Media Engineering and Technology Faculty
German University in Cairo

Learn2Clean Event Data

Bachelor Thesis

Author: Yousef Koka
Supervisors: Dr. David Selby
Prof. Sebastian Vollmer

Submission Date: 28 August, 2024

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowlegement has been made in the text to all other material used

---

Yousef Koka
28 August, 2024

Acknowledgments

I express my heartfelt gratitude to my supervisor, Dr. David Selby, for his expert mentorship, insightful feedback, and unwavering support throughout the challenges of this project. I am also grateful to Dr. Sergey Redyuk for his valuable suggestions and feedback, which significantly improved the quality of this work. I am thankful for DFKI and GUC for providing the resources and environment that made this research possible.

I extend my sincere thanks to Dr. Mohamed Gamal for his invaluable assistance in obtaining the necessary resources and for his patience and understanding. I also thank Dr. Mervat Abuelkheir for her guidance and support in connecting me with the right individuals for this project.

To my beloved parents, I owe a debt of gratitude beyond words—for their unwavering love, support, and encouragement, not only throughout my academic journey but for my entire life.

I am thankful to my friends for their support and encouragement and the original developers of Learn2Clean for their pioneering work.

Abstract

Data preprocessing plays a crucial role in the success of machine learning (ML) models, particularly in the context of survival analysis, where the goal is to predict time-to-event outcomes. Learn2Clean, a tool that utilizes Q-Learning to optimize data preprocessing pipelines, has shown promise in improving ML model performance. However, its applicability to survival analysis and its flexibility in handling different scenarios were limited. This thesis presents an extension of Learn2Clean to address these limitations.

The extended Learn2Clean framework incorporates three prominent survival analysis models: Cox Proportional Hazards, Random Survival Forest, and DeepHit Neural Network. It adapts the reward structure and action space of the Q-Learning algorithm to effectively optimize preprocessing pipelines for these models. Additionally, the framework enhances categorical data handling through ordinal encoding and introduces a configuration file for greater user customization. Dynamic reward matrices, defined using JSON files, further increase the tool's adaptability to diverse datasets and objectives.

To validate the effectiveness of the extended Learn2Clean tool, experiments were conducted on various datasets. The results demonstrate that the tool successfully identifies preprocessing pipelines that improve the performance of survival analysis models compared to baseline approaches. The flexibility offered by the configuration file and dynamic reward matrices allows users to tailor the tool's behavior to their specific needs.

This research contributes to the field of survival analysis by introducing an extended framework for automated data preprocessing. By adapting the Learn2Clean tool, this work addresses the specific challenges posed by missing data in this domain, aiming to improve the accuracy and robustness of survival models.

Keywords: Survival Analysis, Data Preprocessing, Q-Learning, Reinforcement Learning, Machine Learning, Cox Proportional Hazards, Random Survival Forest, DeepHit

Contents

Chapter 1

Introduction

1.1 Motivation and Objectives

In the era of big data and machine learning (ML), the ability to extract meaningful insights from complex datasets is paramount. A critical step in this process is data preprocessing, which involves cleaning, transforming, and preparing raw data to be suitable for analysis. The quality of data preprocessing significantly impacts the performance and reliability of ML models. This is particularly crucial in the field of survival analysis, where the goal is to predict the time until an event of interest occurs, such as patient death or customer churn.

Learn2Clean[1], a tool developed by Berti et al. (2020), offers an innovative approach to data preprocessing. It leverages Q-Learning, a reinforcement learning technique, to dynamically select the optimal sequence of preprocessing tasks for a given dataset and ML model. This optimization aims to maximize the quality of the ML model's results. However, the original Learn2Clean tool has limitations. It lacks built-in support for survival analysis, a vital area of ML with applications in various domains. Additionally, its handling of categorical data is limited, and it lacks flexibility in terms of model hyperparameter tuning and reward function customization.

This thesis addresses these limitations by extending Learn2Clean into Learn2Clean4Events or L2C4E for short, which will incorporate survival analysis capabilities. The primary objectives are:

- **Survival Analysis Integration:** Enable Learn2Clean to perform survival analysis using multiple models, including Cox Proportional Hazards, Random Survival Forest, and Neural Networks.
- **Improved Categorical Data Handling:** Enhance the tool's ability to handle categorical data through ordinal encoding.
- **Configuration File:** Introduce a configuration file to allow users to easily customize model hyperparameters and other settings.

- **Dynamic Reward Matrices:** Implement dynamic reward matrices using JSON files to provide greater flexibility in defining reward functions for Q-Learning.

1.2 Outline

The remainder of this thesis is organized as follows:

- Chapter 2 provides background information on data preprocessing, survival analysis, reinforcement learning, Q-Learning, and the Learn2Clean architecture.
- Chapter 3 details the methodology used to develop L2C4E, including the integration of survival analysis models, handling of categorical data, and the implementation of configuration files and dynamic reward matrices.
- Chapter 4 presents the results of experiments conducted to evaluate the effectiveness of L2C4E tool, along with a discussion of its limitations.
- Chapter 5 concludes the thesis with a summary of the findings and suggestions for future work.

Chapter 2

Background

2.1 Automated Data Preprocessing

The ever-increasing volume and complexity of data in today's world necessitate efficient and effective data preprocessing techniques. Data preprocessing, which involves cleaning, transforming, and organizing raw data into a suitable format for analysis, is a crucial step in the machine learning pipeline. The quality of the preprocessed data significantly impacts the performance and reliability of machine learning models. However, manual data preprocessing can be time-consuming, error-prone, and often requires domain expertise.

To address these challenges, automated data preprocessing techniques have emerged as a promising solution. These techniques leverage algorithms and heuristics to automate various data cleaning and transformation tasks, reducing the need for manual intervention and potentially improving the efficiency and effectiveness of the preprocessing stage.

Existing approaches to automated data preprocessing encompass a wide range of techniques, including:

- **Imputation:** This involves handling missing values by estimating or replacing them based on observed data.
- **Outlier Detection and Removal:** This involves identifying and addressing data points that deviate significantly from the norm, which can skew analysis results.
- **Feature Selection and Engineering:** This involves selecting the most relevant features for analysis and creating new features that capture meaningful patterns in the data.
- **Data Transformation** This involves converting data into a more appropriate format. This can include scaling or normalizing numerical features, encoding categorical features (e.g., one-hot encoding, label encoding), and applying mathematical transformations (e.g., logarithmic or exponential transformations).

While these techniques offer significant advantages in terms of efficiency and automation, they also face challenges. The selection of appropriate preprocessing techniques often requires domain knowledge and experimentation. Additionally, the effectiveness of automated methods can vary depending on the characteristics of the data and the specific analysis task.

2.2 AutoML Pipelines

Automated Machine Learning (AutoML) pipelines aim to streamline the entire machine learning process, from data preprocessing to model selection and hyperparameter tuning. These pipelines leverage various algorithms and techniques to automate the traditionally manual and iterative steps involved in building machine learning models. The goal of AutoML is to make machine learning more accessible to non-experts and to accelerate the development and deployment of machine learning solutions.

Several AutoML pipelines have been developed in recent years, each with its own strengths and weaknesses. Some popular examples include:

- **Auto-WEKA:** An early AutoML system that uses Bayesian optimization to search for the best combination of preprocessing steps and machine learning algorithms.
- **TPOT:** A tree-based pipeline optimization tool that uses genetic programming to evolve pipelines of data cleaning and machine learning operations.
- **Auto-Sklearn:** An extension of Auto-WEKA that incorporates more recent advancements in machine learning and hyperparameter optimization.

These AutoML pipelines have demonstrated promising results in various domains, including image classification, natural language processing, and tabular data analysis. However, they often focus on general machine learning tasks and may not be specifically tailored to the challenges of survival analysis, particularly in the presence of missing data.

2.3 Survival Analysis

Survival analysis is a statistical method for analyzing time-to event data. It is widely used in various fields, including medicine, engineering, and social sciences. In survival analysis, the primary goal is to model the time until an event of interest occurs, such as death, disease progression, or customer churn.

Key concepts in survival analysis include:

- **Survival Function**: This function represents the probability that an individual survives beyond a certain time point.

- **Hazard Function**: This function represents the instantaneous rate of experiencing the event of interest at a given time, given that the individual has survived up to that time.
- **Censoring**: This refers to situations where the event of interest is not observed for some individuals within the study period. Censoring can be right-censored (the event occurs after the study period), left-censored (the event occurs before the study period), or interval-censored (the event occurs within a known interval).

Various models are used for survival analysis, including:

- **Kaplan–Meier:** This model is a cornerstone of survival analysis, serving as a non-parametric method for estimating the survival function from lifetime data. It provides a visual representation of the probability of surviving beyond a given time point.
- **Cox Proportional Hazards:** This model assumes that the hazard ratio between two individuals is constant over time. It is widely used due to its interpretability and ability to handle multiple predictors.
- **Random Survival Forest:** This model is an extension of random forests to survival analysis. It uses multiple decision trees to estimate survival probabilities and can handle complex relationships between predictors and survival time.

2.4 Reinforcement Learning and Q-Learning

Reinforcement learning (RL) is a type of machine learning where an agent learns to interact with an environment by taking actions and receiving rewards or penalties based on its actions. The agent's goal is to maximize its cumulative reward over time by learning the optimal policy, which is a mapping from states to actions.

Q-Learning is a model-free RL algorithm that uses a Q-table to store the expected cumulative reward for each state-action pair. The agent updates the Q-table iteratively based on its experiences and learns the optimal policy by choosing actions that maximize the expected cumulative reward.

2.5 Learn2Clean Architecture

Learn2Clean is a tool for automating data preprocessing tasks using Q-Learning. It selects the optimal sequence of preprocessing tasks for a given dataset and ML model to maximize the quality of the ML model's results. The architecture of Learn2Clean consists of the following components:

- **Q-Learner**: This is the core component that implements the Q-Learning algorithm. It learns the optimal policy for selecting preprocessing tasks.
- **Environment**: This represents the dataset and ML model that the Q-Learner interacts with. It provides feedback to the Q-Learner in the form of rewards based on the quality of the ML model's results after each preprocessing task.
- **Actions**: These are the preprocessing tasks that the Q-Learner can choose from. They can include imputation, encoding, scaling, and feature engineering tasks.
- **States**: These represent the current state of the data after each preprocessing task. The Q-Learner uses the state information to select the next action.
- **Rewards**: These are numerical values that the environment provides to the Q-Learner based on the quality of the ML model's results. The Q-Learner uses the rewards to update its Q-table and learn the optimal policy.

2.6 Related Work

This thesis builds upon the Learn2Clean framework, which utilizes reinforcement learning to automate the data cleaning process. While the original Learn2Clean focused on general data preparation tasks, this work extends its capabilities to the specific domain of survival analysis with missing data.

Several AutoML pipelines have been developed in recent years, but most of them do not explicitly address the challenges of missing data in survival analysis. Some notable examples include:

- **Auto-WEKA:** This system focuses on automating the selection of machine learning algorithms and hyperparameters but does not explicitly address data cleaning or missing value imputation.
- **TPOT:** While TPOT includes some data cleaning operations in its pipeline, it does not specifically target the unique characteristics of survival data or the complexities of missingness mechanisms.
- **Auto-Sklearn:** Similar to Auto-WEKA, Auto-Sklearn primarily focuses on algorithm selection and hyperparameter optimization, with limited support for data cleaning and missing value handling.

In contrast to these general AutoML pipelines, L2C4E offers a more targeted approach to data cleaning in survival analysis. By leveraging reinforcement learning, it can learn adaptive cleaning strategies that optimize the performance of survival models in the presence of missing data. Furthermore, the extensions implemented in this thesis, such as the customizable reward function, selective preprocessing, and edge weight editing, provide additional flexibility and control over the cleaning process, making it more adaptable to diverse datasets and analysis tasks.

Chapter 3

Methodology

The original Learn2Clean[1] framework, while powerful in its ability to optimize preprocessing pipelines for traditional machine learning models, lacked the capacity to address survival analysis tasks. Additionally, its handling of categorical data and reward mechanisms presented opportunities for improvement. In the following sections, these improvements will be discussed.

3.1 L2C4E for Survival Analysis

This section details the core contribution of integrating survival analysis models into the framework, along with the necessary adaptations to the reward structure and action space.

3.1.1 Survival Model Selection and Integration

Three survival analysis models were carefully selected to integrate into L2C4E, each chosen for its unique strengths and applicability to a variety of survival analysis scenarios:

- **Cox Proportional Hazards (Cox PH) Model:**[2] This widely-used model is valued for its interpretability, allowing researchers to quantify the impact of different factors on the hazard rate. Its assumption of proportional hazards, while a limitation in some cases, makes it a valuable tool for many applications.
- **Random Survival Forest (RSF):**[6] The RSF model, an ensemble method based on decision trees, offers robustness to nonlinearities and interactions in the data. Its non-parametric nature makes it a flexible choice when the underlying relationships are not well understood.

- **DeepHit Neural Network:**[4] This deep learning model leverages the power of neural networks to capture complex patterns and interactions in survival data. Its ability to model multiple competing risks makes it particularly well-suited for scenarios where individuals may experience different types of events.

Integrating these models required some important modifications to the Q-learner's action space. Actions were defined for selecting and configuring each model, ensuring seamless compatibility with the existing Learn2Clean architecture. Additionally, the TensorFlow backend of the DeepHit model was migrated from version 1 to version 2 to ensure smooth operation within the Python 3.8 environment of Learn2Clean.

3.1.2 Adapting the Reward Structure

To guide the Q-learner effectively in the context of survival analysis, the reward structure was adapted. C-Index[5] metric from survival analysis was incorporated:

- **Concordance Index (C-index):** The C-index evaluates the model's ability to correctly rank individuals based on their risk of experiencing the event. A higher C-index indicates better discriminatory power.

By incorporating the C-index into the reward function, the Q-learner was incentivized to select preprocessing actions that improved the performance of the survival models according to this metric. This involved striking a balance between rewarding immediate improvements and encouraging exploration of potentially beneficial preprocessing sequences in the long run.

3.1.3 Modifying the Action Space

The action space of the Q-learner, representing the set of possible preprocessing actions, was expanded to accommodate specific requirements of survival analysis. New actions were introduced for handling censored data, which is common in survival analysis, as well as for encoding time-varying covariates, which change over the course of the observation period.

3.2 Enhancing Flexibility and Customization

3.2.1 Configuration File

To enhance the usability and adaptability of L2C4E, a JSON-based configuration file was implemented. This file allows users to customize various aspects of the tool's behavior, including:

- **Model Hyperparameters:** Users can specify values for the hyperparameters of each survival analysis model, tailoring them to specific datasets or preferences.
- **Data Preprocessing Options:** The configuration file enables the selection and configuration of preprocessing techniques, such as imputation methods or scaling options.
- **Q-Learning Settings:** Parameters related to the Q-learning algorithm, such as the learning rate and exploration rate, can be adjusted through the configuration file.

L2C4E is designed to read and parse the configuration file, providing clear error messages if any invalid settings are encountered.

3.2.2 Dynamic Reward Matrices

Dynamic reward matrices, defined in JSON format, offer a powerful mechanism for customizing the reward function. This approach allows users to define rewards that are specific to the dataset, the chosen survival model, or the particular analysis goals. The Q-learner dynamically loads and utilizes these reward matrices during the learning process, enabling more efficient exploration and optimization of preprocessing sequences. Users can also import a JSON file defining a custom reward graph instead of editing the default file, allowing for tailored optimization based on specific domain knowledge or evaluation criteria. Moreover, the framework supports the ability to disable specific preprocessing methods (e.g., Median imputation) or entire preprocessing steps (e.g., imputation altogether). This enables users to fine-tune the cleaning process and exclude techniques that might not be suitable for their data or analysis goals. Furthermore, users can directly modify the weights of individual edges connecting preprocessing steps within the L2C4E reward graph. This allows for more granular control over the transition probabilities between steps, enabling the exploration of alternative cleaning sequences or the exclusion of specific transitions.

3.2.3 Working Modes

To provide users with a range of options for data preprocessing and analysis, four distinct working modes were implemented in L2C4E:

1. **Main L2C4E Algorithm:** This mode utilizes the core Q-Learning algorithm to identify the optimal sequence of preprocessing steps that maximize the performance of the chosen survival analysis model. This is the primary mode for automated pipeline optimization.

2. **Random Cleaning Mode:** In this mode, users can specify a desired number of random experiments. The tool generates random preprocessing pipelines and evaluates their performance, providing insights into the impact of different preprocessing choices. This mode can serve as a baseline for comparison with the optimized pipeline.

3. **Custom Pipeline Mode:** This mode allows users to define their own preprocessing pipelines using a simple text file format. Each line in the file specifies a sequence of preprocessing methods, providing flexibility for testing specific hypotheses or domain knowledge.

4. **No Preparation Mode:** This mode bypasses all preprocessing steps, directly passing the raw dataset to the chosen survival analysis model. This can be useful for establishing a baseline performance without any preprocessing.

The inclusion of these working modes significantly enhances the utility of the framework. Modes 2, 3, and 4, in particular, offer valuable baseline comparisons for evaluating the effectiveness of the optimized pipeline generated by the Q-learning algorithm.

Chapter 4

Results & Limitations

4.1 Experiments Setup

To evaluate the effectiveness of L2C4E, a series of experiments were conducted. The core approach involved simulating missingness in a real-world dataset and comparing the performance of the framework to baseline pipelines. Missingness was simulated using the Jenga framework[8], a tool designed to introduce missing values in a controlled manner. Three missingness mechanisms were employed:

1. **Missing Completely at Random (MCAR):** The probability of a value being missing is independent of both observed and unobserved data.
2. **Missing at Random (MAR):** The probability of a value being missing depends on observed data but not on the missing value itself.
3. **Missing Not at Random (MNAR):** The probability of a value being missing depends on the unobserved missing value itself.

For each mechanism, five datasets were generated with varying degrees of missingness: 10%, 20%, 30%, 40%, and 50%.

The performance of three distinct pipelines was evaluated on each dataset:

1. **L2C4E:** The reinforcement learning-based pipeline under investigation, designed to learn the optimal sequence of data cleaning operations.
2. **Custom Pipeline:** A baseline pipeline incorporating established techniques for imputation (MICE)[10], feature selection (LASSO with Cox Proportional Hazards)[9], deduplication, and outlier removal (Martingale Residuals method).
3. **Random Pipelines:** Ten random pipelines were generated for each dataset, each with a random combination of cleaning operations. The average performance of these pipelines served as an additional baseline.

The primary evaluation metric was the C-Index, a well-established measure for assessing the discriminatory power of survival models.

4.2 Dataset Description

The experiments utilized the Rotterdam (breast cancer) dataset which is part of the survival package in R, as well as the FLCHAIN dataset which studies a type of cancer called multiple myeloma. Rotterdam comprises 2982 records and 14 features, while FLCHAIN comprises 7874 records and 11 features. Rotterdam dataset was chosen due to its relevance in survival analysis and its lack of inherent missing values, allowing for controlled simulation of missingness while the choice of FLCHAIN is because of its fame in survival analysis and unlike Rotterdam, it contains missing values by default so no missingness simulation is needed.

4.3 Experiment 1

The initial experiment was conducted on the Rotterdam dataset. To assess the impact of different missingness patterns, five variations of the dataset were generated for each missingness mechanism (MCAR, MAR, and MNAR), each with increasing percentages of missing values. The performance of L2C4E, the custom pipeline, and the random pipelines was evaluated on these datasets, and the results are presented in the following line graphs.

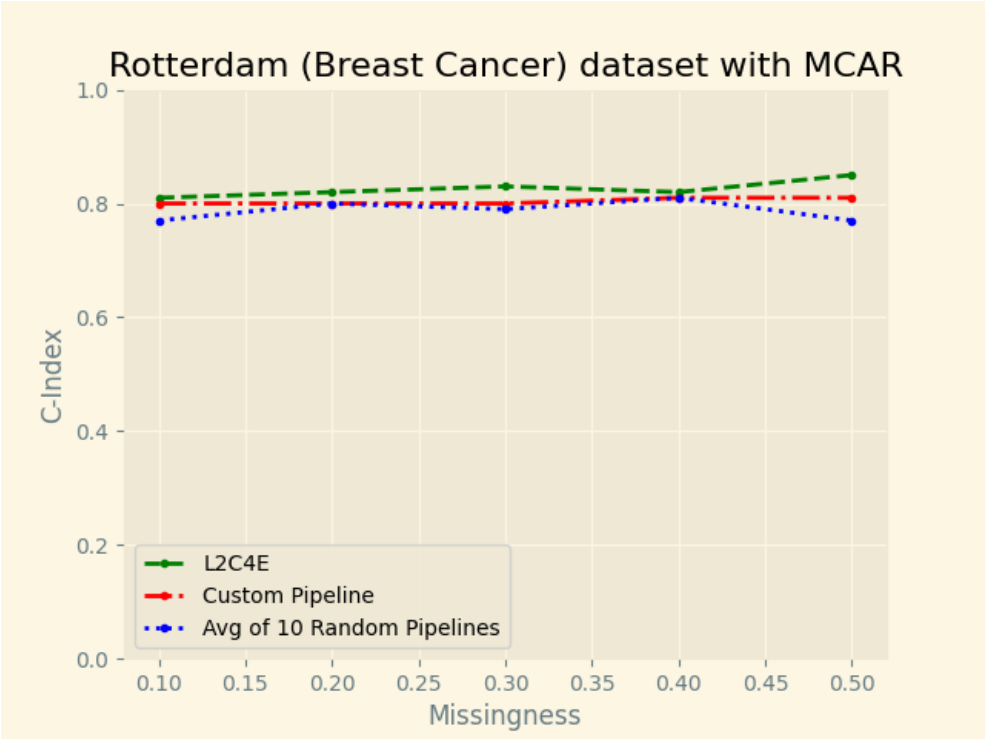

Figure 4.1: C-Index Comparison for MCAR Missingness on the Rotterdam Dataset

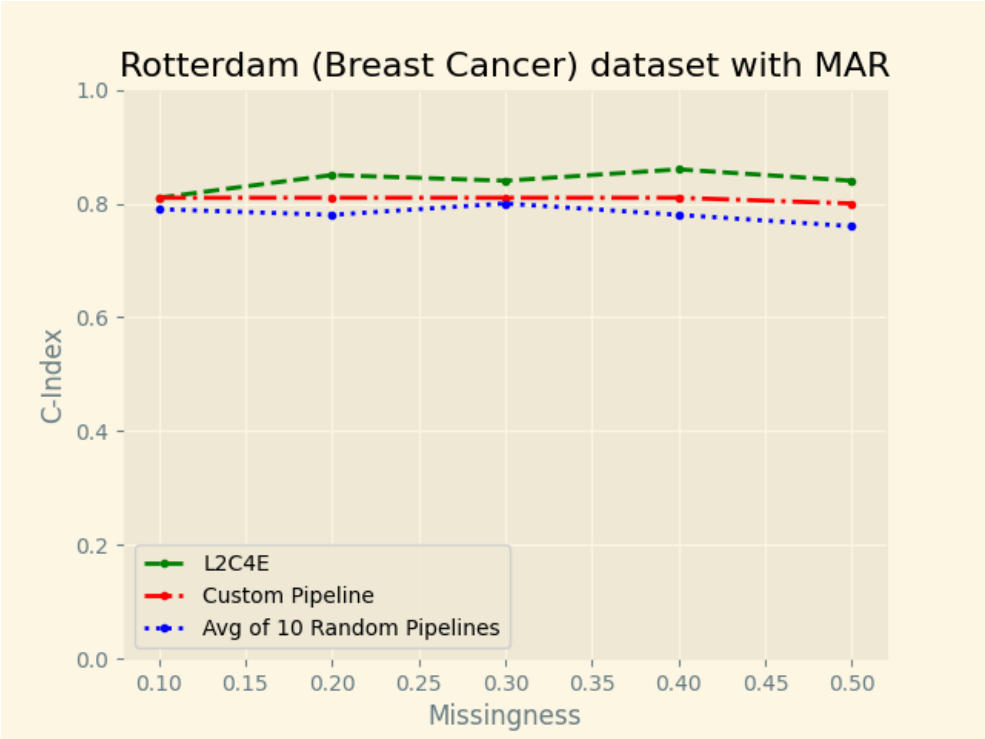

Figure 4.2: C-Index Comparison for MAR Missingness on the Rotterdam Dataset

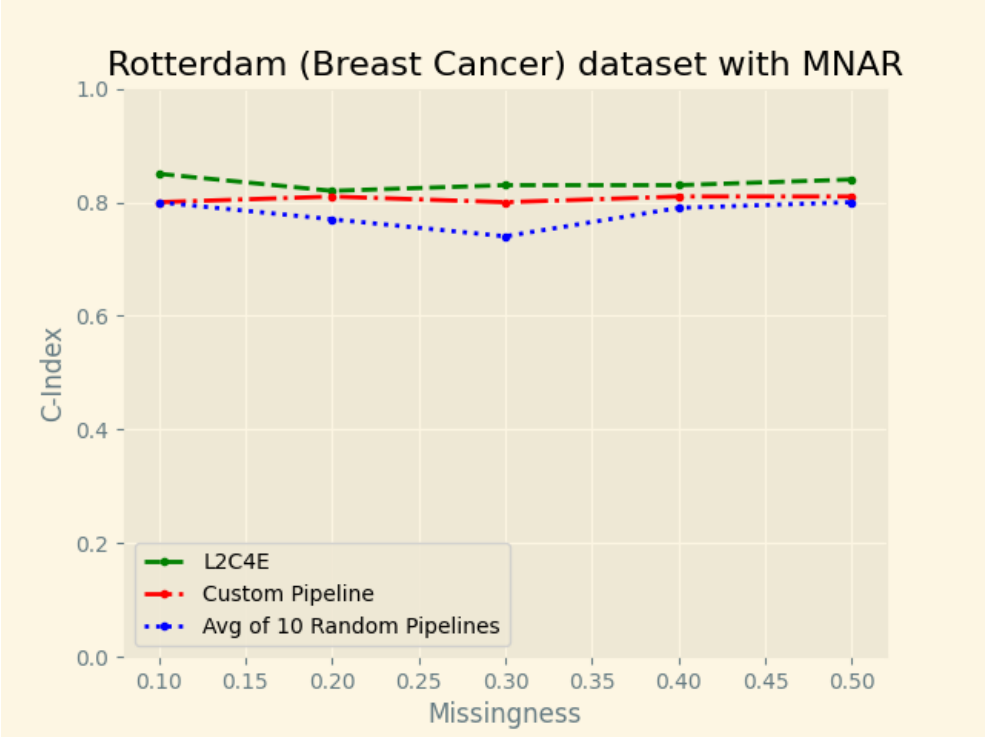

Figure 4.3: C-Index Comparison for MNAR Missingness on the Rotterdam Dataset

4.4 Experiment 2

On the FLCHAIN dataset with inherent missing values, L2C4E achieved a C-Index score of 0.66, outperforming the custom pipeline (C-Index = 0.58) and the average of random pipelines (C-Index = 0.61). This further demonstrates the practical applicability of L2C4E in real-world scenarios where data cleaning is crucial for accurate survival analysis.

4.5 Results Analysis and Discussion

The experimental results consistently demonstrate the superior performance of L2C4E in handling missing data for survival analysis. It outperformed both the custom pipeline and the random pipelines across different missingness mechanisms, rates, and datasets. This suggests that L2C4E's ability to learn adaptive cleaning strategies is effective in improving the quality of data and, consequently, the performance of survival models.

The superior performance on the FLCHAIN dataset, which inherently contains missing values, further highlights the practical relevance of the framework. It showcases its potential to address the challenges posed by missing data in real-world settings, where data cleaning is often a critical step in the analysis process.

However, it is important to acknowledge certain limitations of the experiments. The evaluation was conducted on a limited number of datasets, and further research is needed to assess the generalizability of the findings to other domains and data characteristics. Additionally, the focus on C-Index as the primary evaluation metric might not capture all aspects of model performance. Future work could explore the impact of L2C4E on other evaluation metrics and different types of survival models.

Despite these limitations, the results provide strong evidence for the effectiveness of L2C4E in handling missing data for survival analysis. The framework's ability to learn adaptive cleaning strategies has the potential to improve the accuracy and reliability of survival models, ultimately leading to better decision-making in various fields, including healthcare and clinical research.

Chapter 5

Conclusion & Future Work

5.1 Conclusion

This thesis investigated the application of reinforcement learning, through the L2C4E framework, to address the challenge of missing data in survival analysis. The experiments demonstrated the effectiveness of the framework in learning optimal data cleaning strategies, leading to improved performance of survival models compared to traditional and random pipeline approaches. The consistent superiority of the framework across different missingness mechanisms, rates, and datasets, including both simulated and real-world scenarios, highlights its potential to enhance the accuracy and reliability of survival analysis in various domains.

5.2 Future Work

While this research provides promising results, there are several avenues for future exploration to further advance the capabilities of L2C4E and its applications:

- Investigate the performance of L2C4E on datasets with more complex or mixed missingness mechanisms, beyond the traditional MCAR, MAR, and MNAR scenarios.
- Evaluate the scalability and generalizability of the framework by testing it on larger and more diverse datasets from various domains.
- Explore the use of additional evaluation metrics beyond C-Index to gain a more comprehensive understanding of the impact of the algorithm on model performance.
- Investigate the integration of domain-specific knowledge instead of the Q-learning algorithm to enhance the performance in specific applications where expert insights can guide the data cleaning process.

- Develop a user-friendly interface for the framework, in order to make it more accessible to researchers and practitioners who may not have extensive expertise in reinforcement learning.
- Explore the potential of meta-reinforcement learning techniques to enable the adaptation of reward functions in L2C4E, allowing it to generalize better to new datasets and tasks. Additionally, investigate the use of other reinforcement learning algorithms beyond Q-learning to potentially discover even more effective cleaning strategies.

By pursuing these future research directions, we can further unlock the potential of L2C4E and contribute to the development of more robust and reliable survival analysis methods in the face of missing data.

Appendix A

List of Abbreviations

C-Index	Concordance Index
CoxPH	Cox Proportional Hazards Model
DFKI	German Research Center for Artificial Intelligence
GUC	German University in Cairo
KNN	K-Nearest Neighbors
LASSO	Least Absolute Shrinkage and Selection Operator
MAR	Missing at Random
MCAR	Missing Completely at Random
MICE	Multiple Imputation by Chained Equations
MNAR	Missing Not at Random
RSF	Random Survival Forest

List of Figures

Bibliography

[1] Laure Berti-Equille. Learn2clean: Optimizing the sequence of tasks for web data preparation. *The WebConf*, 2019.

[2] Cameron Davidson-Pilon. lifelines, survival analysis in Python. https://github.com/camDavidsonPilon/lifelines.

[3] Carlos Vladimiro Gonzalez Zelaya. Towards explaining the effects of data preprocessing on machine learning. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 2086–2090, 2019.

[4] Changhee Lee, William R. Zame, Jinsung Yoon, and Mihaela van der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. https://github.com/chl8856/DeepHit.

[5] Enrico Longato, Martina Vettoretti, and Barbara Di Camillo. A practical perspective on the concordance index for the evaluation and selection of prognostic time-to-event models. *Journal of Biomedical Informatics*, 2020.

[6] Sebastian Pölsterl. scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212):1–6, 2020.

[7] Cedric Renggli, Luka Rimanic, Gürel Nezihe Merve, Bojan Karlaš, Wentao Wu, and Ce Zhang. A data quality-driven view of mlops. *arXiv preprint arXiv:2102.07750*, 2021.

[8] Sebastian Schelter, Tammo Rukat, and Felix Biessmann. Jenga - a framework to study the impact of data errors on the predictions of machine learning models. *EDBT 2021 Industrial and Application Track*, 2021.

[9] R Tibshirani. The lasso method for variable selection in the cox model. *Statistics in medicine*, 16(4):385—395, February 1997.

[10] Stef van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67, 2011.

[11] Hadley Wickham. Tidy data. *Journal of Statistical Software*, 2014.