Case study: predicting service times in package delivery operations development and execution of an XGBoost algorithm

Borst, J.

Award date:
2019

Eindhoven, December 2018

Case study: Predicting service times in package delivery operations

Development and execution of an XGBoost algorithm

BSc Joran Borst

Student identity number 0857374

in partial fulfilment of the requirements for the degree of

Master of Science

in Operations Management and Logistics

commissioned by:

Coolblue

Supervisors:

dr. A. Akçay, TU/e, OPAC IE&IS, TU/e

dr. M. Firat, TU/e, Information Systems IE&IS, TU/e

MSc. J. Verhoeven, Shipping & Delivery, Coolblue

TUE. School of Industrial Engineering.

Series Master Theses Operations Management and Logistics

Subject headings: Service Times, Consumer Goods Delivery, Machine Learning, Retail

Contents

Executive Summary

This case study analysis and predicts service times in delivery of customer goods in package delivery vehicle routing. A machine learning XGBoost algorithm is used to make service time predictions. The new service time prediction results are described, thereby indicating the accuracy of the XGBoost algorithm prediction performances in the context of consumer goods delivery service times. In this study the XGBoost algorithm replaces the current rule-based service time prediction model currently used by Coolblue. The performance of the XGBoost algorithm is described relatively to the current rule-based service time prediction model, as well as to a simple linear regression model. Besides the service time mean absolute error (MAE) performance metric used in this study, also customer satisfaction score estimations are provided in this case study. Customer satisfaction is the most important KPI used by Coolblue. The customer satisfaction is measure by the NPS (Nett Promotor Score) metric. By usage of XGBoost algorithm for creating service time predictions, the delivery service time MAE decreases by 1.09 and 1.14 minutes for the two-indicated configuration of best models input parameter settings. This prediction performance improvement is measured relatively to the current rule-based model used by Coolblue. Furthermore, resulting from this study it turns out the XGBoost algorithm creates 0.39 better service time predictions relatively to a linear regression model prediction. Although the application of the XGBoost algorithm results in more accurate service time predictions no significant improvement of the customer satisfaction NPS score is found.

Coolblue is the second largest online retailer in the Netherlands. In 2017 it reaches a yearly revenue of 1.2 billion Euros. Coolblue sells a wide range of electronical devices, but also sells gardening tools, fitness equipment and travel accessories. This project is performed at the largest department of Coolblue, named 'Shipping and Delivery'. This department is responsible for both process improvement and growth of the Coolblue privately owned delivery service. Coolblue wants to improve the accuracy of the service time predictions. Besides indirect impact of better service time predictions for potential improvement of NPS scores, Coolblue indicates six reasons why more accurate service time predictions are beneficial for the company. Among others, Coolblue thinks about indicating a shorter arrival time interval (than the current 1-hour time interval) to the customer in the future. Furthermore, more accurate service time predictions result in less variability in the deliver operations, less customers calling the customer help service and happier delivery employees.

In this study a wide range of variable is extracted from the company database. Moreover, new floor level information is obtained by floor level surveys filled in by the delivery employees after the products are successfully delivered to the customer. Also new features are engineered by combination and transformation of existing features. In this case study two cross-validation methods, four variable scaling techniques, 5 train / test splits data partition configuration and the quantile outlier reduction method are used. A scenario analysis is performed to detect successful method configuration and input dataset features. The input dataset applied to the prediction model turns out to be most important for achieving accurate service time predictions. Different model scenarios in this case study are different configuration of one of the following model attributes; input dataset, algorithm, pre-processing method, parameter values.

All model scenarios got assigned some parameter values and hyperparameter values. The difference between parameters and hyperparameters are its acceptance of adaptation of the (hyper)parameter values during the model training phase. While parameters are determined before the model is trained, the hyperparameters are adjusted in subsequent model iterations. The hyperparameters are tuned by using a Bayesian hyperparameter optimization method, named Tree-Structured Parzen Estimation (TPE). Shuffled and K-Fold Shuffled cross-validation is used to split the train datasets in train set and validation set folds. After a good selection of hyperparameter values is determined, the best performing hyperparameters are used as input parameters for the best model.

Whereas, only the train set is used to determine the optimal hyperparameters, the test set is used to describe the real model performance. The test set is a holdout set split apart from the initial dataset.

Besides MAE analysis, NPS score analysis, (input dataset) scenario prediction performance analysis, also the performance of four visualizations are tested. Three of those four visualizations are aimed at uncovering the feature importance as well as the feature value importance in the model. The remaining visualization assists in pointing out the difference in prediction performances of the current Coolblue prediction model as well as the new XGBoost prediction model. This performance impact plot indicates the XGBoost model is better predicting the delivery service times in than the current Coolblue prediction model. Beside prediction model comparison, the performance impact plot assists in recognizing the strength and weaknesses of the XGBoost service time prediction even on the feature value level. The plot is very useful and indicative both before and after the implementation the XGBoost machine learning prediction in company operations.

1. Introduction

This study creates and analyses service time predictions in package delivery vehicle routing operations. The Service time predictions are obtained by usage of machine learning algorithms. More concretely, the XGBoost Gradient Tree Boosting method is used to create the service time predictions. This study is a Master Thesis Graduation Project that is performed at the Operation Management and Logistics Graduate Program of Eindhoven University of Technology. The company at which the case study is performed is named Coolblue. The project is executed at the Shipping and Delivery Department. Coolblue determined the object of study. Operations Management and Logistics is a multidisciplinary study program that comprises disciplines such as product development, quality management, logistics, information systems and human resource management (Eindhoven University of Technology, 2018). This study can be considered as a trade-off from the product development and information systems research areas.

1.1 Coolblue

Coolblue is the second largest online retailer in the Netherlands (ecommercenews, 2018).
The company's yearly revenue increased substantially last couple of years. Since 2015 the company yearly revenue more than doubled from 540,000 million euro yearly revenue to 1.2 billion euro yearly revenue in 2017 (twinklemagazine, 2018). Coolblue sells a wide range of electronic devices plus some other products such as gardening tools, fitness equipment and travel accessories. In 2015, Coolblue started delivering white good products themselves. Starting a delivery service entailed the establishment of a new department managing all operations involved in delivering the Coolblue product. The department name 'CoolblueDelivers' was chosen. CoolblueDelivers started with a 2-Man delivery service, which delivers medium to large size products (mostly white-goods). The Coolblue 2-Man delivery service evolved rapidly into the most important delivery service for delivering the medium to large size-category products that are sold by Coolblue. In addition to the Coolblue 2-man delivery service, also 1-man delivery service and bike delivery service has been added recently. One of the most important reasons the Coolblue company and the CoolblueDelivers department achieved such fast growth is the excessive focus on customer satisfaction. Coolblue measures customer satisfaction performance in terms of Nett Promotor Score (NPS). NPS is defined as the ratio of customer experience and predicts business growth (Reichheld, 2004). NPS is measured on a ten-point rating scale. Customers indicating a NPS score of 9 or 10 are called promotors. Customers indicating a NPS score of 6 or lower are called detractors. After composing the ratio of promotors and the ratio of detractors, the NPS score is obtained by subtracting the detractor ratio from the promotors ratio. In 2017, the NPS score of Coolblue was ranging around the 73, whereas the NPS score of similar delivery companies as B-Post and PostNL was ranging around 70 and 66 respectively. The relatively high customer satisfaction score is achieved by unique propositions as 'Ordered today before 23.55 - Delivered tomorrow'. Furthermore, CoolblueDelivers just implemented the option for ordering a delivery time slot (morning, afternoon, evening) at the same time of ordering the products. Coolblue wants to enhance the customer satisfaction NPS score even more by using better service time predictions, since a positive relation has been found between arrival time performance and NPS scores.

1.2 Study Motivation

This project was initiated by the Shipping and Delivery department of Coolblue. Coolblue wants to improve the service time predictions because service time predictions are not accurate according to Coolblue. Coolblue delivers a wide range of products to a wide range of customers. All customers have unique housing style and delivery address locations. Nevertheless, the planned service times are only set to distinct values for product groups. For example, all washing machines got assigned the same service times. Coolblue thinks at least more product and client information should be taken into consideration. There will inevitably be a wide range of factors significantly influencing the service time. Therefore, the aim of this study is to take as much as possible useful information into consideration to create the service time prediction. This is the reason a machine learning approach is taken in this study. Machine learning algorithms can learn (complex) relations between predictive (independent) variables and a target (dependent) output variable. A gradient boosting decision tree-based algorithm, named XGBoost, is used in this study to make service time predictions.

The overarching reason why the service time predictions need to be improved retains in the observation of a positive relation between arrival time performance and NPS. Arrival time performance is dependent on service time predictions, since prediction of the service times more accurately will result in better arrival time performance. Coolblue observed that from January 2017 to October 2017 the NPS scores for on-time arrivals turned out to be 75 on average, NPS scores for too-early arrivals had a mean of 74 and NPS scores for too-late arrivals were 71 on average.

Besides the positive relation between arrival time performance and customer satisfaction, there are more reasons that motivating this study:

- Better service time predictions usually lead to less too-late arrivals. Too-late arrivals are harmful in terms of customer arrival time accuracy levels. Furthermore too-late arrivals also have a negative impact on positive company opinion estimations made by the customers.
- Better service time predictions, in general, lead to less waiting time (slack) in the vehicle routing, because inaccurate service time predictions result in lower probability of on-time arrival. This means that better service time predictions reduce the chance of customers not being at home because they expected the Coolblue delivery employee to arrive earlier or later. The costs of waiting time in vehicle routing consists of the delivery employee salary costs.
- A higher amount of waiting time results in more variation in the package delivery operation mean service time. According to the Manufacturing Systems Queueing Formula ($\varphi_B = \frac{c_a^2+c_o^2}{2}\frac{u}{1-u}t_o$), variation in process time, results in higher flow time (φ). Process time variation is included in the flow time formula as coefficient of variation (c_a). Coefficient of variation is obtained by dividing the process time standard deviation by the mean process time ($c_a = \sigma_a/\mu_a$). obtained by inaccurate service time predictions prediction results in higher flow time. Flow time denotes to total time between departure at the depot until arrival at the depot after delivering all products.
- Better service time predictions usually lead to less need to adjust the tour planning, which subsequently reduces planning costs.
- Better service time predictions usually lead to less too-late arrivals. Too-late arrivals are one of the major factors of customers calling the customer care service. This is expensive in terms of customer service salary costs.

- Coolblue potentially wants to indicate shorter time windows to the client (i.e. half an hour) instead of the current 1-hour time window reported to the client the same morning of actual delivery.

1.3 Customer Satisfaction Metric used at Coolblue - NPS

NPS is the most critical performance measurement metric used within the Coolblue company. NPS is an abbreviation for 'Net Promoter Score'. In section 1.1 a definition is provided for NPS. The general aim of NPS is catching the customer loyalty and the likelihood of Coolblue product promotion by the customer. Coolblue measures NPS score is by making use of NPS surveys. Customers with a registered online account are asked to indicate the likelihood of advising Coolblue as a company to family or friends, on a 10-point rating scale. This single question NPS survey is sent to member customers by e-mail after the products have been delivered. In Figure 1.1 the NPS score calculation method is provided. The NPS score is obtained by subtracting the detractor percentage from the promoter percentage (Reichheld, 2004). Customers indicating a 9 or 10 NPS score are called promoters. Customer indicating a 7 or 8 score are called neutral customers. Customers indicating a score ranging from 0 to 6 are called detractors. The distinct names assigned to NPS score values are self-explanatory. Promotors are expected to promote Coolblue products to family and friends. Detractors are expected to warn family and friend for buying products at Coolblue. Neutral customers are expected not to promoting Coolblue to family and friends, either positively or negatively.

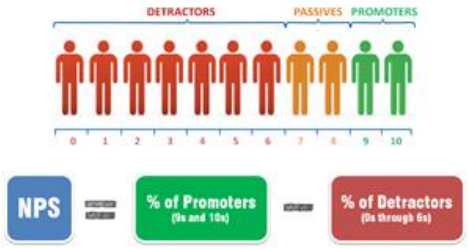

Figure 1.1: Net Promoter Score (NPS) Calculation Method.

1.4 Current Coolblue Service Time Planning

The current Coolblue service time prediction model can be formulated as:

$$OST = \sum PST_i + \sum UT_i + dQT \quad (1.1)$$

OST	*Service time*	*The total time required to service the customer, by handling all products to the customer when arrived at the customer's house. The service time is calculated as: (Departure Time − Arrival Time)*
PST	*Product(s) Service Time*	*The total time required to deliver (and for some products install) the product(s) at the customer indicated location inside the house. Multiple products can be included in the order, therefore we need to sum the UT of all individual products i.*

UT_i	*Unloading Time*	$The\ total\ time\ required\ to\ unload\ the\ ordered\ products\ from\ the\ van\ when\ arrived\ at\ the\ customer\ address\ location.\ Multiple\ products\ can\ be\ included\ in\ the\ order,\ therefore\ we\ need\ to\ sum\ the\ UT\ of\ all\ individual\ products\ i.$
QT	*Questionnaire Time*	$Time\ required\ for\ filling\ in\ a\ short\ questionnaire\ together\ with\ the\ customer,\ after\ the\ product\ has\ been\ delivered\ at\ the\ customer's\ house.\ This\ questionnaire\ should\ only\ be\ filled\ in\ ones\ per\ order.$
d	*Boolean*	$\forall d \in \{0,1\}$ *, where* $0 \quad if \sum(nDPP_i + nFSP_i) = 0$ $1 \quad if \sum(nDPP_i + nFSP_i) > 0$
$nDPP_i$	*Number of DrempelPlus Products*	$Drempelplus\ refers\ to\ the\ service\ that\ is\ attached\ to\ products\ belonging\ to\ this\ category.\ Drempelplus\ Products\ are\ delivered\ at\ the\ customer\ indicated\ location\ inside\ the\ house.$
$nFSP_i$	*Number of Full-Service Products*	$Full\ Service\ refers\ to\ the\ service\ that\ is\ attached\ to\ products\ belonging\ to\ this\ category.\ Full\ Service\ Products\ are\ both\ delivered\ and\ installed\ at\ the\ the\ customer\ indicated\ location\ inside\ the\ house.$

1.5 Project Statement

Setting a project statement is the most critical decision in defining a good study. This is because the project statement affects a lot of individual study components. Some study components will be restricted by the definition of the project goal. The project statement is defined as follows;

Case Study: Select, develop and execute a structured approach to find the most viable service time prediction solution in package delivery vehicle routing by using the XGBoost algorithm.

Concretely, prediction models are designed to predict future delivery service times as accurately as possible. The predictive power of various machine learning models is tested and compared with each other. The most viable solution is concluded in this report and recommended to the management of shipping and delivery department of Coolblue.

1.6 Type of Study

In this study, predictive analysis is performed. Predictive analytics is a broad umbrella term which includes many research areas. The best fitting research area for classifying this study is empirical models for prediction. Empirical models for prediction is part of the 'parent' study area Empirical models (Esearch & Koppius, 2011, p. 554-555). Empirical Models are models that make predictions or

explanations based on empirical quantitative or qualitative data. In this study, empirical quantitative data is used as the machine learning model input. Specifically, empirical quantitative data is data collected from questionnaires or other measurement instruments (Patten & Newhart, 2017, p. 94). The Empirical models' research area is divided into two sub streams:

- Empirical models for prediction
- Empirical models for explanation

Empirical models for prediction seek to maximize predictions accuracy by both *model bias* and *sample variance*. (Esearch & Koppius, 2011, p. 555) In empirical models for prediction, predictive power is prioritised over explanation power.
On the contrary, the specific aim of empirical models for explanation is testing causal hypotheses between empirical phenomena (Esearch & Koppius, 2011, p. 554). To test causal relations, the explanation power of model input variables is tested. Exploratory power refers to the strength of the relationship between two empirical constructs.

1.7 Structure of the Thesis

In this chapter, the introductory information is provided. Among others, the company at which the study is performed is introduced, the project statement is given.

In chapter 2, a short literature review is provided describing the context of service time predictions in vehicle routing. Among others in this chapter it is stated that not much previous research has been done in this specific research area.

In chapter 3, a structured approach is provided to make machine learning predictions. It is described that the empirical model development process scheme, proposed by Schmueli and Koppius (Esearch & Koppius, 2011), is used in this study. The empirical model development process scheme includes steps as; goal definition, data gathering, data aggregation, data pre-processing, exploratory data analysis, model selection, model deployment and output evaluation and validation. The methods used in each of the steps is analysed in detail

Chapter 4 is named conceptual model. In this chapter, a description is given how the methods proposed in chapter 3 are concretely being applied in this study. A description is given how for example the data is aggregated and combined, how exploratory data analysis (EDA) is done, which pre-processing operations are indeed applied to the data, what prediction algorithm is used and how data is evaluated, validated.

Chapter 5 is the results section. Both numerical and graphical results are provided. Central in this result section is the performance on three KPIs; Mean Absolute Error (MAE), Mean Error (ME) and NPS. The best model is indicated as well in this section.

In chapter 6, the conclusion is drawn. Again, the performance of the best machine learning model is analysed in comparison to the current Coolblue service time planning model and other machine learning prediction models. Furthermore, an evaluation is done whether this best model is a viable solution for Coolblue to implement.

Chapter 7 is the discussion section. The shortcomings in the input dataset, as well as any other computation steps applied, are described in relation to the output accuracy.

Chapter 8 gives recommendations for further research. Recommendations can best be useful preparation steps to analyse before the machine learning service time predictions model can be implemented.

Chapters 9, 10 and 11 give a summary of the study, references and the appendix respectively.

2. Literature Review

In this chapter first, the previous research on service time calculation in vehicle routing is reviewed. Second, the importance of customer satisfaction and accurate service time predictions is analysed and confirmed.

2.1 Service Time Calculations in Vehicle Routing

The traditional Vehicle Routing Problem (VRP) encompasses many extensions. Examples of these extensions are for example Capacitated VRP (CVRP), Time-Dependent VRP (TDVRP), Pickup and Delivery Vehicle Routing Problem (PDPVRP), Time Window VRP (TWVRP) and Split-Delivery VRP (SDVRP) (Lin, Choy, Ho, Chung, & Lam, 2014). The Coolblue service time problem also belongs to a specific category type of the vehicle routing problem, namely the Stochastic Delivery and Service Time Vehicle Routing Problem (VRPSST). The VRPSST has received shallow attention in the literature. Gómez et al. (2016) are one of the few researchers that investigated the VRPSST. Gómez et al. (2016) indicate that different algorithms can be used to optimise routes. Examples of suitable algorithms for the VRPSST problem are; tabu search and neighbourhood search. Vehicle routing problems in general have two components. The first component is the driving time. This is driving time it takes from the depot to the customer, or the driving time between subsequent customers in the vehicle routing tour. The second component is the service time. The service time is the difference between departure time at the customer and the arrival time at the customer. The first, component driving time, in literature is mostly approximated by usage of a single or multiple distribution approach (Gómez et al. 2016). Service time in general is significantly less than the driving time. Therefore, in most research, service time is approximated by usage of a single distribution or even a deterministic value. There are several reasons that doubt the accuracy of deterministic service time values by using it in predictive context. Remind that some service times will inevitably take longer than others because it takes some time to arrive at the front door of the house. Furthermore, some addresses are located at higher floor levels, for example addresses that are in a flat.

2.2 Importance of customer satisfaction and accurate service time predictions

In the last couple of decades, the primary objective function in most routing models was minimising the service costs (Kovacs, Golden, Harti, & Parragh, 2014). The simplest and widely used method for minimising service costs is achieved by minimising the delivery service time. But on the contrary, excellent service will result in satisfied customers. Satisfied customers form a bond with the company and are very likely to repurchase again in the future (Kovacs et al., 2014). The effects of satisfied customers can be found in Figure 2.1 below. Figure 2.1 gives the service-profit chain configuration. To achieve a high perceived value by the customer and consequently high-profit margins, customer satisfaction must be enhanced as much as possible. Historical customer satisfaction data from Coolblue indicates a relation between on-time arrival and customer satisfaction scores. In general, the rule of thumb holds that the better the delivery employees arrive on time, the higher the customer satisfaction is. Historical data about too-early and too-late arrivals show lower customer satisfaction scores. Indirectly, the accuracy in arrival time is pleasant for the customer, thereby enhancing the perceived value and eventually profitability.

The focus in vehicle routing problems should not be on minimisation of the service time but on reasonable service time. With reasonable service time, it is meant that the delivery employees got assigned an accurate service time to deliver products but are not pushed towards unreasonable low service times. Unreasonable low service times will negatively impact the time to help the client, thus

reducing the received value and profitability. Creating reasonable service times thus is in line with the customer-centric approach used at Coolblue.

Additionally, accuracy in service times is also in line with the queueing theory. Higher variance in service time amplifies the total flow time, represented as route duration in case of the VRP. The relation between flow time and variation is presented as; $\varphi_B = \frac{C_a^2+C_o^2}{2}\frac{u}{1-u}t_o$. Flow time is indicated by (φ). The flow time increases among other by higher values of the numerator c_o. c_o stand for the coefficient of variation observed in the process, or in this case the vehicle routing tour. The coefficient of variation is composed by dividing the standard deviation by the mean service time. The standard deviation of route durations increases by relatively bad service time predictions, resulting in longer flow times.

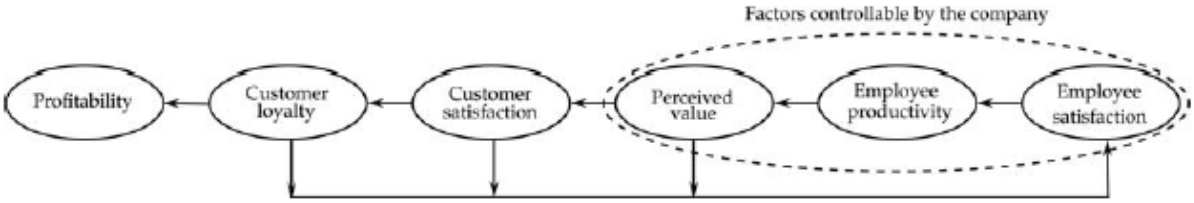

Figure 2.1. Service Profit Chain Relation (Kovacs et al., 2014)

3. Methodology

The method used to perform this study is abstracted from the article 'Predictive Analytics in Information Systems Research', written by Schmueli and Koppius (2011). Shmueli and Koppius (2011) describe a way of integrating predictive analytics into information systems research (Shmueli & Koppius, 2011, p. 553). A diagram including a sequence of steps for building a predictive empirical model is presented. This diagram is adjusted slightly and used as the guideline to perform the predictive analysis in this study. The new diagram is presented in Figure 3.1 below. The main reason for using this specific diagram is the clarity of actions described in individual part of the diagram. By following the steps sequentially, a complete analysis of viable machine learning prediction models can be done. The model created is flexible in its design, therefore in the remainder of this report the name 'machine learning tool' will be used for referring to the model. The machine learning tool is flexible for inserting inputs and at the end the tool outputs both statistical performances as well as visual representations of the service time prediction performance. More input and output options of the machine learning tool are discussed in this method section and in Chapter 5. In the remainder of this method section, all individual steps of the machine learning process diagram in Figure 3.1 are discussed in separate sections.

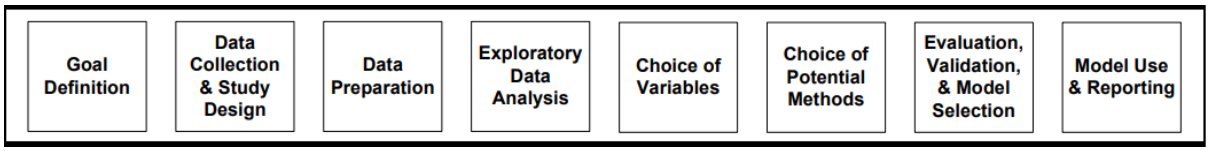

Figure 3.1. Process diagram scheme for building the empirical service time prediction model

3.1 Goal Definition

According to the process diagram scheme in Figure 3.1, the first step in creating a service time prediction model is the definition of a clear project goal. The goal of this study is; finding and selecting the best service time prediction model. Coolblue remain having the right to reject a model based on for example complexity reasons. In this study, a vast amount of different model scenarios is created and tested on the indicated performance metrics. Scenarios in the service time prediction tool are considered unique if the two models are inconsistent about any of the following model machine learning model attributes; input dataset, algorithm used, pre-processing method, input parameter values or hyperparameter input values.

3.2 Baseline Model Selection

The baseline model is the model that serves as the initial performance baseline to compare against the new prediction model. The current Coolblue service time prediction model is used as the baseline model in this project. To make service time prediction, Coolblue currently makes use of a rule-based model. The current Coolblue service time prediction is composed from 3 dimensions, which are product group, unloading time and service time. The current Coolblue service time prediction model is given in Formula 1.1 and explained in detail in section 1.4. The current Coolblue service time prediction model is selected as baseline model because this study aims to discover the service time prediction improvement achieved by usage of machine learning models relative to rule-based planning solutions.

3.3 Data collection and study design

The third step of the service time prediction model diagram in Figure 3.1 is: 'Data collection and study design.' According to Shmueli and Koppius (2011), three things need to be considered in this step:

1. Data Collection Instruments
2. Variable Inclusion
3. Input Data Sample Size

The three pre-mentioned steps are discussed in more detail below.

3.3.1 Data Collection Instruments

The data collection instruments used in this project are two relational database management systems. Relational database management systems enable loading in data from structured database environments. The relational databases used in this study are SQL Servers and DireXtion. SQL Servers is the internal database in which almost all information of Coolblue is stored. SQL Servers give us historical data for most of the product and order related variables in the service time machine learning tool. DireXtion is a Route Planning Tool used by the Coolblue. This application is provided by a third-party software supplier. The DireXtion database provides is with all historical planned and realised route planning times.

3.3.2 Variables Included

To select the complete set of variables, a list of variables potentially impacting the service time is composed. According to Shmueli and Koppius (2011, p. 564) variables for prediction should be chosen based on domain knowledge and empirical evidence of association with the response. Therefore, the list of input variables is composed by interviewing delivery employees, delivery team leaders and operation employees.

There are two reasons why none of the variables are deleted upfront:

1. Empirical evidence of predictive association with the response is hard to detect upfront. A predictive association can only be guaranteed by just using the variable in a prediction model. Using a correlation to determine the predictive association is not accurate since correlation relation does not imply predictive association.
2. Additionally, Shmueli and Koppius (2011, p. 1165) state that variables that are completely useless by itself can be useful for predictions by taking them together.

3.3.3 Input Data Sample Size

Different sample sizes are used in this project. The sample size changes by unique definition of the input data scope belonging to the various machine learning models tested in this study. The historical dataset used to make predictions has 310,595 rows. The original 26 columns are provided in Table B.1 in the appendix. The 310,595 orders were delivered to the customers during the period starting at 14-12-2017 and ending at 08-10-2018.

3.4 General Data Preparation

The next step belonging to the process diagram of Figure 3.1 is general data preparation. General data preparation is decomposed into five categories;

1. Variable Conversion
2. Variable Creation
3. Missing Value Completion
4. Variable Value Correction
5. Data Aggregation

In this section, each of the five different steps is separately discussed.

3.4.1 Variable Conversion

Some value formats of variables in the input dataset needed to be converted. The overarching reason for converting those variable values is enabling the prediction model to process the variables. Machine learning methods cannot process nominal and ordinal variables. Therefore, this datatype variables needed to be transformed into numerical variables. In Table B.2 in the appendix, an overview of all three datatype conversion transformations performed to variables are given. In general, three types of conversion operations are applied; coercing, mapping and factorising.

Some variables are coerced from their current datatype into numeric datatype. Coercing is performed if the datatype format is incorrect, meaning that a wrong datatype initially is attached to the variable. This happens if the variable already is represented in numerical format, although the dataset does not recognise this.

Other variables are mapped from their current datatype into numeric datatype. The mapping conversion is applied if the current categorical variable values need to be converted into *specific* numeric values. The mapping operation thus is a manual process. Variables selected for this operation are Boolean variables or categorical variables with a small number of values. Examples of variables that are mapped to specific numerical values are the ‘b2bcustomer' variable and the ‘normal-delivery/pickup/swap' variable. The later variable is often named ‘service type’ in this study for convenience.

Last, some variables are factorised from their current datatype into numeric datatype. The Factorising operation transforms categorical data into numerical data. Factorizing is applied for transforming categorical variables with a vast number of values into numerical values. Mapping the vast number of variables is considered a massive operation, therefore factorising is the best way to create numerical data. Factorizing transforms categorical data into numeric variables based on alphabetical order.

3.4.2 Variable Creation

A small number of variables need to be created since those variables are currently not yet available in the database. In total four variables were created, these are;

1. Normal-Delivery/Pickup/Swap
2. Floor Level Front Door House
3. Floor Level Delivery Product Inside House
4. Stoptime Minutes

5. Visit Counter
6. Visit Counter Mean Service time
7. Real Quantity Including Combi Boxes
8. SubProductType_DeliveryService_ServiceType_Sizecategory
9. Mean Service Time
10. Mean Prediction Error
11. Mean Service Time (Clustered)
12. Mean Prediction Error (Clustered)

The 'Normal-delivery/ Pickup/Swap' variable simple was not present in the company database but was inherited in the 'order id' variable values extracted from the DireXtion database. Therefore, the 'Normal-delivery / Pickup / Swap' variable is created by splitting off this information from the 'order id' variable. The next two additionally crated variables were categorised as highly important by domain experts. The expert opinion was the ultimate trigger for adding two extra question to the 'post-delivery-survey' to obtain extra floor level information. The 'post-delivery-survey' is filled in by the delivery employees after every delivered order. However, since the survey question related to variable 2 and 3 are added to the 'post-delivery-survey' at 08-05-2018, orders before this date do not include variable 2 and 3 information. The last variable that is created is 'Stoptime Minutes'. 'Stoptime Minutes' refers is the created from subtracting the arrival time from the departure time at each delivery stop in the data. This feature needed to be created because the 'Realised Service Time' feature, which was extracted from the DireXtion database, turned out to be inaccurate. The 'Realised Service Time' variable gave values that were rounded to the nearest integer. Furthermore, a visit counter variable is feature engineered. This variable counts the number of visit belonging to unique customers. It could be that the delivery service time is impacted by the number of times a product has been delivered to the same customer. To capture this information the 'Visit Counter' variable and the 'Visit Counter Mean Service Time Variable' is created. Next, the order quantity is adjusted slightly in by creating a new 'Real Quantity Including Combi Boxes' variable. Products belonging to the size categories S/M/L1 are combined in Combi boxes. This affected the quantity size of the order delivered to the customer. Next a new product aggregation level is created. In this new product variable, the Sub Product Type information is combined with the Delivery Service information, the Service Type information and the Size category information. More new products variables are created by combining existing variables, but they are more thoroughly explained and analysed in section 5.1. Last, 'Mean Service Time' and 'Mean Prediction Error' variables are created. For 13 variables, among other the products variables, the mean service times and mean prediction error are calculated. These variables are tested in 5.1. Also, Clusters of the mean service time and mean prediction error outcomes are created for each variable separately as well as summed up. More explanation is provided in section 5.1.

3.4.3 Completing Missing Variables Values

Next, the dataset is checked on missing values. Some machine learning algorithms cannot handle missing data. For using null-value sensitive algorithms, it is necessary to complete missing values. It is worth considering the potential reasons that are causing the missing variable values to be missing. Missing variable values are present in the data for several reasons such as (Witten & Frank, 2011, p. 58);

- Malfunctioning measurement equipment
- Changes in experimental design during data collection
- Collation of several similar but not identical datasets

The historical input dataset includes some missing data as well. As mentioned in section 3.4.2, two floor level related variables include missing values. This is because the floor level variable information, that is obtained by a 'post-delivery-survey' was not filled in before 08-05-2018. No obvious indication was found causing some other variables to include missing values. According to Shmueli and Koppius, there are 9 ways of handling missing data (Shmueli & Koppius, 2011, p. 563);

1. Remaining null values in the original format
2. Removing observations
3. Removing variables
4. Using proxy variables
5. Indicate missingness by creating dummy variables
6. Completing null values to a numerical value
7. Completing null values with the mode, median or mean
8. Completing null values with 'neighbour sample variable values
9. Using advanced regression techniques to handle missing data

Three of the in total nine variable completion methods are tested in this study, which are completion method 1, and 6. Since only for three variables in historical input dataset missing values are present, it is decided to apply the 1st variable completion method (remaining null values in the original format) to most of the model scenarios. The reason for retaining the null values is that the XGBoost Tree based boosting algorithm can handle Nan values. Retainment of the variables is beneficial over removing the variables, because some variable information is potentially missing for a reason. By assigning a concrete number to this missing values, the 'missingness information' could be taken into consideration by the machine learning model. (Witten & Frank, 2011, p. 58). Furthermore, insertion of missing values by a numerical value and mean value is checked. No reason was found for inserting mode, median, mean or a neighbour sample variable value. The integer -9, -2, 0 and 9 are selected to replace the Nan values. 0 is the standard value to replace missing values with. -2 is chosen because many variables already include 0 as variable value and the 'floor level installation product inside the house' variable also has a -1 value. The -1 variable value here refers to the installation of a product in the basement of the house. The -9 and 9 values far from the scope of most other input variables, which satisfies the research on those completion variable values.

3.4.4 Correcting Variables Values

Next, some variables values needed to be corrected. Incorrectly logged variable values are tackled specifically in this stage. Incorrect logged variable values can arise by intentional or unintentional human incorrect logging or by a logging failure somewhere in the automated data logging pipeline. Incorrect logging, in general, should be considered as even more dangerous than missing values. This is because incorrect logging often is less visible on first sight and thus a more thorough investigation is needed to find an incorrect variable. Incorrectly logged variables behave as outliers. Next, incorrect variable values are discovered manually. The variable 'planned driving time' turns out to provide incorrect variable values. The 'planned driving time' gives different values from what is logically expected. The expectation of the 'planned driving time' is (arrival at client i) - (departure at client i-1). This turns out not to be in line with the observed data. To tackle this problem in this project specifically, the driving time is left out of the NPS simulation. An indirect method is used now that accurately measures predicted arrival time, which need to take the incorrectly logged 'planned driving time' variable into consideration.

3.4.5 Data Aggregation

Last, the complete input dataset is exposed to an aggregation transformation. All rows in the initial input dataset stand for specific products that are ordered by customers. The product aggregation level is not desired as input for our machine learning models since the delivery service time is linked to orders instead of products. One order consists of 1 to many products. Therefore, an aggregation of input dataset rows with same ‘order id' is applied to the complete input dataset. In section 4.1 a more extensive description and visual explanation are provided about this ‘order id' based aggregation method.

3.5 Exploratory Data Analysis

Exploratory data analysis (EDA) is an essential step that need to be applied in every machine learning project. According to Schmueli and Koppius (2011), EDA should be divided into two distinct parts;

1. Summarizing data numerically
2. Summarizing data graphically

In this study, both the input dataset variables and the objective function (NPS) needs to be explored both numerically and graphically.

3.5.1 Summarizing Data Numerically

Numeric summarisation of the input dataset variables is vital for discovering data structures, variable distributions and variable relations. Especially descriptive statistics, datatypes analysis and the unique variables count are used to summarise data numerically. In order to facilitate extracting knowledge from initial numerical statistics data, the service time prediction script automatically bundles all initial statistics information into a single excel file. The information is visualised for each variable individually. In Table B.2 in the appendix, an overview of these statistics excel is provided. The type of information is given in the ‘type_descriptive_statistics' column. The following column indicates a single variable that is used in the model.

Numerical summarisation of the objective function is performed as well. The objective function in this study is NPS. In global, the time performance can be divided into arriving; too-early, on-time and too-late. The NPS scores for too-early, on-time and too-late arrivals must be analysed extensively in order to use the NPS scores to compute new NPS predictions belonging to machine learning models. In section 4.2.2 an extensive description is given about the historical NPS score numerical analysis.

3.5.2 Summarizing Data Graphically

The term ‘exploratory visualisation' is often used in literature to indicate graphical exploration of data structures (Shmueli & Koppius, 2011). Exploratory visualisation helps in detecting variable commensurateness. Variable commensurate refers to the distribution of the input variable values in the model. The distribution of values for six important input variables is provided later in this report in section 4.2.1.

Same as for numerical data summarisation, also graphical summarisation is applied to the model objective function (NPS). In section 4.2.2, graphical summarisations of historical NPS scores are provided to underline the numerical historical NPS observations.

3.6 Model Specific Data Preparation

In this section, model specific data preparation steps are discussed. The model-specific data preparation steps are distinct for all different models created in the machine learning prediction model. This is in sharp contrast with the earlier specified general data pre-processing operations that are applied as input of all models in the machine learning prediction tool. In total four model-specific data preparation steps are discussed in the remainder of this section. The four model-specific data preparation steps are;

1. Variable Selection
2. Handling Outliers
3. Variable Scaling
4. Data Partitioning

3.6.1 Variable Selection

The number of variables available to solve problems in this advanced digital time is extensive. As indicated in Table B.1 in the appendix, 26 original variables are used in this service time prediction problem. It is very important to select the right set of features as input for the prediction model. As with any other model, capturing the reality is mostly dependent on the description of the reality. In In machine learning models, the most predictive variables should be used. Most of the variable in this study could directly be extracted from existing company databases. Some other variables were created adding extra questions to an existing 'post-delivery-survey' to discover new information. In section 3.3.3, it is indicated that the unique combination of Sub Product, delivery service id, normal-delivery/pickup/swap and size-category variables taken together as a unique variable value. This resulted in a new column that is created from concatenation of other variable columns. The importance of selecting the right set of variables is underlined by the statements of Guyon & Elisseeff (2003). They show that under-specified models can sometimes produce better predictions. Retaining irrelevant variables to a dataset sometimes "confuses" machine learning algorithms (Witten & Frank, 2011, p. 288). Guyon & Elisseeff (2003) even prove that it is sometime beneficial to remove some of the less influential variables from the prediction model in order to reach more accurate predictions. These two proves give enough reason to check machine learning prediction by inputting a lower variable space, thereby reducing the variable space. However, note that variable reduction is a complex study area, which is classified as NP-hard (Almaldi & Kann, 1998). An extensive field of variable reduction methods exist, each of them claiming to provide high variable reduction performance.

In global, variable selection can be reached by applying in one of the following three methods:

- Filters
- Wrappers
- Embedded Methods

The three distinct variable reduction methods enumerated above are different from each other in the level of inclusion of the method within the machine learning algorithm.

Filters reduce variables entirely before the deployment of machine learning model predictions. Filters thus can be considered as a pre-processing step and are completely independent of the machine learning prediction phase.

Wrappers make use of machine learning models to reduce the variable space. Wrappers compare a subset of variables with the original complete set of variables. As an example, Support Vector Regression (SVR) methods can first be used to select the most important variables. Next, these most important variables are selected and inserted to the prediction model (XGBoost in this study) as input variables.

Embedded Methods include variable subset testing into the machine learning process. Instead of using the feature importance to delete unimportant variables, embedded methods just use the machine learning algorithm itself to select important variables. For example, decision trees use a nested subset method, a specific embedded method type, by selecting variables based on the coefficient of variation.

By reviewing Guyon & Elisseeff (2003) and Witten & Frank (2011) in total, ten possible variable reduction methods are obtained. The ten variable reduction methods (indicated with variable reduction types) are;

1. Manual Variable Reduction (filter)
2. Machine Learning Method Variable Reduction (wrapper / filter)
3. Machine Learning Model Internal Process Variable Reduction (embedded method)
4. Backward Elimination (wrapper)
5. Forward Selection (wrapper)
6. Statistical Significance Tests (filter)
7. Normal and Hierarchical K-Means Clustering (filter)
8. Incremental Clustering (filter)
9. Probability-Based Clustering (filter)
10. Bayesian Clustering (filter)

In total five of the variable reduction methods are implemented in the service time prediction tool. The in total four variable reduction methods used in this study are described below.

3.6.1.1 Manual Variable Reduction

The first and most straightforward method of variable reduction in the service time prediction tool is manual variable reduction. The name of this variable reduction method is very indicative. Specific variables are reduced by hand. Mostly, input variables are deleted from the variable input space based on expert advice. Experts are referring to persons having a thorough understanding of the system that is analysed. Experts in this study are company workers that are executing work related to the delivery operations. Examples are planners, continuous improvement employees, team leaders and delivery employees.

3.6.1.2 Machine Learning Method Variable Reduction

Besides operating as a predicting method, machine learning models can also operate as (pre-processing) variable reduction wrapper or filter. Machine learning models themselves can perform

variable reduction since they include the ability to rank the input variables on importance. For example, decision tree algorithms, as the XGBoost algorithm, have a variable importance method implemented in the wrapper code. In general, in python, the 'Select From Model' method can be used as a variable reduction wrapper. Besides, the wrapper variable reduction method described above, it is also possible that a machine learning method is used as a pre-processing variable reduction filter. A filter is different from wrapper in that it is completely separated from each other, still placed in sequence. The XGBoost algorithm filter for example first filters the most important variables. Next, the most important variables are used by another predictor, which could be for example a K-Nearest-Neighbours algorithm. According to Witten & Frank (2011), the performance of the K-Nearest-Neighbour algorithm can surely be improved by using different machine learning methods as variable reduction filter. This is predominantly caused by the fact that the K-Nearest-Neighbours is very susceptible to extraneous variables.

3.6.1.3 Machine Learning Model Internal Process

Some machine learning models have a regularisation component inhibited in the internal machine learning structure. The regularisation component reduces the number of variables automatically. GLM net regression, neural networks and ensemble tree methods are examples of algorithms that include a regularisation term. GLM net regression filters the input variable space based on regularization of the coefficient in the regression model. Neural Networks create second-order constructs in the hidden layers by making linear combinations of input variables algorithms (Witten & Frank, 2011, p. 227). In the second order construct creation process, some weights between input variables and second-order constructs are set to very low values, in this way regularizing the importance of input variables. In case of regularizing the input variable contributions to zero, this means that the information from input variables is not incorporated in the second order constructs at all. Not including information from some variables, in fact, is like the deletion of the variables. Thus, machine learning algorithms regularise the input variable space just by the construction of the algorithm. The main machine learning method used in the machine learning prediction tool is the XGBoost prediction method. The XGBoost algorithm prunes trees based on the coefficient of variation. This process is explained in more detail later in section 3.7.1.1.

3.6.1.4 Cluster Method

Besides variable selection methods, also a variable creation method is example, which is called the cluster method for convenience. The cluster method approach is applied in two ways. First, for in total 13 variables, mean service time and mean prediction errors variables are created for distinct variable values. As an example, the mean service time and mean prediction error of all Washing Machines are computed. Next, these newly created mean service time and mean prediction errors are clustered in bins of 0.5 minutes. The reasoning behind applying this clustering method relies in the statements made by Guyon & Elisseeff (2003, p.1165). They indicate that; "A variable that is completely useless by itself can provide a significant prediction performance improvement when taken with others." This initial clustering is followed by a next, more high-level clustering. In this second order clustering, the mean service time values are averaged, and mean prediction error values are summed up to form two new variables. These variables are called 'All Summed Up Mean Service Time' and 'All Summed Up Mean Prediction Error' respectively. The contribution of the two types of clustering methods described in this section on the service time prediction is explained in chapter 5 'Results'.

3.6.1.5 Backward Elimination (Recursive Feature Elimination)

Forward selection and backward elimination methods add or reduce variables respectively based on the evaluation of the error prediction after one variable change in the variable space region. In forward selection, a low number of variables is started with, but sequentially more variables are added. On the contrary, backward elimination reduces the input variable space of the data, by eliminating unimportant variables sequentially. The backward elimination method is not used in the service time prediction model because it has a very long time to converging to an optimal dataset, in which features are eliminated.

3.6.1.6 Statistical Significance Tests

The next variable reduction method is variable reduction based on a statistical significance test. Statistical significance test measures the difference in predictive performance between the independent variables and the dependent variable (Witten & Frank, 2011). Independent variables showing a low statistical significance test performance result are removed if the predictive performance of those variables is below a certain significance threshold. Various statistical methods can be used. The χ^2 test measures the similarity between two variables. In general, the following formula is used to define the chi-square test: $\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$. In this formula, O_i refers to the independent variable and E_i refers to the dependent variable. χ^2 test provides a the significance indicator 'p-value' as output. A p-value below 0.05 is indicates a significant relation between the independent variable and the target variable. Besides the χ^2 test, the statistics method 'mutual information regression' is used to compare the target variable with the independent variables. Mutual Information measures how much information is communicated, on average, in one variable to another (McEliece, 2013, p. 4). The definition and definition formulation of the mutual information function is copied from the article of Erik Miller (2013, p. 4):

> *"The formal definition of mutual information of two random variables X and Y, whose joint distribution is denoted by P(X; Y) is given by:"*

$$I(X;Y) = \sum_{x \epsilon X} \sum_{y \in Y} P(x,y) log \frac{P(x,y)}{P(x)P(y)} \quad (3.1)$$

To limit the scope of this research and focus more on the other aspect of feature selection, the statistical significance test in not used in the service time prediction model.

3.6.2 Handling Outliers

Outliers can be characterised as observations that have significantly different values compared to most of the observation values. It is essential to detect and delete outliers because outliers do not represent the data well. Suspicious observation values may be outliers, but interpretation is needed to assess suspicious data points as being real outliers (Rousseeuw & Hubert, 2011, p. 73). Three different types of outlier detection techniques can be categorized, which are:

1. Univariate Outlier Deletion Methods
2. Robust Scaling Operations
3. Advanced Outlier Deletion Methods

In the remainder of this subsection, those three outlier techniques are described. In the execution of this study, the univariate outlier deletion method and the robust scaling operation method is used as pre-processing techniques in the service time prediction model.

3.6.2.1. Univariate Outlier Deletion Methods

Univariate outlier deletion methods outlier techniques that make use of the specific range of the variable values themselves. Univariate outlier deletion methods assume that the variables are independent identically distributed (i.i.d.) from other variables (Rousseeuw & Hubert, 2011, p. 73). Three distinct types of univariate outlier deletion methods can be distinguished:

- Interquartile Range Outlier Deletion
- Z-Scores Outlier Deletion
- Quantile Outlier Deletion

Interquartile range outlier deletion uses the theory of boxplots to compose bounds for outlier deletion. Data points not belonging to the range indicated in Formula 3.2 are considered to be outliers (Rousseeuw & Hubert, 2011, p. 74):

$$\{Q1 \; - \; 1.5 * IQR \mid Q3 \; + \; 1.5 * IQR\} \tag{3.2}$$

The z-score outlier deletion is calculated the same way as standard scaling is calculated. The z-score is determined by subtracting the mean variable value from the individual data point observation value and subsequently divided by the standard deviation (Rousseeuw & Hubert, 2011, p. 73). Subsequently, the x percentage of the highest z-scores and x percentage of the lowest z-scores are deleted from the data. The specific value of x needs to be determined. It is possible to define the value of x as hyperparameter in the service time prediction model. The Z-score formula can be described as:

$$z_i = \frac{(x_i - \tilde{x})}{s} \tag{3.3}$$

Last, the quantile outlier deletion method deletes variable values outside a specified quantile range. For example, setting the quantile range as $\langle 0.01 | 0.99 \rangle$ means that all 1 percent of lowest variable values and all 1 percentage of highest variable values are deleted from the data. Based on the simplicity in theory of this quantile outlier deletion method, accompanied with the flexibility in setting input parameters, the quantile outlier deletion method is used in the service time prediction model.

3.6.2.2. Robust Scaling Operations

Robust scaling operations make changes to the range of variable values. Besides absolute variable value reduction, also the relative distance between variables is decreased. In section 3.6.3, four types of scalers are described in detail. All those four scalers can be classified as being robust. The four scalers are:

- Logarithmic Scaling
- Polynomial Scaling
- Standard Scaling
- Robust Scaling

Note that the standard scaler does not reduce the relative variable value distance a lot and thus is not a very strong robust scaling method.

3.6.2.3. Advanced Outlier Detection Methods

Advanced outlier detection methods make use of machine learning methods to classify observation as being either normal value or outliers. Advanced outlier techniques are increasingly becoming popular since they are more applicable with the current availability of fast RAM processors in modern computers. The performance of advanced outlier techniques is good in most situations. The most widely used advanced outlier detection methods are:

1. Isolation Forest
2. Local Outlier Factor

Isolation Forest makes use of the Random Forest tree ensemble method. In essence isolation forest checks the path length from the root node to the terminal node for all unique values of an observations. Isolation Forest is classifies the observations as outliers, if the observation follow a relatively short path to the terminal leaf (Liu, Liu, Ting, & Zhou, 2008, p. 3). The logic behind the short path theory used by the isolation forest algorithm is that outliers mostly cause extremely high values of the target variable. Consequently, to decrease the prediction error, the outlying variable value is split at a high level in the decision tree. Especially in high-dimensional datasets, isolation forest is very useful (Liu, Liu, Ting, & Zhou, 2008, p. 2). Like the good performance of random forest in case of large variable space dimensions, isolation forest effectiveness is high when a lot of irrelevant variables are included in the dataset. Isolation forest includes a linear time complexity and only needs a small memory requirement (Liu, Liu, Ting, & Zhou, 2008, p. 2).

The local outlier factor (LOF) technique is based on the Nearest-Neighbours algorithm. It attaches to the observations an 'degree' of being an outlier, which is called the LOF. The term local refers to the density of neighbours in the region in which the data observation is located (Breunig, Kriegel, Ng, & Sander, 2000, p. 1). If the data observation lives in a less dense region in comparison to the density region of its neighbouring data observations, the algorithm assigns a higher outlier value to the former data observation in comparison to the later data observation. Density regions of neighbour data observations give the algorithm an indication of expected average density levels of the complete dataset. Formula 3.4 below gives the local outlier factor of observation 'p':

$$LOF_{MinPts}(p) = \frac{o \in \sum_{N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|} \quad (3.4)$$

In the LOF formula provided above, the average value of observation p's local reachability density (denoted as lrd) is compared with the lrd of its neighbours. The symbol o indicates the neighbouring data observations (Breunig et al., 2000, p.4). Consecutively, A higher LOF value is obtained by low values of p's own local reachability density and high values of its o neighbouring data points local reachability density.

To limit the scope of this service time prediction research and bypass the negative consequence (long run time) of these outlier detection algorithms, the decision is made not to test the LOF and Isolation Forest in the service time prediction model.

3.6.3 Variable Scaling

Variable scaling is the next model-specific data preparation step used in the service time prediction model. Variable scaling transforms variables values to new values by applying a scaling function. In general, there are four reasons why scaling is applied. First, scaling is an effective method to reduce the impact of outliers. Especially robust scalers can detect and delete outliers. Second, scaling helps algorithms to interpret the value ranges of different variables better. Especially distance-based algorithms are sensitive to inequality of variable values over different variables. Examples of distance-based algorithms are; PCA and K-Nearest-Neighbours. Scaling helps creating a more linear relationship between the independent variable and the dependent variable. Linear predictors performance can be improved with scaling the variable values. Linear regression, support vector regressors (SVR) and neural networks are examples of linear predictors for which variable scaling is likely to be helpful. This could be explained by just referring to the name of this prediction class 'linear', because linear methods can best predict the data if the data is linearly separated.

It should be noted that decision tree models, like the baseline XGBoost model, performs quite well for unscaled data. This is because decision tree models are not belonging to the distance-based model classifications. XGBoost can make variable value splits anywhere in the variable value range. This means that XGBoost is not linearly interpolating results for variable values from different variable values. In all cases scaling the data potentially speed up the gradient descent method. That is why scaling is applied for input datasets of the XGBoost model. Four types of variable scalers are used in the service time prediction script:

1. Logarithmic Scaling
2. Standard Scaling
3. Robust Scaling
4. Polynomial Scaling

3.6.3.1 Logarithmic Scaling

Logarithmic scaling of variables is a commonly used method to increase the predictive performance. There are some situations in which logarithmic scaling is often beneficial. The first effective situation is when the dependent variable and the independent variables do more resemble an exponential relation than a linear relation. By applying a logarithmic scaling to the dependent variable, the relation between the independent variable and the dependent variable will be more linearly shaped. Logarithmic scaling of the dependent variable in a simple linear function example can mathematically be defined as:

$$\log(y_i) = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_n x_{ni} \tag{3.5}$$

OR the Inverse Function;

$$y_i = e^{\alpha} + e^{\beta_1 x_{1i}} + e^{\beta_2 x_{2i}} + \ldots + e^{\beta_n x_{ni}} \tag{3.6}$$

The second situation is when the relation between the dependent variable and the independent variable is very heteroskedastic. Heteroskedasticity indicates a dependent variable on the independent variable relation in which the error term does not show a normal and consistent distribution around the linear distribution line. By applying a logarithmic scaling to both the dependent variable and the independent variable, the error term will be scaled more towards the linear

distribution line. In Figure 3.2 below, the transformation from heteroskedastic data toward a more homoskedastic data, by application of logratihmic scaling, is depicted.

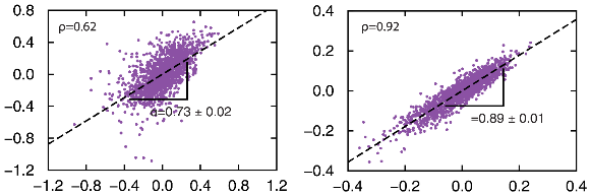

Figure 3.2. Logarithmic Scaling Power: from a heteroskedastic relation to a homoskedastic relation

Last, logarithmic scaling will create relations that are more normally distributed. The magnitude of left or right skewness will be decreased. Most of the algorithms can better interpret input data if the relation between the dependent variable and the independent variable show a normal distribution.

3.6.3.2 Polynomial Scaling

Polynomial scaling helps in finding relations between variables by changing the magnitude of variable coefficients. In Formula 3.7, the application of a polynomial scaling on a linear function is expressed:

$$y_i = \alpha + \beta_1 {x_{1i}}^n + \beta_2 {x_{2i}}^n + \ldots + \beta_n {x_{ni}}^n \tag{3.7}$$

By using polynomial scaling, the relation between the dependent variable and the independent variable transforms from being a linear relation towards a polynomial relation. Some variable relations do more resemble a polynomial relation than a linear relation. In Figure 3.3 below an example is provided that indicates the effect of using a polynomial scaling operation to assist in finding the right data separation line.

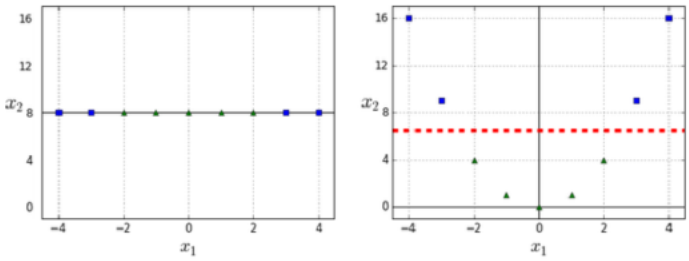

Figure 3.3: Polynomial scaling power: turning an inseparable data space into a separable data space

Additionally, polynomial scaling in some cases helps stochastic gradient descent algorithms to converge faster. The polynomial scaler in most cases is a powerful scaler and is thus integrated as pre-processing operation in some the service time prediction model.

3.6.3.4 Standard Scaling

The standard scaler redistributes the variable values on a new axis scale length where the variable mean ($\bar{x}$) is assigned the value 0 and one standard deviation (σ) from the mean is assigned the value 1. Another name for standard scaling is standardisation. Standard Scaling is a robust scaling operation because relative variable value distances are changed. The further away an observation value is located from the mean value of the variable, the more the observation value will be scaled towards the mean. The high-level motive of the standard scaling operation is trying to fit the variable value distribution as a normal distribution. Most of the algorithms can better interpret input data if the relation between the dependent variable and the independent variable show a normal distribution. Therefore, standard scaling is a popular pre-processing scaling operation, which is used service time prediction model as well. The mathematical formulation of the standard scaler is provided in Formula 3.8:

$$x' = \frac{x - \bar{x}}{\sigma} \tag{3.8}$$

3.6.3.6 Robust Scaling

Robust scaling is very similar to min-max scaling. Instead of assigning the minimum and the maximum values as boundary values, the robust scaler uses the Q1 and Q3 values as boundary values. The Q1 and Q3 variable values represent the variable values assigned to 25% and 75% of the variable value range respectively. By making this change to the min-max formula, the robust scaler, scales observations with a very high or very low value relatively more than observations that have less extreme values to the mean. Robust scaling, as the name indicates, is more robust to the effect of outlying observations in the data. The robust scaling operation can mathematically be formalised as:

$$x' = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)} \tag{3.10}$$

3.6.4 Data Partitioning

Last model-specific data preparation step is data partitioning. In data partitioning, the dataset is split into a distinct part to enable both training and testing processes. The decision about composing data partitioning cut-off point is of central importance since different partitioning solution can result in very different prediction outcomes. In general, there is one universal rule that needs to be followed while creating data partitions; the partitioned dataset that is used to test model's performance must not be used in any way to train the model and finetune the parameters (Witten & Frank, 2011, p. 149). In this project, the dataset is split into two distinct parts; training set and test set. By using cross-validation, 80% of the training set is used to train the model. The remaining 20% of the training set is used to test the trained model performance. This 20% training set split off is called the validation set. The validation set is used to test the training set performance while input hyperparameter are incrementally improved. The test set is used to test the trained model on holdout test dataset. By using a holdout test dataset, which is representative in modelling real system behaviour, the realistic predictive performance of the model can be determined. In literature, no consensus has been reached about the best dataset split distribution between train and test set. Each prediction problem is unique, meaning that the test dataset size that can models real system performance best differs for each problem. Therefore, there always needs to be a made trait-off between prediction accuracy and test accuracy. In general, the following rule holds; the more data that is fed as input training set to the

prediction algorithm, the better the predictions will be (Witten & Frank, 2011, p. 149). However, testing a good regressor on limited data does not give a good notion of the error estimate. The training set/test set split rule of thumb is the partitioning of the complete dataset into 30% test set and 70% training set. Besides the training set/test set data partitioning, also the model parameters (referred to as hyperparameters) must be optimised. The initial dataset, consisting of 310,595 orders, is considered large enough to split off parts of the data as a test set. It is not needed to solely make use of cross-validation because the dataset is not large enough. The default train set partitioning percentage of 70 percentage used, provide us 240.000 orders to train the data on, which still is an extensive dataset for creating a good model. The most popular data partitioning method used in predictive analytics to optimise hyperparameters is cross-validation. Cross-validation splits the data in a number of folds (n), trains the data on $(n-1)$, folds and checks performance by predicting the trained model for the remaining $(n-(n-1))$ folds (Witten & Frank, 2011, p. 153). Mention that extensive tests on numerous different datasets, with different learning techniques, have shown that using 10 folds is about the right number of folds to get the best estimate of error (Witten & Frank, 2011, p. 153). However, I. H. Witten and E. Frank place this statement into perspective later, by mentioning that using 5- or 20-fold cross-validation is almost as good as using 10-folds. Therefore 5-fold cross-validation is used in the service time prediction model. Last, one decision needed to be made, which is determining the type of cross-validation. Three cross-validation variants are applicable to the service time prediction problem, which are:

1. K-Fold order based Cross-Validation
2. Shuffled day/tour based Cross- Validation
3. K-Fold Shuffled day/tour based Cross-Validation
4. Leave-One-Out Cross-Validation

3.6.4.1 K-fold Cross-Validation

K-fold cross-validation creates folds by combining observations in chronological time. It assures that each of the observations is ones being represented in the validation dataset. This property deducts the adverse effect that would arise when some of the observations are represented more than ones in the complete dataset. The procedure used by K-fold cross-validation is not correctly enabling prediciton of the new system. K-fold cross-validation induces that predictions are being made for several weeks or even months ahead. In reality, one day in advance, the complete set of delivery orders of the next day is clear. Therefore, the service time predictions be created one day in advance. The misalignment in prediction horizon created by usage of K-fold cross-validation is undesirable and therefore not included in the service time prediction model. Figure 3.5 below visualises the K-fold cross-validation scheme.

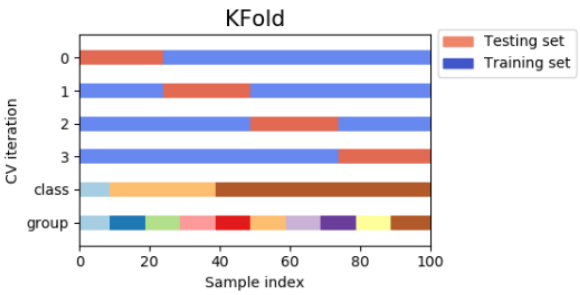

Figure 3.5. K-Fold cross-validation scheme

3.6.4.2 Shuffled Cross-Validation

The decision was made to include shuffled cross-validation as one of the cross-validation options in the service time prediction tool. Tests have shown that predictive performance can be improved by making use of shuffled cross-validation instead of K-fold cross-validation. In contrast to K-fold cross-validation, the shuffled cross-validation process include information from day n to make predictions for day n + 1. In essence, shuffled cross-validation creates random splits of the observations in the input dataset. The shuffled cross-validation process used in this study is adjusted slightly. Not individual observations are taken randomly from the input dataset, but unique days or tours in the dataset are randomly shuffled and clustered in folds in the input dataset. Day based shufflesplit protects for data leakage in the training process. Splitting on the order_id aggregation level potentially leads to data leakage between training and test data. By using order_id based splitting, the model already knows, for example, some information like the working pace of delivery employees on a particular day. There is one more reason why shuffled cross-validation is prefered over K-fold cross-validation. Two input variables have missing values for the first several months of the training dataset. By using k-fold cross validation, some validation fold will not include any information from those two variables, resulting in inaccurate performance measurements. This is in opposition to the most important data partitioning rule mentioned in section 3.6.4 stating that about equal proportions of the complete dataset should be represented in the training set and the test set. Figure 3.6 below visualises the shuffled cross-validation scheme.

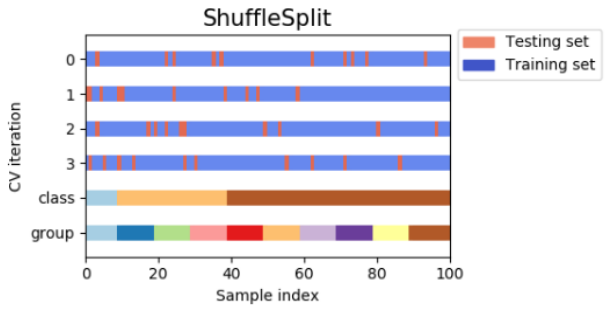

Figure 3.6. Shuffled cross-validation scheme

3.6.4.3. Shuffled K-fold Cross-Validation

Besides ordinary shuffled cross-validation, also a shuffled K-fold cross-validation scheme is implemented in the service time prediction tool. The shuffled K-fold cross-validation scheme is best describing real model behaviour and therefore used as default cross-validation method in this study. The shuffled K-fold cross-validation first shuffles the data on the day based aggregation level. Next, K-fold cross-validation is applied to the data. By using this cross-validation method the information from day n is included to make predictions for next day n + 1. All days exactly ones are part of the validation fold. This takes away the effect of randomness in the creation of folds, resulting in more reliable prediction estimates by averaging all folds predictions.

3.6.4.4 Leave-One-Out Cross-Validation

Leave-one-out cross-validation is able to include information from day n for making service time predictions for day n +1. Despite the leave-one-out cross-validation is a proper method for the service time prediction problem, the decision is made not to make use of this cross-validation scheme. The reason for not including the leave-one-out cross-validation scheme in this study is because the complete leave-one-out process is computationally too expensive. The leave-one-out cross-validation works as follows. Sequentially one of the observations is left out to identify the validation set. The remaining observations are combined and together form the training set. After the training set model is tested on the validation set, the process starts again by leaving out the next observation from the dataset. The leave-one-out cross-validation can be considered as an extensive K-fold cross-validation scheme, for which in turn, all the variables are ones forming the validation fold. This means that the number of folds is equal to the number of days, which is determined to be inappropriately large.

3.7 Choice of Methods

In this study, a structured machine learning approach is developed for generating package delivery service time predictions. In section 3.2 it is mentioned that the current Coolblue service time prediction method generates deterministic service time predictions. It lacks the capability to include more than just the general order information (Product Type and Delivery Service information) in the prediction process. A lot of information is available in Coolblue databases enabling us to develop a machine learning prediction that learns from the information that is provided to the model. Machine learning models are popular nowadays for creating prediction models. They mostly outperform manually created rule-based prediction models, because machine learning models are able to detect the complex relations that are existing between variables in the dataset (Chen & Guestrin, 2016, pp. 1). The machine learning models used in this study is XGBoost, often classified as a tree ensemble learning method. Machine learning models come with a lot of parameters and hyperparameters. Both of them need to be tuned in order to reach accurate predictions. In section 3.7.2 the parameter tuning process is explained. Last, the performance of the predictions is described in terms of customer satisfaction (NPS). This transformation method from transferring service time predictions into expected customer satisfaction scores is described in section 3.7.3.

3.7.1 Machine Learning Algorithm Used

In this modern and digitally advanced world, a lot of machine learning algorithms are available. Each algorithm has its own pros and cons. The performance and the efficiency of the algorithms are mainly dependent on the problem it tries to solve and the dataset that is input to the machine learning algorithm. For selecting proper algorithms, algorithm recommendations in scientific articles are examined. Tianqi Chen and Guestrin (2016) mentions that the gradient boosting decision tree method ‘XGBoost' turns out to be the winning solution in 17 online machine learning competitions released on the machine learning platform Kaggle[1], from the in total 29 challenges published on this platform. The decision is make use of the XGBoost as default prediction algorithm in this study.

XGBoost is an evaluation of the older Random Forest decision tree algorithm. Random Forest creates an extensive set of decision trees and subsequently combines the result of the individual trees. In this way, the error prediction inaccuracy of one decision tree is shadowed by the better accuracy of

[1] Kaggle is the world's largest community of data scientists and machine learners, owned by Google Inc. Kaggle got its start by offering machine learning competitions and now also offers a public data platform, a cloud-based workbench for data science, and short form AI education.

the other decision tree. However XGBoost does this in a much smarter way. It accurately locates the inaccuracy of one decision tree and transports it to the next decision tree. Moreover, XGBoost is a sparsity and catch aware algorithm, enabling relatively fast computation. This are the main reason swhy XGBoost is used in the study and thus preferred over Random Forest.

Besides preferring XGBoost over Random Forest, XGBoost is also preferred over the competitor algorithm Neural Network. The first reason is already mentioned; the XGBoost algorithm turns out to be the winning algorithm in most of the machine learning competitions released on Kaggle (XGBoost winning 17 times vs Neural Networks winning 11 times). The second reason is the underperformance of Neural Networks in cases of low covariance between the dependent variable and independent variable. Benjamin et al. (2017) show that Neural Networks performs 1.5 times worse than XGBoost in predicting neural spikes. Benjamin et al. (2017) expect that the low covariance causes the underperformance of the Neural Network algorithm among the variables in the dataset. In total, they tested the performance of XGBoost, Neural Network, Ensemble Methods and GLM on three different datasets that include neural data. For all the three datasets, XGBoost is outperforming the Neural Network algorithm. In addition, Benjamin et al. show that the XGBoost algorithm is only outperformed by an Ensemble method. The Ensemble method combines the predictions of all four pre-mentioned algorithms and used this set of combined predictions as input for a 'second level' XGBoost algorithm. The Ensemble prediction method is not tested in this study since the predictive performance gain is minimal in comparison to the XGBoost algorithm. In the next section, it is described how the XGBoost algorithm works and why it creates such very accurate predictions.

3.7.1.1 XGBoost

The XGBoost algorithm belongs to the class of gradient boosting decision tree algorithms. Decision trees algorithms create trees from sequential if-then rule split. Starting from the root of the trees, in each split the dataset is split based on a specific variable value. The decision tree evolves out wide adding more tree levels. Eventually, all observation end up in one of the terminal leaves of the tree. The average recorded service time of all the observations in one single leave determine the service time prediction. Decision trees models are constructed by usage of historical data. This process is called 'training'. In the training phase, a subset of the complete dataset, called the 'train set', is provided. The XGBoost algorithm creates node splits by using the coefficient of variation (CV) property. Decision tree algorithms can handle outliers in the datasets better than most other algorithms. The high performance accuracy can be attributed to the very powerfull coëfficient of variation (CV) method included in decision trees. CV is measured on individual features. Features having the highest relative standard deviations are selected for creating the the next splits. By splitting features with relatively high standard deviation, the prediction gain can be improved most. Computation of the coefficient of variation (CV) formula for a feature (F) is computed as follows:

$$CV\ (F)\ =\ \frac{Std}{\bar{x}} * 100\% \tag{3.11}$$

$$Std\ =\ \sqrt{\frac{\sum(x-x)^2}{n}} \tag{3.12}$$

$$where,\ \bar{x}\ =\ average$$

$$n\ =\ count$$

In contrast to classical decision tree algorithms, the XGBoost algorithm creates multiple decision trees. The reasoning behind using a complete set of decision trees instead of one is that one decision tree is sensitive to inaccurate parameter settings in the input dataset. To reduce this inaccuracy present in a single decision tree, the XGBoost algorithm creates multiple decision trees that are boosting the prediction results towards better and better predictions.

Each Individual decision tree use K additive functions to predict the output (y_i). The resulting output is given in Formula 3.12 below. Each function f_k corresponds to an independent tree structure (q) with leaf weights (w) (Chen & Guestrin, 2016). Here q represents the structure of each tree that maps an example to the corresponding leaf index. T is the number of leaves in the tree (Chen & Guestrin, 2016).

The predictive performance of the individual decision trees is combined in the regularised boosting objective $\mathcal{L}(\phi)$ given in formula 3 below. $\mathcal{L}(\phi)$ sums up the individual tree prediction errors that are multiplied by a differentiable convex loss function (l) and regularises it with a regularisation term Ω. According to Chen & Guestrin(Chen & Guestrin, 2016), the regularisation term is the secret power of XGBoost which helps to create a model that is simple but still achieves very accurate predictions. The regularization term Ω furthermore fights against undesirable overfitting predictions. It does so by penalizing the complexity of the model.

Regularisation of the prediction outcomes happens in two distinct ways in the algorithm. In Formula 3.13 below, it can be found that first of all there is a regularisation term T. T stands for the number of leaves in the tree. T is multiplied by parameter gamma. Furthermore, there is a regularisation term w, denoting the individual tree leave weights. A tunable parameter λ is placed in front of w. The regularisation boosting objective $\mathcal{L}(\phi)$ is the reason why XGBoost is outperforming other algorithms. This boosting regularisation term is very powerful, and therefore this boosted regularisation objective is described in more detail. In section 3.7.2.1.2. In the next subsection 3.7.2.1.1, the single decision tree construction is described.

$$\hat{y} = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), \qquad f_k \in F \tag{3.12}$$

$$F = \{f(x) = w_{q(x)}\}(q: \mathbb{R}^m \rightarrow \mathrm{T}, \mathrm{w} \in \mathbb{R}^T) \tag{3.13}$$

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{3.14}$$

$$where\ \Omega(f) = \gamma T + \frac{1}{2}\lambda ||w||^2 \tag{3.15}$$

3.7.2.1.1 Individual Decision Tree Construction

As mentioned earlier in this subsection, XGBoost creates multiple decision trees from subsets of the training data. In bagging decision tree ensembles models, the subsets are created by averaging individual decision tree predictions. In contradiction to bagging tree ensembles, boosting tree ensembles do not average the individual decision trees, but sum the individual tree results. There are more differences between bagging and boosting, for example in the creation of new decision trees. In essence, decision trees are in XGBoost by focusing more on dataset observation that gave poor predictions. XGBoost creates subsamples of the training dataset based on error prediction of observations in previous XGBoost decision tree iterations. This means that observations with high prediction errors, measured in previous XGBoost decision tree creation iterations, got assigned a higher value than observation with a low prediction error in previous XGBoost decision tree iterations.

3.7.1.1.2 Gradient Tree Boosting Method

Last, the gradient tree boosting method used by XGBoost is described in this section. Gradient methods make iterative steps, in the direction toward the minimum of the derivation of the loss function $\mathcal{L}(\phi)$. Mathematically, the corresponding optimal value of the loss function is represented as:

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i+\lambda} + \gamma T \tag{3.16}$$

Besides the predictive accuracy, the XGBoost algorithm is constructed with the inclusion of some powerful advanced features. In total five advanced features are added to the XGBoost model making the algorithm highly scalable, efficient and sparsity aware. The five powerful features are given below in dotted enumerations. The inclusion of those powerful features inside the XGBoost algorithm makes the complete XGBoost algorithm highly scalable and able to deploy billions of observations by using far fewer resources than existing algorithms (Chen & Guestrin, 2016). The five advanced features are:

- Approximate Algorithm
- Sparsity-Aware Split Finding
- Weighted Quantile Sketch
- Block structure for Parallel learning and Out-of-Core Computation
- Cache-Aware Access

3.7.2 Hyperparameter Optimization

Optimising a machine learning model is a complex process. Apart from the extensive set of preprocessing operations, that could be applied and the decision to select a proper algorithm, also the hyperparameters need to be tuned. Hyperparameters are different from normal parameters since normal parameters are set to a constant value. It looks very counterintuitive that parameters take on a constant value. However constant values in parameters refer to the stability in parameter values while creating the prediction model. The reason for assigning a constant value to parameters in training phase is based on either the study objective, expert knowledge or just because default settings are used.

On the other hand, hyperparameters are parameters that need to be tuned in model training phase in order to minimise the objective function. Finding good hyperparameter values can be done in three different ways; manual, semi-manual or automated (Bergstra, Bardenet, Bengio, & Kégl, 2011, p. 1). The manual hyperparameter search strategy is called; manual hyperparameter selection. The semi-manual hyperparameter search strategy is called; grid search. Last, there are two automated hyperparameter search strategies; random search and Bayesian search.

In this study, we make use of the Bayesian hyperparameter search. The specific name of the Bayesian search model used in the service time prediction tool is; Tree-structured Parzen Estimator (TPE). The reason for selecting the specific Bayesian hyperparameter search strategy technique type TPE is twofold. First of all, TPE is theoretically better than its counterparts Grid Search and Random Search. TPE belongs to Bayesian optimisation methods, which are characterised for makes use of known information to enhance predictions (Tipping, 2004, p. 1). Concretely, TPE uses a smart learning approach to test hyperparameter sets sequentially. Manual search, grid search and random search lack such smart learning approaches. Second, TPE shows better performance on the objective function in comparison to the three pre-mentioned competing hyperparameter optimisation techniques. J.

Bergstra et al. (2011) compare four types of hyperparameter optimisation techniques on two prediction problems; TPE, Gaussian Process (GP), Manual Search and Random Search. The performance of the four hyperparameter optimisation techniques is given in Figure 3.12 below. For both of the investigated datasets, TPE outperforms the other three hyperparameter optimisation methods. It even achieves an error reduction of 4.50% and 2.84% for the first and second dataset respectively in comparison with manual search hyperparameter selection technique.

	convex	**MRBI**
TPE	**14.13** ± 0.30 %	**44.55** $\pm 0.44\%$
GP	$16.70 \pm 0.32\%$	$47.08 \pm 0.44\%$
Manual	$18.63 \pm 0.34\%$	$47.39 \pm 0.44\%$
Random	18.97 ± 0.34 %	$50.52 \pm 0.44\%$

Table 3.1. Performance four hyperparameter selection techniques (TPE, GP, Manual Search, Random Search) on the convex dataset and the MRBI dataset.

3.7.2.1 Method Used By TPE

The TPE optimisation process is graphically presented in Figures 3.7 and 3.8 below and explained in the remainder of this section. First of all, remind that an objective function is always required in order to select better hyperparameters. The objective function in the service time prediction model can be expressed as $x^* = arg \min_{(x \in X)} f(x)$, where x^* is the optimal set of hyperparameter for the objective function expressed relative to the feature space $f(x)$. The optimisation of this objective function is known as an NP-hard problem, therefore very difficult to solve.

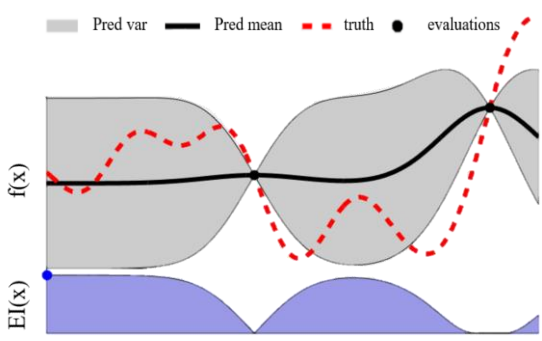

Figure 3.7. 2nd Iteration TPE Hyperparameter Optimization

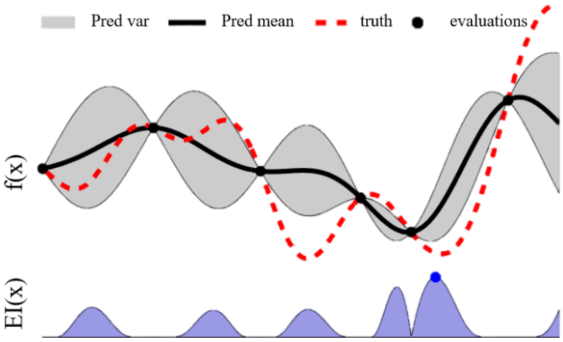

Figure 3.8. 6th Iteration TPE Hyperparameter Optimization

To solve the NP-hard objective function optimisation problem, TPE uses an evolutionary mixture model. The evolutionary mixture model can be classified as a greedy sequential method and is called; sequential model-based global optimisation (SMBO). The SMBO tries to uncover the hyperparameters minimizing the objective function (Bergstra et al., 2011). The SMBO algorithm have been used in many applications in which the evaluation of the objective function is difficult (Bergstra et al., 2011, p. 2). Difficulty in objective function optimisation arises by the high combinatorial possibilities of the hyperparameter set. To circumvent the high combinatorial possibility problem, the SMBO approach makes use of a stepwise approach. In each sequential step, new hyperparameter settings are created

based on the performance of previous specified hyperparameters. SMBO uses an approximation of the objective function as strategy. The approximation is named 'the surrogate model' and is denoted as $\mathrm{p(x|y)}$. The $\mathrm{p(x|y)}$ formulation of the surrogate model resembles the universal Bayesian formulation given as $\mathrm{p(B|A)}$. One could refer to the surrogate model as follows; 'the probability of x, given y takes a particular value' (Bergstra et al., 2011). The $\mathrm{p(x|y)}$ calculation method used by TPE is given in formula 3.17 below.

$$p(x|y) = \begin{cases} l(x) \ \ if \ y < y^* \\ g(x) \ \ if \ y < y^* \end{cases} \tag{3.17}$$

Concretely, TPE creates two distributions $\mathrm{g(x)}$ an $\mathrm{l(x)}$ from the configuration space. $\mathrm{l(x)}$ is created by forming a distribution based on the point in the configuration space x, where y is lower than y^*. $\mathrm{y(x)}$ is created by forming a distribution based on the point in the configuration space x, where y is higher than y^*. Based on the two distributions, TPE uses a selection function for making the decision about which set of hyperparameter values could best be investigated in the next step. For finding the best set of hyperparameter values, the selection method uses the Expected Improvement (EI) criteria (Bergstra et al., 2011). The EI criteria is visually represented as the blue bottom surface in Figures 3.11 and 3.12. The maximum value of the EI derivation will give the next hyperparameter set that will be tested. After each hyperparameter set check, the EI function will update itself, creating new a new surface. This can be observed by comparing Figure 3.11 with Figure 3.12. The mathematical formulation of the EI formula is given below in Formula 3.18. Note that EI is maximised by maximisation the $\mathrm{g\ (x)/l(x)}$ ratio given in the latter part of Formula 3.18.

$$EI_{y^*}(x) = \frac{\gamma y^* l(x) - l(x) \int_{-\infty}^{y^*} p(y)dy}{\gamma l(x) + (1-\gamma) g(x)} \propto (\gamma + \frac{g(x)}{l(x)}(1-\gamma))^{-1} \tag{3.18}$$

3.7.3 Arrival Time Calculator

The main performance measurement metric in this study is the Nett Promoter Score (NPS). The NPS metric is used for measuring customer satisfaction. The goal and calculation method used by NPS is completely explained in section 1.3. After investigation of historical NPS score data, it turns out that customers assign lower NPS scores if the drivers arrive more than 30 minutes too early or 1 minute or more too late. These arrival time indications are both relative to the 1-hour timeslot that is send to the customer at the same morning of delivery. The NPS historical data analysis is described in later section 4.2 in more detail. In the current situation, the Coolblue delivery employees do not get instructed by arrival time guidelines they should try to follow to improve NPS scores. However, it is very likely that the NPS score can significantly be improved in the following way; assigning extra or less driving time to each customer stop on the vehicle routing. Both arrival time configurations are focussed on preventing much too-late deliveries later in the tour. To improve NPS scores most, relatively to the current situation, in total four arrival time policies are included in the service time prediction tool:

1. Too-early arrivals are accepted as it is in the current situation.
2. Like policy 1, plus additional driving time p assigned to each customer. In this way, the drivers get an extra time of p minutes for each delivery. This results in less too-late deliveries, but on the other hand, causes more too-early deliveries. The arrival time calculator determines the effect on NPS while adding different values of parameter p.

In sections 5.1 and 5.2 the NPS results of the two arrival time policies are described in detail.

3.8 Evaluation and Validation

In this section, the model evaluation methods and model validation methods used in the service time prediction tool are described. In the model evaluation the following question is of central importance; what is the best service time prediction model scenario among the investigated service time model scenarios tested? In the model validation phase has a different question with central importance; Does to model make logical predictions in various input and model execution situations?

3.8.1 Model Evaluation

In this study, two metrics are used to evaluate the model performance. The two metrics are the service time prediction error and the NPS score. In the remainder of this subsection, an explanation is provided how the two metrics are concretely implemented and used in the service time model.

The first and most important performance metric used to evaluate the predictive performance of the model is the service time prediction error. Because NPS score on their own is not able to provide a functional output parameter for machine learning algorithms to optimize, the service time prediction error is used as main performance metric. The service time prediction error is measured as the difference between the service time predictions and the realised service time. In the service time prediction tool, the Mean Absolute Error (MAE) is set as the default error metric. In contrast to the standard Mean Error (ME) the MAE only uses the magnitude of the prediction error and does not make use of the direction of the prediction error (Witten & Frank, 2011, p. 181).

MAE is the default error metric in this study. By setting the MAE metric, as default metric in this study, it means that the performance of other metrics like Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) is not checked. The reason for assigning MAE as default error metric is the following; The MAE relatively treats all prediction error magnitude the same. The favourable consequence of treating all prediction error the same is that all observations will be optimised linearly by the magnitude of the error. This means that there is no exceptional focus on the potentially unrepresentative service time outliers. In this way, the effect of incorrect outliers in the input dataset is not transferred to the prediction creation process. However, it could be that the high service time observations should not be classified as outliers.

The MAE metric is used as main performance metric in the service time calculations. However, sometimes is it interesting to see the real distribution of the error. Then it is recommended to check the mean error (ME). ME gives unadjusted positive and negative error values.

Furthermore, the decision is made not to test relative error measures. Potential relative error metrics that could be tested are the Relative Squared Error (RSE), Root Relative Squared Error (RRSE) and the Relative Absolute Error (RAE or MAPE). Relative error metrics normalise the error measures according to the mean value (Witten & Frank, 2011, p. 181). This means lower weights are placed on values higher than the mean value. These metrics diminishes the effect of outliers, which is undesirable. Last also the coefficient of determination (R^2) and AIC/BIC error metrics are not used in the service time prediction tool. The R^2 error is not used because the total error in non-linear models

does not add up to 1, like in linear models (Colin Cameron & Windmeijer, 1997). In non-linear models, the total error term could not be calculated objectively, since the total error term applicable in linear regression models ($SS_{total} = SS_{regression} + SS_{error}$) does not hold. The reason for not using the AIC or the BIC error metrics is that these metrics are only capable of making a performance comparison between different models. The magnitude of the AIC and BIC value is uninformative. Since AIC and BIC are composed based on the R^2 model fit, AIC and BIC are considered invalid for non-linear models (Hastie, Tibshirani, & Friedman, 2013). The mathematical formulas for the three performance metrics that are used in the service time prediction tool are given below (Witten & Frank, 2011, p. 181):

$$Mean\ Error\ (ME) \qquad \frac{p_1 - a_1 + \ldots + p_n - a_n}{n} \tag{3.19}$$

$$Mean\ Absolute\ Error\ (MAE) \qquad \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{n} \tag{3.20}$$

The second evaluation metric is the NPS score. In section 1.3 a detailed description is provided about how the NPS score is obtained from customer surveys. The NPS score is computed after the MAE service time check. As explained in section 1.3, historical NPS scores turns out to indicate different NPS scores for too-early, on-time and too-late arrivals at the customer.

The new predicted NPS scores is composed by setting new customer arrival time in which the XGBoost service time prediction is inserted as service time. Next, the new arrival times are mapped to the historical NPS scores on 15-minute intervals. In this way, new machine learning model NPS scores are obtained. Those new machine learning model NPS scores are compared with the current realised NPS score by Coolblue. This results in the notion of customer satisfaction increasement or reduction realised by the implementation of the new machine learning model.

In reference to the most important evaluation metrics described in this section (MAE and ME), the following hypothesis is defined:

H1: The MAE and ME of the machine learning computed service times predictions is Expected to be significantly lower than the MAE of the current Coolblue service time predictions.

3.8.2 Model Validation

Most important for any model is that the model accurately represents reality. It is essential to relate the performance of the model against the behaviour of the real system.

First, Numerical model statistics are compared with numerical statistics of the real system. Therefore, the service time predictions are validated. The actual question was; are the new machine learning service time predictions giving reasonable values? To answer this question both visualisation and descriptive statistics will be used.

Besides the service time predictions validation, also the distributions of too-early, on-time and too-late arrivals are analysed. Nevertheless, since the expectation raised in H1 is that the machine learning model more accurately predicts service times, the NPS calculator is likely to provide higher NPS scores. Taking this into consideration the following validation hypotheses are proposed:

H2: The number of too-early arrivals is expected to be higher in the current Coolblue service time model in comparison to the machine learning service time model.

H3: The number of too-late arrivals is expected to be higher in the current Coolblue service time NPS model in comparison to the machine learning service time NPS model.

H4: The new machine learning model NPS scores significantly outperforms the current Coolblue model NPS score.

3.9 Graphical Model Performance Comparison

Besides numerical model evaluation and validation, graphical model evaluation and validation significantly helps in determining the best model. In section 3.8, the numerical model comparison is described. In this section, the graphical model comparison is described. Three categories of visualisations used in this study are described. Visualisations help in discovering the best model from the set of candidate models. By reviewing the visualisations together with the numerical output, relations in the model behaviour can be observed which assist in selectin the best model. In this section, the theoretical approach behind the visualisations is described.

The MAE numeric model comparison is executed for all individual model. Graphics, however, are only computed for the five best machine learning models. The five best machine learning predictors graphically are compared to the current Coolblue service time prediction model in every single visualisation. In total three visualisation categories are defined. The first category of visualisations shows KPI performance. In these visualisations, either the MAE, ME or NPS score is provided on the Y-axis. The second category of visualisations show the error distribution. The third category of visualisation give an inside into the inner structure of the model by pointing out feature importances.

3.9.1 KPI Plots

First and most essential visualisations category is the KPI plots category. The name KPI Plots refer to visualisations that represent the metric performance results obtained by the model. The three KPI's in this study are MAE, ME and NPS. In the KPI plot visualizations one of those three metrics is placed at the Y-axis. The theoretical descriptions of those three KPI's is already provided in sections 3.8.1 and 3.8.2.

3.9.2 SHAP Value Feature Importance Plots

Last visualisation category is the SHAP Value Feature Importance Plots. SHAP values describe the importance of variables included in the model. Machine learning predictors mostly give a very accurate prediction. However, the computation method used by machine learning algorithms is rather complex and mostly seen as a black-box structure. To advance understanding of the complex machine learning black-box structure, an analysis of feature importance is done.

Lundberg notes almost all feature importance calculation methods belong to the set of additive feature attribution methods (Lundberg, Erion, & Lee, 2018). Additive features attribution methods sum the individual observation values, thereby creating a model output score. Feature Additive attribution models however must satisfy three important properties to be valid methods; local accuracy, missingness and consistency. Local accuracy means that the feature importance outcome for all features together always sum to the same standard number (Lundberg et al., 2018). The missingness property states that a missing feature should logically be assigned feature importance of 0 (Lundberg et al., 2018). The consistency property states that, changing the order of the features so

a single feature will have a more substantial impact on the model, will never decrease the feature importance value of this feature (Lundberg et al., 2018). In the article Lundberg et al. (2018), all pre-mentioned model validation properties are checked for the following four feature importance metrics:

- Feature importance based on split
- Feature importance based on gain
- Permutation
- Shaabas.

It results that none of the four-enumerated feature importance metric satisfies to all the three model validation properties local accurate, consistency and ability to handle missing data. Fortunately, Lundberg proposes one new feature importance metric that does satisfy all three properties. The name of this feature is; SHAP (SHapley Additive exPlanations) values. SHAP values assign feature importance to the model based on the Shapley theorem. Because SHAP value is the only feature importance methods satisfying the three pre-mentioned properties the decision is made to use SHAP values in this study to determine feature importance. There even is one more reason to use SHAP values in favour of the pre-mentioned other feature importance methods; SHAP values best recognise model performance changes. The proof is given in Figure 3.9 below. This Figure represents the model prediction improvement if the worst feature is removed from the input dataset. By removing the worst features, the prediction of the model should increase. SHAP values can best recognise the performance increase by observing that the SHAP value line is higher than the other feature importance lines.

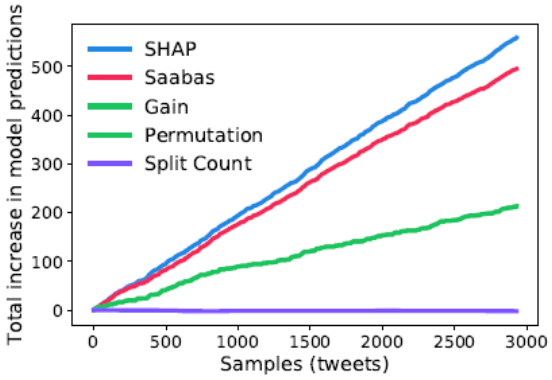

Figure 3.9. Performance Improvement Recognition of SHAP values vs other feature importance methods.

For determining SHAP values for each feature, the SHAP metric uses a two-stage procedure. Sequentially one of the variables is removed from the input dataset. The prediction will be made and an error term of the model results. Next, the feature is retained in the input dataset. Again, the prediction will be made and the error term results. By comparing the error terms of the model in both situations, the feature importance of all variables in the model can be drawn. SHAP values denote the feature importance in log odd values. Log odd values are created by transforming the probabilities to the probabilities of a logistic function. In Formula 3.24 below, the logistic function is presented. Visualising feature importance in terms of log odds values is desirable for one main reason; Log odd values will create equal axis length for both predictions lower than the mean, or predictions higher

than the mean. In ordinary cases, it is tough to compare probabilities. A simple example can be drawn. If you team attends a soccer tournament, based
on the team's quality, they will have a chance of winning a match 1 of the seven times $(1/6)$ = 0.17 Another team is better and has a probability of winning a match 6 of the 7 times; $(6/1)$ = 6. However, the two probabilities cannot be compared by using the 0.17 and 6 outcome probabilities.

Log odds create a logistic transformation to the function. In this way the 0.17 and 6 outcome scores are transformed into:

$$Log\ Odd \quad Low\ Chance\ of\ Winning: \quad log(0.17) = -1.79$$

$$Log\ Odd \quad High\ Chance\ of\ Winning: \quad log(6) = \ 1.79$$

Observe that the log odds correctly translate the probabilities in easily recognisable numbers.

In section 4.5, the three types of SHAP plots used in this study are defined. Later, in section 5.2, the 3 SHAP Plot visualisations are provided and described in detail.

$$f(x) = \frac{1}{1+e^{-x}} \tag{3.21}$$

4. Conceptual Model

In Chapter 3, the methodology used to compute machine learning service time prediction is described. In addition, also the model validation and evaluation metrics and methods used are explained. In this chapter, a conceptual model diagram is proposed for transforming the methodology into real process outcomes. The high-level service time prediction model setup diagram is presented in Figure 4.1 below. In short terms Figure 4.1 describes the system prediction behaviour that a historical dataset is inserted in the model, a subset of 6 possible pre-processing operations are applied to the data and the Bayesian solvers finetunes the hyperparameters by using cross-validation in the train set. Next, the optimal model is determined by the final performance testing optimizer, the MAE of de service time is determined and the NPS scores is estimated.

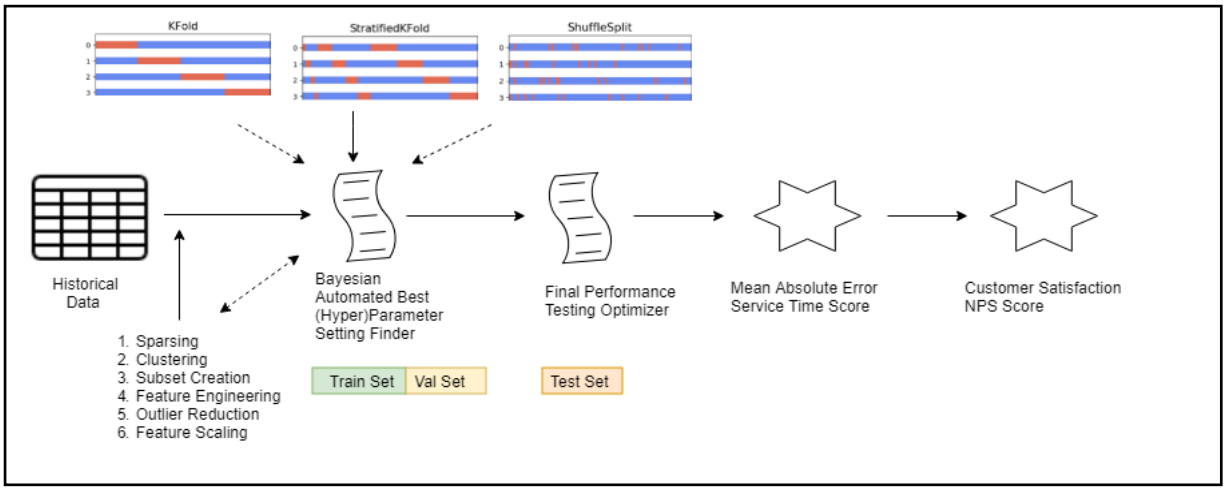

Figure 4.1: Setup Service Time Prediction Model - High Aggregation Level

4.1 Dataset Construction and General Data Pre-processing Operations

In this subsection, the construction of datasets and general data pre-processing operations are described. First, the dataset construction is considered.

To start making machine learning predictions, a dataset must be constructed that includes indicative features. Therefore, as much as possible service-time related features need to be captured. All features that potentially influence service time predictions are selected. The features are selected based on interviewing Coolblue employees that are most closely involved in the delivery operations. The function title of the colleagues that were interviewed; Logistic Planners, Business Analysts, Team Leaders and Drivers. With the help of information received from these colleagues, a UML diagram of the service-time dependent variable features system is created. This UML diagram is presented in Figure 4.2 below. Figure 4.2 provides an overview of the association, aggregation and composition relations between all features used in the service time prediction model.

In total two databases are used for extracting the data, one internal Coolblue database and a third-party logistics service provider database. From the logistics company database, all historical planning and all historical realisation data were extracted. Additionally, the differences between planning and realisation is extracted from this logistics company database. Planning and realisation data is available for all the following indicator segmentations; service time, driving time, arrival time and departure time. The data is uniquely available for each order, and therefore it is possible to merge this data on order id with the SQL Servers Internal Coolblue Database. The SQL Servers Internal Coolblue Database contains both order-related data and product-related data.

In Figure 4.2 below, the UML diagram of the service time model feature relation system is provided. In Figure 4.2 it is indicated that products are aggregations of orders, meaning that an order consists of 1 up to n products. Furthermore, the products are belonging to one specific product group. Product groups are generalisations of product types and sub-product types. After the route planning is carried out, individual orders got assigned a sequence number and a tour number. A tour consists of multiple sequence numbers. Sequence numbers are just indicating a ranking of stops numbers in the tour. The first stop in the tour is assigned sequence number 1, the next stop in the tour is assigned sequence number 2. The sequence number logic is applied similarly to later stops in the tour. One tour is carried out by a driver, a co-driver and in exceptional situations also a second co-driver. The driver, co-driver and second co-driver are driving the van to all delivery locations and are delivering or installing the products at the customer desired location within the house.

Next, the general data pre-processing operations are described. As is mostly the case, the data is not completely clean and free of inconsistencies. General data pre-processing operations can be applied in many ways. In Table B.2 in the appendix, all General Data Preparation operations are given. Unclean variables are indicated for the need to be converted, created, completed, corrected or transformed.

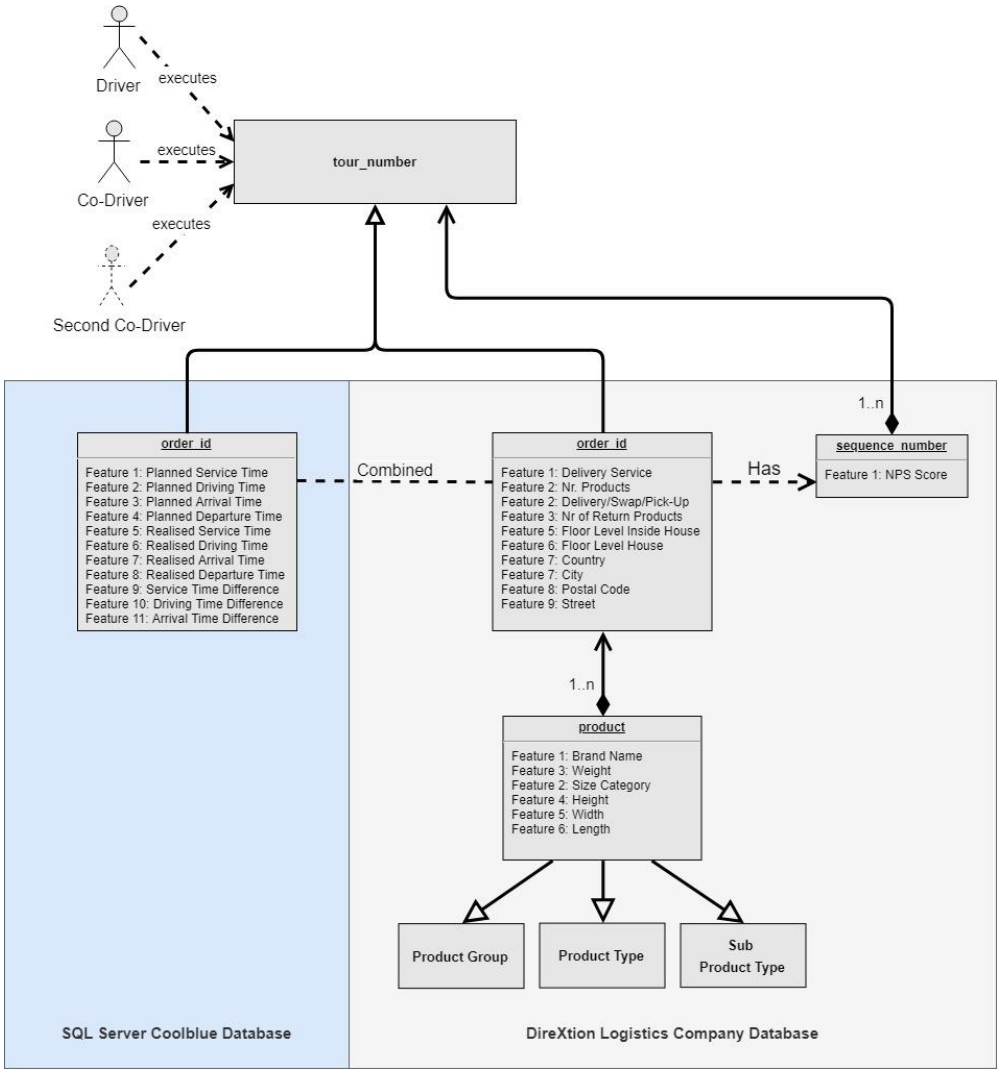

Figure 5.2. UML Diagram Service Time Model Feature Relation System

4.2 Exploratory Data Analysis

Exploratory data analysis (EDA) is of the highest importance before proceeding to model creation and deployment. EDA is so seriously important because it discovers the model requirements and model bounds. Also, the relations and distributions between variables, that are included in the model can be tracked down by doing EDA. In this section, a distinction is made between the model input variable EDA and model objective function EDA. Both numerical and graphical analysis is performed in each subsection 4.2.1 and 4.2.2.

4.2.1 Input variable EDA

Input Variable EDA refers to the analysis of individual features distributions or analysis of relations between features. All features that were initially having an object datatype are transformed into numeric datatype, meaning that for all features, numerical evaluations can be performed. The most important numeric data analysis techniques are descriptive statistics. The first numeric data analysis technique, descriptive statistics, gives most important information for a variable in the dataset. In Table 4.1 below, the descriptive statistics of the target variable 'Stoptime Minutes' is given.

The 'Stoptime Minutes' feature is the target variable in this study. It qualifies the historical realised service times. Initial EDA validations checks can be done by reviewing Table 4.1. In the next part the non-outlier transformed test dataset descriptive statistics will be provided. The test dataset consists of 97357 orders for the non-outlier transformed dataset. The original datasets contain 310595 orders. On average, the service time of the delivery order is 11.69 Minutes. The standard deviation of the service time target variable is 9.77. Moreover, 75% of deliveries are lasting more than 5.78 minutes, meaning that a high percentage of orders (25%) is delivered in a very short time window. This could be due to the inclusion of so-called 'DrempelPlus' products in the historical orders dataset. 'DrempelPlus' products are products that are delivered at the customer desired location inside the house. However, those products are not completely installed. Examples of 'DrempelPlus' products are; Home trainers, Cross-trainers and Ovens. On the other side of the range, 25% of deliveries are lasting more than 15.76 minutes. By further analysing the minimum value and maximum value descriptive statistics of the target variable, there turns out to be a small percentage of wrong service times, the maximum service time duration in the dataset is 381.62 minutes and the minimum service time duration is -1422.86.

The maximum service time value is approximately equal to 11.5 hours. This is unreasonable large and therefore classified as an outlier. The minimum service time duration denoted is a negative value. Negative service time duration cannot happen, that is why all negative service times should be deleted from the input dataset. Because the minimum and maximum service time values are outliers, a decision needs to be made which service times durations are considered as outliers and therefore reduced from the dataset. To define the right service time value bounds for which the service times are outliers, the count distribution of the service times is presented in Table 4.1 below. The service time duration count distribution in Figure 4.3 shows the number of occurrences of service time durations accumulated in 1-minute bins. In Figure 4.3 a sharp drop of in total 49 deliveries lasting 54 minutes to 26 orders lasting 55 minutes is observed. While observing the service time bins higher than 55 minutes, the total amount of orders per one-minute bin even more decreases. The decision was made to delete all orders lasting longer than 54 from the dataset to test the predictive performance of the model if this high service time values were removed. Observe that service times durations smaller than 5 minutes occur most. The service time duration 5 minutes up to and including 12 minutes turn out to appear in almost equal quantities. For each subsequent service time duration 1-minute bins, the number of

Test Set Historical Realised Service Time		
Statistic Type	Normal Coolblue Model	Outlier Red. Coolblue Model
Count	97357	93222
Mean	11.69	11.99
Std	9.77	7.36
Min	-1422.86	1.65
Max	381.62	50.23
25%	5.78	6.37
50%	10.52	10.85
75%	15.76	15.98

Table 4.1. Test Dataset Historical Realized Service Time No Outliers Reduced and Outlier Reduced Configuration

More numerical EDA statistical are combined in Table B.3 in the appendix. Table B.3 gives us the following information; The percentage of deliveries that were ordered by loyal customers is 2.67%. Loyal customers ordered at least twice in the past 7. Furthermore, the percentage of orders delivered by the 2-man Coolblue delivery service is 82.55%. This means that the 2-man delivery service is used much more than the 1-man delivery service and the bike delivery service. Next, Table 6 indicates that half of the orders have a sequence number lower than 10 (62.47%). The meaning of sequence number in this study is described in section 4.1. Last three observations from Table 6 are; 'Depot' orders are filtered out correctly ('Visit' orders are retained), only a small percentage of orders is not delivered successfully (0.65%) and 0.47% of deliveries resulted in damage to the client personal belongings.

Besides numerical input features EDA also Graphical input features EDA is performed to see the objective function distributions in the data quickly. Distribution plots are created for; realised service time, time-related features (orders per month, orders per hour, orders per sequence number), the number of products per order and front door floor level. The realised service time distribution is already discussed in this section. A graphical representation is provided in Figure 4.3 below. Sequence number distribution is analysed earlier in this section as well. The distribution plot can be found in Figure 4.4 below. Figure 4.5 gives us the histogram of delivery moment over the day in bins of hours. 64.47% of all orders are delivered in the morning, where peak moment has a timespan from 8.30 to 11.30. Figure 4.6 gives the number of orders per month. A strong trend in the number of orders per month turns out to be present. Except for small dips in February and April, the total amount of orders delivered per month is sharply increasing, especially in the months May and June. Figure 4.7 shows that most front doors of delivery addresses is located at ground level. Only 19.6% of delivery addresses are located on higher floor levels. Last, Figure 4.8 shows the number of products per order. 73.19% of orders consist of just 1 product, 16.13% of orders consist of 2 products and 6.58% of orders consist of 3 products. Only a small amount of orders (4.32%) includes more than 3 products.

Last, to get an indication of the relations in the data and get an understanding of feature interaction a correlation heatmap is created. The heatmap is provided in Figure B.1 in the appendix.

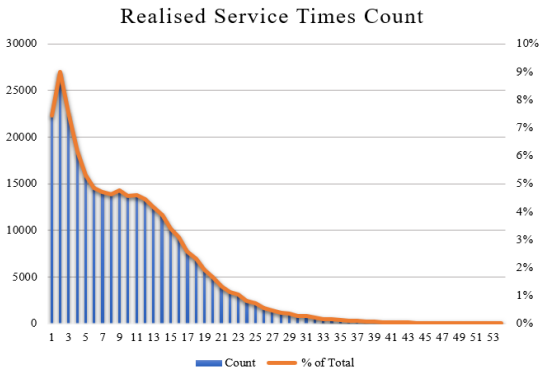

Figure 4.3. Realised Service Time Count Distribution Plot

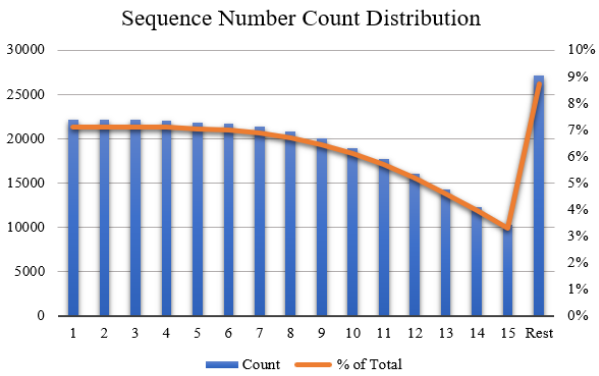

Figure 4.4. Sequence Number Count Distribution Plot

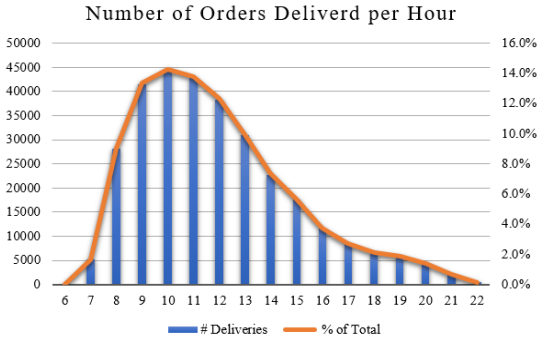

Figure 4.5. Number of Orders Delivered per Hour

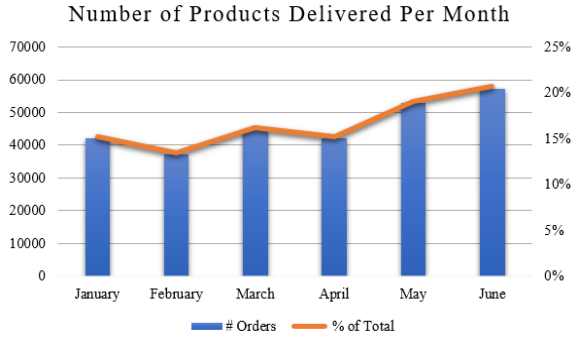

Figure 4.6. Number of Orders Delivered per Month

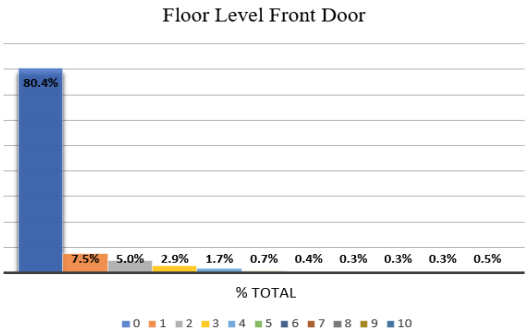

Figure 4.7. Floor Level Front Door Distribution Plot

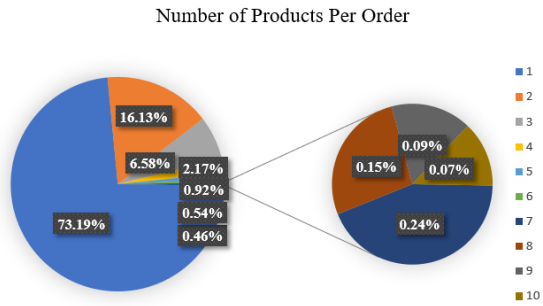

Figure 4.8. Number of Product per Order

4.5 Review Numeric and Graphical Output and Best Model Selection

The machine learning prediction tool is designed to test a wide range of machine learning model scenarios. Each machine learning model scenarios is a variation to other machine learning model scenarios by the uniqueness of one of the following dimensions; input variables inserted, pre-processing method applied, the algorithm used, or arrival time calculation method employed. Besides data pre-processing and prediction operations, the machine learning prediction model has a numeric statistics generation option and several graphical representation options. Therefore, the tool is both suitable for prediction and reviewing purposes. In this section, the most critical numerical output statistics and graphical output statistics are described.

Numerical output statistics of the two most important KPIs for the prediction models are generated. The output statistics of the two most important KPIs help in selecting the best model. The two most important KPIs are; Mean Absolute Error Individual Customer Stop (MAE) and the NPS score. All the models that are tested are evaluated on MAE. Based on MAE, the three best models are examined in more detail. The three best models, giving the lowest MAE scores, are subsequently compared on NPS score as well. Moreover, a deep dive is performed to this set of 5 best models to improve the predictive performance even more by making changes to input variables, (hyper)parameters, or pre-processing operations. Scenarios resulting from adjustments to the three best models are again tested on MAE and NPS score performance. The arrival time calculation method stays the same because otherwise, the interpolation of NPS score will come in play. In the result section 5.1, the numerical output of the three best models is provided.

Graphical output can be considered as an extension to numerical output as well as a friendly representation to light results. A good visualisation often is preferred over tabular information, just because it is attractive. In total four different result visualisations, types are created. In section 3.9, the categories of visualisations are described. Each category includes multiple plots. The first category of visualisations shows KPI performance. In these visualisations, the MAE, ME and NPS score is provided. The second category of visualisations shows the error distribution. The third category of visualisation is giving an inside into the inner structure of the model by plotting feature importance. The three visualisation categories are described in detail in the next subsections.

4.5.1 KPI Plots

The first category of visualisations shows KPI performance. In these visualisations MAE, ME and NPS score plots are provided. In total four plots are created;

The first output visualisation gives the distribution of the MAE in a histogram. The second output visualisation gives a violin plot histogram of both the model MAE and ME. Furthermore, in the scenario analysis provided in section 5.1, the various model scenario MAE and confidence interval range as provided in a line plot visualization. Last, also an NPS score barchart visualization is provided. This barchart is presented in section 5.4.

4.5.2 Error Joint Plot

In this section, the joint plot visualisations are described. A joint plot is a multifunctional plot; it combines two visual representations; the error scatterplot and the regression prediction line. On the Y-axis, the realised service times are located. On the X-axis, the planned service times are located. The error scatterplot is mostly referred to as an essential plot for validating the model. In the ideal situation, the regression scatterplot line indicates that the scatters are approximately normal

distributed. The normality is measured as distance to the linear regression line. In Figure 4.9, four different regression error joint plots are provided. The linear regression plot is provided in the upper left Figure. The linear regression line ideally is approximating the diagonal line that increases with similar magnitude for the realised service times and the planned service times. If this is not the case, it means the model is overpredicting or underpredicting the model slightly. In Figure 4.9 the upper left plot regression line can be compared with the diagonal line in the upper right scatterplot. The scatters scatterplot in both Figures are the same, same scales are used. In the down left scatterplot, both the X-axis and the Y-axis range is doubled in comparison with the upper two plots in Figure 4.9. This gives a better indication of the distribution of the 'Planned Service Time' and the 'Realised Service Time' variables. Some outliers are still not reflected in the down left plot. Last, the downright plot is a special type of scatterplot, referred to as the hexian scatterplot. This scatterplot type reflects density regions by usage of different colours. Especially in very dense scatterplot regions, the hexian scatterplot is useful. From hexian scatterplot the observation can be made that the planned service time in general is little overestimating the realised service time. Later, in section 5.3 it is pointed out that the planned service time is indeed overestimating the realised service time on average. In this case, it is essential to check the ME distribution plot, which will be created as well as is indicated in section 4.5.1.

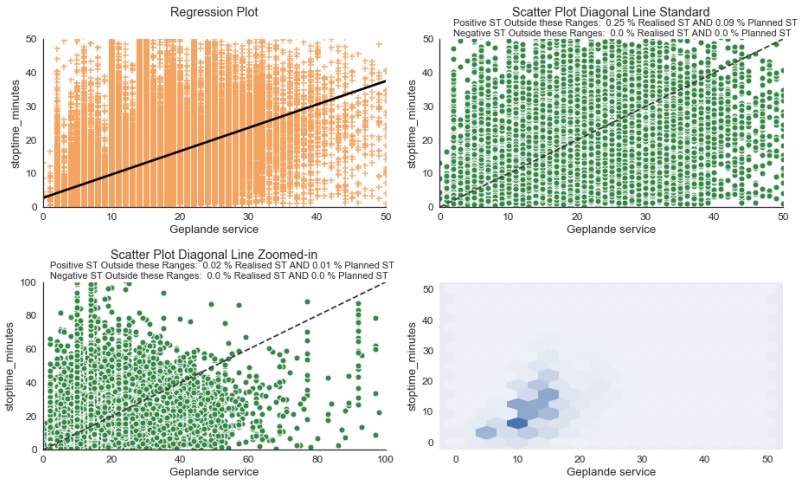

Figure 4.9. 4 Configurations of Error Joint Plots

4.5.3. SHAP Value Feature Importance Plots

Getting an approximation of the internal functioning of the machine learning predictor and at the same times discovering feature importances, SHAP value visualisations are provided. In section 3.9 the explanation is provided why SHAP values are favoured over other variable importance plots to determine feature importance. It is also described that SHAP values make use of log odd scales to define the importance of the variables in the model. Furthermore, an example is provided how log odds are calculated and why log odds are visually attractive. In total three different SHAP value plots are used in this study to indicate feature importance;

1) SHAP Vertical Summary Barcharts
2) SHAP Vertical Summary Coloured ScatterPlots
3) SHAP Distribution Dependence Plot

The SHAP Vertical Summary Barchart ranks the features vertically based on importance. The most important feature is placed on top, whereas the least important features place on the bottom. Getting an understanding of the importance of features in the model is useful for two reasons. Analysis of feature importance helps in understanding the model better. Second, it is a perfect indicator for determining for selecting the features to drop. Third, it gives an understanding of potentially interesting features that could be added by reviewing the type of features that are ranked at the top.

The SHAP Vertical Summary Coloured ScatterPlots ranks the features in the same way as the SHAP Vertical Summary Barchart. However, this plot is more advanced. With the help of coloured dots, it indicates the SHAP value distribution of individual SHAP values over high or low values of that features. It thereby leverages individualised feature attributions in comparison to standard feature importance bars.

SHAP Distributional Dependence Scatterplot gives the interaction effect of two variable in the model by remaining the initial target variable scores. Like SHAP Vertical Summary Coloured ScatterPlots, SHAP Distributional Dependence Plots make use of coloured dots to represent the interaction effect. This third kind of SHAP plot is very informative since it can capture the interaction effect by both vertically and horizontally.

In section 5.2 the all the plots described in this section are given and described in detail.

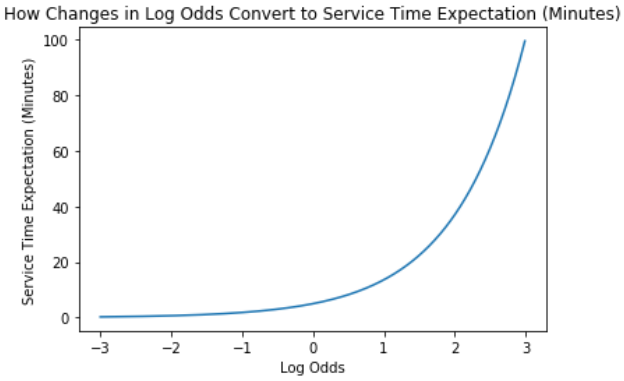

Figure 4.10. Log Odds and Service Time Relation Curve

4.2.2 Objective Function EDA

Objective function numerical EDA reviews the objective that needs to be improved in the model. In section 3.8.1 it is mentioned that two metrics are assigned as main KPIs in this study; service time prediction error and NPS score. Because the service time prediction error is analysed in the results sections 5.1 and 5.2, only the NPS metrics are analysed graphically and numerically in this section. In section 1.3, an explanation is given that in simple terms, the time performance can be divided in too-early arrival, on-time arrival and too-late arrival. By reviewing the historical NPS scores of the on-time performance distribution in Table 4.1 and Figures 4.11 and 4.12 below, it can be observed that there is a positive relation between NPS an on-time performance. The NPS on average are higher the delivery employees arrive on time, relative to arriving too-late or too-early.

Especially, arriving too-late at the customer is harmful in terms of NPS as can be seen in Table 4.1 and Figures 4.11 and 4.12 below. The historical NPS scores provided in Table 4.1 are scores inserted as input parameter in the NPS calculation model. First, by analysing Figure 4.11 and 4.12 and Table 4.1 carefully, it is decided to state one single deterministic NPS score for the time interval range 0 minutes lateness up to 1-hour lateness. This on-time arrival range is indicated in Figures 4.11 and 4.12 as the area between the black vertical line. 60 minutes lateness refers to the end of the one-hour time slot that is indicated to the customer at the beginning of the day. 0 minutes lateness refers to arriving at the strict beginning of the time interval indicated to the customer the same morning of delivery execution. The average NPS score of the 0 minutes lateness to 60 minutes lateness time interval, is set to the deterministic (mean) value of this time interval, which is 74.46. The NPS scores outside on-time arrival range are clustered in 15-minute time-intervals. This is done to regularize the model outcomes by protecting for outlying NPS observations in the 15-minute time window. The number of NPS scores obtained in the single minute interval is not large enough for creating a grounded NPS difference distinction. Most of the time, single minute time interval average NPS scores are based on less than 500 NPS surveys. The number of 500 NPS survey is evaluated not to be representative in dialogue with Coolblue management.

	# NPS Surveys	Time Interval	2017 + 2018 NPS Average
Too-Early	307	-Unlimited < Arrival Time < 1h15min	74.33
	249	-1h30min < Arrival Time < 1h15min	74.33
	635	-1h15min < Arrival Time < 1h	71.34
	1119	-1h < Arrival Time < 45min	71.13
	1924	-45min < Arrival Time < 30min	73.08
	3530	-30min < Arrival Time < 15min	74.76
	6233	-15min < Arrival Time < 0min	74.75
On-Time	30458	-0min < Arrival Time < 1h	74.37
Too-Late	1974	1h < Arrival Time < 1h15min	73.81
	1164	1h15min < Arrival Time < 1h30min	73.45
	598	1h30min < Arrival Time < 1h45min	70.07
	363	1h45min < Arrival Time < 2h	71.90
	224	2h < Arrival Time < 2h15min	57.14
	332	2h15min < Arrival Time < Unlimited	53.90

Table 4.1 2017 and 2018 NPS Aggregation Data.

Figure 4.11. Historical NPS Score Distribution 2017

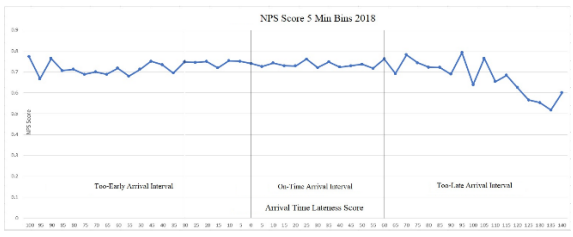

Figure 4.12. Historical NPS Score Distribution 2018

4.3 Model Specific Pre-processing, Algorithm Deployment and Service Time Evaluation

This section consists of three parts, which are; model-specific data preparation, algorithm deployment and service time prediction error evaluation.

First, model-specific data pre-processing operations are performed. Different from the general data pre-processing operations discussed in section 4.2, model specific data pre-processing operations are adjustment made to:

1. Variable Selection
2. Data Partitioning
3. Scaling Operations
4. Handling Outliers

In section 3.6 a thorough explanation of all 4-enumerated model specific data pre-processing operations is given. In this section model specific data pre-processing operations that are applied in the service time prediction model is described.

Next, the prediction algorithm is selected, deployed and evaluated. Model selection, deployment and evaluation are performed many times before an optimal solution is found. There are two reasons causing the high amount of cycle repetition needed before convergence of an optimal solution. The first reason is that some boosting algorithms, like XGBoost, aggregate predictions of multiple individual models for creating the prediction. The boosting methodology used by XGBoost is explained 3.7.1.1. The second reason is that hyperparameters are selected by a sequential model-based global optimisation (SMBO) approach. In subsequent iterations of model deployment, the hyperparameters are improved. Because a sequential iteration strategy is used to find optimal hyperparameters, the model must be deployed several times in a row.

Next, the prediction error is obtained by usage of an evaluation metric. Section 3.8.1 describes the prediction error metric that is tested in the service time prediction tool. Mean absolute error (MAE) is used as the default prediction error metric. MAE measures the prediction error for the complete set of orders in the test set. Besides calculation of the MAE of individual service times, also the mean error (ME) of all stops is determined.

4.4 Arrival Time Module and the NPS Calculation Module

This section is devoted to two process steps arrival time method constructor and the NPS calculation module. In the arrival time method constructor, two possible arrival time calculation methods are initiated. Both arrival time calculation methods come with different restrictions and solution spaces for computing expected arrival times.

4.4.1 4 Arrival Time Scenarios

The first arrival time calculation method reflects the current delivery procedure. The drivers depart from the depot and arrive at the delivery locations belonging to the tour sequentially. No restrictions are placed on too-early arrivals, meaning that too-early arrivals are accepted. In sections 4.2, the meaning of too-early arrivals is explained.

The second arrival time method assigns an additional constant driving time value to the model. By using a parameter p, driving time is added to each driving time duration between customer i and customer i+1, where i is the sequence number in the tour.

For describing the relation between the parameter p and the NPS a grid search Table is designed. The grid search Table is explained in section 5.1, and a visualisation of the values in the grid search Table is given in section 5.1 as well.

4.4.2 Arrival Time Lateness Module

The arrival time lateness module determines the arrival time lateness of all individual stops in the tour. The arrival time lateness is computed by simulation. The formulas used in this calculation are given below. In the remainder of this section, the arrival time calculation module is explained by referring to the mathematical formulas used.

The arrival time calculation consists of three components, which are; 'realised arrival times' ($RAT_{o,i,t}$), 'predicted arrival times' ($PAT_{o,i,t}$) and 'arrival time lateness' ($ATL_{o,i,t}$). The o, i, t subscript refers to the order, sequence number and tour number that is attached to a specific delivery order. It is important to add this extra information, because $ATL_{o,i,t}$ of orders consisting to the same tour are dependent on each other. In Formula 4.3 and Formula 4.4 it can be observed that the arrival time lateness is obtained by subtracting the predicted arrival times from the realised arrival times. The Coolblue model predicted arrival time does not have to be calculated since this value already exists in the company's database. For clarification reasons, the calculation method used by Coolblue is given in Formula 4.1. The Coolblue predicted arrival time $CPAT_{o,i,t}$ results from adding up predicted driving time ($PDT_{o,i,t}$) and the predicted service time ($PST_{o,i,t}$) to the current customer to the predicted arrival time calculation at the previous customer. However, the Machine learning predicted arrival time needs to be calculated again, since this predicted arrival time includes the new service time estimations. Unfortunately, in the company database the predicted arriving time turns out to provide incorrect values. Therefore, the calculation method for composing the machine learning predicted arrival time ($MLPAT_{o,i,t}$) is subtracted from the Coolblue predicted arrival time ($CPAT_{o,i,t}$), by creation of a machine learning service time lateness variable ($MLSTL_{o,i,t}$). The service time lateness ($STL_{o,i,t}$) provides the difference between the Coolblue predicted service time ($CPST_{o,i,t}$) and the machine learning predicted service time ($MLSTL_{o,i,t}$). Since the driving time of both the Coolblue model and the machine learning model is equal, the only difference relies in the service time. By adding up the machine learning service time lateness ($MLPAT_{o,i,t}$) to the Coolblue model predicted arrival time ($CPAT_{o,i,t}$), the new machine learning predicted arrival time is created. This mathematical operation is provided in Formula 4.2.

Next, the arrival time lateness can be determined for both the Coolblue model and the machine learning model. The arrival time lateness scores are provided in Formula 4.1 and Formula 4.2. For the Coolblue Model, the arrival time lateness ($CPATL_{o,i,t}$) of an individual order is composed by subtracting the individual order Coolblue planned arrival time ($CPAT_i$) from the individual order realised arrival time of an individual order ($RAT_{o,i,t}$). The same methodology holds for composition of the machine learning model arrival time lateness ($MLPATL_{o,i,t}$). The arrival time lateness ($MPATL_{o,i,t}$) of an individual order is composed by subtracting the individual order machine learning model planned arrival time ($MPAT_i$) from the individual order realised arrival time of an individual order ($RAT_{o,i}$).

Predicted Arrival Times

$$\textit{Coolblue Predicted Arrival Time } (CPAT_{o,i,t}) = CPAT_1 + \sum_{i=1}^{n-1} CPST_{o,i,t} + \sum_{i=2}^{n} CPDT_{o,i,t} \; \forall o \in F, \quad (4.1)$$

$$\textit{Machine Learning Predicted Arrival Time } (MLPAT_{o,i,t}) = CPAT_{o,i,t} + \sum_{i=1}^{n-1} MLSTL_{o,i,t} \; \forall o \in F, \quad (4.2)$$

Arrival Time Lateness

$$\textit{Coolblue Planned Arrival Time Lateness } (CPATL_{o,i,t}) = RAT_{o,i,t} - CPAT_{o,i,t} \quad \forall o \in F, \quad (4.3)$$

$$\textit{Machine Learn. Pred. Arr. Time Lateness } (MLPATL_{o,i,t}) = RAT_{o,i,t} - MLPAT_{o,i,t} \quad \forall o \in F, \quad (4.4)$$

where,

PAT_1 = *Predicted Arrival Time* 1*st customer in the tour* (*sequence number* 1)
PST_i = *Predicted Service Time at all customer stops*
PDT_i = *Predicted Driving Time at all customer stops i*
RAT_i = *Realised Arrival Time at all customer stops i*
RST_i = *Realised Service Time at all customer stops i*
RDT_i = *Realised Driving Time at all customer stops i*
STL_i = *Service Time Lateness at stops i*
T = *Total order_id's in the test dataset*
n = *Current sequence number that is analysed*
i = *Sequence number on tour*
t = *Tour number*
F = *Set of orders in the test dataset*

4.4.3 NPS Calculation Module

After the arrival time lateness module has determined the arrival time lateness, the NPS calculation module is activated. Formulas 4.5 and 4.6 provided below are used to compose new NPS scores. The NPS calculation module uses the arrival time lateness scores of each order as input. It does so by assigning the arrival time lateness ($CPATL_o$ *and* $MLPATL_o$) of each order an NPS score. The NPS score is based on the historical NPS score for lateness intervals and therefore indicated as NPS_l. The historical NPS scores for specific lateness intervals is provided in Table 4.1 in section 4.2.

In Formula 4.5 the Coolblue Model Predicted NPS score ($CBNPS$) is calculated. The $CBNPS$ is computed by summing up the NPS lateness interval scores (NPS_l) of all test dataset orders and subsequently dividing this value by the total amount of dataset orders (T).

Formula 4.6 applies the exact same calculation method for determining the machine learning model predicted NPS score. The $MLNPS$ is computed by summing up the NPS lateness interval scores (NPS_l) of all test dataset orders and subsequently dividing this value by the total amount of dataset orders (T).

Estimated NPS Score

$$\textit{Coolblue Model Predicted NPS Score } (CBNPS) = \frac{\sum_{o}^{T}(NPS_l \mid CPATL_o)}{T} \quad \forall o \in F, \quad (4.5)$$

$$\textit{Machine Learning Model Pred. NPS Score } (MLNPS) = \frac{\sum_{o}^{T}(NPS_l \mid MLPATL_o)}{T} \quad \forall o \in F, \quad (4.6)$$

where,

NPS_l = *Lateness Interval NPS Score*
T = *Latest order_id's in the test dataset*
l = *lateness interval*
o = *order_id*
F = *Set of orders in the test dataset*

5. Findings and Results

In this Chapter the service time and NPS score results are discussed for the machine learning prediction model and the current Coolblue prediction model. As explained in the introduction of chapter 4 and indicated graphically in Figure 4.1, a wide variety of model scenarios are tested on performance. Model scenarios in the service time prediction tool are considered unique if the two models are inconsistent about any of the following model attributes; input dataset, algorithm, pre-processing method, parameter values. The determination of the model scenario happens in the before model training and in Figure 4.1 by the two gear wheel symbols. In total 50 model scenarios are composed. Two configurations of best models are proposed. Resulting from this scenario test, conclusion can be drawn about the current found best performing machine learning models, satisfying consistent application of the data and data leakage between train and test set is protected for. In the remainder of this chapter first a scenario analysis visualization is provided in chapter 5.1. In chapter 5.2 a scenario sensitivity analysis is provided. In chapter 5.3 the two best models are analysed on MAE and ME of service time by comparison to the current Coolblue prediction model service time performance and a simpler Linear Regression prediction model service time performance. In chapter 5.4 the NPS score of the two proposed machine learning models are compared to the current Coolblue NPS score. Last, in chapter 5.5 the accuracy of the proposed XGBoost prediction model solution is analysed by usage of several plots. These plots uncover the internal calculation methods used by the 'black-box algorithm' XGBoost, thereby enhancing system user understanding and assisting in performance reporting before and after implementation of the system.

5.1. MAE Input Dataset Scenario Analysis

In this section the MAE performance of a subset of all tested scenarios is presented. In the introduction section of this chapter it is described that model scenarios are created by uniqueness of the following model attributes; input dataset, algorithm, pre-processing method, parameter values. The input dataset turns out to be most impacting the service time predictions. Therefore, Figure 5.1. visualizes the performance of model scenarios for which the input dataset is unequal.

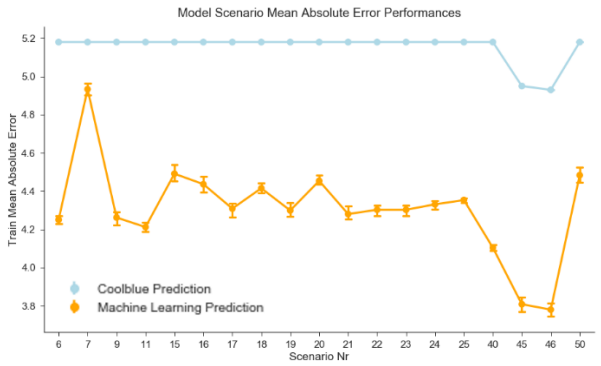

Figure 5.1. MAE Model Scenario Analysis Line Plot

Scenario 6: Product4 + Extra Var Set 1 + Mean Pred Error > 30 + All Summed 2
Scenario 7: No Product4 + Extra Var Set 1
Scenario 9: Product4 + Extra Var Set 1
Scenario 11: Product4 + Extra Var Set 1 + Planned Service Times
Scenario 15: ProductGroup + Extra Var Set 1
Scenario 16: ProductType + Extra Var Set 1
Scenario 17: SubProductType + Extra Var Set 1
Scenario 18: ProductType_+_Brand + Extra Var Set 1
Scenario 19: SubProductType_+_Brand + Extra Var Set 1
Scenario 20: ProductID + Extra Var Set 1
Scenario 21: Product4 + Extra Var Set 1
Scenario 22: SubProductType + Del. Service ID + Extra Var Set 1
Scenario 23: ProductType + Del. Service ID + Extra Var Set 1
Scenario 24: SubProductType NormalDelivery + Extra Var Set 1
Scenario 25: ProductType NormalDelivery + Extra Var Set 1
Scenario 40: Scenario 11 + All Product Aggregations + Mean Service Time + Counter Values + Extra Var Set 3
Scenario 45: Scenario 40 + Outlier Reduction lowest 4% and highest 0.3%
Scenario 46: Scenario 40 + Outlier Reduction lowest 8% and highest 0.3%
Scenario 50: Linear Regression Prediction + Product4 + Extra Var Set 1 + Planned Service Times

Table 5.1. Model Scenario Description

In Figure 5.1 both the machine learning prediction model MAE scenario performances (orange) and the current Coolblue prediction model scenario performances (light blue) are provided. In all the model scenarios, the orange line is placed lower than the blue line, meaning the machine learning model is outperforming the current Coolblue prediction model. In case no outlier reduction is performed to the input dataset, the current Coolblue prediction model reaches a service time prediction MAE of 5.18 minutes. To clarify the different model scenarios that are analysed in Figure

5.1 a scenario description table is provided in Table 5.1. The scenario numbering is not linearly increasing since more scenarios checks are performance in this study than the input dataset adjustment checks given in Figure 5.1.

Input Dataset Scenario 6

The first scenario of Figure 5.1 is scenario 6. This scenario from initial point of view is considered as potential best scenario. As given in Table 5.1, scenario 6 is composed from the original dataset of 26 features. The product is provided in scenario 6 input dataset by Sub Product name, enriched with the delivery service indication, the size category and the delivery type. The input dataset is enriched in this model scenario with adding the extra variables set 1. Extra variable set one consist thee feature engineered variables; 'Real Quantity Including Combi Boxes', 'Visit Counter' and 'Visit Counter Mean Service Time Reduction'. The first feature is a slight adjustment of the normal 'Quantity' feature, for which the real quantity is adjusted because some small products are delivered in a combination box. The second feature counts the number of times any Coolblue order has been delivered to the customer in the past. This feature is added since it may be possible that delivery employees need shorter service time to deliver the products at a well-known customer. The third created variable gives the average service time reduction for unique visit counters, a value that is obtained from historical data. The input dataset of scenario 6 is extended with the mean service time and mean prediction error that belong to the product that is delivered to the customer. Each product has its own service time and prediction error. Last two complex variables 'All Summed Mean Prediction Error 2' and 'All Summed Mean Service Time 2' are added. These variables accumulate the mean service times and mean prediction error for all features. Scenario 6 reaches a relatively good performance with a service time prediction MAE of 4.22 and a low confidence interval range of 0.007. In total 3 different train/test split are used for composing this model.

Input Dataset Scenario 7

Scenario 7 is a rather simple input dataset configuration. The 26 variables in the standard input dataset is enriched with the three variables of the extra variable set 1. However, to check the importance of the product information for creating an accurate prediction, the product information is removed from the input dataset. Due to removal of the product information from the input dataset, a poor service time prediction MAE of 4.95 with a confidence interval range of 0.122 is obtained. In total 11 train/test split configuration are created to test this model.

Input Dataset Scenario 9

Scenario 9 is a simple baseline scenario, because the input dataset is equal to the standard 26 variables plus the three variables composing the extra variable set 1. Scenario 9 provides a good option to compare this simple model performance with scenario 6, in which more service time and prediction error variables are added. The MAE of scenario 9 turns out to be 4.23 with a confidence interval range of 0.026. In total 3 train/test split configuration are created to test this model. Reviewing the MAE of scenario 7 and 9, it turns out scenario 7 is not significantly achieving better service time predictions although extra service time and prediction error variables are added to this scenario.

Input Dataset Scenario 11

Scenario 11 is equal to scenario 9, except from one extra added variable; 'Planned Service Time'. Currently, Coolblue composes a planned service time for all delivery orders. This planned service time in fact is the same as the Coolblue service time prediction often referred to in this study. By providing the planned service time as an input variable to the model, it gives the prediction model an extra guidance to the correct service time approximation. Scenario 11 reaches a service time prediction of

4.202 minutes with a confidence interval range of 0.013. In total 15 train/test split configuration are created to test this model. Although scenario 11, including the extra variable 'Planned Service Time' performs better in MAE terms than Scenarios 6 and 9, it does not perform better in the 95% confidence interval.

Input Dataset Scenario 15 – Scenario 25

The next 11 model scenarios, scenario 15 up to and including scenario 25, are various in the product aggregation level used. The input dataset consists of the 26 standard features including the 3 features in the extra variable set 1. Products can be indicated in the input dataset with different aggregation levels. Ascending from low aggregation level to high aggregation level it is possible to indicate products by; Product Group, Product Type, Sub Product Type and Product ID. Product Group for example state a washing machine for example as 'White Good', whereas Product ID states a washing machine as 'Miele YS 1000'. Scenarios 15 and 20 indicate that both the Product Group aggregation level and the Product ID aggregation level turn out to provide relatively low service time prediction performances of 4.50 minutes and 4.48 minutes respectively. The Sub Product Type and Product Type product aggregation level variables give MAE performances of 4.42 and 4.30. The Sub Product Type product variable aggregation level seems to be the best aggregation level, followed by the Product Type product variable aggregation level. Because, the product variable turns out to be very important in the service time prediction, more aggregation levels are composed. New product aggregation level features are created by combining each of the variables 'Brand', 'Delivery Type' and 'Delivery Service' with either the existing Sub Product Type variable or the Product Type variable. Resulting scenarios 18, 19, 22, 23, 24 and 25 it can be concluded that none of the extra variables 'Brand, 'Delivery Type' and 'Delivery Service' results in a significantly better service time prediction in the confidence interval. Brand is the only variable improving the service time predictions, however this improvement in not significant in the 95% confidence interval. The product aggregation level is obtained by the variable that combines the Sub Product Type, the Delivery Service, the Service Type and the Size category of the product. This product input variables are tested in scenario 21. Although scenario 21 gives best performance, the Sub Product Type and Sub Product Type + Brand product aggregation level gives similar performance in the 95% confidence range.

Input Dataset Scenario 40

Scenario 40 is the best performing model scenario for which no outlier service time outlier reduction is performed. Scenario 40 reaches a MAE of 4.09 on average with confidence interval range of 0.007. In total 16 train/test split configuration are created to test this model. The input dataset used in scenario 40 consists of the standard 26 features, the 3 extra features in extra variable set 1 and 52 extra features of extra feature set 3. In extra variables in feature set 3 are first a concatenation of all product aggregation level. Whereas in model scenarios 15 up to and including 25 one single product aggregation level is inserted in the input dataset, all product aggregation levels are inserted in scenario 40. Furthermore for 13 features an extra count variable is added to the input dataset. The extra count variables give the machine learning model extra help. It may be possible that product, drivers or even specific floor levels are turned on in the dataset. In this way the machine learning model can adopt this information. The count variable however is a bit tricky by usage of products, drivers or even floor levels that do not occur much. For example, if a product is only ordered ones in the complete dataset, then the model could use this information to define the service time prediction. Although this behaviour is not expected to occur much, it may occur in a small amount of predictions. Besides all product aggregation levels and the count variable also the planned service time variable, the service time average and the prediction error average is added.

Input Dataset Scenario 45

In scenario 45 the best performing model, that is obtained in scenario 40 is used again. However, in scenario 45 the service time outliers are removed from the dataset. All 4% lowest variables and 0.3% highest variables are deleted from the input dataset. The reason for deleting the 4% lowest variables is that Coolblue defines the realized service times that are lower than 2 minutes as incorrect loggings. By deleting 4% of lowest variables from the data, all orders up to 1.65 minutes service times are deleted from the input dataset. By deleting 0.3% of highest variables from the dataset, all orders taking longer than 50 minutes are deleted from the dataset. Orders with a service time longer than 50 minutes occur infrequently, however this could negatively impact the internal structure of the machine learning method. Scenario 45 reaches a MAE prediction of 3.81 minutes with confidence interval range of 0.015 minutes. In total 4 train/test split configuration are created to test this model. Because outliers are removed from the dataset, also the Coolblue prediction improves from a MAE 5.18 minutes to a MAE 4.95 minutes. The Coolblue prediction improves with 0.23 minutes and the machine learning prediction improves with 0.28. This means that removing the 4% lowest service time variables and 0.3% highest variables does significantly improve the service time prediction relative to the current Coolblue prediction.

Input Dataset Scenario 46

Scenario 46 is like the scenario 45 except that not 4% of lowest service times are deleted from the dataset, but the lowest 8% of service times are deleted from the dataset. This means that all service times order with a duration less than 2.53 minutes are deleted from the input dataset. The machine learning prediction achieves a MAE of 3.78 minutes with a confidence interval range of 0.013 minutes. In total 4 train/test split configuration are created to test this model. The representative Coolblue prediction gives a MAE of 4.93. Considering the performance improvement of the machine learning model of 0.31 and the Coolblue prediction performance improvement of 0.25, this means that service time outlier reduction of 8% lowest and 0.3% highest variables significantly improves the machine learning prediction relative to the current Coolblue prediction.

Input Dataset Scenario 50

The last scenario in Figure 5.1 is scenario 50. To clarify the power of this algorithm, the MAE obtained by XGBoost is ones compared with the MAE obtained by usage of a linear regression model. Scenario 40 indicates that XGBoost makes service time predictions with a MAE of 4.09 for the best observed dataset. In scenario 50 the performance of the linear regression model on this exact same input dataset is provided. It turns out the linear regression model reaches a MAE of 4.48, significantly lower than the XGBoost prediction, but still better than the current Coolblue prediction model.

Since all input variable scenario are described in the beginning of this subsection, it is interesting to see the distribution of the predictions that are created with the different scenarios. The most important scenarios 9, 40 and 45 service time realization and service time prediction histograms are provided in Figures 5.2, 5.4 and 5.5 respectively. To check the difference in performance of a worse prediction model, in Figure 5.3 the service time realization of scenario 15 is added. This scenario gives the service time prediction of the product ID level aggregation level product variable. The blue lines in Figure 5.2 up to and including Figure 5.5 gives the current Coolblue prediction. Observe that Coolblue rarely make service time predictions lower than 4 minutes. The two peaks in the blue Figure indicate that service times of 10 minutes and service times of 14 minutes are often predicted by Coolblue. The Green line in Figure 5.2 up to and including Figure 5.5 indicate the realized service times. The realized service times are well separated over the interval 1 minutes up to 20 minutes. Realized Service times longer than 20 minutes do not occur on regular bases. The red lines in Figure 5.2 up to

and including Figure 5.5 give the machine learning model performance. The red line is more closely located to the green line, thereby indicating that the service time predictions created by the machine learning model is more accurate than the current Coolblue prediction. Furthermore, it is interesting to see the distribution of service time predictions for the various scenarios. Scenario 15, given in Figure 5.3, is less closely located to the green realized service time line than the other scenarios. This is as expected, because the MAE of scenario 15 is significantly higher than the MAE of the three other scenarios visualized here. The curve of the red lines is similar over the scenarios, meaning the XGBoost creates relatively consistent predictions for different scenarios. Furthermore, observe that the XGBoost prediction red line of scenario 45 is more closely located to the green line than scenario 40. The XGBoost prediction red line of scenario 40 in its term is more closely located to the green line than the red line in scenario 9.

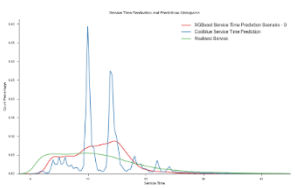

Figure 5.2. Scenario 9 Service Time Histogram

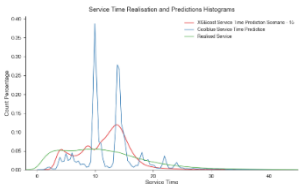

Figure 5.3. Scenario 15 Service Time Histogram

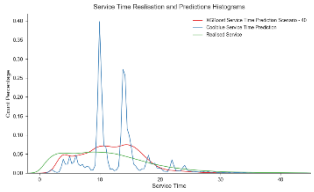

Figure 5.4. Scenario 40 Service Time Histogram

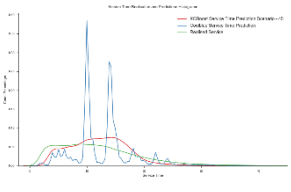

Figure 5.5. Scenario 40 Service Time Histogram

5.2. Scenario Sensitivity Analysis

Besides model scenarios that are created by adjustments made to the dataset, more model scenarios are created by using additional scaling operations, using a different train/test set partitioning, using a different cross-validation technique or using different hyperparameter settings. In Table 5.2 below, the outcomes of the various applied techniques sensitivity analysis are provided. The outcomes in Table 5.2 indicate the improve or decreased performance by applying a certain technique before the model is trained. The outcomes are measured as prediction performance improvements or reductions while applying the model relative to scenario 11 that is pointed out in detail in section 5.1. The prediction performances or improvements or reductions are measured in minutes.

First, 4 different scaling techniques are applied to the scenario 11 input dataset, which are; log scaling applied to the target variable, polynomial scaling applied to all independent prediction variables, standard scaling applied to all independent prediction variables and robust scaling applied to all independent prediction variables. Log scaling turns out to minimally increase the service time prediction with 0.001 minutes, which however is not significant in the 95% confidence interval range. The other three scaling techniques do not improve the service time prediction, in fact minimally negatively impacting the service time prediction. Performance impact results are given in Table 5.2 below.

Next, the impact of extra added or deleted variables is analysed. In section 5.1 already different input dataset configurations are described, however isolated performance impact is not discussed extensively. The model impact on adding the ‘Real Quantity Including Combination Box variable and the ‘Mean Service Time’ variables is described in Table 5.2. Adding the ‘Real Quantity Including Combination Box’ variable does not increase, nor decrease the service time model prediction. This means the ‘Real Quantity Including Combination Box’ variable does not have much impact on the model. Later, in section 5.5 it is visually shown that this variable does not have a high feature importance value. By adding the ‘Mean Service Time’ variable, the model performance improves with 0.02 minutes, which turns out not to be significant. Besides adding extra variables, some variables are sequentially deleted from the dataset. The first variables that are deleted from the dataset are; co-

driver number, second co-driver number, gender. By deleting those variables from the input dataset, the model is made privacy proof. Using privacy sensitive data as employee number and customer gender may not be acceptable to based predictions on. By deleting those 4 variables from the original dataset, the service time prediction MAE decreases by 0.04 minutes. Next, 3 other variables are one by one deleted from the dataset. These variables are; 'Floor Level Front Door House', 'Floor Level Installation Product Inside House' and 'Number of Return Products'. Those three variables are belonging to the set of most important variables for the prediction model. Later, in section 5.5, a further analysis is given about the importance of the features in the input dataset. By deleting the 'Floor Level Front Door House' the service time prediction MAE increases with 0.022 measured relatively to the dataset in which privacy sensitive data is deleted. Deleting the variable 'Floor Level Installation Product Inside House' negatively impact the service time prediction minimally with 0.001. Deletion of the 'Number of Return Products' variable has a very significant impact on the service time prediction. The MAE service time prediction error increases with 0.1 minute.

The following technique sensitivity check that is performed is the cross-validation technique used. In total 3 cross-validation schemes are inserted in the service time prediction model. In section 3.6.4 an it is argued that the K-Fold cross-validation technique is an improper cross-validation scheme for splitting up the train dataset. The other two cross-validation methods are shuffled day or tour based cross-validation or K-Fold shuffled day or tour based cross-validation. Both the shuffled and K-Fold shuffled techniques gives same prediction performance in the 95% confidence interval. Furthermore, the day base splitting process gives 0.015 better prediction performance than the tour based splitting process. This result is unexpected, because one could argue that the machine learning prediction model might be possible to see relations on service time durations happing on same days, which are included in in the same dataset while creating a tour-based splitting. With this proof, the day-based k-fold splitting scheme can now be used as main splitting process to describe the service time prediction MAE.

Next technique sensitivity check that is performed is the data partitioning check. Different data partitioning percentage splits are inserted to the model. In five iterations, the service time performance MAE is obtained from the test dataset percentages 0.1, 0.2, 0.3, 0.4 and 0.5. The difference in MAE between the 0.1 dataset partitioning and the 0.5 dataset partitioning is rather small with the value of 0.04. However, the MAE of the train/test set partitioning in the range of the above percentages are giving maximum MAE performance differences of 0.09.

To accurately compare all model scenario's, confidence intervals are calculated for all model scenarios. The confidence intervals are calculated by deploying the model several times with different random seeds. This assures that the orders composing the test dataset and the orders composing the train dataset are altered in by usage of different random seeds. The impact of random seeds in fact is already provided by notion of the confidence intervals belonging to the model scenarios given in Figure 5.1. Overall, it can be stated that service time prediction MAE impact of random seed adjustment is on average 0.07 minutes.

Last, Table 5.2 gives information about the service time prediction MAE impact of diversity in hyperparameter setting. The most important hyperparameters in the XGBoost predictions turn out to be the learning rate (or eta), the gamma (regularization term) and the maximum tree depth. In general, the learning rate is preferably set to a value as low as possible, although this increases the time to convergence. In Table 5.2 the prediction MAE impact of different max tree depth and gamma hyperparameter settings is provided. For the input dataset belonging to scenario 11, the best max tree depth turns out to be 7. However, it should be noted that for input dataset, belonging to other scenario models, the best max tree depth value is an integer value close to 7. The max tree depth value should be at least 4, but optimal max tree depth must be examined for each input dataset individually. Furthermore, it turns out the gamma parameter improves the performance relative to the model in which no gamma parameter is used with 0.04 minutes. For the input dataset belonging to the model scenario 11, the optimal gamma parameter value is found to be 0.7.

		Impact In Minutes
Scaling		
Log scaling Y variable	-	0.001
Product Variable all variables	+	0.091
Standard scaling all variables	+	0.046
Robust Scaling all variables	+	0.043
Add Variables		
PostalCode + Letters Concat Columns		0
Add Variable Real Quantity --> no impact		0
Add Variable Mean Service Time All Products > Count 30	+	0.02
Delete Variables		
Privacy Proof: driver / co-driver / second co-driver / genc	+	0.04
Floor Level Front Door House	+	0.062
Floor Level Installation Product Inside House	+	0.041
Nr Return Products	+	0.161
Cross-Validation-Procedure		
Day- based CV splitting vs tour-based CV:	-	0.015
Test Size %		
Test size 0.0 vs 0.5	+	0.04
Random Seed Train/Test Splitting		
Random Seed in Days - day based train/ test split used	+/-	0.07
Hyperparameter - Max Depth Tree		
4	+	0.02
6	+	0
7		Best
8	+	0.01
10	+	0.06
15	+	0.13
Hyperparameter - Gamma		
0.4	+	0.03
0.6	+	0.01
0.7		Best
0.8	+	0.07

Table 5.2. Model Scenario Sensitivity Analysis

5.3. Best Model MAE and ME

As denoted in section 4.4, numerical output statistics of the three most important KPIs are provided in this results chapter 5. The three most important KPIs are; MAE, ME and NPS score. In this subsection the MAE and ME performances are analysed. Later, in section 5.4, the NPS score performances are analysed. In section 5.1 of this study, it is indicated that two models can be pointed out as best models which are still viable to implement. The first best model is the model scenario indicated as model scenario 40. In this model scenario, the 26 standard input variables are enriched with the 3 variables composing extra feature set 1, all product aggregation level variables, 13 mean prediction error variable and mean service time variables, 13 count variables and the current 'planned service time' variable used for the current Coolblue service time prediction. Besides inputting these variables, the hyperparameters are turned by the Bayesian optimizer. The second-best model is equal to model scenario 45 provided in Figure 5.1 and analysed in section 5.1 of this study. The second-best model that is selected is the same concerning input dataset variables used. However, before the model is trained, the 4% lowest service times and 0.3% highest service times are removed from the dataset. The reason for indicating two models as best models in this research is that Coolblue indicated not to be sure the service time outlier deletion is grounded. More research needs to be performed to confirm the acceptance of service time outlier deletion.

In Table 5.3 up to and including Table 5.6 the MAE and ME performance of the two indicated best model scenario 40 and 45 are provided. Figures 5.6 and 5.7 present the numerical input provided in Table 5.3 up to and including Table 5.6 graphically. Note that in Figure 5.6 x percent of the data is deleted from the plot to enhance interpretability.

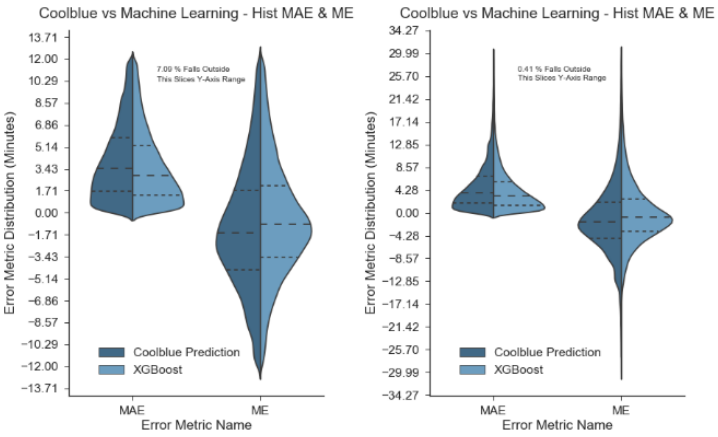

Figure 5.6. MAE and ME Performance Model Scenario 40 – 2 Y-Axis Ranges

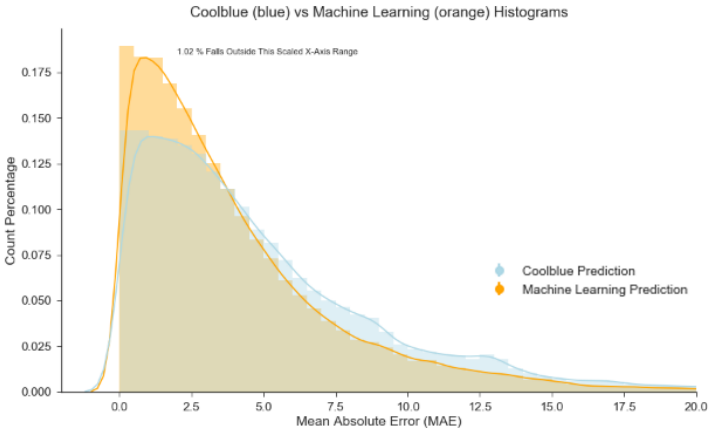

Figure 5.7. MAE Histogram Model Scenario 40

Both the service MAE and ME scores of the machine learning models as well as the representative current Coolblue prediction models are given. By reviewing Table 5.3 up to and including Table 5.6, it can be concluded that by using the XGBoost algorithm, the MAE of the service time improves significantly 1.09 minutes relative to the current Coolblue model by usage of model scenario 40. If model scenario 45 is used to make the predictions, the prediction improvement is even higher with the value 1.14 compared to the current Coolblue prediction method.

Besides the MAE prediction performance improvement, also the ME prediction is improved. The current Coolblue model overestimates the service times on average significantly with a value of 1.13 minutes for scenario 40 and a value of 1.07 minutes for scenario 45. By making use of the XGBoost machine learning prediction the ME of the service times is decreased to 0.19 by using model scenario 40 and 0.10 by using model scenario 45.

The 50% output statistics provided in Table 5.3 up to and including Table 5.6 includes another interesting statistic, which is the median value of the data. By reviewing the 50% statistics, it can be noted that both the current Coolblue model as well as the machine learning model predict the duration of most of delivery orders too long rather than too short. This overestimation of most of the orders is smaller by using the machine learning model compared to the current Coolblue prediction model.

Scenario 40 - MAE Score (in Minutes)		
Statistic Type	Current Coolblue Model	Machine Learning Model 1
Count	97366	97366
Mean	5.18	4.09
Std	5.41	4.94
25%	1.68	1.45
50%	3.43	2.98
75%	6.01	5.13
Min	0	0
Max	852.1	850.3

Table 5.3. Scenario 40 MAE Descriptive Statistics

Scenario 40 - ME Score (in Minutes)		
Statistic Type	Current Coolblue Model	Machine Learning Model 1
Count	97366	97366
Mean	1.13	0.19
Std	8.16	5.92
25%	-4.73	-3.87
50%	-1.89	-1.42
75%	1.24	1.73
Min	-207.3	-204.2
Max	852.1	850.3

Table 5.4. Scenario 40 ME Descriptive Statistics

Scenario 45 - MAE Score (in Minutes)		
Statistic Type	Current Coolblue Model	Machine Learning Model 1
Count	93218	93218
Mean	4.95	3.81
Std	5.12	4.21
25%	1.52	1.34
50%	3.24	2.74
75%	5.86	4.79
Min	0	0
Max	50.2	49.3

Table 5.5. Scenario 45 MAE Descriptive Statistics

Scenario 45 - ME Score (in Minutes)		
Statistic Type	Current Coolblue Model	Machine Learning Model 1
Count	93218	93218
Mean	1.07	0.1
Std	7.51	5.83
25%	-4.62	-3.64
50%	-1.72	-1.26
75%	1.22	1.56
Min	-14.5	-28.3
Max	50.2	49.3

Table 5.6. Scenario 45 ME Descriptive Statistics

5.4. NPS Results

In previous section 5.3 the MAE and ME scores are provided for new machine learning models as wel l and compared with the current Coolblue model. The MAE is the most important statistics metric us ed in this study. In section 1.2, the six reasons are provided that motivate the importance of service t ime prediction accuracy improvement for Coolblue. Resulting from the MAE scores, a simulation is cr eated in which new expected machine learning based NPS scores are estimated. In section 4.4, the mathematical expression of the NPS calculation is provided. In Figures 5.8 below the new machine le arning NPS estimations of model scenario 40 are provided in orange. In Figure 5.8 also the represent ative current Coolblue model estimated NPS performances are provided as light blue bars. In each of the intervals presented on the X-axis in Figure 5.8, p-value adjustments are indicated and separated by a '-' symbol. P-value adjustments are needed to be made, because resulting from Table 5.3 up to and including Table 5.6, the current Coolblue model is overestimation the mean service times by app roximately 1.1 minutes. Overestimating the service times on average means that the current Coolblu e model on average arrive 1.1 times the sequence number earlier at the customers house than the n ew machine learning model. To compensate for this discrepancy a p-value used that adjust the servic e time 1.1 minutes for both the current Coolblue model as well as the machine learning model. In fac t, a complete range of comparable p-values is created to come up with the best performance p-value in terms of NPS score. By reviewing Figures 5.8 the observation can be made that the new machine l earning model NPS scores in scenario 40 is not significantly improving the NPS scores. Comparing onl y the 1.1 minutes adjusted Coolblue model with the unadjusted machine learning model gives signifi cant improvement in the 95% confidence interval range. The machine learning confidence intervals i s [73.77 – 73.80] and the current Coolblue Model confidence interval is [73.72-73.76]. However, this significant improvement is only measured by using this p-values. In Figure 5.6 the machine learning model and Coolblue model bars are located more closely to each other at the end by using higher pa rameter p values. It is up to Coolblue to decide the most suitable parameter p-value. It does not hold that for all p-values the p parameter interval turns out to be significant and thus the NPS is not signifi cantly improving by implementation of the new machine learning model.
The optimal p parameter value is even obtained for the machine learning model as -1.4 and for the current Coolblue model as -1.2.

To deep dive on the NPS results, the number of too-early, on-time and too-late deliveries are compared for the current Coolblue model and the new machine learning model by using the model scenario 40 input dataset. The results are provided in Table 5.7 below. In line with expectation raised in Hypothesis 3, the percentage of too-late deliveries decreases by usage of the new machine learning model relative to usage of the current Coolblue prediction model. However, the percentage of too-

early and on-time deliveries is not decreased by usage of the new machine learning model relative to usage of the current Coolblue prediction model. An explanation of this simulation results is that the current Coolblue model arrival times are describing real historical arrival time behaviour. Delivery employees tried to arrive at the customer exactly on time and were successful in arriving on time on most of the deliveries. The machine learning prediction arrival times are derived from the current Coolblue arrival times. Because the machine learning prediction model arrival time estimations are expectations instead of real historical arrival times, the machine learning model in more deliveries arrive too-early at the customer. This harms the NPS score, resulting that no significant NPS score improvement can be concluded from this simulation analysis.

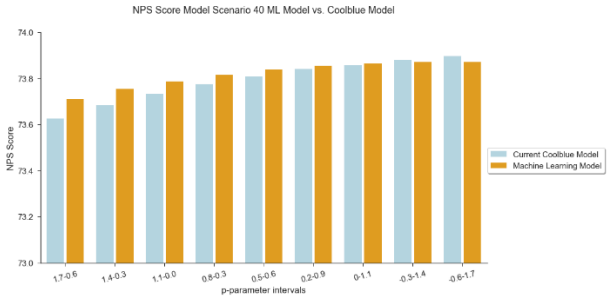

Figure 5.8. Scenario 40 NPS Estimation And Parameter 'p' analysis

	Too Early	On Time	Too Late
Current Coolblue Method	30%	59%	11%
Machine Learning Method	23%	64%	14%

Table 5.7. Time Performance Analysis

5.5 Accuracy Check Machine Learning XGBoost Model

The machine learning model gives more accurate service time of 1.09 minutes on average by using model scenario 40 and 1.14 minutes on average by using model scenario 45. The improvement in prediction performance triggers the implementation of the new machine learning model thereby replacing the current Coolblue prediction method. Still, as is always pointed out as the main disadvantage of complex machine learning algorithms, the internal decision rules used by the algorithm are not clear. The XGBoost algorithm is often referred to as a black-box algorithm. However, by creation of several deep dive plots, the accuracy and the internal decision rules used by the XGBoost algorithm can be revealed. In this section 5 types of plots are analysed. By using this type of plots, before and after implementation, the Coolblue companies does not need to scare away from implementation. The plots are providing accurate insides into the performance, the feature importances and the most important decision rules.

First, in Figure 5.9 the MAE of the machine learning model and the MAE of the current Coolblue model is provided on the product aggregation level. This scatterplot with confidence interval gives a very detailed expected performance of specific products. This is extremely useful to describe to potential effect of implementation of the XGBoost machine learning algorithm on product level to Coolblue management. Furthermore Figure 5.9 gives guidance for selection products

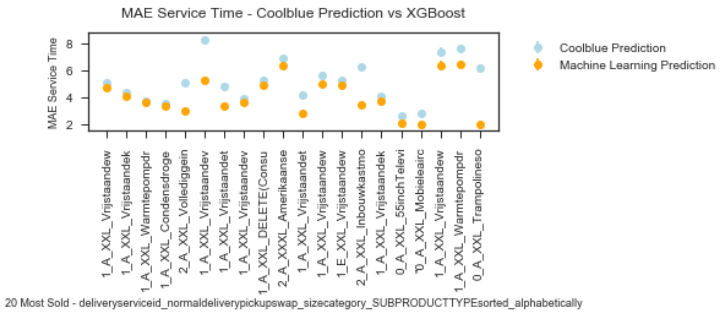

Figure 5.9. 20 Most Sold Product Performances impact Scatterplot

to focus on more with the aim creating even more accurate service time predictions. In Figure 5.9, the 20 most sold products are selected for presentation in this plot.

Second type of indicative plots are the feature importance plots. Feature importance plots help in uncovering the decision rules used by the XGBoost algorithm. There are three types of feature importance plots, each of them uniquely measuring importances of features.

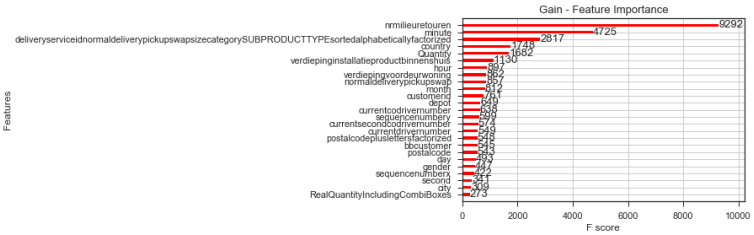

Figure 5.10. Gain Feature Importance

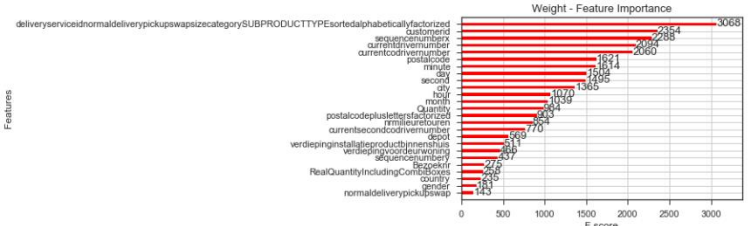

Figure 5.11. Weight Feature Importance

The first type of feature importance is based on weight. This is the number of times the XGBoost model splits on a specific feature in the input dataset. the second type of feature importance is based on gain ratio. In this plot the average of the gain achieved by splitting on a specific feature is determined. The third type of feature importance plot is based on cover ratio. The cover ratio describes the average percentage of samples that are input a split for specific features. Figure 5.10 and 5.11 give the feature resulting from model scenario 6. Observe that the ranking highly unequal by comparing both Figures. The gain feature importance type is determined to be more indicative than the weight feature importance type to base feature importance ranking on. It turns out that the number of return products is of highest for the XGBoost model to compose service time predictions. The product variable is indicated third most important variable by using the gain feature importance. Very surprisingly on ranked second the feature minutes is found. However, the importance of the minutes feature is highly unlikely to be accurate. It turns out this is caused by the low number of splits that are performed on the minutes feature. This surprising result indicate the inaccuracy belonging to determining feature importance by using this feature importance method. A new method for measuring feature importance is proposed with is called SHAP values.

Third type of plots giving detailed information about the model structure are the feature importance variable value level scatterplots. In total four scatterplots are created. Three of them describe the feature importance on variable value level instead of the classical feature importance plots presented in Figure 5.10 and Figure 5.11. For all the four scatterplot the 20 most important features are extracted based on the gain feature values importance ratio. Next, the mean

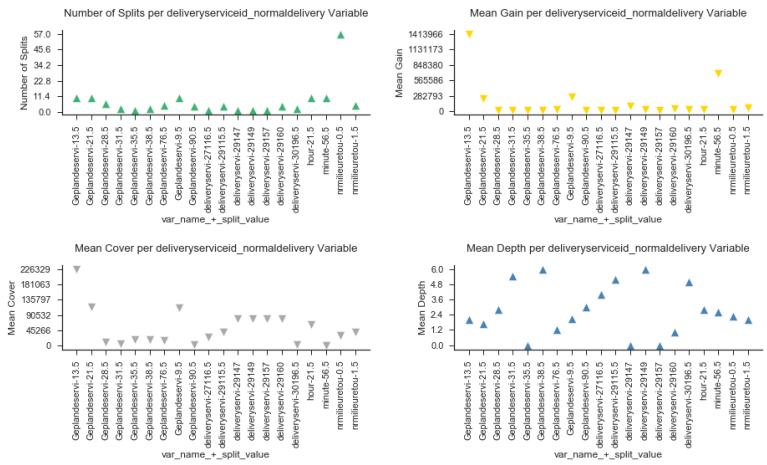

Figure 5.12. Feature Importance Variable Value Level Scatterplots

gain is provided for each of the 20 most important feature values by the yellow scatters. In grey, the mean cover is provided for each of the 20 most important feature values.

The mean weight of the 20 most important feature values are provided as green scatters. The blue scatters give a new, but very informative feature value importance indicator, which is the mean depth in the tree, where the splits are made for feature. The lower the mean tree depth level the feature values splits are made, the more important the features are, because more sample will be separated on in higher level of the tree on average.

Last type of visualizations is the SHAP value visualizations. In section 3.9.3 an explanation is provided why SHAP values is the only consistent and accurate and reliable feature importance metric. SHAP value is the reliable feature importance metric replacing the classical feature importance metric provided above. The classical feature importance metric can still be used, but only as indicator, not to base real feature importance on. Three types of SHAP value plots are perfectly indicating the feature importances used in the XGBoost algorithm. First, the SHAP summary barchart is provide in Figure 5.14. The SHAP summary barchart is equal to the classical feature importance plots in construction. It presents the feature importance in Log Odds values. Next SHAP plot is the summary scatterplot, provided in Figure 5.14. Like the summary barchart, the summary scatterplot gives the 20 most important features for the XGBoost model. However, different from the summary barchart, coloured scatters are being used to give an understanding of the importance of the relation between high and low values of the variable in terms of log odds. Last, in Figure 5.15 and Figure 5.16, 2 individual SHAP value plots are provided. In these individual SHAP value plots the SHAP values of the complete range of variable values can be observed. This is very informative. In Figure 5.15 for example it can be observed that the XGBoost computes higher service time predictions if the number of return products increases from 0 to 1 and if the number of return products increases from 1 to 2. The service time predictions belonging to higher number of return products are comparable to the service time prediction for 2 return products. By analysing Figure 5.16 it can be concluded that the XGBoost model is able to translate the floor level front door house accurately into the prediction composition. Higher floor levels on average results in in higher SHAP values, meaning higher service time predictions.

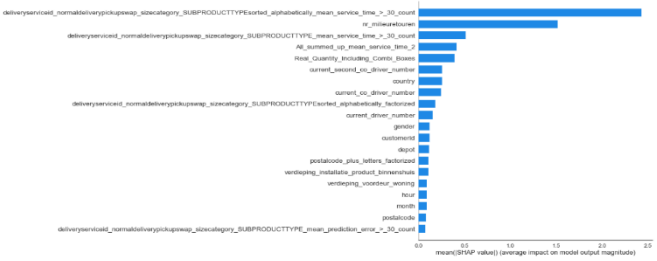

Figure 5.13. SHAP Summary Barchart Model Scenario 40

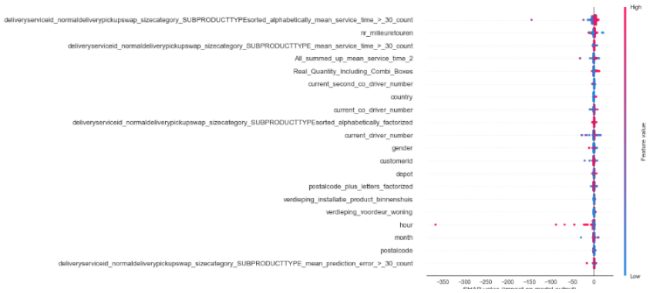

Figure 5.14. SHAP Summary Scatterplot Model Scenario 40

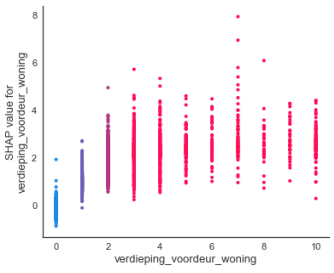

Figure 5.15. SHAP Individual SHAP Value

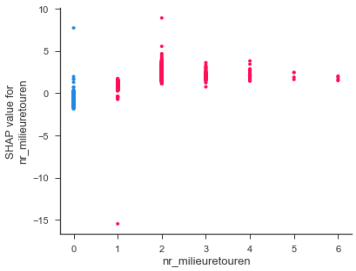

Figure 5.16. SHAP Individual SHAP Value

6. Conclusion

In this chapter the conclusions are drawn by using the results obtained by using a wide variety of model scenarios as input for the XGBoost machine learning model. Two main conclusions are drawn for the most important KPIs analysed in this study; the MAE and the NPS score. Next, some model (input) settings conclusions are drawn. Last, conclusions are formulated for the power of the model performance plots that are analysed.

First, the delivery service time prediction MAE decreases by using the new XGBoost machine learning model with 1.09 or 1.14 minutes relative to the current Coolblue service time prediction model. The XGBoost service time prediction are significantly more accurate than the linear regression service time predictions (by 0.39 minutes). Therefore, hypothesis one raised in section 3.8.1 is accepted. The prediction performance increasement is dependent on the input dataset that is applied to the model. 1.09 minutes MAE improvement is achieved by usage of the input dataset belonging to model scenario 40 and 1.14 minutes MAE improvement is achieved by usage of the input dataset belonging to model scenario 45. The hyperparameters are tuned for the predictions created in model scenario 40 and 45 for achieving the 1.09 minutes and 1.14 minutes MAE performance improvement respectively.

The XGBoost machine learning model can create more accurate service time predictions than the current Coolblue model. However, the new NPS scores estimations created by simulation of the expected Coolblue model predicted arrival times and the XGBoost model predicted arrival times are not increasing in the 95% confidence interval. The XGBoost model with its representative (p-value adjusted) current Coolblue model, NPS scores are 73.79 and 73.73 respectively. Although the difference between these two models are significant, still the observation is made in section 5.3 that for most p-value model combination the difference is not significant. Therefore, the decision is made to reject hypothesis 4 raised in section 3.8.2. Furthermore, hypothesis 3 stating that the number of too-late deliveries decreased by usage of the XGBoost model is confirmed. On the contrary, the number of too-early deliveries does not decrease by usage of the XGBoost model. Therefore hypothesis 2 is rejected.

Besides the KPI conclusion drawn above some extra conclusions can be drawn about the optimal input parameters that can be used in the model. The high-level conclusion can be drawn that the XGBoost model is capable of accurately separating the important features from the less important features. Although in some model scenarios no significant improvement is obtained, the conclusion can be stated that the more variables that are added to the input dataset, the better the service time predictions created by the XGBoost model. To achieve service time prediction improvements, it remains very important to add indicative features to the input dataset. It is recommended to add all product aggregation levels to the input dataset. The XGBoost algorithm can subtract useful information from each individual product aggregation level variable. Furthermore, if one product aggregation level wants to be inserted as input to the model, it is recommended to use either the Sub Product Type product aggregation level or an aggregation level that is derived from the Sub Product Type aggregation level. Adding the size category, delivery service, and delivery type to the Sub Product Type aggregation level is beneficial in terms of performance MAE.

By reducing the 4% lowest service time values and the 0.3% highest service time values, the model prediction accuracy increases even more significantly with a value 0.05 minutes. Adding the variables

'planned service time', 'real quantity including combi boxes', 'mean service time visit counter', 'mean service time' and 'mean prediction error' are not significantly improving the XGBoost prediction, however these variables together still helps in reaching a small amount of prediction performance improvement. Last, the 'counter variables' are increasing the prediction performances of the XGBoost model. Those variables describe the number of times for example a product, driver or floor level is put on. In section 5.1 it is already indicated that those 'count variables' are tricky in terms of train / test dataset data leakage perspective.

Furthermore, no significant improvement in service time prediction MAE is obtained by using any of the four feature scaling techniques applied to the data. Log scaling of the target variable is the only scaling technique improving the prediction minimally, with 0.001 minutes. This value however is not significant. The standard scaling, robust scaling and polynomial scaling techniques are not capable of improving the service time prediction. This conclusion can be made that the XGBoost algorithm is insensitive to diversity in model feature axis scales.

The service time MAE performance difference by using different test size partitions are not very large. Using 10% of the complete dataset as test size gives comparable performances as using 50% of the complete dataset as test size (difference is 0.04 in MAE). However, it is still recommended to use a reasonably large test set as for example 30% of the complete dataset.

Two representative cross-validation methods are discovered that suitable for the service time prediction problem. The first cross-validation method is shuffled cross-validation. The second cross-validation method is shuffled K-Fold cross-validation. The performance of both cross-validation methods is comparable, however K-Fold cross-validation performs slightly better than the shuffled cross-validation method. By using one of the two cross-validation methods, the cross-validation unit must be provided. The optimal unit to split on is day-based, however it is recommended to use tour-based splitting in case NPS score are being determined with the prediction results. The day-based unit split gives 0.04 minutes better MAE performance, which is significant.

The learning rate (eta) hyperparameter setting could best be set to a value in the range 0.05 until 0.2. Lower learning rate hyperparameter settings results in very high computation times, whereas learning rate hyperparameter values higher than 0.2 are negatively impacting model performance. Besides the learning rate hyperparameter, also the max tree depth and gamma parameter turn out to be most important hyperparameters in the service time prediction model. By test applied to model scenario 9, it turns out the optimal max tree depth parameter turns out to be 7 and the optimal gamma parameter turns out to be 0.7. However, not that these parameters are very sensitive for the input dataset provided to the XGBoost prediction model. Therefore, if other input dataset is used than the input dataset in model scenario 9, the optimal max tree depth and optimal gamma hyperparameter values of 7 and 0.7 respectively are likely to chance as well.

Last, several conclusions can be drawn from analysis of the variable impact plots and the feature importance plots. The variable importance plot given is a good way of analysing and describing the prediction model impact for specific variable values. To analyse the construction of the model and uncovering the decision rules used by the model, the classical feature importance plots, the feature value importance scatterplots and SHAP value plots can be used. Note, that the classical feature importance plots are not reliable, consistent and very accurate, still the plots can be used to understand the model structure better. It is highly recommended to use the SHAP values to measure feature importances and feature value importances.

7. Recommendations

In this chapter the recommendations for feature research are provided. The recommendation is useful to analyse before implementation of the XGBoost prediction model is practiced.

7.1 Recommendation 1

The most important recommendation is that the XGBoost machine learning model can be implemented as service time prediction method for Coolblue Delivers. The XGBoost machine learning prediction model preferably replaces the current Coolblue rule-based prediction method. By usage of the variable impact plots and different the three types of feature importance plot the model structure and potential model inaccuracies can be detected both before and after the implementation phase. Also, without the privacy sensitive variable 'driver number', 'co-driver number', 'second co-driver number' and gender together with the unavailable variables 'number of return products', 'floor level front door house' and 'floor level installation product inside house' the XGBoost prediction outperforms the current XGBoost model on average with 0.93 minutes in terms of MAE.

7.2 Recommendation 2

It is recommended to create more variable impact plots, or variable value impact plots. For example, 'Floor Level Front Door House Variable Impact Plots', 'Number of Return Products Variable Impact Plots' and 'Real Quantity Variable Impact Plots' could be created to get a better understanding of machine learning model prediction accuracy and inconsistencies. Also, deletion of the 4% of lowest service time values and 0.3% of highest service time values slightly increases the XGBoost prediction performance with 0.05 in relative to the current Coolblue prediction model.

7.3 Recommendation 3

Reviewing the NPS scores of different value of the p-parameter, the highest NPS score results from the p-parameter -1.4 for the machine learning prediction model and -1.2 for the current Coolblue prediction model. It is up to Coolblue to make the decision whether it is desirable to increase the planned service time of the current Coolblue model by 1.2 minutes and the predicted service time of the XGBoost machine learning model by 1.4 minutes. The values of 1.2 minutes and 1.4 minutes must be multiplied by the sequence number of the delivery order.

7.4 Recommendation 4

In the conclusion section the statement is made that adding extra variables very rarely results in less accurate service time prediction MAE results. The XGBoost algorithm can separate unimportant variables from the important variables. Therefore, it is recommended to engineer more features to enhance the model performances. Always keep a critical eye on potential data leakage between train dataset and test dataset. Furthermore, try to make a shortlist of variable that, also by logical reasoning, are helpful in guiding the XGBoost algorithm to a more accurate service time prediction. Also, usage of third-party delivered extra information, as Cadastre related data is advisable to test in the service time prediction model. Cadastre related data is very likely to increase the service time prediction further.

7.5 Recommendation 5

If the Bayesian optimization method is performed for determining the right hyperparameter settings, it is recommended not to use a very wide range in hyperparameter settings since this results in a very long time to convergence before a good fitting hyperparameter set is determined. Select the indicated max depth, learning rate (eta) and gamma range as given in section 5.2 or chapter 6.

7.6 Recommendation 6

The delivery service times and the driving times together composes all components related to the customer arrival time predictions. No investigation has been done on the inconsistency of driving time predictions created by Coolblue. The driving times taking longer on average than the service times. To reduce the variability in arrival time prediction more in the future, it is advisable to research the driving time prediction improvement options.

8. Discussion

The accuracy of the predictions made in this delivery service time prediction study are highly dependent on the accuracy and quality of the input parameters provided to the model. In this chapter the accuracy and quality of the service time predictions and input variables is discussed.

First, the service time error calculation is dependent on the quality of the target variable. The target variable is 'stoptime minutes' in this service time prediction study. In this study it is assumed that the 'stoptime minutes' variable provides use with reliable historical realised service time data. However, the data composing the 'stoptime minutes' variable is logged by the driver employees themselves. The driver employees indicate the time of arrival at the customers house by pressing the arrival time button on the mobile phone at the actual time of arrival. The same hold for the departure time from the customers house. The driving employees are expected to press the departure time button on the mobile phone at the moment they start driving the van after the products has been delivered at the customers house. However, the exact time at which the customers press the arrival time button and the departure time button will undoubtedly be different than the exact and expected arrival time.

Besides the most impactful service time logging accuracy inaccuracy that undoubtedly is present in the 'stoptime minutes' target variable, the accuracy of the service time predictions is also dependent on the quality of the independent predicting variable in the model. Most of the variables are directly obtained from the Coolblue interval database. Inconsistencies of the data in the Coolblue database system could are likely to be present. This harms the results obtained in this service time prediction study.

In the pre-processing phase some orders are deliberately deleted from the input dataset. This is because for some orders much of the data was missing, a problem occurred while delivering the product (i.e. customer not at home at the exact time of delivery) or extremely high or low realised service times where deleted from the dataset because they are classified as incorrect service time loggings (scenarios 45 and 46). By deletion of orders in the test dataset, average number of order per tour number in the test dataset decreases. This subsequently also decreases chance of too-early and too-late arrivals, because the chance of too-early and too-late arrivals is dependent on the accumulation of service time variability later phases of the tour execution.

The results depicted in Figure 5.1 and described in section 5.1 are mean values of the x number of random train / test splits that are used to compose this service time prediction MAE number. However, the amount of train / test splits differs for the model scenarios. To enhance the accuracy of model scenario interpretation, the amount of train / test split partitions applied to the model scenario is indicated in the description of the different model scenarios in section 5.1 as well.

Last, 8 May 2018 the driving employees started logging the floor level front door house information and the floor level installation product inside house information. This means that 56% of the input dataset consist floor level related information logged by the data. For the remaining 44% of the input dataset the floor level related information is not present. This impact both the accuracy in service time prediction and the feature importance metric values for these two variables.

9. References

Almaldi, E., & Kann, V. (1998). On the approximation of minimizing non zero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, *209*, 237–260.

Benjamin, A. S., Fernandes, H. L., Tomlinson, T., Ramkumar, P., VerSteeg, C., Kording, K. P., … Kording, K. P. (2017). Modern machine learning far outperforms GLMs at predicting spikes. *BioRxiv*, 111450. https://doi.org/10.1101/111450

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. *Advances in Neural Information Processing Systems (NIPS)*, 2546–2554. https://doi.org/2012arXiv1206.2944S

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *2000 ACM SIGMOD International Conference on Management of Data*, 93–104. https://doi.org/10.1145/335191.335388

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. https://doi.org/10.1145/2939672.2939785

Colin Cameron, A., & Windmeijer, F. A. G. (1997). An R-squared measure of goodness of fit for some common nonlinear regression models. *Journal of Econometrics*, *77*(2), 329–342. https://doi.org/10.1016/S0304-4076(96)01818-0

ecommercenews. (2018). ecommercenews. Retrieved from https://www.ecommercenews.nl/grootste-webwinkels/

Eindhoven University of Technology. (2018). Eindhoven University of Technology. Retrieved from https://www.tue.nl/en/education/graduate-school/master-operations-management-and-logistics/

Gómez, A., Mariño, R., Akhavan-Tabatabaei, R., Medaglia, A. L., & Mendoza, J. E. (2016). On Modeling Stochastic Travel and Service Times in Vehicle Routing. *Transportation Science*, *50*(2), 627–641. https://doi.org/10.1287/trsc.2015.0601

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research (JMLR)*, *3*(3), 1157–1182. https://doi.org/10.1016/j.aca.2011.07.027

Hastie, T., Tibshirani, R., & Friedman, J. (2013). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. *Springer Science & Business Media*. https://doi.org/10.1111/j.1467-985X.2004.298_11.x

Kovacs, A. A., Golden, B. L., Harti, F. H., & Parragh, S. N. (2014). Vehicle Routing Problems in Which Consistency Considerations are Important: A Survey. *Wiley Online Library*.

Lin, C., Choy, K. L., Ho, G. T. S., Chung, S. H., & Lam, H. Y. (2014). Survey of Green Vehicle Routing Problem: Past and future trends. *Expert Systems with Applications*, *41*(4 PART 1), 1118–1138. https://doi.org/10.1016/j.eswa.2013.07.107

Liu, F. T., Liu, F. T., Ting, K. M., & Zhou, Z. (2008). Isolation Forest. *IProceeding ICDM '08 Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, 413--422. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.678.3903

Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2018). Consistent Individualized Feature Attribution for Tree Ensembles. Retrieved from http://arxiv.org/abs/1802.03888

McEliece, R. (2013). Entropy and mutual information. *The Theory of Information and Coding*, 17–49. https://doi.org/10.1017/CBO9780511606267.006

Patten, M. L., & Newhart, M. (2017). *Understanding research methods: An overview of the essentials*. Taylor & Francis.

Reichheld, F. (2004). The One Number you Need to Grow. *Harvard Business Review*, *81*, 46–54.

Rousseeuw, P. J., & Hubert, M. (2011). Robust statistics for outlier detection. *John Wiley & Sons*, 73–79. https://doi.org/10.1002/widm.2

Shmueli, G., & Koppius, O. R. (2011). PREDICTIVE ANALYTICS IN INFORMATION SYSTEMS RESEARCH.

MIS Quarterly, *35*(3), 553–572. https://doi.org/10.5121/ijccms.2016.5301
Tipping, M. E. (2004). Bayesian inference: An introduction to principles and practice in machine learning. *Advanced Lectures On Machine Learning*, *3176*, 41. https://doi.org/10.1162/15324430152748236
twinklemagazine. (2018). twinklemagazine.
Witten, I. H., & Frank, E. (2011). *Data Mining. Practical Machine Learning Tools and Techniques*. *Journal of Visual Languages & Computing*. Morgen Kaufmann.

10. Appendix

Original Input Dataset Features	
1 shipper id	16 quanitity
2 sequence number	17 postalcode plus letters
3 tour number	16 country
4 year	17 customer id
5 month	18 city
6 day	19 shipmentaddressid
7 hour	20 gender
8 minute	21 floor level front door house
9 second	22 floor level installation product inside house
10 postalcode	23 driver number
11 normaldelivery / pickup / swap	24 co driver number
12 visit type	25 current second co driver number
13 depot	26 number of retour products

Table B.1: Original Input Features

Creation		Stoptime_Minutes Normaldelivery / Swap / Pickup Floor Level Front Door House Floor Level Delivery Location Product Inside House
Conversion	Coercing	Shipmentaddressid
	Mapping	Normaldelivery / Swap / Pickup Demaging Client Belongings Floor Level Installation Product Inside House B2B Customer Country Visit Type
	Factorizing	Depot Postalcode City
Correction		Stoptime_Minutes
Transformations		Order_id Aggregations
Missing Value Completion		Replaced With Value "-2"

Table B.2: General Data Preparation Operations

Delivery Status - Delivery / Depot		Percentage Nr. Sequence Number on Tour	
Delivery Orders	310595	% Orders where Sequence Number < 10	62.47%
Depot Orders	0	% Orders where Sequence Number < 15	87.97%
% Delivery Orders	100%	% Orders where Sequence Number < 20	91.27%
Delivery Status -Delivered / NOT Delivered		**Shipper ID**	
Total Nr. Orders Delivery Status Recorded	273599	Nr. 2 Man Delivery Orders	256411
Total Nr. Orders Delivery Status NOT Recorded	35205	Nr. 1 Man Delivery Orders	35853
Nr. Orders Products NOT Delivered	1791	Nr. Bike Deliveries	18331
Percentage NOT Delivered	0.65%	% 2 Man Delivery Orders	82.55%
Damage During Product Delivery / No Damage		**Same Client Deliveries Last 7 Months**	
Total Nr. Orders Damage Recorded	255463	More Than 5 Times	0.007%
Total Nr. Orders Damage NOT Recorded	53932	More Than 3 Times	0.037%
Nr. Orders with Client Damage	1200	More Than 1 Time	2.670%
Percentage Damage	0.47%	Just 1 Time	97.33%

Table B.3: 6 Features Numeric Exploratory Data Analysis (EDA)

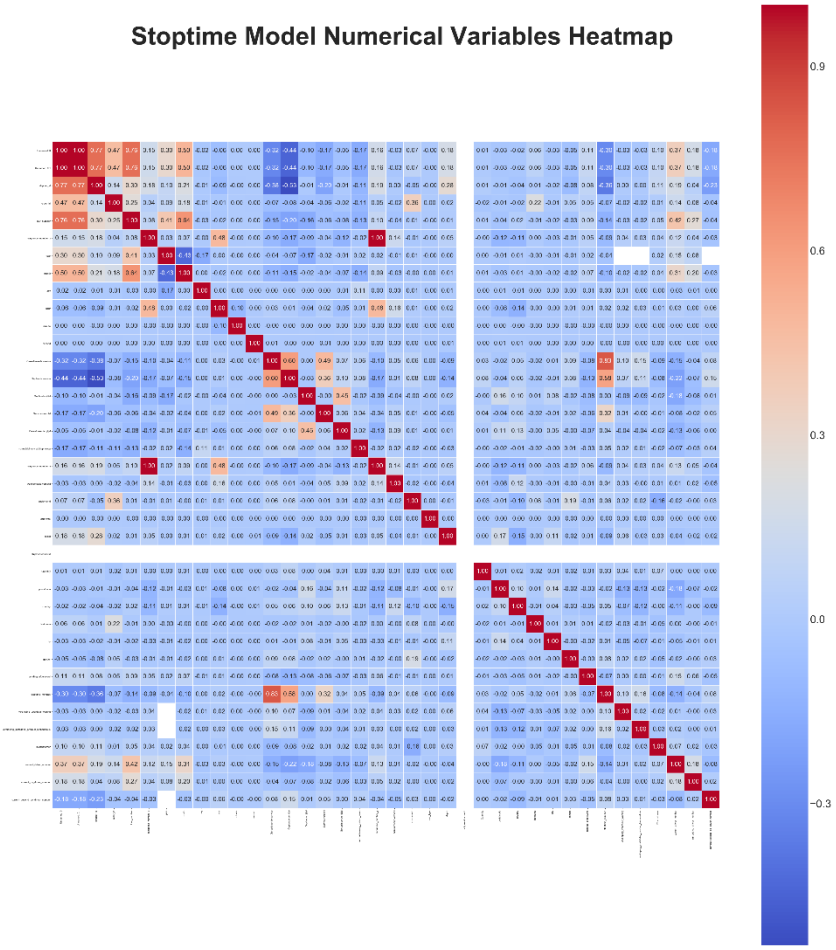

Figure B.1: Heatmap Variable Correlation Interaction