TU WIEN Informatics

Primary Delay Injection Models: A Data-Driven Approach to Robust Stochastic Railway Simulation

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Valentin Peter
Registration Number 12143831

to the Faculty of Informatics

at the TU Wien

Advisor: DI Dr. techn. Nikolas Popper
Assistance: Nadine Schwab
Hannah Kastinger
Matthias Rössler

Vienna, March 25, 2025

Valentin Peter

Nikolas Popper

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Valentin Peter

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 25. März 2025

Valentin Peter

Acknowledgements

I would like to sincerely thank my supervisor, Nikolas Popper, for his openness to my interests and for taking the time to help me find a topic that is both engaging and challenging. The meetings with him and my co-supervisors Hannah Kastinger, Nadine Schwab and Matthias Rössler, provided me with a deeper understanding of the intricacies of railway modeling and stochastic simulation. Without their support, this thesis would not have been possible. Especially their willingness to regularly discuss new ideas and approaches while ensuring that I kept the bigger picture in mind was invaluable. My special thanks also go to the larger team at DWH GmbH, whose input during broader discussions contributed to the development of this thesis.

I am deeply grateful to my parents for their support throughout my seven years of study. They provided the security, freedom, and encouragement that allowed me to explore my own interests and passions. Our lively discussions on all aspects of life continually inspired me to challenge norms and pursue new ideas. Also, I want to thank my sister for always having an open ear and offering her thoughtful advice.

Lastly, I would like to express gratitude to my girlfriend for her support throughout the entire process of writing this thesis. Her patience and willingness to listen to my thoughts and uncertainties meant a lot to me.

Kurzfassung

Diese Arbeit befasst sich mit der Herausforderung, Primärverspätungen im Eisenbahnbetrieb genau zu erfassen. Diese stellen einen entscheidenden Faktor für die effektive Fahrplangestaltung von Eisenbahnverkehrsunternehmen dar. Traditionelle Ansätze beruhen oft auf vereinfachten Annahmen über Verspätungsverteilungen oder auf deterministischen Modellen. Solche Modelle sind jedoch nicht in der Lage die komplexen, stochastischen Verspätungsmuster der Realität adäquat abbilden. Ziel dieser Arbeit ist es daher, die Modellierung von Primärverspätungen zu verbessern. Hierzu werden Primärverspätungs-Injektionsmodelle (PDIMs) entwickelt und bewertet. Die damit generierten Primärverspätungen dienen anschließend als stochastische Eingabe für makroskopische Eisenbahnsimulationssysteme.

Die Methodik umfasst eine detaillierte Analyse realer Eisenbahndaten. Dazu gehören Zugfahrtaufzeichnungen und Infrastrukturmodelle, die von der ÖBB-Infrastruktur AG bereitgestellt werden. Daraufhin werden verschiedene statistische und maschinelle Lerntechniken untersucht. Besonders wichtig ist hierfür die Identifikation optimaler Wahrscheinlichkeitsverteilungen (Log-Normal und Pareto) zur Modellierung von Primärverspätungen. Auf dieser Grundlage wurden verschiedene Modelle entwickelt, wie zum Beispiel CatBoost-basierte Modelle zur Parametrisierung von Wahrscheinlichkeitsverteilungen (DistBoost), Ensemble-Modelle und bayesianische hierarchische Modelle.

Ein zentrales Ergebnis dieser Diplomarbeit ist die überlegene Leistung des DistBoost-Modells im Vergleich zu anderen Verfahren. Insbesondere kann es epistemische Unsicherheit effektiv erfassen, indem es Wahrscheinlichkeitsverteilungen mittels maschinellen Lernens parametrisiert. Dabei kann ein ausgewogenes Verhältnis zwischen Genauigkeit, Effizienz und Anpassungsfähigkeit hergestellt werden. Die Arbeit verdeutlicht, wie wichtig es ist, sowohl aleatorische als auch epistemische Unsicherheit in der stochastischen Verzögerungsmodellierung zu berücksichtigen. Zudem werden Grenzen in der Anwendung von punktuellen Fehlerkennzahlen, wie dem mittleren absoluten Fehler, aufgezeigt. Es erweist sich auch die Notwendigkeit, die Anpassung der Verteilung für verschiedene Eingaben systematisch zu evaluieren. Besonderes Augenmerk wird auf die genaue Modellierung von Primärverspätungen für die Verbesserung von Simulationsergebnissen gelegt, besonders im Umgang mit Extremwerten. Insgesamt liefert diese Arbeit einen fundierten theoretischen Rahmen zur Modellierung primärer Verspätungen und leistet damit einen wesentlichen Beitrag zur Optimierung der Fahrplangestaltung.

Abstract

This thesis addresses the challenge of accurately sampling primary delays in railway operations, which represent a critical factor for effective scheduling and disruption management for railway operators. Traditional approaches often rely on simplified assumptions about delay distributions or use deterministic models, failing to capture the complex, stochastic nature of real-world delays. This thesis aims to enhance the modeling of primary delays through the development and evaluation of Primary Delay Injection Models (PDIMs), which serve as a stochastic input to macroscopic railway simulation systems.

The methodology involves a detailed analysis of real-world railway data, including train travel records and infrastructure models provided by the Austrian Railway Infrastructure provider (ÖBB-Infrastruktur AG). Various statistical and machine learning techniques are explored. Particularly important is the identification of optimal probability distributions (Log-normal and Pareto) for modeling primary delays. Based on this foundation, various models were developed, including CatBoost-based parameterized distribution models (DistBoost), ensemble models, and Bayesian hierarchical models.

A key finding of this thesis is the superior performance of the DistBoost model compared to other approaches. It effectively captures epistemic uncertainty through machine learning parameterization of probability distributions, achieving a balance between accuracy, efficiency, and adaptability. The thesis demonstrates the importance of considering both aleatoric and epistemic uncertainty in stochastic delay modeling, highlighting the limitations of relying solely on pointwise error metrics and emphasizing the necessity of evaluating distributional fit. Furthermore, the research underscores the critical role of accurate primary delay modeling in improving simulation outcomes, including the proper representation of extreme values. Ultimately, this work contributes to the advancement of railway simulation by providing a robust framework for modeling primary delays and thereby optimizing scheduling and disruption management.

Contents

CHAPTER 1

Introduction

1.1 Problem Statement

The Austrian Federal Railway (ÖBB) is the primary operator of freight and passenger services in Austria. In the railway sector, various planning tasks require a framework to effectively and accurately assess the real-world feasibility of a given plan. One of the crucial planning tasks in this domain is the preparation of traction unit circulation. Currently, this task is handled by experts who have a deep understanding of the railway network and operational constraints. However, growing complexity, personnel shortages and the need for rapid rescheduling - especially in the face of unforeseen disruptions such as floods - are making manual planning increasingly difficult [sta24].

These scheduling problems can be formulated as optimization problems; in the case of traction unit circulation, the Locomotive Scheduling Problem (LSP)[Gle57], which aims to find the optimal assignment of traction units to scheduled trains. Whereby the optimal assignment is defined by finding the minimum of the overall operating costs. The definition of operating costs varies by the operator but could be e.g. as few empty runs as possible or low energy consumption.

Numerous approaches have been proposed to solve this problem. For instance, Frisch et al. [FHJW19] developed a model based on a sparse multi-graph structure to represent scheduling requirements and utilized a Mixed-Integer Linear Program (MILP) to find optimal solutions. This approach also incorporated maintenance constraints, allowing for a more practical schedule optimization. Although optimization models are computationally efficient and generate theoretically optimal schedules, real-world railway operations face uncertainties that cause delays. These delays are categorized as primary (from unexpected events like equipment failures or high passenger volumes) and secondary (caused by operational conflicts from primary delays, also known as congestion). Incorporating these into the optimization model is complex.

To address this, Rößler et al.[RWJ+20] developed an agent-based simulation model with an event-based time update to evaluate schedules. Agents in the model represent trains and traction units, but also the infrastructure elements of a rail network like stations and sections. Each of these agents has predefined behaviors. The events and agents operate on a macroscopic level, which defines behaviors only between Operational Control Points (OCP) representing mostly train stations or larger junctions but partially also signals. Therefore, travel times cannot be correctly simulated and are therefore extracted from the schedules. Microscopic systems, on the other hand, model each switch and signal in the network individually. Further, they calculate acceleration and braking as well as times needed to hand out movement authorities in a train protection system. Some systems also model driver behavior. This makes the microscopic system much more precise but also computationally expensive. One of the assumptions made is that, while on a coarser granularity, the model accurately simulates train movements in the network and therefore has the capability to correctly calculate secondary delays.

To evaluate schedules in real-world conditions, the model needs to accurately represent primary delays as well, requiring these delays to be injected in each simulation step. However, differentiating between primary and secondary delays is not straightforward. Although datasets differentiating between primary and secondary delays exist, a railway operator's classification may not align with the needs of a simulation system. For the simulator, any delay it cannot model, such as those caused by unknown trains from other rail transport providers, is considered a primary delay, whereas the rail provider would classify these as secondary. In response to this challenge, a network-analysis-based method for disaggregating primary and secondary delays was proposed by Schwab et al. [SRK+24], which forms the underlying data basis for this thesis. Against this background, this thesis aims to improve modeling of primary delays within the simulation.

1.2 Contribution

The goal of this thesis is to develop and test different Primary Delay Injection Models (PDIMs). Rößler et al. [RWJ+20] developed an initial PDIM, where different Machine Learning (ML) Models were employed to tackle the regression task. The data used to fit these models was still aggregated total delay data. This model operated under the assumption that delays follow a normal distribution. Various models, including neural networks, k-nearest neighbors regression and random forest regression were used to predict two key parameters for two probability distributions: (1) ρ, the probability of a primary delay occurring as input into the binomial distribution, and (2) μ, the mean of a normal distribution from which the delay was drawn.

This master thesis seeks to improve upon this initial model. The current model's assumption of normally distributed primary delays may not accurately reflect the complexities and irregularities of real-world railway operations. To adress this, the thesis will explore alternative approaches to better model delays and will be the first using disaggregated delays. Primary delay sampling is, to my knowledge, in all read papers on railway

simulation, microscopic and macroscopic, a neglected part of the simulation model, as the fitting of the delay distributions is only explained briefly.

In the course of this thesis, three main model families will be evaluated. The first approach, similar to the previous implementation, uses a ML Model to parameterize a probability distribution for each event in the dataset. These parameter predictions can then be used to sample new delays from the probability distribution. However, instead of assuming normality, a detailed analysis is conducted to determine which distribution best fits the data. Additionally, a state-of-the-art model is trained to estimate the appropriate parameters.

The second approach employs Bayesian modeling techniques based on Markov Chain Monte Carlo (MCMC) sampling. For these Bayesian models, selecting the best-fitting distribution is also essential for modeling the likelihood. Multiple Bayesian models with varying levels of complexity will be implemented and evaluated.

Finally, the third approach trains an ensemble of ML models. This is done independently with a random component to enhance robustness. In the end, this ensemble can be used to predict many delays given one input.

The evaluation of the proposed models will be conducted on two levels. First, its theoretical performance will be assessed by measuring the pointwise error between predictions and true values, as well as evaluating the overall fit of the predictions. For the Bayesian models, examining the Expected Log Pointwise Predictive Density (ELPD) will be used to compare posterior distributions to the observed data. Since all models are stochastic, it is important to examine the diversity of their predictions, whether they generate a range of possible outcomes or consistently produce similar values for the same inputs.

However, theoretical performance alone provides only limited insight, as the models are intended to function within a simulation framework. Therefore, the preferred evaluation metric is their performance in conjunction with the simulation model, allowing for a more comprehensive assessment of their effectiveness in practical applications.

1.2.1 Research Questions

The research questions to be answered are:

- R1: Which likelihood distribution best captures the statistical characteristics of disaggregated primary delays in railway operations and which test is best to evaluate the fit?
- R2: How do models which represent either aleatoric uncertainty, epistemic uncertainty or those who represent both impact model performance?
- R3: How can the stochastic delay model be extended to generalize for predicting primary delays in novel railway scenarios (e.g. new trains, routes, altered timetables)?

- R4: How can the stochastic primary delay model, in conjunction with the simulation loop, be validated by evaluating its ability to reproduce observed total delays?

1.3 Methodology

The methodology involves the following steps:

1. **Literature Review**

   A comprehensive review of existing research on train delay modeling will be conducted, covering primary, secondary, and total delays. Additionally, studies on delay disaggregation, railway simulation models, stochastic simulation and the evaluation of such models will be examined. Special attention will be given to methodologies for modeling stochastic random variables, particularly Bayesian models.

2. **Data Preprocessing**

   While the dataset provided has undergone initial preprocessing, additional steps will be undertaken to refine it for the task of primary delay prediction. These steps include standardizing and normalizing the values, engineering new features (such as infrastructure-related features) and conducting a thorough investigation of data quality.

3. **Selection of Likelihood or Target Distribution**

   Before developing the models, an appropriate likelihood distribution (for Bayesian models) or target distribution (for parameterized ML models) must be identified. Previous studies will be analyzed to determine suitable distribution candidates based on the characteristics of the problem. Various statistical tests will be conducted and evaluated to select one or more candidate distributions.

4. **Model Implementation and Training**

   The previously described models will be implemented and trained. The structure of the Bayesian models will be defined with the help of domain experts, while for the ML models hyperparameter optimization will be implemented to improve performance.

5. **Theoretical Model Evaluation**

   The model will be validated using multiple approaches. Standard error metrics, such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), will be used to assess prediction accuracy, alongside inter-sample diversity metrics to evaluate stochastic variability. Using statistical tests the fit of the predictive distribution will be evaluated.

6. **Simulation Integration**

However, these metrics alone do not fully capture the model's interaction within the simulation system. Therefore, validation will also include integrating the Primary Delay Injection Model (PDIM) into an agent-based simulation model. The integration process requires connecting the Python-based prototype to the existing C# simulation framework. The model's effectiveness will be assessed by testing its ability to generalize to previously unseen timetables. While the impact of the model on entirely new schedules and infrastructure cannot be evaluated due to data constraints, potential implementation strategies for such cases will be discussed.

CHAPTER 2

Background

In this section, the state of the art on multiple topics regarding railway modeling will be reviewed, even though the main topic of this thesis is the stochastic modeling of primary delays, a broader introduction into the field is necessary. First, a primer on delay types and disaggregation will be given. Then, the focus will be on delay modeling and the difference between delay prediction and delay sampling. The delay prediction task aims to predict a future delay as accurately as possible. This is especially needed in passenger information systems, freight tracking systems and delay management [OFC+17]. Delay sampling, on the contrary, has a different goal; instead of a deterministic prediction for each input, multiple possible outcomes representing many realistic scenarios need to be sampled.

Additionally, various approaches to railway simulation models will be reviewed. These models serve multiple purposes in the rail industry, including capacity estimation [TNv20], timetable optimization [HB23] and traction unit circulation planning [RWJ+20]. The focus will be on the underlying modeling paradigms, rather than specific applications, as this thesis aims to enhance the models' ability to simulate real-world behavior and is therefore independent of any particular task.

The field of train delay modeling and prediction can be categorized along four axes:

- **Methodological Approaches**:
  - **Multi-step event-driven models**: Incorporate future states into the calculation of current states, using frameworks such as graph models, Bayesian networks, Markov chains, Petri nets and equation systems. These models are inherently stochastic [STBC22].
  - **Single-step data-driven models**: Predict delays without modeling event dependencies, often using regression, decision trees, neural networks, and support vector machines. These models are typically deterministic [STBC22].

- **Uncertainty Representation**:
  - Stochastic models capture the probability distribution of delay realizations but often assume fixed distributions, which may not account for real-time variability [CK18].
  - Deterministic models produce a fixed output for a given input but are often more complex and therefore capable at capturing complex relationships. As they are easier to fit, they can often deal with many input features [OFC+17].
- **Temporal Scope**:
  - **Static (Offline, Long-Term)**: Models run on a historic static dataset [CK18, TMP23].
  - **Dynamic (Online, Short-Term)**: Models adapt to real-time train position and delay information [CK18, TMP23].
- **Horizon Scope**:
  - **One station ahead**: One shot prediction for the delay at current event [TMP23].
  - **Static multiple stations prediction**: One shot prediction of delay at station S_i [TMP23].
  - **Dynamic multiple stations prediction**: Model updates the predictions as railway traffic information evolves (either real online information or simulation) [TMP23].

2.1 Delay Types and Disaggregation

For larger diverse train networks including freight and passenger transport, historical train records only contain the Total Delay (TD) of each train. In small studies on e.g. high-speed lines, disaggregated data can be available [WLL+17]. If this is not the case, a disaggregation model is needed. Palmqvist et al. [PJS23] introduced a technique to estimate the proportion of primary delays by injecting delays into a macroscopic simulation tool and observing the percentage required to reproduce actual running times. While this method reproduced running times effectively, it has certain limitations. The fact that a timetable can be reproduced with discerned primary and secondary delays does not indicate that the disaggregated values represent the real ones. Further, it assumes that delays are either purely primary or secondary, whereas real-world delays often involve a mix of both.

Network Based Approach

Schwab et al. [SRK+24] developed a network-based disaggregation model which determines blocked edges in the graph to discern primary from secondary delays. In the

proposed system, a delay is regarded as secondary if "one or more other trains can be found that were on the downstream following node or edge while train A was at the considered node or edge. The assumption is that these trains caused train A to stay longer at its current stop" [SRK+24].

The network based model can be tuned by two parameters, t_{node_free} is the time that has to pass after a train left the node until the next train can enter it, respectively t_{edge_free} is the time that has to pass before the next train can enter the edge. Thus the model only considers one node or edge ahead. All trains which can be explained by this process were labeled as *explained_delays*. Apart from the blocking, delays which are due to circulation delays of locomotives were also labeled as explained delays and the remaining delays are labeled as primary delays. The number of trains needed to block a train is defined by the capacity which is extracted from the schedules. This approach has the advantage of providing an explanation for the process while eventually missing microscopic processes, which could give more detailed information about the type of delay.

The model was validated in conjunction with the Agent Based Simulation model. When t_{node_free} and t_{edge_free} were set to zero, the true delays could be reproduced by the simulation model. With the threshold set to zero there are no additional delays injected yet, except the circulation-based delays. As soon as this threshold was higher e.g. 25s the error compared to the true values increased. The authors noted that there is "certainly potential in the simulation model to further improve the quality of the simulation results" [SRK+24].

Reinforcement Learning based Approach

Instead of disaggregating the delays, there is also the possibility to directly fit the primary delay parameters such that they, in combination with a simulation model, reproduce a given timetable. Cui et al. [CMZ16] used reinforcement learning as a method to fit parameters of disturbance distributions with the goal to reproduce real-world delays utilizing the Microscopic Simulation Environment RailSys. Due to limitations of the used simulator RailSys only the negative exponential distribution could be used as it is the only one supported by RailSys. Additionally, the models only accounted for three types of events and lacked information regarding the train type, stop, or track, which could have been used to adjust the model parameters. This approach simplifies the process by merging disaggregation and modeling into a single step. The method yielded great results, but the runtime performance was poor, even though the model had only six parameters. One reason for this was that the state-of-the-art simulator (RailSys) used in this paper was quite slow. To address this performance bottleneck, they developed a more efficient simulator, which improved performance even though their tested network had only approximately 73 stations.

In conclusion, there are multiple approaches for discerning primary and secondary delays. While the fractional sampling is simple, the network-based method resembles the

simulation model and is therefore effective to be used in conjunction with the simulation system.

2.2 Delay Modeling

Delay prediction and modeling is a widely studied field, as railway operations are vulnerable to delays, making accurate estimation highly valuable. A review on train delay prediction approaches identified a total of 47 papers on the topic up to 2022 [STBC22]. Research on delay prediction in the railway domain has traditionally focused on predicting total train delays and therefore modeling congestion and disturbances as one process. These methods can either be single-step data-driven, which means to predict the delay just by fitting a model to historic data, or multi-step event-driven, which have an underlying model of the railway systems dynamics. Most data-driven methods are deterministic, while the event-driven approaches are more likely to be stochastic. In the following two subsections we will mainly split the approaches along the deterministic vs. stochastic axis [STBC22].

According to the Review of Tiong et al. [TMP23] there is a substantial increase in the number of publications on the short-term prediction with a trend towards models that consist of a mixture of different models called hybrid-based models. These models can adapt to numerous railway traffic conditions through the combination of multiple models, resulting in a more robust train-delay-prediction-model [TMP23]. In recent studies utilizing data driven methods, there is an emphasis on one station ahead prediction. These models can then be extended to accomplish dynamic and static multiple stations prediction. To realize static multiple stations prediction, a simulation model, where data-driven model results are injected to simulate delays at downstream stations, is utilized. To achieve dynamic multiple station predictions the inputs are continuously updated as new information e.g., real-time train status or network conditions becomes available [TMP23].

2.2.1 Deterministic Delay Prediction Models

Although delay prediction is not the primary goal of this thesis, it is worthwhile to provide a brief overview of potential approaches, as these are extensively studied and their concepts can be applied to Disturbance and Delay Sampling as well.

In this thesis, deterministic delay prediction models are defined as those trained to minimize a loss function based on real-world delay values, which is commonly referred to as supervised machine learning. With the increasing availability of large datasets of train operations, data-driven methods have become more prominent. By framing the train delay (TD) prediction problem as a time-varying multivariate regression task [OFC+17], the rapidly advancing field of machine learning (ML) offers numerous approaches to address this challenge. For instance, Oneto et al. [OFC+17] employed Deep and Shallow Extreme Learning Machines to develop a data-driven model for predicting delays, using

a large dataset provided by Rete Ferroviaria Italiana (RFI), the Italian infrastructure manager (IM). Enhancing the dataset with external factors, such as weather data, resulted in a 10% improvement in prediction accuracy.

Nair et al. [NHL+19], in collaboration with the German railway operator Deutsche Bahn (DB), developed a dynamic ensemble/hybrid prediction model for train delays. Their approach combined a single-step, data-driven component using a random forest and kernel regression model to capture train-specific dynamics, with a multi-step, event-driven component employing a mesoscopic simulation model. The simulation accounted for variations in travel and dwell time, inferred track occupation conflicts, train connections, and rolling stock rotations [NHL+19].

Pineda-Jaramillo et al. [PJV23] explored several ML models to address the delay prediction task for freight trains on intermodal networks. They reformulated the problem as a binary classification task, categorizing trains as either delayed or not. Their results indicated that the CatBoost model outperformed other ML approaches based on predefined evaluation metrics.

These deterministic approaches, particularly those presented by Oneto et al. [OFC+17], demonstrate that ML models can be effectively trained on extensive datasets. However, since the output of Probabilistic Delay Impact Models (PDIMs) must be stochastic, deterministic models cannot be directly applied in their basic form. Nevertheless, they can be leveraged to estimate parameters for distributional models, making them feasible for use as PDIMs.

2.2.2 Primary Delay & Disturbance Modeling with Distributions

The simplest form of stochastic delay models are distributional models, which estimate one or more distributions from which delays or disturbances are sampled. Many railway simulation systems, such as RailSys and PROTON/PRISM, utilize this straightforward approach [Sip23a, CMZ16, ZBB+19].

Fitting primary, secondary, or combined delays to probability distributions is a well-established practice in railway scheduling. Early work favored normal or exponential distributions for modeling overall delays at stations [YGH02]. More recent studies have increasingly adopted flexible distributions, such as log-normal, gamma, and Weibull distributions, to model primary delays [Yua06, WLL+17, YHP+19, WLL+19]. For instance, multiple studies on high-speed rail (HSR) lines in China examined the fit of these distributions for primary delays, using maximum likelihood estimation to derive distribution parameters. Wen and Yang et al. [WLL+17, YHP+19, WLL+19] demonstrated that the log-normal distribution provided the best fit for their data, as validated by the Kolmogorov-Smirnov test.

Goverde [GCD13] analyzed the capacity of different railway signaling systems under disturbed conditions and fitted three distinct Weibull distributions to model disturbances for freight, intercity, and sprinter services. Yang [YHP+19] had detailed data on the

types of disturbances and therefore modeled each type separately. Wen [WLL+17] used operational data from a 1,096-km double-track line with 18 stations. Delays were recorded between February 24 and November 30, 2015, with delays exceeding 90 minutes discarded due to their random distribution. For primary delays on this line, the log-normal distribution was found to provide the best fit.

As these distributions are primarily used to sample delays in simulators, some studies were constrained by the available distributions. Consequently, many studies rely on the negative exponential distribution [CMZ16, HB23], as it is the default distribution provided by RailSys [RMC24].

2.2.3 Stochastic Delay Prediction Models

The approaches in the previous section assume fixed probability distributions for train delays and do not account for the influence of static or real-time information on train positions and delays on the parameters of these distributions [CK18].

There are also more advanced approaches in the field of stochastic models, for instance, Corman et al. [CK18] developed a delay prediction model utilizing a Bayesian network, which represents the railway network. The structure of the Bayesian network is determined under the assumption that train routes and orders are known. However, capturing the delay dependencies between freight trains operating on ad hoc paths and other trains poses challenges due to their low frequency of occurrence in historical data. This limitation highlights difficulties in modeling interactions between different train types under realistic operating conditions. The authors specifically examined the propagation of delays using the Bayesian network, demonstrating its potential to predict complex interdependencies between delays in a railway network.

2.3 Stochastic Simulation

Almost all real-world systems contain one or more sources of randomness, [Law13] this is especially true in macroscopic simulation which inherently does not model complete processes. Therefore, these random processes need to be modeled. In the past, this was done commonly by sampling from probability distributions. This kind of simulation is called stochastic simulation and requires input modeling, which is the construction of appropriate probability models that "characterize the stochastic behavior of the system inputs" [CAX20]. A prominent example are queuing models, where inter-arrival and service times are modeled.

Law et al. [Law13] illustrate the most common pitfalls when estimating distributions. The first pitfall is to replace the distribution by a static mean value as e.g. congestion only occurs when rare events from the tails of a distribution happen. The second pitfall is to use the wrong distribution to model a process, for example, the normal distribution cannot model exponential processes [Law13]. They proposed a method to address these issues by fitting a probability distribution to the data using e.g. maximum likelihood

estimation. Evaluation of the fit can be achieved with graphical procedures (e.g. Density-Histogram Plots) or goodness-of-fit tests (e.g. Kolmogorov-Smirnov (K-S) test). But for some data sets, there is simply no theoretical distribution which has a good fit. The reason for this can be that the data originated from two or more heterogeneous populations or processes or that the data were significantly rounded (e.g. train delays to the full minute) [Law13]. Another challenge in modeling a stochastic process (e.g., delays) using probability distributions is that the input process is assumed to be univariate, stationary, independent, and identically distributed (IID), which may not accurately reflect real-world conditions [CAX20].

The so far discussed methods can only represent aleatoric uncertainty, which represents intrinsic randomness of a phenomenon that cannot be reduced even if more information were to be collected [KD09]. Supervised machine learning models are by their very nature mostly used for interpolations by predicting new data points on the basis of historical data, which makes them deterministic in their outputs and their parameters [WdLNvV24]. However, a deterministic machine learning model can be used to parameterize a distribution as was done in [RWJ+20]. This then makes it capable of representing the aleatoric uncertainty.

Another important uncertainty is the epistemic uncertainty (a.k.a. input uncertainty), which is due to a lack of knowledge or data. This uncertainty is especially important if we have few data points. Approaches to modeling epistemic uncertainty generally fall into two categories: Frequentist and Bayesian. Both of these approaches use the available real-world data to make inference. The Bayesian approach makes it possible to incorporate prior subjective expert opinion. While it is possible to compute closed-form exact posterior distributions in the Bayesian case, more complicated models need analytical approximations either utilizing Variational Inference or Markov Chain Monte Carlo methods to approximate the posterior [CAX20].

Lately there has been a lot of research including more advanced ML-models to apply the combined approach of simulation and machine learning. A subfield of this is Machine-Learning Assisted Simulation. A prominent approach is to use surrogate models as cheap approximators of more time-consuming models [vRMS+20].

2.4 Railway Simulation

Simulation models in the railway domain can be broadly categorized into microscopic and macroscopic approaches. Microscopic simulation models aim to replicate real-world dynamics as accurately as possible. This often involves detailed representations of switch layouts, signaling systems, physical modeling of braking, and processing times for operations such as movement authorities within train signaling systems [JPS+22]. Microscopic simulation models are primarily used to estimate capacity concerning design decisions for specific segments of infrastructure [WS18], to evaluate the capacity of an existing railway line [LNC24, SNTG21], or to assess schedules [HB23]. The most notable

microscopic simulators, OpenTrack and RailSys, are both closed source, which limits the reproducibility and comparability of studies conducted using these tools [JPS+22].

The downside of microscopic simulation is often its computational complexity. While this is negligible for localized studies with a limited number of train runs, it becomes a significant challenge for large-scale railway modeling [ZBB+19]. Furthermore, it is well known that driver behavior varies among individuals and rarely aligns perfectly with the rigid rules set in microscopic simulators [JPS+22].

This is where macroscopic simulation systems come into play. Detailed infrastructure data is often unavailable on a global scale for large systems, such as the train network of an entire country. For example, the German Railway Operator Deutsche Bahn (DB) must simulate approximately 40,000 trains on a typical day. Macroscopic simulation is often much more practical for matching train schedules and historical running data, making it particularly suitable for scheduling tasks. The PRISM system (now called PROTON), developed by DB, models the network using a graph-based approach, where stations are represented as nodes and tracks as edges. Basic infrastructure information is provided for these elements. For nodes, attributes such as overtaking capability are included, while for edges, attributes such as the type of train protection system, the number of tracks, electrification status, and maximum speed are specified. Trains are also assigned various attributes. PRISM employs Monte Carlo discrete event simulation to model interactions based on a given schedule [ZBB+19, Sip23b].

The macroscopic simulator used in this thesis is RailwaySim, an agent-based macroscopic simulation tool used for robustness assessment studies. RailwaySim also represents the rail network as a graph, with tracks as edges and stations as nodes. Active agents, such as traction units, interact according to predefined schedules, occupying and releasing track segments as they progress [RWJ+20, BBP+23].

CHAPTER 3

Methodology

3.1 Datasets

The datasets used in this thesis were provided by ÖBB-Infrastruktur AG. The main dataset is the delay dataset which contains records of train travel in the Austrian Train Network, including scheduled times, for two consecutive days: the 14th and 15th of December 2022. The dataset includes both freight and passenger trains and provides entries for each stop or pass at an Operational Control Point (OCP), which primarily represents stations but also includes large junctions. The second dataset is the infrastructure model dataset which contains information about the OCPs and tracks in between them.

3.1.1 Delay Dataset

Each data entry is classified as either a stop or pass event and contains scheduled arrival and departure times, as well as the actual arrival and departure times. Therefore the dataset provides not only the delay records, but also implicitly encodes the train schedule. In the dataset, a stop event implicitly represents both a stop and a run, unless it is the first stop of a train journey. For clarity, this composite event is decomposed into separate stop and run events in the following analysis. Each train is uniquely identified by the trainpart_id, and consecutive events are numbered using a sequence number. Due to the decomposition of the stop event, a new sequence number is generated, where the run event precedes the stop event. The delay of a train is defined as the deviation between the actual and scheduled arrival or departure times. Therefore, the additional delay for a run event is calculated as the difference between the arrival delay and the departure delay at the previous OCP. For a stop event, the additional delay is defined as the deviation between the departure delay and the arrival delay. This newly decomposed structure makes the events much easier to interpret, as illustratively shown in Figure 3.1. In the revised structure, each event explicitly has a scheduled and actual duration, as well as an

additional delay. Although, this information could also be represented using timestamps, it has been transformed into durations for the purpose of clearer illustration.

Type	Sequence #	Arrival	Departure	Scheduled Arr.	Scheduled Dep.
Stop	1	14:00	14:00	14:00	14:00
Pass	2	14:30	14:30	14:20	14:20
Stop	3	14:50	14:55	14:40	14:50

Type	Sequence #	Actual Duration	Scheduled Duration	Additional Delay	Total Delay
Stop	1	0	0	0	0
Run	2	30	20	10	10
Run	3	20	20	0	10
Stop	4	5	10	-5	5

Table 3.1: Examplary decomposition of the composite stop event into seperate events for a given train.

For the training of the models the disaggregated primary delay is used instead of the additional delay. This primary delay stems from the research by Schwab et al. presented in section 2.1. For stop events the primary delay stop and for run events the primary delay run where used. The finding of the study suggested that an acceptable threshold of t_{node_free} and t_{edge_free} was around 25 seconds, representing a balance between error and amount of injected delays. With higher thresholds the error increased heavily and with lower ones the percentage of injected delays was quite small. The disaggregated primary delays - although based on a model and therefore only an abstraction - will be referred to in this thesis as the true primary delays. They serve as the ground truth for the PDIM. However, even with these true primary delays, it is not possible to fully reproduce the true total delays when injecting them into the simulation system.

Apart from the schedule and the historical running times, the dataset contains numerous features. In Table 3.2 all the features used in at least one of the PDIMs are listed.

Feature	Description	Value Summary
ocp_type	Indicates if the event is a stop or a pass	Either: *Stop* or *Pass*.
segment_or_stop	Either contains the ocp_id or a tuple of the previous and next ocp	Either one of 1313 ocp_id or one of 4457 segment tuples (prev_ocp, next_ocp), depending on the ocp_type.
passenger	Boolean indicating if the train is a passenger or freight train	Either: True or False
is_first_stop	If the stop is the first stop of the train	Either: *True* or *False*
category	Train category (e.g., Eurocity, Intercity, etc.)	Examples include *Eurocity*, *Intercity*, and *Regional*, in total 32 distinct values.
trainpart_speed	Speed of the train part	Ranges between 0 km/h and 230 km/h, with 21 distinct levels.
operational_type	Operational type of the OCP	Either *station*, *undefined* or *junction*
num_platform_edges	If the ocp_type is *Stop* the number of platform edges at the current OCP	Ranges between 0 and 14 with only the number 13 missing
num_siding_tracks	If the ocp_type is *Stop* the number of siding tracks at the current OCP	Ranges between 0 to 67 with 41 distinct values
distance_geo	Geographical distance in kilometer	Ranges from 5 m to 11.4 km
v_max	Maximum allowed speed on segment	Ranges from 20 km/h to 250 km/h, with 24 distinct values

Table 3.2: Summary of the features present in the dataset. The entries with a white background are from the Delay Dataset, those with a gray background are from the Infrastructure Model Dataset.

Even though the time frame of the dataset is quite small (2 days), it is a very vast dataset including 14k trains running over 4457 segments and passing or stopping at 1313 OCP generating 574k events. However, the short time frame also introduces some challenges, especially concerning the choice of the model and the interpretation of evaluation metrics.

Data Anonymization

The exact delays of trains should not be made public and are therefore anonymized for all charts in this thesis. This is achieved by applying Min-Max normalization, which is a rescaling technique, that normalizes values into a specific range, typically $[0, 1]$. Given a collection of delays denoted as a, with a minimum value of $x_{\min}$ and a maximum value of $x_{\max}$, the normalized values x' are computed as follows:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

This normalization preserves the relative differences within the data, while concealing the true range.

3.1.2 Infrastructure Model Dataset

The infrastructure model comes in the standardized XML Format RailML 2.5. RailML was introduced to improve interoperability between different infrastructure datasets and simulation and operation software [NHSK04]. The dataset contains OCP including metadata such as the operational type (station, junction, undefined) or references to the tracks connected to the OCP. For each track their is metadata describing the maximum allowed speed and the type of the track (track-running, siding, undefined) as well as to which other tracks or OCPs its connected to. The tracks and OCPs were connected in a graph-based way to calculate additional features such as the count of siding tracks or the count of platform edges at a station. Furthermore, the physical distance was calculated using a geospatial library, as this was missing in the initial dataset. In the following, connections between two OCPs will be referred to as segments instead of tracks because they could consist of multiple tracks representing one segment.

3.1.3 Data Quality & Cleaning

The simulation model had already undergone preliminary data processing as the datasets were pre-processed before integration. However, to better interpret the results, the overall data quality was carefully examined. In general, the data quality is very high. Nevertheless, occasional issues arise, such as skipped OCPs, which result in minor data inconsistencies. In addition, some entries span unusually large distances, indicating potential data errors. Since the simulation model's infrastructure is directly derived from the dataset, these anomalies do not significantly affect the model. Therefore, this thesis retains all entries, including potentially faulty ones, as the simulation model relies on the complete dataset.

Outlier detection and removal present challenges for delay values. First, there is no definitive evidence that the outliers represent data errors, particularly since large delays are plausible, especially for freight trains. However, outlier removal could still be beneficial. If one considers a scenario with n simulation runs, each involving $n \times m$ events, the probability of drawing a large value from the extreme tails is very low across all events in the experiment. Consequently, with a limited number of simulation runs, removal of large delays might improve the robustness of the results. This is why primary delays above 2 hours were removed, resulting in a removal of 0.02% of the total events. The discussion will explore both the benefits and drawbacks of this decision. Moreover, if a greater amount of training data was available or more simulation runs would be feasible, the inclusion of outliers could enhance the reliability of PDIMs in areas of extreme values.

3.2 Modelling Distributions

Before developing models for delay prediction, an appropriate target distribution from which the PDIM can sample from, must be determined. Identifying this target distribution depends on two key questions. The first question looks at which proposal distributions should be evaluated. As mentioned in Section **??**, the most prominent distributions in this field are the gamma, log-normal, and Weibull distributions. Additionally, other prominent extreme value distributions, such as the Pareto and the Generalized Extreme Value (GEV) distributions, are included in the analysis. The Pareto distribution, in particular, is evaluated due to its comparable features in capturing extreme values.

The second question concerns the kind of test that will be used to compare the distributions. The selected distribution must also adequately represent large delays, which implies that extreme values should be well captured. In addition, the overall fit of the distribution should be assessed.

As a first step, the distributions were fitted to the dataset using Maximum Likelihood Estimation (MLE) via the Python Scipy Library. The log-normal distribution in Scipy has a location parameter, in addition to the scale and shape parameters. When using MLE, the location parameter was fitted as well, even though the distribution should start at zero, which is why it was fixed to zero.

Most studies on disturbance and delay modeling have used the Kolmogorov–Smirnov (KS) test, which "is a non-parametric statistic for comparing two empirical distributions, defining the largest absolute difference between the two cumulative distribution functions (CDF) as a measure of disagreement" [LRH]. For a hypothetical CDF $F(x, \theta)$ and an empirical CDF $F_n(x)$, the KS test measures the maximum absolute difference between them:

$$\max_x \mid F_n(x) - F(x, \theta) \mid \tag{3.1}$$

The KS test is highly flexible as it is distribution-free, relying instead on an empirical

distribution. Since the p-value becomes very small with reasonably large datasets, the KS statistic itself is often used as the metric of interest.

Another test, which is more powerful for analyzing response time distributions, is the Anderson–Darling (AD) test. However, this test does not support the log-normal and gamma distributions. While it is possible to log-transform the data and then apply the AD test, this approach would render the resulting statistic incomparable.

The maximum absolute difference measured by the KS test does not account for the overall fit, which is also of interest for this task. To address this, one can use a quadratic statistic, such as the Cramér–von Mises (CVM) statistic, which measures the mean squared difference between the empirical and hypothetical CDFs [Lai04]:

$$Q^2 = n \int_{\text{allx}} [F_n(x) - F(x,\theta)]^2 \, \mathrm{d}F(x) \tag{3.2}$$

For both the KS test and the CVM test, smaller values indicate either a lower overall quadratic error or a smaller maximum difference.

Table 3.3 lists the test statistics for all proposed distributions. The Gamma distribution was excluded from further analysis due to its poor performance in both tests.

Table 3.3: KS-Statistic and CVM-Statistic for the fit on the full dataset. The bold value represents the best value in each row. In the last row, both statistics are combined. To account for their different ranges, they were first normalized using min-max scaling before being summed.

	Weibull	Lognorm	Genextreme	Pareto
KS Statistic	0.127	**0.038**	0.047	0.056
CVM Statistic	621.314	62.495	62.930	**59.425**
Combined	2.000	**0.005**	0.112	0.202

While the KS statistic for the log-normal distribution is notably low for a reasonably large and heterogeneous dataset, the CVM statistic indicates a poor fit across nearly all distributions. As noted in the background section, if no theoretical distribution provides a good fit, the data likely originates from multiple processes [Law13]. Collaborating with domain experts and analyzing the data revealed distinct event types corresponding to different processes. These processes include differentiating between freight and passenger traffic, stops and passes, as well as initial stops, which often involve significant delays.

The KS test results for each subset are presented in Table 3.4, while the CVM test results are shown in Table 3.5. According to the KS statistic, the log-normal distribution performs best in four out of six subsets. However, for the "First Stop/Freight" event, the Pareto distribution fits better and for the "Stop/Freight" event, the Generalized Extreme Value distribution achieves the best fit.

The CVM statistic paints a slightly different picture. Here, the log-normal distribution performs best for all passenger-related events. The Pareto distribution has the best fit for the freight events, with the Weibull and Pareto distributions performing almost identically in one instance.

Ideally, both statistics could be weighted again, and individual sub-PDIM models could be built for each best-performing distribution. If only one target distribution was to be chosen for a single model, the overall best-performing option would be the log-normal distribution. However, if two models were to be developed, the Pareto distribution could be used for all freight-related events, while the log-normal distribution could be applied to all passenger-related events.

Table 3.4: KS-Statistic

	Distribution	Genextreme	Lognorm	Pareto	Weibull
type	mode				
First Stop	freight	0.560	0.093	0.024	**0.019**
	passenger	0.032	**0.023**	0.026	0.080
Pass	freight	0.047	**0.034**	0.041	0.445
	passenger	0.060	**0.051**	0.071	0.496
Stop	freight	**0.124**	0.168	0.131	0.178
	passenger	0.241	**0.045**	0.056	0.464

Table 3.5: Cramer von Mises-Statistic

	Distribution	Genextreme	Lognorm	Pareto	Weibull
type	mode				
First Stop	freight	168.78	4.33	0.14	**0.13**
	passenger	1.68	**0.59**	0.90	9.04
Pass	freight	11.60	5.29	**2.13**	647.13
	passenger	48.75	**22.53**	26.30	3949.87
Stop	freight	7.27	13.18	**5.64**	36.73
	passenger	998.72	**20.69**	29.72	1600.32

3.3 Models

The goal of the Primary Delay Injection Model (PDIM) is to accurately simulate stochastically occurring primary delays. This is not merely a regression task, but rather a stochastic sampling task. As outlined in the Background section, the PDIM is a single-step, data-driven model. For each train, whether passing or stopping, a primary delay is sampled. Furthermore, it is stochastic, designed to be used statically, and has a horizon of one station ahead. However, when combined with the simulation model, it becomes

event-driven, with the ability to make both static and dynamic predictions across multiple stations ahead. Consequently, this thesis represents a combinatory approach. In a review of train delays, Spanninger et al. concluded that "hybrid methods can combine the strengths of data-driven and event-driven approaches" [STBC22]. Each PDIM samples a predefined number of primary delays for each input, generating multiple predictions that approximate the potential distribution of future delays. To ensure meaningful evaluation, these predictions must capture diverse scenarios. This diversity is crucial for the next step, where the sampled delays are injected into the simulation model to assess a wide range of realistic outcomes.

To enhance the existing machine learning approach, this thesis explores models capable of directly handling stochastic random variables. An initial analysis identified three models that can explicitly represent stochastic processes: Bayesian Neural Networks (BNNs), Gaussian Processes (GPs) and Bayesian (Hierarchical) Models (BMs). However, these models are not ideal for very large datasets. As an alternative ML-based regression models can be used to parameterize one or more distributions from which delay samples are drawn. This approach becomes especially relevant when the dataset spans months of operational data and when training time is critical.

Bayesian Neural Networks (BNNs) treat each parameter as a random variable enabling them to estimate uncertainty in predictions. Unlike traditional neural networks, which provide fixed outputs, BNNs generate a distribution of predictions offering insights into the model's confidence. [MKH18]

Gaussian Processes (GPs) are non-parametric models that employ multivariate Gaussian distributions and kernel functions to predict data with spatial or temporal correlations. They are particularly effective in tasks such as time series prediction or spatial mapping, where relationships between data points are crucial [SSK18, WYI+22].

Bayesian models provide a robust framework for constructing probabilistic representations of stochastic processes, such as delay generation. They are especially effective when strong priors - beliefs or assumptions about a parameter based on past knowledge - and domain expertise guide the modeling process. Features like hierarchical modeling and pooling, along with a variety of probability distributions, enable Bayesian models to efficiently capture complex processes. Furthermore, the learned parameters can be used to identify delay-contributing factors, allowing statistical testing on these variables. Bayesian models are typically fit using Markov Chain Monte Carlo (MCMC) methods, but these methods can struggle with large datasets. In such cases, approximate methods like Variational Inference (VI) can be employed to estimate posterior distributions, making Bayesian models more practical for large-scale applications [VDSDK+21a].

Although Gaussian processes can model stochastic variables, they are not well suited for this specific problem due to the nature of the input features. Many features, such as the segment or stop a train is passing through, are categorical rather than continuous and do not exhibit the spatial correlations required for Gaussian processes to perform effectively [WYI+22]. For instance, two train tracks might be geographically close but

differ significantly in capacity or error rates due to variations in age or condition. Here, spatial proximity does not imply shared properties. While the temporal component of the data may hold some relevance - since primary delays used in this thesis exhibit auto-regressive patterns - this property is not utilized in the current setup. Consequently, the limitations of Gaussian processes outweigh their potential benefits in this context.

Bayesian Neural Networks (BNNs), Bayesian Hierachical Models, and Bayesian Networks share a common ability to handle uncertainty and stochastic processes. However, given the inherently unpredictable and highly stochastic nature of primary delays as well as the heterogeneous dataset, deploying a large model like a BNN is unnecessary for this task.

Instead, Bayesian Hierachical Models were found to be the best inherently stochastic solution. They provide the flexibility and precision required to model complex relationships between variables while maintaining interpretability. Additionally, the resulting parameter distributions can be used for further investigation.

For larger datasets, however, deterministic machine learning models are much more efficient to train. A variety of models, such as Support Vector Machines (SVMs), Neural Networks (NNs), or tree-based methods, could be employed. Given the high number of categorical features in the dataset, the CatBoost model was chosen. CatBoost is a gradient boosting toolkit that is particularly effective for handling categorical features [DEG18].

3.3.1 CatBoost - Gradient Boosting Decision Trees

Gradient Boosted Decision Trees (GBDT) are a popular ensemble method which uses multiple decision trees trained in sequence and can therefore achieve state-of-the-art results in numerous practical tasks [DEG18]. In each iteration the ensemble learns the new tree by fitting the negative gradient which is also known as the residual errors. This is a method known as gradient boosting [KMF+17]. CatBoost is a gradient boosting library that has improved handling of categorical features and outperforms other libraries like XGBoost or LightGBM [DEG18]. As the dataset in this thesis mostly contains categorical features, a compatible library, which is also tuned towards categorical features, was chosen. The most prominent technique in machine learning to encode categorical features is one-hot encoding which is the process of replacing a categorical feature with n labels into n binary columns. High cardinality features (like, e.g., "Train ID") when one-hot encoded lead to a large number of sparse binary columns which is not desired. [PGV+18]

Another approach is to substitute the category label with a numeric value which is computed via a target statistics (TS). An approach to deal with the categorical feature i is to substitute the category x_i^k of k-th training example with a numeric feature $\hat{x}_i^k$ derived form the TS. The expected value of the target y given the category is often used:

$$x_k^i \approx \mathbb{E}(y \mid x^i = x_k^i) \tag{3.3}$$

A greedy approach would be to calculate this by taking the average of all samples. For low frequency categories the average is noise, which is why a prior p defined as the average target value of the dataset is added:

$$x_k^i = \frac{\sum_{j=1}^{n} \mathbb{1}\{x_j^i = x_k^i\} \cdot y_j + ap}{p \sum_{j=1}^{n} \mathbb{1}\{x_j^i = x_k^i\} + a} \tag{3.4}$$

Where a is a tunable parameter. The prior value is especially interesting later for new unseen data. The downside of this algorithm is that the TS is calculated using the target y which leads to target leakage. In order to fix this problem, Catboost uses a technique called ordered TS where an artificial time, realized as random permutation, is used to compute the TS based on observed history. To reduce variance, multiple permutations are used.

Catboost also utilizes combinations of categorical features to capture high-order dependencies between them. In the delay modeling case, features like the train category and segment are often related. Therefore capturing high-order dependencies is desired for the PDIM. Due to the state-of-the-art performance of Catboost on heterogenous datasets with many categorical features, this was chosen as the ML based model.

DistBoost - Parametrized Distribution Model Architecture

As Catboost is a deterministic ML-model, it first needs to be adapted in order to work as a stochastic model. In total, three different Catboost models were trained, the mu, sigma and zero model. The zero model is a binary classification model trained on the full dataset, the target y represents if a delay is either zero or not. Zero delays were modeled separately as the log-normal distribution used to sample the delays cannot represent the heavy zero inflation present in the dataset. Later the class probabilities of the binary classification were used to model zero delays.

The mu and sigma models are trained on only positive delays as the zero delays are already modeled by the zero model. As explained in the distributions section, the log-normal distribution has the best overall fit to the positive delays. The log-normal distribution has the property that if $log(X)$ is normally distributed then X is log-normally distributed.[DS12] This property is utilized by log-transforming the target. For each data point x_i the target which represents the primary delay y is now $\hat{y} = log(i)$ which makes it normally distributed. Therefore, the mu model estimates the target $\mu_i ... \mu_n$ for each data point i in the dataset:

$$\mu_i = \hat{y}_i \tag{3.5}$$

Depending on the chosen features, some events may share identical values with others, leading to identical predictions due to the model's deterministic nature. However, to simplify the training process and allow for future feature expansion, each event is assigned

its own prediction. This approach leverages the predictive capabilities of ML models. To predict the mean, the regression-based Catboost model is used. After training the mu model, the variance is calculated as such:

$$\sigma_i^2 = (\hat{y}_i - \mu_i)^2 \tag{3.6}$$

Then, the sigma model is trained to estimate the residuals which is equivalent to the standard distribution of the normal distribution.

In the prediction phase, multiple samples are generated for each event. These samples serve as an inputs for multiple simulation experiments for the respective event i. First, the class probabilities of the zero classifier are used as parameter p_i in a Bernoulli distribution and n samples are generated:

$$X_{is\_zero} \sim \text{Bernoulli}(p) \tag{3.7}$$

Then, the same amount of n samples per data point is generated by drawing from a normal distribution:

$$X_{delay} \sim \exp(\mathcal{N}(\mu, \sigma^2)) \tag{3.8}$$

The delays are then masked with zero by the zero samples. This setup allows for more flexibility in choosing a distribution as zeros are modeled separately. Further, any ML regression model could be utilized to parameterize the distributions. From here on, the parametrized distribution model trained with Catboost will be referred to as DistBoost short for Distribution Boosting.

Ensemble Model Architecture

Another approach to introduce stochasticity into a deterministic model is by constructing an ensemble of multiple models. While this method is more of a workaround than a true representation of underlying distributions, it remains a valuable technique - particularly because it leverages an ML model in its pure form. To introduce randomness, each submodel is trained with variations in the process, ensuring that each model produces different outputs. CatBoost provides multiple ways to introduce stochasticity during training, thereby enhancing diversity among models. Additionally, model-agnostic approaches such as training data sub-sampling, can also be applied.

Each CatBoost model in the ensemble is assigned a random seed, which influences various procedures during training. The most relevant of these will be briefly presented. Bootstrap sampling is a technique that assigns weights to the samples used for evaluating splits. CatBoost offers multiple sampling methods, such as Bernoulli sampling, which either includes or excludes certain samples. This approach corresponds to stochastic gradient boosting and has the advantage of accelerating computation since not all samples need to be evaluated. Since CatBoost is already a fast training algorithm, Bayesian

bootstrap is used instead. In this approach, each weight is defined as $w = a^t$, where a is independently generated as $-\log(\text{Uniform}(0, 1))$ and t is a tunable hyperparameter. A higher t leads to more aggressive bagging, resulting in a more diverse or unpredictable subset of data [Dev24]. Additionally, since CatBoost employs ordered boosting and target-based encoding, the encoding process is influenced by the order of permutations, which are also affected by the random seed.

A challenge with the ensemble architecture is that a separate model must be trained for each sample. Since 100 or more samples are often required, this leads to high computational costs. Although CatBoost models train relatively quickly, this computational burden must be carefully considered.

3.3.2 Bayesian Hierarchical Models

Bayesian statistics is an approach to statistics, in which one has background knowledge and initial beliefs about the parameters commonly known as the prior. These priors can be constructed by subject experts. If there are strong assumptions and little data, one can use an informative prior and if the assumptions are weaker, a weakly informative prior. If no assumptions are made, flat priors can be used. These parameters are then updated utilizing a likelihood function in combination with a dataset. The resulting parameter distributions are commonly referred to as the posteriors [VDSDK$^+$21a]. While usually a lot of research and discussion focuses on the prior, when sufficient data is available and flat priors are chosen, the likelihood becomes the dominant factor [VDSDK$^+$21b].

In contrast to the frequentist framework, where for each parameter a single point estimate is inferred, the Bayesian framework represents each parameter as a probability distribution [VDSDK$^+$21b]. Therefore, Bayesian models can "formulate epistemic uncertainty as a probability distribution over the model parameters" [ZW17]. These posteriors for complex models were mostly intractable. However, with Markov Chain Monte Carlo (MCMC) methods, it became possible to sample from these complex posteriors using computer simulations. As a result, the posteriors are not represented as continuous distributions, but approximated through samples drawn from the Markov Chain. It is important to note, that the samples from the Markov Chain are auto-correlated, thus dependent on the previous ones. This becomes relevant when selecting a specific subset of n samples for the simulation.

Apart from prior predictive checks, likelihood function determination and chain diagnostics, the last part in evaluating a model is posterior predictive checking. With help of the models trace, one can predict new values from the posterior predictive distribution and observe if the simulated data resembles the observed data.

Model Architecture

The Bayesian model also requires three different submodels to efficiently sample new delays. While theoretically one model would be sufficient, and could even lead to the same predictions, the task was split into three subtasks in order to improve performance.

First, a Bernoulli model was developed to predict if the delay is zero or not. The two other models were used to predict either stop or run events.

The following section presents the final model architectures. Four different model variants were developed and evaluated: the first, referred to as the baseline model, is a simple model with few parameters, while the second is a more advanced variant. First, the baseline variants will be described. The baseline zero model samples a Bernoulli event which has one parameter p representing the probability of a zero event. All binary delays are considered to obtain the posterior of this submodel:

$$\text{is\_zero} \sim \text{Bernoulli}(p) \quad \text{where} \quad p \sim \text{Uniform}(0, 1) \tag{3.9}$$

$$\text{is\_zero} = \begin{cases} 1, & \text{if delay} = 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

For the stop and the run model we have the same baseline architecture. They are both models with a log-normal likelihood, the parameter μ representing the mean of the log-normal and the parameter σ^2 representing the variance. This results in two different posteriors for μ and σ: one for the stop model and one for the run model. Since both parameters, mu and sigma, in the log-normal must be strictly positive for the priors, a half normal distribution with sigma equal to five was chosen. This can be considered a relatively flat prior:

$$\mu \sim \text{HalfNormal}(5) \tag{3.11}$$

$$\sigma^2 \sim \text{HalfNormal}(5) \tag{3.12}$$

$$\mathit{delay} \sim \text{LogNormal}(\mu, \sigma^2) \tag{3.13}$$

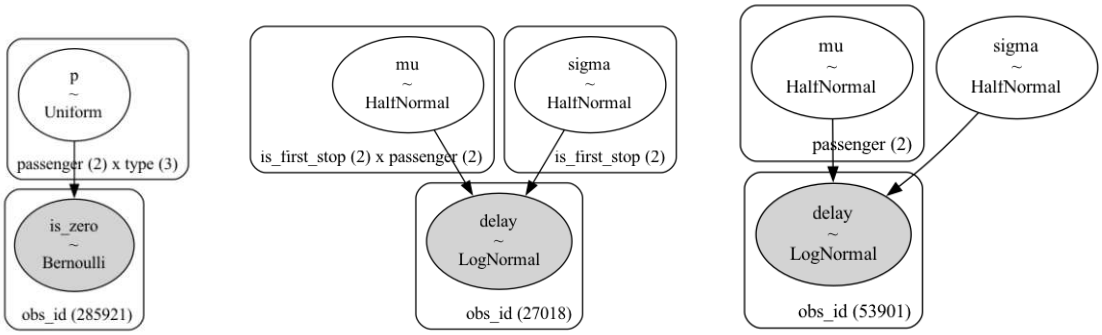
The resulting baseline model has therefore only five parameters. New samples can be generated from the model as follows:

$$\text{is\_zero} \sim \text{Bernoulli}(p) \tag{3.14}$$

$$\text{delay} = \begin{cases} 0, & \text{if is\_zero} \\ \text{LogNormal}(\mu_{\text{stop}}, \sigma^2_{\text{stop}}), & \text{if } \neg\text{is\_zero} \wedge \text{event} = \text{stop} \\ \text{LogNormal}(\mu_{\text{run}}, \sigma^2_{\text{run}}), & \text{if } \neg\text{is\_zero} \wedge \text{event} = \text{run} \end{cases} \tag{3.15}$$

Apart from the baseline models, three additional variants were developed for each submodel, referred to as advanced-1, advanced-2 and advanced-3. The advanced-1 model is visualized in 3.1. In this variant, the dimensions of the parameters have changed. For

example, the p parameter now has the shape (passenger, type), resulting in six different parameters for p, which model the Bernoulli distribution.

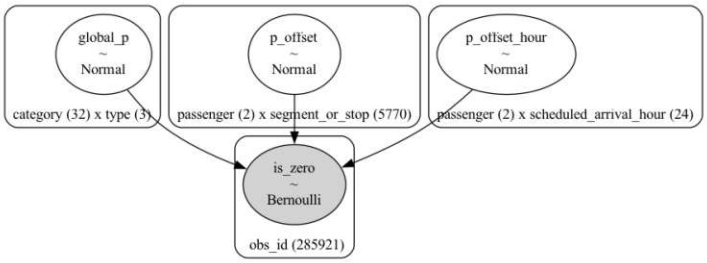


(a) Zero Model (b) Stop Model (c) Run Model

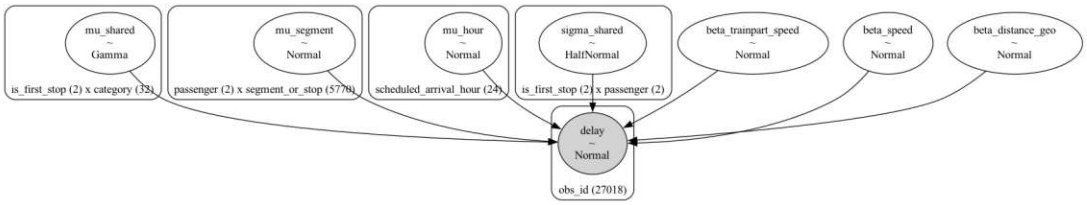
Figure 3.1: Visualization of the advanced-1 model. Each bubble represents a random variable, labeled with its name and specified distribution. The surrounding square indicates the shape of the random variable. An arrow indicates that the variable is used as a parameter of another distribution or if the target bubble is light gray, that it is part of the likelihood function. In the case of the zero model, we have a parameter p that is uniformly distributed with a shape of (passenger (2), type (3)). This variable serves as a parameter p of a Bernoulli distribution, which models the likelihood of an delay being zero.

In the advanced-3 model family, the dimensionality of each parameter was further expanded. The shared mu across all segments and stops now incorporates the detailed feature "category" instead of just differentiating between passenger and freight. Additionally, for each segment or stop, a normally distributed offset μ_{segment} is learned, which is then added to μ_{shared}. The same was done for μ_{hour} which represents an learnable offset for each hour of the day. These additions can result in negative values for the overall μ, which would break the model, as the μ of the log-normal distribution must be positive. One possible approach to ensuring positivity was to take the exponential of the result. However, this significantly slowed down sampling. To address this issue, the data was log-transformed, allowing it to be modeled as a normal distribution which allows negative values as μ. As shown in Figure 3.2, the μ_{segment} of the advanced-3 variant follows a normal distribution, meaning that the sum $\mu_{\text{offset}} + \mu_{\text{segment}}$ can be negative.

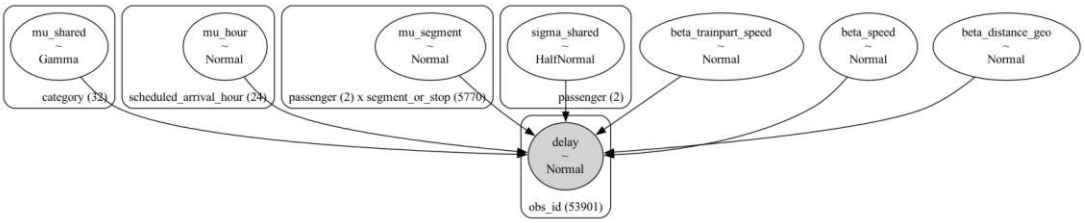
In addition to the segment offset, a regression-based component was implemented, where numerical features such as *trainpart_speed* are multiplied by a parameter such as $\beta_{trainpart\_speed}$. In the model visualizations, only the dimensions of each parameter are shown, but not how they are combined into the input parameters of the likelihood function. Therefore, the final formula for calculating μ in the advanced-3 stop model is presented separately:



(a) Zero Model



(b) Stop Model



(c) Run Model

Figure 3.2: Visualization of the advanced-3 models. From this visualization the function how the different parameters are combined is not visible please refer to Equation 3.16.

$$\begin{aligned}\mu = \mu_{\text{shared}} + \mu_{\text{segment}} + \mu_{\text{hour}} \\ + \beta_{\text{distance\_geo}} \cdot \text{distance\_geo} \\ + \beta_{\text{trainpart\_speed}} \cdot \text{trainpart\_speed} \\ + \beta_{\text{speed}} \cdot \text{speed}\end{aligned} \tag{3.16}$$

The advanced-2 variant is quite similar to the advanced-3 variant but missing the hourly offset (mu_hour) in all submodels, and for the stop model does not contain the regression components. Due to the large size of the model visualizations, this variant can be found in the Appendix A.

Bayesian models cannot inherently handle missing values. Therefore, these missing values were processed using a custom logic. The maximum allowed speed in each segment was filled with the median value across all trains and the same approach was applied to the train part speed. The values for the distance between two OCPs, the number of platform

edges, and the number of siding tracks were set to zero, as these variables are used in the regression component, where a zero value has no impact on the predictions.

3.4 Simulation Model

The simulation model used in this thesis is the simulator RailwaySim presented in [RWJ+20]. As explained in Section 2.4, it is an agent-based macroscopic simulation tool. Each of the n generated primary delay samples from each model will be injected into the simulation model. The model tracks multiple values for each event. The total delay represents the absolute deviation from the simulated schedule for a given train at a specific event after the event has been simulated. In contrast, the initial delay refers to the absolute deviation before the event has been simulated. The primary metric logged will be referred to as the secondary delay which is defined by:

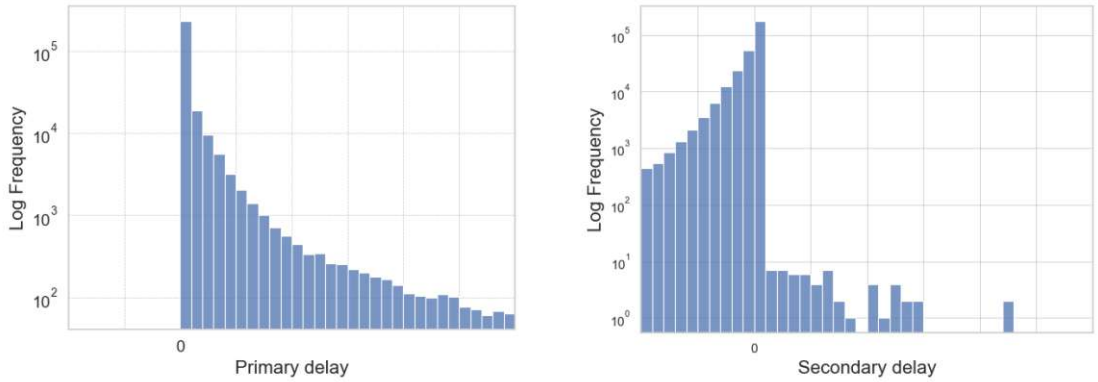
$$secondary\_delay = total\_delay - delay\_initial - primary\_delay$$

With this metric, the primary delay samples from the total additional delay can be easily recreated by adding the primary delays. As the simulator sometimes did not terminate with very large primary delays, samples above 12 hours were cut off.

In order to compare models with the simulator, it is first necessary to understand how the simulator works and secondly, how it responds to delay inputs. Therefore, first only zero delays or the true primary delays were injected and analyzed. In Figure 3.3 the distribution of the true primary delays and the remaining secondary delays are plotted. The secondary delays are calculated by subtracting the primary delay from the additional delay. The additional delay is the deviation of the actual data from the scheduled data. Figure 3.3 clearly shows that, with the current configuration of the simulation system, most delays are primary delays.

To understand how these delays affect the simulation system, delays were injected in stages before utilizing the PDIMs: first, zero delays and then the actual primary delays the model was trained on. In Figure 3.4 the distribution of simulated secondary delays created by these two configurations are shown. Only very little positive secondary delays arise with both configurations. A secondary delay is the total delay subtracted by the initial and primary delay. A negative secondary delay thus explains that the trains in the simulator were able to compensate for the primary delay injected at that step. Consequently, most of the delays observed within the simulation system are actually primary delays.

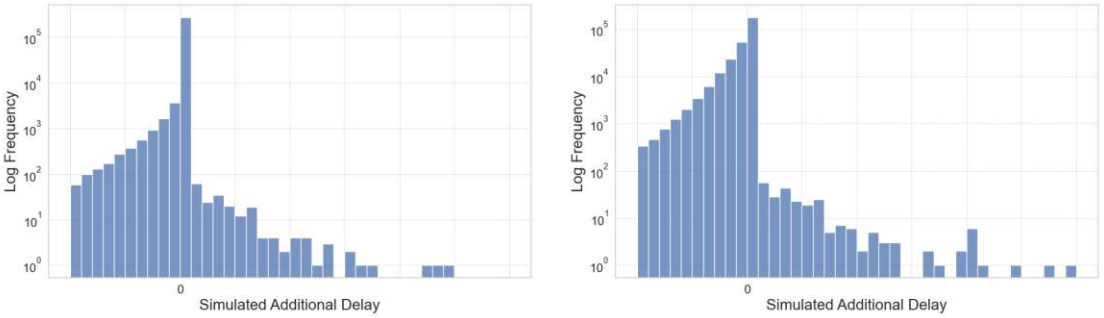
As the primary delays are subtracted from the additional delay to compute the secondary delays, they remain present in the secondary delay distribution. For example, if the additional delay is zero and a primary delay of 10 is deducted, the resulting secondary delay is -10. Consequently, analyzing the full distribution of additional delays leads to a re-evaluation of the injected primary delays.



(a) True Primary Delays (b) True Secondary Delays

Figure 3.3: Histogram of the true primary delays and true secondary delays. Most of the additional delays are in fact negative and therefore also the true secondary delays are mostly negative. The data has larger ranges but to increase visibility outliers are not shown in these charts. Furthermore, the bin size as well as the ticks were hidden to provide data anonymity. In Figure (b) it becomes apparent that only very few positive secondary delays exist.

Conversely, validating the model using only positive secondary delays results in a significant loss of information, as illustrated in Figure 3.4. Therefore, the analysis will consider both positive and negative secondary delays produced by the simulator. The key question is which model can inject a similar amount of delays while still yielding a comparable distribution of secondary delays.



(a) Zero Primary Delays Injected (b) True Primary Delays Injected

Figure 3.4: Comparison of the simulated secondary delays with the true delays or the zero delays injected.

3.5 Validation

Validation will be divided into two approaches: Theoretical Performance and Simulation-Based Validation. Theoretical Performance involves testing and validating metrics that assess how well the sampled primary delays correspond to the true delays. In contrast, Simulation-Based Validation injects the sampled primary delays into the simulation system and compares the resulting secondary delays either to the true secondary delays or to those generated by injecting the true primary delays instead of the sampled ones.

3.5.1 Theoretical Model Performance

In order to evaluate and compare the models, appropriate evaluation metrics must be defined. These metrics should not only capture predictive accuracy, but also measure the diversity of the generated samples, thereby reflecting the stochastic capabilities of the models. Lastly, we will look at the overall fit of the sample distribution compared to the true distribution. Other metrics like runtime are also relevant.

Error Metrics

For each dataset with features x and true delays y, where y consists of delays $y_1, y_2, \ldots, y_i$, the aim is to predict the delay $\hat{y}$.

The most commonly used metrics in regression tasks are the Root Mean Squared Error (RMSE):

$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{N}} \tag{3.17}$$

and the Mean Absolute Error (MAE):

$$\mathrm{MAE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1}|y_i - \hat{y}_i|}{N}. \tag{3.18}$$

RMSE places greater emphasis on larger errors due to the squaring operation, whereas MAE treats all errors equally. Since our models should prioritize accurate predictions, particularly for extreme values, a custom error metric is required that assigns higher weights to larger true values. This can be achieved through a custom weighted error.

The weight w_i can be set to the true value y_i, making errors relative to the prediction target. Given the large range of values, an alternative is to set w_i to $\log(y_i + 1)$, where the additional term ensures that zero delays do not result in undefined logarithmic values. The weighted MAE is then defined as:

$$\mathrm{MAE}_{\mathrm{weighted}}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} w_i|y_i - \hat{y}_i|}{N}. \tag{3.19}$$

Other approaches, such as evaluating zero-delay predictions separately, computing a quantile error or measuring the number of exact predictions within a certain tolerance, were also considered. However, these methods were found to be less suitable for the task compared to the weighted MAE.

Evaluating Sample Diversity

Apart from having a low error on the test data, the model should create diverse predictions. The first approach was to understand the spread of the error metrics across multiple sample datasets. This is further defined as the span of a certain error metric, for example, the RMSE-Span is defined as deviation between the largest and lowest RMSE between the different sample datasets generated by one model. This is an important metric, especially to understand the performance of the best sample. However, there can be high prediction diversity, although the error metrics are similar across datasets. Therefore, the pairwise distances were calculated between all of the sample datasets.

Suppose there are n samples in a prediction dataset, where each sample represents a delay profile observed over a full day of train traffic. Let each sample be represented as a vector, with each element of the vector corresponding to the delay observed at a specific train event.

The distance between two sample vectors u and v, representing two different delay profiles, can be calculated using a distance function, denoted $d(u, v)$. This distance function quantifies the dissimilarity between the two delay profiles by comparing their individual delay values across the stops.

In the context of the delay dataset, two common distance metrics, Euclidean distance (L2 Norm) and Cityblock distance (L1 Norm) will be used to compute the pairwise diversity between delay profiles. These norms have similar properties as the RMSE (L2) and MAE (L1) and therefore align with our other metrics. The L2 puts a focus on the outlier deviation between the samples while the L1 puts an equal emphasis on each pair. Concluding, L2 is well suited for measuring how diverse sampled outliers are while L1 is more appropriate for detecting whether there are overall large differences between the samples. The distance across all samples should be maximized to achieve high sample diversity.

Distributional Analysis

So far, the metrics which define the fit according to a point-wise error and the diversity of prediction were presented. These metrics do not capture information about the overall distribution of delays which is crucial for proper delay sampling.

The quality of the chosen distributions is assessed by determining how similar the distribution of primary delay samples is compared to the empirical true distribution. Further, it is of interest to evaluate the fit, if the dataset is subsetted along multiple axes such as hour of day, segment or event type. To compare these distributions, the

statistics presented in Section 3.2 will be reused, more specifically the KS-Statistic and the Cramér-von-Mises Statistic.

3.5.2 Model Performance Simulation Loop

Performance metrics, such as RMSE, provide insight into the theoretical performance of the model. However, due to the stochastic nature of the processes involved, achieving a low RMSE value is not always feasible. The primary objective of the model is its integration into the simulation model. Consequently, model validation should be performed using the delays generated by the simulation model when primary delays are introduced.

Various metrics can be used to evaluate the model. For passenger and freight traffic the amount of acceptable delay differs significantly. There are two types of metrics: one measures how well the simulated delays (created by adding PDIM delays to the model) match real-world operations, and the other one evaluates how accurately the model reflects key factors important to railway companies and passengers. For the former, delays at the most granular level, specifically at each operational control point (OCP), are crucial. For the latter, passenger train delays are evaluated at stops, while for freight trains, the total delay is considered. For passengers, the location of the delay is often irrelevant, as only the overall impact matters.

Ability to Reproduce Real-World Data

Given a test dataset, the simulated delays are considered accurate if they reproduce the same additional delay at each OCP. This can be evaluated using the true data from the test set. However, the challenge lies in determining whether the error originates from the PDIM or the simulation model. Therefore, the primary comparison will be between the delays generated by the simulation model using the true primary delays, which also serve as input for PDIM inserted into the simulation. This approach helps eliminate simulation-related errors, allowing for a clearer interpretation of the results. Ideally, the delays produced in this configuration would match the true delays, but this is not the case due to the inherent margin of error in the simulation model.

However, even if the model achieves this level of precision by achieving a very low error, it may indicate overfitting, as each simulation day contains a specific sample of primary delays. A model capable of exactly reproducing every delay would fail to generate diverse scenarios, which are essential for meaningful simulations that run multiple times.

An alternative perspective is to evaluate whether the model achieves a low error compared to actual data in one or more simulation runs. If this occurs, it could indicate that the model successfully simulated diverse scenarios and in some cases, the scenario closely resembled reality. The issue with this approach is that the solution space due to its stochastic nature is possibly quite large which can lead to limitations given the small number of samples. To further evaluate the model, the overall distribution of observed and simulated delays can be compared, providing a more comprehensive assessment.

CHAPTER 4

Results

4.1 Training

The models were trained on a M1 Macbook Pro equipped with a 10-core CPU (8 performance cores and 2 efficiency cores) and 16GB of RAM. For all models the dataset was split into a training and a test set, while for the ML based models the dataset was further divided including a validation set to implement early stopping. When training models a training test split such as 80/20 is common. Though for the evaluation in combination with the simulation model a full day of operations is needed. Therefore the data was split into a training day (December 14. 2022) and a test day (December 15. 2022). For the ML based models the training set was the further split by a 80/20 ratio into a train and validation set.

The number of samples predicted for each event is a trade-off between evaluation runtime and increased robustness from additional experiments. Ideally, this number should be as high as possible while still allowing all experiments to be simulated within the given time frame. For this thesis, each model was configured to generate 300 primary delay sample datasets for the test day.

Below, the training process of the models, including the final model choice and hyperparameter tuning, will be elaborated on.

4.1.1 Catboost - Parametrized Distribution Model

The model design of the DistBoost explained in section 3.3.1 was implemented using the CatBoost Python package, which supports both regression and classification for the submodels [Tea25]. To optimize the hyperparameters of the CatBoost model, Optuna, an open-source hyperparameter optimization framework, was used. Optuna efficiently tunes the model by exploring a defined search space of hyperparameters, which in this case included:

- **Learning rate** (*learning_rate*): A float value between 1e-3 and 0.1, optimized on a logarithmic scale to account for wide variations.
- **Depth** (*depth*): An integer value between 1 and 10, controlling the depth of the trees.
- **Subsample** (*subsample*): A float value between 0.05 and 1.0, determining the fraction of samples used to build each tree.
- **Colsample by level** (*colsample_bylevel*): A float value between 0.05 and 1.0, specifying the fraction of features to sample at each level of the tree.
- **Minimum data in leaf** (*min_data_in_leaf*): An integer value between 1 and 100, controlling the minimum number of samples required in a leaf node.

The number of iterations (trees) was fixed at 2000, as the learning rate and number of iterations are tightly coupled and runtime is not a concern for this model. Since theoretical performance is not the sole evaluation method, it is of interest how different loss functions affect other metrics, two final models with optimized parameters but different loss functions (RMSE, MAE) were trained.

Table 4.1 lists the best parameters for each submodel. It shows that no boundaries of the search space were reached, except in the MAE-zero variant, where the maximum depth of the search space was attained. This indicates that the initial starting points for hyperparameter optimization were satisfactory. Furthermore, the table highlights the significant differences in optimal hyperparameters not only within the submodels, but also between the different loss functions, which suggests that hyperparameter tuning was beneficial.

Figure 4.1 presents the objective value for each optimization iteration. It illustrates that in most cases the optimal hyperparameters were found within ten iterations. Therefore, fewer hyperparameter iterations could be sufficient in future studies.

Table 4.1: Best parameters after optuna optimization.

			MAE			RMSE
	mu	sigma	zero	mu	sigma	zero
Hyper Parameter						
learning_rate	0.05	0.02	0.01	0.07	0.09	0.08
depth	8.00	9.00	10.00	6.00	4.00	8.00
subsample	0.78	0.77	0.67	0.60	0.95	0.71
colsample_bylevel	0.97	0.67	0.75	1.00	0.13	0.75
min_data_in_leaf	44.00	89.00	65.00	62.00	58.00	96.00

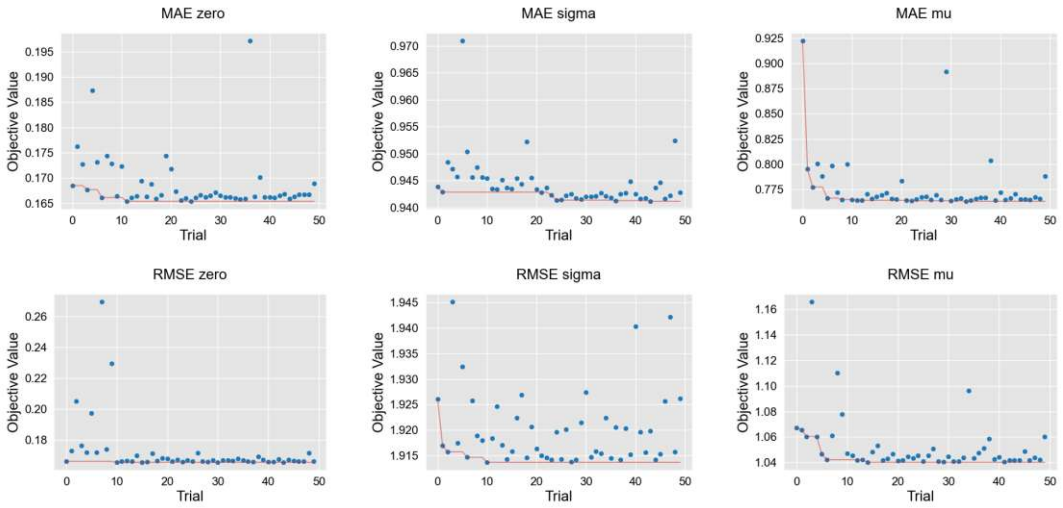


Figure 4.1: Optuna catboost study results. Each dot represents the objective value for each iteration.

4.1.2 Catboost - Ensemble Model

The CatBoost ensemble model, as demonstrated later, exhibits low pointwise errors in its most basic setup. However, the inter-sample diversity is relatively low. Inter-sample diversity measures how much variation there is within the multiple sample datasets on a pointwise basis. A detailed introduction to this metric is provided in Section 4.2.2.

Before improving inter-sample diversity, it is essential to review the approaches used to measure it. The error metric span and the mean inter-sample distances, calculated using either the City Block or Euclidean distance metrics, will be used for this assessment. These metrics will serve as the basis for comparing different strategies to increase or decrease the diversity of ensemble model predictions.

Rotating the seeds of the CatBoost model only led to a small divergence within the ensemble. Therefore, a grid search experiment was conducted to evaluate which parameters had an influence on the variability of learners in terms of RMSE. The grid search tunes the hyperparameters of a CatBoost model by iterating over various loss functions ("RMSE" and "MAE"), bagging modes ("Bayesian" and "Bernoulli") and bagging temperatures in the Bayesian case (0, 1, 10) or subsampling in the Bernoulli case [0.3, 0.7, 1] to identify the optimal combination for the model. This led to only very small divergences between the generated samples. Therefore, training the models on fractions of the training dataset (0.3, 0.7, and 1) was also added to the grid search exploration space.

The large grid search conducted that the loss function and the subsampling of the training data has the largest effect while the bagging modes only have little effect. This could be due to the fact that with 1000 iterations and many splits, every data point is visited. Details on this can be found in the Appendix B.

Therefore, Table 4.2 shows only the grid search over the two proposed loss functions and subsampling fractions for the training data. To get a general overview over the performance, all four diversity metrics are included in the table, as well as the Mean Weighted MAE. The first observation is that the RMSE loss function leads to a higher WMAE. This is because RMSE penalizes large errors and favors larger predictions, while the model trained with the MAE loss function produces more conservative predictions. Later, this will be analyzed in detail in the theoretical results section. As expected, the subsampling consistently leads to higher pointwise errors, while it generally increases diversity in terms of the distance metric. Though this trend is less clear than the decrease in performance. This raises the question of how the different diversity metrics relate. The Euclidean and City Block distances as well as the MAE-Span have a high correlation and therefore lead to the same conclusions. Only the RMSE-span has a different trend. The goal of this analysis is to decide on the parameters for the CatBoost ensemble model such that there is an acceptable trade-off between low error and high diversity. For the RMSE-loss-based variant, a subsample value of 0.4 was chosen due to a reasonable trade-off between error and diversity. For the MAE-loss-based variant also 0.4 was chosen to ensure consistency, even though 0.6 subsample rated had more sample diversity.

Table 4.2: Subset of the full grid search for the CatBoost ensemble model. The Table is sorted by the Euclidean distance.

Loss	Subsample	Mean-WMAE	Euclidean Distance	Cityblock Distance	MAE-Span	RMSE-Span
RMSE	0.2	84.06	12355	1234868	2.42	3.67
	0.4	82.23	9486	861122	1.11	1.73
	0.6	81.68	7681	678555	0.96	1.89
	0.8	81.40	5366	556722	1.23	1.16
	1.0	80.99	3705	333868	0.58	0.73
MAE	0.2	101.19	2969	301168	0.12	1.78
	0.6	101.05	2390	207340	0.15	2.98
	0.4	101.31	2058	232882	0.13	1.14
	0.8	101.30	1716	167237	0.15	1.20
	1.0	101.06	1609	157057	0.13	2.28

4.1.3 Bayesian Models

To train the Bayesian models, the probabilistic programming library PyMC [APAC+23] was used. The library provides a Python API to build Bayesian models. It includes state-of-the-art inference algorithms including MCMC based No-U-Turn Sampler (NUTS) and Variational Inference (VI). Due to the limited model complexity, VI was not used in this thesis, although training time could have been improved.

All four Bayesian models proposed in the method section were trained. These model variants were tuned for 2000 iterations, followed by drawing 1000 samples. Four chains were sampled in parallel. Training of MCMC-based Bayesian models requires careful checking of the sampling procedure. For this purpose, there are three main statistics to check: Divergences, Effective Sample Size (ESS) and R-Hat statistic.

Often, wrong priors or too few tuning iterations lead to the model having divergences in the final sampling procedure. Divergences are drawn samples, where the sampler was not able to make a valid move across the posterior distribution. Therefore, the resulting points are possibly not representative samples from the posterior. [Dev25] During the model design phase, divergences often occurred while sampling, but with proper calibration of priors and the correct number of tuning steps these divergences were nearly eliminated. Only two were remaining in the stop submodel of the advanced-3 family.

The next thing to check is the Effective Sample Size (ESS). In Bayesian inference, MCMC algorithms generate samples that are correlated. This is due to the nature of MCMC chains where each sample is dependent on the previous one. If the samples auto-correlate too much, the chains need to be prolonged to reach a high ESS. [GCS+13] This value

should be as high as possible, although in most cases values above 1000 are sufficient. [Bü17]

Lastly we need to check the R-hat value, which provides information on the convergence of the chains. If the R-hat value is considerably greater than 1, precisely larger than 1.01, the chains have not yet converged properly. [GR92]

In Table 4.3 the statistics of the MCMC sampling procedure can be found. First of all, it can be observed that model complexity varies heavily, ranging from models with only five parameters to models with 35k parameters.

Each parameter in the model is more complex than in traditional machine learning, as it consists of a trace representing the parameter's posterior distribution. To assess model performance, we calculated the mean of the ESS metric across all parameters within a model family, as well as the number of parameters with an ESS below 1,000. Since each PyMC model comprises three submodels - run, pass, and zero - the results are aggregated across all submodels. In the table, mean values (e.g., for the ESS metric) represent averages across all parameters from these three submodels.

For the advanced-2 models, only a few parameters exhibit R-hat and ESS values that deviate from the desired thresholds. However, in the advanced-3 models, more than 100 parameters show lower ESS and higher R-hat values than expected. This discrepancy is likely due to the hierarchical structure of the model, which may be less reliable for categories with low sample sizes.

Table 4.3: Training Statistics for the Bayesian Models.

	#Parameters	Mean ESS	Min Ess	#R-hat > 1.01	ESS < 1000
baseline	5	7356	3711	0	0
advanced-1	15	12474	10772	0	0
advanced-2	29051	18181	128	22	23
advanced-3	34920	15648	230	127	204

4.1.4 Training & Inference Runtimes

Before evaluating the models' performances and their predictive and stochastic capabilities, the training and inference times will be analyzed.

Bayesian

First, the training and inference times for each Bayesian variant and submodel will be presented, as their durations are of particular interest. As shown in Table 4.4, a significant portion of the training time is spent on the Zero model, which separately models zero delays. This is especially true for more advanced hierarchical models. In these cases, a sigmoid function must be applied after combining the different parameters to ensure that p remains within the valid range [0, 1], which likely slows down the sampler. Apart

from that, the simpler models train very quickly, whereas the more complex ones require significantly more time.

Table 4.4: Training times in seconds for the Bayesian models. Training consists of 2000 tuning steps and 1000 sample steps across 4 chains.

	Zero	Stop	Run	Total Time
baseline	7s	3s	5s	15s
advanced-1	148s	15s	10s	173s
advanced-2	1709s	182s	337s	2228s
advanced-3	2228s	853s	503s	3584s

Inference, in this case more commonly referred to as posterior predictive sampling, is relatively fast for almost all models, ranging from approximately one to three minutes, depending on model complexity, as shown in Table 4.5.

Table 4.5: Posterior predictive sampling duration for the Bayesian models.

	Stop	Zero	Run	Total Time
baseline	8s	30s	30s	68s
advanced-1	13s	57s	31s	101s
advanced-2	15s	114s	41s	170s
advanced-3	18s	122s	53s	193s

DistBoost

Before analyzing the runtime of the CatBoost model, which parameterizes the log-normal distribution, the runtime of the hyperparameter optimization is examined. A total of 50 trials were conducted, each DistBoost model was trained for 2000 iterations to determine the optimal hyperparameters. Optimizing the CatBoost model with RMSE loss took 1 hour and 54 minutes, while the MAE version required 3 hours and 3 minutes. As observed in the optimization analysis, fewer trials would have been sufficient, and the optimization process does not need to be rerun for every new dataset.

CatBoost is a highly optimized ML library, therefore training and inference are very fast, the training of the model takes only 170 seconds and inference under 3 seconds. The detailed benchmarks can be found in Table 4.6. Further, it is very fast to generate new samples by drawing from the predefined distributions with the previously predicted parameters.

Ensemble

Lastly, the training times for the CatBoost ensemble model are presented. The hyperparameters of the CatBoost ensemble model were not optimized, instead, the case study in

Table 4.6: Train and predicting times for the catboost MAE model in seconds.

	Zero	Mu	Sigma	Total Time
train	119s	26s	25s	170s
pred	1s	<1s	<1s	<3s

Section 4.1.2 determined the hyperparameters that ensure a balanced trade-off between error performance and prediction diversity. These models have the longest training times, since a dedicated model needs to be trained for each sample dataset. Due to these prolonged training times, the number of iterations was limited to 1000, as observations showed that the best model is usually found within the first 1000 iterations. Training and prediction for the MAE variant took 92 minutes, while the RMSE variant required 67 minutes.

The parameterized CatBoost model is by far the fastest to train and scales best for larger datasets. In contrast, the CatBoost ensemble model becomes particularly problematic when more samples are needed as its training time increases linearly with the number of desired samples. Bayesian models can be fast to fit when they are simple, but more complex ones require significant tuning and are susceptible to poor model definitions, leading to slow convergence. While the CatBoost ensemble model scales poorly with sample size, Bayesian models scale poorly with dataset size, as each sampling step requires evaluating the likelihood over the entire dataset. In contrast, the parameterized CatBoost model can generate an infinite number of samples per input at a low computational cost, making it the most efficient model in terms of scalability and performance.

4.2 Theoretical Performance

Theoretical performance refers to all metrics that can be evaluated without incorporation of the delays into the simulator. As described in the Method Section 3.5, these metrics include error metrics, evaluation of prediction diversity and the overall distribution error.

4.2.1 Error Metrics

In this section, the results based on classical regression and classification error metrics are presented. What differentiates this section from a standard machine learning model results analysis is that, instead of a single e.g. mean error metric, multiple mean errors are calculated for each sample dataset. Since 300 samples were generated for each model, this results in 300 error metrics for different samples. Therefore, the metrics must be aggregated. The focus will be on reporting the mean, the best value and the span, which is defined as the absolute difference between the best and worst predictions within the samples.

Zero Delay Prediction

Before discussing the MAE, RMSE, and Weighted MAE, an evaluation of zero and non-zero predictions is conducted. In the Bayesian and parameterized CatBoost models, zeros are modeled separately, whereas the CatBoost ensemble is purely regression-based. Since the dataset consists predominantly of zero delays, it is important to analyze these predictions separately. Therefore, classification metrics are used. Accuracy is not a suitable metric, as simply predicting zeros would result in a high accuracy score. Instead, the recall, precision, and macro averaged F1-score are examined separately.

Figure 4.2 presents the confusion matrices for the models. For the ensemble models, which do not model zero delays separately, it is evident that the loss function has a significant impact on the range of predicted delays. When using the RMSE loss function, almost all predicted delays are strictly positive, leading to an imbalanced prediction and resulting in the worst F1-score (Table 4.7). In contrast, for the ensemble model, the MAE loss function yields the best performance for the classification task. Furthermore, the Bayesian model performs worse than the DistBoost models with the advanced variant outperforming the baseline variant. The DistBoost model shows slightly worse performance than the ensemble. Only one of these models is displayed, as the classification CatBoost model is trained on the same loss function.

Table 4.7: Macro averaged F1 Scores for the classification task.

	Mean Macro avg. F1
Catboost Ensemble MAE	0.76
DistBoost MAE	0.72
DistBoost RMSE	0.72
PYMC advanced-3	0.64
PYMC advanced-2	0.64
PYMC advanced-1	0.51
PYMC baseline	0.50
Catboost Ensemble RMSE	0.37

These results may seem counterintuitive, as the regression-based ensemble model outperforms the classification-based component in the classification task. This could be explained by the fact that class probabilities, rather than direct classifications, are used to sample zero delays in the DistBoost models. In conclusion, the quality of zero-delay modeling varies significantly. However, only the RMSE-based ensemble CatBoost model exhibits performance that is not acceptable.

Regression Based Error Metrics

Apart from achieving low error, it is also essential that the model does not underestimate the total amount of delays, as this would artificially improve error metrics. Therefore, we will first present the total amount of injected delays divided by the test set delays.

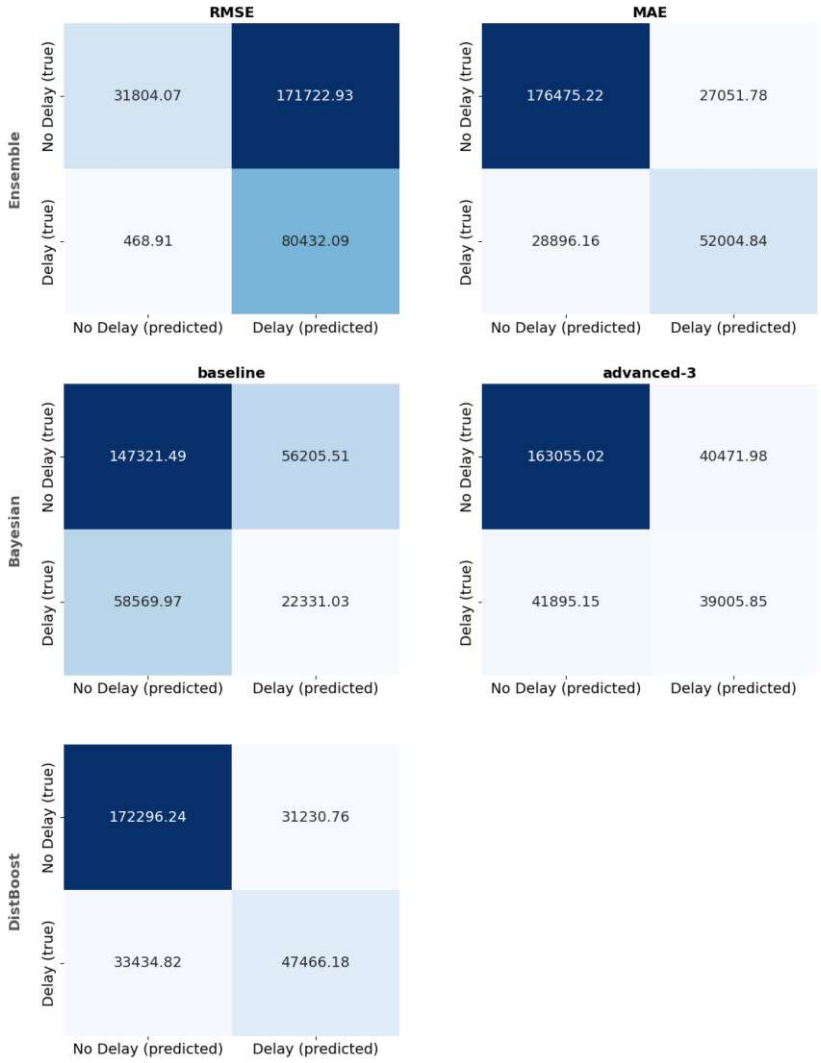


Figure 4.2: Confusion matrices for the delay/no delay prediction. The sum of values in each bin is averaged across all samples. Due to space constraints, the Bayesian "advanced-1" and "advanced-2" variants are not included.

Table 4.8 shows the evidence that both the Bayesian baseline and the ML-MAE-based models underpredict the target. In particular, the ensemble captures only 13% of the total delays, which is clearly too low. In contrast, the ML-RMSE-based models and the more advanced Bayesian models accurately represent the total amount of delays.

The easiest error metric to interpret is the Mean Absolute Error (MAE), where the value represents the absolute difference of each prediction. The CatBoost ensemble, as well as the parameterized CatBoost model, performs best on this benchmark (Table 4.9),

Table 4.8: Total amount of injected delays relative to the test set with the mean, min and max value across all 300 samples.

Model	mean	min	max
Bayesian advanced-2	1.15	0.82	18.71
Bayesian advanced-3	1.14	0.85	18.83
DistBoost RMSE	1.00	0.91	1.15
Ensemble RMSE	0.96	0.91	1.00
Bayesian advanced-1	0.78	0.70	0.97
DistBoost MAE	0.70	0.67	0.73
Bayesian baseline	0.67	0.64	0.70
Ensemble MAE	0.13	0.11	0.16

particularly when the loss and evaluation metric is MAE. This outcome is expected, as the model optimizes directly for this value. In terms of MAE, the Bayesian models perform the worst, especially the more complex ones. This may be due to the fact that, with fewer data points per parameter, the results become less reliable compared to models that utilize only a few parameters for each prediction.

Table 4.9: Result based on Mean Absolute Error

Model	Mean MAE	Best MAE	MAE Span
Ensemble MAE	13.69	13.62	0.18
DistBoost MAE	16.46	16.09	0.77
Ensemble RMSE	17.79	17.29	1.54
DistBoost RMSE	20.88	19.66	3.23
Bayesian baseline	22.17	21.66	0.94
Bayesian advanced-1	23.62	22.50	3.81
Bayesian advanced-2	27.95	23.32	254.02
Bayesian advanced-3	27.78	23.61	255.33

The RMSE results are presented in Table 4.10. The Bayesian models - particularly the simpler ones - perform better in terms of RMSE compared to their performance with MAE. The WMAE results 4.11 exhibit a similar performance pattern across the models and do not lead to different conclusions. All tables are consistently sorted by the best error.

In stochastic models, certain datasets may yield predictions that differ significantly from the target dataset, as each sample represents only a single realization of a model of the real process. The span provides insight into the variability of model performance. The highest variability is observed in the Bayesian models, followed by DistBoost, whereas

the ensemble models exhibit very low variability. In general, models optimized with an RMSE loss function display greater variability than those trained with MAE. This is expected, as previously demonstrated, since MAE-based models primarily predict small delays.

Table 4.10: Result based on Root Mean Squared Error.

Model	Mean RMSE	Best RMSE	RMSE Span
Ensemble RMSE	121.60	120.93	1.80
Ensemble MAE	137.47	134.79	3.68
DistBoost MAE	143.44	135.32	34.24
Bayesian baseline	153.25	147.83	29.08
Bayesian advanced-1	281.93	183.90	923.50
DistBoost RMSE	297.36	192.39	704.66
Bayesian advanced-3	1222.38	241.09	116770.39
Bayesian advanced-2	1302.60	260.08	69484.93

Table 4.11: Result based on Weighted Mean Absolute Error with the weight based on the log delay of the target.

Model	Mean WMAE	Best WMAE	WMAE Span
Ensemble RMSE	82.22	81.53	1.77
DistBoost MAE	91.69	89.92	4.08
Ensemble MAE	101.18	99.48	2.85
DistBoost RMSE	107.73	101.77	20.46
Bayesian baseline	108.02	107.49	1.31
Bayesian advanced-1	109.38	107.70	6.58
Bayesian advanced-2	119.14	111.23	477.04
Bayesian advanced-3	118.15	112.66	155.53

However, the overall error across all models remains very high. A model that simply predicts zero delays would have an MAE of 14 making it one of the best-performing models. This clearly indicates that standard regression metrics are far from ideal. Instead of solely computing error metrics, it is also important to analyze whether the distributions of predicted event types align with those observed in the test set, this will be done in Section 4.2.3.

4.2.2 Sample Diversity

So far, variability within the samples has been measured by the span of the error metric. In the CatBoost ensemble training section 4.1.2, it was shown that sample diversity

correlates with the error metric span. However, to obtain more reliable results, variability will now be measured directly by point-wise distance metrics. Prediction diversity measures the extent of inter-sample differences and, therefore, quantifies the diversity of model predictions. Bayesian models, which also account for epistemic uncertainty, exhibit the highest prediction diversity. In contrast, both the baseline and advanced models demonstrate lower diversity, as they contain only a few parameters. Consequently, a large amount of training data is available for these parameters, reducing epistemic uncertainty. However, as shown in Table 4.12, the advanced models contain parameters estimated from only a few data points, leading to higher epistemic uncertainty and, therefore, greater prediction diversity. In general, DistBoost models also perform well on this metric. The most notable result is that ensemble models exhibit very low prediction diversity, which could be attributed to the fact that they are not truly stochastic models.

Table 4.12: The inter sample distances across all samples.

Model	Cityblock	Euclidean
Bayesian advanced-2	28.764	4.180
Bayesian advanced-3	28.233	3.933
DistBoost RMSE	20.798	0.727
Bayesian advanced-1	19.916	0.647
Bayesian baseline	17.305	0.168
DistBoost MAE	11.781	0.206
Ensemble RMSE	1.839	0.019
Ensemble MAE	0.608	0.008

4.2.3 Distributional Fit

The regression-based error metrics provide only an indicator of the model's predictive performance. However, they are not sufficient to fully understand the model's ability to capture probability distributions. Therefore, a detailed distributional analysis will be conducted. The analysis will begin with an examination of the overall distribution and will progressively refine by splitting the dataset along feature axes, such as the type or the segment/ stop of the event. This approach aims to assess how well the model adapts its distributional shape to different inputs.

Illustration of Delay Distributions

First, it will be demonstrated that the true empirical distributions differ depending on the input. In Figure 4.3 one can see that the density plots of the true empirical delays significantly differ between the event types. The same can be observed for delays between different segments and stops in Figure 4.4. The models are expected to capture these

very different shapes of distributions. This will first be evaluated by statistical measures and later by visualizing examples.

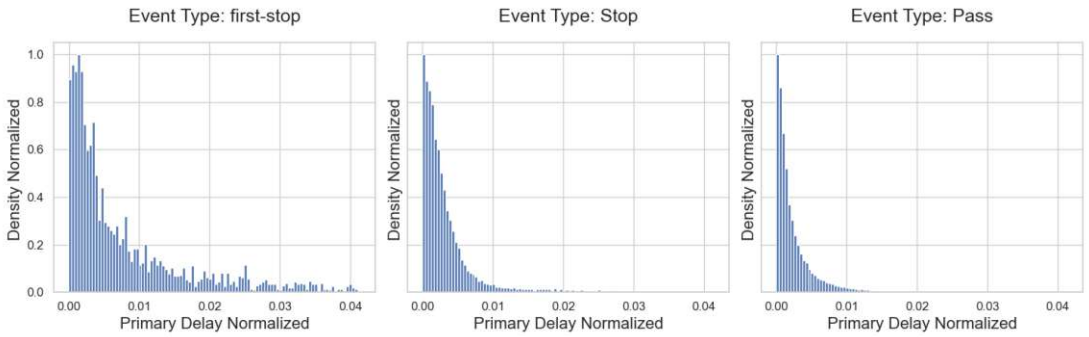


Figure 4.3: Density Plot of the empirical true primary delays larger than zero. The delays themselves as well as the counts where min max normalized. The chart does not show the full tails of the distribution to increase readability. It serves as an illustration that there are significant delay differences between different types of events.

Distributional Fit Statistics

To assess the fit between predicted values across different subsets, we use the KS-Statistic and CVM Statistic. Table 4.13 presents the mean and best values for both statistics across all samples in the full dataset.

Table 4.13: Distributional fit for the whole dataset.

Model	KS-Statistic		CVM-Statistic	
	Mean	Best	Mean	Best
Bayesian baseline	0.013	0.010	6.338	**2.845**
DistBoost RMSE	**0.010**	**0.009**	**4.689**	3.177
Bayesian advanced-1	0.015	0.013	6.850	3.601
Bayesian advanced-2	0.016	0.014	6.316	3.814
Bayesian advanced-3	0.016	0.014	6.481	3.897
DistBoost MAE	0.015	0.015	5.582	4.851
Ensemble MAE	0.122	0.108	305.601	235.639
Ensemble RMSE	0.603	0.481	18061.629	12244.712

According to CVM statistics, the Bayesian baseline now provides the best overall fit, followed by the RMSE-based DistBoost variant. In contrast, the ensemble models have the worst overall fit.

For the KS metric, the baseline and the RMSE-based DistBoost model slightly outperform the other variants. However, the differences are marginal, except when compared to the

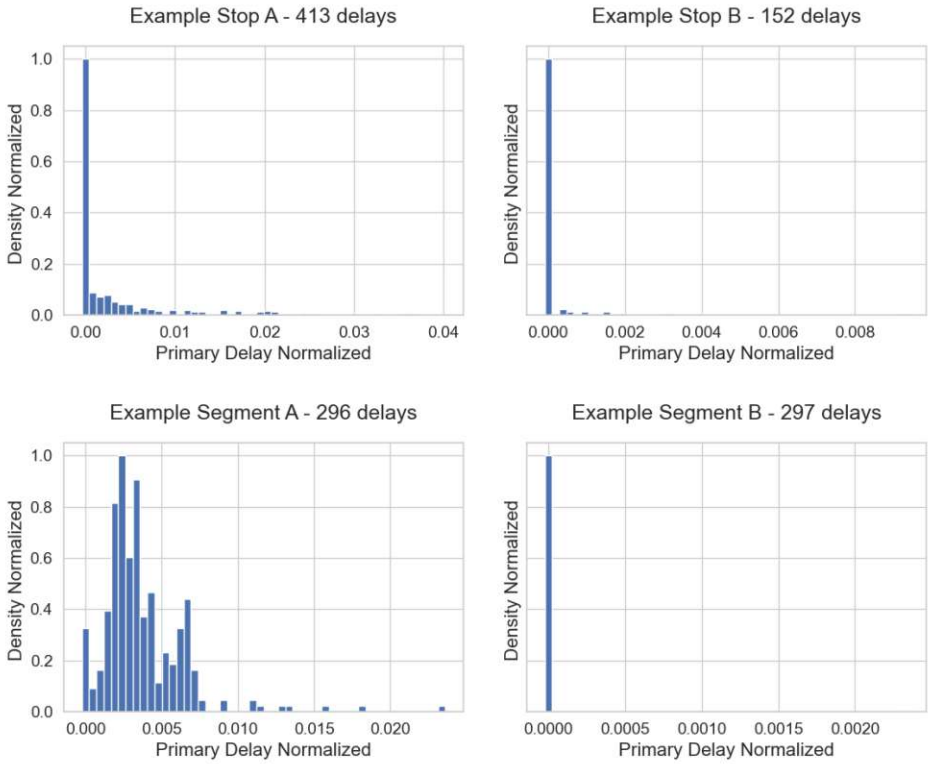


Figure 4.4: Density Plot for two illustrative choices of segments or stops to showcase the difference between them.

ensemble model, which performs significantly worse. This result is particularly logical for the Bayesian baseline, as it primarily fits the overall distribution, only distinguishing between stops and passes. In general, Bayesian models always use pooled (shared) parameters and therefore should perform well on this benchmark, though the posterior distribution has a more robust shape in models with fewer parameters.

To validate these metrics, the distribution of all non-zero primary delays is shown in Figure 4.5. As observed, the Bayesian primary delay distribution closely resembles the True Delays distribution. Additionally, the ensemble models that performed the worst according to the metrics also exhibit a poor visual fit.

Overall, models that explicitly or implicitly capture a distribution tend to achieve a much better overall fit. However, RMSE appears to be the better loss function for the parameterized distribution model, as it allows the model to predict larger mean and sigma values.

To better understand how the shapes of predictive distributions adapt to the input features, the dataset will be split along a specific axis. Reporting the mean and best

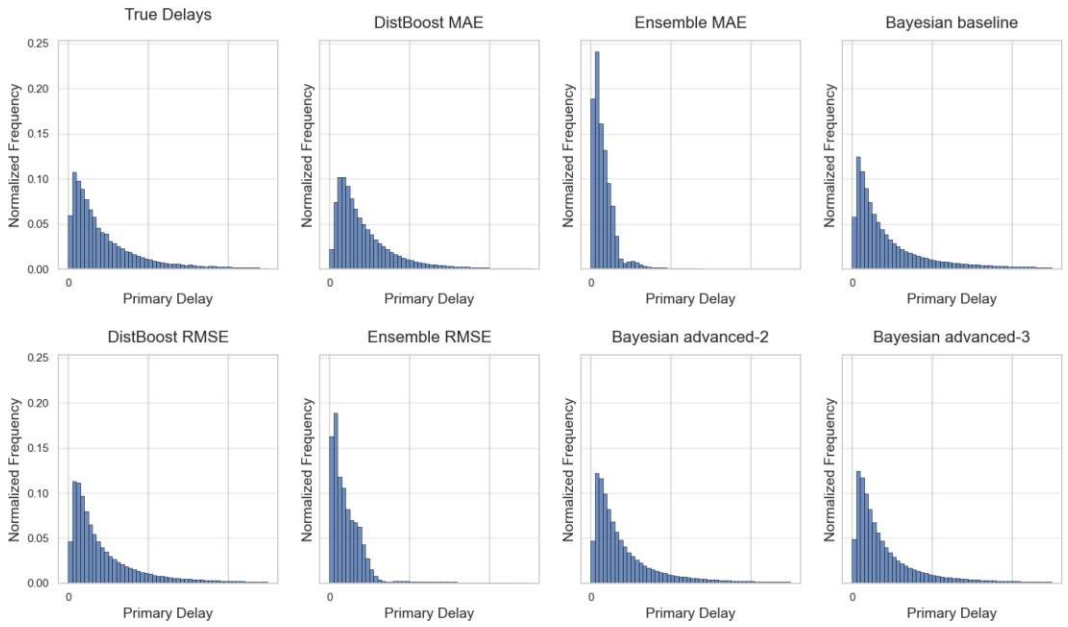


Figure 4.5: Primary Delays overall distribution. The x-axis ticks as well as the bin size are hidden to provide data anonymity.

values for each subset along the axis would result in a three-dimensional table, which could be confusing. Therefore, for now, only the best value of the CVM statistics across all samples for each subset will be presented. Table 4.14 shows the predictive distributions fitted across the six main event differentiations used in the distributional analysis. For each event, the best-performing model is always one of the DistBoost models. These models clearly outperform the Bayesian models, which, in turn, significantly outperform the ensemble models.

Similar results become apparent when splitting the dataset across its stops and segments. Since displaying the best statistics for all 5,591 segments and stops is not feasible, the summary statistics for all of them are presented in Table 4.15. Segments and stops with less than 10 delay entries were removed to increase robustness of the statistic.

Lastly, a visual analysis will be conducted. A segment with larger-than-usual delays was "cherry-picked" to illustratively visualize the model's predictive distribution. The predictive distributions for all models on this segment are plotted in Figure 4.6. As expected from the distributional statistics analysis, the CatBoost ensemble models achieve a low RMSE/MAE by underpredicting the target and only predicting low delays. In contrast, the Bayesian models, in their simple form, are inflexible as they do not utilize features such as the segment in this case. Nevertheless, they still provide a proper modeling of large values. When the Bayesian models become more complex and utilize input features, we observe that the peak of the distributions shifts in the right direction,

Table 4.14: The dataset is split as for the distribution analysis by the type (first_stop, stop, pass) and passenger boolean variable. For each model and subset the best CVM-Statistic is calculated.

	Run		Stop		First Stop	
	freight	passenger	freight	passenger	freight	passenger
Model						
DistBoost MAE	0.878	**3.585**	0.286	3.554	0.239	5.313
DistBoost RMSE	**0.102**	4.754	**0.096**	**0.274**	**0.083**	**0.331**
Bayesian advanced-2	1.331	7.586	6.376	29.48	26.26	96.236
Bayesian advanced-3	1.287	7.629	6.364	29.945	26.234	95.628
Bayesian advanced-1	0.941	8.405	5.997	23.007	25.779	83.94
Bayesian baseline	20.101	14.618	5.721	77.402	32.841	261.028
Ensemble MAE	53.158	88.457	7.834	150.235	41.494	76.504
Ensemble RMSE	3981.524	3995.441	400.043	3941.33	62.241	207.004

Table 4.15: The dataset is split by each segment or stop but every subset which has less than ten delays is removed to increase reliability of the test. This results in 4890 splits. As it is not possible to display the best CVM statistic for each subset summary statistics across all subsets are presented.

	mean	75%	max
Model			
DistBoost RMSE	**0.037**	**0.019**	19.795
DistBoost MAE	0.041	0.024	**10.439**
Bayesian advanced-2	0.119	0.036	14.813
Bayesian advanced-3	0.121	0.036	14.293
Bayesian advanced-1	0.461	0.338	31.515
Bayesian baseline	0.470	0.345	32.502
Ensemble MAE	0.861	0.913	54.176
Ensemble RMSE	4.095	5.581	122.755

although the predictions remain somewhat conservative. This could be due to the tight priors on the segment and stop offset, which will be discussed later. Finally, the DistBoost model, based on this one example, appears to have the best fit and adapts well to the inputs.

4.2.4 Bayesian Model Comparison

So far, the models have been compared using error metrics and distributional fit. However, Bayesian models allow for more advanced comparison methods by evaluating the expected log predictive density (ELPD) which utilizes the posterior predictive distribution on new

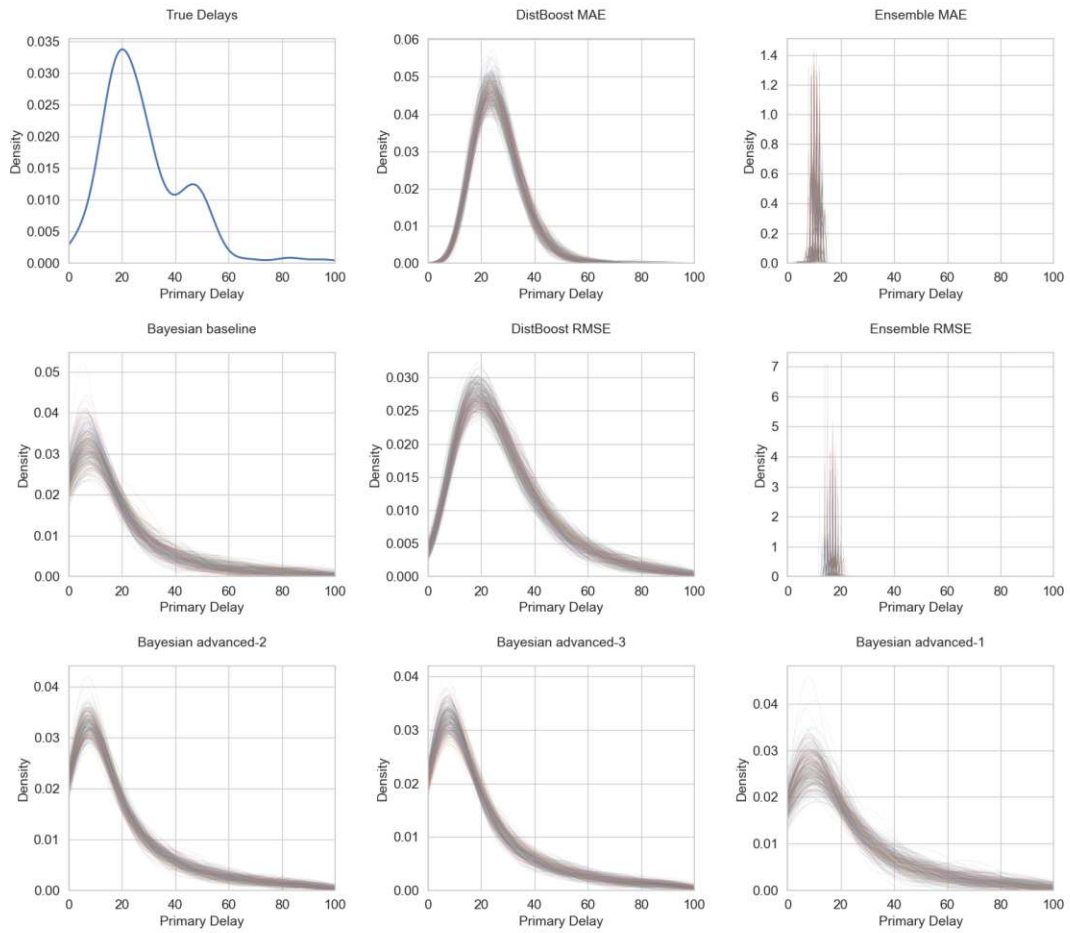


Figure 4.6: Kernel density estimate (KDE) plot for the true primary delay values and for each model for all 300 samples. Data is only a subset representing a segment with unusual large primary delays to illustrate the adaptability.

samples, rather than relying solely on pointwise errors. This approach takes the entire predictive distribution into account when comparing to the true value, which is a superior way to evaluate fit with stochastic models. This comparison is typically performed using Leave-One-Out Cross-Validation (LOO) or the Widely Applicable Information Criterion (WAIC). [VGG17]

In this thesis, LOO was used as an estimate of the out-of-sample predictive fit. LOO cross-validation approximates the performance of the model on unseen data by systematically leaving out one observation at a time and evaluating the model's predictive accuracy on that omitted point.

Cross-validation is also widely used in the machine learning (ML) domain, where it involves repeatedly partitioning the dataset into training and validation (or holdout)

sets. The model is trained on a subset of the data and then evaluated on the remaining portion to assess its generalization performance.

In Table 4.16 the ELPD-LOO results for each submodel and variant are listed. Here we can see that when evaluating the posterior predictive fit, the more complex models outperform the simpler ones. This is in line with the distributional analysis, even though this method is statistically more sophisticated.

Table 4.16: The expected log predictive density (ELPD) for the three submodels and 4 variants. As the overall values are quite high and visually hard to compare the difference to the best results was calculated. Therefore zero presents the best result and every other value the absolute difference.

Model	ELPD diff zero	ELPD diff run	ELPD diff stop
advanced-3	0.0	236.2	0.0
advanced-2	1310.5	0.0	125.6
advanced-1	13123.4	131972.6	84863.0
baseline	16319.3	133272.2	89417.7

4.3 Feature Importance

CatBoost, as an ML library, provides the capability to calculate feature importance. This was done early in the study and was used in collaboration with domain experts to select the input features for the Bayesian models, as using all features would result in high computational complexity. Table 4.17 presents the feature importance for the best-performing model in terms of distributional fit, the RMSE DistBoost model.

A majority of the model's variability is explained by a few key input features, such as the segment or stop, the category (which can be merged with passenger since it is a superset), the type of event (stop, pass, first-stop) and, lastly, the hour of the day. Therefore, the general practice of excluding some of the less important features for a more robust or stochastically capable model is acceptable.

4.4 Model Performance Simulation Loop

For the train operators, mainly the positive additional delays are of interest. Due to the small amount of positive secondary delays in Figure 4.7, the secondary delay distributions averaged across all samples are plotted to get an understanding of the data distribution. The first striking finding is that, even though, the positive delays are the main point of interest, there is also a lot of information in the negative delays. Furthermore, no model has as few positive secondary delays as when the true primary delays are injected.

Table 4.17: Feature importance values from the DistBoost model trained with RMSE loss.

Feature	mu	sigma	zero	sum
segment_or_stop	29.37	19.55	51.56	100.47
category	18.92	16.50	17.75	53.18
type	10.24	8.06	7.78	26.08
operational_type	5.47	16.59	3.19	25.26
scheduled_arrival_hour	7.11	10.83	6.56	24.50
passenger	7.60	4.65	1.26	13.51
trainpart_speed	3.21	4.41	4.73	12.34
num_platform_edges	2.88	7.73	1.64	12.25
distance_geo	6.55	1.97	2.62	11.14
num_siding_tracks	2.51	7.10	0.95	10.55
v_max	3.08	1.76	1.52	6.36
is_first_stop	1.77	0.85	0.44	3.06
ocp_type	1.29	0.00	0.00	1.29

When looking at the fit of the overall distribution compared to the simulation experiment with the true primary delays, the results in Table 4.18 are similar to the visual examination. Leading in terms of KS and CVM-statistics are the DistBoost models followed by the Bayesian models. The worst fitting models are the ensemble models. When comparing pointwise predictions via MAE, the Ensemble RMSE model has the lowest error, closely followed by the DistBoost models. Within the Bayesian family, only the baseline model reaches an acceptable MAE.

Model	Mean KS-Statistic	Mean CVM-Statistic	Mean MAE
DistBoost RMSE	0.012	10.093	4.614
DistBoost MAE	0.022	12.245	4.724
Bayesian baseline	0.020	12.444	5.336
Bayesian advanced-3	0.022	23.301	13.785
Bayesian advanced-1	0.030	35.879	11.713
Bayesian advanced-2	0.035	67.402	18.484
Ensemble RMSE	0.097	541.788	4.080
Ensemble MAE	0.144	1282.217	6.807

Table 4.18: Mean Distributional fit and Error metric across all 300 simulated secondary delays compared to the simulated secondary delays with the true primary delays injected.

When considering the total amount of injected delays, the ensemble MAE exhibited the

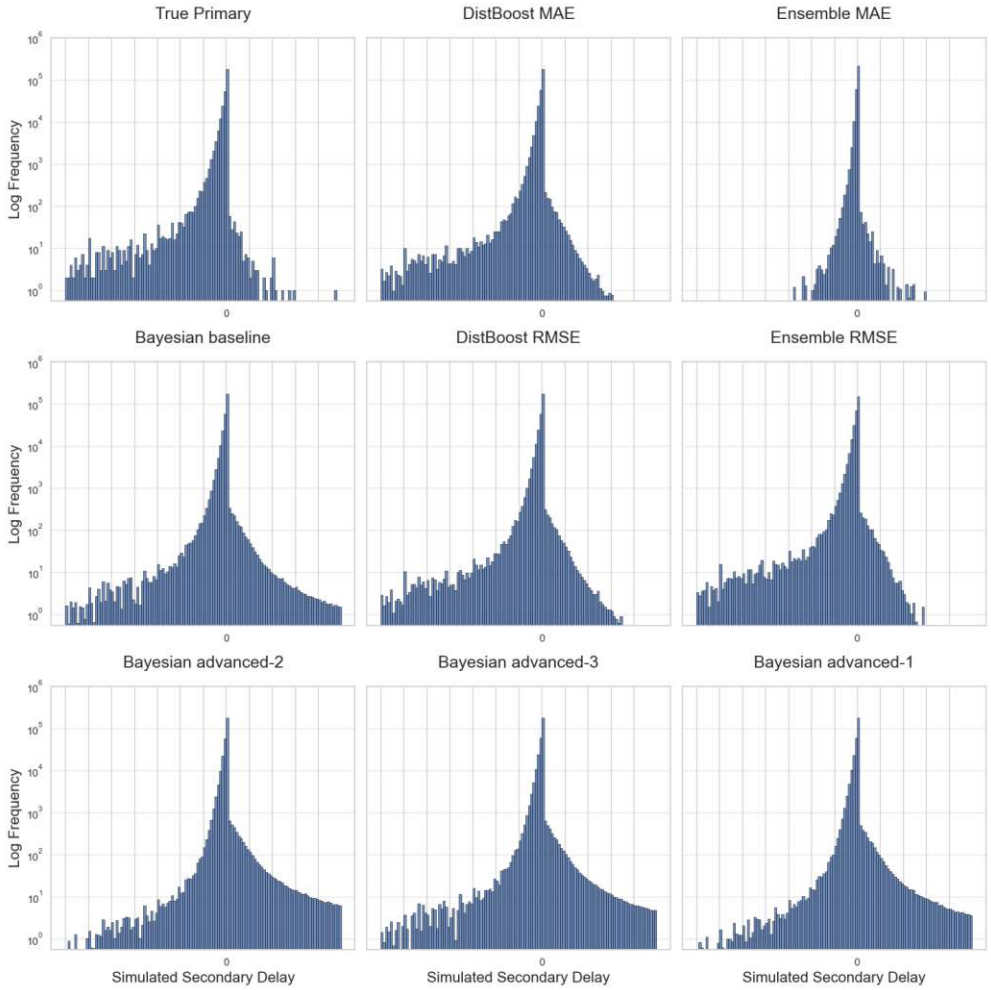


Figure 4.7: Average histogram computed by first generating histograms for each simulated secondary delay sample and then averaging the bin counts across all columns to capture the overall distribution trend. Additionally, the histogram of the true primary delays and zero delays is included for comparison.

lowest value, with a mean of only 13% of the total injected delays. However, it also had the lowest MAE among all models. The simulation results indicate that, in interaction with the simulation, this model performs worse in terms of MAE compared to models such as DistBoost. This suggests that optimizing a model for an error metric like MAE does not necessarily result in a low error or a good overall fit when compared to the true simulated delays.

One might assume that Bayesian models, given their low MAE and strong performance in statistical tests before simulation, would maintain these advantages afterward. However, this is not the case. After simulation, the Bayesian models perform only moderately well

on the distributional fit and exhibit the worst performance in terms of MAE. This leads to the conclusion that, during theoretical performance evaluation, achieving a balance between error metrics and fit tests is crucial.

CHAPTER 5

Discussion

This chapter discusses the main findings from the results section in order to answers the four research questions of this thesis. Additionally, new directions for future research, identified through the detailed analysis of the results, are presented.

5.1 Best Fitting Distribution

Upon analyzing the PDIM models and their performance, an appropriate target distribution was identified. To determine the best-fitting distribution, multiple statistical tests were conducted, including the Kolmogorov–Smirnov (KS) and Cramér–von Mises (CVM) tests. While arguments exist both for and against each test, the analysis concluded that using both in combination provides a more robust evaluation of the most suitable distribution.

Testing the full dataset alone proved insufficient, as it was essential to assess the fit while distinguishing between different event types. This distinction included passenger versus freight traffic, as well as specific event types such as stops, first stops, and passes. Although further segmentation - such as by time of day or track segment - could have been considered, an initial division into six main event types offered a balanced trade-off between complexity and model accuracy within the scope of this thesis.

The results demonstrated that models based on the best-performing distributions effectively captured the characteristics of the observed data. Based on these findings, Research Question 1:

> Which likelihood distribution best captures the statistical characteristics of disaggregated primary delays in railway operations and which test is best to evaluate the fit?

can now be addressed. The analysis confirms that both the KS and CVM tests are suitable evaluation methods and should ideally be used in conjunction. If a single target distribution were to be selected, the log-normal distribution provides the best overall fit. However, if multiple distributions are allowed for different event types, the Pareto distribution is most appropriate for freight-related events, while the log-normal distribution remains the best choice for passenger-related events.

5.2 Aleatoric vs Epistemic Uncertainty

The primary objective of this thesis was to develop multiple Primary Delay Injection Models (PDIMs) and assess which is most suitable for integration into a large-scale macroscopic simulation system. Various approaches, each with multiple subversions, were evaluated. The Ensemble model, being the simplest approach, showed little promise, as the CatBoost predictions generally exhibited low inter-sample diversity and a poor distributional fit. These models had low error rates which is expected as this is exactly what they optimized for and what ML-models generally excel at. However, as explained in the Results section, pointwise error is not an appropriate metric for evaluating a stochastic PDIM. Due to the randomness in the data and the loss functions used for training, the target values were predominantly under-predicted.

Performance on the "is zero" task heavily depended on the chosen loss function, and it is therefore recommended to explicitly model the "is zero" case. The Bayesian approach offered two key advantages. First, even the simplest Bayesian models demonstrated good overall performance, despite having only five parameters. Second, more complex Bayesian models exhibited higher inter-sample diversity. However, some parameters had limited data available for sampling, or the prior and model specification may have been inadequate, occasionally leading to the generation of extremely large and unrealistic delays. One potential solution would be to group certain categorical features that occur infrequently into broader categories while preserving those that appear frequently. This could help reduce epistemic uncertainty, which is the primary cause of outliers.

In general, the Bayesian framework is highly powerful, but careful consideration is required at every stage (from model design and prior specification to feature engineering and missing data imputation) to develop a well-performing model. Future research could explore modeling freight and passenger delays with different distributions (e.g. log-normal, Pareto) or using a mixture of both. However, small undocumented experiments suggested that mixture models are slow to fit and require carefully crafted priors to avoid divergences. Since not all possible approaches were explored, it is likely that Bayesian models could perform better when incorporating these findings.

This leads to the DistBoost model family, which used a machine learning model to parameterize two probability distributions: a Bernoulli distribution to model zero and non-zero delays and a normal distribution, which - through logarithmic and exponential transformations - represents a log-normal distribution. The DistBoost models performed well in terms of both error and distributional fit and were easy and fast to train and

interpret. A significant advantage is that, due to the optimization within the CatBoost library, models trained on much larger datasets (100+ days) can be handled efficiently. Additionally, the machine learning component can be easily replaced.

This leads to Research Question 2:

> How do models which represent either aleatoric uncertainty, epistemic uncertainty or those who represent both impact model performance?

The general conclusion is that incorporating aleatoric uncertainty leads to higher inter-sample diversity, making the model statistically more capable. However, this is only beneficial if the posterior distributions of the model parameters are not excessively wide, which would otherwise result in extreme and unrealistic predictions. As demonstrated in Figure 4.6, the model is capable of adapting to out-of-distribution segments. For the models examined in this thesis, the disadvantages of aleatoric uncertainty outweighed the benefits, especially when compared to a fast, powerful and easy-to-train model that accounts for epistemic uncertainty but not aleatoric uncertainty. The Ensemble model, which does not effectively incorporate either aleatoric or epistemic uncertainty, demonstrated with its poor performance that epistemic uncertainty is a required property in stochastic delay modeling. While the comparison between DistBoost and the Ensemble models showed that aleatoric uncertainty, even though theoretically desired, is not necessary to build a PDIM.

5.3 Adaptability

One of the main objectives in selecting and developing the PDIM was to ensure adaptability to new, unseen data, including potentially new input features such as additional stops or train categories. From the outset, the models were designed to facilitate this adaptability. Due to the limited dataset size, this capability was not explicitly evaluated. However, the following section outlines approaches for enabling the models to adapt to previously unseen data.

For the Bayesian models, adaptation to new data is relatively straightforward. In the baseline model, no modifications are required, as the only inputs are the event type and whether the train is a passenger or freight train. In the advanced model, each segment is represented as an offset added to or subtracted from the overall mean for a given event. Consequently, the parameter $\mu\_segment$ for an unknown segment can be initialized to normal distribution with mean zero and sigma the same as the prior. Due to the addition with the shared mean, it is effectively defaulting to the overall mean until new data becomes available. In the Bayesian case, true online learning - where parameters are updated as new data arrives - would also be feasible. This allows Bayesian models to easily adapt to new and unseen scenarios.

For the CatBoost-based model, adapting to new data is not as straightforward as with Bayesian models. This challenge begins in the training and feature selection phase.

Highly specific features, such as the *train_number*, rely on the assumption that the same schedule is present during both the fitting and sampling phases. However, this assumption does not hold for passenger traffic, which changes quarterly to yearly, or for freight traffic, where unique trains often run only once. As a result, the *train_number* cannot be used to access information about a new freight train. Therefore, only trains that reoccur over a longer time period should have a label for *train_number*. All non-reoccurring trains or those from an outdated schedule should be assigned a missing value label. For this reason, the *train_number* was not utilized in the models of this thesis.

In contrast, features such as *passenger*, *is_first_stop*, or *segment_or_stop* are always present. However, in rare cases, a new stop could be introduced. In such situations, the corresponding data entry should not be marked as missing, since the CatBoost library treats missing values as a separate distinct category, which could introduce bias if the training dataset contained specific target distributions for missing values. Instead, the new, previously unknown station should be mapped to an existing station with similar properties determined by a domain expert. Another option would be to train a separate fallback base model using only features that almost never change, which could be utilized when new stations are introduced. Nonetheless, even if the label was simply replaced by a missing value, CatBoost still has many other features to rely on for predictions. Therefore, if such cases occur infrequently, this approach could be acceptable.

All of these considerations are only necessary until the model is retrained on the updated data, which is fortunately a fast process for DistBoost models. One benefit of the CatBoost model architecture is that it utilizes a prior, such as the average target value, when no information is available for a particular feature, as previously discussed.

Therefore, regarding Research Question 3:

> How can the stochastic delay model be extended to generalize for predicting primary delays in novel railway scenarios (e.g., new trains, routes, altered timetables)?

It can be concluded that the Bayesian model can adapt to new data without modifying the training process, simply by introducing new prior distributions for previously unknown features. In contrast, ML-based models require careful consideration during both training and prediction to account for the potential impact of unknown elements on the predictions.

5.4 Simulation Loop Validation

The simulation validation demonstrated that, while the primary focus is on positive secondary delays, considering the full distribution - including negative delays - is essential. Negative additional delays influence subsequent simulation steps, impacting the overall system behavior.

Since primary delays dominate the combined system, the simulation-based evaluation played a less critical role than initially expected. A key finding was that models balancing theoretical performance - achieving both low error and high distributional fit - also exhibited lower errors in simulated secondary delays.

Thus, for Research Question 4:

> How can the stochastic primary delay model, in conjunction with the simulation loop, be validated by evaluating its ability to reproduce observed total delays?

In conclusion, both positive and negative secondary delays must be taken into account, as they collectively influence the total delays within the system. Additionally, the primary delays themselves play a crucial role in shaping the final delay distribution. The results also indicate that injecting the true primary delays leads to lower errors compared to evaluating models in isolation. Therefore, validation should not rely solely on pointwise error metrics such as MAE, but should also consider the overall distributional fit to provide a more comprehensive assessment of model performance.

5.4.1 A Note on Extreme Values in Stochastic Simulation

In the context of railway primary-delay modeling for simulation systems, an important question to explore is how extreme delay values should be sampled to ensure a comprehensive exploration of the probability space within a limited number of simulation experiments. Consider a simple example with a single track segment where only 10 trains pass on a given day. Additionally, assume that the delay distribution is not agnostic to any specific features. For instance, the probability of observing a delay greater than 100 seconds is 0.01. In this case, simulating 100 experiments would result in drawing a delay greater than 100 seconds roughly once per 10 simulations. This approach effectively facilitates the exploration of extreme delay events, with one such event potentially representing a large delay in each simulation.

However, the situation changes when extending this scenario to multiple segments or when delays occur consecutively. In this case, rare, large delays might not be captured within the 100 simulations. For example, if the delay threshold is increased to 600 seconds (10 minutes), and the probability of such an extreme delay is 0.0001, it becomes unlikely that any of the 100 simulations would include a delay of that magnitude. Consequently, the model might fail to capture such rare, high-impact events.

This leads to the question of whether the model should focus primarily on more frequent, realistic delays that are likely to occur more often and are less extreme. If a schedule can accommodate these smaller delays without causing significant disruption, a large delay occurring at a random point in time might not lead to a major issue, if the track's capacity is not overloaded. However, if a track is typically operating near full capacity,

even a single additional train with a large delay could severely impact the system's performance, potentially causing cascading delays.

The key decision here is whether to prioritize modeling these rare extreme delays in the simulation or focus on the more common delays, and whether it is acceptable to exclude certain rare delays to maintain computational efficiency. The overall goal is to design a scheduling system that avoids congestion and ensures that the system remains robust, even in scenarios where multiple delays may occur. If the track segment is not frequently used to capacity, it might be acceptable to allow a large delay to occur, as long as the system's capacity is not exceeded. However, if the track segment is often operating near full capacity, even small delays could lead to significant disruptions, making it critical to consider both frequent and rare delays in the model.

CHAPTER 6

Conclusion

The modeling of primary delays in railway systems is a well-established research area, having been studied for decades. The advent of simulation systems - whether macroscopic or microscopic - has provided train operators with the ability to test schedules in isolated environments. As computational power has grown, the ability to run numerous simulation experiments within short time frames has become increasingly feasible. However, these isolated environments fail to capture the complexities and issues that arise in real-world operations. This is where stochastic input modeling plays a critical role: by introducing realistic errors, such as delays, into the system, we can evaluate how the system responds to these uncertainties.

With the ongoing digitalization of railway networks and traffic management systems, an increasing amount of data is becoming available each year. This thesis serves as an initial investigation into different modeling approaches for primary delays, their benefits and their limitations. The data available for this study significantly influenced the choice of models. The goal was to model the predictive distribution of primary delays for a given train on a specific segment or stop. Future research could use a broader set of features (if available) or incorporate live data feeds. However, in the context of train operators, having a model that can reliably sample delays from limited information is particularly valuable, given that the digitalization of a full railway network is a long-term process.

Ensemble models, while initially seeming to be a straightforward solution for primary delay modeling, proved inadequate in reliably capturing the stochastic nature of delays. These models lacked a truly stochastic component and, as a result, they were unable to simulate the randomness inherent in delay patterns. Nevertheless, they provided valuable insights into the best evaluation methods for PDIMs and their shortcomings highlighted the need for more advanced modeling approaches.

The DistBoost models, which parameterized the log-normal and Bernoulli distributions using the machine learning model CatBoost, showed significant promise. These models

effectively captured the stochastic processes involved in delay modeling and demonstrated excellent performance in both training and prediction. With larger datasets - potentially including weather, passenger information, and other variables - fitting a state-of-the-art model like CatBoost becomes relatively straightforward and computationally efficient. However, a key limitation of the DistBoost model is its inability to account for epistemic uncertainty, which is particularly important when working with limited data, as was the case in this thesis. This limitation makes it challenging to model the uncertainties that arise from sparse or incomplete datasets.

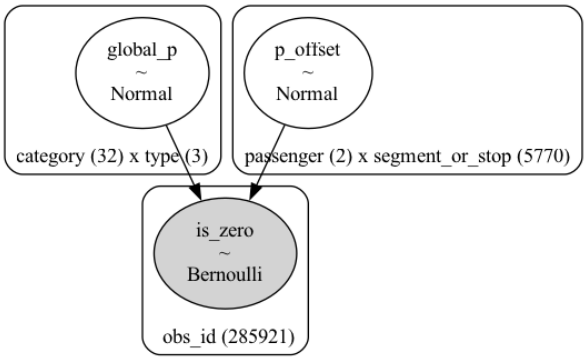
In contrast, Bayesian models excel at modeling epistemic uncertainty and are inherently more explainable. They allow for the integration of expert priors and can be tailored with custom architectures in collaboration with domain experts. However, these models are challenging to build and are sensitive to bad priors and tuning. Furthermore, the sampling process can be slow compared to machine learning-based models like DistBoost. As the amount of data and number of features grows, these challenges become more pronounced. Despite these issues, Bayesian models can still offer good performance with relatively few parameters and are particularly useful for understanding outliers, although more complex models sometimes struggle in this area.

In addition to theoretical performance, the interaction between the PDIMs and the simulation system was validated. While primary delays are the primary contributors to total delays, changes in validation metrics after simulation revealed that the overall system behavior is also influenced by secondary delays. The results indicated that while primary delays were central to the overall delay distribution, a comprehensive evaluation of both positive and negative delays was necessary for an accurate simulation-based validation.

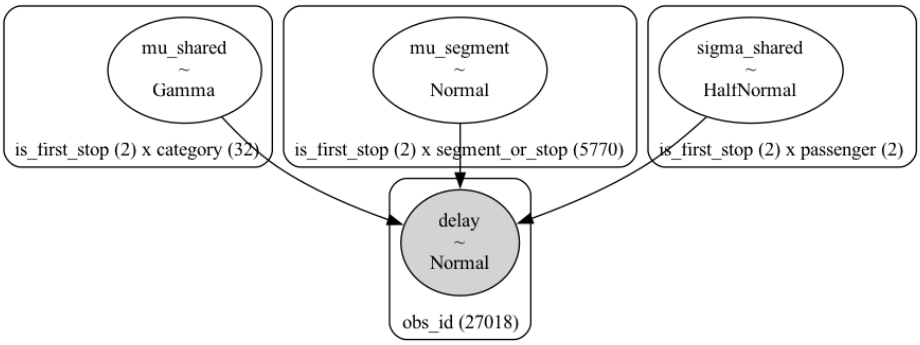
APPENDIX A

Bayesian Models

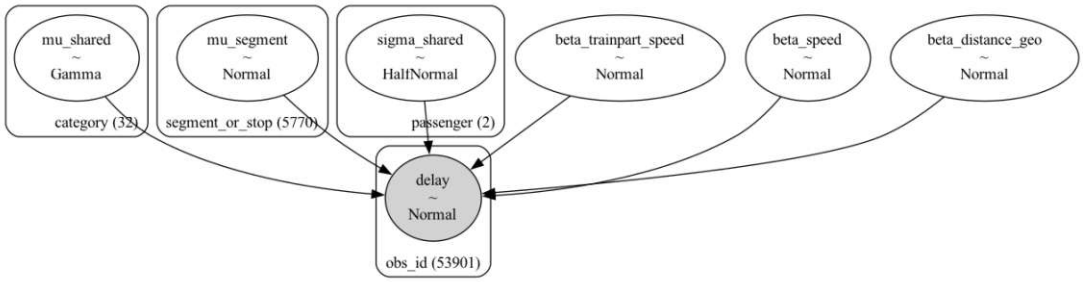
Due to the large visualization size the advanced-2 Bayesian model is only shown in the appendix:



(a) Zero Model



(b) Stop Model



(c) Run Model

Figure A.1: Advanced-2 models structure.

APPENDIX B

Model Training

B.1 Larger Gridsearch Ensemble Model

The results of the full grid search on the CatBoost-Ensemble with only 10 models per ensemble due to the large search space are presented in the table below (B.1). The dataset is sorted by the mean MAE. The "Value" column indicates either the subsample or the bagging temperature, depending on the mode. As shown in the table, the Bootstrap mode has minimal effect on the variability.

Loss	Subsample	Mode	Value	MAE mean	MAE span
RMSE	0.3	Bayesian	0.0	18.28	1.71
		Bernoulli	1.0	18.28	1.71
			0.7	18.10	1.62
		Bayesian	1.0	18.01	1.38
		Bernoulli	0.3	18.04	0.91
		Bayesian	10.0	18.09	0.83
	0.7	Bernoulli	0.3	17.80	0.82
		Bayesian	1.0	17.75	0.74
		Bernoulli	0.7	17.80	0.72
	1.0	Bernoulli	0.7	17.58	0.71
		Bayesian	0.0	17.64	0.71
		Bernoulli	1.0	17.64	0.71
		Bayesian	1.0	17.53	0.66
	0.7	Bernoulli	1.0	17.82	0.56
		Bayesian	0.0	17.82	0.56
	1.0	Bernoulli	0.3	17.62	0.46
	0.7	Bayesian	10.0	17.98	0.42
	1.0	Bayesian	10.0	17.84	0.24
MAE	1.0	Bernoulli	1.0	13.58	0.17
		Bayesian	0.0	13.58	0.17
	0.3	Bayesian	1.0	13.69	0.16
	0.7	Bernoulli	1.0	13.65	0.15
		Bayesian	0.0	13.65	0.15
	1.0	Bernoulli	0.7	13.56	0.15
	0.7	Bernoulli	0.7	13.61	0.13
	0.3	Bernoulli	0.7	13.70	0.13
	0.7	Bayesian	1.0	13.60	0.12
	1.0	Bayesian	1.0	13.58	0.12
	0.3	Bernoulli	1.0	13.72	0.11
		Bayesian	0.0	13.72	0.11
	0.7	Bernoulli	0.3	13.57	0.08
	0.3	Bernoulli	0.3	13.67	0.07
	1.0	Bayesian	10.0	13.74	0.07
		Bernoulli	0.3	13.53	0.05
	0.3	Bayesian	10.0	13.83	0.04
	0.7	Bayesian	10.0	13.76	0.04

Table B.1: Large gridsearch for catboost ensemble model.

Overview of Generative AI Tools Used

Generative AI was utilized in multiple ways throughout the research and writing of this thesis. The primary technology used was OpenAI's GPT-4o model, accessed via the interactive ChatGPT interface. Additionally, GitHub Copilot was employed during coding to assist with completing lines of code and generating boilerplate from comments.

ChatGPT played several roles in the writing process. First, interacting with the model helped me grasp new concepts and explore potential research directions. However, it was only used as a preliminary tool to guide my understanding and was never cited as an information source.

Second, ChatGPT served as a writing assistant, helping to improve sentence structure, particularly in cases of long or complex sentences. To maintain consistency and ensure that the model focused solely on structure and grammar without introducing new information, I used the following predefined prompt:

> You're my writing coach. Improve grammar and clarity in this section of my railway modeling thesis. Please do not add new information, just make the structure clear and easy to understand. Retain LaTeX citations and remove generic "we" formulations:

All AI-generated content was carefully reviewed and revised, as the model sometimes introduced unintended modifications or misinterpreted the original meaning.

Lastly, ChatGPT was used to generate boilerplate code, such as utility functions and visualizations. However, the main model and data import code were entirely written by the author.

List of Figures

List of Tables

Bibliography

[APAC+23] Oriol Abril-Pla, Valentin Andreani, Colin Carroll, Lin Dong, Christopher J. Fonnesbeck, Maxim Kochurov, Ravin Kumar, Junpeng Lao, Colin C. Luhmann, Osvaldo A. Martin, Michael Osthege, Ricardo Vieira, Thomas Wiecki, and Ricardo Zinkov. PyMC: a modern, and comprehensive probabilistic programming framework in Python. *PeerJ Computer Science*, 9:e1516, May 2023.

[BBP+23] Dominik Brunmeir, Martin Bicher, Niki Popper, Matthias Rößler, Christoph Urach, Claire Rippinger, and Matthias Wastian. Four Years of Not-Using a Simulator: The Agent-Based Template. In *2023 Winter Simulation Conference (WSC)*, pages 255–266, San Antonio, TX, USA, December 2023. IEEE.

[Bü17] Paul-Christian Bürkner. **brms** : An *R* Package for Bayesian Multilevel Models Using *Stan*. *Journal of Statistical Software*, 80(1), 2017.

[CAX20] Canan G. Corlu, Alp Akcay, and Wei Xie. Stochastic simulation under input uncertainty: A Review. *Operations Research Perspectives*, 7:100162, January 2020.

[CK18] Francesco Corman and Pavle Kecman. Stochastic prediction of train delays in real-time using Bayesian networks. *Transportation Research Part C: Emerging Technologies*, 95:599–615, October 2018.

[CMZ16] Yong Cui, Ullrich Martin, and Weiting Zhao. Calibration of disturbance parameters in railway operational simulation based on reinforcement learning. *Journal of Rail Transport Planning & Management*, 6(1):1–12, June 2016.

[DEG18] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. CatBoost: gradient boosting with categorical features support, October 2018. arXiv:1810.11363.

[Dev24] CatBoost Developers. Catboost documentation: Bootstrap options, 2024. Accessed: 2025-02-04.

[Dev25] PyMC Developers. Introductory overview of pymc, 2025. Accessed: 2025-02-18.

[DS12] Morris H DeGroot and Mark J Schervish. Probability and statistics, Fourth, 2012.

[FHJW19] Sarah Frisch, Philipp Hungerländer, Anna Jellen, and Dominic Weinberger. A Mixed Integer Linear Program for Optimizing the Utilization of Locomotives with Maintenance Constraints. In Bernard Fortz and Martine Labbé, editors, *Operations Research Proceedings 2018*, pages 103–109, Cham, 2019. Springer International Publishing.

[GCD13] Rob M.P. Goverde, Francesco Corman, and Andrea D'Ariano. Railway line capacity consumption of different railway signalling systems under scheduled and disturbed conditions. *Journal of Rail Transport Planning & Management*, 3(3):78–94, August 2013.

[GCS+13] A. Gelman, J.B. Carlin, H.S. Stern, D.B. Dunson, A. Vehtari, and D.B. Rubin. *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2013.

[Gle57] V. B. Gleaves. Cyclic scheduling and combinatorial topology: Assignment and routing of motive power to meet scheduling and maintenance requirements Part I - A statement of the operating problem of the Frisco railroad. *Naval Research Logistics Quarterly*, 4(3):203–205, 1957. Publisher: John Wiley & Sons.

[GR92] Andrew Gelman and Donald B. Rubin. Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4):457–472, November 1992. Publisher: Institute of Mathematical Statistics.

[HB23] Johan Högdahl and Markus Bohlin. A Combined Simulation-Optimization Approach for Robust Timetabling on Main Railway Lines. *Transportation Science*, 57(1):52–81, January 2023.

[JPS+22] Ingrid Johansson, Carl-William Palmqvist, Hans Sipilä, Jennifer Warg, and Markus Bohlin. Microscopic and macroscopic simulation of early freight train departures. *Journal of Rail Transport Planning & Management*, 21:100295, March 2022.

[KD09] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2):105–112, March 2009.

[KMF+17] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[Lai04] Francesco Laio. Cramer–von Mises and Anderson-Darling goodness of fit tests for extreme value distributions with unknown parameters. *Water Resources Research*, 40(9):2004WR003204, September 2004.

[Law13] Averill M. Law. A tutorial on how to select simulation input probability distributions. In *2013 Winter Simulations Conference (WSC)*, pages 306–320, December 2013. ISSN: 1558-4305.

[LNC24] Jan Lordieck, Michael Nold, and Francesco Corman. Microscopic railway capacity assessment of heterogeneous traffic under real-life operational conditions. *Journal of Rail Transport Planning & Management*, 30:100446, June 2024.

[LRH] Raul H C Lopes, Ivan Reid, and Peter R Hobson. The two-dimensional Kolmogorov-Smirnov test.

[MKH18] Vikram Mullachery, Aniruddh Khera, and Amir Husain. Bayesian Neural Networks, January 2018. arXiv:1801.07710 [cs].

[NHL+19] Rahul Nair, Thanh Lam Hoang, Marco Laumanns, Bei Chen, Randall Cogill, Jácint Szabó, and Thomas Walter. An ensemble prediction model for train delays. *Transportation Research Part C: Emerging Technologies*, 104:196–209, July 2019.

[NHSK04] Andrew Nash, Daniel Huerlimann, Jörg Schütte, and Vasco Paul Krauss. Railml† a standard data interface for railroad applications. *WIT Transactions on The Built Environment*, 74, 2004. Publisher: WIT Press.

[OFC+17] Luca Oneto, Emanuele Fumeo, Giorgio Clerico, Renzo Canepa, Federico Papa, Carlo Dambra, Nadia Mazzino, and Davide Anguita. Dynamic Delay Predictions for Large-Scale Railway Networks: Deep and Shallow Extreme Learning Machines Tuned via Thresholdout. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(10):2754–2767, October 2017.

[PGV+18] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[PJS23] Carl-William Palmqvist, Ingrid Johansson, and Hans Sipilä. A method to separate primary and secondary train delays in past and future timetables using macroscopic simulation. *Transportation Research Interdisciplinary Perspectives*, 17:100747, January 2023.

[PJV23] Juan Pineda-Jaramillo and Francesco Viti. Identifying the rail operating features associated to intermodal freight rail operation delays. *Transportation Research Part C: Emerging Technologies*, 147:103993, February 2023.

[RMC24] RMCon International GmbH. RailSys® Suite. `https://rmcon-int.de/railsys-suite/`, 2024. Accessed: 2024-10-22.

[RWJ+20] Matthias Rosler, Matthias Wastian, Anna Jellen, Sarah Frisch, Dominic Weinberger, Philipp Hungerlander, Martin Bicher, and Niki Popper. Simulation And Optimization Of Traction Unit Circulations. In *2020 Winter Simulation Conference (WSC)*, pages 90–101, Orlando, FL, USA, December 2020. IEEE.

[Sip23a] Hans Sipilä. Simulations with proton and railsys: Use of a macroscopic and microscopicrailway simulation tool in swedish applications, 2023.

[Sip23b] Hans Sipilä. Simulations with PROTON and RailSys: Use of a macroscopic and microscopicrailway simulation tool in Swedish applications, 2023.

[SNTG21] Jaromir Siroky, Petr Nachtigall, Erik Tischer, and Jozef Gaaparik. Simulation of Railway Lines with a Simplified Interlocking System. *Sustainability*, 13(3):1394, January 2021.

[SRK+24] Nadine Schwab, Matthias Röler, Hannah Kastinger, Günter Schneckenreither, Matthias Wastian, and Niki Popper. Validation and Quantification of Possible Model Extensions for a Railway Operations Model Using Delay Data Disaggregation. In *2024 Winter Simulation Conference (WSC)*, pages 1563–1574, December 2024. ISSN: 1558-4305.

[SSK18] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, August 2018.

[sta24] Unwetterschäden zwingen Öbb zu neuem fahrplan in ganz Österreich — derstandard.at, 2024. `https://www.derstandard.at/story/3000000238084/unwetter-g252terverkehr-auf-bahn-weststrecke-massiv-eingeschr228nkt` [Accessed 15-10-2024].

[STBC22] Thomas Spanninger, Alessio Trivella, Beda Büchel, and Francesco Corman. A review of train delay prediction approaches. *Journal of Rail Transport Planning & Management*, 22:100312, June 2022.

[Tea25] CatBoost Team. Catboost: A high-performance gradient boosting library, 2025. Accessed: 2025-02-05.

[TMP23] Kah Yong Tiong, Zhenliang Ma, and Carl-William Palmqvist. A review of data-driven approaches to predict train delays. *Transportation Research Part C: Emerging Technologies*, 148:104027, March 2023.

[TNv20] Erik Tischer, Petr Nachtigall, and Jaromír Široký. The use of simulation modelling for determining the capacity of railway lines in the Czech conditions. *Open Engineering*, 10(1):224–231, January 2020. Publisher: De Gruyter Open Access.

[VDSDK+21a] Rens Van De Schoot, Sarah Depaoli, Ruth King, Bianca Kramer, Kaspar Märtens, Mahlet G. Tadesse, Marina Vannucci, Andrew Gelman, Duco Veen, Joukje Willemsen, and Christopher Yau. Bayesian statistics and modelling. *Nature Reviews Methods Primers*, 1(1):1, January 2021.

[VDSDK+21b] Rens Van De Schoot, Sarah Depaoli, Ruth King, Bianca Kramer, Kaspar Märtens, Mahlet G. Tadesse, Marina Vannucci, Andrew Gelman, Duco Veen, Joukje Willemsen, and Christopher Yau. Publisher Correction: Bayesian statistics and modelling. *Nature Reviews Methods Primers*, 1(1):16, February 2021.

[VGG17] Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27(5):1413–1432, September 2017.

[vRMS+20] Laura von Rueden, Sebastian Mayer, Rafet Sifa, Christian Bauckhage, and Jochen Garcke. Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions. In Michael R. Berthold, Ad Feelders, and Georg Krempl, editors, *Advances in Intelligent Data Analysis XVIII*, pages 548–560, Cham, 2020. Springer International Publishing.

[WdLNvV24] Maylin Wartenberg, Marvin Auf der Landwehr, Laura H. M. Nguyen, and Christoph von Viebahn. Supervised Machine Learning for Input Modelling of an Agent-Based Simulation Model for Autonomous On-Demand Shuttle Services. In Miguel Mujica Mota and Paolo Scala, editors, *Simulation for a Sustainable Future*, pages 227–241, Cham, 2024. Springer Nature Switzerland.

[WLL+17] Chao Wen, Zhongcan Li, Javad Lessan, Liping Fu, Ping Huang, and Chaozhe Jiang. Statistical investigation on train primary delay based on real records: evidence from Wuhan–Guangzhou HSR. *International Journal of Rail Transportation*, 5(3):170–189, July 2017.

[WLL+19] Chao Wen, Zhongcan Li, Javad Lessan, Hongguo Shi, Liping Fu, and Matthew Iulian Muresan. Temporal and Spatial Distributions of Primary

Delays in a High-Speed Rail System. In *2019 5th International Conference on Transportation Information and Safety (ICTIS)*, pages 399–405, Liverpool, United Kingdom, July 2019. IEEE.

[WS18] Maximilian Wirth and Andreas Schöbel. Mindestzugfolgezeiten bei ETCS Level 2 und Level 3 auf der Wiener S-Bahn-Stammstrecke. *Signalling+ Datacommunication (112)*, 4(2020):21–26, 2018.

[WYI+22] Liwei Wang, Suraj Yerramilli, Akshay Iyer, Daniel Apley, Ping Zhu, and Wei Chen. Scalable Gaussian Processes for Data-Driven Design Using Big Data With Categorical Factors. *Journal of Mechanical Design*, 144(2):021703, February 2022.

[YGH02] J. Yuan, R.M.P. Goverde, and I.A. Hansen. Propagation Of Train Delays In Stations. *WIT Transactions on The Built Environment*, 61, 2002.

[YHP+19] Yuxiang Yang, Ping Huang, Qiyuan Peng, Jie Li, and Chao Wen. Statistical delay distribution analysis on high-speed railway trains. *Journal of Modern Transportation*, 27(3):188–197, September 2019.

[Yua06] Jianxin Yuan. *Stochastic Modelling of Train Delays and Delay Propagation in Stations*. Eburon Uitgeverij B.V., 2006. Google-Books-ID: YMQ4kXBx2EUC.

[ZBB+19] Markus Zinser, Torsten Betz, Maurice Becker, Michael Geilke, Carla Terschlüsen, Adam Kaluza, Ingrid Johansson, and Jennifer Warg. PRISM: A Macroscopic Monte Carlo Railway Simulation. *The 12th World Congress on Railway Research (WCRR), Tokyo, Japan, October 28-November 1 2019*, 2019.

[ZW17] Enlu Zhou and Di Wu. Simulation Optimization Under Input Model Uncertainty. In Andreas Tolk, John Fowler, Guodong Shao, and Enver Yücesan, editors, *Advances in Modeling and Simulation: Seminal Research from 50 Years of Winter Simulation Conferences*, pages 219–247. Springer International Publishing, Cham, 2017.