier verification score is formulated as:

$$P = \frac{1}{n} \sum_{j=1}^{r} \sum_{i=1}^{n} (f_{ij} - E_{ij})^2 \tag{21}$$

A low, zero or near-zero score, implies a perfect prediction while a high score implies a bad prediction. This score both awards predicting the correct event and penalizes predicting the wrong event, making sure that predictions that heavily predict the most common event do not gain a low score. If there are only 2 categories (event or no event), as is the case in this study, the formula is considerably simplified, see Equation 22.

$$BS^c(t) = (f_t - E_t)^2 \tag{22}$$

Here $t$ denotes the timestep, $f_t$ is the predicted probability of the event and $E_t \in \{0, 1\}$ is the observed event outcome. This score can then be summed and divided over each subject to get a score for all predictions of a survival analysis model.

Graf et al. (1999) [32] proposed an incorporation of the IPCW in use with the Brier score which uses an estimation of censorship probability derived from the Kaplan-Meier estimator. The authors describe that at a specified time point $t$, a subject $i$, with a survival time of $T_i$ and observation time of $C_i$, where $\delta_i = I(T_i \leq C_i)$ and $\tilde{T}_i = min(T_i, C_i)$, can be in 1 of 3 categories:

1. $\tilde{T}_i \leq t$ and $\delta_i = 1$
2. $\tilde{T}_i > t$ and $\delta_i = 1$ or $\delta_i = 0$
3. $\tilde{T}_i \leq t$ and $\delta_i = 0$

Category 1 subjects are subjects that have a known survival time, as they experienced the event before the observation time ended. Category 2 subjects are known to have survived

until $t$. Category 3 subjects however are censored at $t$, and they cannot be classified to have experienced the event or not. Thus for the Brier score, only category 1 & 2 subjects can contribute since they have a known event status at $t$. To make up for this lack of input from censored subjects, the Kaplan-Meier estimator, $\hat{G}(t)$, can be calculated based on risk of censorship instead of survival. For each time $t$ an estimation of censorship probability is thus known, and the IPCW can be calculated for the three categories:

1. $\omega_i = \frac{1}{\hat{G}(\tilde{T}_i)}$
2. $\omega_i = \frac{1}{\hat{G}(t)}$
3. $\omega_i = 0$

The observations can then be multiplied by the weight corresponding to their category, making censored subjects contribute to the score based on their available data before censorship by weighting the fully observed subjects. Using this the weighted "empirical Brier score under random censorship" is calculated as seen in Equation 23.

$$BS^c(t) = \frac{1}{n}\sum_{i=1}^{n}\left\{(0-\hat{S}(t|\mathbf{x}_i))^2 I(\tilde{T}_i \leq t, \delta_i = 1)\left(\frac{1}{\tilde{G}(\tilde{T}_i)}\right) + (1-\hat{S}(t|\mathbf{x}_i))^2 I(\tilde{T}_i > t)\left(\frac{1}{\hat{G}(t)}\right)\right\} \tag{23}$$

The Brier score can also be integrated over time for a metric that represents how well the model performs overall instead of at specific time points, see Equation 24. The Integrated Brier Score (IBS) uses an additional weight function $W$ which emphasizes predictions made for later time points, as seen in Equation 25. This is due to the nature of survival analysis, early on there are low amounts of censoring and events combined with the fact that long-term predictions are often more useful.

$$IBS = \int_0^{t_{max}} BS^c(t)\,\mathrm{d}W(t) \tag{24}$$

$$W(t) = \frac{t}{t_{max}} \tag{25}$$

The IPCW weighted integrated brier score is implemented in this study utilizing the pycox Python package [25], where the Kaplan-Meier censorship estimation is calculated based on the test set.

### 5.5.5 Precision

In binary classification, precision and recall are two commonly used evaluation metrics to see how well the model can distinguish positives from negatives and find all positives in

a test set [33]. For survival analysis, there is no classification, meaning the models do not explicitly state whether a unit belongs to a class, it simply ranks units in terms of survival probability. To extend these metrics to a ranking output, the metric can be calculated based on the top $k$ ranked units instead, known as $\mathrm{Precision}_k$, see Equation 26.

$$\mathrm{Precision}_k = \frac{\text{Number of relevant items in } k}{\text{Total number of items in } k} \tag{26}$$

In this study, $\mathrm{Precision}_k$ is evaluated at a fixed time horizon $t$ by ranking individuals according to their predicted survival probability at $t$. The top k individuals with lowest survival are selected, and $\mathrm{Precision}_k(t)$ is defined as the proportion of these that experience the event prior to or on $t$, see equation 27.

$$\mathrm{Precision}_k(t) = \frac{1}{k} \sum_{i \in \mathcal{S}_k(t)} \mathbf{1}\{T_i \leq t,\ \delta_i = 1\} \tag{27}$$

$\mathcal{S}_k(t)$ denotes the $k$ individuals with the lowest predicted survival probability at $t$. Observations censored before $t$ are excluded from the evaluation, as their event status within the time window is unknown.

This metric requires a choice for the values of $k$ and $t$ based on the application. For this use case, statistics from the data shows that there are three to four replacements per week, thus $k$ should be based on $t$. For $t$, the operationally significant threshold is somewhere around five weeks, giving time for network operators to act on the highest ranked units. Thus in this thesis, the models are evaluated based on $\mathrm{Precision}_{20}(5)$. From here on, $\mathrm{precision}_\mathrm{k}$ refers to this metric with $k = 20$ and $t = 5$.

## 5.6 Feature Evaluation

In this Section, the methods that are used to evaluate the feature importance and effect on model output is presented. These methods explain how features effects model output as well as model performance.

### 5.6.1 SHAP

SHapley Additive exPlanations (SHAP) [34] is a method to explain the output of machine learning models based on the input features, including black box models like neural networks. The idea behind the explanations is to utilize cooperative game theory, where the model's output is seen as credit allocated to the input features, in this case seen as players in a game.

Lloyd Shapley defined the Shapley value in 1953 [35] to provide a fair distribution of credit among players that contribute unequally towards a common goal. The value was defined for abstract games, where the games rules are set by the involved players and their unique roles. For example, a game consists of players A, B and C, and from that coalitions

can be derived from all possible combination of players. For these abstract games a set of axioms was defined;

1. Symmetry: If two players contribute the same across all coalitions their credit should be the same.
2. Efficiency: The yield of the game must equal the sum of all player's contribution.
3. Additivity: A player's assigned credit in a combined game equals the sum of their assigned credits in each individual game.
4. Null player: If a player contributes nothing to any coalition they do not get any credit.

Consider the game payout $V$, with coalitions $C$, and the set of players $N$, where each player $j$ is assigned a Shapley value $\phi_j$, the value is then derived as seen in Equation 28.

$$\phi_j = \sum_{C \subseteq N \setminus \{j\}} \frac{|C|!(|N| - |C| - 1)!}{|N|!} \left(V(C \cup \{j\}) - V(C)\right) \tag{28}$$

The use of Shapley values for machine learning analysis was first introduced by Štrumbelj and Kononenko [36] in 2010, who aimed to design an explanation method for any type of classifier. They derived a formula by letting each input's feature form a coalition that causes a change in the classifier's prediction, the yield. Then they assign the Shapley values based on how each feature contributes across all possible coalitions. By doing this, the features, coalitions and model output followed the axioms set by Shapley. The method was then further extended and popularized by Lundberg et al. in 2017 [37]. They combined Shapley values with other available machine learning explanation methods to create what they call SHAP explainers. These explainers exist for different type of models, the one used in this study is KernelExplainer which is model agnostic. KernelExplainer expects one value only as the model output. Since the models in this study output a survival curve over time the survival probability at the median event time in the dataset is used as the output to analyze. The SHAP models are available as part of an open source python package [34]. SHAP analysis was done using the KernelExplainer with a background size of 300 inputs, an explanation size of 600 inputs and 400 number of re-evaluations per explanation input.

For the neural network models the input is three dimensional since all 96 time steps of the daily measurements are used as input. Due to this, SHAP values were calculated for all 96 time steps, and the overall SHAP value for each feature is the sum of all 96 SHAP values related to that feature. Subsequently, the feature value shown is the mean over all 96 time steps, to observe the directional relationship of SHAP.

### 5.6.2 Leave One Feature Out

Leave One Feature Out (LOFO) importance is a method for estimating feature importance by retraining the model with one feature excluded at a time [38]. By iterating over all

features and comparing the resulting performance to the baseline model by a difference of a given score $Sc$, the contribution of each feature $j$ to overall model performance can be estimated, see Equation 29.

$$LOFO_j = Sc_{baseline} - Sc_{-j} \tag{29}$$

A negative LOFO importance indicates that the feature hurts model performance, whereas a positive importance indicates that the model performs better with the feature. LOFO is model-agnostic and therefore suitable for this study, where the evaluated models differ substantially in structure. Although LOFO can be applied for feature selection, it is used here primarily as an interpretability method to analyze how individual features influence model performance. In contrast to SHAP, which explains how a trained model utilizes feature values in its predictions, LOFO measures how the removal of a feature affects predictive performance. This distinction makes LOFO particularly useful for understanding the informational value of features and the underlying patterns in the data that the models rely upon. The LOFO importance was computed for each evaluation metric presented in this section, capturing how certain features affect different aspects of model performance. To get the overall LOFO importance of a feature, the mean of the percentage change across all metrics was calculated.

# 6 Results

The result is presented in two parts, corresponding to the two research questions. First the model performance is presented and then the models interpretability are evaluated.

## 6.1 Model Performance

In this section, the results of the models' performance are presented. First, the results of the four best configurations of each model are shown. Then the best performing within each model type is selected as the champion model configuration and compared to each other.

### 6.1.1 Kaplan-Meier Baseline

As mentioned in Section 3, the Kaplan-Meier estimator null model provides baseline evaluation scores that the other models should outperform. Since Kaplan-Meier provides a population wide estimation for survival, the metrics that are based on discrimination are less informative. Both AUC and concordance index results in 0.5 when all inputs have the same survival curve. The $\text{precision}_k$ can not be calculated when all inputs have a tie in predicted survival, however a representative baseline is the event rate in the dataset, which is 6%. What is most interesting is the Kaplan-Meier baseline for the integrated Brier Score, which in this case is $0.0423$. The Kaplan-Meier survival curve is illustrated in Figure 3.

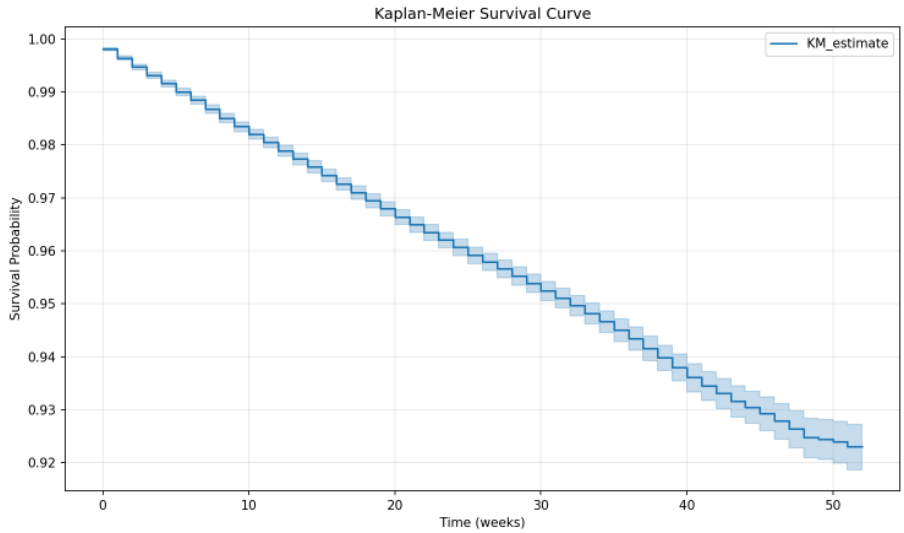



Figure 3: The Kaplan-Meier curve, illustrating survival based on average events and durations in the dataset

### 6.1.2 Random Survival Forests

In Table 3, the different Random Survival Forest model configurations are identified by their depth, as this is the only hyperparameter that varies across the evaluated models. Overall, all configurations achieve similar performance levels, with smaller differences across metrics.

The depth 10 configuration achieves the highest concordance index and the lowest integrated brier score, indicating the strongest performance in terms of ranking and calibration. The depth none configuration performs closely, with a slightly lower concordance index and higher integrated brier score, but it achieves the best AUC score and a higher $\text{precision}_k$ score compared to depth 10.

The shallower configurations, depth 3 and 5, achieve higher $\text{precision}_k$ than the two previously mentioned models but have lower concordance index scores, indicating weaker ranking performance.

Both depth 10 and none can therefore be considered the strongest models, as they outperform across the different evaluation metrics. However, the depth 10 configuration is selected as the champion model. This is primarily based on its superior concordance index, the standard metric for evaluating discriminative performance in survival analysis as well as its lower integrated brier score, the only calibration metric considered in this study. Together, these results indicate a slightly better combined ranking and calibration performance for depth 10, despite depth none demonstrating better scores for AUC and $\text{precision}_k$.

Table 3: RSF performance metrics across depths, mean and standard deviation for five folds. Bold highlights the best result across model configurations.

| max_depth | C-Index | IBS | Mean AUC | $\text{Precision}_{20}(5)$ |
|---|---|---|---|---|
| 3 | 0.6906 ± 0.0489 | 0.0259 ± 0.0022 | 0.7096 ± 0.0484 | **0.4000 ± 0.1061** |
| 5 | 0.7039 ± 0.0474 | 0.0254 ± 0.0022 | 0.7129 ± 0.0520 | 0.4000 ± 0.1275 |
| 10 | **0.7241 ± 0.0486** | **0.0246 ± 0.0027** | 0.7040 ± 0.0553 | 0.2900 ± 0.0742 |
| None | 0.7217 ± 0.0356 | 0.0252 ± 0.0027 | **0.7182 ± 0.0421** | 0.3300 ± 0.1151 |

### 6.1.3 Weibull Accelerated Failure Time Model

Among the evaluated configurations, the Weibull AFT model using a penalizer of 0.01 achieved the strongest overall performance, see Table 4. It obtained the highest concordance index, the lowest integrated brier score and the highest mean AUC. In contrast the highest $\text{precision}_k$ was achieved by the penalizer 0.1 configuration, although the difference compared to 0.01 was relatively small.

Overall the penalizer 0.01 configuration was selected as the champion Weibull AFT model due to its consistently strong performance across the primary survival evaluation metrics.

Table 4: Weibull model performance metrics across penalizers, mean and standard deviation for five folds. Bold highlights the best result across model configurations.

| Penalizer | C-Index | IBS | Mean AUC | $\text{Precision}_{20}(5)$ |
|---|---|---|---|---|
| 0.001 | 0.7006 ± 0.0375 | 0.0260 ± 0.0020 | 0.6878 ± 0.0.0451 | 0.2900 ± 0.1387 |
| 0.01 | **0.7046 ± 0.0421** | **0.0260 ± 0.0019** | **0.7013 ± 0.0499** | 0.2900 ± 0.1636 |
| 0.1 | 0.6796 ± 0.0545 | 0.0266 ± 0.0022 | 0.7005 ± 0.0559 | **0.3000 ± 0.1581** |
| 1.0 | 0.6643 ± 0.0598 | 0.0270 ± 0.0023 | 0.6877 ± 0.0598 | 0.2800 ± 0.1095 |

### 6.1.4 Neural Networks

Overall, the performance is similar for DeepHit and Logistic Hazard, with the architectures showcasing solid performances across the board, see Table 5. For the DeepHit specific hyperparameters, based on tuning results from CNNNet, they were fixed to $\alpha = 0.2$ and $\sigma = 0.1$.

Choosing an overall best architecture for both models is difficult. The worst performing architecture is LSTMNet, which fails to produce a top score for any metric, however the performance is similar to other architectures. For Logistic Hazard, CNNLSTMNet produces the highest concordance index and lowest integrated brier score, while showcasing the next best AUC and $\text{precision}_k$ score just behind CNNNetDilated. Since CNNNetDilated produced the worst concordance index and integrated brier score for the model, CNNLSTMNet was chosen as the best performing Logistic Hazard architecture. For DeepHit, CNNNet showcases a similarly good performance, outperforming the other architectures for concordance index and integrated brier score, while placing second for AUC behind CNNNetDilated. CNNNet performs the worst for $\text{precision}_k$ in this case, however since $\text{precision}_k$ is much more similar across the board for DeepHit it is still the best performing architecture overall.

Table 5: DeepHit & Logistic Hazard Performance Metrics across architectures, mean and standard deviation for five folds. Bold highlights the best result across model configurations.

| Architecture | C-Index | IBS | Mean AUC | $\text{Precision}_{20}(5)$ |
|---|---|---|---|---|
| | | *DeepHit* | | |
| LSTMNet | 0.6991 ± 0.0350 | 0.0319 ± 0.0029 | 0.6420 ± 0.0484 | 0.5200 ± 0.1605 |
| CNNNetDilated | 0.7069 ± 0.0337 | 0.0329 ± 0.0032 | **0.6708 ± 0.0329** | 0.5000 ± 0.0935 |
| CNNNet | **0.7170 ± 0.0234** | **0.0316 ± 0.0031** | 0.6579 ± 0.0360 | 0.4900 ± 0.0652 |
| CNNLSTMNet | 0.7060 ± 0.0188 | 0.0319 ± 0.0027 | 0.6410 ± 0.0240 | **0.5200 ± 0.0758** |
| | | *Logistic Hazard* | | |
| LSTMNet | 0.6745 ± 0.0519 | 0.0315 ± 0.0032 | 0.6449 ± 0.0463 | 0.5200 ± 0.1304 |
| CNNNetDilated | 0.6665 ± 0.0726 | 0.0317 ± 0.0040 | **0.6696 ± 0.0580** | **0.5900 ± 0.0894** |
| CNNNet | 0.6925 ± 0.0298 | 0.0317 ± 0.0030 | 0.6349 ± 0.0382 | 0.4900 ± 0.1140 |
| CNNLSTMNet | **0.7129 ± 0.0393** | **0.0307 ± 0.0028** | 0.6646 ± 0.0388 | 0.5500 ± 0.0500 |

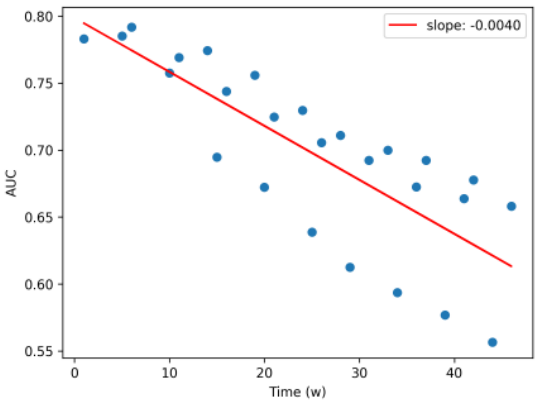
### 6.1.5 Model Comparison

When comparing the best result for each model, it is clear that all models showcase good performance, especially when comparing to the Kaplan-Meier baseline, see Table 6. Random Survival Forest stands out as the best performing model, with the best scores except for $\text{precision}_k$.

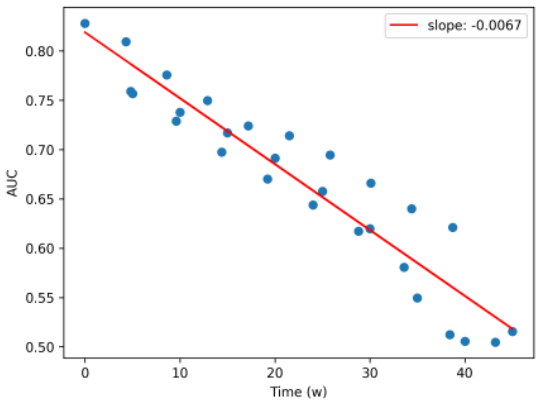
Table 6: The performance metrics for the best versions of each model, mean and standard deviation for five folds. Bold highlights the best result across model types.

| Model | C-Index | IBS | Mean AUC | $\text{Precision}_{20}(5)$ |
|---|---|---|---|---|
| Kaplan-Meier | 0.5000 ± 0.0000 | 0.0423 ± 0.0000 | 0.5000 ± 0.0000 | 0.0600 ± 0.0000 |
| RSF | **0.7241 ± 0.0486** | **0.0246 ± 0.0027** | **0.7040 ± 0.0553** | 0.2900 ± 0.0742 |
| Weibull AFT | 0.7046 ± 0.0421 | 0.0260 ± 0.0019 | 0.7013 ± 0.0499 | 0.2900 ± 0.1636 |
| DeepHit | 0.7170 ± 0.0234 | 0.0316 ± 0.0031 | 0.6579 ± 0.0360 | 0.4900 ± 0.0652 |
| Logistic Hazard | 0.7129 ± 0.0393 | 0.0307 ± 0.0028 | 0.6646 ± 0.0388 | **0.5500 ± 0.0500** |

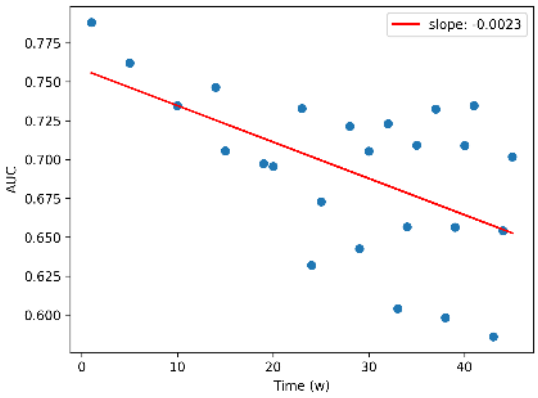
The neural network based models get outperformed significantly when looking at integrated brier score and AUC, but showcase a much better $\text{precision}_k$ score. This observation points to those models having a better risk prediction in the short term but falling behind in long term predictions. This trend is further observed in the AUC scores, see Figure 4, where the neural network models have a better score for the earliest time points but degrade more heavily for later time points.



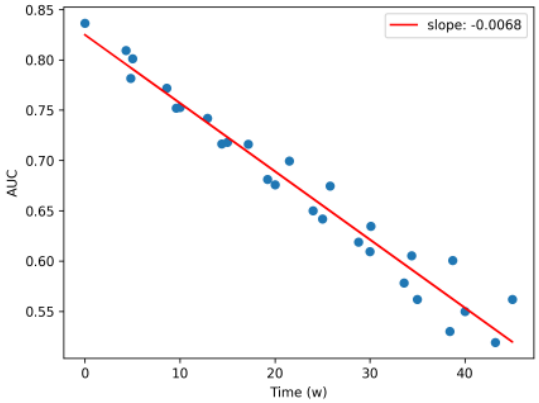

(a) Random Survival Forest, max_depth = 10




(b) DeepHit CNNNet




(c) Weibull AFT, penalizer = 0.01




(d) Logistic Hazard CNNLSTMNet

Figure 4: Time-dependent AUC scores for champion survival models across five-fold cross-validation. Lines show linear trends with slopes in legends.

## 6.2 Model Interpretability

In this Section, the models interpretability are presented. Weibull AFT provides intrinsic interpretability from it's regression weights, whereas the rest of the models require the use of explanation methods presented in Section 5.6.

### 6.2.1 Weibull AFT Regression Weights

The Weibull AFT is the only parametric regression model used in this study and its coefficient vector $\lambda$ can be extracted directly. This vector includes the intercept term $\beta_0$ as well as the coefficients for each input feature, see Figure 5 for the largest coefficients. See Table 12 in Appendix F for all coefficients.

The intercept for the scale parameter is $\beta_0 = 6.5076$. This intercept is substantially larger than the feature coefficients, indicating that the baseline failure time is the strongest predictor. The baseline failure time dominating the other features means that the other features slightly shift the failure time backwards or forwards in time, but the failure time is

relatively stable. The model also calculates the shape parameter, $\rho$. The shape parameter is $\exp(0.1157) > 1$, meaning that risk of failure increases over time.

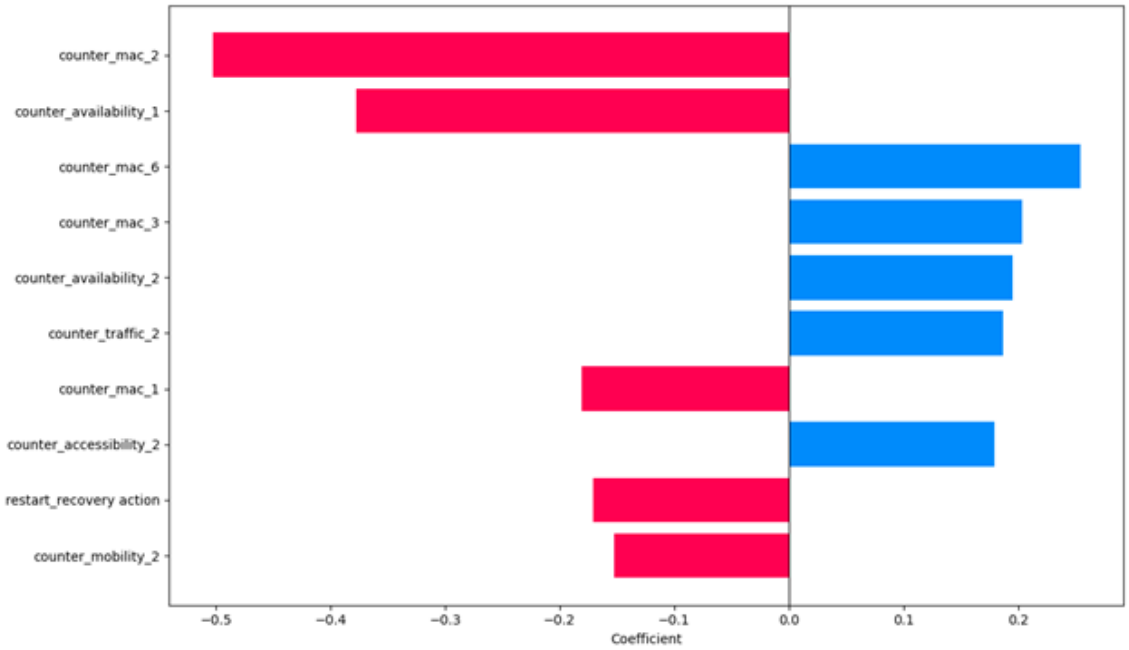



Figure 5: The top ten largest coefficients for Weibull AFT. A negative coefficients indicates that a larger value decreases survival time, and vice versa for a positive coefficient.

### 6.2.2 SHAP KernelExplainer

In this rubric, the SHAP analysis is presented, based on the best performing configuration of each model type from section 6.1.5. SHAP importance was calculated based on the cumulative risk at the median event time, thus a higher SHAP value is associated with a higher replacement risk.

Overall, Weibull AFT produces the most interpretable SHAP results, see Figure 6, which is to be expected given it has the least complex input-output relationship. The neural network models, see Figures 8 and 9, produce SHAP values far smaller than the other models and the values are also more evenly distributed across the top. This is contributed to the fact that each feature input can contribute to a higher and lower risk score, such that the sum over the 96 time steps becomes smaller than when simply evaluating one feature value. The SHAP values of Random Survival Forest are much larger than the neural networks, but still very small compared to Weibull AFT, see Figure 7. All calculated SHAP values are illustrated in Table 10 in Appendix D.

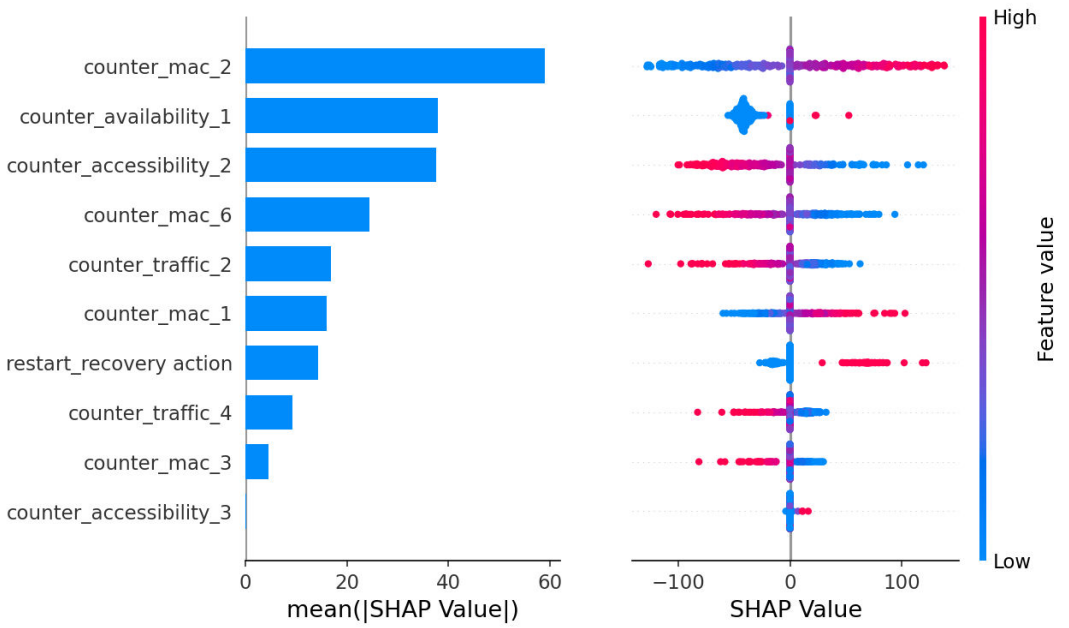



Figure 6: SHAP bar plot (left) and beeswarm plot (right) for Accelerated Failure Time top 10 features with outliers removed.

The Accelerated Failure Time model shows structured SHAP value patterns. Several features display clear relationships between feature value and time to event, although exceptions occur. The second highest ranked feature, `counter_availability_1`, is ambiguous. It has smaller values that increases the time to event but also some that has no affect on the risk. In contrast, larger values are divided across decreasing, increasing or not affecting at all. The model ranks `restart_recovery` action higher than alarms among its most important features. Higher values of this feature are associated with decreased time to event, while lower values are associated with either increased or unchanged time to event.

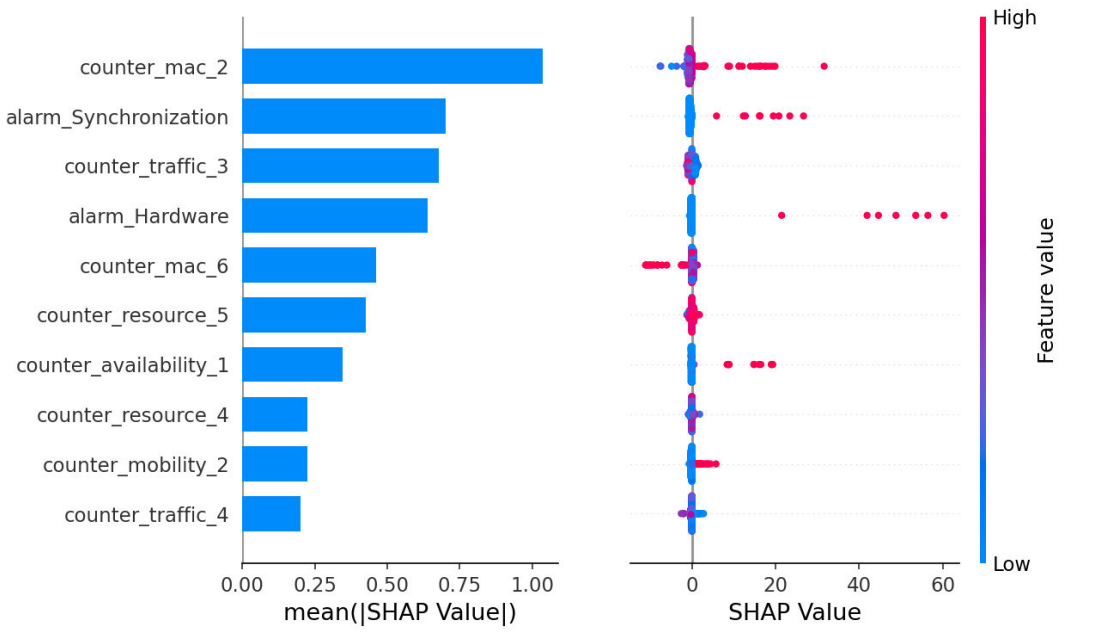



Figure 7: SHAP bar plot (left) and beeswarm plot (right) for RSFs top 10 features.

The Random Survival Forest demonstrates both structured and mixed SHAP values patterns across the features. The highest ranked feature, `counter_mac_2`, shows that larger feature values are associated with decreased time to event. In the low to mid range, the relationship is less consistent. Lower features values are mainly associated with an increased time to event, though there are some exceptions where lower values have negligible impact on the prediction. Mid range values does not have an impact on the time to event. The features `alarm_Synchronization` and `alarm_Hardware` are among the most important features in the model. The presence of alarm features is associated with decreased time to event. The third most important feature is `counter_traffic_3`. However, the beeswarm plot for this feature does not show a consistent directional relationship with time to event.

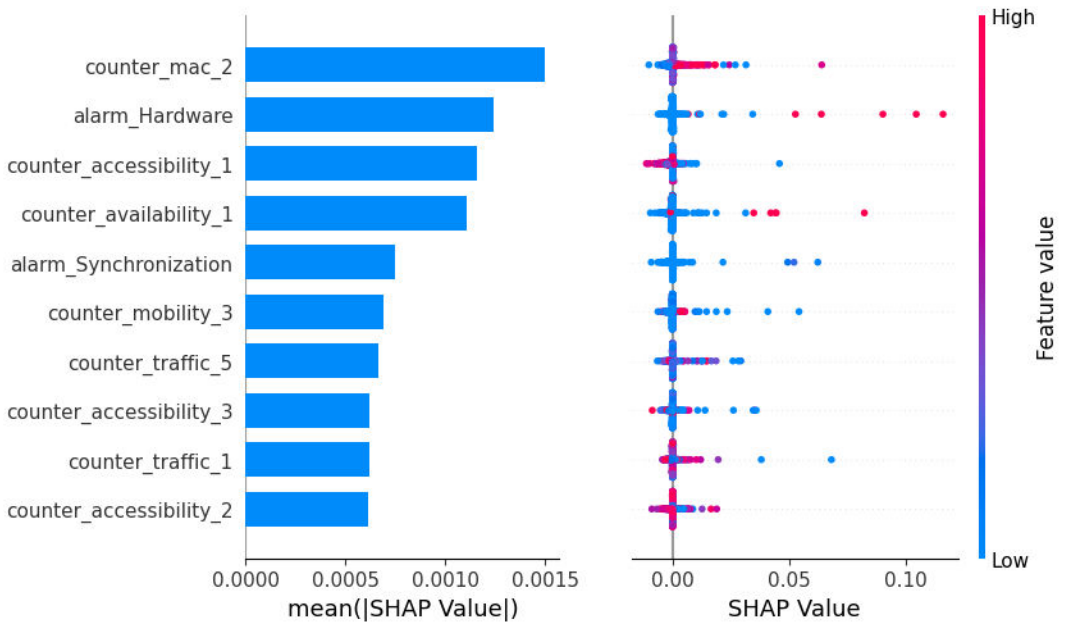



Figure 8: SHAP bar plot (left) and beeswarm plot (right) for DeepHit top 10 features.

DeepHit showcases less structured SHAP values patterns across features. The highest ranked feature is `counter_mac_2`. However, the beeswarm plot remains difficult to interpret. It does not clearly indicate whether a large or small feature value is associated with an increase or decrease in the time to event. While most large values decrease the time to event some small values decrease the time to event even more. For `alarm_Hardware`, larger feature values are generally associated with decreased time to event. However, some smaller feature values are also associated with decreased time to event.

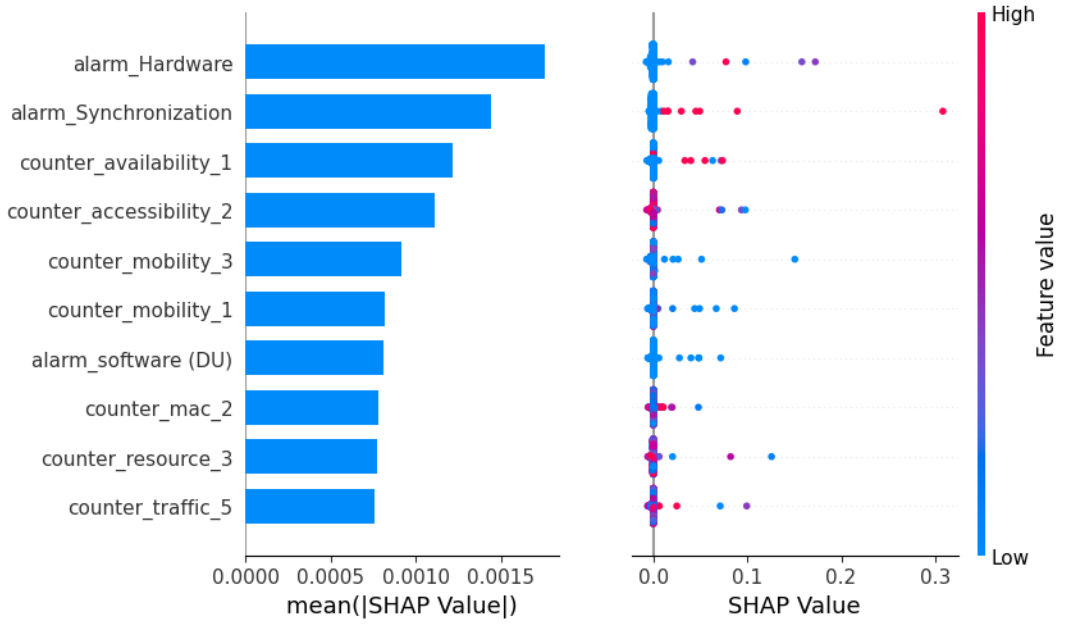



Figure 9: SHAP bar plot (left) and beeswarm plot (right) for Logistic Hazard top 10 features.

Logistic Hazard displays few structured and mostly mixed patterns for SHAP values across the features. The two highest ranked features are `alarm_Hardware` and `alarm_Synchronization`. Larger values of `alarm_Hardware` are loosely associated with a decreased time to event. The mid range of values seems to be strongly associated with a decrease in time to event, while low values are mostly associated with no effect on the time to event. Larger values of `alarm_Synchronization` is associated with a decrease time to event, whereas lower values are associated with no impact on the time to event.

To aggregate the produced SHAP importance scores a ranking system is implemented, see Table 7. The ranking system utilizes the average positional ranking of features across all model types. The position is based on the mean absolute SHAP value, which is shown in the SHAP bar plots. This provides a clearer comparison of feature importance across the different models.

Table 7: Top 10 features ranked by mean absolute SHAP importance, with individual model rankings and the mean rank across models.

| Feature | RSF | Weibull AFT | Logistic Hazard | DeepHit | Mean |
|---|---|---|---|---|---|
| counter_mac_2 | 1.00 | 1.00 | 8.00 | 1.00 | 2.75 |
| alarm_Synchronization | 2.00 | 12.00 | 2.00 | 5.00 | 5.25 |
| counter_availability_1 | 7.00 | 9.00 | 3.00 | 4.00 | 5.75 |
| alarm_Hardware | 4.00 | 31.00 | 1.00 | 2.00 | 9.50 |
| counter_mobility_3 | 11.00 | 17.00 | 5.00 | 6.00 | 9.75 |
| counter_accessibility_2 | 18.00 | 8.00 | 4.00 | 10.00 | 10.00 |
| counter_mac_6 | 5.00 | 3.00 | 24.00 | 18.00 | 12.50 |
| counter_traffic_5 | 15.00 | 23.00 | 10.00 | 7.00 | 13.75 |
| counter_traffic_2 | 17.00 | 6.00 | 22.00 | 12.00 | 14.25 |
| counter_traffic_3 | 3.00 | 22.00 | 18.00 | 16.00 | 14.75 |

### 6.2.3 LOFO Importance

The calculated LOFO importance varies a lot more depending on the model, see Figure 10 and Table 11 in Appendix E. Weibull AFT produces mostly negative LOFO importance scores for all features whereas Logistic Hazard produces mostly positive importance scores, and the magnitude of importance differs across models as well. DeepHit produces the smallest importance scores overall and the lowest maximum importance, with its most important feature only decreasing model performance by 5% if excluded.

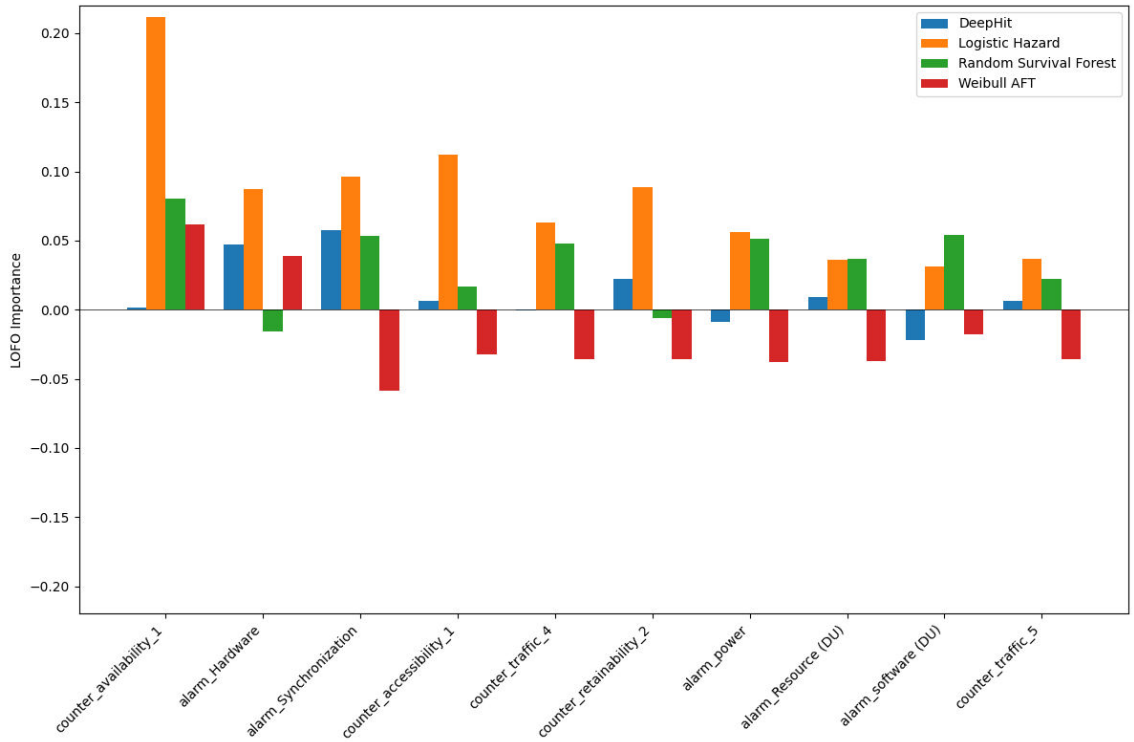



Figure 10: LOFO importance for the 10 features with the highest positive combined score across champion models, split on model type. Negative importance indicates that the feature hurts model performance, whereas a positive importance increases model performance.

# 7 Discussion

In this section, the results will be discussed from the two perspectives of the thesis, model performance and model interpretability.

## 7.1 Model Performance

Overall, the evaluated survival analysis models perform well above the Kaplan-Meier baseline level in predicting customer-driven replacements, indicating that the underlying replacement behavior can sufficiently be captured by the chosen models and descriptive features. This suggests that the problem is well-suited for survival modeling, and that meaningful signal exists for both replacement probability estimation and unit-level risk ranking. Given this strong overall performance, the main differences between models are not in whether they can learn the task, but in how they distribute predictive performance across time horizons and calibration versus discrimination objectives.

For the integrated Brier score, which evaluates calibration performance, the neural network models are outperformed by both Random Survival Forest and Weibull AFT. This arises from the loss in performance for long term predictions, where the neural network

models performances decrease more heavily. The poorest performance is observed for DeepHit, which can be expected since it is the only model that explicitly optimizes a ranking objective in its loss function which hinders it's calibration performance. Weibull AFT relies on a parametric survival distribution that is adjusted according to the co-variates, allowing it to generate smooth survival trajectories that more closely resemble the observed outcomes. Random Survival Forest similarly benefits from the averaging effect introduced by bagging across many trees, producing more conservative and stable predictions.

Despite these differences in calibration, the concordance index remains relatively similar across the models. This is likely because the metric is computed from comparable pairs at observed event times, and most events occur during the first half of the prediction horizon. The concordance index thus places greater emphasis on short-term discrimination, whereas the time-dependent AUC more clearly reveals the deterioration in long-term ranking performance for the neural network models. This result validates the need to utilize both discrimination metrics, as they together paint a more informative picture.

The discrete-time neural approaches achieve strong short-term discrimination by estimating survival through a sequence of conditional hazards or event probabilities across time intervals. These models have a further advantage since they can utilize the full timeseries data of each day, compared to the other models which only utilizes the static mean. This enables the models to capture localized and highly non-linear feature effects associated with elevated early risk, which also contributes to the higher $\text{precision}_k$ observed for these models. However, the same flexibility may reduce robustness over longer horizons due to error accumulation across time bins and increased sparsity of late-event observations. In contrast, the stronger structural assumptions of Weibull AFT and the ensemble averaging of Random Survival Forest appear to produce smoother and more stable long-term survival estimates. However, this also causes the aforementioned models to produce more similar short-term risk scores which degrades the $\text{precision}_k$ performance.

The observed differences across metrics indicate that model performance is highly dependent on the evaluation perspective, where the horizon-dependent and top-k metrics reveal substantial divergence in predictive behavior. Derived from the result is also the variance of the scores, i.e. the dependence on dataset for training and evaluation. The neural models exhibit a generally lower variance across runs and metrics compared to the other models, suggesting improved robustness to sampling fluctuations in both training and evaluation splits. In contrast, Random Survival Forest and Weibull demonstrate moderate variability, particularly in ranking-sensitive measures like AUC and $\text{Precision}_k$, indicating a stronger dependence on the data split.

## 7.2 Model Interpretability

In this study, all champion models feature-risk relationships have been evaluated in three ways; regression model coefficients, SHAP feature importance and LOFO feature importance. This is motivated by the model performance, where all models showcase predictive performances significantly better than the Kaplan-Meier baseline. The SHAP-based ranking results, summarized in Table 7, indicate agreement between the models in terms of the most influential feature. Across the models, a small set of features consistently appear

among the highest ranked variables. This suggests that the models identify similar underlying signals associated with replacements, which makes the insights more reliable in terms of what drives the replacement decision. The strongest detected signal is from the feature `counter_mac_2`, which is consistently associated with increased replacement risk across model evaluations.

### 7.2.1 SHAP Analysis

The Accelerated Failure Time model appears to provide more structured SHAP patterns compared to the other models. This is consistent with its parametric structure, which produces more direct relationships between features and the predicted outcome. The parametric structure may limit utilization of discrete features, such as alarms and restarts, and how well it captures non-linear feature effects. In contrast to the other models, it does not rank alarms among its most important features, instead assigning higher importance to `restart_recovery action`. This coincides with the models weights, however the model ranks the `restart_recovery` action higher in terms of coefficient magnitude than SHAP, suggesting a difference between SHAP explanations and model parameters. There are several examples of the SHAP values showing different importance than the coefficient magnitudes, while some only appear in one of the two. This highlights that SHAP explanations and model coefficients capture different aspects of feature influence. SHAP model uses a sample of the data to estimate important features, corresponding to local explanations of the model whereas the coefficient uses all data and provides a global explanation.

Compared to the Accelerated Failure Time model, Random Survival Forest produce less interpretable feature-risk relationships. The SHAP values produced lack clear directionality for the features. Unlike the Accelerated Failure Time, it assigns greater importance to alarms. These alarms show more consistent directional relationships for the Random Survival Forest compared to the counters. Additionally, some features such as `counter_traffic_3` appear highly important according to SHAP ranking but lacks interpretable directional relationships.

The DeepHit neural network follows this pattern, producing increasingly less interpretable feature risk relationships compared to the two previous models. Numerous features highlight strong predictive performance but lack clear directionality. For instance, the highest ranked feature does not provide a consistent directional relationship with time to event. Compared to Weibull AFT, `alarm_Hardware` shows a more structured directional pattern. Both DeepHit and Random Survival Forest capture non-linear patterns, however this does not necessarily improve the interpretability of feature effects. Overall, the DeepHit model provides limited insight into how individual features influence replacements, despite identifying several important predictors.

The Logistic Hazard model has similar interpretability limitations as DeepHit and Random Survival Forest. Across most features, SHAP values vary substantially across feature ranges. The same feature values are associated with different effects on the model, which is not surprising given how the dense layers in the neural network combine features for output, creating significance for certain feature value combinations. However, `alarm_Hardware` and `alarm_Synchronization` demonstrate more stable direc-

tional relationships. The overall inconsistency of the SHAP patterns limits the interpretability and provides limited insight into how features influence replacement risk.

### 7.2.2 LOFO Analysis

The LOFO importance scores differ substantially more between the models than the SHAP values. This discrepancy is likely caused by the fundamentally different ways the models utilize the input features. Whereas SHAP evaluates the marginal contribution of a feature within the fitted model, LOFO importance measures the change in predictive performance when a feature is removed entirely. As a result, LOFO becomes more sensitive to feature redundancy, feature interactions, and model-specific dependence on certain inputs.

One contributing factor is the differing structure and assumptions of the models. Weibull AFT is a parametric regression-based survival model which generally performs better with lower-dimensional and more structured inputs. This likely contributes to the predominantly negative LOFO scores observed for the model, as removing some features may reduce noise or multicollinearity and thereby improve generalization performance. The only positive importance features for Weibull AFT is the large coefficient features `counter_mac_2` and the `alarm_Hardware`.

Furthermore, previous analyses shows that Weibull AFT has difficulties utilizing the discrete alarm and restart features effectively compared to the other models. Consequently, the negative LOFO scores for these variables may not necessarily indicate that the features lack predictive value, but rather that the model itself is unable to utilize the information efficiently. For this reason, Weibull AFT may be less suitable when evaluating the importance of alarms and restarts. Excluding Weibull AFT from the comparison produces a more coherent result for these features, where all alarm and restart variables obtain net positive LOFO importance scores across the remaining models.

The architectural differences between the models further explain the variation in feature importance. Random Survival Forest constructs decision trees based on recursive feature splits, making it highly sensitive to threshold-based relationships and feature interactions. DeepHit and Logistic Hazard utilizes neural-network-based feature extraction through convolutional and LSTM layers to capture both local and temporal dependencies. Because these models process information differently, it is not surprising that they rely on different subsets of features for prediction performance.

The only feature with consistently positive LOFO importance across all four models is `counter_availability_1`, indicating that it provides robust and model-independent predictive information. No other feature shows complete agreement across all models, either positively or negatively. This suggests that most variables contribute in a model-dependent manner, where their usefulness depends strongly on the assumptions and learning mechanisms of the specific model.

It is important to remember the nature of LOFO, where the score is based on that all other features are kept. While this study calculated pair-wise correlation of features and removed redundant features, it is likely that a subset of features may contain the same informational patterns as other standalone features. This could further help explain why

many of the calculated LOFO importance scores are negative, as the same informational patterns exist across combinations of other features. A feature can contribute strongly to predictions when present, yet its removal may have limited impact if other correlated or redundant variables provide similar information. Thus removing a feature can sometimes improve performance by reducing noise or overfitting, resulting in negative LOFO importance scores despite the feature having noticeable SHAP contributions.

## 7.3 Limitations

For this study, there were hardware limitations, making it impossible to train models on all available data. This led to an undersampling of the negative class, not replaced units, shifting the class ratio in the data away from the real world scenario. Furthermore, there was a sampling across time, where a chosen weekday represented a full week for counters. While this sampling was done through data exploration it does not capture all possible variations across the range of the year. These limitations has most likely had an impact on model performance and the insights gained from feature importance evaluations. However, the aim of this thesis is to assess the viability of the method and the possible insights gained from feature importance evaluations and the findings contribute meaningfully to this objective.

# 8 Conclusions

The aim of this study was to evaluate survival analysis models for predicting customer-driven replacements of in-field radio units, in respect to their predictive performance and interpretability of predictions. This was done utilizing qualitatively selected descriptive features, capturing high-level unit performance, in a dynamic format where each observation is treated as a separate input per unit. The selected models reflect different types of modeling approaches, where the result provides some insight directly based on the models underlying assumptions. Furthermore, the interpretability of the models feature-output relationship has been evaluated in the study to examine how informative they are as a method of understanding the underlying decision-making behind replacements.

Based on the result, see Section 6.1.5, all models perform substantially better than the calculated Kaplan-Meier baseline. The study can thus conclude that the chosen descriptive features reflect the unit behavior which are used as grounds for a replacement decision. Furthermore, the application of Survival Analysis models produces results that reinforces that their use is motivated in this setting.

When comparing the models, it is clear that Random Survival Forest is the best performing model overall, based on the chosen evaluation metrics of the study. While DeepHit and Logistic Hazard are much better at ranking the top risk units for short term prediction horizons, they get significantly outperformed by the other two models for predictions at longer time horizons. Based on this, one can conclude that the daily patterns of features captured by the neural networks, combined with their discrete time hazard output, are much better at utilizing signals that appear close to a replacement. However, the smoother

outputs generated by Random Survival Forest and Weibull AFT can generally represent the expected survival probabilities better over time, resulting in a higher calibration score.

Through evaluating the models interpretability, both intrinsically and through explanation methods, this study investigated how consistent and informative the derived feature risk relationships are for explaining customer-driven replacements. The results show consistency for the top ranked features across the models. Several counter-based features consistently appear at the top, with `counter_mac_2` emerging as the strongest overall predictor. This suggest that the models capture similar underlying predictive signals.

However, the models are considerably less consistent in how the features influence the predictions. The Weibull AFT model produces more structured feature-risk relationships, due to its parametric nature. Meanwhile, the Random Survival Forest, DeepHit and Logistic Hazard models generally capture more complex and less stable relationships. These flexible black-box models capture discrete and non-linear patterns better but the flexibility comes at the cost of directional interpretability.

Furthermore, the evaluation results demonstrate that the interpretability depends strongly on the explanation method used. The methods used in this study were SHAP and LOFO, which frequently produce different conclusions regarding feature importance. The SHAP explanations indicate consistency across the models in regards to feature inference importance, whereas LOFO showcase substantial differences in how dependent the models are on individual features for predictive performance. This discrepancy highlights how the methods are fundamentally different, with SHAP capturing feature contribution and LOFO capturing feature dependence. Features may contribute strongly to individual predictions while still being replaceable due to redundancy, as similar informational patterns are also captured by other features.

Overall, the feature-risk relationships can therefore be considered moderately consistent but only partially informative. The models generally agree on which features contain predictive information, but they differ substantially in how these features are utilized internally and how clearly their effects can be interpreted. The findings suggest that simpler parametric models provide more interpretable relationships, while more flexible machine learning models capture complex patterns, such as those from discrete features, at the expense of explanatory clarity. Consequently, no single interpretability method or model is sufficient on its own for fully explaining customer-driven replacements. Instead, combining multiple models and explanation approaches provides a more comprehensive understanding of both predictive behavior and feature relevance, despite differences in predictive performance.

## 8.1 Future Work

KernelSHAP explanations are computationally demanding due to the sampling-based approximation procedure. While increasing the number of sampled observations may improve stability of feature importance estimates, the marginal benefit is expected to diminish beyond moderate sample sizes. Future work could investigate whether larger SHAP evaluation samples lead to greater consistency in feature rankings across models, especially when comparing to regression coefficients. Time-dependent explanation methods

such as SurvSHAP [39] may further provide insight into how feature contributions evolve across the prediction horizon. However, these methods were not applied in this study because of two main reasons. First, they are substantially more computationally intensive than standard KernelSHAP. Second because the Weibull AFT model imposes a structurally time-invariant co-variate effect, limiting the expected benefit from comparing it to the other flexible models utilizing SurvSHAP.

As for the hardware limitations set for this thesis, conducting further studies on more data would give further insights. This would enable adhering to the real-world class ratio and remove the dependency on weekday samples. It could also enable the study of longer time effects, since most units are operating in field for multiple years, whereas this thesis had to limit the scope to one year of observations.

# References


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

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

[3] H. Ishwaran, U. Kogalur, B., E. Blackstone, H, and M. Lauer, S, "Random survival forests," *The Annals of Applied Statistics*, vol. 2, no. 3, pp. 841–860, 2008.

[4] C. Lee, W. Zame, J. Yoon, and M. Schaar, "Deephit: A deep learning approach to survival analysis with competing risks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, pp. 2314 – 2321, 2018.

[5] Ericsson. (2026) Shaping history. fetched 2026-02-16. [Online]. Available: https://www.ericsson.com/en/about-us/history/shaping-history

[6] Ericsson. (2026) Corporate story. fetched 2026-02-16. [Online]. Available: https://www.ericsson.com/en/about-us/history/shaping-history/corporate-story

[7] Ericsson. (2026) New world of possibilities. fetched 2026-02-16. [Online]. Available: https://www.ericsson.com/en/about-us/new-world-of-possibilities

[8] Ericsson. (2026) 5G RAN. fetched 2026-02-16. [Online]. Available: https://www.ericsson.com/en/ran

[9] E. Dahlman, S. Parkvall, and J. Skold, *5G NR : the next generation wireless access technology*, 1st ed. London: Academic Press, 2018.

[10] M. Girnyk, H. Jidhage, and S. Faxér, "Broad beamforming technology in 5g massive mimo," *Ericsson Technology Review*, vol. 10, October 2023.

[11] M. Mills, *Introducing Survival and Event History Analysis*. SAGE Publications Ltd, 2012.

[12] E. L. Kaplan, P. Meier, N. L. Johnson, and S. Kotz, "Nonparametric estimation from incomplete observations," in *Breakthroughs in Statistics*, ser. Springer Series in Statistics. United States: Springer New York, 1991, pp. 319–337.

[13] F. de Cos Juez, P. García Nieto, J. Martínez Torres, and J. Taboada Castro, "Analysis of lead times of metallic components in the aerospace industry through a supported vector machine model," *Mathematical and Computer Modelling*, vol. 52, no. 7, pp. 1177–1184, 2010.

[14] S. R. Gross, B. O'Brien, C. Hu, and E. H. Kennedy, "Rate of false conviction of criminal defendants who are sentenced to death," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 111, no. 20, pp. 7230–7235, 2014.

[15] H. F. Martz, *Reliability Theory*, third edition ed., R. A. Meyers, Ed. New York: Academic Press, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B0122274105006591

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[17] L. Breiman and A. Cutler. (2004, March) Random forests. fetched 2026-05-24. [Online]. Available: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#intro

[18] H. Wang and G. Li, "A selective review on random survival forests for high dimensional data," *Quant Biosci*, vol. 36, no. 2, pp. 85–96, 2017.

[19] S. Pölsterl, "scikit-survival: A library for time-to-event analysis built on top of scikit-learn," *Journal of Machine Learning Research*, vol. 21, no. 212, pp. 1–6, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-729.html

[20] M. Newby, "Accelerated failure time models for reliability data analysis," *Reliability Engineering & System Safety*, vol. 20, no. 3, pp. 187–197, 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0951832088901147

[21] J. D. Kalbfleisch and R. L. Prentice, *The Statistical Analysis of Failure Time Data*, 2nd ed. Hoboken, NJ: John Wiley Sons, 2002.

[22] C. Davidson-Pilon, "lifelines: survival analysis in python," *Journal of Open Source Software*, vol. 4, no. 40, p. 1317, 2019. [Online]. Available: https://doi.org/10.21105/joss.01317

[23] H. Kvamme, Ørnulf Borgan, and I. Scheel, "Time-to-event prediction with neural networks and cox regression," *Journal of Machine Learning Research*, vol. 20, no. 129, pp. 1–30, 2019. [Online]. Available: http://jmlr.org/papers/v20/18-424.html

[24] M. Gensheimer and B. Narasimhan, "A scalable discrete-time survival model for neural networks," *PeerJ (San Francisco, CA)*, vol. 7, pp. e6257–, 2019.

[25] H. Kvamme. (2026) pycox. fetched 2026-04-24. [Online]. Available: https://github.com/havakv/pycox/tree/master

[26] F. E. Harrell, R. M. Califf, D. B. Pryor, K. L. Lee, and R. A. Rosati, "Evaluating the yield of medical tests." *JAMA: The Journal of the American Medical Association*, vol. 247, no. 18, p. 2543–2546, November 1982.

[27] L. Antolini, P. Boracchi, and E. Biganzoli, "A time-dependent discrimination index for survival data," *Statistics in Medicine*, vol. 24, no. 24, pp. 3927–3944, 2005.

[28] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[29] scikit survival. (2026) sksurv.metrics.cumulative_dynamic_auc. fetched 2026-02-17. [Online]. Available: https://scikit-survival.readthedocs.io/en/stable/api/generated/sksurv.metrics.cumulative_dynamic_auc.html#id1

[30] J. Robins, A. Rotnitzky, and B.-P. Zhao, "Analysis of semiparametric regression models for repeated outcomes in the presence of missing data," *Journal of the American Statistical Association*, vol. 90, no. 429, pp. 106–121, 1995.

[31] W. Brier, G, "Verification of forecasts expressed in terms of probability," *Monthly Weather Review*, vol. 78, no. 1, pp. 1 – 3, 1950.

[32] E. Graf, C. Schmoor, W. Sauerbrei, and M. Schumacher, "Assessment and comparison of prognostic classification schemes for survival data," *Statistics in Medicine*, vol. 18, no. 17-18, pp. 2529 – 2545, 1999.

[33] Scikit-Learn. (2026) Precision-recall. fetched 2026-05-29. [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[34] S. Lundberg. (2026) SHAP Documentation. fetched 2026-04-22. [Online]. Available: https://shap.readthedocs.io/en/latest/index.html

[35] L. Shapley, "A value for n-person games," *Contribution to the Theory of Games*, vol. 2, pp. 307–317, 1953.

[36] E. Strumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *Journal of Machine Learning Research*, vol. 11, pp. 1–18, 2010.

[37] S. Lundberg and S. Lee, "A unified approach to interpreting model predictions," *CoRR*, vol. abs/1705.07874, 2017.

[38] C. Molnar, *Interpretable Machine Learning*, 3rd ed., 2025. [Online]. Available: https://christophm.github.io/interpretable-ml-book

[39] M. Krzyziński, M. Spytek, H. Baniecki, and P. Biecek, "Survshap(t): Time-dependent explanations of machine learning survival models," *Knowledge-Based Systems*, vol. 262, p. 110234, 2023.

# A List of Features and Metadata

- serialNumber (str)
- timestamp (datetime)
- Replacement Date (datetime)
- event (boolean)
- duration (int)
- Counters (float)
  - Traffic
  - Resource
  - Accessibility
  - Mobility
  - Retainability
  - MAC
  - Availability
- Alarms (float)
  - Alarm Hardware
  - Alarm Antenna Calibration
  - Alarm Synchronization
  - Alarm Temperature
  - Alarm Software (DU)
  - Alarm Resource (DU)
  - Alarm Other (DU)
  - Alarm Power Saving
  - Alarm Power
- Restarts (float)
  - Restart Recovery Action
  - Restart Lost Contact
  - Restart Software Problem

# B RSF Hyperparameters

Table 8: RSF Model parameters

| max_depth | n_estimators | min_samples_leaf | max_features |
|---|---|---|---|
| 3 | 200 | 5 | log2 |
| 5 | 300 | 5 | log2 |
| 10 | 100 | 20 | 0.5 |
| none | 200 | 20 | log2 |

# C Neural Network Hyperparameters

Table 9: DeepHit and Logistic Hazard hyperparameters based on five-fold cross-validation tuned for concordance index

| Architecture | Learning Rate | Batch Size |
|---|---|---|
| *DeepHit* | | |
| LSTMNet | 0.002 | 128 |
| CNNNetDilated | 0.0005 | 256 |
| CNNNet | 0.0001 | 128 |
| CNNLSTMNet | 0.002 | 256 |
| *Logistic Hazard* | | |
| LSTMNet | 0.0001 | 128 |
| CNNNetDilated | 0.0001 | 256 |
| CNNNet | 0.001 | 128 |
| CNNLSTMNet | 0.0005 | 128 |

# D SHAP Importance

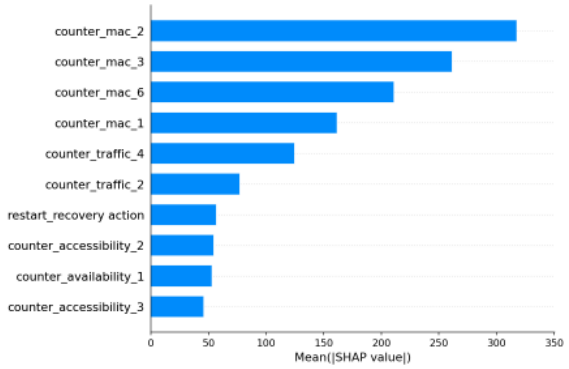



Figure 11: The Weibull AFT SHAP bar plot with outliers.

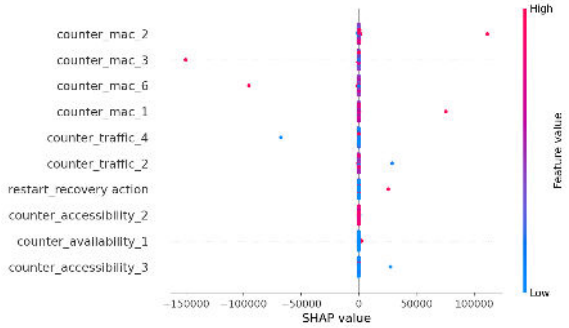



Figure 12: The Weibull AFT SHAP beeswarm plot with outliers

Table 10: SHAP Absolute Feature Importance

| Feature | Random Survival Forest | Weibull AFT | Logistic Hazard | DeepHit |
|---|---|---|---|---|
| alarm_Antenna Calibration | 0.0000 | - | 0.0000 | 0.0000 |
| alarm_Hardware | 0.6399 | 0.1598 | 0.0018 | 0.0012 |
| alarm_Other (DU) | 0.0031 | 0.0000 | 0.0000 | 0.0000 |
| alarm_Resource (DU) | 0.0098 | 32.8086 | 0.0000 | 0.0005 |
| alarm_Synchronization | 0.7025 | 41.3973 | 0.0014 | 0.0007 |
| alarm_Temperature | 0.0033 | - | 0.0007 | 0.0006 |
| alarm_power | 0.0034 | 0.0000 | 0.0005 | 0.0004 |
| alarm_power_saving | 0.0000 | - | 0.0000 | 0.0000 |
| alarm_software (DU) | 0.0009 | 0.2038 | 0.0008 | 0.0001 |
| counter_accessibility_1 | 0.0057 | 41.8848 | 0.0007 | 0.0012 |
| counter_accessibility_2 | 0.0383 | 54.6945 | 0.0011 | 0.0006 |
| counter_accessibility_3 | 0.0125 | 46.1464 | 0.0005 | 0.0006 |
| counter_accessibility_4 | 0.0086 | 0.3560 | 0.0006 | 0.0005 |
| counter_availability_1 | 0.3451 | 53.0420 | 0.0012 | 0.0011 |
| counter_availability_2 | 0.0877 | 32.8235 | 0.0005 | 0.0005 |
| counter_mac_1 | 0.0344 | 161.7544 | 0.0006 | 0.0005 |
| counter_mac_2 | 1.0368 | 317.8650 | 0.0008 | 0.0015 |
| counter_mac_3 | 0.0238 | 261.4863 | 0.0006 | 0.0005 |
| counter_mac_4 | 0.0287 | 0.0855 | 0.0008 | 0.0005 |
| counter_mac_5 | 0.1246 | 0.2889 | 0.0006 | 0.0006 |
| counter_mac_6 | 0.4599 | 211.0891 | 0.0006 | 0.0006 |
| counter_mobility_1 | 0.0087 | 8.1697 | 0.0008 | 0.0006 |
| counter_mobility_2 | 0.2238 | 9.7491 | 0.0006 | 0.0004 |
| counter_mobility_3 | 0.1371 | 9.8151 | 0.0009 | 0.0007 |
| counter_resource_1 | 0.0503 | 0.8247 | 0.0004 | 0.0005 |
| counter_resource_2 | 0.0164 | 0.3693 | 0.0004 | 0.0005 |
| counter_resource_3 | 0.0213 | 27.1094 | 0.0008 | 0.0005 |
| counter_resource_4 | 0.2243 | 1.2792 | 0.0005 | 0.0005 |
| counter_resource_5 | 0.4254 | 9.1625 | 0.0006 | 0.0005 |
| counter_resource_6 | 0.0869 | 16.0491 | 0.0006 | 0.0005 |
| counter_resource_7 | 0.0183 | 0.0276 | 0.0005 | 0.0006 |
| counter_retainability_1 | 0.0199 | 8.6775 | 0.0007 | 0.0005 |
| counter_retainability_2 | 0.0034 | 0.1585 | 0.0004 | 0.0006 |
| counter_retainability_3 | 0.0065 | 0.0755 | 0.0005 | 0.0005 |
| counter_traffic_1 | 0.0116 | 0.4288 | 0.0005 | 0.0006 |
| counter_traffic_2 | 0.0416 | 77.4460 | 0.0006 | 0.0006 |
| counter_traffic_3 | 0.6769 | 4.4378 | 0.0006 | 0.0006 |
| counter_traffic_4 | 0.2007 | 124.5698 | 0.0005 | 0.0005 |
| counter_traffic_5 | 0.0533 | 4.1368 | 0.0008 | 0.0007 |
| restart_lost contact | 0.0000 | - | 0.0008 | 0.0000 |
| restart_recovery action | 0.0064 | 56.9309 | 0.0007 | 0.0005 |
| restart_software problem | 0.0000 | - | 0.0004 | 0.0000 |

# E LOFO Importance

Table 11: LOFO Feature Importance

| Feature | Random Survival Forest | Weibull AFT | DeepHit | Logistic Hazard |
|---|---|---|---|---|
| alarm_Antenna Calibration | 0.0457 | -0.0364 | -0.0195 | 0.0205 |
| alarm_Hardware | -0.0156 | 0.0385 | 0.0475 | 0.0876 |
| alarm_Other (DU) | 0.0203 | -0.0376 | -0.0202 | 0.0280 |
| alarm_Resource (DU) | 0.0370 | -0.0369 | 0.0088 | 0.0363 |
| alarm_Synchronization | 0.0536 | -0.0587 | 0.0574 | 0.0965 |
| alarm_Temperature | 0.0034 | -0.0349 | -0.0225 | -0.0258 |
| alarm_power | 0.0515 | -0.0381 | -0.0088 | 0.0564 |
| alarm_power_saving | 0.0201 | -0.0355 | -0.0131 | 0.0438 |
| alarm_software (DU) | 0.0538 | -0.0181 | -0.0219 | 0.0312 |
| counter_accessibility_1 | 0.0168 | -0.0327 | 0.0066 | 0.1125 |
| counter_accessibility_2 | 0.0363 | -0.0400 | -0.0154 | 0.0461 |
| counter_accessibility_3 | 0.0273 | -0.0356 | -0.0266 | -0.0004 |
| counter_accessibility_4 | -0.0079 | -0.0356 | -0.0285 | 0.0270 |
| counter_availability_1 | 0.0804 | 0.0618 | 0.0014 | 0.2117 |
| counter_availability_2 | -0.0095 | -0.0024 | -0.0307 | 0.0285 |
| counter_mac_1 | -0.0055 | -0.0358 | -0.0185 | 0.0170 |
| counter_mac_2 | -0.0179 | 0.0367 | -0.0363 | 0.0470 |
| counter_mac_3 | 0.0384 | -0.0400 | -0.0557 | 0.0129 |
| counter_mac_4 | 0.0259 | -0.0495 | 0.0077 | 0.0304 |
| counter_mac_5 | 0.0165 | -0.0366 | -0.0207 | 0.0143 |
| counter_mac_6 | 0.0073 | -0.0260 | -0.0226 | 0.0359 |
| counter_mobility_1 | -0.0249 | -0.0410 | -0.0322 | 0.0091 |
| counter_mobility_2 | -0.0100 | -0.0607 | -0.0190 | 0.0667 |
| counter_mobility_3 | 0.0197 | -0.0608 | -0.0283 | 0.0718 |
| counter_resource_1 | 0.0041 | -0.0356 | 0.0004 | -0.0272 |
| counter_resource_2 | 0.0128 | -0.0365 | -0.0470 | 0.0622 |
| counter_resource_3 | 0.0068 | -0.0345 | -0.0230 | 0.0058 |
| counter_resource_4 | 0.0086 | -0.0357 | -0.0304 | 0.0812 |
| counter_resource_5 | 0.0403 | -0.0439 | 0.0159 | 0.0048 |
| counter_resource_6 | 0.0115 | -0.0561 | -0.0238 | 0.0060 |
| counter_resource_7 | 0.0137 | -0.0366 | -0.0283 | 0.0196 |
| counter_retainability_1 | 0.0088 | -0.0372 | -0.0265 | 0.0009 |
| counter_retainability_2 | -0.0061 | -0.0357 | 0.0223 | 0.0889 |
| counter_retainability_3 | -0.0056 | -0.0356 | -0.0016 | 0.0123 |
| counter_traffic_1 | -0.0270 | -0.0366 | -0.0372 | 0.0614 |
| counter_traffic_2 | 0.0102 | -0.0515 | -0.0343 | -0.0164 |
| counter_traffic_3 | 0.0490 | -0.0370 | -0.0383 | 0.0478 |
| counter_traffic_4 | 0.0477 | -0.0355 | -0.0005 | 0.0627 |
| counter_traffic_5 | 0.0225 | -0.0356 | 0.0062 | 0.0368 |
| restart_lost contact | 0.0203 | -0.0112 | -0.0027 | -0.0240 |
| restart_recovery action | 0.0232 | -0.0212 | -0.0221 | 0.0363 |
| restart_software problem | 0.0163 | -0.0623 | -0.0139 | 0.0542 |

# F Weibull AFT Coefficients

Table 12: Weibull AFT feature coefficients

| Feature | Coefficient |
|---|---|
| $\rho$ | 0.1157 |
| $\beta_0$ | 6.5076 |
| alarm_Hardware | -0.0690 |
| alarm_Other (DU) | -0.0308 |
| alarm_Resource (DU) | -0.0084 |
| alarm_Synchronization | -0.0056 |
| alarm_power | -0.0032 |
| alarm_software (DU) | -0.1050 |
| counter_accessibility_1 | 0.1246 |
| counter_accessibility_2 | 0.1798 |
| counter_accessibility_3 | -0.0287 |
| counter_accessibility_4 | -0.0279 |
| counter_availability_1 | -0.3771 |
| counter_availability_2 | 0.1956 |
| counter_mac_1 | -0.1803 |
| counter_mac_2 | -0.5026 |
| counter_mac_3 | 0.2038 |
| counter_mac_4 | -0.0079 |
| counter_mac_5 | -0.0119 |
| counter_mac_6 | 0.2542 |
| counter_mobility_1 | 0.1058 |
| counter_mobility_2 | -0.1523 |
| counter_mobility_3 | -0.1474 |
| counter_resource_1 | 0.0214 |
| counter_resource_2 | 0.0011 |
| counter_resource_3 | 0.0903 |
| counter_resource_4 | 0.0227 |
| counter_resource_5 | -0.0595 |
| counter_resource_6 | -0.0750 |
| counter_resource_7 | -0.0019 |
| counter_retainability_1 | -0.0879 |
| counter_retainability_2 | -0.0431 |
| counter_retainability_3 | -0.0205 |
| counter_traffic_1 | 0.0166 |
| counter_traffic_2 | 0.1874 |
| counter_traffic_3 | 0.0416 |
| counter_traffic_4 | 0.0812 |
| counter_traffic_5 | -0.0619 |
| restart_recovery action | -0.1704 |