Leveraging LightGBM for Accurate Train Turnover Time Prediction

A Thesis in Collaboration with Nederlandse Spoorwegen

Author:
Stephan Berende

First Examiner: Prof. dr. A.P.J.M. Siebes
Second Examiner: Dr. A.J. Feelders
Daily Supervisors: Johannes Dijkstra, Judith Helgers

Utrecht University

Applied Data Science

0.1 Abstract

This research aims to enhance the prediction of train turnover times using machine learning techniques, specifically focusing on data from the Nederlandse Spoorwegen (NS). The primary objective was to develop a model that improves upon the existing planning by the NS, thereby increasing operational robustness. Data collection and preprocessing were followed by the implementation of a LightGBM model, which outperformed the baseline by approximately six seconds in Mean Absolute Error (MAE) and ten seconds in terms of Root Mean Squared Error (RMSE). Despite these promising results, the model was built using data that excluded delayed and exceptionally long turnovers. Consequently, the model is best suited for providing quick estimates of turnover times for typical scenarios, while supplementary domain knowledge remains necessary for accurately predicting edge cases.

Contents

Chapter 1

Introduction

In this chapter, an introduction to the topic is given by providing context. Furthermore, a literature review is conducted and the research questions are elaborated upon.

1.1 Motivation and Context

Rail transport in the Netherlands has a ridership of over 438 million per year, with passengers travelling over 17.1 billion kilometers per year [11]. It is undoubtedly a cornerstone of public transportation in the Netherlands by providing efficient and reliable mobility for all these passengers on a daily basis. In this context, it is critical to be aware of the robustness of the train schedules to maintain punctuality and minimize delays. Of all the different operators in this network, Nederlandse Spoorwegen (NS) is the biggest. Thus, this thesis focuses on improving the NS train timetable by trying to predict the turnover times of trains under varying conditions more precisely.

Turnover times are the period required for a train to switch directions at a station. Factors influencing the turnover times are diverse and include for example the type of the rolling stock, whether a driver change is performed during the turnover, the length of the train and included buffers. Being able to predict this time better enhances the robustness of the train schedule, leading to fewer delays. Additionally, the ability to predict turnover times more precisely can lead to unforeseen events being better handled, such as technical issues or unexpected delays. With a more reliable schedule, NS can respond more dynamically to such disruptions.

Moreover, the methodologies developed in this research have the potential to be applied beyond NS and the Dutch rail networks. Other railways operator and different forms of public transportation could benefit from similar predictive models.

Using machine learning models, this research uses advanced analytical methods that can uncover patterns that are not easily detectable through more traditional approaches. The need of NS timetable planners for further insights and tools for developing robust schedules is what motivated this research.

1.2 Literature Overview

This section gives the reader more context and knowledge on what turnovers exactly are, how they influence timetable planning and different methods that exist or can be experimented with to improve the turnover times to reflect reality better and improve the timetable planning of the train network.

1.2.1 Turnovers defined

As mentioned in the Introduction a turnover time is the period required for a train to switch directions at a station, other names for this occurrence are turnbacks, station reversals, short-turns or runaround. Each of these names results in the same final result, a train changes directions into the opposite direction it went, but each with a different method.

What we call turnovers are generally called turnbacks in the international environment. In Dutch, it is called "kopmaken en omlopen" which consists of the train stopping and afterwards leaving in the direction it originally came from. In the Netherlands, this usually happens by having the machinist deactivate the front train cabin (the locomotive), walking to the train cabin on the other side of the train and activating it. Afterwards, the train can be pulled in the direction it originally came from. This happens at final stations, which in Dutch is called "kopstations", where the traintrack ends, but it can also occur in other stations due to that part of the track ending there[20].

The time this process takes can vary depending on a variety of factors such as the station, the type of train, the length of the train, whether cabins get attached or detached. Without any delays, the time generally planned for turnovers varies between 2 minutes at the fastest, for the quickest trains with a change driver "wisselmachinist" to almost 14 minutes, for a very long train that also is attaching or detaching cabins. The minimal turnover time at NS under different circumstances are visible in Section 6.1 of the Appendix.

1.2.2 The timetable problem

The problem of timetabling for railways has had quite a few studies and solutions proposed for it. Research generally falls into two categories: cyclical and non-cyclical timetables. Cyclical timetables have the advantage of being easy to remember for passengers. However, because they operate the same way during both peak and off-peak hours, they can lead to increased operating costs.[8].

Non-cyclical timetable scheduling decisions are based on evaluating the benefits of running various types of services at specific times[24]. This results in the planner having to decide whether a time or an event is populous enough to warrant public transportation. As one might imagine, this involves the planner having to keep a very expansive view of the happenings of every location and the flow of people in the area they are planning for. This can be an expensive process, especially so for large cities with lots of events happening at any time and place. Still, it may also result in a very minimal deviation of the timetable if said resources are present [9].

Given that trains are a form of public transport aimed at being easily accessible and capable of transporting many passengers, most countries opt for a cyclical timetable. This approach maximizes accessibility and comprehensibility while minimizing daily adjustments.

The Dutch Timetable

At the end of 2006, the Netherlands introduced a completely new train timetable, replacing the one from 1970. This change led to an increase in passenger numbers and improved the punctuality of train arrivals and departures, while also accommodating future growth of the transportation system[16].

To create this new timetable, the problem was initially modeled by developing a cyclical timetable for a single day, which was then extended to an entire week by replicating it. For the daily timetable, a variety of scheduling models and techniques were used. To define the scheduling problem, constraints were set up using a Periodic Event Scheduling Problem(PESP). To solve the smaller instances of these timetable constraints, Mixed Integer Programming(MIP) was applied to solve the problem as a mathematical formula and find the optimal solution. To solve all the instances and make sure that all the constraints line up to a single planning, a special module of CADANS was developed. CADANS is based on constraint programming techniques. It was used to solve, what in essence is a giant mathematical formula, in acceptable computing time while still adhering to the initial PESP constraints. This feasible solution was then optimised for the final timetable[16].

With some adjustments, this is still the timetable that the Netherlands is currently using as a basis for its weekly cyclical operation. These adjustments were made by taking into account data from conductors, passenger counts, OV-chipcard data, crowding reports, forecasts of passenger growth, and holiday periods. Using this data, the number, size and types of trains in the timetable are adjusted to fit the current needs of the passengers [2].

Timetable optimisation techniques

The research on the timetable optimisation problem can be differentiated on what the objective function to be optimised is. Some objectives functions of existing research are: maximising profit, minimising deviation, minimising travel time, or maximising the number of people transported. Nevertheless, it sometimes comes down to looking at a certain situation in regards to its infrastructure capacity and finding the "best" solution for said situation manually [25].

Research aimed at minimizing deviations from the timetable while largely retaining existing infrastructure and planning involves implementing new techniques and strategies. These approaches either compensate for delays by adapting the timetable in real-time or modify conditions to reduce delays.

Real-time methods require extensive data and a system capable of adapting the timetable to meet current demand, thereby preventing potential deviations. This data can include smart card information from passengers, which helps identify travel patterns and predict congestion by analyzing time, location, and passenger profiles. Such a dynamic timetable design shows significant potential and could be an effective method for improving robustness.[22].

Another method to make turnover times more robust is stop-skipping. This allows trains to skip some stations when the deviation time is relatively high [10], decreasing the overall time loss.

Larger structural changes to the railway network such as allowing for unfixed platform time and flexible rail track allocation can improve the capacity of turnover operations thereby reducing delays [14]. These kinds of larger changes, which involve the physical world, have proven to be effective before. Nevertheless, for the costs and time allocation that they involve, they may not always be feasible to achieve. There are plenty of alternative methods which are more cost-effective and require less investment. One such method is utilising machine learning to optimize the existing railway operations.

1.2.3 Machine learning for timetable optimisation

In this study, the goal is to minimize the deviation between the expected turnover time and the actual turnover time. Thereby decreasing delays and improving the timetable to be more robust. Some restrictions however are that the current timetable cannot be adjusted too much (e.g. no changing the rolling stock types or amount of carriages).

By their admission, the amount of data analysis methods that the NS employs to improve their turnover times is quite low and that is also where this thesis comes into play. As the objective is to minimize deviation, an accurate prediction of turnover time must be made based on the specific circumstances. To solve this problem, a model can be created that, given data on the situation, returns the minimal turnover time needed or an estimation of whether the planned turnover time is correct for the circumstances. In short, the problem can be solved through regression or classification. The problem is most suited to a regression model, however as the NS already has a workflow to select these turnover times, a classification model may also be useful in a more supportive role.

1.2.4 Regression: Model-Specific Literature

Gradient Boosting Decision Trees (GBDT) comprise a category of machine learning techniques that have found widespread usage across various domains, including regression problems such as turnover prediction[21][23]. Despite the recent rise of Artificial Neural Networks (ANN), boosting methods still offer state-of-the-art performance in many regression benchmarks[6]. Moreover, this category of models provides an exceptional balance between accuracy and computational efficiency, making it well-suited for this short research project.

Gradient boosting is an ensemble strategy that aims to create an accurate classifier by iteratively building upon several weak classifiers. Within the family of GBDT, three state-of-the-art packages exist. These are XGBoost, LightGBM and CatBoost. Of the many GBDT implementations available, each with its advantages, these are by far the most popular among them[13].

XGBoost, the oldest and most widely used, has proven its effectiveness in numerous ML competitions and benefits from extensive documentation due to its popularity. LightGBM is supposed to be the successor to XGBoost, claiming to be more accurate with much shorter training times[15]. While the latter is often found to be true, XGBoost may still exhibit superior performance in certain prediction tasks. CatBoost represents a newer entrant in the GBDT landscape and excels particularly with datasets containing many high-cardinality categorical features[18]. In such scenarios, CatBoost often outperforms both XGBoost and LightGBM in terms of both predictive accuracy and training efficiency. In reality, none of these methods is objectively better than the others. As demonstrated in the comparative analyses by Shyam et al.[19] and Al Daoud[3], the effectiveness of each is highly dependent on the given input data.

Hancock and Khoshgoftaar [12] observe that the impact of hyperparameter tuning on GBDT performance can also vary greatly, being either negligible or quite significant, depending on the specific dataset. They also highlight the complexity of hyperparameter optimization as a common challenge across these GBDT models. While generally time-efficient, it remains challenging due to the numerous co-dependent parameters involved.

1.3 Research Question

The primary objective of this research is to explore the potential of machine learning techniques to predict the realized turnover times of trains at NS. Thus, the research question that guides this study is:

How can machine learning methods help NS to predict train turnover times and thereby increase robustness?

In the context of railway operations, turnover time is a metric that affects the overall efficiency and punctuality of the trains. It refers to the period of time that is required for a train to switch directions at a station. Accurate predictions of this are required for developing robust train schedules and minimizing delays.

The turnover times can be influenced by a multitude of factors, including the type of rolling stock, driver changes, train length and buffer times in the schedule. Traditional methods of predicting turnover times may not fully capture the complexity of these factors. Therefore, there is a need to employ more advanced predictive models that can handle such complexities.

Machine learning in general offers a promising approach to address this challenge. Machine learning models can learn complex patterns from large datasets that might otherwise not be fully captured by more traditional methods, which makes them well-suited for predicting outcomes that are influenced by multiple interdependent variables. By employing such a model this research aims at accurately predicting these realized turnover times based on the provided historical data.

The research objectives are as follows:

1. **Data Collection and Preparation:** Gathering and preprocessing the historical data from NS.
2. **Feature Selection:** Identifying and selecting relevant features that influence turnover times, such as the rolling stock type, driver changes etc.
3. **Model Development:** Developing a machine learning model to predict the realized turnover times.
4. **Model Evaluation:** Assessing the performance of the model using appropriate evaluation metrics to ensure its accuracy and reliability.
5. **Recommendations:** Providing insights and recommendations for NS timetable planners to improve schedule robustness based on the model's predictions.

Chapter 2

Data

In this chapter, the data used for the model is discussed. This includes an overview of the data collection and description, a detailed explanation of the preprocessing steps taken, and an exploratory analysis of the data.

2.1 Data Collection & Description

The data used in this research is sourced from the NS databases, and exported as CSV files from the server. The dataset contains detailed historical records of train activities, including arrivals and departures at different stations. The dataset spans a period of one year, form January 1st 2023 to December 31st 2023, providing a comprehensive view of train activities.

Ensuring the reliability of the data is fundamental for the predictive model, as it needs to represent real-world turnover times accurately. The data from NS is considered highly reliable as it is systematically recorded and maintained by the organization on their servers. NS follows strict data collection and validation protocols to ensure that the recorded information accurately reflects realized operations. Despite these measures, preprocessing and validation are necessary to address any inconsistencies that may still arise, ensuring data integrity for accurate modeling.

As the dataset contains only operational and logistical information, no ethical considerations regarding personal information had to be made. As NS wished their data to be kept from the public, the dataset was stored locally and safely.

The dataset consists of 28,233,618 rows and 18 columns. Each row represents a train activity, such as an arrival or a departure from a station. The content and description of each column is presented in Table 1.1. As can be seen in the table, turnover times are not included in the dataset. This needs to be calculated based on the available data.

Table 2.1: Description of variables

Column	Description	Type of variable	Unit
TRAFFIC_DATE	Date of the activity	Nominal	-
TRAINNUMBER	Number of the train	Nominal	-
TRAINSERIE	Series of the train	Nominal	-
TRAINSERIE_DIRECTION	Direction of the train	Nominal	-
STATION	The station where the activity takes place	Nominal	-
ACTIVITYTYPE	The type of activity	Nominal	-
DISTANCE_M	The distance traveled since the last activity	Ratio	Meters
PLAN_DATETIME	The planned date and time of the activity	Ratio	YYYY-MM-DD HH:mm:ss.sss
REALIZED_DATETIME	The realized date and time of the activity	Ratio	YYYY-MM-DD HH:mm:ss.sss
DELAY	The time between planned and realized time	Ratio	Seconds
TURNOVER_INDICATOR	Indicates wether a turnover took place	Binary	-
PREVIOUS_TRAINNUMBER	The number of the previous train, if a turnover took place with a change of trainnumber	Nominal	-
COMBINE	Indicates wether the train combined with another train into a train with more carriages	Binary	-
SPLIT	Indicates wether the train split into multiple trains with fewer carriages	Binary	-
ROLLINGSTOCK_TYPE	The type of rolling stock	Nominal	-
NUMBER_CARRIAGES	The number of carriages	Ratio	Carriages
DRIVER_CHANGE	Indicates wether a change of driver took place	Binary	-
DEPARTURE_SIGNAL_SHOWS_SAFE	The date and time when the train was signalled it could safely depart	Ratio	YYYY-MM-DD HH:mm:ss.sss

2.2 Preprocessing

Data preprocessing is a critical step ensuring the data is ready for analysis. The preprocessing pipeline ensures that the data is clean, consistent, and ready for modeling. Most importantly the variable of interest, turnover time, is calculated.

2.2.1 Data Type Setting

The first step involves setting the correct data types for each column in the dataset. This ensures that following operations and calculations are performed correctly and efficiently. Columns with dates and times are set to datetime variables, remaining numerical variables are set to integers, and categorical variables are set to strings. Next, the data is sorted by train number, date, and time.

2.2.2 Turnover Time Calculation

Calculating turnover times is a complex process as there are two situations in which a turnover happens, with each a unique way of being recorded in the dataset:

1. **End of route turnover:** This happens when a turnover is performed at the end of a route and the train changes train number as it starts its new route. Here the "TURNOVER_INDICATOR" is set to 1, and the train number it had before performing the turnover is registered in the "PREVIOUS_TRAINNUMBER" column. In this case, the turnover time is the interval between the last arrival of the previous train number and the first departure of the new train number.

2. **Mid-route turnover:** This happens when a turnover is performed in the middle of a route. Here, the "TURNOVER_INDICATOR" is also set to 1, but there is no "PREVIOUS_TRAINNUMBER" since the route doesn't change. The turnover time is calculated based on the last arrival at the same station of the same train.

In both cases, the planned turnover time and the realized turnover time are calculated based on the "PLAN_DATETIME" and "REALIZED_DATETIME" columns.

2.2.3 Data Cleaning

As a next step, the data is cleaned. This consists of the removal of data that is not relevant and outliers. Only records of turnovers are kept, all other instances are filtered out. This still leaves some records that are not relevant however, such as a few instances of German trains that are not representative for this analysis and activity types that are not of interest. Outliers in turnover time and delays were filtered out as well. Turnover times lower than the minimum of 2 minutes or higher than 40 minutes were removed. Trains that departed 10 minutes early or 3 minutes late were also removed. Further removal of outliers or edge cases was performed to improve model performance, these preprocessing steps can be found in Section 6.1 of the Appendix.

2.2.4 Feature Engineering

Lastly, data transformation in the form of feature engineering was performed. The cumulative distance travelled is calculated based on distances between previous stations and the day of the week and hour of the day are extracted from the planned datetime.

2.2.5 Dataset After Preprocessing

As only records of turnovers are kept, the amount of rows in the dataset is greatly reduced from the original 28,233,618 to 769,800. While the amount of columns has increased from 18 to 21 due to the feature engineering. As some of these columns may not be relevant to the model, the amount of columns is further narrowed down during feature selection for the ML-model. Table 2.2 shows the columns of the preprocessed dataset and their data types.

Table 2.2: Preprocessed dataset columns and their data types

Column	Description	Data Type
TRAFFIC_DATE	Date of the activity	datetime
TRAINNUMBER	Number of the train	int
TRAINSERIE	Series of the train	int
TRAINSERIE_DIRECTION	Direction of the train	string
STATION	The station where the activity takes place	string
ACTIVITYTYPE	The type of activity	string
DISTANCE_M	The distance traveled since the last activity	int
PLAN_DATETIME	The planned date and time of the activity	datetime
REALIZED_DATETIME	The realized date and time of the activity	datetime
DELAY	The time between planned and realized time	int
TURNOVER_INDICATOR	Indicates whether a turnover took place	int
PREVIOUS_TRAINNUMBER	The number of the previous train, if a turnover took place with a change of trainnumber	int
COMBINE	Indicates whether the train combined with another train into a train with more carriages	int
SPLIT	Indicates whether the train split into multiple trains with fewer carriages	int
ROLLINGSTOCK_TYPE	The type of rolling stock	string
NUMBER_CARRIAGES	The number of carriages	int
DRIVER_CHANGE	Indicates whether a change of driver took place	int
DEPARTURE_SIGNAL_SHOWS_SAFE	The date and time when the train was signalled it could safely depart	datetime
CUM_DISTANCE_M	The cumulative distance travelled before performing the turnover	int
DAY_OF_WEEK	The day of the week (Monday = 0, Sunday = 6)	int
HOUR	The hour of the datetime	int

2.3 Exploratory Data Analysis

Data exploration has been an integral part of the aforementioned steps. Visualisation guided the creation of the prepossessing pipeline and subsequently helped identify key features in the modelling phase. This section begins by detailing the target variable of this study and then examines how some input variables relate to it. Due to space constraints, only the most notable findings will be highlighted. The full exploratory data analysis is available in Section 6.1 of the Appendix.

2.3.1 Target Variable

The value to be predicted in this study is "REALIZED_TURNOVER_TIME". As depicted in Figure 2.1, turnover times roughly follow an inverse-gamma distribution. In the initial phase of the project, it was determined that predicting turnovers longer than 2500 seconds (roughly 40 minutes) was neither feasible nor of interest. Therefore, the long tail of the depicted distribution is essentially cut off, resulting in a data loss of around 2 percent.

Further filtering was applied to remove turnovers which suffered a severe delay. As the planners at NS are unable to predict delays beforehand, the proposed machine learning models cannot utilise this data either. To account for this fact,

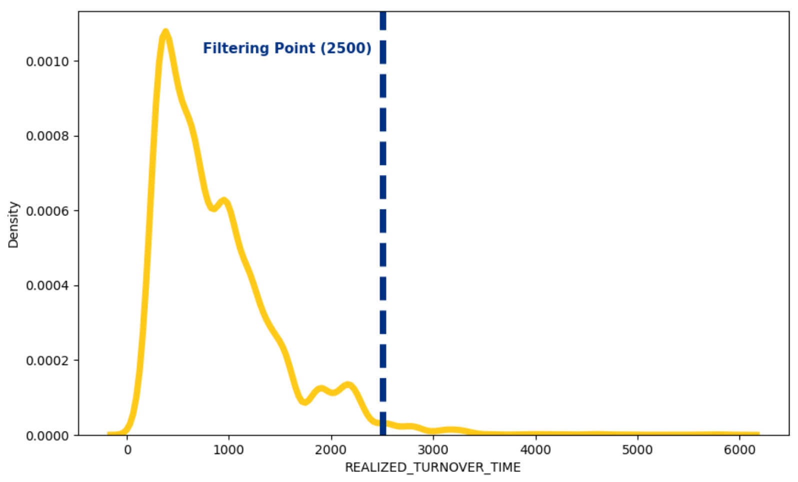

Figure 2.1: Distribution of realized turnover time in seconds

trains which departed more than three minutes late or early were removed. This filtering results in an additional reduction of the original data by ten percent.

Figure 2.2 illustrates the significant impact of the aforementioned filtering steps on the target variable. This filtering results in shorter turnover times with minimal delays and significantly enhances the correlation between planned and realized turnover times.

2.3.2 Input Variables

The input variables mentioned in Section 2.1 are known by the NS to affect turnover time in various degrees. As an example, Figure 2.3 demonstrates how turnover times vary across different types of rolling stock. The data exploration suggests that factors like rolling stock types, driver changes and the number of carriages are particularly relevant for short turnovers but become less significant as turnover times increase.

Combining or splitting has a significant effect as they increase turnover time by an average of three minutes. Still, such operations occurred in less than five percent of the data.

Perhaps one of the most important input variables is the station at which a turnover occurs. Only a third of Dutch stations facilitate turnovers, with the ten largest accounting for more than half of them.

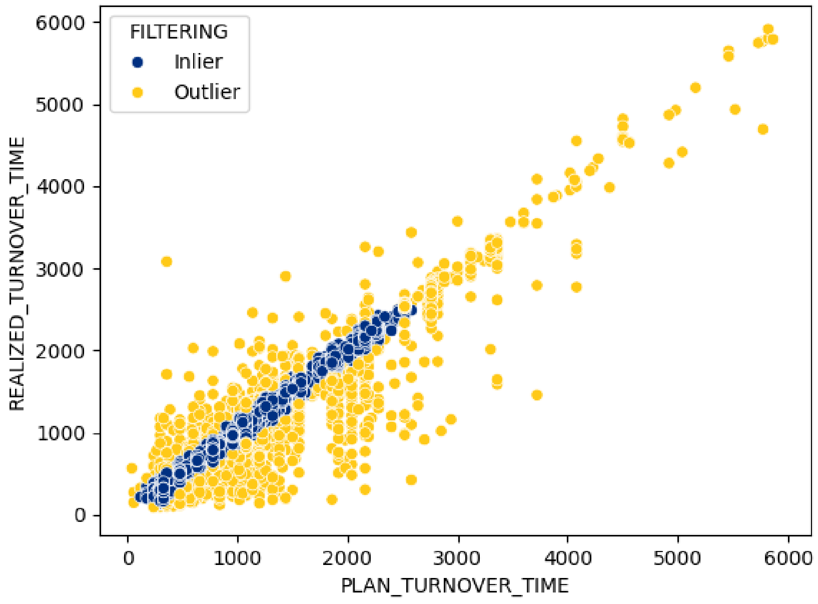

Figure 2.2: Scatterplot showing the relationship between planned and realized turnover times, with outliers and inliers separated

While most turnover stations follow the long-tailed distribution of turnover time, seen earlier in Figure 2.1, some significantly deviate from this pattern with quite irregular distributions. These discrepancies are clearly visible in Figure 2.4. According to NS, such differences are caused by the layout (e.g. track configuration, nearby shunting yard size) and location of the station. Although valuable data may be embedded within these discrepancies, such station-specific data is also difficult to generalise which is why it is not included within this short project.

A less obvious variable affecting turnover time is the hour of the day, shown in Figure 2.5. Few turnovers occur at night, while most occur during the evening rush hour. Interestingly, turnovers are shorter on average during these busy hours, suggesting that turnovers outside of rush hours have additional time-buffers on top of the minimal required time. The varying frequency of driver changes and combining or splitting operations throughout the day may explain some of these changes in turnover times at each hour.

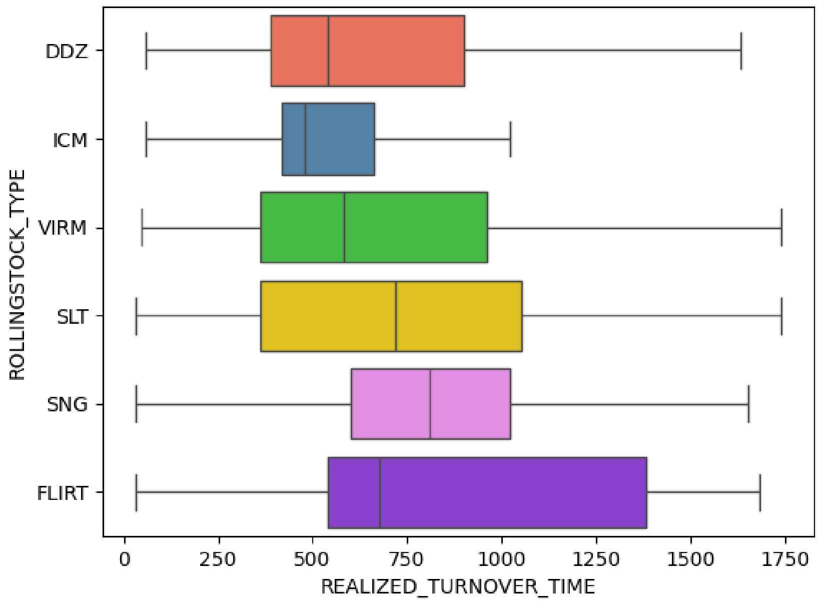

Figure 2.3: Realized turnover times for different types of rolling stock

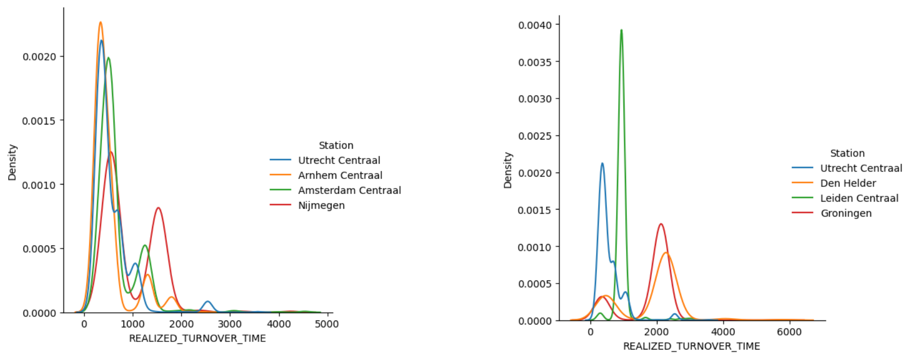

Figure 2.4: Distribution of turnover times at various stations categorized into common distribution patterns (left) and irregular distributions (right), with Utrecht Centraal for reference.

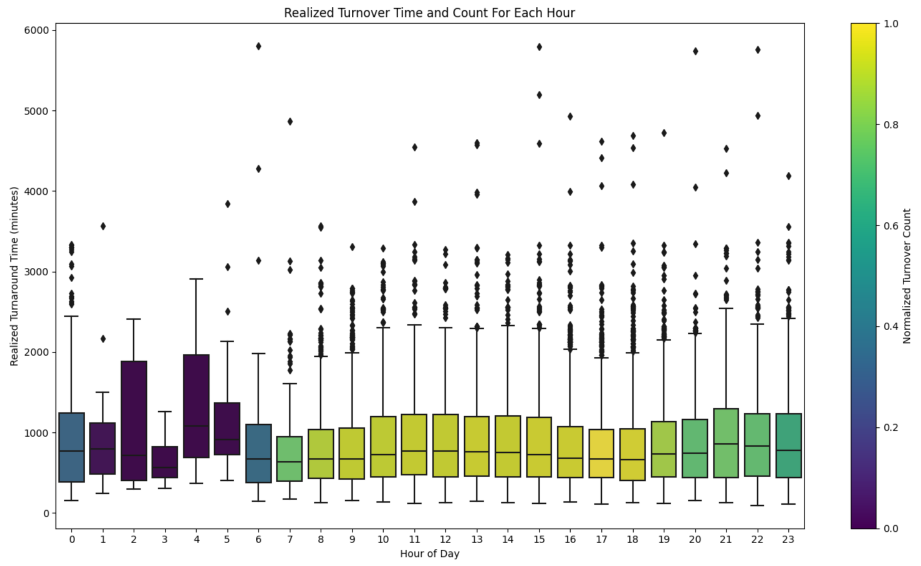

Figure 2.5: Realized turnover time and normalized turnover count by hour of the day

Chapter 3

Methods

In this chapter, the methods used in this study will be discussed. This includes translating the research question to a data science question, selecting a method to use for further analysis, and establishing settings for the selected method.

3.1 Translation of the research question to a data science question

The research question **"How can machine learning methods help NS to predict turnover times and thereby increase robustness?"** highlights the need to apply advanced analytical techniques to improve the railway operations of NS. To address with within a structured framework, the research question is translated into a more focused data science question.

To translate it, the core components are first identified

1. **Objective:** The primary goal is to accurately predict realized turnover times.
2. **Outcome:** The desired outcome is to improve the robustness of train schedules by leveraging these predictions.
3. **Methods:** This should be achieved using machine learning techniques.
4. **Context:** The focus is on the operations of NS, a Dutch railway company.

Next, the following points need to be considered:

- **Data Requirements:** Identifying the data necessary for predicting train turnover times, such as historical turnover times and schedule data.

- **Features and Variables:** Determining which features significantly influence train turnover times and should be included in the model.
- **Model Selection:** Selecting an appropriate machine learning algorithm
- **Evaluation Metrics:** Defining the metrics to measure the performance of the model such as accuracy or Mean Absolute Error.
- **Implementation:** Determining how the model can be integrated into NS's operations.

This leads us to our translated data science question:

How can we develop and implement a machine learning model to accurately predict train turnover times at NS, using historical operational data, and how can these predictions be utilized to enhance the robustness of train schedules?

3.2 Selection of Methods for Analysis

This section discusses the ML-model that will be used to predict turnover times. Furthermore, the data, and optimization strategy to be employed are also elaborated on.

3.2.1 Model & Data

LightGBM was chosen to predict turnover times. As discussed in the literature overview, this is the most time-efficient GBDT package across a wide array of datasets, while still offering competitive accuracy scores. This is particularly relevant for this project, where the input data is large and the available computing power is limited. Although CatBoost may also be well-suited for this dataset due to the high proportion of categorical variables, LightGBM's recent addition of integrated support for categorical features ultimately led to its selection. Conducting a thorough analysis of multiple GBDT packages was beyond the scope of this project.

The input variables for the LightGBM model can be seen in Table 3.1. Categorical features are label encoded as integers, while numerical values are set to type float. Furthermore, the "DAY_OF_WEEK" and "HOUR" variables are temporal values that require additional preprocessing. Since temporal variables are cyclical in nature, a sine and cosine transformation is applied to effectively capture their cyclic properties[17].

Table 3.1: Input variables for the LightGBM model

Input Variable	Description	Data Type
STATION	The station where the turnover took place	int
COMBINE	Indicates whether carriages were added	int
SPLIT	Indicates whether carriages were removed	int
ROLLINGSTOCK_TYPE	The type of rolling stock	int
NUMBER_CARRIAGES	The number of carriages	int
DRIVER_CHANGE	Indicates whether a change of driver took place	int
CUM_DISTANCE_M	The cumulative distance travelled before performing the turnover	float
DAY_OF_WEEK_sin	Sine transformation of day of the week	float
DAY_OF_WEEK_cos	Cosine transformation of day of the week	float
HOUR_sin	Sine transformation of hour of day	float
HOUR_cos	Cosine transformation of hour of day	float
PLAN_TURNOVER_TIME	The planned time for a turnover	float

3.2.2 Model Split

Initial models struggled to accurately predict the full range of turnover times. Accurate predictions for long turnovers often came at the cost of poor performance for short turnovers, and vice versa. To address this issue, the dataset was split into four time intervals, with a separate model created for each time-split:

1. **Short Turnovers:** Turnovers taking from 2 to 10 minutes, representing 45 % of the dataset.
2. **Medium Turnovers** Turnovers taking from 10 to 20 minutes, representing 38% of the dataset.
3. **Long Turnovers** Turnovers taking from 20 to 30 minutes, representing 15% of the dataset.
4. **Very Long Turnovers** Turnovers taking from 30 to 40 minutes,representing 2% of the dataset.

For each data split, a separate LightGBM model is trained and optimized. Equation 3.1 is then used to estimate the combined performance across all four splits. This splitting approach not only enhances performance but also increases model interpretability. It clarifies how the model performs across different time intervals of turnovers.

$$\begin{aligned}\mathrm{MAE}_{\text{total}} = &\left(\mathrm{MAE}_{\text{small}} \times \frac{N_{\text{small}}}{N_{\text{total}}}\right) + \left(\mathrm{MAE}_{\text{middle}} \times \frac{N_{\text{middle}}}{N_{\text{total}}}\right) \\ &+ \left(\mathrm{MAE}_{\text{large}} \times \frac{N_{\text{large}}}{N_{\text{total}}}\right) + \left(\mathrm{MAE}_{\text{very long}} \times \frac{N_{\text{very long}}}{N_{\text{total}}}\right)\end{aligned} \tag{3.1}$$

3.2.3 Model Optimization

The error metric to be optimized in this study is the Mean Absolute Error (MAE), with the Root Mean Squared Error(RMSE) also being recorded for analysis. MAE provides the average error of predictions, whereas RMSE penalizes larger errors more heavily by squaring them. Apart from their distinctive purposes, these metrics were chosen for their intuitive nature. Their unit of the error scores matches the units of the target value being predicted (absolute seconds), unlike the Mean Squared Error(MSE). This is an important criterion since the study results may be presented to employees of NS who might not have a background in machine learning.

To optimize the aforementioned metrics, the parameters of each of the four models are tuned. Holding true to the saying that GBDT parameters can be difficult to grasp, LightGBM has over 100 co-dependent parameters available for such tuning. While it is not necessary to optimize all of them, tuning can still have a significant impact on performance, as seen in the literature overview. The Optuna hyperparameter tuning framework is employed to speed up this complex optimization process. Optuna applies a form of bayesian optimization which is much more time-efficient, while often leading to even better results than standard grid search or randomized search approaches[4].

3.2.4 Baseline Model

To properly assess the performance of the LightGBM model, a baseline score must be established. The most straightforward method is to use the current planned turnover time at NS for this purpose. By subtracting the planned turnover time from the realized turnover time, we can evaluate the performance of the existing planning. This evaluation results in a baseline MAE of **42.11 seconds** and RMSE of **56.70 seconds**.

3.3 Settings for the selected Method

For each of the aforementioned data splits, a separate LightGBM model was optimized. This optimization involved training 800, 5-fold models for each of the four splits. Each model was split into 5 folds to ensure consistent performance with varying input data. An 80-20 training-testing split ratio was used, with a further 20 percent of the training data set aside for validation to expedite hyperparameter tuning.

The grid of parameter values to be explored, shown in Table 3.2, was established through official documentation[1], online guides[5][7], and personal experimentation. To enhance performance, each model was trained on more trees (*n_estimators*) with a slower convergence speed (*learning_rate*) than the default LightGBM model. The *early_stopping* parameter was used to terminate models which had not improved for 50 consecutive trees, thereby reducing training time (utilises validation data). These three parameters, which concern the training process itself, were optimized once on the full dataset and then fixed for the remaining optimization process.

PARAMETER	**DESCRIPTION**	**MIN**	**MAX**
n_estimators	Amount of trees trained for each model	500	500
learning_rate	Speed of model convergence	0.01	0.01
feature_fraction	Randomly selects percentage of features for training	0.8	1.0
bagging_freq	After how many trees new bagging happens	0	12
bagging_fraction	Randomly selects percentage of rows for training	0.8	1.0
max_depth	Maximum depth of the tree	3	16
num_leaves	Maximum amount of leaves of tree	8	512
min_data_in_leaf	How much data must minimally be in a leaf	10	1000
min_gain_to_split	How much information must minimally be gained for split	0	20
lambda_l1	Degree of l1 regularization	0	100
lambda_l2	Degree of l2 regularization	0	100

Table 3.2: LightGBM parameter grid which was optimized using Optuna

The remaining 9 parameters either control tree growth, tree shape, regularization, or the degree of data sampling. With the large amount of optimization models trained, the parameter ranges could be kept relatively broad.

As for the settings for the optimization process itself, Optuna was used with the default *TPEsampler*, which is an implementation of bayesian optimization. An objective function with the aim of minimizing the MAE was passed to Optuna.

Chapter 4

Results and Analysis

In this chapter, the results of the modelling and optimization process will be displayed and briefly analyzed.

4.1 Initial Models

Table 4.1 shows the different models considered in this study, listed in ascending order of performance. The default LightGBM model, without any data splits, already outperforms the baseline. However, the most significant performance gain is achieved by splitting the turnover time into four segments, resulting in an improvement of more than six seconds when compared to the baseline. The effect of tuning was relatively small, further increasing performance by an additional second.

The residuals of the tuned 4-Split model are compared to those of the baseline model in Figure 4.1. As evident from the plot, the baseline model errors are well within three minutes, while the 4-Split model has some outliers beyond this mark. Nevertheless, the RMSE of the 4-Split model is still significantly smaller, indicating that on average, most errors are less extreme than those of the baseline model.

MODEL	**MAE**	**RMSE**
NS Baseline	42.11	56.70
No-Split Model	39.68	53.47
4-Split Model	35.77	48.37
4-Split Tuned	34.50	46.12

Table 4.1: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) in seconds, for different models

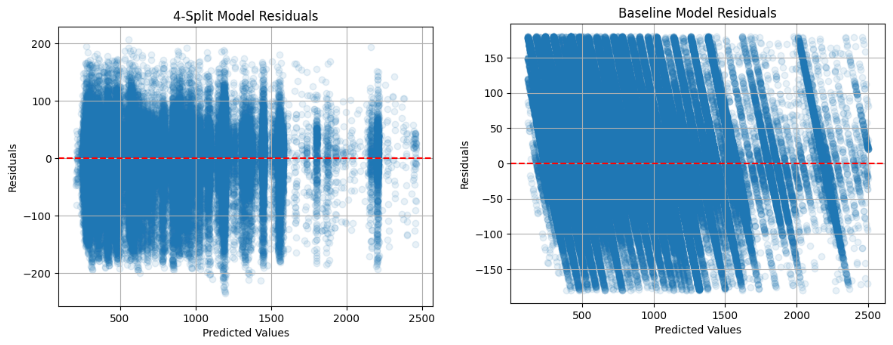

Figure 4.1: Residuals for tuned 4-Split model (left) and baseline model (right)

4.2 Optimization Process

The 4-Split model demonstrated the best performance and was therefore selected for optimization. The optimization histories along with parameter importances and parameter distributions for each split are detailed in Section 6.2 of the Appendix.

To summarize, most splits converged on optimal parameters after the first few hundred runs. The last 400 optimization models of each split resulted in only marginal improvements of a few milliseconds at most, rendering them largely redundant. The most important parameters across all splits were related to the growth and shape of the decision trees. Generally, the best-performing parameters resulted in expansive trees, accompanied by higher lambda values to mitigate the risk of overfitting with these large trees.

While the sampling of observations was relevant for some splits, the sampling of features almost always led to worse results. Presumably, this is caused by planned turnover time and travelled distance having high feature importance. Excluding even one of these two would lead to a large drop in performance, resulting in Optuna viewing feature sampling as an unattractive parameter.

4.3 Tuned Model

The tuned 4-Split model was found to be the most optimal compared to the baseline, and will therefore be discussed in more detail. Table 4.2 depicts the MAE for each split of this 4-Split model before and after optimization. The RMSE is omitted from the table as the pattern is nearly identical.

After optimization, performance increased by approximately one second. Interestingly, the model for short turnovers has the worst performance while it

SPLIT	**DATA**	**MAE**	**MAE TUNED**	**GAIN**
Short Turnovers (120-600 seconds)	45%	38.74	38.57	-0.17
Middle Turnovers (600-1200 seconds)	38%	33.03	32.91	-0.12
Long Turnovers (1200-1800 seconds)	15%	33.75	33.34	-0.41
Very Long Turnovers (1800-2400 seconds)	2%	36.81	36.49	-0.32

Table 4.2: MAE in seconds before and after tuning for each split of the 4-Split model

represents almost half of the total data. Middle turnovers represent almost 40 percent of the data and also had the best performance. Still, both long and very long turnover models performed only slightly worse with much less data. Additionally, these models saw the largest increase in performance after parameter optimization. Since 'middle' and 'short' turnovers account for more than 80 percent of the data, the performance of these sub-models is still the most crucial consideration.

The combined feature importances for the tuned 4-Split model, extracted from LightGBM, are presented in Figure 4.3. Detailed feature importances for individual splits can be reviewed in Section 6.3 of the Appendix. These figures reveal that the planned turnover time and cumulative distance travelled are the most important features by a significant margin. Station, rollingstock type, and the hour of the day also play notable roles, albeit with considerably smaller importance. Interestingly, features related to combining, splitting, and driver changes did not show as much importance in any of the models, despite being recognized by the NS as influential for turnover time. This discrepancy likely arises because these operations, while important, occur in only a small percentage of the turnovers.

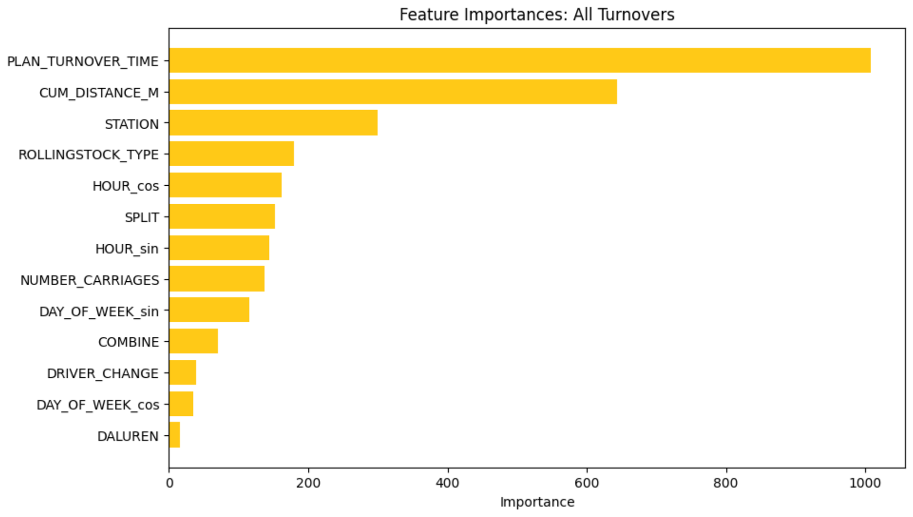

Figure 4.2: Combined feature importances of 4-Split model

Chapter 5

Conclusion and Discussion

This final chapter concludes the findings of the study. First, it provides a practical answer to the data science question posed in the Methods section. This answer is then generalized to address the overarching research question. This second step requires careful consideration of the limitations and practical context of the research. The chapter ends with a brief conclusion of the thesis.

5.1 Data Science Question

The primary, 'data science' goal was to accurately predict turnover times using machine learning techniques, based on data from the NS. The study successfully addressed this purely practical research question. To start, data from the NS was gathered and preprocessed as necessary. LightGBM was then selected as the ML-model and further model-specific data preparation was carried out. A baseline was established using the current planning at NS. Finally, error metrics (MAE & RMSE) to be optimized were selected. The model which came out of this process, outperforms the baseline by approximately 6 seconds, indicating that the predictions are indeed accurate. The remaining task is implementation. While the results are promising, actual integration into day-to-day operations is up to the NS.

5.2 Research Question

In Section 1.3, the following research question was defined:

How can machine learning methods help NS to predict train turnover times and thereby increase robustness?

In theory, the predictions by the ML-model are even more robust than the current planning of the NS. However, due to filtering, the model only focuses on turnovers with minimal delays, never exceeding 40 minutes. The model performs well on this carefully selected dataset, even better than the planning of NS. Despite this, the most difficult turnovers to predict are those with very large delays, and long turnovers in general. The domain knowledge required for planning such unusual turnovers is complex and difficult to encode into a ML model. As such, the model cannot be expected to replace the planning department at NS altogether. Instead, it may serve as an assisting tool. Planners at NS can use this model to quickly obtain a rough estimate of the turnover time for a specific train. Domain knowledge can then be applied to decide if the suggested turnover time should be adjusted. As a result, planners do not have to determine turnover times from scratch, making the planning process more efficient for standard cases.

5.3 Discussion

While the ML-model performed well within the confines of this study, it is difficult to say to what extent the model may improve robustness in practice. Its role as a reliable 'assisting tool' may be limited by several factors:

- **Planned Time Feature:** Model performance may be artificially increased by using planned time as an input feature. In practice, the planned turnover time is not directly available when making a brand-new planning. Still, as the weekly planning is almost exactly cyclical, looking at earlier turnovers of the same train number may achieve a similar effect. Another alternative is utilising the minimal time required for various turnovers, which are also recorded by NS.

- **Model Split:** Building a separate model for different time intervals of turnover length improved performance significantly. Currently, each model is trained and tested on data from a specific interval of realized turnover time. In practice, realized turnover time is not known in advance, making it unclear to which model each observation should be assigned. Utilizing planned turnover time as a starting point could help, but developing a separate model to assign turnovers to the appropriate interval-specific model would be even better. In any case, performance is expected to decrease slightly.

- **Generalisation:** While LightGBM has proven to be an effective regression model, it is also known to be sensitive to data changes. As a result, performance may degrade when the model is presented with data from a new timetable. Retraining the model frequently may address this risk.

- **Data Quality:** Interestingly, expanding the dataset from one week to a full year only resulted in a negligible performance increase. With all outliers filtered out, this minimal improvement can likely be attributed to the lower data quality of the larger dataset. Since numerical features in the extensive dataset were found to be much noisier, applying smoothing techniques could already enhance data quality to some extent.

Even with these limitations in mind, the LightGBM model provides a strong foundation for estimating turnover times using predictive models. However, there is still considerable room for improvement. This improvement should begin by testing other GBDT models (such as CatBoost and XGBoost). A more impactful, yet more challenging performance gain can likely be achieved by including more relevant features. Ideally, these features would specifically help in predicting short turnovers, as the model struggled the most with these and they represent almost half of the dataset. While such variables impacting turnover are well-known (e.g. the number of passengers, track position, weather circumstances, and proximity to nearby shunting yards), these variables could not easily be translated into features in this short research project.

5.4 Conclusion

This thesis has demonstrated the potential of machine learning, particularly LightGBM, to predict train turnover times with a high degree of accuracy. The proposed machine learning model outperforms the existing planning at NS by approximately six seconds. This exceptional result could be achieved by creating several sub-models, each predicting different intervals of turnover times. Extensive hyperparameter tuning offered another small gain in performance. While the performance of this model is promising, several limitations may impede the performance in real-life applications. Most importantly, the model is trained to predict relatively successful turnovers while heavily delayed or lengthy turnovers are not considered. Other limitations are related to the study conditions being slightly more optimal than a real-life scenario may be. Nevertheless, even in its current state, the LightGBM model could assist planners at NS in quickly getting a rough indication of how long a 'normal' turnover should be, within acceptable error margins. For edge cases, additional domain knowledge can be applied to get a good estimate of turnover time. While small performance gains may be made through refinement of the current model, the inclusion of more descriptive features is likely the most optimal avenue for substantial performance gains.

Bibliography

[1] Parameters tuning — lightgbm. `https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html`, 2024. (Accessed on 07/01/2024).

[2] Planning van treinen—reisinformatis—ns. `https://www.ns.nl/reisinformatie/service-verbeteren/planning-van-treinen.html`, 2024. (Accessed on 28/06/2024).

[3] Essam Al Daoud. Comparison between xgboost, lightgbm and catboost using a home credit dataset. *International Journal of Computer and Information Engineering*, 13(1):6–10, 2019.

[4] Hussain Alibrahim and Simone A Ludwig. Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1551–1559. IEEE, 2021.

[5] Bahmani. Understanding lightgbm parameters (and how to tune them). `https://neptune.ai/blog/lightgbm-parameters-guide`, September 2023. (Accessed on 07/01/2024).

[6] Candice Bentéjac, Anna Csörgő, and Gonzalo Martínez-Muñoz. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54:1937–1967, 2021.

[7] Tom Bex. Kaggler's guide to lightgbm hyperparameter tuning with optuna in 2021 by bex t. `https://shorturl.at/s1SXx`, September 2021. (Accessed on 07/01/2024).

[8] Luis Cadarso and Ángel Marín. Integration of timetable planning and rolling stock in rapid transit networks. *Annals of operations research 199: 113-135*, 2012.

[9] Barrena E. Algaba E. Zarzo-A. Canca, D. Design and analysis of demand-adapted railway timetables. *Journal of Advanced Transportation, 48(2), 119–137*, 2014.

[10] Li S. D'Ariano A. Yang-L. Chen, Z. Real-time optimization for train regulation and stop-skipping adjustment strategy of urban rail transit lines. *Omega, 110, 102631*, 2022.

[11] Wikipedia contributors. Nederlandse spoorwegen. https://nl.wikipedia.org/wiki/Nederlandse_Spoorwegen, 2024. Accessed: 2024-06-24.

[12] John Hancock and Taghi M Khoshgoftaar. Impact of hyperparameter tuning in classifying highly imbalanced big data. In *2021 IEEE 22nd international conference on information reuse and integration for data science (IRI)*, pages 348–354. IEEE, 2021.

[13] John T Hancock and Taghi M Khoshgoftaar. Catboost for big data: an interdisciplinary review. *Journal of big data*, 7(1):94, 2020.

[14] Tan Y. Wang F. Bu-L. Jiang, Z. Turnback capacity assessment and delay management at a rail transit terminal with two-tail tracks. *Mathematical Problems in Engineering, 2015, 1–12*, 2015.

[15] 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.

[16] Huisman D. Abbink E. Fioole P.-Fischetti M. Maróti G. Schrijver A. Steenbeek A. Ybema R. Kroon, L. The new dutch timetable: the or revolution. *Interfaces, 39(1), 6–17*, 2009.

[17] Axel Kud. Why we need encoding cyclical features. https://medium.com/@axelazara6/why-we-need-encoding-cyclical-features-79ecc3531232, December 2023. (Accessed on 07/01/2024).

[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.

[19] R Shyam, Sai Sanjay Ayachit, Vinayak Patil, and Anubhav Singh. Competitive analysis of the top gradient boosting machine learning algorithms. In *2020 2nd international conference on advances in computing, communication control and networking (ICACCCN)*, pages 191–196. IEEE, 2020.

[20] Nico Spilt. Langs de rails. https://www.nicospilt.com/index_kopmaken.htm, 2021. (Accessed on 07/01/2024).

[21] Štepec, Dejan, Tomaž Martinčič, Klein, Fabrice, Vladušič, Daniel, and Joao Pita Costa. Machine learning based system for vessel turnaround time prediction. In *2020 21st IEEE international conference on mobile data management (MDM)*, pages 258–263. IEEE, 2020.

[22] Jin J. G. Lee D. Axhausen-K. W. Erath A. Sun, L. Demand-driven timetable design for metro services. *Transportation Research. Part C, Emerging Technologies, 46, 284–299*, 2014.

[23] Jiang Tao, Hua Man, and Li Yanling. Flight delay prediction based on lightgbm. In *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, pages 1248–1251. IEEE, 2021.

[24] A. Nõu J.-E. Nilsson U. Brännlund, P. O. Lindberg. Railway timetabling using lagrangian relaxation. *Transportation Science 32(4):358-369*, 1998.

[25] Marcella S. Dario P. Shaoquan-N. Yuhua, Y. Train timetabling with passenger data and heterogeneous rolling stocks circulation on urban rail transit line. *Soft Computing, 27(18), 12959–12977*, 2022.

Chapter 6

Appendix

The appendix contains supplementary visualisations referred to throughout the thesis in addition to the link to the GitHub repository where the scripts and data exploration notebook can be found.

6.1 Source Code

The scripts that were created for this study can be found in the following GitHub repository.

`https://github.com/timrentenaar/NS_TURNOVER_TIMES/tree/main`

6.2 Optimization Plots

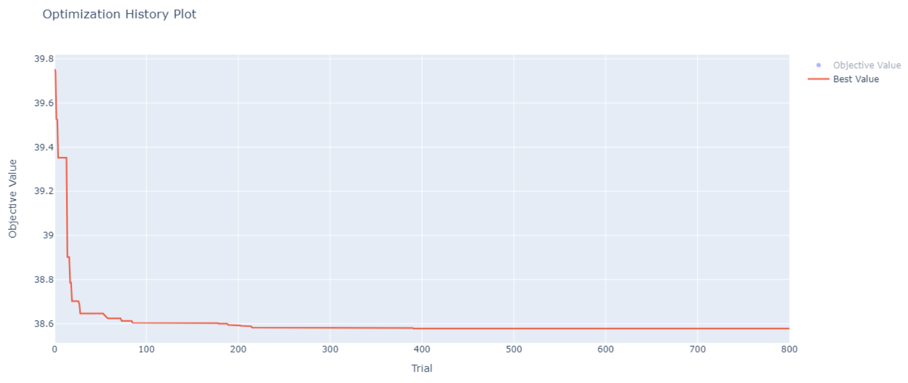

Figure 6.1: Optimization history of short turnover split

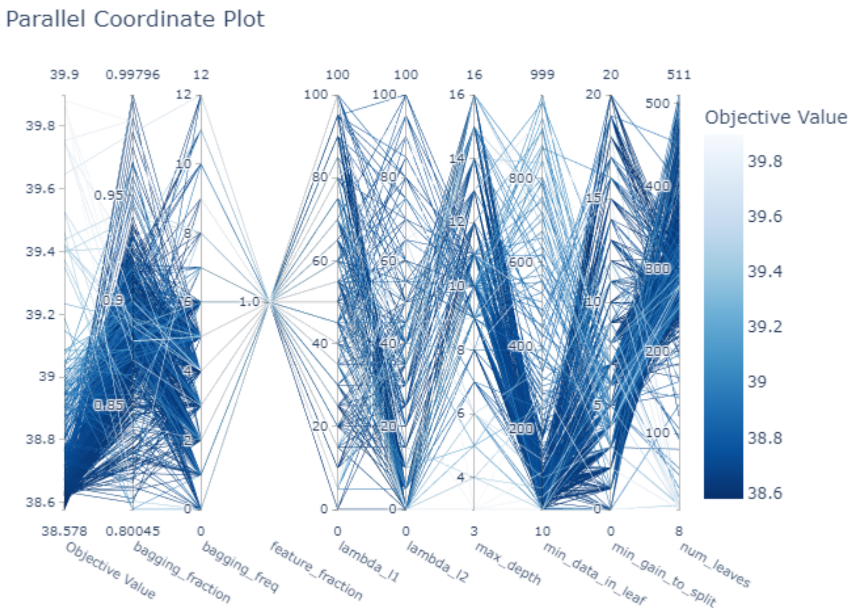

Figure 6.2: Parameter grid explored for short turnover split

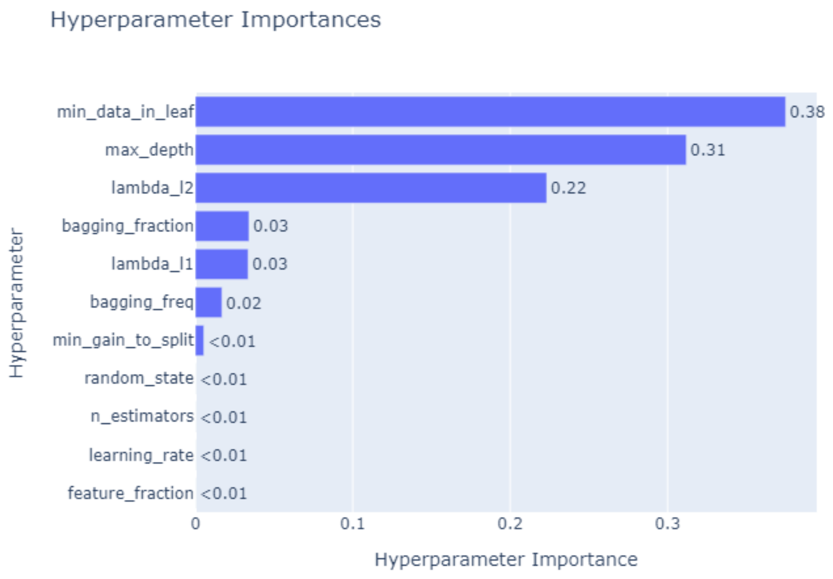

Figure 6.3: Parameter importances of short turnover split

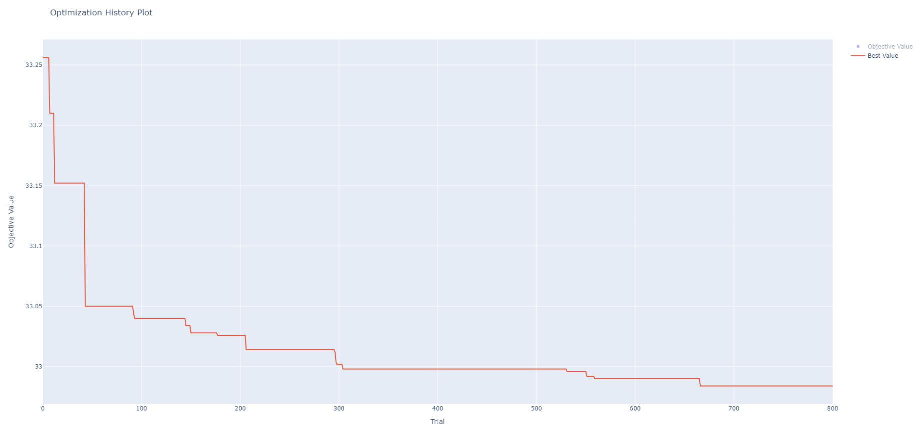

Figure 6.4: Optimization history of middle turnover split

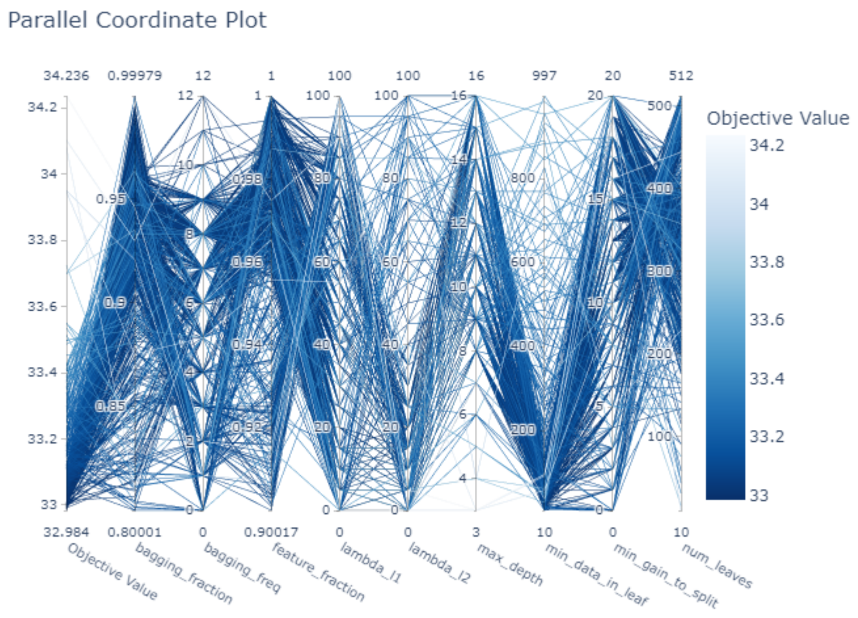

Figure 6.5: Parameter grid explored for middle turnover split

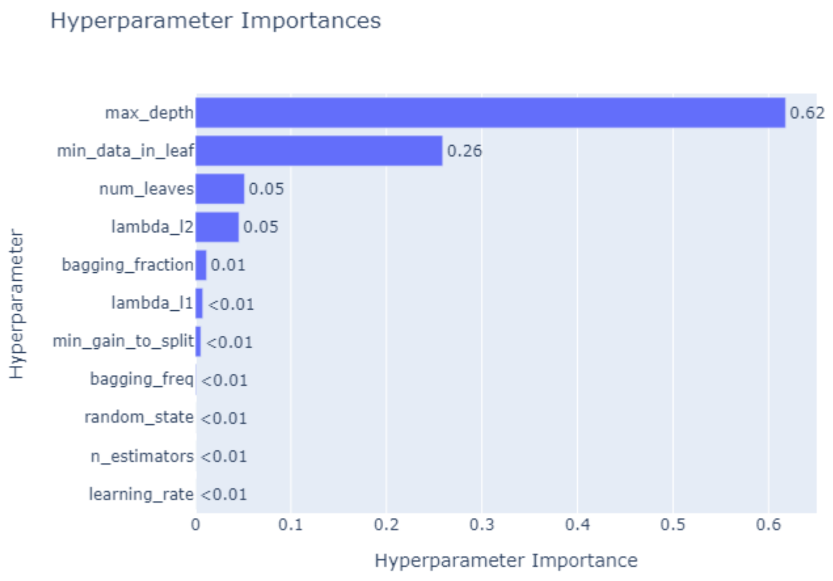

Figure 6.6: Parameter importances of middle turnover split

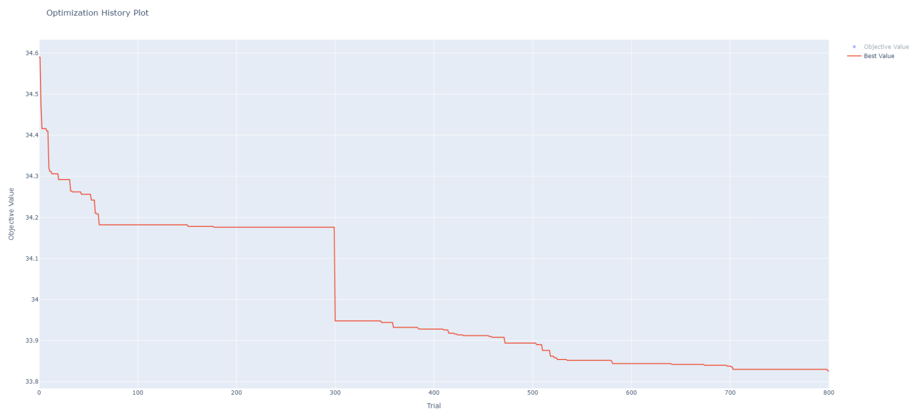

Figure 6.7: Optimization history of long turnover split

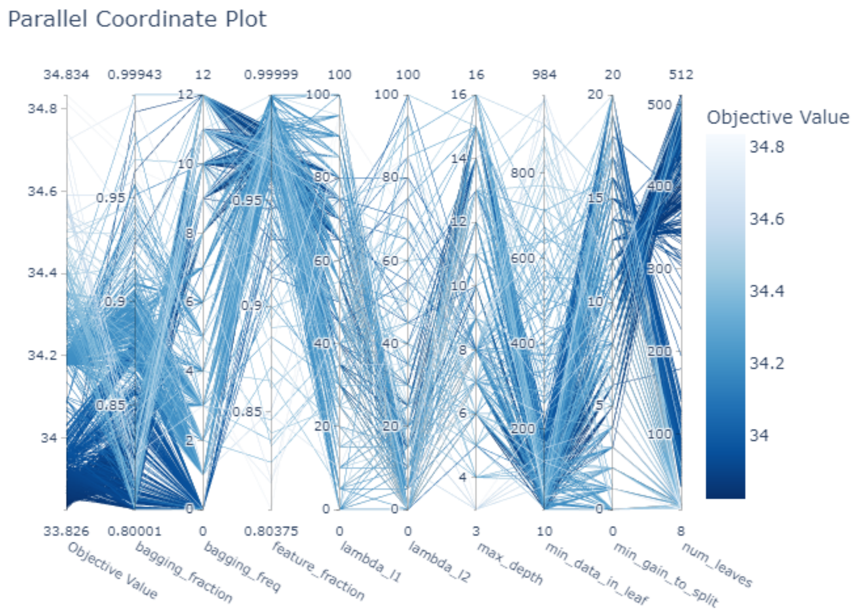

Figure 6.8: Parameter grid explored for short turnover split

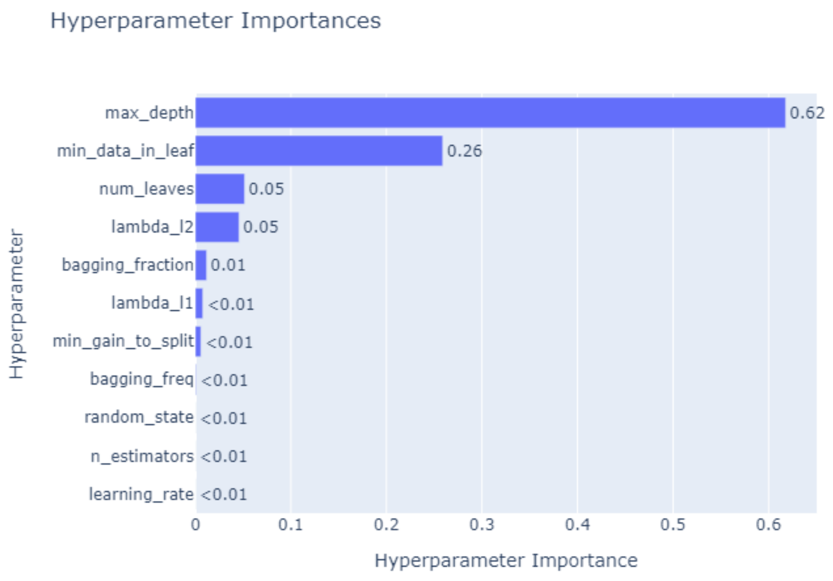

Figure 6.9: Parameter importances of long turnover split

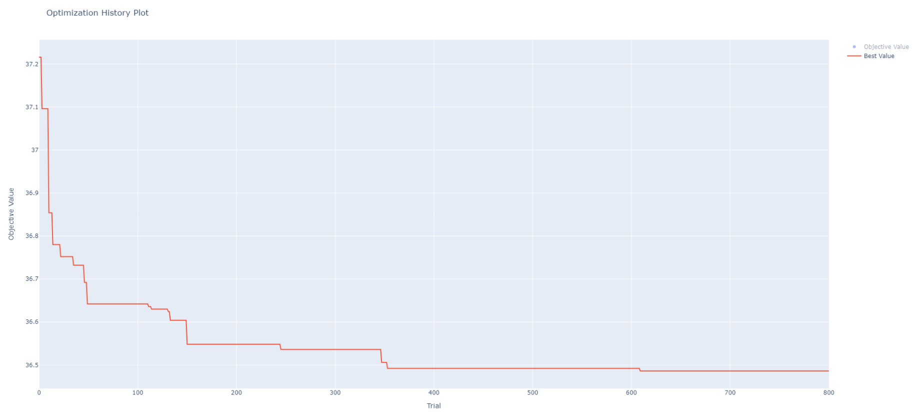

Figure 6.10: Optimization history of very long turnover split

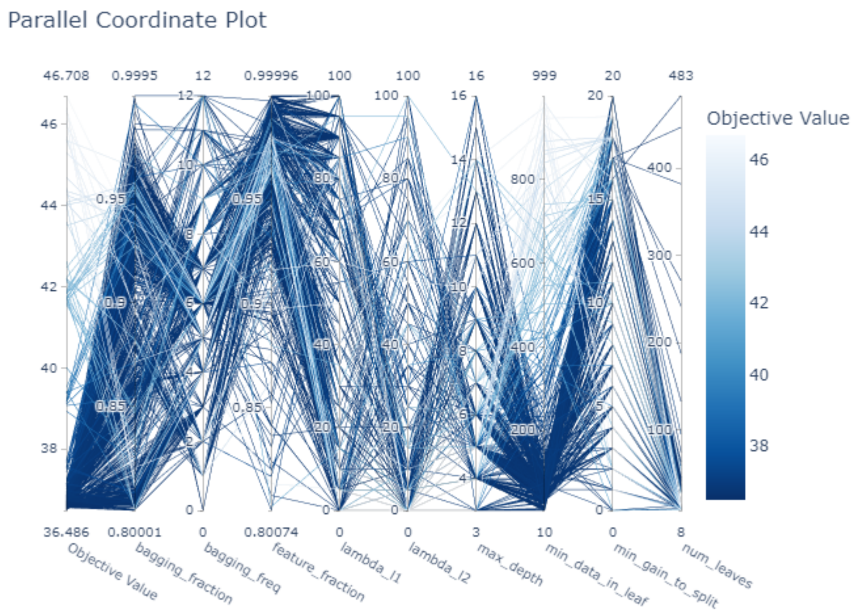

Figure 6.11: Parameter grid explored for very long turnover split

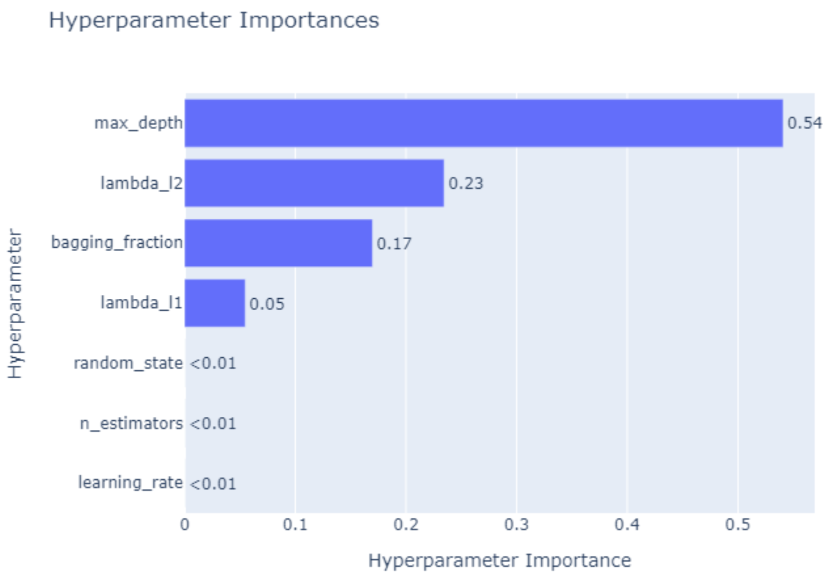

Figure 6.12: Parameter importances of very long turnover split

6.3 Feature Importances

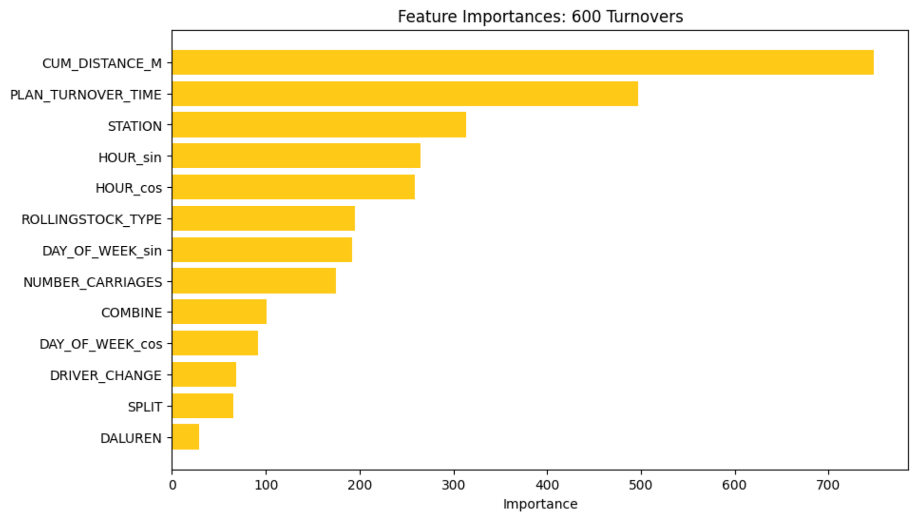

Figure 6.13: Feature importances of short turnover split

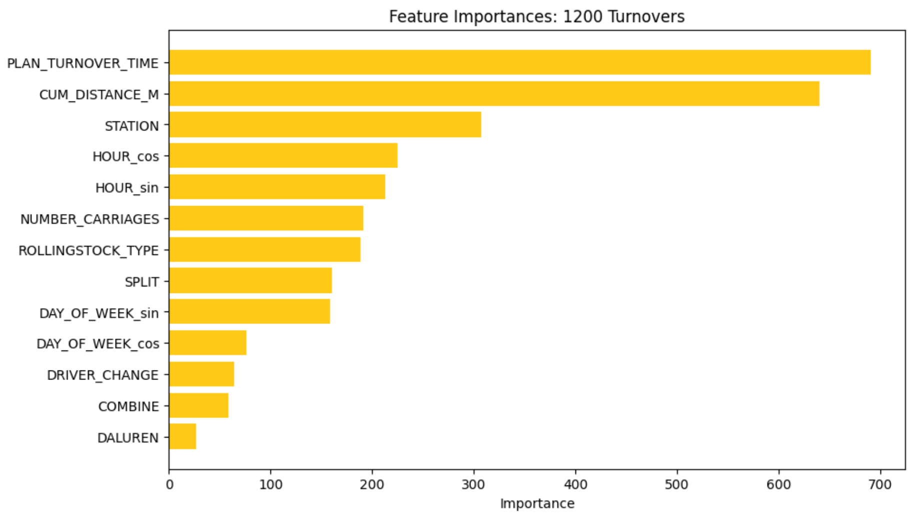

Figure 6.14: Feature importances of middle turnover split

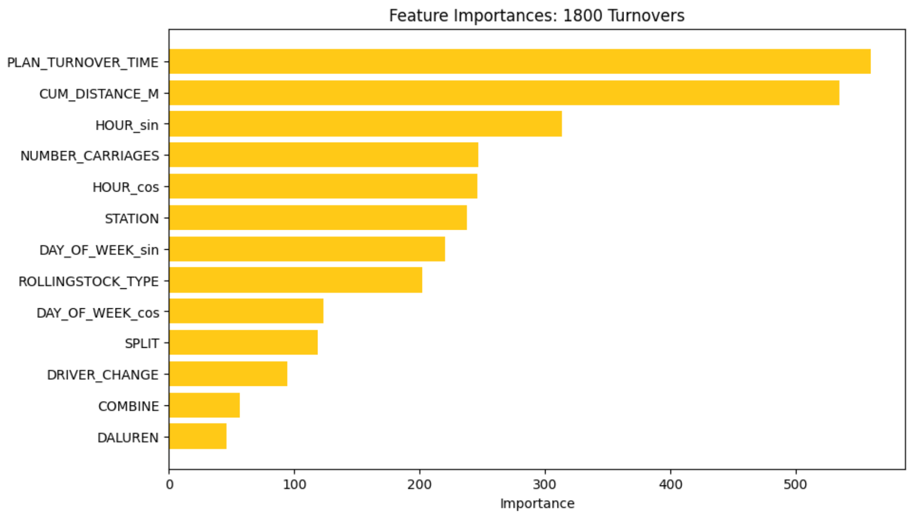

Figure 6.15: Feature importances of long turnover split

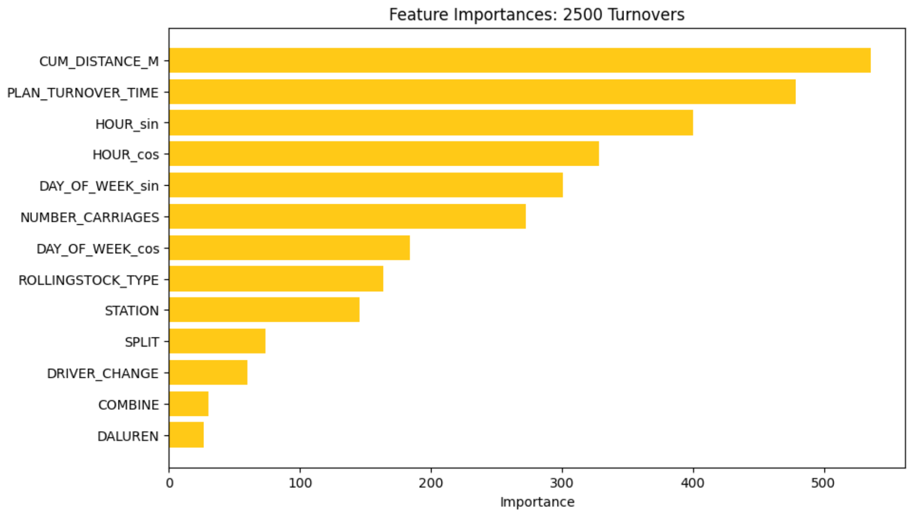

Figure 6.16: Feature importances of very long turnover split