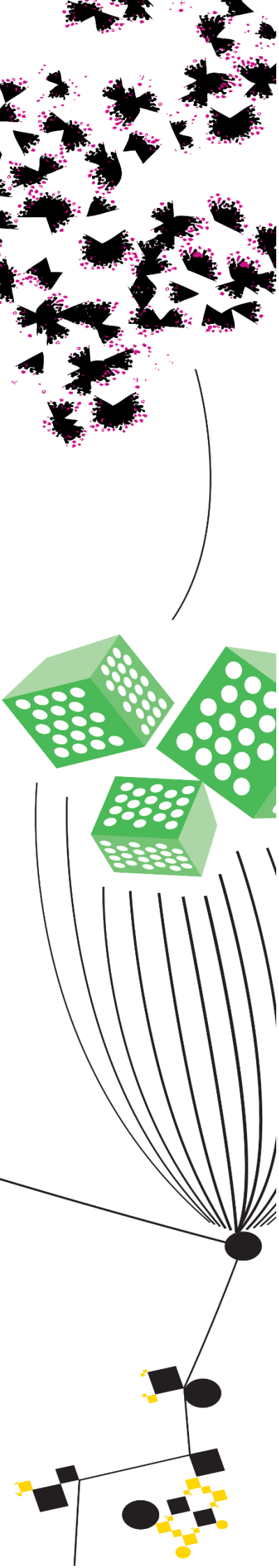


September 2021

# DATA-DRIVEN LOGISTICS:

## IMPROVING THE DECISION-MAKING PROCESS IN OPERATIONAL PLANNING BY INTEGRATING A SUPERVISED LEARNING MODEL

**Master thesis**

Author:
D.M. van den Heuvel

**Examination Committee**

Supervisors University of Twente:
dr. ir. W.J.A. van Heeswijk
dr. ir. M.R.K. Mes

Supervisor CAPE Groep:
B. Knol

UNIVERSITY OF TWENTE.

CAPE GROEP

# Management Summary

The logistics sector is in the middle of incorporating more data-driven methods into different processes of the supply chain. With the increasing availability of data analytics and artificial intelligence outside their original fields, software systems become more adaptable with these modern techniques. Logistic Service Providers use various systems within their operations which are keen for integrating these data-driven opportunities. In our research, we develop a methodology that records real user data from the system and incorporates multiple Supervised Learning models to identify the most important features within the replanning process and improve the planning performance. This research is conducted as a case study at one of the clients from CAPE Groep (a large, logistics service provider) and the proposed methodology and solution design are analyzed on the operational planning system.

The current operational planning system of the client is provided with a tactical planning at the beginning of a month. This tactical planning functions as the input of the system: a division of the distribution area in which all orders will be initially planned. Every day, human planners make adjustments on this initial planning to create a feasible operational planning at the end of their work shift. The total replanning process by the human planners takes a lot of time, since all adjustments are manually converted in the planning. This research focuses on two root causes associated to this problem: there is no learning model present that aids the human planners and there is no insights of the most important replanning factors. We propose a pattern recognition technique that can learn on past adjustments made on the planning in order to predict the total number of adjustments in the future. Based on the DSRM research methodology, we create a minimum viable product (an Artifact) to decrease the total time spent on replanning by the human planners.

Our research proposes a four-stage methodology to answer this question: data collection and preprocessing, learning model exploration, experimental tuning and cross-validation and finally, classification performance and model evaluation. Data is collected with an User Action Recording (UAR) mechanism, which is a hard coded Javascript file that can be added as an external widget on the planning system. After the data preprocessing steps, we ended up with an viable input data (around 76% of the raw data) from two months of replanning adjustments.

Based on an extensive literature study, this research provides a solution design that consists of various Supervised Learning models. This pattern technique best suits our problem, since we are interested in predicting the correct rides that need to be replanned. The output of the models is based on multi-class classification (due to the possible rides in the planning software). The used models in our research are the Decision Tree, Random Forest, Naïve Bayes and Neural Network. Also, data in our research is heavily imbalanced. To take this into account, we proposed additional performance metrics like Cohen's Kappa and Precision-Recall Curve to overcome this challenge.

We tune the parameters of our learning models by several grid-search experiments. For the tree-based models, we tuned the optimal depth, minimum samples per split, minimum Gini gain and the number of trees (Random Forest only) for our classification problem. For the neural network, we tuned both the optimizer (Stochastic Gradient Descent) and architecture parameters. After the tuning experiments, we proposed new parameter settings for each learning model.
Based on 10-fold cross-validation results, the Random Forest classifier scored the highest on five of the seven performance metrics. The scores of the metrics were: Cohen's Kappa (64.84%), Precision (54.37%), Recall (54.39%), ROC AUC (0.92) and Accuracy (64.84%). Looking more in-depth at the Precision-Recall curve, the neural network also showed promising results (Area-Under-Curve = 0.56) by almost outperforming the Random Forest (Area-Under-Curve = 0.560) on the imbalanced data metric.

To find the most important replanning factors, we compared various feature selection methods, both Classifier Specific and Classifier Agnostic. Based on paired t-test, the resulting feature scores from the Permutation Importance technique were significant across all models, with a moderately to very good correlation coefficients (values between 0.622 and 0.921 ($p < 0.05$)). Also, the most important (top-8 ranked) and unimportant (bottom-6 ranked) features of the replanning process were identified, which were used when creating the Artifact.

To improve the operational planning performance, we implemented the Artifact and calculated the estimated benefit on the total replanning time. Three assumptions were made regarding the current replanning process and the Artifact (based on the Random Forest model) is configured on data collected from a one-month period. Two scenarios are compared: manual adjustments (by the human planners) and automated adjustments (by the Artifact), which are tested on five days (five plannings). The Artifact resulted in a decrease of the total replanning time by approximately 30.61% (around 100 minutes), improving the current planning process. Therefore, the Artifact also has some practical significance and we proposed the next steps for the Client to create the decision-support system for the current application.

The results of this research substantiate the potentials of integrating a data-driven method like Supervised Learning into a real-life planning process. From a theoretical point of view, the artificial intelligence methods are able to be properly convert input data in replanning predictions, making them adaptable for logistic planning problems. Also, we overcame well-known challenges within data analytics to find the most promising and best performing model on imbalanced data. From a practical perspective, we were able to successfully create an Artifact (minimum viable product) that can be used in practice to improve the planning performance of the human planners. This would also provide feedback to keep developing the decision-support system in the future.

# Preface

*Before you lies the thesis "Data-Driven Logistics: Improving the Decision-making process in operational planning by integrating a Supervised Learning model", which discusses the potential of integrating machine learning models in the daily planning of a large logistic service provider. The research marks as the end of my master Industrial Engineering and Management, which I was not able to complete it without the help of my supervisors from the University of Twente and CAPE Groep.*

*First of all, I would like to Wouter for the excellent guidance during this project. Every time we had a meeting, which was sometimes quite long, your knowledge and opinion provided a clear path in this complex problem and its challenges. The quality of this thesis would definitely not be as high without your help. Also, I would like to thank Martijn for his critical view on the used methodology and your feedback brought the research to a higher standard.*

*I am also very grateful for the time and opportunities at CAPE Groep. I especially want to thank Bart for being a great supervisor and supporting me from day one. Our weekly meetings were a great moment to discuss the opportunities of this research and I always appreciated your trust in me during the entire period. Also, I wanted to thank both Narasimhan and Rick for their ideas and cooperation during the implementation phase of the research.*

*Besides the challenges of the COVID-19 pandemic during most of my master, I look back at the most incredible time of my life. I am very proud for all the amazing friends I have made for the rest of my life and I want to thank everyone for being there during all the laughter, fun moments and exciting experiences. Also, I want to thank my parents for providing me with the motivation and opportunity to attend an university. Finally, I would like to thank Lotte for her encouragement, loving support and being there during the difficult moments.*

*I hope you will enjoy reading this thesis*

*Mike*

*Utrecht, 8th of September, 2021*

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

**4PL** Fourth-Party Logistics Provider

**AI** Artificial Intelligence

**DC** Distribution Center

**DL** Deep Learning

**DSRM** Design Science Research Methodology

**DSS** Decision Support System

**IID** Independent and Identically Distributed

**KPI** Key Performance Indicator

**LSP** Logistics Service Provider

**ML** Machine Learning

**MLP** Multi-Layer Perceptron

**NN** Neural Network

**OOMPD** Operational Order Management Pakket Dienst

**PCA** Principal Component Analysis

**RL** Reinforcement Learning

**SGD** Stochastic Gradient Descent

**UAR** User Action Recording

# Chapter 1

# Introduction

This research is conducted at CAPE Groep, a company that specializes at creating value for logistics companies by making low-code software applications. Low-code is a method of software development that enables the quick creation of (business) applications with a minimum required amount of coding. The low-code applications are programmed in the software program Mendix. This app-modeling studio is very user-friendly, because it allows various experts and developers to collaborate and create value together. The resulting applications make the end-user's processes more tangible, which help reaching their company strategies and goals. CAPE Groep realizes the ambitions of their customers by creating digital innovation in Mendix to provide clear and versatile solutions. The main industries where CAPE Groep streamlines the customer processes are Logistics, Construction, Energy and Agriculture. This research focuses on a specific software application that CAPE Groep implemented at a major logistics company, which will be named the Client from this point on.

The Client uses a planning software called "Operational Order Management Pakket" Dienst (OOMPD), which is used for the daily, operational planning for the delivery of parcels. Each Distribution Center (DC) of the Client has its own OOMPD software application tailored to its physical specifications and routing configurations. At the beginning of each month, the OOMPD receives a tactical planning, that functions as the input for the entire monthly planning. This planning is basically a division of the total distribution area of the DC. The entire distribution area consists of a lot of unique postal code blocks and each postal code block is linked to one specific ride. More details regarding this division will be discussed later. Essentially, the rides consists of a unique division of postal code blocks. This division is necessary, because each Postal code block is linked to either a Client's delivery van or a subcontractor delivery van. When an order is received in the application, the parcel(s) are automatically linked to a certain ride and by extent, to a specific delivery van. The purpose of the OOMPD application is to make a daily planning of all parcels, based on the division of the shifts and rides.

Multiple times a day, three to four human planners use the OOMPD software to make adjustments on the input planning. Orders that need to be replanned are triggered by alerts that enter the system. Alerts are based on incoming messages from subcontractors and contains relevant details of what exactly needs to be adjusted in the current planning. The human planner makes sure that all adjustments are carried out, by manually replanning the orders on the right shift and/or correct ride. Besides the alert information, the human planners take also various constraints and parameters into

account (i.e., the physical limitations of a delivery van). This replanning process therefore consists of all manual actions by the human planners to make the input planning more feasible for the operations. When all adjustments are carried out, the human planners finalize the improved operational planning.

The goal of this research is to introduce the possibilities of data-driven learning methods on logistic planning software. The thesis focuses primarily on the human planner adjustments on the provided tactical planning by measuring all actions with the use of an innovation: the User Action Recording (UAR). The characteristics and implementation of the UAR and other elements of the current planning process will be further described in Section 2. This study introduces the potential of Machine Learning (ML) and its learning methods to discover recurrent patterns in human planner adjustments. The aim is to classify the most important features that occur during the replanning process, by implementing various supervised learning models that fit the UAR data. This will be used to create a Decision Support System (DSS) in order to predict the replanning adjustments and provide them as suggestions to the human planners, which will decrease the total time needed by the human planner.

The remainder of this chapter introduces more aspects of the conducted research. Section 1.1 provides the motivation and relevance of this study. In Section 1.2, we describe the core problem that this thesis wants to solve. The proposed research objective and questions are given in Section 1.3 and finally, the scope of the study (the boundaries and delimitations) is provided in Section 1.4.

## 1.1 Motivation

Wang and Alexander (2015) mention that the logistics sector is in the middle of a digital transformation, from non-analytical to incorporating data-driven methods in its business operations and the decision-making processes. CAPE Groep is one of the Dutch key front-runners of providing IT-based solutions by implementing low-code software applications for business processes. The OOMPD is an operational planning tool created by and for the Client, which can be improved even further by introducing more data-driven logistics. Combining and enriching this software with substantial amounts of real-time data from the human planners, can further develop and improve the process. Originally, pattern learning techniques grew out of the field of computer science (Bishop, 2006, p. 7) and nowadays have substantial influence in the use of image-processing applications and forecast/medical predictions, i.e. in the researches of Lecun et al. (2015) and Kourou et al. (2015). Even though incorporating pattern recognition techniques in the field of logistics is not new in the scientific world, the number of real-life solutions and practices remains limited to this day (Koot et al. (2021)). Adapting these new learning models and methods on existing logistic processes can provide new insights and drive innovations, which is the main motivation of this study. Discovering patterns in the human planner adjustments could benefit the process efficiency of finalizing the operational planning. We achieve this by classifying the most important features that are used for replanning (the business insight) and then propose a methodology to improve the existing process between the input and operational planning.

The relevance of this thesis lies in the practical use of recording actual human planner data to improve the existing planning processes. The data will be obtained from the UAR, which allows us to measure information of every single planning adjustment in the real-life application. This data will be used as input for learning models and pattern recognition techniques to identify important elements of replanning and to classify the most recurrent patterns and important features in the decision-making

process. To make sure that this research contributes to the real-life practice, we propose a Decision-Support System (DSS) that integrates the human planner data and the best learning algorithm within the OOMPD application. This will result in a strong foundation for a continuous and lasting implementation process.

This research also provides scientific relevance. The major strength of this thesis is the introduction of learning algorithms on real-life user data and human planner experiences. This integration of a IT-based logistics planning system and the statistical data analysis strengths of pattern recognition, are unique in the scientific world. This thesis will find the new practices of incorporating data-driven learning models that strengthen the performance of real-life logistics planning software and potential of the decision-making process between operational and tactical planning.

## 1.2 Problem identification

Mentioned previously, the tactical planning functions as the input for the software and consists of the division of the DC's distribution area. Basically, this division results in daily, pre-planned input for the human planners. The Client states that the current planning system lacks some dynamic elements and that the company wants to integrate a more data-driven approach for the operational planning system. To identify the core problem of the system, we use the Root Cause Analysis method from Wilson et al. (1993) to identify the various faults, symptoms and possible root causes. Figure 1.1 shows our Root Cause Analysis and we will briefly explain each layer in the following paragraph.

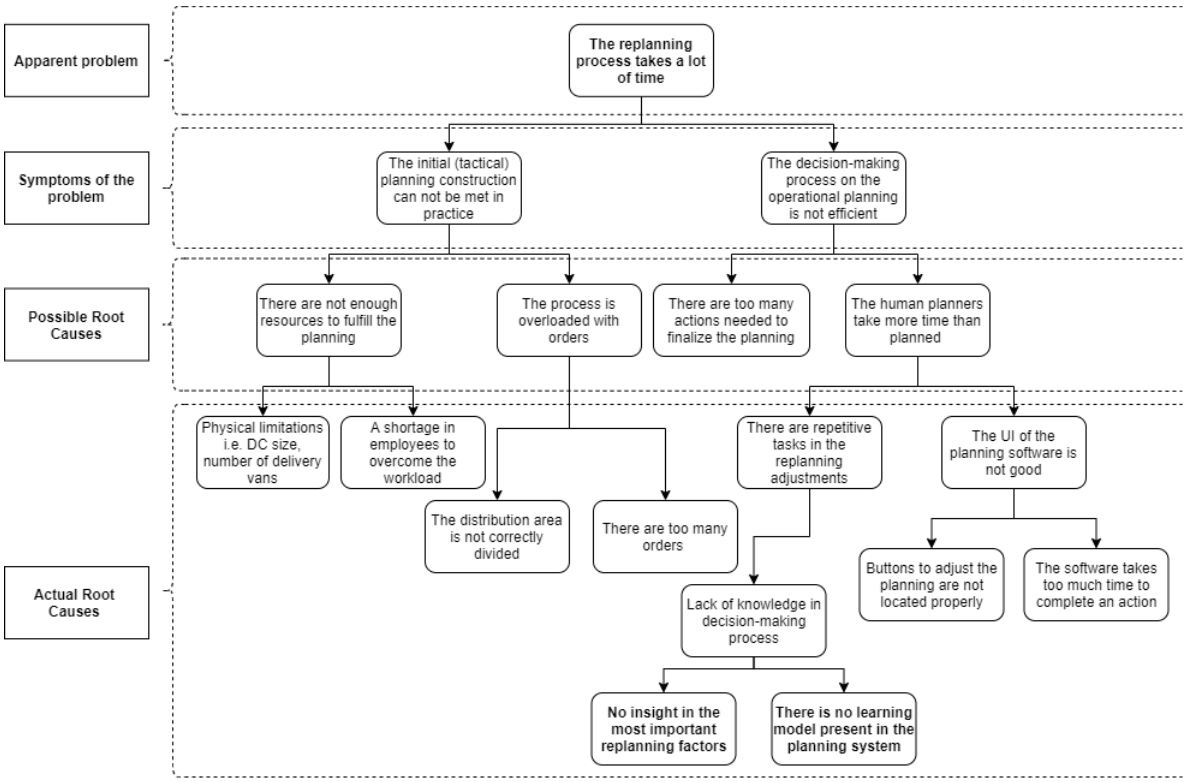



Figure 1.1: Root Cause Analysis diagram used for this research

The current system is operated by around three to four human planners, who need to finalize the operation planning by adjusting the input every day. These actions take time, because a lot of adjustments are made or even communicated last minute and each human planner has to adjust the planning manually. The business problem linked to this is that the total time the human planners take to finalize a planning is relatively high. A possible reason for this long duration of adjustments is the number of repetitive tasks based on the provided input and the new customer orders. The root causes associated to repetitive tasks is two-fold: There is no insight to the most important factors that determine replanning actions. Second, there is existing method to learn from past decisions made by the human planners. Tackling these two root causes could provide more knowledge regarding the replanning behavior of the human planners. Creating a feedback mechanism in the OOMPD application that recognizes patterns in the replanning tasks, can decrease the total adjustments needed and by extent, the total replanning time. This would increase the planning efficiency of the human planners. It might even provide decision support on the tactical planning, by identifying defects or recurrent flaws earlier than normal.

This research is using the Design Science Research Methodology (DSRM), proposed by the research of Peffers et al. (2007). This serves as the framework of conducting research in the field of information systems, by creating successful model implementation that evaluates the business performance. The DSRM process consists of a series of steps which are illustrated in Figure 1.2.

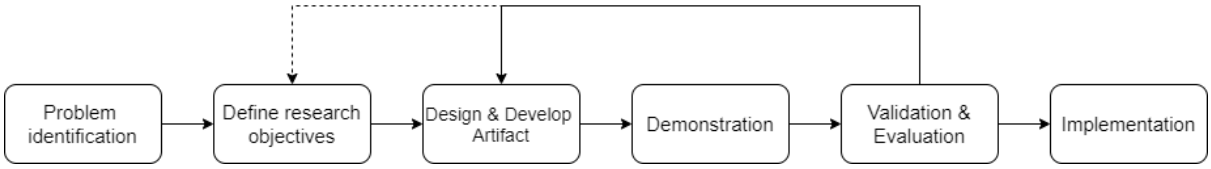



Figure 1.2: Design Science Research Methodology (DSRM) Process Model (Peffers et al. (2007)).

The model follows a nominal process, in which a priori knowledge from the problem is used to identify key objectives. The most important step in this process is the designing and development of the Artifact. An Artifact is the creation of an innovation within an existing process that contains new knowledge (Peffers et al. (2007)). Possible Artifacts are new models, methods, resource properties or designs. Relevant theory following from a literature study is used to design the Artifact, which is the basis for the solution design of this research. After demonstrating and validating the model at a case study from the Client, the DSRM process model creates two feedback mechanisms: a direct feedback on the Artifact to further improve and develop its design; and an indirect feedback loop on the research objectives, to test if all proposed objectives are solved or new objectives need to be defined. Our research takes the process model into account by introducing each step as a separate chapter in this thesis. Details regarding each chapter are further explained in the next section.

To aid the design of the Artifact, the first step of this research is to collect the right data that can be analyzed further by pattern recognition techniques. The input data (i.e., the tactical planning) for the OOMPD remains constant for the data analysis, so we focus solely on the human planner adjustments on the input data. We measure the total adjustments needed to finalize the daily operational planning in order to determine the quality of the input data. The human planners can make different types of adjustments on the operational planning and the UAR records and exports these adjustments to our

proposed learning models. This allows the study to focus on tuning the parameter settings of different algorithms to try and find the best fitting model for recognizing patterns in replanning. The reasoning behind which type of adjustments are affecting the realization, how these adjustments can be quantified and the collection and measuring procedures from the UAR are further described in Chapter 2.

The next step is to identify which pattern recognition techniques and learning algorithms fit the structure of the obtained data. Building learning models can be either online or offline. The main difference between both approaches is that offline learning takes all available, mostly historical data into account when configuring the algorithm and implementing the model afterwards. Online learning ingests data one observation at a time and utilizes learning steps to update the algorithm while it is implemented. For this research, the offline learning method seems more appropriate to use. We want to train a learning model locally, based on observed data from the UAR. The human planner can use the trained model to provide replanning suggestions on his current operational planning. There are different types of learning procedures available in the ML world. Choosing the right estimator and the fitting algorithm are one of the hardest part of solving learning problems, so an extensive literature review is conducted to find the best fit for this problem. This research provides the first steps to create the integration of offline learning with a real-life application.

## 1.3 Research Objective and questions

The previous sections clearly state both the motivation behind implementing a data-driven method in the current planning software and the identification of the core problem that needs to be solved. The goal of this research is then formulated as the following main research question:

*"How can we improve the decision-making in operational planning by classifying the most important features and patterns with a pattern recognition learning algorithm on actual replanning data¿'*

To find the solution to this research objective, several sub-research questions are defined to achieve a better understanding of different domains. The answers to these questions help finding the solution to the main objective more effectively and provide as a guidance for the remainder of this thesis. The research- and sub-questions are defined below.

First, all relevant information of this problem have to be gathered. This research is conducted as a specific case study at the Client, meaning that the current situation needs to be addressed specifically. The current software application and the type of human planner adjustments have to be investigated. These determine the performance metrics and have an effect on the input features for our methodology in the solution design. Chapter 2 will provide the relevant information regarding the following questions.

1. What are the critical components following from our context analysis?
    - What are the situation characteristics specific to this case at the Client?
    - How can we describe the current replanning process in the OOMPD application?
    - Which adjustments by the human planner are measured?
    - What types of data are available from installing the UAR?

After defining the context of the study and knowing which specific information is needed to solve the research objective, more knowledge about the problem solving method is needed. This information is gathered from a literature review and will provide the theoretical framework for the thesis, which can be found in Chapter 3. We will look more in-depth on the previous mentioned learning procedures and pattern recognition techniques. Also, previous research of applying learning methods on operational planning problems will be discussed and provide the foundation of our learning models. Finally, it is useful to find data analytic techniques that fit the input data correctly on the learning models and also to overcome challenges in imbalanced data.

2. What are the techniques present in literature related to pattern recognition, machine learning algorithms and logistics planning performance?

    - What type of pattern recognition or learning techniques are related to our research problem?
    - How can data analytics/mining be used to process the obtained data?
    - Which performance metrics are relevant for assessing the performance of logistic software applications?

When all relevant knowledge from literature is obtained and the context of the problem is defined, we focus our attention on creating the methodology for our solution design. This is done in Chapter 4, where we combine the known theories that best fit the practical context of this research. We need to know how the obtained data from the User Action Recording can be processed into input data for the learning models. Also, we need to define how to incorporate the models in order to assess their classification performance.

3. What are the key characteristics of our proposed solution design?

    - How can the operational planning performance be quantitatively measured?
    - Which data preprocessing steps are necessary to convert the raw data into useful input data?
    - How can the appropriate learning methods found in literature be integrated with the planning software
    - How can we validate the proposed learning algorithm with the OOMPD software?
    - How can we quantify the relevant performance metrics to test our solution design properly?

When a solution design and methodology is defined, this research proposes a model to test different learning techniques in order to improve the performance metrics of the OOMPD. For this, an optimization algorithm is proposed and then trained on the obtained data. After that, the model will be tested and validated by implementing the model on new input data. The results of the experiment testing can be found in Chapter 5.

4. Which machine learning algorithms can be used to find the best prediction results and how does this effect the planning performance?

    - What type of learning problem best describes this research objective?
    - Which hyper-parameter settings provide the highest classification accuracy?
    - Which set of features are the most important for the replanning process?

The solution method that this thesis proposes is applied and validated at one single customer, at one single DC. From an academic point of view, this research implements a ML algorithm in a real-life solution, to increase the planning performance and to improve te decision-making process. Also, it would greatly benefit CAPE Groep and their future business, if the implementation can be made universal. To meet this, a generic solution or methodology and the series of steps for its implementation will be discussed in Chapter 6. The learning algorithm or the decision support system can then be implemented at other DCs and potentially also by other companies.

5. What are the main contributions of our research to the relevant scientific fields and practical solutions?

    - How can this research help CAPE Groep by making the solution design more generic, so that the learning method can be implemented outside this study?
    - Are there any assumptions needed to fit our proposed methodology for future implementation?

To conclude the proposed research objective and questions, Table 1.1 provides a brief overview that lists the main methods and approaches that are used to answer each research question. This functions as a layout for the remainder of this thesis. The main chapters are also provided in the table and chapters answer one or multiple research question(s).

Table 1.1: An overview of the proposed Research Questions and applied Methods

| **Chapter** | **Research Question** | **Methods/approaches** |
|---|---|---|
| Introduction | - | DSRM, Research Objective and Questions |
| Context Analysis | 1 | Process and System description, Single-case study, UAR |
| Literature Review | 2,3 | DSS, Pattern Recognition, AI, Data Analytics |
| Solution Design | 3,4 | Methodology, learning models, performance metrics, building the Artifact |
| Experimental Results | 3,4 | Model evaluation, hyper-parameter tuning, Validation |
| Implementation | 5 | Creating the DSS, implementation steps |
| Conclusions | Main Question | Conclusion, Discussion and Recommendations |

## 1.4 Scope of the study

Before the research is conducted, the boundaries and size of the study has to be defined. These are the chosen delimitations that clarify which particular data is analyzed and which areas will and will not be explored. The delimitations are described below. In the next chapter, we will go more in-depth into how these delimitations affect our research situation and case study.

1. **The data is derived from one DC**
   The Client has a lot of DCs, each having its own OOMPD software application tailored to the specific distribution size and environment. This research focuses on the data from only one DC, in order to make the data collection and analysis independent from other DCs. If multiple DCs are taken into account, the complexity of problem increases heavily, i.e., bias towards certain DCs, generalization problems due to extrapolation issues or environmental factors (i.e., certain DCs have a larger district than others).

2. **The tactical planning is fixed**
   The tactical planning that provides the input for the OOMPD will remain fixed. No changes on this tactical planning will be made, this research focuses solely on implementing learning algorithms on the obtained data from the human planner adjustments.

3. **Representative quality of input data**
   This research collects and analyses its data within a certain period of time. The product owners stated that the daily orders have increased substantially compared to last year, possibly due to the COVID-19 pandemic. This overshadows the representativeness of the obtained planning adjustment data, since more daily orders result in more possibilities of planning adjustments. This can influence the strength of our found patterns and replanning factors.

4. **Fixed parameter settings**
   Other data that is available in the OOMPD software are the parameter settings (the constraints) of the distribution planning, like the total number of available delivery vans or the maximum weight of parcels per delivery van. These constraints have fixed values which represent the feasible range in which a delivery is possible. Since these values are based on the real-life physical limitations, i.e. the size of the van (in cubic meters) and the total number of vans per DC, they cannot be altered and are therefore left out in this research.

# Chapter 2

# Context analysis

In this chapter, we introduce the situation and elements from the particular case in which this research is held. To describe the current situation and its context, a critical analysis of all known components is conducted. The current process and system descriptions will provide a more detailed demarcation of the core problem. Section 2.1 provides the background information of the Client. In Section 2.2, the configuration and practical use of the OOMPD software will be explained. Finally, in Section 2.3, the configuration of the used data collection method and the types of replanning adjustments are described.

## 2.1 The Client

This research focuses on a practical case at one of the customers of CAPE Groep. Before the case specifications are described, background information of the Client is given because this provides the practical context in which the research is conducted.

### 2.1.1 Background

The Client is one of the largest Dutch logistics service provider in parcels and mail logistics. As a Logistics Service Provider (LSP) they do not only transport goods, but also store them and provide the logistic service for an entire value chain. Based on their position and theory, the Client plays the most important role as the fourth-party logistics provider (4PL) in the logistics supply chain (Ghiani et al. (2004)). This means that the company is not only responsible for the operation logistics and all its related processes, like warehousing and transportation, but also makes its own strategic vision and tries to continuously innovate its business activities. The Client is a leading 4PL in its industry and is continuously searching to improve their position even further. One of the key drivers for the Client is to deliver smart logistics solutions and lead through business model innovations. The aim of this research is to aid the Client into achieving this goal.

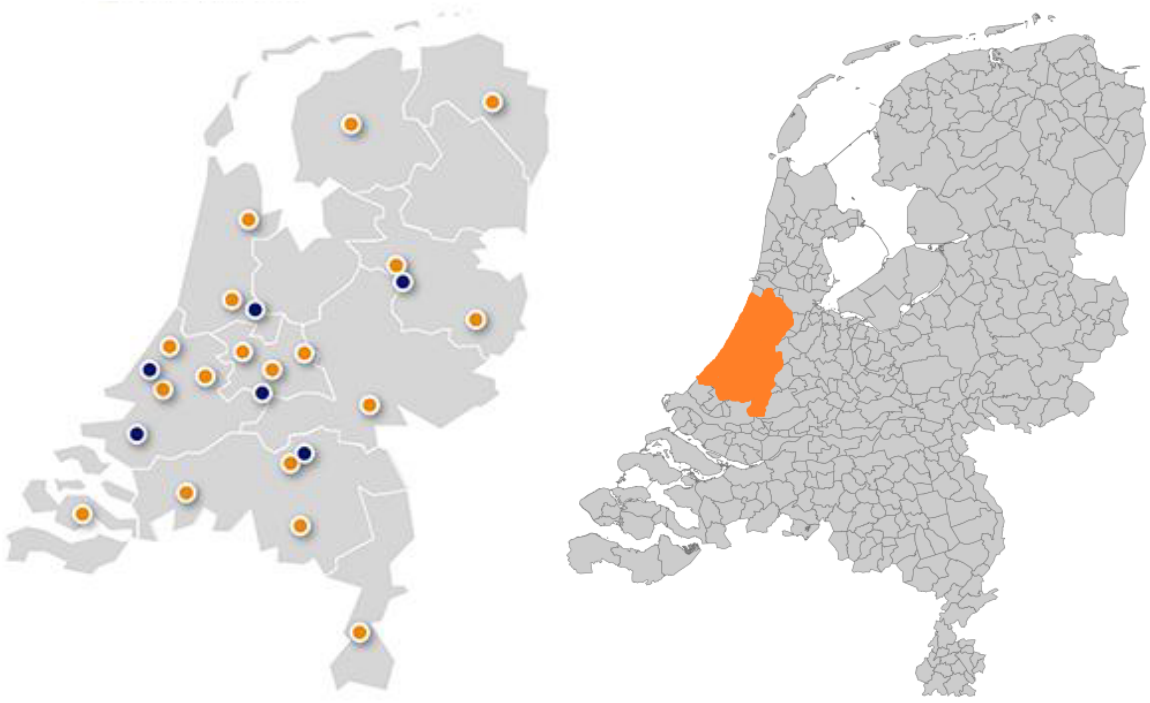
The entire logistic supply chain of the Client delivers more than a million parcels every day to customers all over the world. The operations of the Client are divided into multiple distribution channels, which are based on a global scale. Both on the domestic and international scale (the Benelux), the Clients uses the OOMPD application, which is configured and implemented tailored to each DC. This means that for each of the 34 DCs in the Benelux, the software is configured specifically to its characteristics. This includes the total distribution area, the number of available delivery vans, the size of the DC and

more. This research focuses on developing and applying a learning method and decision support system for a particular case at the Client, which will be introduced in the next section.

### 2.1.2 The case

As mentioned in the scope of the study, we focus our research on one DC of the Client. This makes our research a single case study, which has some advantages. According to Ridder (2017), the detailed description and analysis of the contextual conditions needs to provide a better understanding of a so-called "black-box". This black-box describes the underlying reasoning that is present in the environmental context, but is still intangible and hard to understand for management. The single case study can really zoom in on the "how" and "why" of this underlying process, because we can focus on one solution strategy resulting from our research Flick (2009). The patterns and insights found from this study can then be used to setup a cross-study analysis, by applying the solution strategy on multiple DC's in the future.

The DC is chosen carefully, because it needs to represent the real-life scenario as well as possible. For example, choosing a smaller DC results in a smaller distribution area and fewer shifts that need to be finalized in the operational planning. The input for the DC is therefore affected which could influence the total number of adjustments and therefore the quality of the obtained data. Also, choosing a DC that is relatively large, will have more human planners that operate the OOMPD software. As a result, a large DC in the Netherlands is chosen. All current DCs locations of the Client can be found in Figure 2.1a and the distribution are of the chosen DC is visualized in Figure 2.1b.



(a) DC locations in the Netherlands.

(b) The distribution area of the chosen DC

Figure 2.1: Graphical representations of the Client's operating area and scope of this research

The chosen DC is the largest DC in the province of South-Holland. The DC has a distribution area

consisting of municipalities in several provinces, which can be found specifically in Figure 2.1b. The DC delivers approximately 60.000 parcels each day, which are delivered to around in 375 separate delivery vans. The total number of parcels differs per day due to the known trend in demand; from Monday till Wednesday, there are more parcels that need to be delivered than the period Thursday till Saturday. At this DC, a total of four employees work in the planning team. Every day, the human planners obtain new information regarding the orders of that day and adjust the input planning accordingly in the OOMPD application to improve its construction. We will now go more in-depth in this relation between the tactical and operational planning.

### 2.1.3 Relation between operational and tactical planning

Complementing the information specific to this case study, we will explain the relation between the input planning and the operational planning. This relation describes how the OOMPD application obtains its initial planning and why the human planners need to make adjustments. Figure 2.2 visualizes this relation and we explain each element in the remainder of this section.

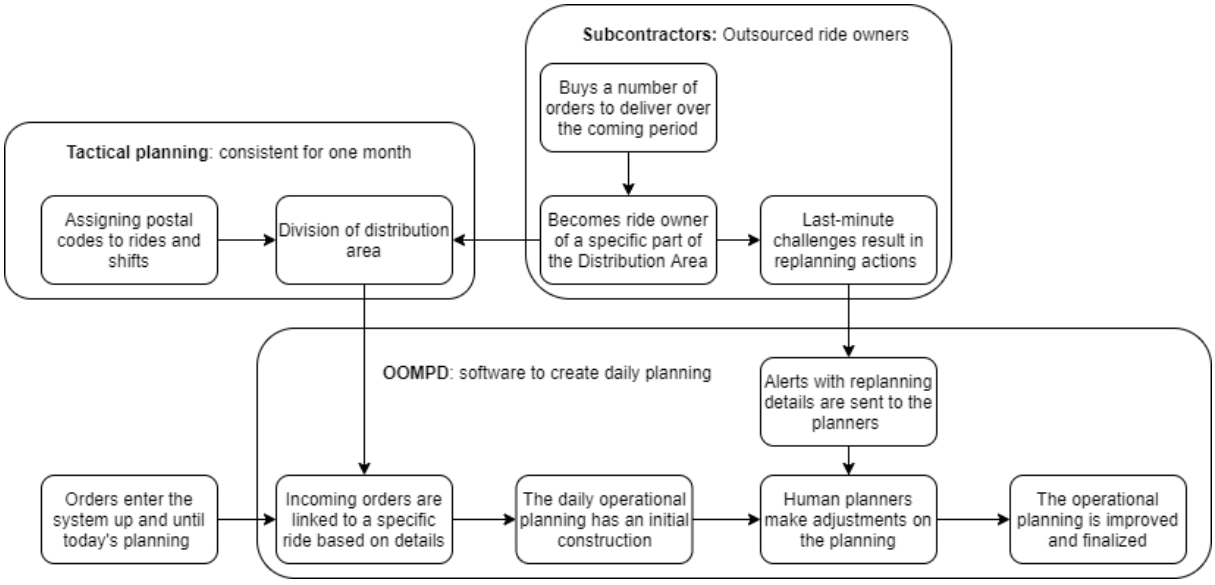



Figure 2.2: Visualization of the relation between the operational and tactical planning

There are three main actors/components within the operational and tactical planning relation. First, there is the tactical planning in itself. Discussed earlier, this planning functions as the input of the OOMPD planning software by creating the distribution area based on the division of rides and shifts. This distribution area is fixed for the upcoming month and is also important for the distinction of ride owners that are responsible for delivering the orders. There are two main types of ride owners: internal delivery vans (owned by the Client) and outsourced delivery vans (owned by subcontractors). The subcontractors are the second main actor/component and they represent all external companies that are used to deliver all orders of the planning. By buying a certain amount of orders, each subcontractor makes an agreement with the Client and becomes responsible for a small portion of the distribution area. In practice, the subcontractors encounter daily challenges that affect their delivery agreements. When this happens, the subcontractor manually sends an alert to the human planners with the necessary

replanning details.

Both the tactical planning and the subcontractor replanning alerts are important input elements for the final actor/component: the OOMPD planning application. Due to the division of distribution area, incoming orders that enter the planning application are automatically linked to a specific ride (due to their order details). At the start of the day, the human planners have an initial constructed operational planning due to the framework of the tactical planning and the incoming orders in the system. During their work shift, the human planners make adjustments on this initial planning to meet today's requirements. These requirements are based on the numerous alerts received from their subcontractors and the inspection of overloaded rides due to the capacity limitations. Mentioned earlier, the alerts are based on last-minute challenges that the subcontractors encounter and all information is sent manually to the human planners. Inside the OOMPD application, the human planners then manually adjust the specific orders of the initial planning to their new location (i.e., the new ride or shift). The planning is continuously improved to meet these last-minute challenges and when all adjustments are conducted, the operational planning is finalized. This gives an overview of the operational and tactical planning relation and how actors like the subcontractors are involved in this process. We will now go more in-depth in the replanning process inside the OOMPD application.

## 2.2 The OOMPD software

This section provides a short explanation of the OOMPD software application. We first explain the used terminology for different elements of the OOMPD, which are relevant for the remainder of the research. After this, we describe more in detail the current replanning process.

### 2.2.1 Terminology

The software that is used for the Client's operational logistics planning is tailored to their routing schemes and terminology. This terminology may differ from the explanations in scientific papers or journals, which can cause confusion. Therefore, a clear explanation of the relevant terms that are used in the OOMPD is provided below. There are two main data elements that the OOMPD takes into account: parameters and constraints. Parameters describe certain aspects about the order characteristics or the dimensions of the DC. Constraints are the limitations or capacities associated to the parameters. This differs from the formulation in i.e., a Linear Programming model, so we will define them in more detail for this research. As an example, the human planners can plan multiple deliveries for a certain ride. Each order has some parameter values like volume or weight. The human planner can plan orders on a ride until the delivery van reaches its limits (i.e., the volume or weight limit constraints). Below this section, we describe the most important parameters and constraints that the human planners take into account. The remainder of this thesis uses these descriptions to keep the terminology clear and information following from the literature review will be expressed in the same terms.

#### Parameters

**Address:** An order is linked to a stop address at which the order needs to be delivered. The address consists of the street name and number. Each postal code block consists of unique set of addresses.
**Channel:** The OOMPD is used for various distribution channels. These are the different sectors and

time periods in which the Client operates their business. This research focuses on only one sector, the Home Distribution Channel. This channel distributes all parcels that are ordered from individuals or private companies. All orders within this channel are planning during the day (except Sunday), from 05:00 until 16:00.

**Day:** The day of the week in which the operational planning is made. The day is represented with a number ranging from 1 (Monday) to 6 (Saturday).

**Input:** The input for the OOMPD is the provided tactical planning from an external database. This static planning consists of the division of all postal code blocks over the possible rides. This division allows the system to automatically link new orders on a certain ride. The input is made at the beginning of a month and stays consistent for the remainder of the month.

**Order:** An entity that holds all details of one delivery within a ride. The order can contain multiple parcels and a certain destination can have multiple orders.

**Parcel:** The total number of products a Order can have is equal to the number of parcels. Each parcel has a weight and a volume, that influence the total amount of products each delivery van can transport. For example, a parcel weights 0.2 kg and has a volume of 0.1 m3.

**Postal code block:** The postal code is a combination of four numbers and two letters that defines the graphical location of the order. A postal code can have multiple orders and the postal code block consists of a various postal codes. For example, a postal code block consists of 100 postal codes. One of these postal codes has 3 orders, two have only one parcel and the third has 2 parcels.

**Quantity:** The number of parcels that are linked to one order. For example, The quantity of parcels from one order at one postal code is 5.

**Ride Number:** The ride number is a numerical value that describes the specific route a delivery van has to ride on the associated shift. The ride number represents a neighborhood in the shift, which consists of a subset of postal codes from the associated shift.

**Ride Name:** Complementing the ride number, each ride has a ride name that represents the town or city. A ride name can consist of various postal codes, for example, ride "0248 City 1" consists of various postal codes ranging from 2801BB to 2801ZZ.

**Ride Owner:** Each ride has a specific owner, which is responsible for the actual delivery of the orders linked to the ride. There are two possible ride owners in the system: either the Client's internal employee or a subcontractor.

**Shift:** Each day contains of 12 shifts. These are moments in time when all delivery vans leave the DC and deliver their respective orders. Also, each shift consists of an unique set of postal code blocks and contains all planned trips within one hour period. For example, shift 3 has 100 postal code blocks. These postal code blocks are only scheduled in shift 3.

### Constraints

**Loading Bay Capacity:** The DC has a limited number of available loading bays. The amount of orders that can be handled.

**Max stops:** During each shift, a delivery van has a maximum number of stops that it can make. This is due to the available time each shift takes and in which all parcels have to be delivered. On each ride, a delivery van can stop a maximum of 160 times.

**Max pieces:** The van is also limited on the total number of parcels it can take. This limitation is set due to the working hours of a ride owner. For each of the vans, the maximum amount of parcels is set

to 200.

**Max volume:** Besides the quantity, the volume of the parcels is also a loading capacity. The size of the vans limits the total volume of the parcels that can be scheduled on a shift. Parcels can have different sizes and volumes, which affect the total number of high volume parcels that can be loaded in one delivery van. The delivery vans in our research all have the same volume, which is set to 6 cubic meters.

**Max weight:** The final limitation on the delivery vans is the maximum weight of the parcels. Besides the quantity and the volume, the van can only take as much parcels onboard as the physical load can manage. The weight limit of the vans is 1000 kilograms.

### 2.2.2 Current replanning process

In the daily planning operations, the Client uses the OOMPD to adjust the input planning into feasible, operational plannings. Alerts provide the necessary details regarding which specific order has to be replanned and the human planners take certain constraints into account to make the adjustment possible. These constraints are associated with the physical limitations of a delivery van and they provide the boundaries in which the human planners must finalize their feasible, operational plannings.

With the parameter and constraint terminology defined and the relation between the operational and tactical planning acknowledged, we now briefly discuss how these two elements are used in the OOMPD application. This will describe the current replanning process. In Figure 2.3, the main interface of the OOMPD software is visualized. There are four main parts highlighted within the User Interaction window (visualized with the light blue boxes), which represent the replanning process of the Client. The numbering of the parts serve as the sequential steps that the human planners follow in order to make adjustments in the planning. We will briefly explain each step below.

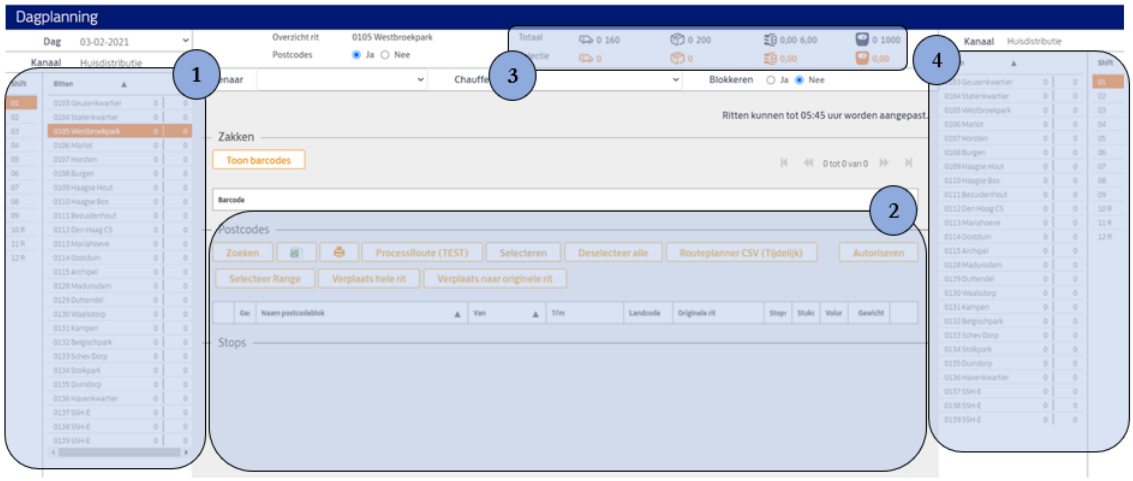



Figure 2.3: The User Interaction window of the OOMPD application including the marking of the four, sequential replanning steps

**Step one**

The first step shows two lists: the shifts and the rides. Basically, the entire input planning and the division of all Orders over the shifts and rides can be found in this first step. On each day, the parcels are planned over a total of nine shifts throughout the day, with three reserve shifts being available at the end of the day. Each shift contains all planned parcels and routing information of a one-hour period. Any replanning regarding a shift can be done up until 15 minutes before the start of the shift. For example, shift 4 starts at 11:00 am. Before this shift starts, it is possible to make adjustments on the planning until 10:45. After that, the shift is finalized and the planning can not be adjusted anymore. This is because the remaining fifteen minutes are used to connect the adjustments in the planning with the physical systems in the DC.

**Step two**

Each ride consists of a table of information that represent all the details of each postal code block of the delivery van. This contains the number of stops per postal code block, the quantity and volume of all orders and all the addresses of each stop within the ride. This information is represented in the second step in Figure 2.3. When the human planner wants to make an adjustment on the planning, the planner first needs to find the specific set of orders that needs to be replanned. Deciding which set of orders is based on the provided replanning details from the subcontractors. After clicking the right shift and the ride subsequently in step one, the adjustment can be made in the details of the ride, which is done in step two. In here, the human planner can select which part of the order needs to be replanned (based on the information in the alert). It is also possible that adjustments are made on the rides of which the Client is the ride owner. In this case, the human planners obtain the same kind of replanning details via an alert directly from own employees.

**Step three**

Each adjustment that is being replanned contains certain parameter information i.e. the order details. To make sure that the adjustments are viable, the selected adjustments need to fit in the constraints of their new planning destination (the new ride or shift). These constraints are represented in the third step of the OOMPD application. These constraints represent the physical limitations associated to both an order and a ride. For example, the volume of all parcels that are being replanned can not exceeds the remaining volume in the available delivery van. Also, the delivery van can make a limited number of stops during the ride. When more Orders are replanned on already a busy ride, the number of stops are a big limitation. This is where the previously mentioned reserve shifts and rides are a good option, since these are a build-in option with zero orders planned (and therefore, zero stops) beforehand.

**Step four**

When all details of the adjustments are correctly indicated and fit the constraints, the replanning process is finalized in the final part: step four. This section has the same layout and interface as step one; a list of all shifts and rides. The human planner selects the new designation of the order details by assigning the adjustments from step two to a new shift and/or ride. The human planner takes the capacity limitations of step three into account when the adjustment is carried through. When all human planner adjustments are conducted, the resulting division of orders in step four represent the finalized operational planning.

Due to the time windows of a shift, the entire daily replanning process is divided in multiple sections. For instance, human planners obtain the alerts all throughout the day and due to the structure of shifts, some alerts are processed earlier then others. Also, shifts that depart early during the day need to replanned first and adjustments from later shifts cannot be replanned back to earlier shifts, because they already left the DC. The human planners know these time constraints when making the adjustments, so these decisions will be represented in the collected data.

To conclude this section, we discussed the four sequential steps that the human planners follow in order to adjust the planning. This research focuses on finding recurrent patterns in this replanning process. We introduce several innovations that will aid this, one of which is the implementation of a User Action Recording (UAR) mechanism. This will record the human planner adjustments in the OOMPD application and create the raw data used for the learning models. The best learning model is then implemented as the second innovation, the artifact (based in the DSRM methodology). The remainder of this chapter will describe how the UAR is configured and which types of human planner adjustment data is collected.

## 2.3 User Action Recording

This section describes the main data collection method used in this research: the User Action Recording (UAR). This is a software-interface recording tool based on a hard-coded Javascript. We describe the configuration of the tool and how it complements the OOMPD software of the Client. Following from this, we describe the types of adjustments that are able to be measured and indicate which type is used in the remainder of this study.

### 2.3.1 Configuration

Mentioned previously, the human planners receive an alert from their subcontractors if a replanning adjustment needs to be made. These alerts contain the relevant information to make adjustments on the planning, which the human planners modify in the OOMPD application. All data within the application is processed via microflows. The UAR is able to measure data attributes that are processed within in a microflow. Before we describe the configuration of the UAR, we briefly describe how microflows are used in Mendix applications and how data is managed within these microflows.

Microflows are a visual way of expressing a textual program code, which is a main low-code feature in Mendix. It is used to express logic processes by performing actions on objects and can be used for any application and its goal (Mendix Technology (2021)). Examples of practical uses of microflows are opening new accounts in a customer system, creating a new booking form on a hotel website or moving/storing data on a cloud page. Every microflow is composed of several elements. We will explain these elements with a small, practical example, which can be found in Figure 2.4. This figure shows a visualization of a microflow that checks if two passwords are equal when the user wants to login on an application (i.e., on the Client's OOMPD). With this example, we explain the different elements within a microflow.

There are four main elements that can be found in Figure 2.4. These represent the different actions that are possible within a microflow and how data attributes are transferred throughout the logical flow. We briefly explain the elements to give a small background on the basics of a microflow, since these are also

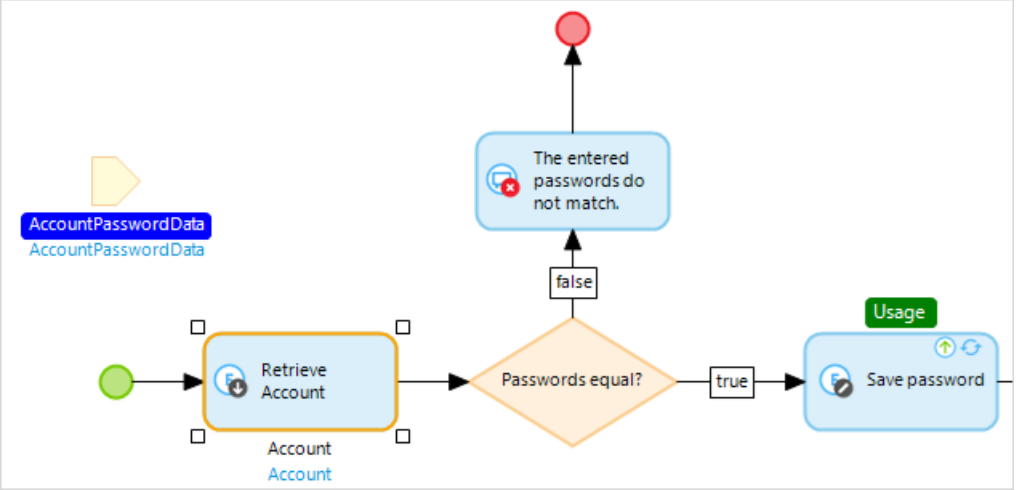



Figure 2.4: An example of the logic process within a Mendix Microflow.

part of our data collection method in the OOMPD application.

- **Event**: This element represent the start and endpoints of a microflow or special events that iterates a loop. A microflow has only one *start Event* (green circle) and at least one *endpoint Event* (red circle). *Iteration loops* consist of a *Continue and Break Event* that check whether the loop needs to stop or not. In the provided example, there is no iteration loop so this microflow is iterated only once.

- **Flows**: These form the connections between different elements. The flows are represented with the arrows and there are two types: *Sequence Flow* (filled arrow) and *Annotation Flows* (dashed arrows). The first is used to links events, decisions and activities and thereby defines the order of execution within the microflow. The latter is a connection between an annotation to another element. The provided example demonstrates the presence of *Sequence Flows*.

- **Decisions**: Decisions deal with making choices and merging different paths. This element is represented with a colored diamond in the example and there are three decision types possible. The *Normal Decisions* makes a choice based on a certain condition which result in one and only one possible outgoing flow. A *Object Type Decision* is an element that makes a choice based on the specialization of the selected objects. Finally, the *Merge Decision* can be used to combine multiple sequence flows into one based on a choice. The example illustrates a *Normal Decision*.

- **Activities**: This element are the actions that are executed within a microflow. The blue squares in Figure 2.4 represent the activities. These define the main purpose of the microflow and there are a lot of activity types available. Examples of these activities are creating or retrieving new objects, create new lists of multiple objects, change a variable value or import/export external functions.

With the basic description of a microflow now known, we will move to the configuration of the User Action Recording. The UAR is a coded Javascript file that can extract attribute values from data elements that are parsed through Microflows. To integrate this with the OOMPD application, the coded file can be imported as an external widget. Inside the OOMPD application, we define which data attributes and variables we want to measure by creating a whole new data entity. This data object will then be coded

within the Javascript file of the UAR, so it knows which data object to fill with recorded data. In order to record the right data, the UAR widget is being linked to the replanning User Interface, previously seen in Figure 2.3. It is installed on the background of the application, so when a human planner is making an adjustment on the planning, the UAR widget runs and measures the corresponding data from the adjustment simultaneously. A part of the Javascript file can be found in the Appendix (Appendix A.1).

Our research focuses on data from two months. The data is collected and extracted every day within this period and this process is illustrated in Figure 2.5.

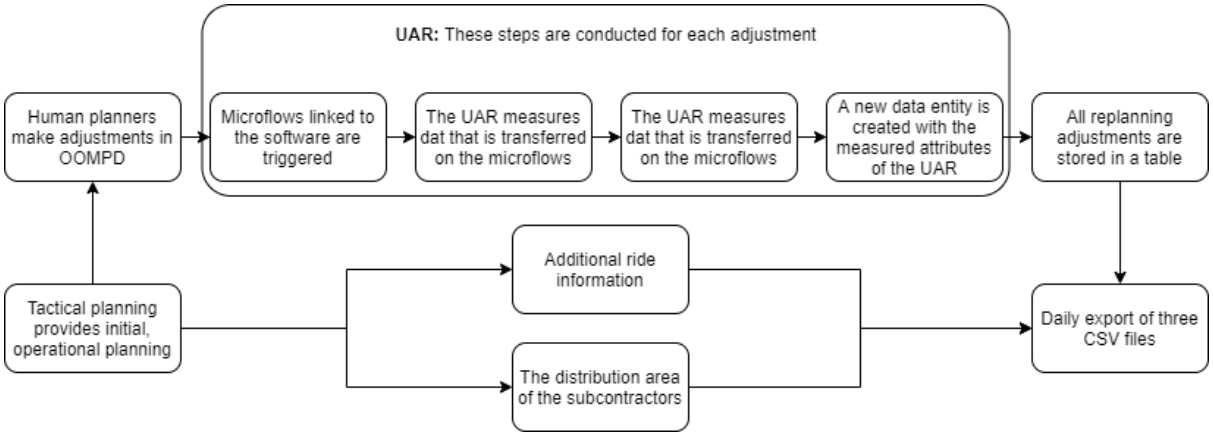



Figure 2.5: The configuration of the UAR and data collection steps used in our research.

Besides the framework that the tactical planning provides for the OOMPD, it also provides additional information that is used for our research. This consists of two separate files: additional ride information (like the volume and number of stops) of a ride and the division of the subcontractors over the distribution area. The first file contains the initial parameter values of each ride, which will be used as additional features for the learning models. The second file contains the information of which subcontractor is responsible for each ride. This daily information describes the ride owners associated to each ride of that day. When the human planners make an adjustment on the planning within the OOMPD application, there are microflows triggered to convert the order information. The UAR is then called to measure the data attributes of these microflows and creates a new object with these attributes. This process is repeated every time a human planner is making an adjustment, so each adjustment results in one data object. At the end of the day, the UAR data and the additional files are is extracted. In Chapter 4.1, we will come back to this process and discuss how the extracted data is converted into input data for our research.

### 2.3.2 Types of adjustments

The human planners receives daily alerts that provide the replanning details. All the adjustments that are made in the application can be categorized in a number of types. These replanning types are based on certain parameters within OOMPD and Figure 2.6 provides an overview how these replanning parameters can be combined and represent different scenarios. We briefly explain the possible types of replanning scenarios and indicate which type will be focused on during this research.

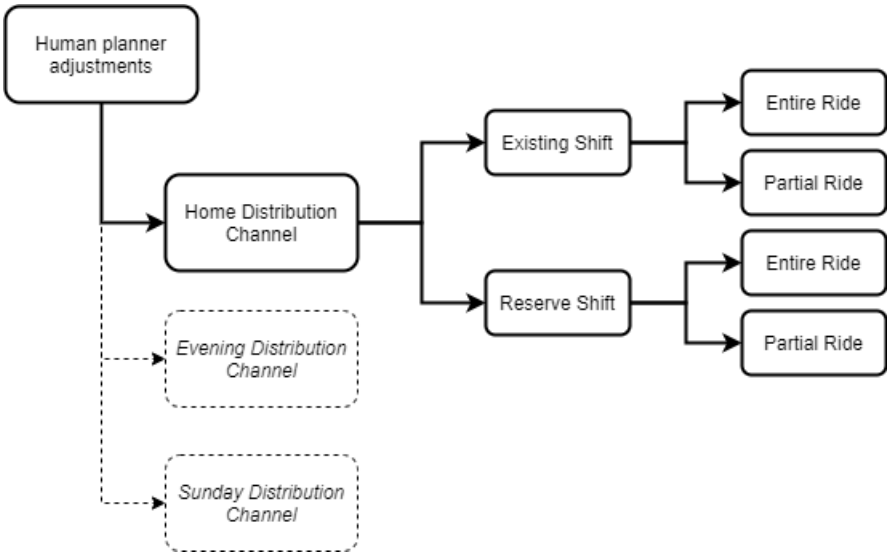



Figure 2.6: Visualization of the possible replanning adjustments in the planning software

**Channel adjustment** The OOMPD application is used for three types of distribution channels at the Client. These are the home delivery, the evening delivery and the sunday delivery. Mentioned previously in the scope of the research, we focus on only the operational plannings of the Home Distribution Channel. This channel includes all parcels that are delivered from Monday till Saturday, except the deliveries planned in the evening. It is not possible for the human planners to adjust orders from one channel to another, making this more our research.

**Shift adjustment** The second type of adjustment is possible on the shift parameter. Orders are preplanned on a specific shift, which are linked to a certain time. Mentioned earlier, the shifts have certain time windows in which they can be replanned. For example, all orders on shift 1 leave the DC first. When shift 2 starts, nothing can be replanned back on shift 1. It is possible that some orders can not be delivered on time, so they have to be replanned on a later shift. Following from this, there are two replanning scenarios: either the orders are replanned on a reserve shift or they are planned on an existing shift.

For the first scenario, there are three reserve shifts allocated for additional rides, which leave the DC at the end of the day. When a ride is created on one of the reserve Shifts, the ride owner needs to be assigned as well. From that point on, the specific ride can only be used for orders linked to the appointed ride owner.

The second scenario for the shift adjustments involves the replanning scenarios where orders are allocated to another, not-reserve shift. For example, certain orders from shift 3 are replanned on a Ride from one of the remaining shift options (shift 4 to shift 9). This replanning scenario is limited, since the framework of the tactical planning already fills these non-reserve shifts with incoming orders before the replanning process starts. These shifts and their rides will already be planned with orders, resulting in less replanning options for the human planners.

**Ride adjustment** It is also a possibility that the human planners make adjustments within the same shift, making the final type of adjustment on the ride level. If a ride is overloaded with the total

number of stops, some scheduled deliveries need to be replanned on other rides. These adjustments are made based on the postal code blocks within the ride, because these are associated to a certain ride owner (once again, based on the input data). There are two scenarios that can occur when human planners adjust on the ride parameter: either the entire ride or only a set or orders is replanned.

The first scenario involves the replanning of all orders from one ride to another. Based on the details within the incoming alerts, the human planner select the specific postal code block sections (and all scheduled rides on these codes) that need to be replanned. These are then allocated to another ride which is not overloaded based on the capacity constraints.

For the second scenario, only a part of all orders within one ride are replanned. A ride contains of different orders, each with their details (own address, number of parcels, etc.). It is possible that certain orders need to be replanned on other rides that are controlled by the same ride owner. The human planner then finds a different Ride within the same shift and ride owner, while making sure that the constraints of this new ride are not exceeded. To provide a better understanding to which specific replanning scenarios we focus our research, we highlighted all types of adjustments under the Home Distribution Channel. The replanning scenarios following from the other two distribution channels (marked with dashed lines) are left out of this research.

## 2.4 Conclusion

In this Chapter, we described the current situation in which this research will be conducted. First, we described the background information of the Client and the characteristics of our Case. Then, we illustrated the relation between the tactical and operational planning and how these components, combined with the subcontractors, are related in the replanning process. In short, the tactical planning provides a framework in which daily orders are initially planned. At the start of the day, the human planners have an initial operational planning and based on incoming alerts and capacity constraints, adjustments are made on this planning. We discussed an overview of how the OOMPD software application is used by the human planners and how adjustments are made in the replanning process. Finally, we introduced the data collection method, the UAR, by describing its configuration on the planning application and how the method can store replanning data. Finally, we discussed the which types of adjustments in the application are relevant for our study and the configuration

# Chapter 3

# Literature review

In this chapter, we provide an overview of the available literature regarding several main scientific fields that are used in this thesis. For each scientific field, the most applicable methods will be chosen based on the literature. Section 3.1 will discuss the practices of Decision Support Systems (DSS) and how it can be used in the process between operational and tactical logistics. It also provides the potentials and possibilities of AI and learning algorithms in logistics planning. Following this, the theoretical framework for pattern recognition techniques will be investigated. Section 3.2 discusses several important methods in the field of AI, which can be used for pattern recognition and classification problems. In Section 3.3, we discuss known challenges in the field of data analytics/mining of processing raw data into viable input data. Each Section will end with a conclusion that will summarize the key findings of the discussed literature.

## 3.1 Decision Support and Processes

In this section, the relation between operational and tactical planning and the consequences of the decision-making will be discussed. Also, we introduce the concepts of Decision Support Systems (DSS) to aid this relation and address a knowledge gap. Finally, the lack of learning models in current operational planning is discussed which will be the foundation of the potential and the use of artificial intelligence in order to make logistics more data-driven.

### 3.1.1 The process between Operational and Tactical Logistics

Our research focuses not only on the identification of planning improvements, but also on the creation of lasting change by introducing a method or framework that complements the current planning system. The stimulation of switching the operational-tactical process towards decision support and data-driven automation is mentioned by Mes and van Heeswijk (2020). This provides the human planners with the opportunity to learn about the challenges in logistic planning and to gain experience with new technologies. The use of a Decision Support System (DSS) can influence this transition greatly and is not yet covered in any research. Overcoming this gap in the literature will help us understand the right knowledge for a data-driven method for planning logistics. According to Mes and van Heeswijk (2020), the main drivers behind technology acceptance are the effect of the perceived usefulness and ease of use of a new system. To complement this, we introduce different concepts and practices of DSSs in literature.

This will help us to overcome the acceptance challenges of introducing new technologies into existing systems. Also, it provides us with a better understanding of both the process between operational and tactical planning and the communication between the two levels of decision-making.

### 3.1.2 Decision Support Systems

Applying a DSS can overcome the barrier between data-driven logistics and operational planning systems. The original DSS methodology is defined by research of Gorry and Scott-Morton (1971), which combined multiple theories together into a structured framework. The DSS is defined as a computer system that deals with at least one problem at an organizational level and should aid the judgment of the decision-maker. As a result, the DSS is constituted as a human and machine decision-making system. This framework takes three elements into account: *intelligence*, which describes the search steps needed to correctly define business problems; *design*, which involves the development of alternatives; and *choice*, which consists of analyzing the alternatives and choosing the implementation (Shim et al. (2002)).

Zak (2010) introduces the implementation of DSS in the transportation world. They state that in transportation, the DSS is used by human planners and provides different functionalities and helps solving various decision problems. A critical finding in this research is that the model-base is a crucial element for transportation-DSS. Model-base consists of a structured framework with analytical tools, modeling techniques and problem solving methods. This allows for a wide range of efficient methods that can be used, most interesting is the use of AI and Operations Research techniques. Zak (2010) provides a great visualization of different transportation-DSS characteristics, which can be found in Figure 3.1.

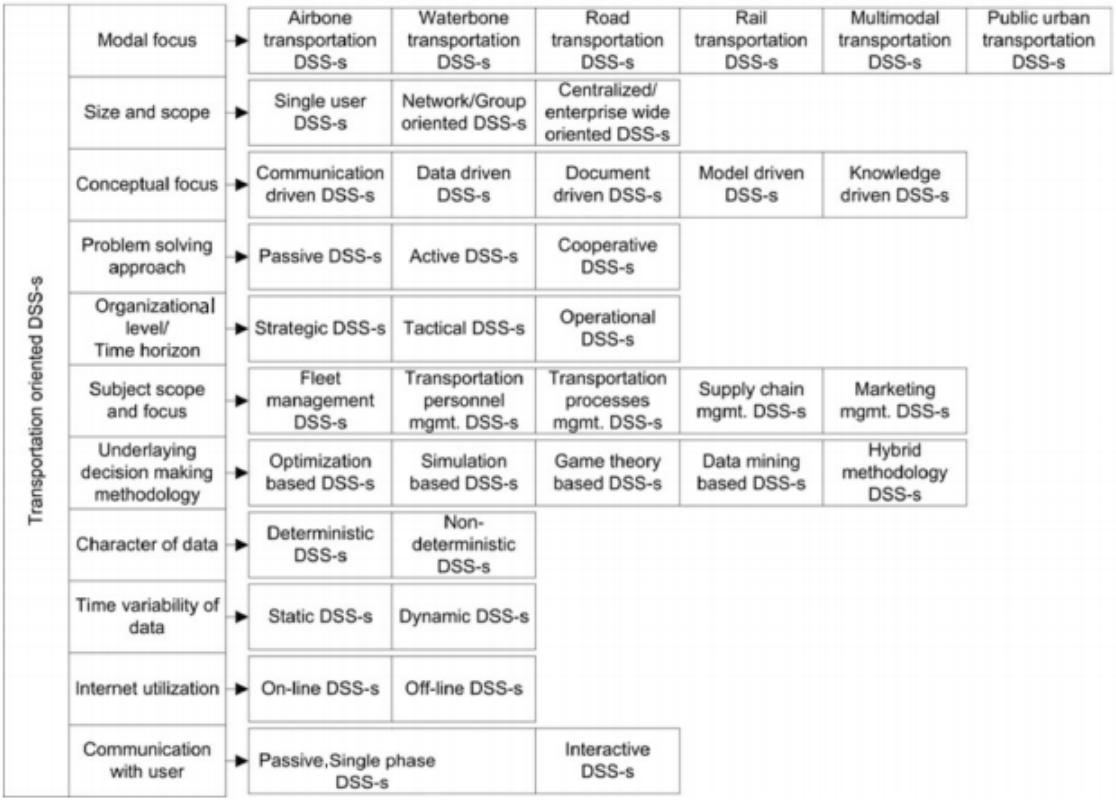



Figure 3.1: The classification of Transportation-DSS (Zak (2010)).

What is also interesting to acknowledge is the conceptual focus of data-driven DSS. The first framework for this type of DSS is defined by Power (2008). Factors that need to be present in a well-designed data-driven DSS are the integration of effective data management, consistent high-quality analysis methods and better informed decisions. Data-driven DSS also relies on the feature "Alerts and Triggers". Very often in transportation and logistic systems, encountered disruptions prevent current operations to run as planned. Most of the time, the planning of logistics relies heavily on human planners, instead of planning algorithms and real-time data. There are multiple reasons why manual planning is preferred for logistical companies. Because the decision process between operational and tactical can be seen as a communication *black box* according to management, having manual planning can be more sensible (Samek et al. (2019)). Also, having algorithmic expertise and experience with sophisticated DSS is a key determinant for successful implementation of new technologies, but it is often limited inside the company (Venkatesh and Bala (2008)). We referred to this at the end of Section 3.1.1 as well, where an integration could help clarify this operational and tactical level *black box*. The framework of the data-driven DSS could provide the necessary elements to overcome the challenges of decisions-making between the two layers of planning.

### 3.1.3 Defining the potential of Artificial Intelligence in Decision Process

According to Zak (2010), the next phenomenon for data-driven transportation-DSS is the implementation of artificial intelligence tools and optimization heuristics to make the systems more "self-educated". The system becomes an Intelligent Transportation-oriented Decision Support System, which uses historical data and encountered facts to generate certain decision rules, interpret patterns and draw certain conclusions that advice the decision-makers. This type of decision support can be combined with expert systems, which allow accumulated knowledge from experts in combination with AI techniques to support rational decision-making (Zak (2010)).

Intelligent systems based on the Artificial Intelligence (AI) methodology are becoming increasingly popular and mature in solving real-life problems; knowledge-based systems have favourable characteristics compared to conventional approaches or pure, symbolic AI systems (Choy et al. (2008)). Model management systems and knowledge-based decision support systems have used techniques from artificial intelligence and expert systems to provide smarter support for the decision-maker (Shim et al. (2002)).

The implementation of learning algorithms in decision support systems is still relatively new. Some recent studies have developed unique DSS tailored to a specific case study. We compare the used methods and key findings from different literature studies in Table 3.1. We discuss which of these methods are most relevant and how their key findings will be the foundation for further exploring the use of AI techniques like pattern recognition.

Table 3.1: Table with different decision-making methods and key findings.

| No. | Authors | Decision-Making Methods | Key Findings |
|---|---|---|---|
| 1 | Manzini and Bindi (2009) | This research proposes multiple strategic models and optimizes an operational planning multi-period Mixed-Integer Linear Programming model. The results show the effectiveness of the proposed model as the operational transportation costs of the vehicle routing is substantially lower based on strategic decisions. | The solution in this research more focused on strategic decisions, i.e. determining the locations of DCs. The study proposed an effective model, which need to be further researched in vehicle scheduling and planning decisions. |
| 2 | Mes and van Heeswijk (2020) | This paper introduces a logistics serious game of an anticipatory planning problem. It compares the decision-making process of manual human planners and automated planning algorithm based on Reinforcement Learning. The logistics service supplier makes daily decisions of assigning containers to trucks, barges and trains. | The model is based on a Markov Decision Process and both the heuristic and RL algorithm outperform the human decision makers. The paper states that besides human expertise, the algorithmic developments are necessary to advance the decision-making in anticipatory logistic planning. |
| 3 | Li et al. (2007) | The proposed DSS improves the scheduling and disruption decision-making process for a Single-Depot Vehicle Rescheduling Problem, an extension on the classic Vehicle Routing Problem. The DSS provides an optimal solution for sequential scenarios and provides an interactive environment to aid the decision-makers by modifying any possible routing solutions. | The proposed algorithm is based on a Mixed-Integer Linear Programming solver, and showed as an effective tool for real-time operational replanning. Also, the DSS was responded positively by the human planners, making it easier to accept the new technology implementation. |

Continued on next page

Table 3.1 – continued from previous page

| No. | Authors | Decision-Making Methods | Key Findings |
|---|---|---|---|
| 4 | Jansen et al. (2004) | The goal of this research is to lower the total costs of the operational planning of a large-scale multi-modal logistic transportation system. The used method is a branch-and-bound model and local heuristics are used to improve the planning objective. | The proposed system is tailored specifically to a logistic and transportation planning system. The mathematical model uses meta-heuristics (i.e. Simulated Annealing or Tabu Search) to improve their planning performance. |
| 5 | Ruiz et al. (2004) | The author proposes a DSS for a real-life vehicle routing problem. The model is a Mixed-Integer Linear Programming problem that functions to minimize the total travel distance. The DSS visualizes the solution process of an implicit enumeration algorithm that solves the Mixed-Integer Linear Programming. | The proposed two-phase approach could be used in similar transportation problems, if the routing selection would be possible. Also, time-windows are a great addition to improve the enumeration algorithm. |
| 6 | Irannezhad et al. (2020) | The main method of this paper is the introduction of an intelligent-DSS prototype for port logistics, which is solved via a dynamic vehicle allocation and routing problem with time-windows. An agent-based simulation algorithm is combined with RL to simulate the adaptive behaviour of the agents. | This research showed the potential of AI, more particularly the use of RL, in a self-learning DSS. The method showed to overcome intrinsic challenges related to adopting the new system and the logistic performance can be improved by making a trade-off between exploration and exploitation. |

Continued on next page

Table 3.1 – continued from previous page

| No. | Authors | Decision-Making Methods | Key Findings |
|---|---|---|---|
| 7 | Fanti et al. (2017) | The research proposes a DSS to aid decision makers that operate on the inter-modal, cooperative logistics paradigm. The decision-making process is based on multiple layers and the use of discrete-event simulation optimizes the activity flow of the transportation process. | The model can improve the transportation process on multiple KPIs and with the DSS, the decision-makers can focus on one of these goals. The proposed model can be improved with real-time information or forecasting models, which can be the basis for new learning algorithms within the DSS. |
| 8 | Ragupathi and Govindarajan (2020) | This goal of this research is to introduce a data-driven DSS to assess the performance of medical disease identification. The proposed methods are based on ML; three classifiers are used (RBF classifier, logistic regression and Naive Bayes classifier) | This research validated the performance of the three classifiers, all of which showed a high prediction accuracy. The use of a Multi-Layer Perceptron as a feed-forward artificial neural network (NN) proved to be a great fit for the medical-DSS. |
| 9 | Rabe and Dross (2016) | The goal of this research is to construct an architecture for logisticsDSS that relies on a Discrete-Event simulation model. Reinforcement Learning is adopted to simulate the performance improvement of the logistics network. | The application of the RL algorithm makes the system able to learn from experience and if more data becomes available, the use of value-function approximation can further improve the decision-making process. |
| 10 | Al Hajj Hassan et al. (2020) | This research proposes a RL framework for the freight demand forecasting to support operational planning. The forecasting method is based on product dimensions (seasonality, trends and clustering) and time dimensions (short-term and long-term). The latter is predicted with the use of RL. | Historical data is used to test the accuracy of the RL forecast. Extending the approach on other network decision i.e. route planning, could improve the forecasting accuracy. |

Continued on next page

Table 3.1 – continued from previous page

| No. | Authors | Decision-Making Methods | Key Findings |
|---|---|---|---|
| 11 | Kuter (2012) | This research discusses the main dimensions of Learning and Planning with intelligent systems. The most used ML methods in planning are *explanation-based learning* and *inductive learning*, but other approaches for automated planning are case-based learning and RL | The main methods are discussed in-depth and the characteristics/differences between each methods are clearly stated. This increases the foundation of an AI implementation substantially. |

### 3.1.4 Conclusion

Table 3.1 shows different methods to improve operational planning problems in various industries and we can compare these decision-making methods to the planning problem of our research. Most of the studies showed the presence of a DSS to aid the decision-making process. However, many of these studies are not related to the operational planning of 4PL systems. Most of these operational planning problems are related to the Vehicle Routing Problem, which can be mathematically optimized with i.e. a Mixed-Integer Linear Programming model (Li et al. (2007)). Also, an intelligent-DSS solution like the agent-based model of Irannezhad et al. (2020) is discussed, but since the agents in our problem are limited and the problem is not two-phased, this is most likely not a great model to use. However, the proposed learning algorithms in the field of AI like (inductive) machine learning, can provide the additional insights needed to recognize patterns in replanning behavior. Based on the framework of Zak (2010), creating an intelligent-DSS can open the *black box* decision-making process by integrating learning models. The potential of an AI algorithm will be a great solution method for this. The abilities and classification accuracy of the discussed learning algorithms suit very well on our operational replanning problem.
Based on these characteristics found in literature provided in Figure 3.1, we try to design the most fitting DSS for our research. We will focus on a road transportation, centralized system used on the operational level for transportation processes management. We primarily use deterministic data to provide an underlying decision-making method in the replanning process. Based on the research of Zak (2010), a hybrid methodology of both data mining and optimization could contribute to the current application. We prefer a DSS that makes replanning suggestions based on the output from our learning model. The algorithm should therefore be able to provide correct labels on given input data, so this will be the foundation for our pattern recognition literature study.

## 3.2 Pattern recognition

In this section, an introduction to various pattern recognition models is given. This includes the pros and cons of each model and indicates in which scenarios they are best applicable. We provide a more in-depth description of artificial intelligence and, in particular, the difference between machine learning (ML), reinforcement learning (RL) and deep learning (DL). After explaining the difference, the learning method that best fits on our problem characteristics will be chosen and the right estimator algorithms are discussed in the conclusion.

### 3.2.1 Known Pattern Recognition Models

There are different models opted for pattern recognition, which all have their own characteristics and strengths depending upon the specific tasks it needs to perform (Asht and Dass (2018)). There are six models known in practice, which will be briefly discussed below.

#### Statistical Models

The statistical method for pattern recognition describes patterns based on terms of features that express the probabilistic nature of a class (Fazel and Chakrabartty (2011)). Analyzing these features and the probability distribution of each class, results in the decision boundary for the statistical model. Patterns are projected based on three sequential steps (Asht and Dass (2018)): First, pre-processing operations make the features ready for training and test purposes. Second, the features are measured and selected upon analyzing their performance in training. Then, the model learns and adapts itself for unknown patterns. The test patterns are used to check the classification performance of this learned model. There are two techniques possible to the nature of the statistical model: supervised and unsupervised statistical techniques (Asht and Dass (2018)). The difference between these two techniques is that the first is used for discovering pattern classifications by dimensionality reduction and various feature subsets. The latter uses feature extraction and analysis to detect new patterns in the data (Bishop (2006)). Statistical models are highly effective on noisy and high-dimensional data sets (Asht and Dass (2018)).

#### Structural Models

Structural Models take an additional element into account compared to Statistical Models. This element is the presence of underlying and inherent structures in the patterns. These complex pattern recognition problems include the presence of sub-patterns, that hold vital information regarding the main patterns (Pavlidis (1977)). This model is therefore concerned more with underlying structure and attempts to recognize patterns based on the form of the data. This makes the structural models more applicable for increased complexity patterns like textured images or image interpretation, where patterns have a definite structure (Asht and Dass (2018)). The drawback to this is that the model requires a lot of training data and a lot of computational power. Also, when the basic structure of complex patterns becomes too difficult to define, it can be more applicable to use the statistical models instead (Asht and Dass (2018)).

### Template Matching Models

The third type of model is Template Matching, which is the most simplest and primitive pattern recognition technique. The technique is used in image processing and is a feature-based approach to find similarity between two points/pixels (Asht and Dass (2018)). The pattern can be found by recognizing the feature elements in a stored/matched sample, in order to find a correlation between the two. The efficiency lies in the size of the training set; the more images to compare to, the higher the chance of recognizing the same pattern in the test image. This is also where this approach experiences its shortcomings, since the lack of enough training data and the presence of distorted patterns can decrease the effectiveness of the template matching (Bajcsy and Kovačič (1989)).

### Neural Network Based Models

Neural Networks are a very efficient model in the field of classification. The model is based on a structure that is found in the human brain, with layers of "neurons" that pass compare information based on previous information (Asht and Dass (2018), Garbade (2018)). The strength of this model is that the weights and learning rate of the neural network can be updated iteratively, giving it an edge over all other pattern recognition techniques. The most common model is based on a Multi-Layer Perceptron (MLP) and depending on the problem size, the network can be increased with additional hidden layers and the number of neurons. The neural network based models are closely aligned with the Statistical Models, although the neural network has a lower dependency on prior knowledge (Asht and Dass (2018)).

### Fuzzy Based Models

Fuzzy Based models recognize the patterns by modeling certain forms of uncertainty that can not be fully understand by probability theories, which gives them this "fuzzy" behavior (Asht and Dass (2018)). Most real world problems have some kind of vague and imprecise information in its nature, making this model a very fitting technique (Babuska (2012)). There are two main techniques in Fuzzy Based models: Syntactic techniques, which are utilized when patterns are related to the formal structure of a language or text. The second technique is called Semantic, which is used to reproduce fuzzy parts in data. A similarity measure between the fuzzy description and the reference shape is then used to classify the pattern (Asht and Dass (2018)). So, this model is highly effective on fuzzy data.

### Hybrid Models

Due to the practical relevance, it is often the case that a single model for classification and pattern recognition is not that efficient (Asht and Dass (2018)). This is because a single classifier or a set of features is difficult to classify, because a deeper analysis of available data and prior knowledge is required (Duda et al. (2001)). A good combination is the hybrid of the Statistical and Neural Network based models. The statistical approach is used to recognize the patterns in feature extraction. These can then be used in neural networks to train the set of features and classification performance (Asht and Dass (2018)). We discuss the concepts of AI in order to provide a better understanding of the statistical and neural network based models for pattern recognition.

### 3.2.2 Artificial Intelligence

Artificial Intelligence (AI) is a concept that demonstrates the intelligence of machines. The concept of AI began all the way in the 1960s, but the first implementations were reached in companies much later. The rise of expert systems in the 1980s were the "boom" for AI, since knowledge-based systems became the major focus for businesses (McCorduck (2004)). However, the first decades of the $21^{st}$ century allowed for the rise of Big Data, which increased the number of AI-related products (Manyika et al. (2011)). Because cheaper and faster computers could successfully implement advanced learning algorithms, they can mimic cognitive functions that are associated with human (intelligence) behavior (Russell et al. (2010)). Examples are involving logic in sequential decision-making and the use of if-then rules.

AI has multiple fundamentals, which are ML and DL. These are so-called subsets (deeper levels) of AI: deep learning is a subset of machine learning, while machine learning is a subset of artificial intelligence (Garbade (2018)). This can be seen in Figure 3.2. Some examples of AI in real-life practices are reasoning, image recognition, learning and language processing (Russell et al. (2010)). Typically, an AI machine analyzes its environment and takes actions to maximizes its chance of success. What defines a success is determined by the purpose of the model (Bishop (2006)). This purpose is based on the related tasks the machine needs to solve, which are defined in the subsets of AI.

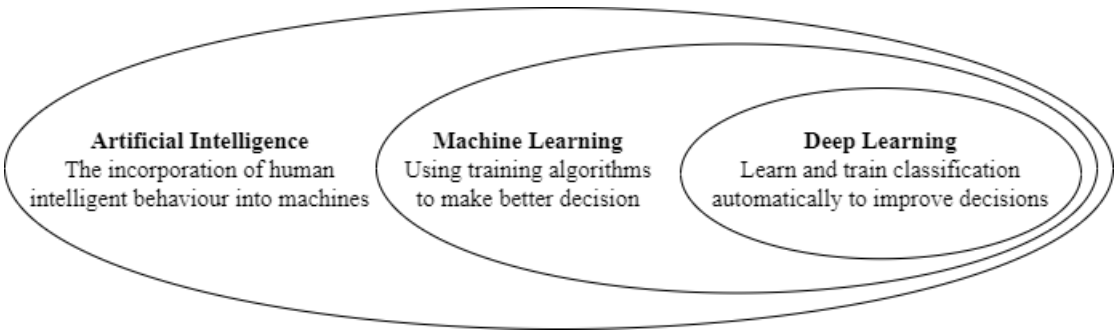



Figure 3.2: The difference between AI, ML and DL (Garbade (2018)).

#### Machine Learning

Machine Learning (ML) is a subset of AI and can be seen as the study of computer algorithms that improve automatically through experience. The main goal is to make machines learn from examples (Garbade (2018), Englebienne (2020a)). It includes the use of statistical techniques and training algorithms to provide the ability to learn (Mes (2020)). Machine Learning can therefore automate the detection of meaningful patterns in data (Shalev-Shwartz and Ben-David (2014)), to perform accurately on new, unseen data samples. There are three main types of machine learning algorithms (Figure 3.3), each differ in their approach/purpose, types of input/output data and the type of tasks they intend to solve. The three types are Supervised Learning, Unsupervised Learning and Reinforcement Learning (RL).

The basic framework of a machine learning model can be divided into four categories: Classification, Regression, Clustering and Dimensionality Reduction (Englebienne (2020a)). The

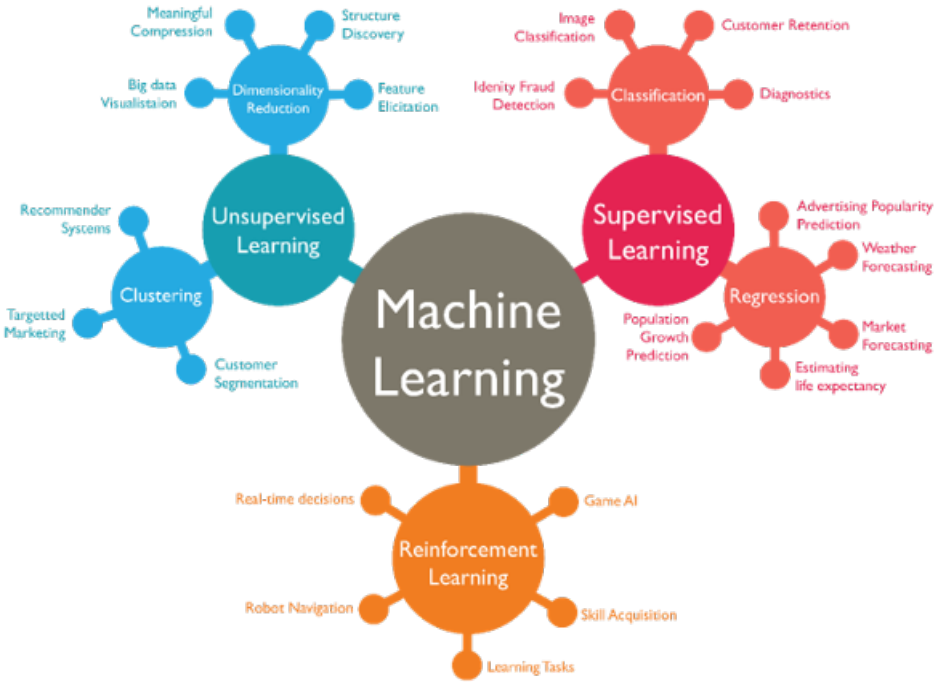



Figure 3.3: The three main types of ML algorithms (Mes (2020)).

following brief overview will define the differences between these approaches and some real-life examples are given to provide the practicality of each algorithm.

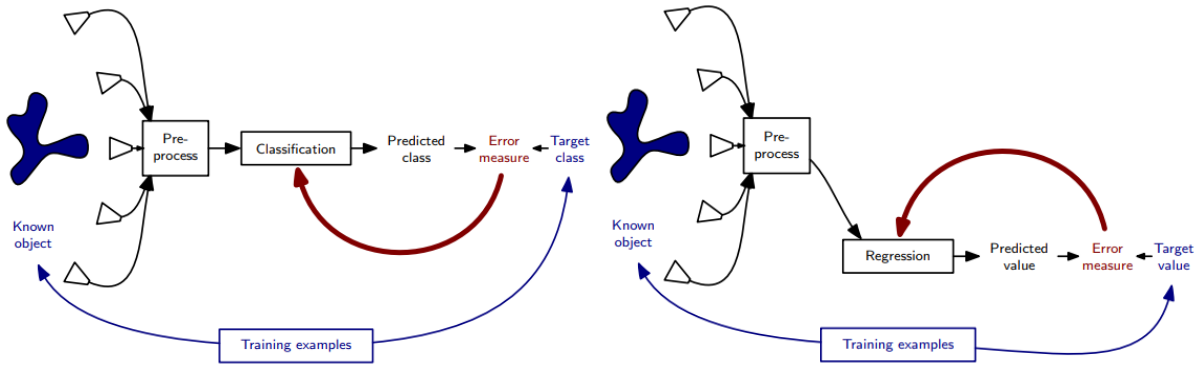
1. **Classification:** Predict a discrete label from features. Discrete variables are associated with a limited number of values, either binary (two possibilities) or a set of mutually exclusive options (McCue (2015)).
   *Examples:*
   - Medical: Classifying X-Rays as "cancer" or "healthy".
   - Spam detection: Classify emails as spam or not.
   - Recognition: Face, speech, image, etc.
2. **Regression:** Predict a continuous value. This is a variable that can take on an unlimited, numerical value between the lowest and highest points of measurement (McCue (2015)).
   *Examples:*
   - Weather forecasting (the wind speed, total millimeters of rainfall, etc.).
   - Financial markets: Predict tomorrow's stock price from past evolution and external factors.
3. **Clustering:** Dividing data into groups, such that items in each group are similar (dissimilar items are in different groups).
   *Examples:*
   - Customer Clustering: Identifying groups of customers with similar buying patterns.
   - Product Clustering: Identifying groups of products that are often bought together.
   - Recommender Systems: Find jointly clusters of movies, books, etc.

4. **Dimensionality Reduction:** Transforming data from a high-dimensional space to a low-dimensional representation, without limiting the loss of information (Englebienne (2020b)).
   *Examples:*
   - Used for data compressing and reconstruction.
   - Used as a pre-processing step, to reduce classifier complexity.
   - Principal Component Analysis (PCA): Technique to provide the most contributing components (independent variables) on the dependent variable.

These four framework categories affect the type of machine learning algorithm that is needed to solve the problem tasks (Shalev-Shwartz and Ben-David (2014)). The earlier mentioned types of ML algorithms in Figure 3.3 will be discussed in the following sections.

### Supervised learning

Supervised Learning algorithms are used to build a mathematical model that contains data that represents both the inputs and the desired outputs (Russell et al. (2010)). The aim of the system is predictive analytics, by using a classifier that is trained on a set of labeled samples in order to predict a previously unseen data element (Kotsiantis et al. (2007), Englebienne (2020a)). The model uses training data, which consists of one or more inputs and the desired outputs. Supervised Learning assumes that new input data is independent of earlier output data (Schmidhuber (2015)). Through optimization and objective functions, the algorithms learns a function that can be used to predict the (labeled) output associated with the inputs (Mohri et al. (2018)). Supervised Learning is task-driven, since the model is trained to label the output data correctly by using either regression or classification (Mes (2020)). Figure 3.4 shows the two model frameworks of Supervised Learning (Englebienne (2020a)). The main difference between the two frameworks is that in Classification algorithms are used when the outputs are restricted to a limited set of values, while Regression algorithms are used when the outputs may have any numerical value within a range (Alpaydin (2014)).




(a) Classification

(b) Regression

Figure 3.4: Two visualizations of the Supervised Learning model framework (Englebienne (2020a)).

Both classification and regression with Supervised Learning algorithms can be implemented with a series of steps (Englebienne (2020a)).

1. Determine and gather of real-world, representative data that is used in the training dataset.

2. Determine the input features that are used in the learning algorithm. Once again, representativeness is important and beware of the *curse of dimensionality* (further addressed in Section 3.3).

3. Determine the structure, classifiers and parameters of the used learning algorithm.

4. Run the learning algorithm on the training set and possibly, the validation set (i.e. Cross-Validation).

5. Evaluate the prediction/classification accuracy on the test set. If necessary, adjust the parameters for an optimized performance and evaluate the algorithm again.

The most widely used Supervised Learning algorithms for pattern recognition are Linear Regression, Logistic Regression, K-Nearest Neighbor, Naïve Bayes, Support Vector Machines, Decision Trees and Neural Networks (Multi-layer Perceptron) (Bishop (2006)). In Table 3.2, the definitions of each model and distinction between the algorithms are discussed.

Table 3.2: The description and practices of the main Supervised Learning algorithms.

| **Algorithm** | **Model Description** | **Examples of Practices** |
|---|---|---|
| Linear Regression | The statistical approach to model the relationship between a scalar response and at least one explanatory variable. The relationship is modeled using linear prediction functions, that focus on the conditional probability distribution of the response variable given the values of the predictors. | Prediction, forecasting, error reduction and variation explanation. |
| Logistic Regression | This algorithm uses a basic function to model the probability of a certain (binary) class or event. This measures the relationship between the categorical (Bernoulli) variable and at least one independent variable. | Medical scales used to assess the patient's surgery risks of certain operations. Also, in engineering, logistic regression can predict the probability of failure of a given product or process. |

Continued on next page

Table 3.2 – continued from previous page

| Algorithm | Model Description | Examples of Practices |
|---|---|---|
| K-Nearest Neighbor | The k-Nearest Neighbor algorithm is both a classification and regression method that uses the number of $k$ closest training points in a dataset to predict the output value. In classification, this output is a labeled class based on the most common class in its neighbors (Altman (1992)). In regression, the output is a property value, based on the (weighted) average of the neighbors. | A useful application of k-Nearest Neighbor are recommender systems. For example, based on certain input values, Netflix can build a "recommendation" section on your previously watched movies. |
| Naïve Bayes | This technique is used for the construction of classifiers. Conditional Probability models are used to assign class labels to new samples, where the labels are drawn from some finite set. Based on the algorithm, the Naïve Bayes Classifier uses a small number of training to model the maximum likelihood of the problem (Al Hajj Hassan et al. (2020)). | Naïve Bayes works particularly well on textual data, hence its practical use in Natural Language Processing problems. These kernel functions make the not linearly separable problem instance linearly separable. |
| Support-Vector Machines | As an addition to linear regression, Support-Vector Machines are able to perform non-linear classification by using kernels, which map the low-dimensional inputs into high-dimensional feature spaces. | Support-Vector Machines are used for outlier detection and the classification of images, hand-written text and biological sciences (Gaonkar and Davatzikos (2013)). For regression, the model uses a Bayesian parameter and the kernel functions allows Support-Vector Machines even possible for Clustering in Unsupervised Learning. |

Continued on next page

Table 3.2 – continued from previous page

| Algorithm | Model Description | Examples of Practices |
|---|---|---|
| Decision Trees | This predictive model is commonly used in data mining, since it creates a model to predict a target variable based on several input variables (Rokach and Maimon (2008)). Observations from data are represented in branches and the target values are represented in the leaves. These values can either be discrete (classification) or continuous (regression). | Decision Trees are a great method for decision analysis, since the tree allows the visualization of all possible decisions in the problem. |
| Multi-layer Perceptron (Neural Network (NN)) | This model is a specific class of the traditional feed-forward NN and consists of at least three layers: an input, hidden and output layer (Bishop (2006)). The latter two layers have activation functions on each node, which allows the network to distinguish data that is normally not linearly separable. | The NN is a very popular ML algorithm, which has many applications in the fields of image- and speech recognition, classification of sequence recognition and robotics. NNs are the pathway to the next subset of ML: Deep Learning. (further addressed in Section 3.2.3) |

Since our research focuses on finding the best method to find patterns and strucure within data, we go more in-depth in the (dis-)advantages of certain supervised learning models Complementing the research of Xhemali et al. (2009), we will now briefly elaborate four supervised learning models.

**Decision Tree** This model is generally an easy and fast method to calculate good predictions. The strength of this model is that the predictions are based on decisions of the most important features first. Each decision is based on one feature and is represented as a node in the tree. The decisions are based on which feature gains the most information regarding the output variable (Bishop (2006)). The Decision Tree model will typically perform better when more data is available. Also, it is not required for the Decision Tree to normalize and scale the input data. However, the model is very unstable in its performance if a small change occurs in the data. Training the Decision Tree is also sensitive to overfitting and made decisions on the test data do not necessarily result in the optimal solution (Bishop (2006)).

**Random Forest** The big difference between this model and the Decision Tree is that the Random Forest does not rely on a single decision. This main advantage allows the classification performance to be more stable because it combines the decision policies of many Decision Trees together and makes

final decisions based on a majority vote across the trees (Bishop (2006)). This reduces the overfitting and stability weaknesses of Decision Trees. Despite these advantages, the Random Forest requires a lot more computational resources and training time. Also, the Random Forest model can be biased to high-cardinality features, which could lead to unreliable prediction scores. Choosing the correct feature selection technique is therefore a crucial step to overcome this problem.

**Naïve Bayes** This method is very popular due to the simplistic nature and high computational efficiency. The statistical classification technique is based on the Bayes Theorem, making it a fast, accurate and reliable algorithm (Xhemali et al. (2009)). The strength of this method is that it works efficiently on large datasets and performs really well on discrete, categorical output variables. The big disadvantage is that this classifier assumes independent features, making it very hard to use on high-dimensional problems. Also, there is a recurrent issue called the "zero-frequency problem", which occurs if a certain class is not available in the training set.

**Neural Network (MLP)** The final proposed method is the most powerful technique to represent complex relations. This big advantage is possible because a Neural Network can cope really well with high-dimensional problems because the structure of the model can be tailored to the problem (Strisciuglio (2020)). When properly designed, the Neural Network is quite robust to noise in training data and any errors in training do not affect the performance on the test data. The huge disadvantage of a Neural Network is the designing itself. There is no assurance or set of specific rules for determining the most promising structure, so the appropriate network is found by setting up experiments and find promising trial and error (Brownlee (2018)).

### Unsupervised learning

Unsupervised Learning is also a predictive analytics method where the algorithms learn patterns from a set of (training) data that only contains input features to try and find structures. There is no given solution set, meaning that the output data has no class labels or categories (Englebienne (2020b)). Since the output label of the data is unknown, the model is data-driven trained to proper cluster/group the data or to reduce the dimensionality of the data. (Mes (2020)). There are two main methods in Unsupervised Learning: Cluster Analysis and Principal Component Analysis (PCA). They aim to discover better representations of the inputs during training (Bengio et al. (2013)).

Cluster Analysis is used to find different groups or segments within the elements in the data that have no label. This assigns the set of observations into clusters, so that observations within a cluster are similar according to at least one criteria (Roman (2019)). This also aids the detection of anomalous data points that do not fit in any of the clusters. Components of the clustering algorithm are: pattern representation (optionally, feature extraction), definition of patterns proximity, clustering of data, data abstraction and assessment of output (Jain et al. (1999)). An example of this process can be found in Figure 3.5.

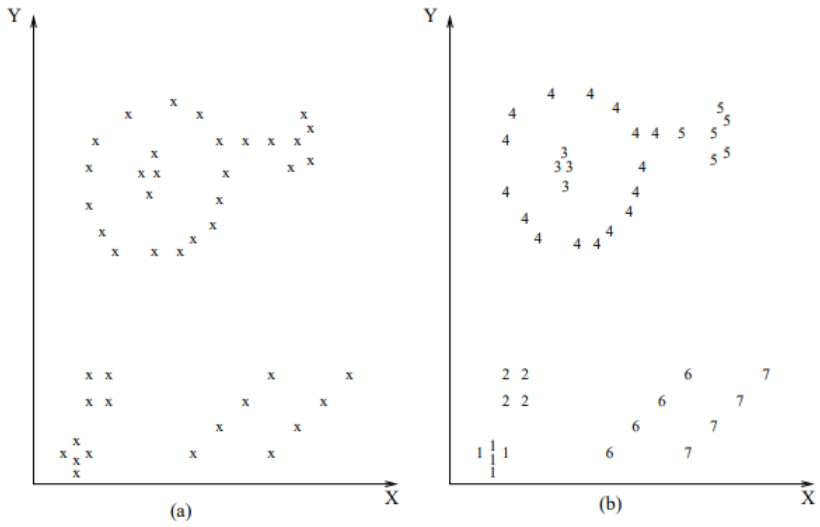



Figure 3.5: Visualization of the Unsupervised Learning method: Cluster Analysis (Jain et al. (1999)).

Principal Component Analysis (PCA) is a dimensionality reduction process where data is transformed from a high-dimensional into a new low-dimensional space (Shalev-Shwartz and Ben-David (2014)). This method compresses and computes the principal components, which are the variables that maximizes the total variance on the output data (Bishop (2006)). These principal components are projected in a orthogonal visualization of the spread out data points. An example of the PCA can be found in Figure 3.6.

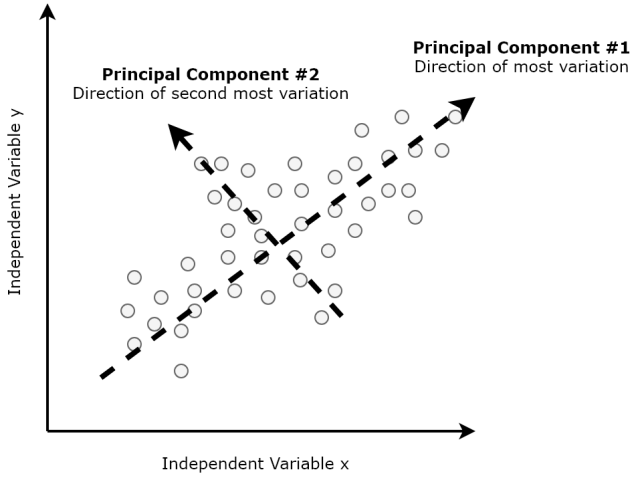



Figure 3.6: Visualization of the $1^{st}$ and $2^{nd}$ principal components (Bishop (2006)).

In practice, Unsupervised Learning is also a great application for the field of probability and statistics. Density estimation is the determination of the underlying data distribution within the input space, based on observable data (Bishop (2006)). This identifies the properties of a given data set and provides very useful insights on the features, skewness and multi-modality. However, the problem of density estimation

is fundamentally ill-posed, since there are indefinitely probability distributions that can represent the observed data. Any non-zero distribution can be a potential candidate and there are many distributions possible (i.e., Gaussian or non-parametric), making it a central issue within pattern recognition (Bishop (2006)). Because a density estimation problem can be defined as a Conditional Density Estimation, the conditional probabilities can be modeled via theoretical, empirical or advanced statistical models like Neural Networks and Random Forests (Breiman (2001)).

**Reinforcement learning**

The final type of ML algorithm is Reinforcement Learning (RL). This learning method allows an intelligent agent to interact with its environment and modifies its actions or policies in order to maximize the rewards to its actions (Lewis and Vrabie (2009)). The agent predicts what and why certain actions will happen and the algorithm provides actions to take advantage of the predictions (Mes (2020)). This defines RL as a prescriptive analytic and reward-based learning method. The difference between RL and Supervised Learning is the focus on finding the sequence of actions that balances the exploration and exploitation of the problem (Mes (2020)). This influences the type of data that this learning method requires. Previous methods required that a set of data points were independent and identically distributed (IID), but RL requires the data to be measured sequentially or based on time-series (Bishop (2006)). The environment for RL is defined by a Markov Decision Process, which is a classical method of formalizing sequential decision-making with a discrete time setting (Sutton and Barto (2018)). The interaction between agent and environment is visualized in Figure 3.7. Due to the characteristics of RL, its practical applications make it particular attractive for tasks that require planning and sequential decisions. Examples of implementations are Game Theory, Strategic Decisions, Routing problems and Robotics.

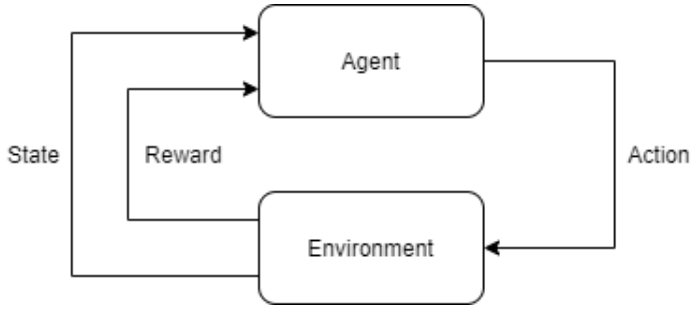



Figure 3.7: The interaction space and sequential decision-making process of RL (Mocanu (2020)).

To determine the most applicable RL algorithm, a set of learning dimensions need to be defined. following from this, we can determine the learning model and algorithm. Some RL methods are Approximate Dynamic Programming, Temporal Difference learning, Q-learning and Deep Q-Networks. The following learning dimensions describe the use of a RL algorithm (Mes (2020), Mocanu (2020)):

- **Model-free or Model-based**: Reinforcement Learning can be modeled with two approaches. Model-free RL is applicable when the sampling and environment is represented based on the real world. The algorithm learns by every interaction to predict the best reward in the world. Model-based RL uses a defined model to work and learn from. The interaction space of the model

is based on Bellman equations, which is a set of equations that break the optimization problem into a sequence of sub-problems (Dreyfus (2002)).

- **Real-world (online) or simulator (offline)**: Online models can be trained on running systems/applications and use data sequentially at each moment in time to improve the decision/policy. Offline models train based on static data sets and simulate the sequential decision rewards on a local simulator. The model is then utilized and updated interchangeably with the actual system.

- **On-policy or off-policy**: On-policy means that the agent learns the value of a policy following from the exploration steps. Exploration means that the systems favors the gathering of new information over making the best decision on current information (known as exploitation). This influences the exploration strategy of the learning process and allows the algorithm to improve during interaction with the environment. Off-policy allows the agent to learn the best decision making policy independently from the interaction with the environment, so only on available information.

- **Active or passive learning**: Active learning allows the agent to learn from the value functions and the chosen policy. There are no fixed decisions, the goal is to determine and learn the optimal policy. Passive learning provides the agent with a predetermined policy which can not be altered. The objective is to execute and evaluate the fixed policy (sequence of actions).

### 3.2.3 Deep Learning

The final AI subset is Deep Learning, which is the next evolution of ML. Deep Learning algorithms are inspired by the human brain; when new information is received, it tries to compare it to known/experienced items before making sense of it (Garbade (2018)). This allows the learning model to train itself to perform tasks and automate the decisions-making process. To model this process, Deep learning architecture focuses primarily on Artificial Neural Networks (NNs) that have multiple layers between the input and output layer, making them Deep Neural Networks (Schmidhuber (2015)). The components in a NN are always consistent: layers, neurons, weights, biases and functions. An example of a simple Multiplayer Perceptron (MLP) neural network can be found in Figure 3.8. This NN shows the five components that are mentioned previously, which will be explained below (Bishop (2006)).

1. **Neuron**: Each node (circle) in the model represents a neuron, which is a variable that contains certain information. The location in the network (more specifically, the layer) determines the type of variable. There are three types of neurons: input (the nodes containing an X), hidden and output.

2. **Layer**: All nodes that are arranged vertically represent a Layer. Layers are connected via arcs which pass information between layers. Inside a layer, the neurons do not exchange information with each other.

3. **Weight**: The link between each neuron, the arc, has a certain weight parameter. This represents the strength between the neurons and the importance of the value. Weights can be updated with an optimizer. (Brownlee (2018))

4. **Bias**: The bias value (represented by the Nodes containing the value "1") are able to provide a direction to the activation functions of the Neurons. Bias requires no input neurons to have effect and it helps fitting the model better on the observed data.

5. **Function**: The final component in NNs are the activation functions inside the Neurons. These functions defines how given input is transformed into the resulting output. Determining the correct activation function depends on the data distribution (i.e. Gaussian or Bernoulli) and the classification problem.

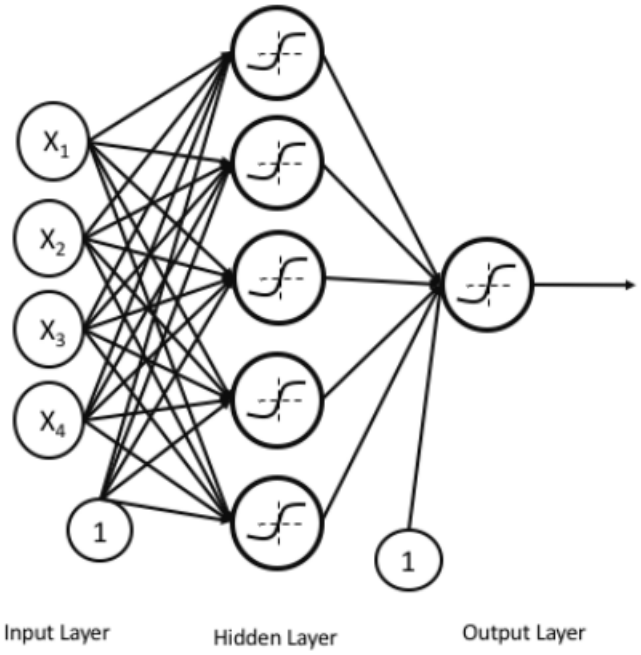
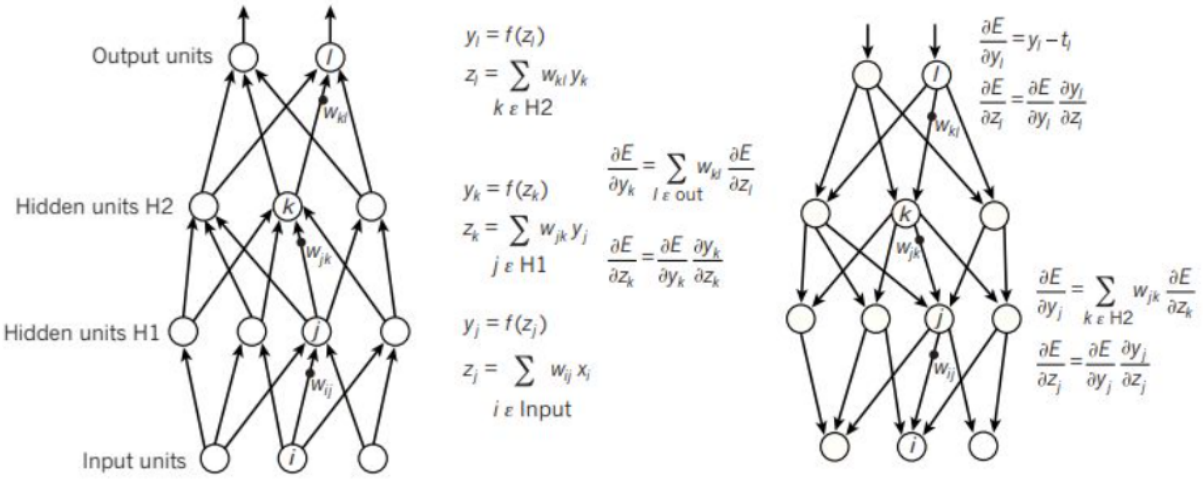



Figure 3.8: The basic Multilayer Perceptron consisting of three layers.

A Deep Neural Network is basically a MLP that stacks many hidden layers on top of each other. The large network is then able to use high-dimensional data in order to automatically discover representations needed for classification or detection (Lecun et al. (2015)). These layers feed information both towards the output units (forward) and back to the input units (backward), which is shown in Figure 3.9.

The forward step in Figure 3.9a represents the response of units that are computed by linear combination of both their inputs and bias, which are then passed through a non-linear activation function. This results in the classification accuracy of the neural network. The backward step in Figure 3.9b does two things: it computes the error derivative (the gradient) of the neuron's input features and it updates the weights of the arcs on the hidden- and output layers neurons. Brownlee (2018) mentions two well-known optimizers used in practice: Adaptive moment estimation (Adam) or Batch- or Stochastic Gradient Descent (SGD). The back propagation step is used to actually train the neural network which enables it to learn the appropriate internal representations of input to output (Rumelhart et al. (1986)) and it was first introduces for training Recurrent Neural Networks (Graves et al. (2013)).




(a) Forward Step

(b) Backward Step

Figure 3.9: Visualization of the Forward Step (activation) and Backward Step (learning) (Lecun et al. (2015)).

Since Deep Neural Networks are able to progressively extract higher-level features from raw data, most practical applications are linked to highly complex problems like image, speech or natural language processing. These raw data sets contain several hundreds of data points that are modeled via tens to hundreds layers and millions of neurons into a one-dimensional output layer. However, keeping a neural network relatively small, makes it more applicable to recognize patterns and even has its advantages over other ML algorithms (Xhemali et al. (2009)).

### 3.2.4 Conclusion

We first described known models that can be used for pattern recognition. After describing the pros and cons, a hybrid of both the statistical and neural network based models would be applicable to our problem. More importantly, we introduced the concepts of AI and how various learning algorithms can be used for prediction (by regression or classification) and decision-making. The goal of this study is to find patterns in the replanning data, making this research more in line with the Supervised Learning algorithms. Based on Table 3.2, we will implement multiple algorithms and the statistical comparison of the performance metrics of each classifier will be made. Following from the literature, we focus our research on the classification comparison of the Naive Bayes, Decision Trees & Random Forest and Neural Networks. Details regarding the chosen parameters for each learning model will be described in the Solution Design chapter.

## 3.3 Challenges in Data analytics and mining

In this section, we discuss multiple challenges that might be encountered during the implementation of statistical and machine learning algorithm. To overcome these phenomena, we will use literature from the field of Data Analytics. This provides different methods for each challenge and we will conclude at the end of this section which methods will be used on our research problem.

### 3.3.1 Curse of Dimensionality

The first known problem is the *Curse of Dimensionality*, which was originally discovered by Bellman (1957). Dimensionality means simply the total number of attributes and features in a dataset. The main difficulty that arises in machine learning algorithms, particularly for (pattern) classification applications, is that the complexity grows exponentially with the growth of features in the data (Arel et al. (2010)). The phenomena is problematic because an increase in dimensionality and volume space, have influence on the statistic significance and reliability of the results (Bellman (1957)).

Reducing the dimensionality of the dataset can be overcome by certain learning models like PCA (described in Section 3.2.2) or by feature selection techniques. The main difference in feature selection methods are Filter-, Wrapper- and Embedded methods (Shrajluhaniwal (2020)).

- **Filter methods**: These methods are not incorporated specifically to a machine learning algorithm, but are used to filter out non-contributing predictors by using a statistical function. As a result, the methods have a low computation time and are less prone to overfitting. Examples of filter methods are statistics like Chi-square tests, ANOVA-tests, Feature Importance and Information Gain.

- **Embedded methods**: Embedding means adding features during the model building process. Feature selection is done at each training iteration and can reduce overfitting by penalizing coefficients if the model becomes too complex. Examples are Lasso and Ridge regression or tree- and rule-based models.

- **Wrapper methods**: These methods evaluate a specific machine learning algorithm in order to find optimal subsets of features. They have a high computational time for datasets with many features and have a higher chance of overfitting due to extensive training.

Wrapper methods are the step-wise selection methods, which have two main approaches: forward and backward. Both approaches use various statistics to measure the quality of the features used in the model (James et al. (2013)). Forward selection starts with an empty model, the null model, and adds features subsequently that contribute the most to the output (response) variable. This is repeated until the next feature does not add any significant improvement to the model prediction accuracy. Backwards selection starts the other way around. all features are included from the start and removed one by one. The variables with the largest p-value and least statistically significant value are removed. The reduction is stopped by a certain stopping rule, i.e. all not statistically significant features are removed. Forward Selection is always possible, but is however a greedy approach which can lead to redundant variables. Backward Selection provides a better set of features, but can only be implemented if the number of unique features sets is larger than the sample size (James et al. (2013)).

### 3.3.2 Overfitting

Another challenge that follows from using learning models and statistics is the danger of *Overfitting*. This is the result of modeling a bad and too complex learning model on the training data. As a result, the model remembers a huge number of unnecessary data instead of learning to important features (Englebienne (2020a)), which results in a bad performance on the test data. The model unconsciously obtains some residual variation, i.e. noise, which could lead to bad prediction and classification of future observations (Everitt and Skrondal (2010)).

There are two main solutions to tackle the problem of overfitting. First, the data can be separated into three, distinct sets (Bishop (2006)). A train set is used for fitting the objective function; a validation set for the model selection and hyper-parameter settings; and test set for the actual prediction and model performance. The ratio of splitting the data into these three subsets depends on the total amount of data, but a common rule-of-thumb states a good starting point of 60% Training, 20% Validation and 20% Test (Isaac (2015)). When the availability of training data is limited, a solution is to use k-fold cross validation to repeatedly split the training and validation data in $k$ different and unique sets, and average the results over all folds (Bishop (2006)). An example can be found in Figure 3.10. It is also possible to use cross-validation on only training and test sets, which could be used to assess the performance stability of a model across the different divisions of both data sets.

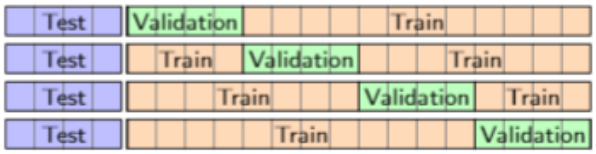
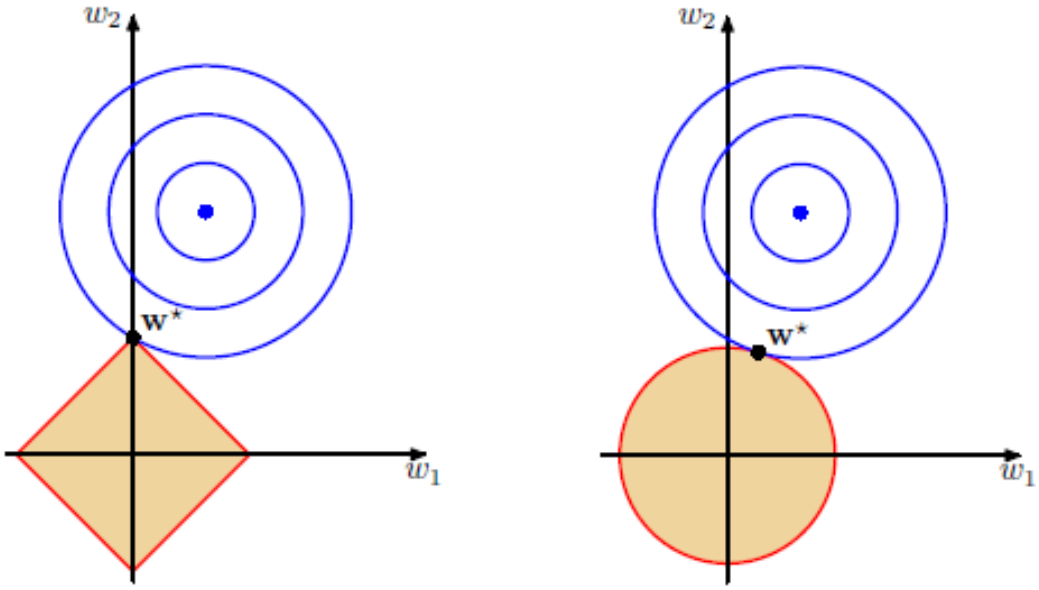



Figure 3.10: Visualization of using k-fold Cross Validation (k = 4).

The second solution to overfitting is the use of regularization techniques. This allows complex models to be trained on data sets of limited size without severe overfitting, by simply limiting the model complexity by using penalty terms on the optimization function (Bishop (2006)). This adaptation uses parameters to reduce the generalization error of the trained model. Implementations of regularization are mostly on neural networks and statistical models, and the most used practices are Lasso Regression (the $L_1$ norm), which is used to induce sparsity by adding an absolute value of magnitude as a penalty term on the loss function; and Ridge Regression (the $L_2$ norm), which decays the weight of neurons and shrinks the value of their coefficients by using a squared magnitude penalty term on the loss function (Nagpal (2017)). The key difference between both norms is that Lasso shrinks less important features to absolute zero, making it better for feature selection. A visualization of how both techniques update the weights ($w$) can be found in Figure 3.11.




(a) Lasso Regression ($L_1$ norm)

(b) Ridge Regression ($L_2$ norm)

Figure 3.11: Plots of the regularization contours of both $L_1$ and $L_2$ Regularization (Bishop (2006)).

### 3.3.3 The bias-variance trade-off

This well-known phenomenon in the field of statistics and machine learning is a central problem in supervised learning. The ideal model is both accurate in capturing the regularities in the training data and the generalization of test (unseen) data (Geman et al. (1992)). A model that suitably learns on a training dataset and generalizes well to the hold out, test dataset is a good fitted model (Brownlee (2018)). Both the bias and variance are errors that measure the expected Mean Square Error of the learning algorithm (James et al. (2013)) on the test dataset. This measure is expressed as the expected loss of the algorithm, with respect to each individual dataset ($D$), and the expression can be found in Figure 3.12 below (Bishop (2006)). The formula consists of the two components: bias and variance.

$$\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right] = \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}$$

Figure 3.12: Formula that expresses the bias-variance trade-off.

The *bias* represents the extent to which the average prediction over all data sets differs from the desired regression function (James et al. (2013)). A high bias can cause an algorithm to miss relevant relations between features and outputs (known as *underfitting*). Basically, incorrect models lead to high bias (Geman et al. (1992)). The *variance* measures the extent to which the solutions for individual datasets vary around their average, hence it measures the extent to which the algorithm is sensitive to the choice of datasets (Bishop (2006)). High variance causes the algorithm to model random noise and irrelevant features in the training data instead on the intended test data (known as *overfitting*). Total model-free algorithms that converge slowly due to large parameters typically have a high variance. Since the model

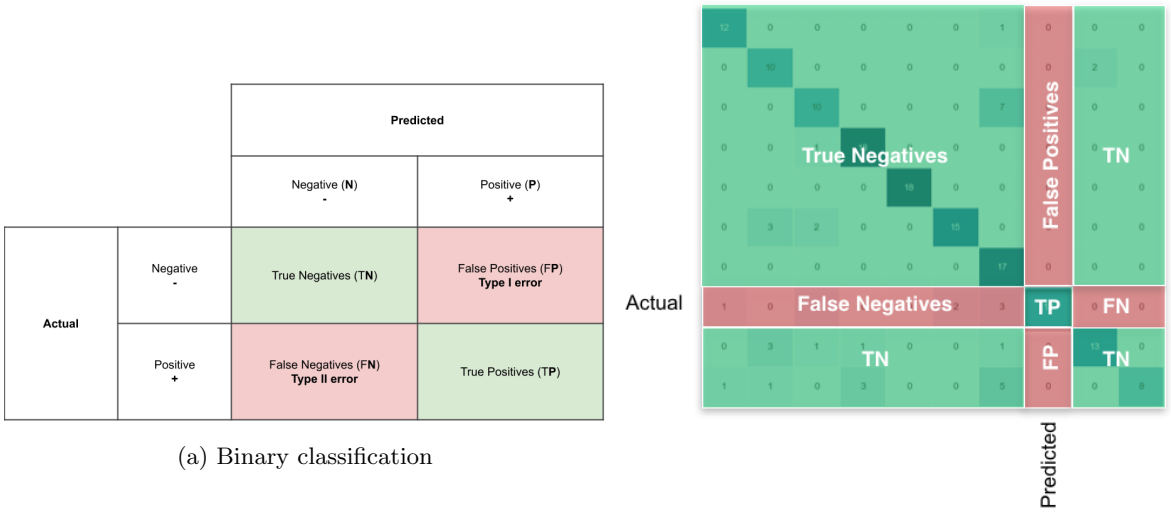
has no boundaries in finding its parameters, the model becomes over-trained and sensitive to noise (Geman et al. (1992)).

The bias-variance decomposition can be implemented in both regression and classification learning models. Main approaches to tackle this problem is to use dimensionality reduction and feature selection techniques to simplify the model complexity which results in a decease in variance. The extent of these techniques and their resulting level of control differs per learning algorithm (James et al. (2013)). For example, in neural networks, the variance increases and the bias decreases with the increase of hidden layers (Geman et al. (1992)). In decision trees, the depth of the tree influences the variance and a common method to control this is pruning (James et al. (2013)). Pruning is a strategy that allows us to grow a very large tree and compress it back in size by removing nodes that have the lowest test error rates (James et al. (2013)). In the end, each learning algorithm has some tuning parameters to control the bias and variance. In regression, the regularization methods introduce bias and reduce variation. In classification, the expected loss of the learning algorithm is described as a miss-classification rate or as a probabilistic classification error. Some measures used in practice are the Gini Index or Cross-Entropy Loss (Bishop (2006), James et al. (2013)).

### 3.3.4 Imbalanced data

It is possible that the distribution of classes in the training and test data is not equal, making the classification problem imbalanced (Brownlee (2020)). Typically, the class distribution in the data is severely skewed to certain labels, making the classification a hard challenge. For example, in a binary (2-class) problem with 100 samples, it is possible that 90 instances are labeled with class 1. This example has a highly imbalanced data set (90:10). Most classification data sets do not have an exactly/perfect distribution of classes over the number of instances, so imbalance is a common situation (Brownlee (2020)). To assess the performance of the classification model in the previous example, we can not solely rely on the classification accuracy anymore, since it gives a worst-case performance for the minority class. There are a couple of techniques to overcome an imbalanced dataset.

A good method is to resample the data. The goal is to even-up the classes in the data by resampling in one of two ways (Brownlee (2020)). It is possible to either add copies / instances from the under-represented class (over-sampling) or to delete instances from the over-represented class (under-sampling). There are some rules of thumb for both methods, like the size of instances determines either under- of over-sampling and the resampled ratio between the classes can be experimented with (Brownlee (2020)). Besides resampling with additional copies, it is also possible to generate synthetic samples. A well-known method is the SMOTE technique: Synthetic Minority Over-sampling Technique (Brownlee (2020)). This technique works as follows: consider some training data with $s$ samples and $f$ features. To perform oversampling, take a random sample from the dataset and take its $k$ nearest neighbors. Then, create a synthetic data point by taking the vector between one of those $k$ neighbors and the current data point. This vector will then be multiplied with a random number between 0 and 1 to create the new, synthetic data point.




(a) Binary classification

(b) Multi-class classification

Figure 3.13: Two examples of a confusion matrix in a classification problem.

Another method to overcome the imbalanced data phenomenon is to change the performance metric of the model. Mentioned previously, the classification accuracy can be a misleading metric when the classes are imbalanced. There are other metrics that have been designed to tell a more represent and truthful performance if an imbalanced data set is used. The most clean and unambiguous way to represent the prediction results of a classifier is the confusion matrix (Brownlee (2016)). There are two layouts of confusion matrices, depending the model is either used on a binary or multi-class classification problem. Examples of both matrices can be found in Figure 3.13. Both the confusion matrices indicate four types of classification possibilities: *true positives (TP)*, *true negatives (TN)*, *false positives (FP)* and *false negatives (FN)*. Basically, the perfect classifier labels all samples correctly in the true positive and true negative cells of the confusion matrix (Brownlee (2020)). Then, for each class, the predicted label is the same as the actual label. Incorrect predictions can either be in the form of false negative (the model incorrectly classify a sample as the false class) or false positive (all actual labels that the model did not classify correctly). These four types of predictions can then be used to calculate additional performance metrics. Some examples of these metrics are discussed below (Xhemali et al. (2009)).

**Accuracy**: The proportion of the total number of predictions that were correct. This measures the closeness of all measurements to a specific value. Imbalanced data causes a so-called Accuracy Paradox; if there is a large class (like 95% of the data), this will always have a high classification accuracy. However, the model becomes too crude to be useful, because the other 5% will have a very poor accuracy. This bad performance can be further explained with the Precision metric.

$$\textbf{Accuracy (\%)} = \frac{(TN + TP)}{(TN + TP + FN + FP)} \tag{3.1}$$

**Precision**: The number of True Positives divided by the number of True Positives and False Positives. This represents the number of positive predictions (of a class) divided by the total number of positive class values predicted (also called the *Positive Predictive Value (PVV)*. This metric measures the classifiers exactness. If the precision is low, it indicates that the model predicts a large number of False Positives.

$$\textbf{Precision (\%)} = \frac{TP}{(TP + FP)} \tag{3.2}$$

**Recall**: This metric is calculated by the number of True Positive divided by the True Positives and False Negatives. It represents the number of positive predictions divided by the total number of actual class values, which is called the Sensitivity or *True Positive Rate (TPR)*. It can be seen as a measure of the classifiers completeness. If the recall is low, it indicates that the model predicts a large number of False Negatives.

$$\textbf{Recall (\%)} = \frac{TP}{(TP + FN)} \tag{3.3}$$

$F_1$ **Score**: The $F_1$ Score/F-measure is a weighted average of both the Precision and Recall of the classifier. It represents the balance between the two other metrics, making this metric more significant; the F-measure can only produce a high result if both the Precision and Recall are high.

$$F_1 \textbf{ Score (\%)} = \frac{(2 * Recall * Precision)}{(Recall + Precision)} \tag{3.4}$$

**Cohen's Kappa**: The Kappa score measures the classification accuracy, but it normalizes the score by the imbalance of the classes in the data. In other words, the Kappa score measures the degree of agreement between the actual values and the predicted values (the *inter-rater reliability*).

$$\textbf{Kappa Score} = \frac{2 * (TN * TP - FN * FP)}{(TP + FP) * (FP + TN) + (TP + FN) * (FN + TN)} \tag{3.5}$$

**ROC**: The Receiver Operating Characteristic (ROC) shows the *sensitivity* (the number of TP classifications) and *specificity* (the number of FP classifications) of the classifier. It compares two characteristics, the True Positive Rate and the False Positive Rate. The ROC visualizes curves in a space graph, with the two characteristics on the $x$ and $y$ axes (example is Figure 3.14). The performance of the graph can be assessed with the area under the curve (AUC). In the example below, the Intermediate ROC has a bigger AUC than the No Power ROC, making it a better classifier. The "perfect classification" would be at the top left of the graph. Then, the prediction has a 100% sensitivity (True Positive Rate of 1) and 100% specificity (False Positive Rate of 0).

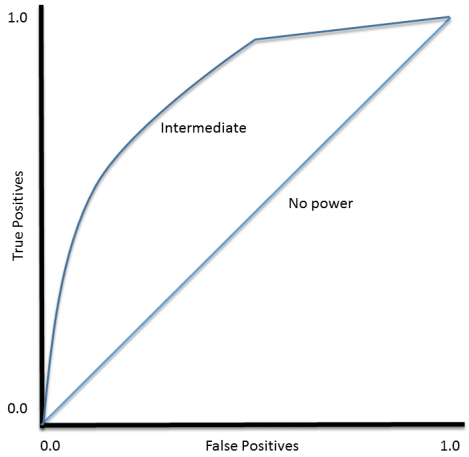



Figure 3.14: Examples of two ROC curves: No Power classifier and Intermediate classifier.

### 3.3.5 Conclusion

In the final section we discussed several important challenges regarding data mining. The explained literature first described the *Curse of Dimensionality* phenomenon regarding the selection of features. This problem can be overcome with various methods and this research will use, depending on the learning model, either a Filter and/or a Wrapper method. This combination could benefit each other in both accuracy increase and computational cost decrease. The second discussed challenge is overfitting, which will be overcome by implementing k-fold cross validation on the pre-processed data. The next challenge discusses the balance between bias and variance of the model. To tackle this problem, we use the appropriate measure depending on the used learning model (i.e., Gini Impurity for Decision Trees). Finally, to counter the classification performance of imbalanced data, we will take the additional metrics into account to assess the performance of each model.

# Chapter 4

# Solution design

This chapter describes the series of steps and actions that were taken for the solution design and the practical application of the proposed learning algorithms on the Client's case. Section 4.1 explains the proposed methodology for the remainder of this thesis. Section 4.2 elaborates the steps that were taken to create the clean dataset for the learning models. In Section 4.3, the model configuration, characteristics and fixed parameter settings of each learning algorithm are described. Finally, in Section 4.4, the settings for the experimental setup are given and the evaluation procedure of the learning models.

## 4.1 Methodology

This research follows a series of steps that define our research methodology. To elaborate on the process, we explain the steps in chronological order. The used methodology can be found in Figure 4.1 and the steps are divided into four stages, indicated by the large squares in the figure. The models are used based on a multi-class classification problem, meaning that the desired output is to predict the correct ride (class) given a set of input features. Each stage of the methodology contributes to finding the most promising results for these models and we will describe each step in remainder of this section.

The first stage of the methodology describes the preprocessing steps that will transform the raw data from the OOMPD into a clean data set for the learning models. Since the UAR measures all human planner adjustments of each day, the acquired data tables need to be joined together in a single data set. The first step of this stage is also relevant to test and implement the UAR on the planning software. Besides this step, the process will adjust existing and create some additional features. Feature creation is needed because variables from the raw dataset need to be processed into usable input features for the learning models. We also create some additional features based on supplementary files that are extracted simultaneously with the daily UAR data. After the feature creation step, the data will be inspected in order to clean and filter the data elements. This consists of handling missing data and deleting faulty data. Resulting form the inspection, we have a clean data set ready to be used as input for the learning models.

The second stage has two inputs: the cleaned data set and the human planner expertise. The latter is a list of criteria and assumptions that define the practical relevance and requirements that the learning models need to take into account. We explore and discuss multiple learning models. Each model requires its own feature selection technique and (hyper)parameter settings methods. After every model is defined,

we determine the KPIs (the performance metrics). These describe how well every learning model predicts the output classes on the test set. We have to take the challenges described in the literature into account when we approach the next stage. We address how we overcome the curse of dimensionality, overfitting and an imbalanced dataset.

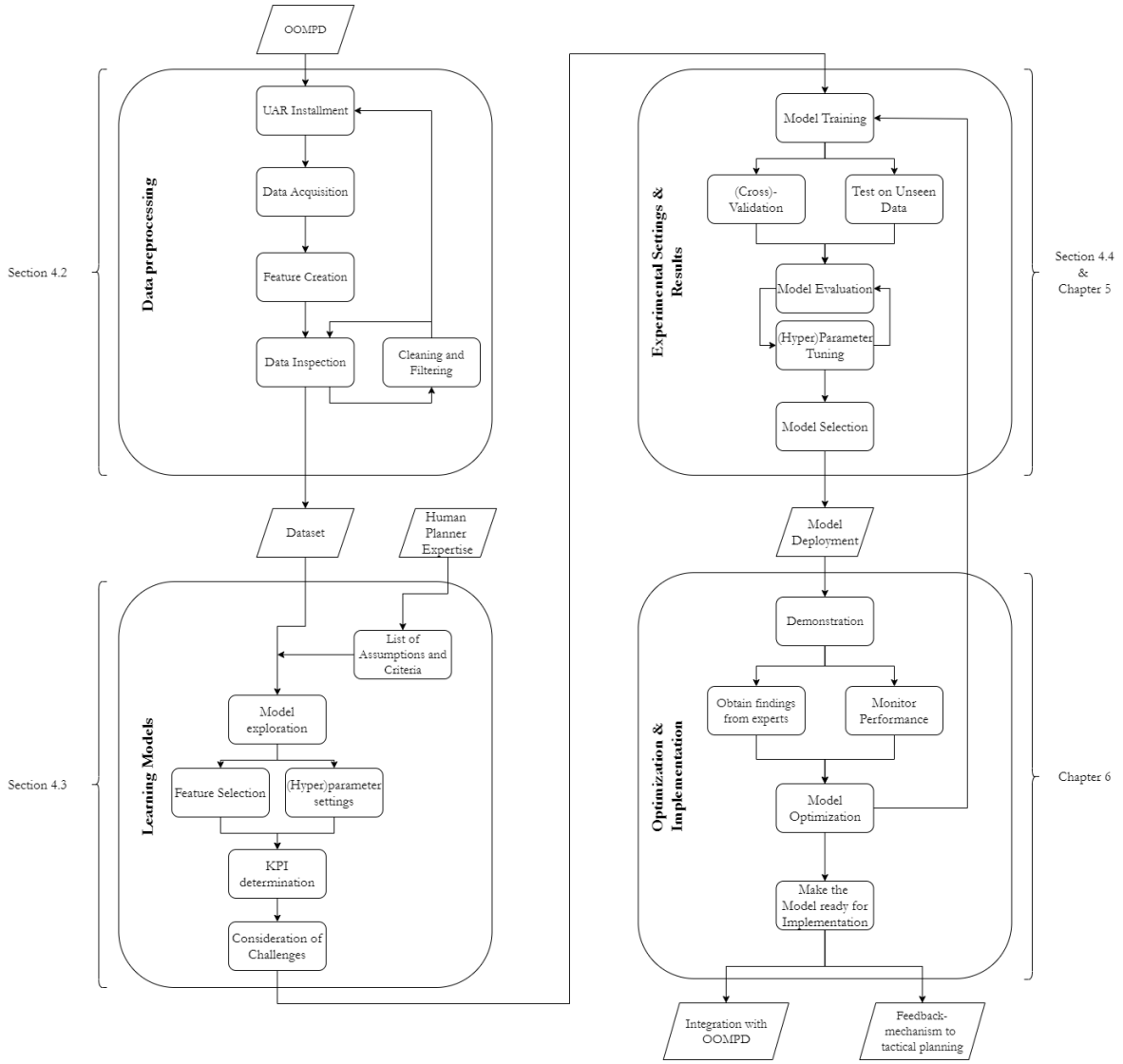



Figure 4.1: Our proposed methodology used in the remainder of this research.

The third stage is used to describe the experimental settings of our research. This starts with the crucial step of training with the use of cross-validation. This is a useful technique to assess the classification performance, because the method compares the metrics over different splits of the input data. All splits are then averaged to find a more stable performance of the metrics. Also, testing the model on unseen data allows us to determine how well the model performs by verifying that the classifier functions produce expected results. Based on these two comparisons, we evaluate each model by tuning the (hyper)parameter settings in order to reach an improved or even optimal classification performance. Afterwards, we select the best learning model based on the performance metrics scores.

After selecting the best learning model, we make the model deployment-ready for testing. This

includes a demonstration on a testing environment of the OOMPD application to validate its performance. Experts and human planners will provide additional findings on the demonstration and the computational performance of the learning model will be monitored. Based on these steps, the model will be further improved when necessary. If a degradation in the performance is detected, we need to re-train the new learning model and conduct new experiments based on stage three. When the model reaches substantial results, the integration with OOMPD will be deployed. Also, a feedback-mechanism towards the input planning will be constructed by making a DSS between the operational and tactical planning systems.

## 4.2 Data preprocessing

In this section, we describe the entire preprocessing process to convert the raw data into a clean, input dataset. First, we describe how the UAR is installed and it is used to acquire the data. Then, we describe the preprocessing steps to create the features for the input data for the learning models. Finally, we inspect the obtained data and clean any faulty data before the dataset is completed.

### 4.2.1 Acquiring the UAR data

Mentioned previously in chapter 2, a total of three raw data (Comma-Separated Values) files are daily exported from the software. These files consist of a table with the recorded entity attributes from the UAR, a table with the division of the ride owners over the rides and finally a table consisting additional ride information from the application. These three data files will be exported from the OOMPD daily and can then be imported into our model separately into one, raw dataset that will be further processed. This process is illustrated in Figure 4.2 and we will briefly discuss the steps in this section.

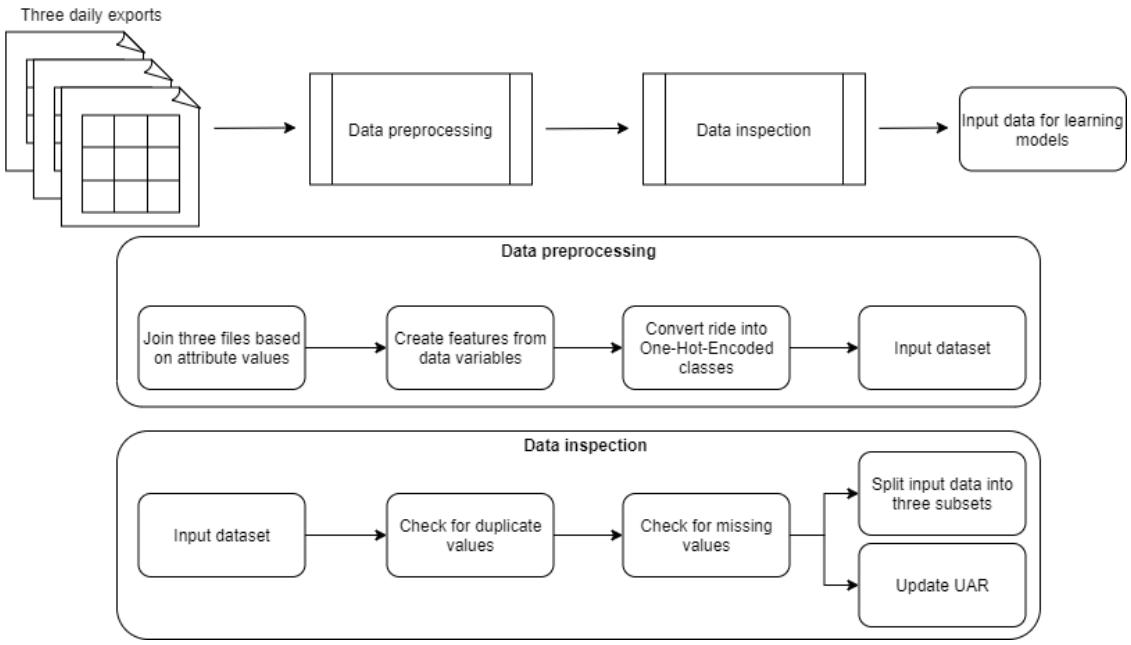



Figure 4.2: The process of acquiring the UAR data and necessary preprocessing steps to create input data.

After obtaining the daily data exports, we used two processes to convert them into viable input data for our learning models. The first process is to tackle variables and data types that cannot be used directly as input for the learning models. These steps are referred to as feature & output creation, which will be discussed in detail later in this section. The second process is necessary to inspect any remaining errors or missing values in the input data. These values need to be dealt with great care, to make sure no faulty data can influence the learning models. The inspection steps and their results are discussed in detail later in this section. Essentially, after these two data preprocessing steps, we obtain one large input dataset that consists of multiple daily human planner adjustments. All detailed descriptions of the input data elements can be found in Appendix B.1. Each entity in the input data represents one replanning adjustment and all feature values are the order, ride and ride owner details associated to this one adjustment. The output of one data entity is the new ride it is replanned to, which will be converted into an output class for the learning models. There are a total of 300 different rides in the planning of OOMPD, so the models are used on a multi-class classification problem and try to predict the correct output class. Following from the data preprocessing steps, there are 24 features and 91 output classes present in the input data. We will describe the preprocessing steps in the upcoming sections

### 4.2.2 Feature & Output preparation

A critical step in creating a clean, input dataset for the learning models is the creation of features. All variables described in Table B.1 will be used as features, so we will discuss the necessary feature creation steps below.

Boolean variables that have string values (i.e."True"/"False") need to be adjusted into integer values, since the learning models can not handle the string data type properly. The positive value of the Boolean variables are converted into ones, negative values into zeros.

For the string variables, we need to convert unique values into unique ID's. For example, there are a lot of ride owners present in the data (due to the many subcontractors). The UAR measures each adjustment separately, so it can occur that multiple Ride adjustments are associated with the same ride owner. To create the ride owner as a feature for our learning models, we need to identify each unique ride owner and convert them into an numeric value (an unique ID). We created an Unique ID converter in Python, which we will briefly discuss based on the pseudo-code in Algorithm 1.

The pseudo-code provides the step wise approach to convert unique string values from any variable into a unique integer. The pseudo-code creates both a list and an array, based on this column. The first is a copied list of all values, which then drops all duplicate values in order to only contain unique values. Then, a 2-D array is created with the shape of the copied list and an empty $2^{nd}$ column. This $2^{nd}$ column will contain the Unique ID, which is based on a simple loop counter. For each unique string value (the first for loop in the pseudo-code), we add the Unique ID to the array and we compare for each string value in the Input column if it matches the current unique string value (the second For Loop). If the two strings are the same, we adjust the string in the UAR into the Unique ID. The algorithm is finished when all string values are converted to their unique ID. This Unique ID converter is used for any feature that contains a string value, which are the *Subcontractor*, *RideName*, *ConsignorParty_CountryCode*, *ConsignorParty_PostalCode*, *ConsignorParty_CityName*, *ConsigneeParty_CountryCode*, *ConsigneeParty_PostalCode*, *ConsigneeParty_CityName*, *Party_CountryCode*, *Party_PostalCode* and *Party_CityName*.

**Algorithm 1:** Pseudo-code: Converting Unique String values into Unique IDs.

**Input:** Column from UAR containing string values
**Output:** Unique integer for each string value

```
Create a copied list of Input values;
Delete all duplicates within the list ;                    // by using drop.duplicates()
Create array of shape (CopiedList, 2) ;                    // columns:  String value, ID value
Unique ID Converter;
Set ID counter to 0;
for i in range(length of list) do
    ID counter += 1;
    Array[i,1] = ID counter ;                              // write ID to the 2nd column
    for j in range(length of UAR column) do
        if String value in List = String value in UAR then
            String value in UAR = ID value in List
        end
    end
end
```

The output variable for our learning models is the categorical variable *NewRide*. This variable represents the possible Ride the human planner can change their adjustment to. There is a limited set of Ride options available due to the distribution area of the DC. In order to use this variable as an output for the neural network, another processing step is needed. Multi-class classification for neural network require the output labels to be processed from integers to binarized values. This process is called *One-Hot-Encoding* as it converts each unique value into a 1-dimensional array of binary values. For example, two data element containing either the value "1" or "2" are converted into the arrays *[01]* and *[10]* respectively. For our data, we implement the One-Hot-Encoding preprocessing step with the *LabelEncoder* from the *sklearn.preprocessing* library. After that, we use the *to_categorical* function from the *keras.utils* library to convert the vector of unique integers to a binary matrix. An example of this resulting matrix can be found in Table 4.1, which shows the first sixteen output variables from the test set being One-Hot-Encoded into binary 1-D arrays.

Table 4.1: Example of a One-Hot-Encoding matrix of the first sixteen output Rides.

| **Ride classes** | **One-Hot** | **Ride classes** | **One-Hot** |
|---|---|---|---|
| 137 | 000000010 | 427 | 001000000 |
| 137 | 000000010 | 315 | 000100000 |
| 109 | 000000001 | 314 | 000010000 |
| 139 | 000000100 | 452 | 010000000 |
| 139 | 000000100 | 452 | 010000000 |
| 250 | 000001000 | 452 | 010000000 |
| 250 | 000001000 | 452 | 010000000 |
| 314 | 000010000 | 505 | 100000000 |

### 4.2.3 Data Inspection

After all features and output classes are created, we need to inspect the data in order to create the final dataset as input for the learning models. We first need to handle all data samples that have missing values. We can either delete or impute missing values and since the data resembles actual replanning behavior, we can not impute all missing data values. For example, calculating a missing value for a subcontractor based on its neighboring values does not hold any practical relevance. Therefore, any missing data regarding specific subcontractor or ride information will removed by using list wise deletion. Data that can be imputed, like the *ScenarioDayNumber*, will be handled with interpolation. Because we obtain daily replanning data in separate files, the value of the day number should be the same for all data elements within one file. Deleting data elements that lack this value is redundant, so we use interpolation to set these values to the related known values in the data.

To further clean the dataset, we used some visual inspection to look for faulty data. Based on some descriptive (summary) statistics and visualizations, we look for any data values that do not represent the realization and should contain certain, expected values. The summary of the data inspection step can be found in Table 4.2. First, there are some data entities in which the boolean variable *Replanned* is "False". These adjustments are not carried through in the planning by the human planners, but the UAR still recorded the data attributes due to the configuration. We have to remove these entities since they do not represent actual replanning. Secondly, there are some duplicate data entities present in the input data. These duplicates occur because the UAR is configured to measure multiple microflows in order to obtain all replanning actions from the human planners. Sometimes, one adjustment passes more than one of these microflows and therefore it is measured multiple times. We remove any fully duplicate entities, meaning that for any adjustments that have the same values across all features, the duplicates are deleted. Finally, there are data entities that still miss feature values after the data preprocessing steps. When no information is passed throught the microflows, the UAR stores an empty value on the data attribute. Mentioned earlier, we impute the missing values of some features based on the neighboring entities. For some features this will not be possible, so when any of these values are empty, we remove the entity from the input data.

Table 4.2: Summary of the data inspection step.

| **Action** | **# of data elements** | **% of data** |
|---|---|---|
| Raw data | 1953 | 100% |
| Remove if Replanning not finalized | 305 | 15,62% |
| Remove duplicates | 132 | 6,76% |
| Remove missing values | 63 | 3,23% |
| **Total input data** | **1453** | **74,40%** |

## 4.3 Learning models

This section will describe how this research will use the proposed learning models on the obtained data. First, each model will have fixed (hyper-)parameter settings and their foundation will be briefly discussed in Section 4.3.1. Then, in Section 4.3.2, we describe the feature selection techniques used in each model to find the most important replanning factors. Finally, we introduce the performance metrics that are used to assess the performance of each model in Section 4.3.3.

### 4.3.1 Fixed parameter settings

#### Decision Tree & Random Forest

The most important parameter setting for these methods is the splitting criteria. Nodes split the decision based on the Information Gain following from the feature. There are two criteria that can be used to calculate this Information Gain: the Gini Impurity or Entropy Loss (Aznar (2020)).

The Gini Impurity (Equation 4.1) measures the frequency at which an entity in the dataset is miss-classified when we randomly classify. When a decision is made, the criteria aims to find a Gini Index close to 0. This means that the node is pure, meaning that all data elements inside the node are from one, unique class. The optimum split is then found by a (set of) feature(s) that provide the lowest Gini Index.

$$GiniIndex = 1 - \sum_{j} p_j^2 \tag{4.1}$$

Entropy (Equation 4.2) measures the information gain that indicates the disorder of the features with the target. Just like the Gini Impurity, the criteria tries to find the best split by using features that provide the lowest Entropy. This indicates that the node is homogeneous and consists of only one output class. The information gained from the feature can be calculated by measuring the difference in Entropy of the root node and the nodes following from the decision.

$$Entropy = -\sum_{j} p_j \cdot log_2(q_j) \tag{4.2}$$

In both equations, $p_j$ represents the true probability of class $j$ and $q_j$ the estimated distribution. Both criteria are good methods to calculate the information gain, but the Gini Index is less computational expensive (Raileanu and Stoffel (2004)). This is because Entropy requires some logarithmic functions (as seen in Equation 4.2), which take more time to calculate. Therefore, we will use the Gini Impurity as the Decision Tree and Random Forest criterion. The models will look which decisions on the features contribute the most to the Gini Index.

#### Naïve Bayes

This algorithm can perform real-time predictions on multi-class problems, due to its fast calculation technique. The classifier assumes a class conditional independence based on the Bayes rule, which gives this classifier the so-called naïve characteristic. This Bayes rule can be found in Equation 4.3.

$$\arg\max_{n} P(C_n|D) = \frac{P(D|C_n) * P(C_n)}{P(D)} \tag{4.3}$$

The above function maximizes the probability of a certain class ($P(C_n)$) given a dataset $D$ (the posterior probability). The numerator of Equation 4.3 multiplies the likelihood of data given a certain class

($P(D|C_n)$) by the prior probability of the class being classified ($P(C_n)$). The denumerator provides the evidence, by dividing the previous calculation by the prior probability of the entire data ($P(D)$). For our experiments, we use the Gaussian Naïve Bayes classifier from the *sklearn* library, since this classifier supports continuous, numerical features.

**Neural Network**

We focus in our research on the Multi-Layer Perceptron architecture, which is seen as the foundation for many neural networks. There are some rules of thumbs when setting up the parameters of this model, which we wil discuss below (Ranjan (2019)).

We first describe the *architecture* (the arrangement of layers and nodes) of our proposed NN. Based on Ranjan (2019), the number of hidden layers will be initially set to two. For each hidden layer, the total number of neurons (the *width* of the layer) should be based on the geometric progression of 2 (i.e., 2, 4, 8, 16, etc.) and the first hidden layer should have around half of the number of input neurons. Therefore, the total number of hidden layers (the *depth* of the network) will be initially set to two and the total number of neurons in the hidden layers (the *size* of the network) to 16 and 8.

The number of input neurons are based on the features. The size of the output layer is based on the total number of output classes in the data, so each output class is associated to one neuron. To prevent overfitting, a neural network uses a Dropout layer to randomly ignore certain neuron information. The resulting effect makes the network more capable of processing noisy data and encourages the network to actually learn representations in the data. For our model, we use a Dropout layer after every hidden layer, with a dropout rate initially set to 0.5. This will later be tuned in the experimental settings.

With the architecture defined, we now address the activation functions that define the output of a neuron based on the input and weights. For a Multi-Layer Perceptron, it is common to use the *Rectified Linear Activation Function (ReLU)* for the hidden layers and a *He Uniform* weight initialization in the input layer (Brownlee (2021)). The *ReLU* function can be found in Equation 4.4 and calculates the non-negative activation value based on the input feature value $x$.

$$ReLU = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases} \tag{4.4}$$

The activation function for the output layer is determined by the classification problem, which is in our research a multi-class prediction problem. Since one node per class is present in the output layer and they are mutually exclusive, the activation function should be the *softmax* function. This function outputs a vector of values that sum to 1, and can therefore be interpreted as the predicted probabilities of the class memberships. The *softmax* activation is calculated based on Equation 4.5, where $x$ is a vector of outputs and *exp* is the mathematical constant based on a natural logarithm.

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{4.5}$$

With the activation functions described, we move to the optimizer technique and the loss function. The optimizer technique is a learning algorithm that navigates through the space of possible weights in order to make good predictions. In our model, the weights are trained and updated with the Stochastic Gradient Descent optimizer, due to the potential of better learning on imbalanced data. This potential can be achieved by tuning the parameters associated with the optimizer, which are the *learning rate*,

*momentum* and the *batch size* (Brownlee (2019)). The algorithm for Gradient Descent to update a weight ($w_{t+1}$) can be found in Equation 4.6.

$$\begin{aligned} v_t &= \eta * v_{t-1} + \gamma * \nabla w_t \\ w_{t+1} &= w_t - v_t \end{aligned} \tag{4.6}$$

The learning rate ($\eta$) describes the slope or degree in how much the weight will be updated and is always a value between 0 and 0.5. The momentum ($\gamma$) allows the neural network to prevent sensitive weight updating and enables the model to not get stuck in local minima. Momentum typically has a value between 0.5 and 1 and is multiplied on the weight gradient before updating ($\nabla w_t$). Finally, the batch size describes how many training samples are used when updating the weights. This affects the speed and stability of the learning method. All three learning parameters will be tuned during the hyper-parameter experiments. The loss function evaluates a candidate solution, in this case a total set of weights, by minimizing a objective function. The loss function associated with multi-class classification is the *categorical cross-entropy loss*, which penalizes the predicted probability score by a logarithmic function. This means that small errors receive small penalty scores, while relatively large errors receive enormous penalty scores. The algorithm for the loss function can be found in Equation 4.7 and the pseudo-code in Algorithm **??**.

$$CrossEntropy = -\sum_{c=1}^{m} y_{o,c} \log(p_{o,c}) \tag{4.7}$$

In the above equation, the cross-entropy is calculated over all classes $m$. For each class, the binary indicator $y_{o,c}$ indicates whether class $c$ is correctly classified for observation $o$. This binary indicator is based on the one-hot-encoded value of the target. This is the multiplied by the logarithmic value of the predicted probability $p_{o,c}$ of observation $o$ over class $c$. The weights are updated if the mean loss score is minimized.

### 4.3.2 Feature selection methods

According to the found literature, feature selection techniques are intended to reduce the number of input variables (like dimensionality reduction) and it is a critical technique to set up the performance of prediction models. For each model, a specific method is used that best suits the model characteristics. Brownlee (2019) discusses some selection techniques which are a good fit for our proposed learning models. We briefly discuss each feature selection technique and how they are implemented.

**Filter method**: This method uses a statistical measure to score the correlation between the features and results in a *k-best* subset of features importance scores. Filters methods are so-called Classifier Agnostic (CA) techniques, meaning that the feature scores are calculated without using specific model details. The used filter method is based on the *scikit-learn* library called *SelectKBest*, which selects the $k$ best features and filters out the rest based on a scoring function. The type of data determines which score function should be used. Table 4.3 provides a framework of the statistical measures that can best be used based on the type of data (Brownlee (2019)). Since we have a categorical output variable, we only focus on these statistical measures. The obtained data from the UAR consists of both numerical and textual data. After the data preprocessing steps, we converted all input variables to both numerical and categorical (nominal) values and the output variable to categorical (nominal) values.

Table 4.3: Framework for filter-based feature selection statistics

| Input variable | Output variable | Statistic/Score function |
|---|---|---|
| Numerical | Categorical | Linear: ANOVA correlation<br>Nonlinear: Kendall's rank |
| Categorical | Categorical | Linear: Chi-Squared test<br>Nonlinear: Mutual Information |

This allows us to use either the Chi-Square test or the Mutual Information statistic. The latter proves to be a useful and powerful method for both types of input variables (agnostic with respect to data types), so we decided to use this filter method for our Naïve Bayes Classifier.

**Intrinsic method**: These feature selection methods are the ML algorithms that automatically perform feature selection during training. These methods are Classifier Specific (CS), meaning that these feature scores are tailored specific to the classifier's internals. This includes the algorithms that already look for predictors that contribute the most to the classification accuracy, like the tree- and rule-based models. Both these models select the features based on the Feature Importance obtained from the model. It assigns scores to the input features that resembles the relative importance to the prediction value, which provides useful insights on both the dataset and the predictive performance of the model. We use the Classification and Regression Trees (CART) feature importance method from the *sci-kit.learn* library for both the Decision Tree and Random Forest model.

The only model that does not have a direct method to calculate the feature importance is the Neural Network. To overcome this challenge, we introduce a straightforward method that calculates the estimated feature importance in an easy way. The formula to calculate this estimate can be found in Equation 4.8, which we will briefly explain below.

$$f_j = s - \frac{1}{n}\sum_{i=1}^{n} s_{ij} \tag{4.8}$$

Essentially, a feature's importance ($f_j$) is the difference between the baseline performance score of the model ($s$) and the average performance score obtained by permutating the feature ($s_{ij}$). This will result in a difference between the two scores. If this difference is small, then the model is not sensitive to this feature, resulting in a low importance value. Inversely, if the difference is large, then the feature's importance is high. After permutating each feature separately, we will have the relative importance of all features. We can control the number of permutations per feature ($n$) in order to balance out computation time and better estimate stability.

### 4.3.3 KPI determination

To overcome the imbalanced data challenge and the Accuracy Paradox mentioned in the literature, we discussed some extra performance metrics found in the literature that could counter this phenomena. The additional metrics are the Precision, Recall, F-Measure, Kappa Score and ROC. These will be, besides the prediction accuracy, the performance metrics in which we will assess our learning models. According to Brownlee (2020), the ROC-area under curve and the Kappa Score are the most common and effective metrics to evaluate the classifiers performance for imbalanced class distribution. In our

study, all classes are equally important; there is no Ride more important than any other. Also, the *false positive rate* and *false negative rate* are equally important, meaning we take the $F_1$ *measure* into account as well. The models should not only correctly predict the relevant Ride, but also the irrelevant Rides. There are both important because if our model classifies a replanning adjustment falsely, it has a large implication on the real-life practices. For example, it affects the validity of the remaining planning and it can result in distribution errors inside the DC. Finally, the integration of the models effect the computational performance of the planning system. It is therefore relevant for practical implementation to measure the computational time of each model. To conclude, we rank all performance metrics that are used to assess our models below. This ranking is based on a trade-off between the literature and human planner expertise.

1. Cohen's Kappa
2. ROC Score
3. $F_1$ measure
4. Precision
5. Recall
6. Accuracy
7. Computational Time

## 4.4 Experimental approach

In this final section of the solution design, we describe the setup of our experiments and how each model will be evaluated and selected. First, we describe the necessary steps to create the datasets used for training, validation and testing. Next, we outline the experimental setup of two different experiments: hyper-parameter tuning and performance evaluation experiments. Thirdly, we illustrate and interpret the results following from the experiments and how these can be used to select the most promising Supervised Learning model.

### 4.4.1 Creating input datasets

The preprocessed input data from the UAR will be randomly divided into three different subsets: the training set, validation set and test set. The entire input dataset will be divided into two parts of 80% and 20%. The larger subset will then be divided again with a 80/20 split, so we end up with the three subsets: 64% for training, 16% for validation and 20% for testing. Based on the viable input data from the *Data Inspection* step, all models are trained and validated on 937 samples and tested on 235 samples. The first two subsets are used to train the models and their parameters/weights and the resulting model will then be used on the test subset. To overcome the challenges in data analytics, we use both the training and validation sets to perform k-fold cross-validation. This counters the importance of noisy and unnecessary data and creates stability when the model optimized its objective function. We use two settings of k-fold cross validation, which will be described later in the experimental settings.

### 4.4.2 (Hyper-)parameter tuning experiments

The first set of experiments are about tuning the (hyper-)parameters of the learning models. We conduct grid-search experiments to calculate a variety of parameter values for all models. Each experimental setting is used on one division of training, validation and test data. To improve the performance stability and counter over-fitting, we use 5-fold cross validation to replicate these experiments five times.

For the Decision Tree model, three different parameters are going to be tuned. These parameters are the maximum depth of a tree, the minimum number of samples required for a split and the decrease of Gini impurity needed to split a node. The three parameters and their range of values can be found in Table 4.4, which result in a total of 540 unique parameter settings per experiment. We chose these three parameters because we combine both a stopping criteria parameter and a pruning parameter. This combination makes it possible to counter under- and overfitting and trade-off the models performance with generalizability.

For the Random Forest model, we vary the number of estimators (trees) that are used for the prediction and the depth of the individual trees. We tune these two parameters to improve the predictive accuracy and control-overfitting of the Decision Tree classifier. Furthemore, we use the best setting for the *min_samples_split* and *min_impurity_decrease* parameter found in the Decision Tree grid-search experiment, in the Random Forest model as well. Based on the range of values in Table 4.4, the tuning of the Random Forest consists of 144 unique parameter settings.

Table 4.4: Table with the Decision Tree & Random Forest hyper-parameter settings for the grid-search experiment.

| **Hyper-parameter** | **Range of values** |
|---|---|
| ***Decision Tree*** | |
| Maximum Depth (*max_depth*) | [4, 6, 8, 10, 12, 14, 16, 18, 20, 24, 28, 32] |
| Samples Split (*min_samples_split*) | [0.1, 0.2, 0.3, 0.4, 0.5] |
| Impurity Decrease (*min_impurity_decrease*) | [0.00005, 0.0001, 0.0002, 0.0005, 0.001, 0.0015, 0.002, 0.005, 0.01] |
| ***Random Forest*** | |
| Number of estimators (*Trees*) | [1, 2, 4, 8, 16, 32, 64, 100, 150, 200, 400, 1000] |
| Maximum Depth (*max_depth*) | [4, 6, 8, 10, 12, 14, 16, 18, 20, 24, 28, 32] |

For the neural network, we vary the earlier discussed hyper-parameters to try and optimize its prediction performance based on the categorical Cross-Entropy Loss metric (Brownlee (2018)). An overview of each parameter and their range of values for the experiments can be found in Table 4.5. For the architecture parameters of the neural network, a total of 96 unique parameter settings are tested in the experiment. For the Gradient Descent optimization parameters, each experiment consists of 180 unique parameter settings.

Table 4.5: Table with the Neural Network hyper-parameter settings for the grid-search experiments.

| **Hyper-parameter** | **Range of values** |
|---|---|
| *Optimizer (SGD) settings* | |
| Learning Rate ($\eta$) | [0.1, 0.05, 0.02, 0.01, 0.005, 0.001] |
| Momentum ($\gamma$) | [0.99, 0.95, 0.90, 0.85] |
| Batch Size ($B$) | [1, 2, 4, 8, 12, 16, 32, 64, 128] |
| *Architecture settings* | |
| Hidden Layers ($H$) | [1,2,3] |
| Number of Neurons ($N$) | [4, 8, 16, 20, 26, 32, 40, 48] |
| Dropout rate ($d$) | [0.2, 0.3, 0.4, 0.5] |

### 4.4.3 Feature selection experiments

To find the most important replanning factors, we conduct a feature selection experiment to rank the features across all models. Based on the discussed selection methods, we calculate the feature importance scores of both a Classifier Specific (CS) and a Classifier Agnostic (CA) method for each model. The chosen methods can be found in Table 4.6. For both methods, we conduct a 10-fold cross-validation experiment and to find stability across the models, we use the paired student's t-test to find statistically significant features scores. Equal feature scores and ranks across all models are very practically significant for answering the research question. Especially for the results on the CA selection method are useful, since these calculate the relative importance of the feature compared to each other.

Table 4.6: Framework for filter-based feature selection statistics

| **Learning model** | **CS selection method** | **CA selection method** |
|---|---|---|
| Decision Tree | CART Feature Importance | Permutation Importance |
| Random Forest | CART Feature Importance | Permutation Importance |
| Naïve Bayes | Chi-Squared test<br>Mutual Information | Permutation Importance |
| Neural Network | Permutation Importance | Permutation Importance |

### 4.4.4 Model performance experiments

The final set of experiments are to evaluate the performances of the learning models. When the improved hyper-parameter settings, we can compare the four learning models. Each experiment consists of a 10-fold cross validation iteration on a random split of the input data. These experimental settings are chosen to improve the stability of their classification performance. The most promising model is then used to create the Artifact for our research and is used to improve the planning performance. The model is implemented to predict the number of replanning actions based on the daily planning, which will decrease the total replanning time of the human planners. The Artifact is trained and tested on data collected in August and we will approximate the benefit of the Artifact with estimations and calculations.

## 4.5 Conclusion

In this chapter, we discussed our proposed four-phased methodology (visualized in Figure 4.1) and experimental approach to test the learning models and proposed solution. First, we explained the data processing steps to convert the UAR data into viable input data for our learning models and discussed the goal of the models. Features and output values are created by converting the raw numerical and string values into normalized or one-hot-encoded values. The models are used for multi-class classification, so they use the input data (consisting of 24 features) to try and predict the correct class (the replanned ride number).

After this, we discussed which parameter settings of each models remain fixed and the feature selection methods to measure the importance scores. The fixed parameters are mainly the used loss functions of each models and rules of thumbs for the Neural Network architecture. Also, we provided a total of seven performance metrics used to evaluate the model performance and to overcome challenges like data imbalance and the accuracy paradox.

We then proposed our experimental approach. First, we conduct grid-search experiments to find the optimal hyper-parameter settings of each model and with improved parameter settings, we run a 10-fold cross-validation experiment to assess the classification performance and feature importance scores of two selection methods of each model. The feature scores across the models are tested with a paired t-test, which provide the practical significance of finding the most important replanning factors.

The most promising learning model and subset of features are used to create the Artifact. This will provide the practical relevance of our research by improving the business performance. The Artifact is implemented and tested in order to improve the replanning process, by decreasing the total replanning time by the human planners after using the Artifact's prediction on the operational planning.

# Chapter 5

# Experimental Results

In this Chapter, we discuss our findings from the experimental settings and explain which Supervised Learning algorithm best suits our complex problem. First, we describe the results from the hyper-parameter tuning experiments regarding the Random Forest and Neural Network classifier. The results of these experiment are discussed in Section 5.1. The second set of experiments are used to assess the performance of each model. Section 5.2 provides the resulting feature importance ranks and the scores of the performance metrics.

## 5.1 (Hyper-)parameter tuning

This Section is divided in three subsections, each describing the results of the hyper-parameter tuning experiments associated to a specific learning model. First, we describe the tuning experiments of our Neural Network in Section 5.1.1. Then, Section 5.1.2 provides the results of the Decision Tree parameter tuning experiments. Finally, in Section 5.1.3, where we tune the number of estimators and the depth of the trees in the Random Forest classifier. We conclude this first set of experiments and propose our found parameter settings for each model in Section 5.1.4.

Each section contains two elements that are used to discuss the tuning experiment performances. First, we visualize the performance of each individual experimental setting in a parallel coordinates plot. This plot consists of multiple vertical axes which are used to provide the values of each parameter and the corresponding performance on two metrics (*Cohen's Kappa* and *computation time*). The second element in each section is a table that shows the average performance of the baseline model (with standard parameter settings) and the improved model (with optimal parameter settings). These provide a more clear overview of the experiment results. Both elements are based on 5-fold cross-validation, improving the stability and learning behavior of the model.

### 5.1.1 Neural Network

First, we provide the results from the tuning experiments for the Neural Network. We introduce two separate grid-search experiments to improve the model: The tuning of the optimizer and the tuning of the network's architecture. Results of both experiments are described in the following subsections.

**Stochastic Gradient Descent**

The first experiments were carried out to enhance the training behavior of the Neural Network. According to theory, we can improve the learning behavior of our neural network by configuring the gradient parameters and the number of training samples of the Stochastic Gradient Descent optimizer. The first experiment consists of 180 unique settings of the Learning Rate, Momentum and Batch Size and results are compared with 5-fold cross-validation. The results can be found in the parallel coordinates plot in Figure 5.1.

The parallel coordinates plot shows multiple y-axes which represent the three SGD parameter values and the performance scores of two metrics: Cohen's Kappa score and the computation time. Each individual plot describes the used parameter settings and the resulting performance scores of the two metrics. The plot is also highlighted based on the Cohen's Kappa Score; only the top 10 percent are visible. Furthermore, we used a coloring scale for each experimental result which is based on the computation time. Green indicates that the experiment was relatively fast (within the 20 percent of fastest learning) and red indicates a long computation time. The plot in Figure 5.1 is based on 5-fold cross validation results of the experimental settings.

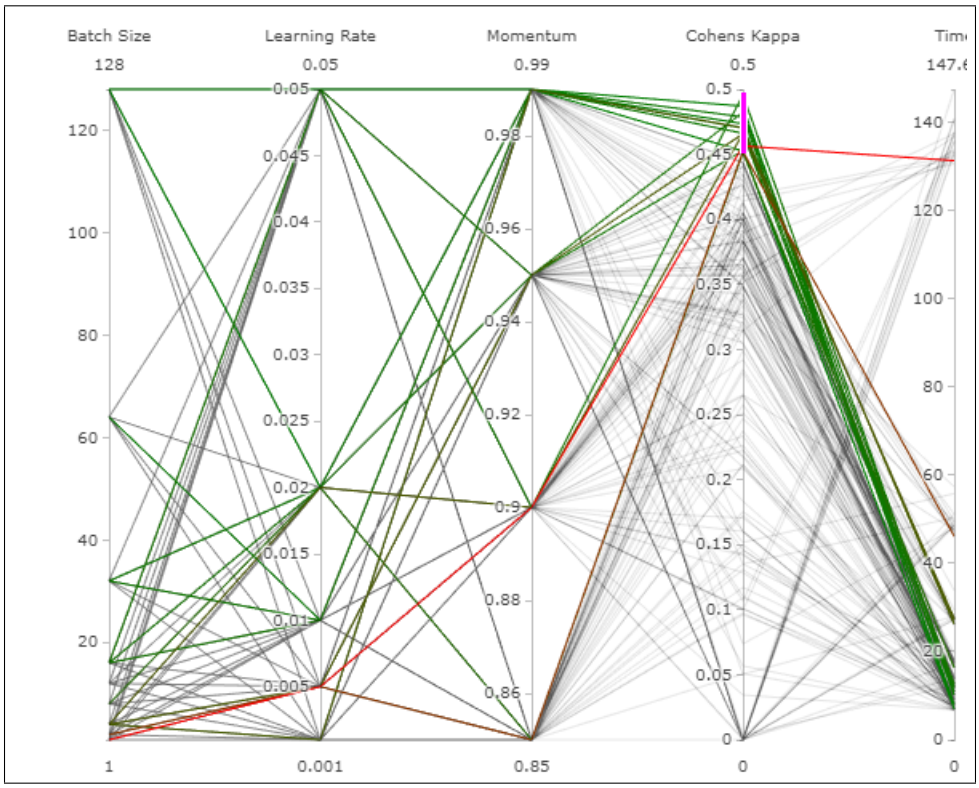



Figure 5.1: Parallel Coordinates Plot of the SGD (Hyper-)parameter Tuning experiment (Top 10% highlighted, based on 5-fold cross-validation).

Complementing the parallel coordinates plot, Table 5.1 shows the performance of the baseline and tuned optimizer. The table also indicates an additional optimizer, the *Adam* optimizer, which is known as a more effective algorithm than the due to its adaptive learning rate parameter. The *Improved SGD()* values are based on the best plot in the Figure 5.1 and represent the average performance over 5 folds.

As the table shows, our tuned SGD optimizer outperforms both the baseline SGD algorithm and the *Adam* optimizer. The most improved SGD parameter settings are: Learning rate = 0.05, Momentum = 0.95 and Batch Size = 16. The entire parallel coordinates plot and performance table can be found in Appendix C.1.

Table 5.1: Classification performance of the optimizer tuning experiment (based on 5-fold cross-validation).

| | **Accuracy** | **Recall** | **Precision** | **F1 measure** | **Cohen's kappa** | **Time (s)** |
|---|---|---|---|---|---|---|
| *SGD()* | 30.09% | 30.09% | 28.98% | 26.66% | 28.62% | 12.59 |
| *Adam()* | 33.74% | 33.74% | 36.60% | 32.71% | 32.49% | 13.90 |
| *Improved SGD* | 50.64% | 34.50% | 53.19% | 48.69% | 49.57% | 11.98 |

### The architecture

The second set of experiments are conducted to tune the architecture parameters of the Neural Network. Based on the range of values in Table 4.5, each experiment exists of a grid-search of 96 unique parameter settings for three parameters: The number of hidden layers (Hidden Layers), the number of neurons per layer (Neurons) and the drop out rate (Drop out). The results can be found in the parallel coordinates plot in Figure 5.2 and the performance metric scores are based on 5-fold cross-validation results.

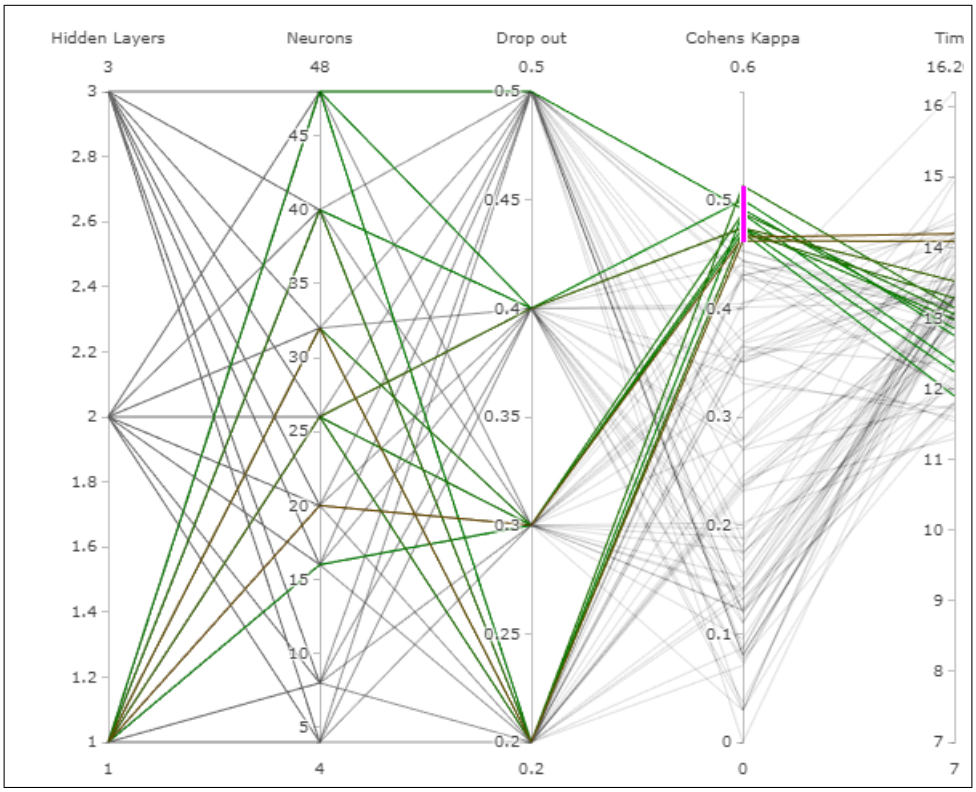



Figure 5.2: Parallel Coordinates Plot of the NN architecture parameter tuning experiment (Top 10% highlighted, based on 5-fold cross-validation).

From left to right, the first three vertical axes in the plot indicate the values of the three architecture parameters. The final two vertical axes indicate the scores of the two performance metrics. Once again, the plot highlights the top 10% individual lines based on the Cohen's Kappa score. Also, the computation time of each unique plot is indicated by coloring the lines on a green to red scale, whether the model was relatively fast or slow. The entire plot can be found in Appendix C.3.

The scores on the performance metrics of the architecture tuning experiment can be found in Table 5.2. This shows the performance of both the standard MLP architecture and our best parameter setting found over 5-fold cross-validation results. Based on the values in the table, the improved architecture settings outperform the standard setting on every performance metric. The most promising architecture parameter settings are: One hidden layer with 26 neurons and a Drop out rate of 0.2. Comparing this with the All experimental results of the architecture tuning experiment can be found in Appendices C.3 and C.4.

Table 5.2: Classification performance of the architecture tuning experiment (based on 5-fold cross-validation).

| | **Accuracy** | **Recall** | **Precision** | **F1 measure** | **Cohen's kappa** | **Time** |
|---|---|---|---|---|---|---|
| *Initial architecture* | 30.09% | 30.09% | 28.98% | 26.66% | 28.62% | 12.59 |
| *Best architecture* | 50.64% | 34.50% | 53.19% | 48.69% | 49.57% | 11.98 |

### 5.1.2 Decision Tree

The third experiment is conducted to tune the parameter settings of the Decision Tree classifier. The experiment consists of grid-search for three different parameters, which results in a total of 540 unique settings. The results are visualized with a parallel coordinates plot found in Figure 5.3. The vertical axes of the plot show the values of all three parameters and the scores on the two performance metrics. Each line in the plot indicates average results of an individual parameter setting over 5-fold cross-validation. Also, the color of the individual line represents its computation time relative to all other parameter settings, equivalent to the previous plots. The entire parallel coordinates plot can be found in Appendix C.5.

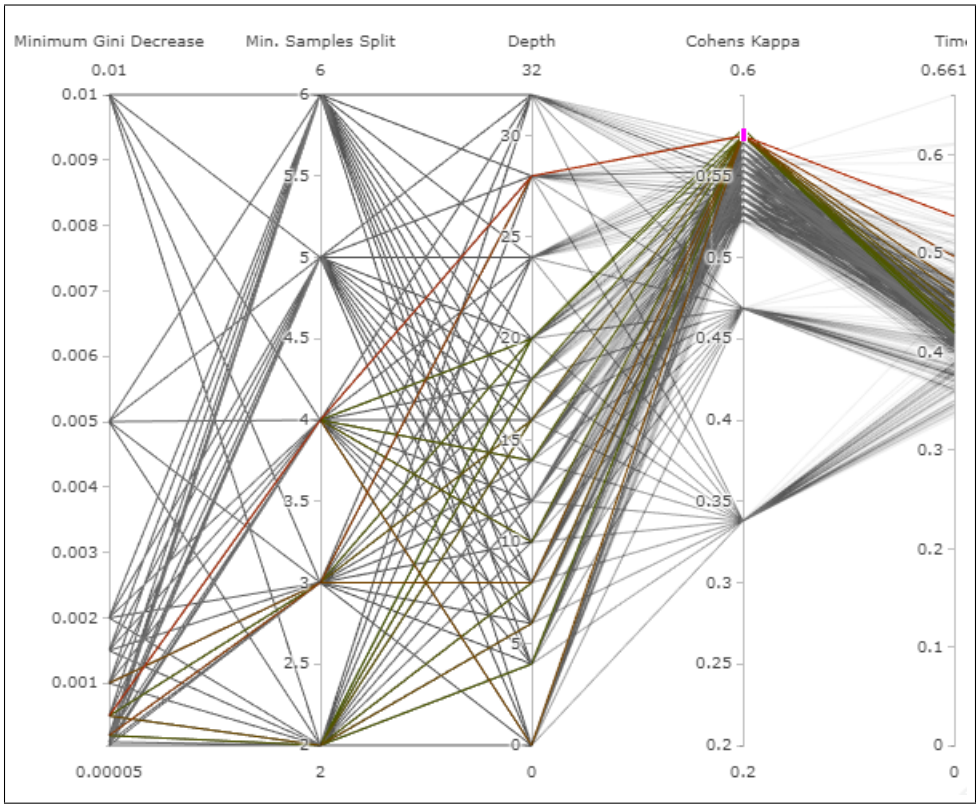



Figure 5.3: Parallel Coordinates Plot of the Decision Tree parameter tuning experiment (Top 10% highlighted, based on 5-fold cross-validation).

Complementing the parallel coordinates plot is Table 5.3, which shows the scores on the performance metrics of two Decision Tree models. The table compares the performance of the standard Decision Tree classifier and our Decision Tree model with the best parameter settings. The found settings have a maximum depth of 8 nodes, with a minimum of 3 samples required to split a node and an Gini Impurity decrease of $5 * 10^{-5}$. The improved parameter settings outperform the standard classifier on every performance metric, except the computation time. The individual results of the performance metric result for every parameter settings can be found in Appendices C.6 through C.10.

Table 5.3: Classification performance of the Decision Tree tuning experiment (based on 5-fold cross-validation).

| | **Accuracy** | **Recall** | **Precision** | **F1 measure** | **Cohen's kappa** | **Time** |
|---|---|---|---|---|---|---|
| *Standard DT settings* | 99.03% | 44.72% | 44.60% | 44.66% | 55.32% | 0.49 |
| *Best DT settings* | 99.08% | 47.68% | 47.27% | 47.47% | 57.91% | 0.45 |

### 5.1.3 Random Forest

The results of the final experiment are based on the parameter tuning of the Random Forest classifier. For this model, we tuned two parameters: the number of estimators *(Trees)* in the forest and the maximum depth of a single tree *(Depth)*. Based on the parameter settings, each experiment consists of a grid-search for 144 unique settings. The results are based on 5-fold cross-validation and visualized in a parallel coordinates plot in Figure 5.4.

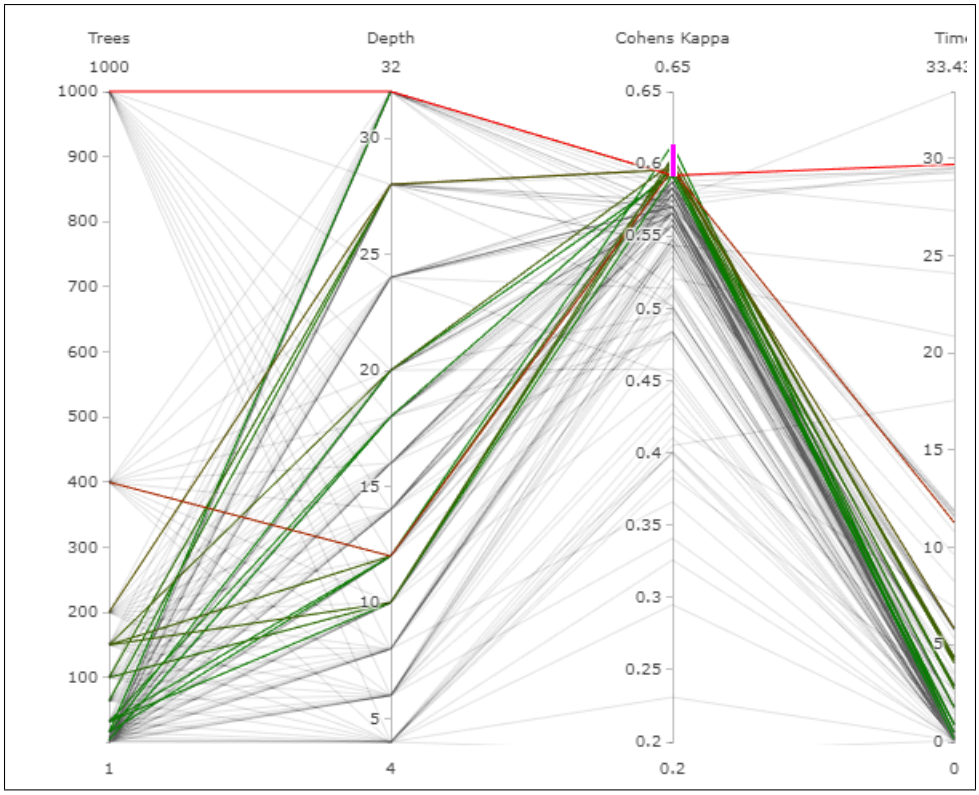



Figure 5.4: Parallel Coordinates Plot of the Random Forest parameter tuning experiment (Top 10% highlighted, based on 5-fold cross-validation).

The parallel coordinates plot has four vertical axes: The first and second axes represent the parameter settings for the number of trees and maximum depth respectively. The third and fourth axes are used to assess the performances of the plots by showing the results on the *Cohen's Kappa* and *computation time*. Also, the coloring of the individual plots is based on their computation time performance relative to each other and only the top 10% of all plots are shown due to their performance on the *Cohen's Kappa* metric.

Additional results of the Random Forest model can be found in Table 5.4. There are two models presented in this Table: The Random Forest classifier with its standard parameter settings and our Random Forest classifier with the best found parameter settings based on the grid-search experiment. Our proposed model obtained these performance metrics by setting the depth of a tree to 12 and the total number of estimators to 150. All individual performance metrics scores of each unique parameter setting can be found in Appendices C.12 and C.13.

Table 5.4: Classification performance of the Random Forest tuning experiment (based on 5-fold cross-validation).

| | **Accuracy** | **Recall** | **Precision** | **F1 measure** | **Cohen's kappa** | **Time** |
|---|---|---|---|---|---|---|
| *Standard RF settings* | 99.05% | 48.67% | 46.90% | 47.76% | 57.47% | 0.49 |
| *Best RF settings* | 99.10% | 51.37% | 54.34% | 52.81% | 60.04% | 4.05 |

### 5.1.4 Conclusion

This Section described the results of our first set of experiments: the hyper-parameter tuning of each learning model. Each learning model found a new parameter setting which resulted in an improved and stable performance. Our proposed parameter settings for each model can be found in Table 5.5. We can see that the parameter settings for each model differs from the standard/initial model provided by theory or Python library. This indicates that our grid-search experiments were profitable and we use these parameter settings for the remaining experiments in this research.

Table 5.5: The standard parameter settings compared to our proposed parameter settings.

| **Model** | **Parameter** | **Standard setting** | **Proposed setting** |
|---|---|---|---|
| Decision Tree | Maximum depth of tree | *None* | 8 |
| | Minimum samples per split | 2 | 3 |
| | Mininum Gini decrease | 0.0 | 0.00005 |
| Random Forest | Maximum depth of tree | *None* | 16 |
| | Number of estimators (trees) | 100 | 150 |
| Neural Network | Learning rate | 0.1 | 0.05 |
| | Momemtum | 0.9 | 0.95 |
| | Batch size | 128 | 16 |
| | Hidden Layers | 2 | 1 |
| | Neurons per Hidden Layer | Layer 1: 12<br>Layer 2: 6 | Layer 1: 26 |
| | Drop out rate | 0.5 | 0.2 |

## 5.2 Model evaluation

In this section, we use the best hyper-parameter settings for each learning model for our second set of experiments. This section is divided in several sub-sections, which all contribute to the evaluation of the most promising learning model. First, we provide an overview of each model's classification performance by discussing the confusion matrix in Section 5.2.1. Then, Section 5.2.2 discusses and visualizes the performance metrics used to evaluate the models. Finally, in Section 5.2.3, we discuss the obtained feature importance scores of each model to find the most relevant set of features in the replanning data.

### 5.2.1 Confusion matrices

To provide additional information to the calculations of the performance metrics, we first present the confusion matrices of all learning models. Since our models are used for multi-class classification on a total of 91 unique classes, we would obtain immense confusion matrices (of size 91 rows by 91 columns). This would hinder their explanation, so we simplified the visualization by creating 2x2 confusion matrices for each model. Each matrix contains two classes: *relevant* and *irrelevant*. This means that the classifier was able to predict a ride in either the correct (the relevant) or incorrect (the irrelevant) class. All four confusion matrices can be found in Table 5.6. Each cell in the confusion matrix represents one of the four classification outcomes (*TP*, *FP*, *TN* and *FN*).

| | | Predicted | |
|---|---|---|---|
| | | Relevant | Irrelevant |
| Actual | Relevant | 125 (TP) | 110 (FN) |
| | Irrelevant | 110 (FP) | 21040 (TN) |

(a) Decision Tree

| | | Predicted | |
|---|---|---|---|
| | | Relevant | Irrelevant |
| Actual | Relevant | 139 (TP) | 96 (FN) |
| | Irrelevant | 96 (FP) | 21054 (TN) |

(b) Random Forest

| | | Predicted | |
|---|---|---|---|
| | | Relevant | Irrelevant |
| Actual | Relevant | 103 (TP) | 132 (FN) |
| | Irrelevant | 132 (FP) | 21018 (TN) |

(c) Naïve Bayes

| | | Predicted | |
|---|---|---|---|
| | | Relevant | Irrelevant |
| Actual | Relevant | 114 (TP) | 121 (FN) |
| | Irrelevant | 121 (FP) | 20981 (TN) |

(d) Neural Network

Table 5.6: 2x2 Confusion Matrices over all classes of each learning models (based on 10-fold cross-validation).

Each matrix has a total of 21385 predicted sampling units. This number can be explained, because there are 91 unique classes (rides) present in the test set of 235 samples. As a result, the 2x2 confusion matrices in Table 5.6 are a summation of many, individual multi-class matrices. From the total 235 samples in the test data, the True Positive classification indicates the number of samples that were correctly predicted. Furthermore, the True Negatives indicate the number of classes that were correctly predicted as not the relevant class. Based on the results, the Random Forest has the highest number of both these outcomes, making it the most promising model. Although, the high number of sampling units are proof of the well-known Accuracy Paradox previously described in the literature. Each confusion matrix has a very large number of True Negatives (TN). This high value is expected, since each output class has so little data compared to all other output options. For example, in our test data, only three of the 235 elements have class 1. Meaning that, for this class alone, the minimum number of True Negatives is 232. Therefore, the total number of True Negatives explode, resulting in the falsely high performance accuracy. We take this into account when interpreting the performance metrics in the upcoming section.

### 5.2.2 Performance metrics

We determine the classification performance of each model based on their scores for the performance metric. The results are based on the calculations following from the equations in Section 3.3.4 and all results are visualized in Table 5.7. The results are based on a 10-fold cross-validation of the input data, to improve their overall stability.

Table 5.7: The performance metrics of all learning models (based on 10-fold cross-validation).

| **Model** | **Cohen's kappa** | **F1 measure** | **Recall** | **Precision** | **Accuracy** | **Time (s)** |
|---|---|---|---|---|---|---|
| Decision Tree | 63.75% | **52.39%** | 52.36% | 52.43% | 99.23% | **0.51** |
| Random Forest | **64.84%** | 50.84% | **54.39%** | **54.37%** | **99.25%** | 11.50 |
| Naïve Bayes | 46.28% | 42.32% | 44.12% | 40.67% | 98.80% | 1.83 |
| Neural Network | 49.23% | 46.71% | 50.21% | 50.31% | 50.21% | 17.89 |

These results show that each learning model performs relatively good on the complex and high-dimensional data. The Random Forest model outperforms all other models on every performance metric, except for the $F_1$ *measure* and computation time. For both these metrics, the Decision Tree achieved the highest score. It is however very promising that the Random Forest model scored higher on the *Kappa Score*, since this takes the imbalance of the data into account, which is the case for our study. Also interesting to acknowledge is that for each model, the *Kappa Score* is higher than the $F_1$ *measure*. This indicates that our data is imbalanced to certain classes, since the metric that takes this into account scores higher than others. We take a more in-depth look at this imbalance later in this section.

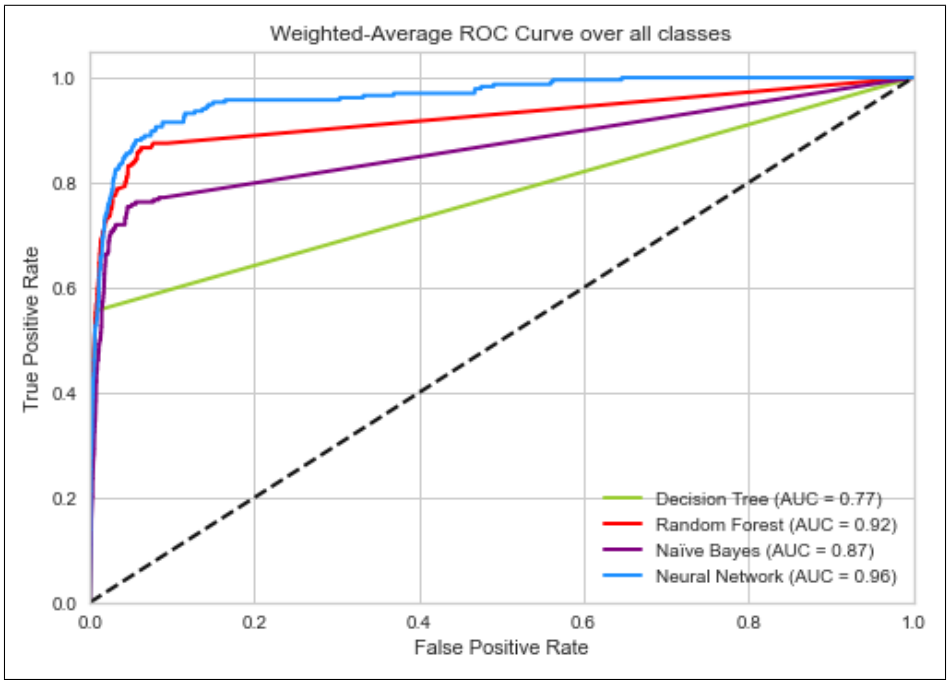



Figure 5.5: Receiver Operating Characteristic (ROC) curve of each models, with values based on weighted-average scores over all classes (based on 10-fold cross-validation).

Complementing the classification performance metrics in Table 5.7, we calculate the final metric: the Receiver Operating Characteristic (ROC). The comparison of each model's ROC space can be found in Figure 5.5. The plot shows the curves from each model, which is based on the weighted average scores for the True Positive Rate and False Negative Rate from all classes. These values are based on the results from 10-fold cross-validation.

Each model is represented by a different color and the dotted line is the so-called "line of no-discriminations". Basically, any point along this line is considered a random guess, making this line the baseline performance of the ROC. All four models have a ROC above the baseline performance, indicating good classification result. Also, all lines show a high "steepness", meaning they achieve a high True Positive Rate while keeping the False Positive Rate at a minimum. Complementing the curves are the Area Under Curve (AUC) values, which can be found in the legend of the graph. The AUC is always a value between 0 and 1, and a higher value indicates that the model is good at distinguishing between classes. The results show us that all AUC values are very good, with the Neural Network having the highest score of all. However, due to the imbalanced data, the quality of these values is damaged. A better measure for success when classes are imbalanced is the Precision-Recall curve, which can be found in Figure 5.6.

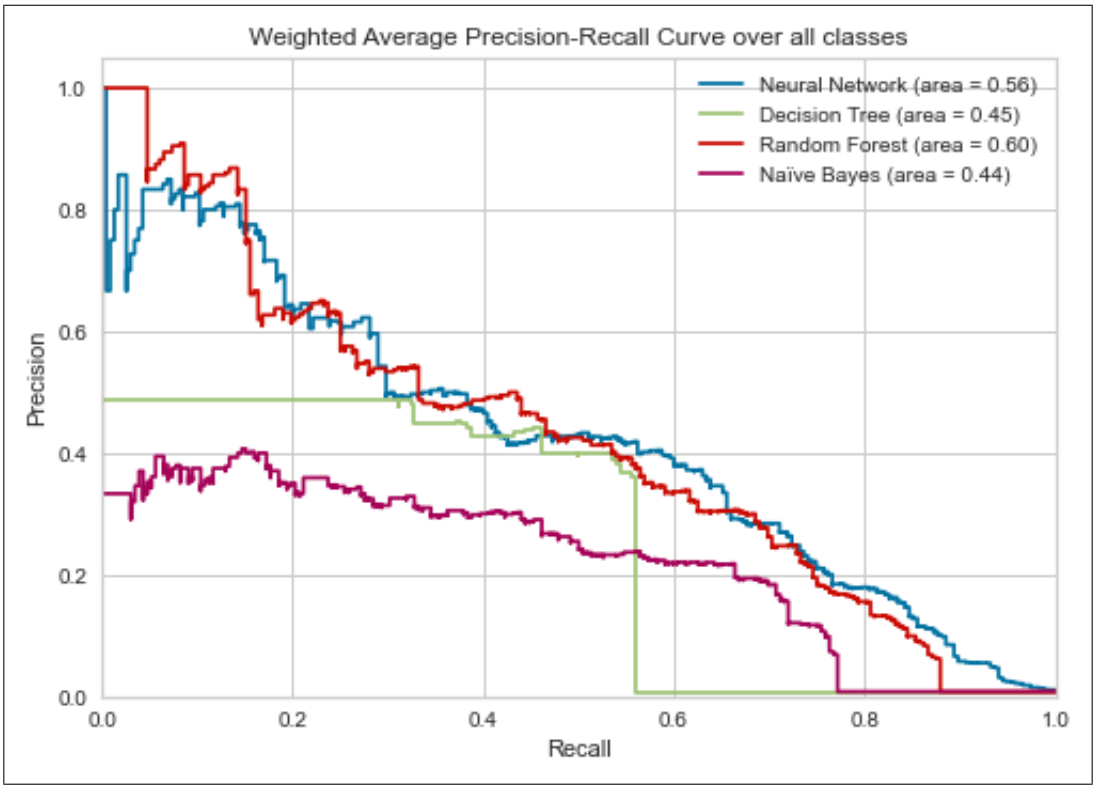



Figure 5.6: Precision-Recall Curve of each models, with values based on weighted-average scores over all classes (based on 10-fold cross-validation).

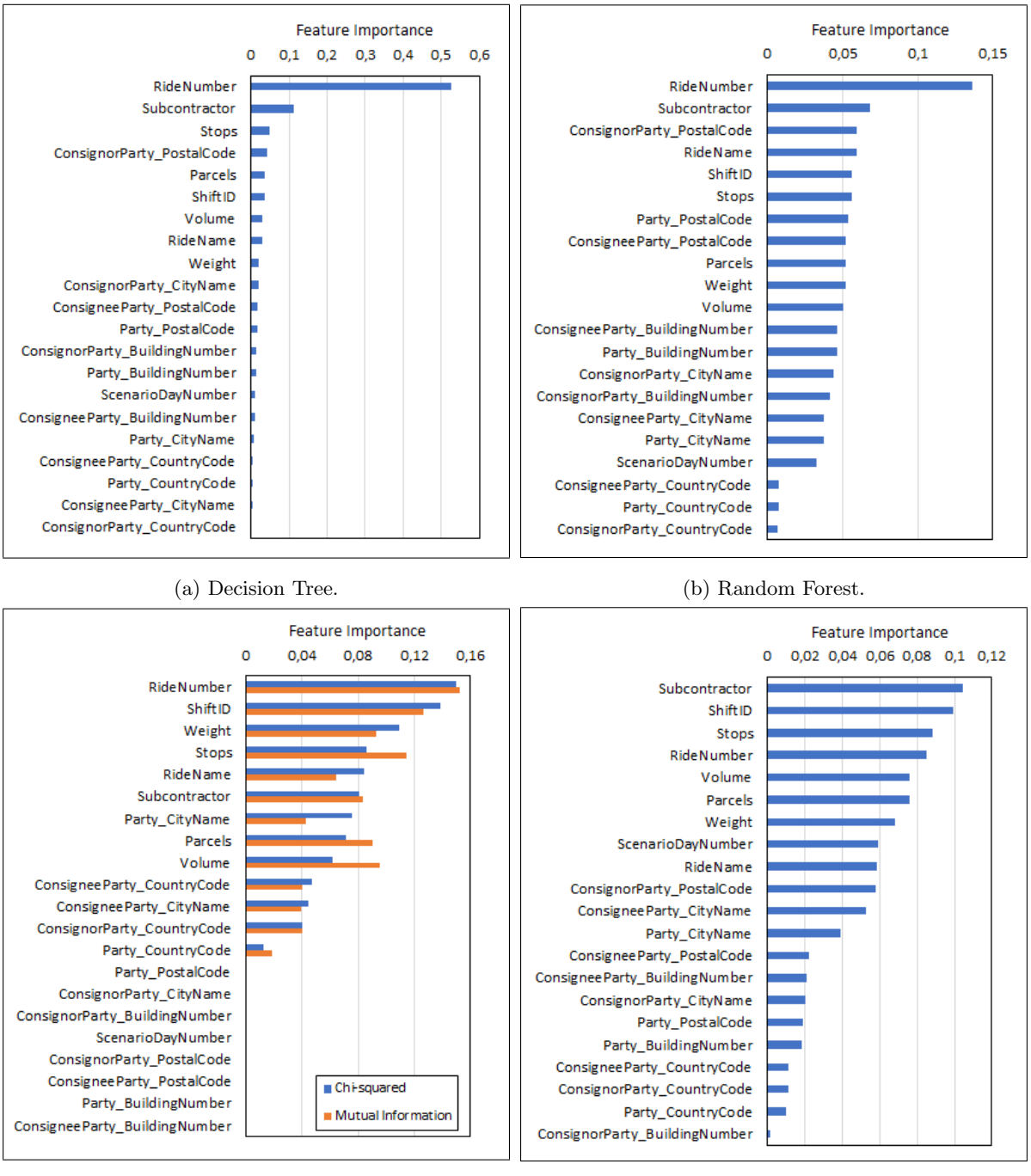
The Precision-Recall Curve (PRC) shows the trade-off between the precision and recall scores of different thresholds. The optimal classifier has a *concave down* curve from the top-left to bottom-right of the plot. The decreasing rate of the PRC should be increased as late as possible, meaning that the *recall* should increase a lot while keeping the decrease in *precision* at a minimum. The different thresholds affect the relationship between the precision and recall by inflecting small changes, which gives the plot its stair-step behavior. Small changes in the threshold considerably reduces precision, while slightly gaining recall. Each plot takes a total of 100 thresholds into account and the total area under the Precision-Recall Curve is calculated by the average precision (Equation 5.1).

$$AP = \sum_{n} (R_n - R_{n-1}) P_n \tag{5.1}$$

In the formula, $P_n$ and $R_n$ are the precision and recall at the $n^{th}$ threshold. The main difference between the ROC and PRC is that the latter avoids the use of True Negatives. It focuses more on the proportion of classes and then averages out the precision-recall score. The Average Precision for each learning model can be found in the upper right of the graph. Based on the results, the Random Forest model outperforms all other models, making it the most promising model based on this metric.

### 5.2.3 Feature importance

After assessing the performance of each model, we now discuss the feature importance results associated to the classification performance. The results of the Classifier Specific (CS) feature selection techniques can be found in Figure 5.7. The goal of this section is to determine which set of features rank highest across all models and compare the results of the different selection techniques.




(a) Decision Tree.

(b) Random Forest.

(c) Naïve Bayes.

(d) Neural Network.

Figure 5.7: The feature importance scores from the CS feature selection techniques of each learning models (based on 10-fold cross-validation).

Each model uses a specific feature selection technique. Since the feature selection for both tree-based models are based on the same intrinsic method, we compare these two feature scores to find consistency among the selection technique. Resulting feature scores for the Decision Tree and Random Forest model can be found in Figures 5.7a and 5.7b respectively.

Both these bar charts have the exact same top-2 feature ranks. Also, there is a big overlap in the top-6 feature ranks of both models, only the *RideName* feature is not present in both subsets. We should treat this feature with care, since the feature selection method is not able to deal with the correlation (called feature interaction) of *RideName* and *RideNumber*. Before we compare the top feature ranks with the other models, we first describe the results of Figures 5.7c and 5.7d.

The Naïve Bayes classifier uses two methods to calculate the best features for its classification: the Chi-square statistic and the mutual information score. Results for both methods can be found in Figure 5.7c. All values of the chi-square test are statistically significant ($p < 0.05$), which means that the importance values very likely describe the dependency between the feature and the output. The higher the chi-square value, the more impact the feature has. Also, the results indicate which features are left out of the model. These features obtained a score of zero, which indicate that they are independent of the output classes. Finally, the top-1 feature of this model matches the top feature of both previous models. This substantiates the importance of the *RideNumber* feature. Lastly, the feature scores of the Neural Network are calculated by using equation 4.8, previously described in Section 4.3.2.

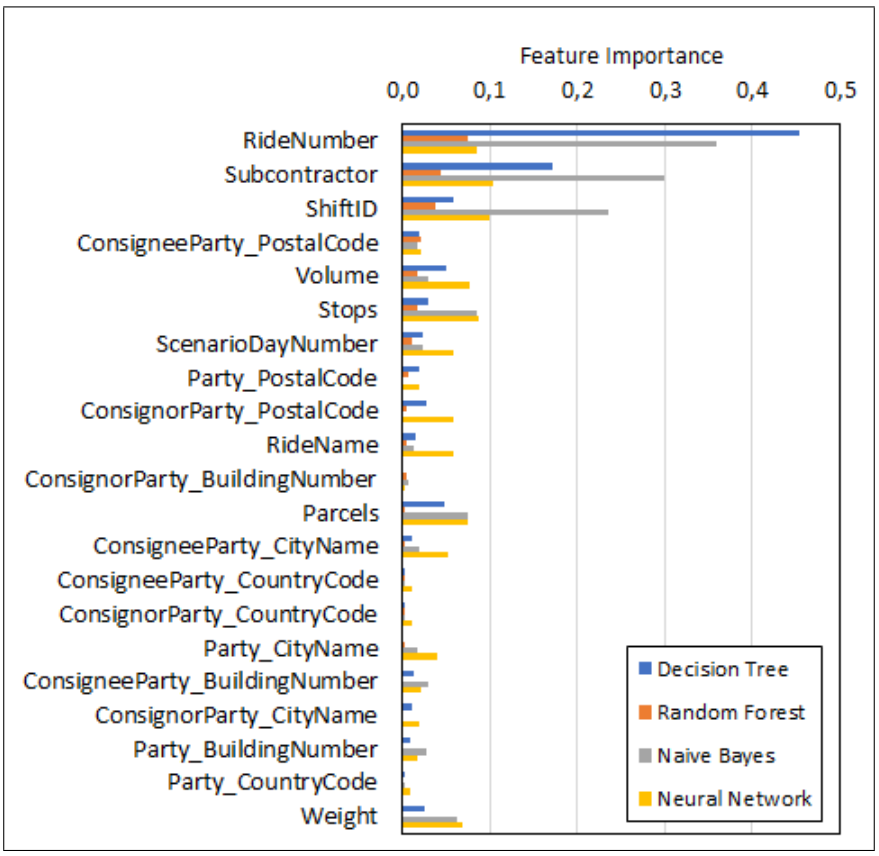



Figure 5.8: Comparison of the CA feature selection technique over all four models (based on 10-fold cross-validation).

The results are based on using a 100 permutations per feature and we assess their estimates on the $F_1$ *measure* scoring metric. The results in Figure 5.7d indicate the feature importance scores relative to each other. The top-6 feature ranks of the neural network show some recurrent features that were also found in the top-6 ranks for the Decision Tree and Random Forest model. Individual values of the CS feature importance scores for all models can be found in Appendices C.3.1.

To make a better comparison across the learning models, we implemented an Classifier Agnostic (CA) feature selection technique. This technique will be the same for each model, which aids us in finding a more reliable subset of features. We used the *Permutation Importance* technique to inspect the feature scores across the models, which is a Classifier Agnostic method and calculates the feature importance based on data permutations. We already implemented this type of feature selection for the Neural Network, so we extend the algorithm to the other models. The results can be found in Figure 5.8, which compares the feature scores of each model in a bar chart. All individual values of the CA selection technique can be found in Appendix C.3.2.

Table 5.8: The correlation coefficients of the feature importance scores from each classifier, based on the paired samples t-test.

| | *Decision Tree* | *Random Forest* | *Naive Bayes* | *Neural Network* |
|---|---|---|---|---|
| Decision Tree | 1 | | | |
| Random Forest | 0.838*** | 1 | | |
| Naive Bayes | 0.544 | 0.516 | 1 | |
| Neural Network | 0.408 | 0.589* | 0.745** | 1 |

(a) Classifier Specific methods.

| | *Decision Tree* | *Random Forest* | *Naive Bayes* | *Neural Network* |
|---|---|---|---|---|
| Decision Tree | 1 | | | |
| Random Forest | 0.897*** | 1 | | |
| Naive Bayes | 0.856*** | 0.921*** | 1 | |
| Neural Network | 0.501* | 0.622** | 0.71** | 1 |

(b) Classifier Agnostic method.

*Significance levels:* * $= p < 0.1$, ** $= p < 0.05$, *** $= p < 0.01$.

With this extension, we now have both a Classifier Specific and Classifier Agnostic method for each model. For both feature selection methods, we compute a paired t-test between the models to see if the found feature scores hold statistical significance. The results of both tests can be found in Table 5.8

Based on the correlation coefficients in Table 5.8a, there are two model combinations that have a good correlation in their feature importance scores. Not surprisingly, the tree-based models have a strong correlation, which was implicated in Figures 5.7a and 5.7b. Besides this combination, the Neural Network and Naïve Bayes classifier seem to have a strong correlation as well. This is interesting, since the feature selection methods use different approaches and the ranking results do not seem very equal. Looking more in-depth in the feature ranking, four out of the top-5 ranked features of both models are the same, despite their position.

More interesting to discuss are the results from Figure 5.8 and Table 5.8b. The correlation coefficients in the table show that all feature ranks obtain from the model appear to be statistically significant to each other. The Random Forest model has the strongest correlations with the other models, so the rankings in Figure 5.8 are ranked based on the feature importance scores from this classifier. If the feature importance scores are consistent over all methods, then we accept the feature as a critical factor of replanning. The top-7 ranked features are very consistent over all models, making a strong case that this subset is important in the replanning process. It is also safe to say that certain features are not relevant for the class prediction. This subset of features have very low importance scores across all four models and due to the stability of this result, these features are therefore rejected as an important replanning factor. Interesting to acknowledge is that a subset of features seem ether extremely important or unimportant, depending on the model. These features are *ScenarioDayNumber*, *Weight* and *Parcels*, and due to the inconsistency across the models, they are not recognized as important replanning factors.

### 5.2.4 Conclusion

This section illustrated the classification performance of each learning model with the tuned parameter settings. The results are based on 10-fold cross-validation and the Random Forest scored best on the performance metrics and could particularly predict good with the imbalanced data. After analyzing the confusion matrices, we introduced an additional performance metric that works great on imbalanced classes (the Precision-Recall Curve) and both the Random Forest and Neural Network showed great potential with a good AUC value. Furthermore, we compared two types of feature selection techniques for each model and found some similarities between the methods and the feature ranks. The most important subset of features (top-5 rank) are *RideNumber*, *Subcontractor*, *ShiftID*, *Stops*and *ConsigneeParty_PostalCode*. To look at this from a practical perspective, a combination of the current ride or shift, ride owner, number of stops and the recipient postal code are most likely indicating to which new ride the human planners need to replan an order. Additionally, there is also a subset of features which look not important in the replanning process. These features are *Party_CountryCode*, *ConsigneeParty_CountryCode*, *ConsignorParty_CountryCode*, *ConsigneeParty_BuildingNumber*, *ConsignorParty_BuildingNumber*, *Party_BuildingNumber* and *Party_PostalCode*.

## 5.3 Improving operational planning performance

In this section, we validate our found learning model by implementing it as an Artifact on the current planning system. First, we discuss some assumptions used to create the Artifact and describe the data and goal of the learning model. The Artifact consists of the most promising model based on Section 5.2.2 results and it uses the most important features based on Section 4.2.2 results. Second, we discuss the results found in our validation experiment of the Artifact and interpret their practical implications.

### 5.3.1 Assumptions and construction of the Artifact

In order to improve the operational planning performance, we created an Artifact to calculate an approximate benefit on the current performance. Since our study depends on experimental results that are being validated by a demonstration, we provide an estimation based on past planning

performances. We make some assumptions in order to create a more straightforward replanning process in order to better understand the improvement from the Artifact.

1. The total number of replanning actions is based on historic data (measured by the UAR). The Artifact is validated on a total of five working days. For each day, we calculate the average number of adjustments from the same day which can be found in the UAR data. This average is used to set the number of replanning actions of our scenarios.

2. The handling time of one adjustment by the human planner differs per size of the order. Since we do not have any that recorded the duration of a single adjustment, we set the time to manually make one replanning based on a normal distribution with a mean of 90 seconds and a standard deviation of 30 seconds. This would be a realistic estimate that takes the experience of the human planner, size of the order and processing time of the OOMPD into account.

3. The incoming alerts containing the replanning adjustments can arrive at any moment during the day. To assess the Artifact's performance on the waiting time, we assume that all adjustments are received within the first two hours of their work shift based on a standard uniform distribution.

With these defined assumptions, we will describe the construction of the Artifact. Essentially, the Artifact is the Random Forest model (with optimized parameter settings) found in Section 5.1.3, which will be called to predict the total number of replanning adjustments (and their new ride) based on that day's initial planning. We trained the model with data from a two-week period of August, which were collected with the UAR. Knowing the practice of the Artifact, there are two different situations to compare: Manual replanning adjustments by the human planners and predicted replanning adjustments with our proposed Artifact. To test the performance of both situations, we take the next five days (so five different plannings) after the data used for training. These five days represent five unique scenarios, which differ due to the total of orders and assumed replanning actions. Per day, we determine the number of adjustments based on historic data (discussed in assumption 1) measured by the UAR. To provide a clear overview, we visualized each daily scenario with the total number of adjustments in Table 5.9.

Table 5.9: Total number of predicted and correctly classified replanning adjustments by the Artifact.

| **Scenario** | **Day 1** | **Day 2** | **Day 3** | **Day 4** | **Day 5** |
|---|---|---|---|---|---|
| Total number of adjustments | 109 | 117 | 135 | 154 | 144 |
| Number of adjustments predicted | 71 | 63 | 65 | 69 | 76 |
| Number of adjustments correctly classified | 40 | 37 | 56 | 52 | 42 |
| Average time (in seconds) | 0.23644 | 0.21963 | 0.19227 | 0.20658 | 0.21157 |

The first row in the table indicates the estimated total number of adjustments on each day. The second row shows the total number of adjustments that the Artifact predicted. this indicates whether the Artifact predicted any replanned rides based on the data. The third row indicates the total number of adjustments that were also correctly classified (the correct ride). This indicates the actual number of replanned rides the Artifact was ably to predict correctly and is therefore the more useful outcome. The final row shows the average computation time of the learning model to calculate the replanning adjustments. These values will be later used to calculate the benefit of the Artifact.

### 5.3.2 Benefit on the planning performance

To calculate the estimated benefit of the Artifact, we first need to calculate the total replanning time of both the manual human planner adjustments and the Artifact solution. First, the time to manually adjust one replanning by the human planner is based on the assumption of a normal distribution ($X \sim N(90, 30)$). Also, the human planners first have to wait for the incoming alert to arrive in the system. The waiting time is based on a uniform distribution between 1 and 120 minutes ($X \sim U(1, 120)$) for each predicted planning adjustment. Therefore, the total replanning time for human planners ($T_{HP}$) is simply the total daily adjustments multiplied ($N$) by the random value from the normal distribution ($D_T$) plus the total waiting time of all replanning adjustments ($W_T$).

$$T_{HP} = N * (D_T + W_T) \tag{5.2}$$

The total replanning time of the Artifact ($T_A$) is defined by three parts: the computation time of the model, the handling time by the human planners on the incorrect (false positive) replanning adjustments and the additional waiting time of these remaining adjustments. The formula can be found below and we will discuss each parameter more in detail.

$$T_A = T_C + (N - P) * (D_T + W_T) \tag{5.3}$$

First, the computation time ($T_C$) of the learning model is a fixed value for each day, which can be found in Table 5.9. The computation time is found by running the Artifact in a test environment of Python, where the daily input is loaded and the learning model is callable from an external file. Secondly, there remain a number of replanning adjustments that the human planners still have to perform after the predictions of the Artifact. The number of correctly predicted adjustments ($P$) can be found in Table 5.9, so the remaining number of adjustments ($N$-$P$) can easily be calculated. The time to handle these remaining adjustments is based on the equation of the human planner adjustments ($T_{HP}$), because these adjustments have to be replanned manually after the model. The final part of the Artifact's replanning time is the additional waiting time on remaining adjustments after the model's prediction. Once again, we assume that the remaining adjustments have an uniform distribution ($W_T$). With both formulas known, we calculated the total replanning time of both situations on the five different days and all results can be found in Table 5.10.

Table 5.10: Estimated benefit (decrease in replanning time) by the Artifact.

| **Scenario** | **Day 1** | **Day 2** | **Day 3** | **Day 4** | **Day 5** |
|---|---|---|---|---|---|
| $T_{HP}$ (in minutes) | 291,39 | 330,81 | 369,92 | 394,24 | 441,37 |
| $T_A$ (in minutes) | 184,46 | 226,20 | 216,47 | 261,12 | 312,64 |
| **Decrease (in percentage)** | **36,70%** | **31,62%** | **41,48%** | **33,77%** | **29,17%** |

The table provides us an estimated benefit of the Artifact on the planning performance. Based on the daily performances, the Artifact decreases the total replanning time on average over all scenarios by 110 minutes (approximately 30.61%). Therefore, the estimated benefit of the Artifact is validated and improves the operational planning performance. Nevertheless, numerous assumptions had to be made for this calculation which slightly affect the interpretation of the practical significance. However, it is interesting to mention that the decrease in time is mainly based on the trade-off between the computation

time of the model and the number of adjustments that were correctly predicted. Since the computation time of the Artifact ($T_A$) is really low compared to the manual replanning process ($T_{HP}$), the benefit should be beneficial even if the correctly predicted adjustments are low. The only way for the Artifact to have a negative impact on the planning performance is when all predicted adjustments are incorrect or the computation time of the model would increase drastically.

### 5.3.3 Conclusion

This section introduces the estimated benefit of the Artifact, which is used to calculate the operational planning improvement of our research. We defined a set of assumptions in order to calculate the total replanning time of both the manual and Artifact scenarios. We tested both scenarios on a total of five days in order to find a stable benefit of our model. To calculate the replanning times of both scenarios, we proposed two formulas and discussed their parameters. After calculation, the estimated benefit of our Artifact is around 110 minutes, which decreases the total replanning time by approximately 30.61%. Even though there were some assumptions taken into account, the Artifact should have an positive impact on the replanning process due to the computation time and prediction performance trade-off.

# Chapter 6

# Implementation of Decision Support

This chapter is divided into two sections. First, we introduce the concept and implementation steps to integrate the UAR, learning model and decision-support system (DSS) in Section 6.1. This provides the foundation for CAPE Groep to implement our research and contributions to other cases and customers. We discuss our main implementation steps based on our methodology in Section 6.2. Finally, in Section 6.3, we propose an initial DSS based on the results of the Artifact.

## 6.1 Integration steps

To provide lasting change in the Client's operational planning process, we provide a series of steps to integrate our methodology and findings properly. We provide an overview in Figure 6.1 and we will describe the process in the sections below.

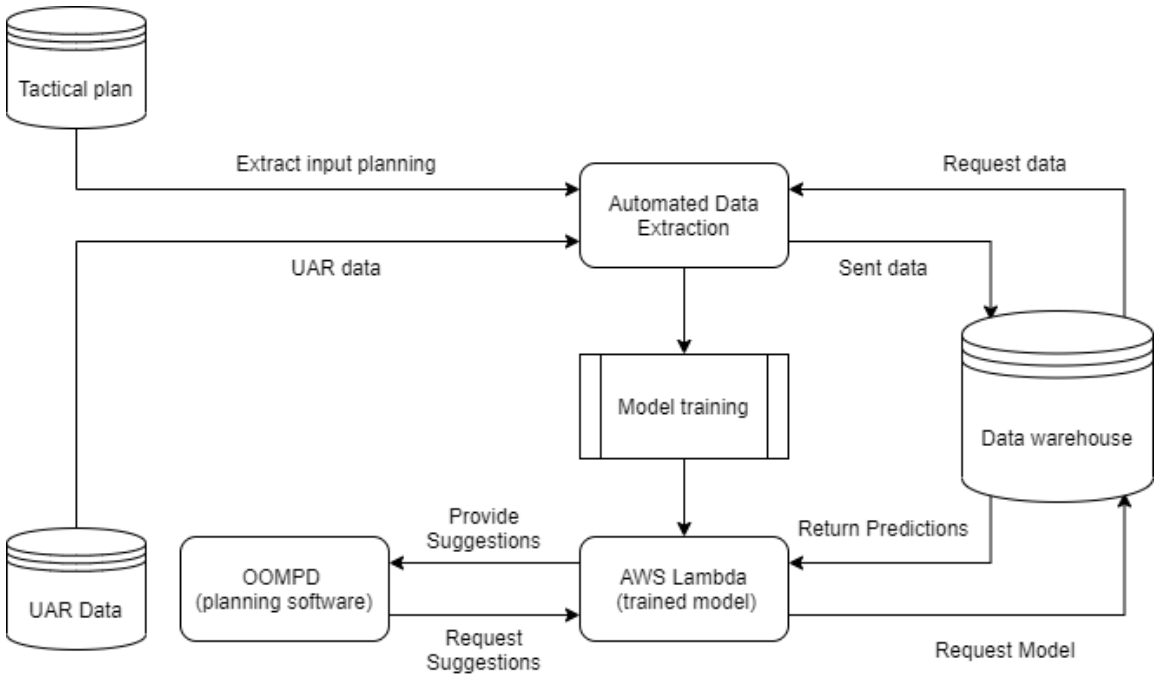



Figure 6.1: Overview of our proposed integration steps to create the decision-support system.

The integration starts with the extraction of several data files. We extract a total of four data files from two servers: The input planning needs to be extracted from a separate server, while the UAR with the supplementing files can be extracted from the operational planning software. To make this extraction more practical, we created an automated program that calls for these data files and saves them to data warehouse every day.

Between the OOMPD and the data warehouse lies a critical component in the integration, which stores our proposed learning model. For our integration, we opt the computing platform *AWS Sagemaker*. We can store our best learning model with tuned parameters on this platform. The model uses the data warehouse as its input and is triggered when the human planner calls for a suggestion in the OOMPD application. The final element in Figure 6.1 is the Model Training process, which is visualized in Figure 6.2. which is based on our research methodology and follows the same four stages found in Figure 4.1 in the Solution Design chapter. Each stage will be briefly discussed how they can be correctly generalized and implemented in this case and other future cases.

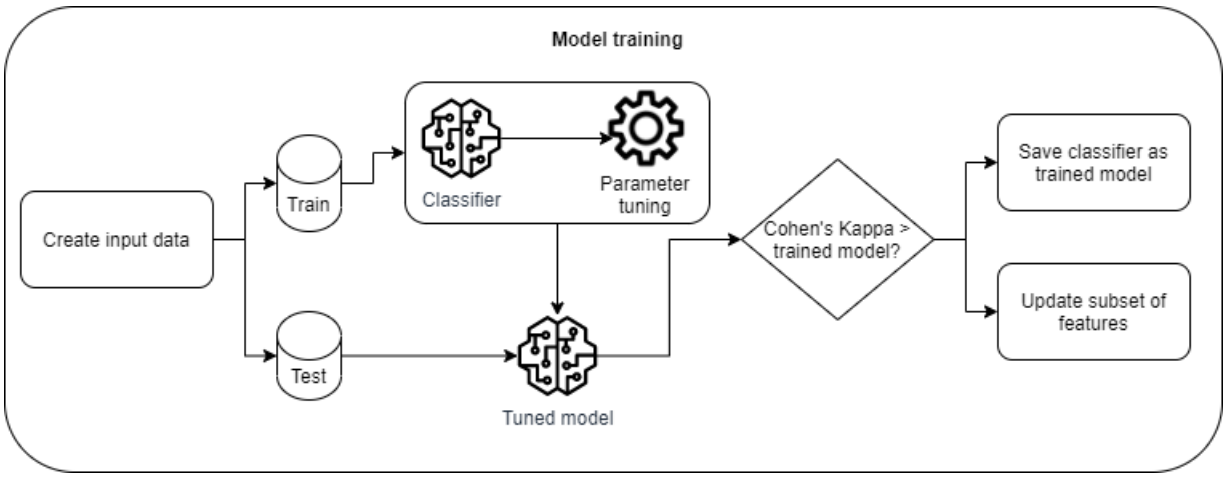



Figure 6.2: Visualization of the Model Training process in the DSS integration

## 6.2 Methodology implementation

The first stage of the methodology are the data preprocessing steps. As described earlier, daily exports will contain multiple data files associated with actual human planner adjustments. The data files are associated with the specific OOMPD, so to implement this step at other cases would require the User Action Recording to be implemented as well. The query to convert the individual data files into a viable input data set for the learning models in coded generically, meaning that all data preprocessing steps, feature creation techniques and output encoding steps should work similar files from other applications. it would be beneficial to improve the UAR for future installments, since a lot of the raw data (around 16%) is lost due to missing or duplicate values. If necessary, List wise deletion and interpolation seem appropriate methods to inspect and clean the data. The remaining input data should be split into three subsets for training, validating and testing, with the proportions being 64%, 16% and 20% respectively.

When all data is extracted and preprocessed into viable input data, we can train the learning models. We first conduct the hyper-parameter tuning experiments to find appropriate settings for each model. The tuning experiments in our research are based on grid-search, but other types of parameter tuning

experiments like random search could be used as well. We opt for using 5-fold cross-validation when conducting he hyper-parameter tuning experiments, since this aids in finding stable performance.

Assuming that the data is still imbalanced, the performance should be assessed primarily on the Cohen's Kappa score. It could also be useful to take the computation time into account, which could be a more critical factor when using the learning model in a fully integrated system. When the model outperforms on the most critical performance metric (we propose Cohen's Kappa), two critical steps should follow. First up, this model should be saved as the new and improved trained model. Secondly, it could be beneficial to reassess the feature ranks from this model to the existing subset of important replanning features. Using some statistical tests like the paired samples t-test or Kendall's tau rank correlation, the subset of feature can be reformed if they are statistically significant.

## 6.3 Creating the DSS

The decision-support system that we opt for the Client is based on the practical significance of the Artifact. The results from the Artifact indicate that the planning performance can be decreased by approximately 30.61% by letting the model predict the number of replanning adjustments. This improvement is, in our perspective, too low to create a DSS that automates the entire replanning process. Therefore, we opt for a system that provides insights and suggestions of replanning instead of automation. To aid the human planners, the decision-support system should be an additional feature inside the OOMPD application which highlights the orders that needs to be replanned and provides the suggested ride it should be adjusted to (based on the Artifact's prediction). To create this system, the Client should first implement the integration steps provided in this chapter. Then, the DSS can be created as minimum viable product inside the OOMPD. This product should use the Artifact and provide replanning suggestions on the daily planning as a list to the human planner

## 6.4 Conclusion

In this chapter, we introduced the implementation steps for the Client to use our solution design and methodology in practice. First, we discussed the integration steps needed in order to connect the Artifact within the current planning application and the necessary databases. We opt for a computing program like *AWS Sagemaker* to store the Artifact as an external file, which can be called by creating endpoints with the OOMPD. The data from the UAR should be stored in a cloud storage (i.e. *Amazon S3*) in which the UAR can export the recorded replanning adjustments and the Artifact can import the training data.

We also discussed how our methodology can be implemented in the future by the Client. The necessary data preprocessing, parameter tuning experiments and model evaluation steps are visualized in Figure 6.2. Finally, we provided our insights and steps to create a DSS for the human planners. The DSS should be based on the Artifact's performance and can be created as a minimum viable product, in order to establish the potential for the human planners. We propose to focus on a suggestion-based replanning approach by the DSS, since the improvement on the replanning process was too weak for an automated mechanism.

# Chapter 7

# Conclusions

This chapter will conclude the research by answering the main research question and its sub-questions. These questions are stated in Section 1.3 and we answer each of the question by the information provided in Chapters 2 to 6. Every chapter contains individual conclusions, which we will briefly restate in the remainder of this paragraph. First, we provided the context of our research by describing the current replanning process of the Client and proposed the UAR in Chapter 2. In Chapter 3, we reviewed the literature from several scientific fields and introduced Supervised Learning models to classify recurrent patterns and the Data Analytic techniques to overcome important challenges. Chapter 4 proposes our solution methodology that describes the preprocessing steps of the UAR data, the creation of our learning models and the experimental settings. In Chapter 5, the results of our experiments and the classification performances are provided. Finally, we provided the implementation steps to integrate the Supervised Learning model with the real-life application in Chapter 6.

We conclude all of our findings in Section 7.1. Then, we interpret the meaning and relevance of our findings in Section 7.2. Also, we critically assess the validity of our found results and propose different options to discover for further studies. We end this chapter by providing several recommendations based on our findings in Section 7.3.

## 7.1 Conclusion

The objective of this research was to incorporate a new data-driven approach in an existing planning application at the Client. We formulated the following research question in Section 1.3 to help us reach this objective:

*"How can we improve the decision-making in operational planning by classifying the most important features and patterns with a pattern recognition learning algorithm on actual replanning data?"*

This research opened-up the decision-making process between tactical and operational planning at a large logistics service provider. Data is collected with the User Action Recording UAR, which measures and stores actual replanning data made by human planners. We developed a Supervised Learning algorithm that can classify the most important features in the obtained data and to classify replanning patterns. We propose an initial sequence of steps to integrate the learning model into the daily

operations of the Client, by providing decision support to the human planners and to create a feedback mechanism between the operational and tactical planning. This will contribute to the complex "black-box" decision-making process and will be a foundation to the integration of AI and operational planning.

Our proposed methodology showed that the use of actual data can be properly processed into input features and output values for machine learning algorithms. In our data preprocessing phase, we used methods like *One-Hot-Encoding* to create a clean input dataset for our learning models. Also, we created features from the raw data by designing generic algorithms to convert strings into unique numeric values. These data preprocessing steps are generically formulated, so they can be used easy for future recommendations and implementation steps.

With our solution design, we compared four different Supervised Learning algorithms to classify: a single Decision Tree, Random Forest classifier, Naïve Bayes (Gaussian) classifier and a Neural Network. We proposed the most fitting initial settings, features selection techniques and discussed why each model should perform effectively. We incorporated new performance metrics into the learning models, to keep challenges like the Accuracy Paradox of imbalanced data into account when we assess the classification performance. We have explored each model by using various experimental settings. First, we tuned the hyper-parameters of the learning models with grid-search experiments with 5-fold cross-validation find stable parameter settings. With tuned parameters for each model, we conducted experiments to validate the classification performance of all four models. We used 10-fold cross validation to improve the stability of the classification performance. We succeeded in implementing every model correctly on the collected data and our results indicate that the Random Forest classifier is the best prediction model. This model obtained the highest value on five of the seven performance metrics: Cohen's Kappa (64.84%), Precision (54.37%), Recall (54.39%), ROC AUC (0.92) and Accuracy (64.84%). Also, the neural network showed some promising results without being too computationally expensive, which is something that other studies like Xhemali et al. (2009) were unable to find.

The experimental results also discussed the outcomes of two feature selection techniques. We compared the Classifier Specific and Classifier Agnostic methods and found some (statistically significant) results regarding the feature importance ranking. Within the Classifier Specific methods, some differences are found within the top-5 ranked features across the models, while the top-3 ranked features seem to be in agreement. The Classifier Agnostic method is implemented to counter feature interactions and resulted in a very stable top-5 ranked features across the models based on paired t-test significance tests. It also provided a subset of features that were consistently unimportant during classification.

Based on our findings, we created an Artifact to calculate the estimated benefit on the planning performance. The current planning process was defined with three assumptions and we tested two scenarios of replanning: manually (by the human planners) and automated (by the Artifact). The Artifact decreased the average replanning time by approximately 30.61%, improving the current planning process and implying the practical significance of our technical solution.

To set the foundation of creating lasting change for the Client, we proposed a series of implementation steps to design a decision-support system for the operational planning software. All necessary steps and required components (i.e. Python files) are all generalized and ready to be used on other cases of the Client as well. The DSS consists of an integration from used methodology elements

in this research like the UAR and the Random Forest classifier, by using computing tools like *AWS Sagemaker* and *Amazon S3*. To aid the human planner, we opt to create a minimum viable product that provides the identification of replanning actions and suggests the ride each action should be replanned to.

## 7.2 Discussion

We will structure our discussion by assessing three types of validity that challenges the strength of our findings: The structure of our study (*internal validity*), the applicability of our findings (*external validity*) and the degree of measure relevance (*construct validity*). The remainder of this section will tackle these validity challenges in the same order they are introduced.

Our study finds patterns in actual replanning data by comparing different types of Supervised Learning models and feature selection techniques. The different models try to predict the correct Ride and the performance of the models is assessed with multiple performance metrics and cross-validation. Our results overcame barriers found in previous studies like Xhemali et al. (2009) by being able to properly implement a neural network classifier. The results in our study are promising and stable, but the complexity of the classification could be a threat to the validity. Applying our models on a less high-dimensional problem (i.e., using Shift as the output classes), could provide some new insights. Also, due to the presence of labeled data, this study compared deterministic and probabilistic Supervised Learning classifiers. It could be beneficial to tackle the problem in our study by introducing an Unsupervised Learning approach like Support Vector Clustering or Principal Component Analysis, to attempt in finding natural clusters within the data.

If we want to go one step further, the implementation of a Reinforcement Learning model can provide insights on the sequential decisions that the human planners make in the replanning process. We propose to create a model-free policy gradient algorithm, since we are interested in finding the optimal policy that maximizes our expected rewards, given a current state (which is the initial planning). It is however critical to properly define the states and environments for this model, since the number of possible plannings can explode to extremely high numbers.

In our study, we used a dataset containing real-life replanning adjustments from one DC during a specific period in time, which were recorded with the UAR. This affects the external validity of our findings and the conclusions and practical implications that follow. The quality and availability of the data need to be further investigated in order to find more stable solutions. For example, we discussed a lot of delimitations in the introduction. These limitations are acknowledged and our findings represent a limited scope. The most prominent limitation would be the time frame of our study, which is an exception compared to the normal operations of the Client due to the influence of COVID-19 and the summer holiday period. To improve the external validity, more data should be collected during a longer period of time to try and replicate the same findings. Mentioned previously, this research focused on a single case study based on one of the DCs at the Client. The strength of our findings from the Artifact are based on several factors (recorded data, number of rides, human planner handling time, etc.) of this single DC. External validity can be improved by implementing the UAR, learning models and DSS at other DCs, which should provide similar results compared to this case study. Also, fitting our methods on more depots could find inter- and intra-DC relations or patterns. These can contribute to the generalizability of our solution, making it more relevant for practical implementations.

Considering the construct validity, we used several grid-search experiments to tune the hyper-parameter settings for three of the learning models. These experiments all found improving classification performances, but we did not conduct a grid-search experiment for the Naïve Bayes classifier due to the limited number of parameters. Combined with the imbalanced data, this resulted in the lower scores on the performance metrics compared to the other models. We can improve the construct validity of this model by introducing an additional step to handle little evidence of data, like the Believed Probability calculation of Xhemali et al. (2009).

Furthermore, the learning models were trained on a relatively small dataset, which resulted in the lack of all output classes in the test data. More precisely, not all 300 rides from the OOMPD were present in the data used for the experiments. Almost two thirds of all rides were not measured, which affect the prediction decisions of the models. Since these rides were missing in the data, the models are trained to not predict these rides as possible replanning actions. Additional experiments should be done with all possible output classes in order to improve the construct validity of the models.

## 7.3 Recommendations

The first recommendation is to update the UAR and to keep collecting replanning data in the future. Updating the UAR will enable measuring more variables from the application and also it would decrease the total number of faulty data, which was now around 16% of the raw data. Collection more data for a longer time period can result in finding additional replanning features like seasonal or trend patterns, overcome the internal validity challenges and could be the foundation towards replanning automation.

After testing our proposed methodology as an Artifact on a local test environment, we recommend to implement the steps provided in Chapter 6. This will create the decision-support system for the OOMPD application and sets up the integration between the data extraction from the UAR, model training and replanning prediction. Implementing the DSS as a minimum viable product would be beneficial in two ways: it would avoid lengthy work of the human planner and provides feedback for future DSS development. When implementing, it is recommended to keep observing the deviations and impact on replanning time from the suggested adjustments. This enlighten the used assumptions of the Artifact and can be used to counter the model for performing worse over time. Also, we recommended to conduct these implementation steps at other DCs as well, which provides a better understanding of our results' external validity across a multi-case study.

Based on our results, we opted to use the Random Forest classifier as the prediction model for the DSS. It is however recommended to also keep training the Neural Network, due to its promising results and the expectation that more input data would benefit this deep learning algorithm over the Random Forest classifier.

To find more practical relevance, future studies could try and use our proposed learning model on different classification problems. For example, using the possible Shifts as prediction labels for a multi-class classification problem (significantly less high-dimensional than the possible Rides) or try and find which Rides are replanned on existing Shifts and reserve Shifts (binary classification). The latter could make an interesting business case since replanning on a reserve Shift costs more resources than the existing Shifts.

An extra step to take would be to create a system that provides feedback to the tactical planning based on the findings of the UAR. This mechanism would convert the recorded replanning adjustments

into improvements on the input planning that is used by the human planners. This is not investigated in our research, but it could be beneficial to adjust the creation period of the tactical planning. In our study, the planning remained constant for a monthly period, even though the same adjustments would be encountered by the human planners. A great step to consider is to adjust the tactical planning two times in the monthly period, with the data collected from the UAR.

# Bibliography


Al Hajj Hassan, L., Mahmassani, H., and Chen, Y. (2020). Reinforcement learning framework for freight demand forecasting to support operational planning decisions. *Transportation Research Part E: Logistics and Transportation Review*, 137.

Alpaydin, E. (2014). *Introduction to Machine Learning, third edition.* Adaptive Computation and Machine Learning series. MIT Press.

Altman, N. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185.

Arel, I., Rose, D., and Karnowski, T. (2010). Deep machine learning-a new frontier in artificial intelligence research. *IEEE Computational Intelligence Magazine*, 5(4):13–18.

Asht, S. and Dass, R. (2018). Pattern recognition techniques: A review. *International Journal of Computer Science and Telecommunucations*, 3:5.

Aznar, P. (2020). Decision trees: Gini vs entropy.

Babuska, R. (2012). *Fuzzy Modeling for Control.* International Series in Intelligent Technologies. Springer Science and Business Media.

Bajcsy, R. and Kovačič, S. (1989). Multiresolution elastic matching. *Computer Vision, Graphics, and Image Processing*, 46(1):1–21.

Bellman, R. (1957). *Dynamic Programming.* Rand Corporation Research Study. Princeton University Press.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.

Breiman, L. (2001). Random forest. *Machine Learning*, 45(1):5–32.

Brownlee, J. (2016). *Machine Learning Algorithms From Scratch with Python.* Machine Learning Mastery.

Brownlee, J. (2018). *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions.* Machine Learning Mastery.

Brownlee, J. (2019). How to choose a feature selection method for machine learning.

Brownlee, J. (2020). *Imbalanced Classification with Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning*. Machine Learning Mastery.

Brownlee, J. (2021). How to choose an activation function for deep learning.

Choy, K., Chow, H. K., Tan, K., Chan, C.-K., Mok, E. C., and Wang, Q. (2008). Leveraging the supply chain flexibility of third party logistics – hybrid knowledge-based system approach. *Expert Systems with Applications*, 35(4):1998–2016.

Dreyfus, S. (2002). Richard bellman on the birth of dynamic programming. *Operations Research*, 50:48–51.

Duda, R., Hart, P., and G.Stork, D. (2001). *Pattern Classification*. ResearchGate, 2nd edition.

Englebienne, G. (2020a). Machine learning 1 lecture 1: Introduction. Presented as the 1st lecture of the Machine Learning I course, University of Twente.

Englebienne, G. (2020b). Machine learning 1 lecture 8: Dimensionality reduction. Presented as the 8th lecture of the Machine Learning I course, University of Twente.

Everitt, B. and Skrondal, A. (2010). *The Cambridge Dictionary of Statistics*. Cambridge University Press.

Fanti, M. P., Iacobellis, G., Nolich, M., Rusich, A., and Ukovich, W. (2017). A decision support system for cooperative logistics. *IEEE Transactions on Automation Science and Engineering*, PP:1–13.

Fazel, A. and Chakrabartty, S. (2011). An overview of statistical pattern recognition techniques for speaker verification. *Circuits and Systems Magazine, IEEE*, 11:62 – 81.

Flick, U. (2009). *An Introduction to Qualitative Research*. SAGE Publications Ltd, UK, 6 edition.

Gaonkar, B. and Davatzikos, C. (2013). Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification. *Neurolmage*, 78:270–283.

Garbade, M. J. (2018). Clearing the confusion: Ai vs machine learning vs deep learning differences.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58.

Ghiani, G., Laporte, G., and Musmanno, R. (2004). *Introduction to Logistics Systems Planning and Control*. Halsted Press, USA.

Gorry, G. and Scott-Morton, M. (1971). A framework for management information systems,. *Sloan Management Review*, 13:50–70.

Graves, A., Mohamed, A.-R., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. pages 6645–6649. IEEE.

Irannezhad, E., Prato, C., and Hickman, M. (2020). An intelligent decision support system prototype for hinterland port logistics. *Decision Support Systems*, 130.

Isaac, E. (2015). Is there a rule of thumb that explains the splitting of a limited dataset into two-three subsets?

Jain, A., Murty, M., and Flynn, P. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer Texts in Statistics. Springer New York, 6 edition.

Jansen, B., Swinkels, P. C., Teeuwen, G. J., van Antwerpen de Fluiter, B., and Fleuren, H. A. (2004). Operational planning of a large-scale multi-modal transportation system. *European Journal of Operational Research*, 156(1):41–53. EURO Excellence in Practice Award 2001.

Koot, M., Mes, M. R., and Iacob, M. E. (2021). A systematic literature review of supply chain decision making supported by the internet of things and big data analytics. *Computers and Industrial Engineering*, 154:107076.

Kotsiantis, S., Zaharakis, I., and Pintelas, P. (2007). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26:159–190.

Kourou, K., Exarchos, T., Exarchos, K., Karamouzis, M., and Fotiadis, D. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal*, 13:8–17.

Kuter, U. (2012). *Learning and planning (intelligent systems)*, volume 9781461418009. Elsevier.

Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(753):436–444.

Lewis, F. and Vrabie, D. (2009). Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50.

Li, J.-Q., Borenstein, D., and Mirchandani, P. (2007). A decision support system for the single-depot vehicle rescheduling problem. *Computers and Operations Research*, 34(4):1008–1032.

Manyika, J., Chui, M., Institute, M. G., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., and Byers, A. (2011). Big data: The next frontier for innovation, competition, and productivity.

Manzini, R. and Bindi, F. (2009). Strategic design and operational management optimization of a multi stage physical distribution system. *Transportation Research Part E: Logistics and Transportation Review*, 45(6):915–936.

McCorduck, P. (2004). *Machines Who Think: A Personal Inquiry Into the History and Prospects of Artificial Intelligence*. Ak Peters Series. Taylor and Francis.

McCue, C. (2015). Chapter 5 - data. In *Data Mining and Predictive Analysis*, pages 75–106. Butterworth-Heinemann, Boston, 2nd edition.

Mendix Technology, B. (2021). Studio pro 9 guide: Application logic: Microflows.

Mes, M. (2020). Operations research techniques 2: Stochastic optimization and learning. University of Twente.

Mes, M. and van Heeswijk, W. (2020). Comparison of manual and automated decision-making with a logistics serious game. *International Conference on Computational Logistics*, 12433 LNCS:698–714.

Mocanu, E. (2020). Machine learning 2 lecture 2: Reinforcement learning. Presented as the 2nd lecture of the Machine Learning II course, University of Twente.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press.

Nagpal, A. (2017). L1 and l2 regularization methods.

Pavlidis, T. (1977). *Structural Pattern Recognition*. Electrophysics Series. Springer.

Peffers, K., Tuunanen, T., Rothenbergern, M., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77.

Power, D. (2008). Understanding data-driven decision support systems. *Information Systems Management*, 25(2):149–154.

Rabe, M. and Dross, F. (2016). A reinforcement learning approach for a decision support system for logistics networks. volume 2016-February, pages 2020–2032.

Ragupathi, T. and Govindarajan, M. (2020). Performance assessment of different machine learning algorithms for medical decision support systems. *Lecture Notes on Data Engineering and Communications Technologies*, 49:941–947.

Raileanu, L. E. and Stoffel, K. (2004). Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41:77–93.

Ranjan, C. (2019). Rules-of-thumb for building a neural network.

Ridder, H.-G. (2017). The theory contribution of case study research designs. *Business Research*, 10:281–305.

Rokach, L. and Maimon, O. (2008). *Data Mining with Decision Trees: Theory and Applications*. Series in machine perception and artificial intelligence. World Scientific.

Roman, V. (2019). Unsupervised machine learning: Cluster analysis.

Ruiz, R., Maroto, C., and Alcaraz, J. (2004). A decision support system for a real vehicle routing problem. *European Journal of Operational Research*, 153(3):593–606. EURO Young Scientists.

Rumelhart, D., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Russell, S., Norvig, P., and Davis, E. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall series in artificial intelligence. Prentice Hall.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L., and Müller, K.-R. (2019). *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. ResearchGate.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

Shim, J., Warkentin, M., Courtney, J., Power, D., Sharda, R., and Carlsson, C. (2002). Past, present, and future of decision support technology. *Decision Support Systems*, 33(2):111–126.

Shrajluhaniwal, V. (2020). A comprehensive guide to feature selection using wrapper methods in python.

Strisciuglio, N. (2020). Machine learning 2 lecture 3: Convolutional neural networks. Presented as the 3rd lecture of the Machine Learning II course, University of Twente.

Sutton, R. and Barto, A. (2018). *Reinforcement Learning, second edition: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press.

Venkatesh, V. and Bala, H. (2008). Technology acceptance model 3 and a research agenda on interventions. *Decision Sciences - DECISION SCIENCES*, 39:273–315.

Wang, L. and Alexander, C. (2015). Big data driven supply chain management and business administration. *American Journal of Economics and Business Administration*, 7:60–67.

Wilson, P., Dell, L., and Anderson, G. (1993). *Root Cause Analysis: A Tool for Total Quality Management*. ASQC Quality Press.

Xhemali, D., Hinde, C. J., and Stone, R. (2009). Naive bayes vs. decision trees vs. neural networks in the classification of training web pages. *International Journal of Computer Science Issues*, 4.

Zak, J. (2010). *Decision Support Systems in Transportation*. Springer Berlin Heidelberg, Berlin, Heidelberg.

# Appendix A

# Context Analysis

## A.1 UAR: Javascript file

```
// mxui.widget._WidgetBase.update is called when context is changed or initialized. Implement to re-render and / or fetch data.
update: function (obj, callback) {
    //logger.debug(this.id + ".update");

    var arr = ["OperationalPlanning.Act_Ride_PlanToOriginal",
            "OperationalPlanning.MoveRide_ShowOverview",
            "OperationalPlanning.MoveItem",
            "OperationalPlanning.MoveRide"]

    this._contextObj = obj;
    // document.getElementsByClassName("btn mx-button mx-name-actionButton6 btn-default").onclick(passStringToMicroflow("btn mx-button mx-name-actionButton6 btn-default"));
    // document.getElementsByClassName("btn mx-button mx-name-actionButton8 btn-default").onclick(passStringToMicroflow("btn mx-button mx-name-actionButton8 btn-default"));
    var list_of_fields_jdata = ['name','RideName','RideIsMoved', 'ScenarioDayNumber', 'Replanned', 'ShiftID','RideNumber']
    var list_of_keys_objdata = ['ConsigneeParty','ConsignorParty','Party']
    var list_of_fields_objdata = ['BuildingNumber','PostalCode','CityName','CountryCode']
    var temp;
    var i;
    var n;
    var m;
    var temp2;
    var temp3;
    var objectsRespParsed;
    var XmlNode = new DOMParser();
    var jobj = {};
    var oldFetch = fetch;
    fetch = (url, options) => {
        if (options.method != "post") { return oldFetch(url, options) } else {
            return oldFetch(url, options)
                .then(response => {
                    response.clone().text().then(data => {
                        if (options.body != undefined) {
                            var parsed = JSON.parse(options.body);
                            var respParsed = JSON.parse(data);
                            // var busname = this._contextObj.jsonData.attributes.BusDisplayName.value
                            console.error("CHECK START");
                            console.error(respParsed);
                            console.error(parsed);
```

Figure A.1: Part of the Javascript file used for the User Action Recording.

# Appendix B

# Solution Design

## B.1 Detailed description of raw data

Table B.1: Table with all variables and descriptions of the input dataset for the learning models.

| Variable | Data type | Description |
|---|---|---|
| timestamp | string | containing the date and time when the replanning was occurred in the system |
| ButtonName | string | name of the button that triggered the recorded Microflow |
| RideName | string | the Ride which is replanned and the name consists of integers and string |
| Replanned | boolean | indicating whether the replanning is finalized in the system |
| NewRide | integer | the number associated with the new Ride |
| ScenarioDayNumber | integer | number indicating the day of the week |
| ShiftID | integer | number indicating the Shift of the replanned Ride |
| RideNumber | integer | contains the number associated to the Ride |
| Subcontractor | string | name of the company responsible for delivering the Ride |
| ConsigneeParty_PostalCode | string | the postal code (example format: 1234 AB) of the recipient |

Continued on next page

Table B.1 – continued from previous page

| Variable | Data type | Description |
|---|---|---|
| ConsigneeParty_BuildingNumber | integer | the building or house number of the recipient |
| ConsigneeParty_CityName | string | the city name of the recipient |
| ConsigneeParty_CountryCode | string | the country of the recipient |
| ConsignorParty_PostalCode | string | the postal code (example format: 1234 AB) of the company shipping the parcel |
| ConsignorParty_BuildingNumber | integer | the building or house number of the company shipping the parcel |
| ConsignorParty_CityName | string | the city name of the company shipping the parcel |
| ConsignorParty_CountryCode | string | the country name of the company shipping the parcel |
| Party_PostalCode | string | the postal code (example format: 1234 AB) of the sender |
| Party_BuildingNumber | integer | the building or house number of the sender |
| Party_CityName | string | the city name of the sender |
| Party_CountryCode | string | the country name of the sender |
| Stops | integer | total number of stops associated to the Ride |
| Parcels | integer | total number of parcels associated to the Ride |
| Volume | float | the total volume of all parcels on the Ride |
| Weight | float | the total weight of all parcels on the Ride |

# Appendix C

# Experimental Results

## C.1 Hyper-parameter tuning

### C.1.1 Neural Network

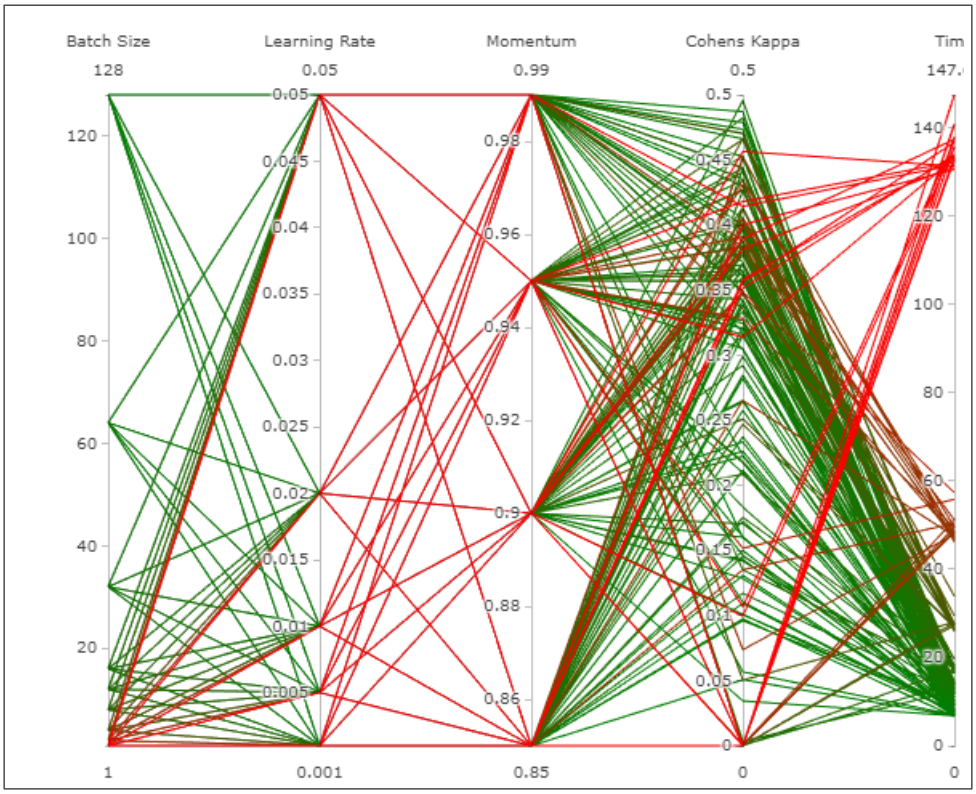



Figure C.1: Parallel Coordinates Plot of the SGD hyper-parameter tuning experiment.

.

| Model | Batch Size | Learning Rate | Momentum | Accuracy | Precision | Recall | F1 Measure | Cohens Kappa | Computation Time | Model | Batch Size | Learning Rate | Momentum | Accuracy | Precision | Recall | F1 Measure | Cohens Kappa | Computation Time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24 | 16 | 0,05 | 0,9 | 50,64% | 53,19% | 34,50% | 48,69% | 49,57% | 11,98 | 122 | 12 | 0,005 | 0,95 | 35,74% | 35,70% | 23,84% | 31,67% | 34,25% | 13,28 |
| 9 | 128 | 0,05 | 0,99 | 49,79% | 53,30% | 35,33% | 46,80% | 48,70% | 6,97 | 35 | 64 | 0,05 | 0,85 | 35,74% | 30,41% | 23,98% | 29,38% | 34,22% | 7,88 |
| 15 | 16 | 0,05 | 0,95 | 49,36% | 46,92% | 35,68% | 45,87% | 48,23% | 12,69 | 105 | 16 | 0,01 | 0,85 | 35,32% | 32,77% | 23,03% | 30,05% | 33,91% | 11,97 |
| 43 | 32 | 0,02 | 0,99 | 48,94% | 48,69% | 37,02% | 46,10% | 47,88% | 10,10 | 34 | 32 | 0,05 | 0,85 | 35,32% | 39,31% | 25,08% | 31,09% | 33,86% | 9,60 |
| 80 | 64 | 0,01 | 0,99 | 48,51% | 45,25% | 32,81% | 44,95% | 47,35% | 8,13 | 111 | 4 | 0,005 | 0,99 | 34,89% | 36,51% | 22,91% | 32,89% | 33,49% | 27,79 |
| 147 | 4 | 0,001 | 0,99 | 48,09% | 51,62% | 34,35% | 46,38% | 47,04% | 26,22 | 27 | 128 | 0,05 | 0,9 | 34,89% | 30,82% | 23,69% | 29,09% | 33,33% | 6,95 |
| 78 | 16 | 0,01 | 0,99 | 48,09% | 53,58% | 34,48% | 46,67% | 47,01% | 12,42 | 83 | 2 | 0,01 | 0,95 | 34,04% | 38,83% | 22,40% | 32,35% | 32,83% | 49,45 |
| 79 | 32 | 0,01 | 0,99 | 47,66% | 51,50% | 36,29% | 45,55% | 46,62% | 10,87 | 156 | 4 | 0,001 | 0,95 | 34,04% | 29,09% | 23,81% | 28,40% | 32,67% | 26,74 |
| 57 | 4 | 0,02 | 0,9 | 47,66% | 50,90% | 34,57% | 44,47% | 46,61% | 27,52 | 88 | 32 | 0,01 | 0,95 | 34,04% | 28,60% | 22,86% | 28,94% | 32,66% | 9,06 |
| 120 | 4 | 0,005 | 0,95 | 47,66% | 53,62% | 35,99% | 44,62% | 46,57% | 27,28 | 54 | 128 | 0,02 | 0,95 | 34,04% | 31,14% | 23,26% | 29,17% | 32,45% | 7,20 |
| 127 | 1 | 0,005 | 0,9 | 46,81% | 45,62% | 32,26% | 42,85% | 45,64% | 131,42 | 139 | 8 | 0,005 | 0,85 | 33,62% | 32,31% | 23,90% | 29,52% | 32,16% | 16,97 |
| 51 | 16 | 0,02 | 0,95 | 46,38% | 49,57% | 28,87% | 44,77% | 45,29% | 11,81 | 89 | 64 | 0,01 | 0,95 | 33,62% | 31,19% | 22,16% | 28,68% | 32,06% | 7,65 |
| 45 | 128 | 0,02 | 0,99 | 46,38% | 45,39% | 31,61% | 42,64% | 45,25% | 7,08 | 117 | 128 | 0,005 | 0,99 | 33,19% | 31,84% | 21,68% | 29,21% | 31,72% | 6,92 |
| 137 | 2 | 0,005 | 0,85 | 46,38% | 45,86% | 31,64% | 43,65% | 45,23% | 46,20 | 157 | 8 | 0,001 | 0,95 | 33,19% | 30,05% | 20,00% | 28,01% | 31,59% | 16,85 |
| 67 | 8 | 0,02 | 0,85 | 46,38% | 44,34% | 30,48% | 42,48% | 45,14% | 16,35 | 70 | 32 | 0,02 | 0,85 | 33,19% | 29,18% | 21,63% | 25,19% | 31,49% | 10,12 |
| 113 | 12 | 0,005 | 0,99 | 45,53% | 49,50% | 31,78% | 43,62% | 44,40% | 13,95 | 131 | 12 | 0,005 | 0,9 | 33,19% | 30,25% | 19,25% | 27,31% | 31,49% | 13,92 |
| 112 | 8 | 0,005 | 0,99 | 45,53% | 47,92% | 32,81% | 42,10% | 44,38% | 18,00 | 118 | 1 | 0,005 | 0,95 | 32,77% | 33,84% | 23,89% | 29,98% | 31,39% | 133,32 |
| 77 | 12 | 0,01 | 0,99 | 45,53% | 45,39% | 29,97% | 42,74% | 44,36% | 14,06 | 124 | 32 | 0,005 | 0,95 | 31,91% | 30,41% | 21,68% | 26,82% | 30,51% | 9,25 |
| 50 | 12 | 0,02 | 0,95 | 45,53% | 48,43% | 31,69% | 44,39% | 44,36% | 13,73 | 151 | 32 | 0,001 | 0,99 | 31,91% | 29,31% | 18,88% | 26,75% | 30,40% | 9,19 |
| 33 | 16 | 0,05 | 0,85 | 45,53% | 45,54% | 31,89% | 42,62% | 44,32% | 12,59 | 140 | 12 | 0,005 | 0,85 | 31,49% | 31,90% | 20,83% | 27,64% | 29,91% | 13,57 |
| 49 | 8 | 0,02 | 0,95 | 45,11% | 50,59% | 34,13% | 43,65% | 44,05% | 16,69 | 132 | 16 | 0,005 | 0,9 | 30,64% | 34,57% | 21,75% | 26,91% | 29,21% | 12,94 |
| 85 | 8 | 0,01 | 0,95 | 45,11% | 47,17% | 33,23% | 42,34% | 44,04% | 17,37 | 141 | 16 | 0,005 | 0,85 | 30,21% | 21,26% | 19,69% | 22,33% | 28,47% | 11,67 |
| 44 | 64 | 0,02 | 0,99 | 44,68% | 47,97% | 29,57% | 42,25% | 43,53% | 7,76 | 36 | 128 | 0,05 | 0,85 | 29,79% | 27,35% | 16,96% | 24,55% | 28,21% | 7,13 |
| 128 | 2 | 0,005 | 0,9 | 44,68% | 44,54% | 33,03% | 41,56% | 43,46% | 47,88 | 62 | 64 | 0,02 | 0,9 | 28,09% | 23,40% | 19,80% | 23,71% | 26,53% | 8,44 |
| 18 | 128 | 0,05 | 0,95 | 44,68% | 43,28% | 31,56% | 39,69% | 43,43% | 6,79 | 173 | 2 | 0,001 | 0,85 | 28,09% | 24,41% | 17,51% | 21,91% | 26,52% | 47,51 |
| 92 | 2 | 0,01 | 0,9 | 44,26% | 43,66% | 30,76% | 40,40% | 43,07% | 49,65 | 97 | 32 | 0,01 | 0,9 | 28,09% | 26,94% | 15,49% | 23,23% | 26,42% | 10,29 |
| 81 | 128 | 0,01 | 0,99 | 44,26% | 40,83% | 28,83% | 39,78% | 43,00% | 7,47 | 71 | 64 | 0,02 | 0,85 | 27,66% | 22,85% | 15,31% | 20,82% | 25,76% | 8,57 |
| 52 | 32 | 0,02 | 0,95 | 43,83% | 47,14% | 28,94% | 41,60% | 42,69% | 10,11 | 165 | 4 | 0,001 | 0,9 | 26,81% | 22,02% | 16,44% | 21,28% | 25,11% | 27,08 |
| 31 | 8 | 0,05 | 0,85 | 43,83% | 44,91% | 28,51% | 40,29% | 42,64% | 17,93 | 174 | 4 | 0,001 | 0,85 | 26,38% | 25,19% | 17,15% | 23,01% | 24,69% | 26,81 |
| 84 | 4 | 0,01 | 0,95 | 43,83% | 46,20% | 30,28% | 42,32% | 42,62% | 27,16 | 72 | 128 | 0,02 | 0,85 | 25,96% | 19,40% | 13,77% | 18,44% | 23,74% | 7,74 |
| 154 | 1 | 0,001 | 0,95 | 42,98% | 41,35% | 31,50% | 39,01% | 41,80% | 132,41 | 63 | 128 | 0,02 | 0,9 | 25,11% | 18,92% | 15,85% | 20,30% | 23,34% | 8,02 |
| 145 | 1 | 0,001 | 0,99 | 42,55% | 47,75% | 29,74% | 39,96% | 41,43% | 131,70 | 98 | 64 | 0,01 | 0,9 | 24,68% | 15,69% | 14,36% | 17,21% | 22,63% | 8,48 |
| 30 | 4 | 0,05 | 0,85 | 42,55% | 39,73% | 30,68% | 37,97% | 41,33% | 27,67 | 152 | 64 | 0,001 | 0,99 | 24,26% | 17,37% | 13,65% | 19,08% | 22,31% | 7,80 |
| 96 | 16 | 0,01 | 0,9 | 42,55% | 42,83% | 27,55% | 39,91% | 41,33% | 13,03 | 133 | 32 | 0,005 | 0,9 | 24,26% | 19,82% | 13,23% | 18,39% | 22,22% | 10,05 |
| 138 | 4 | 0,005 | 0,85 | 42,55% | 35,59% | 28,33% | 36,38% | 41,21% | 26,13 | 158 | 12 | 0,001 | 0,95 | 23,83% | 22,30% | 13,48% | 18,93% | 21,82% | 13,77 |
| 149 | 12 | 0,001 | 0,99 | 41,70% | 47,69% | 31,67% | 38,09% | 40,57% | 13,85 | 90 | 128 | 0,01 | 0,95 | 23,83% | 18,11% | 13,21% | 17,68% | 21,73% | 6,95 |
| 101 | 2 | 0,01 | 0,85 | 41,70% | 44,54% | 29,63% | 38,94% | 40,52% | 48,47 | 142 | 32 | 0,005 | 0,85 | 23,40% | 17,19% | 13,26% | 16,66% | 21,17% | 8,91 |
| 21 | 4 | 0,05 | 0,9 | 41,70% | 46,42% | 27,65% | 39,34% | 40,49% | 29,40 | 106 | 32 | 0,01 | 0,85 | 22,98% | 19,20% | 12,87% | 17,38% | 21,13% | 9,29 |
| 48 | 4 | 0,02 | 0,95 | 41,70% | 41,70% | 30,29% | 37,73% | 40,46% | 33,92 | 125 | 64 | 0,005 | 0,95 | 22,98% | 17,95% | 13,23% | 17,57% | 20,95% | 7,69 |
| 25 | 32 | 0,05 | 0,9 | 41,70% | 43,68% | 29,30% | 36,77% | 40,42% | 9,10 | 166 | 8 | 0,001 | 0,9 | 22,98% | 14,37% | 12,12% | 15,44% | 20,85% | 17,47 |
| 68 | 12 | 0,02 | 0,85 | 41,28% | 41,75% | 27,45% | 38,59% | 40,09% | 13,13 | 159 | 16 | 0,001 | 0,95 | 20,43% | 15,02% | 12,47% | 14,28% | 18,33% | 11,81 |
| 16 | 32 | 0,05 | 0,95 | 41,28% | 41,47% | 28,99% | 36,84% | 40,08% | 9,70 | 175 | 8 | 0,001 | 0,85 | 20,00% | 11,02% | 11,19% | 12,27% | 17,53% | 17,38 |
| 86 | 12 | 0,01 | 0,95 | 41,28% | 40,43% | 30,49% | 37,88% | 40,05% | 13,83 | 107 | 64 | 0,01 | 0,85 | 19,57% | 12,27% | 11,41% | 12,69% | 17,26% | 7,68 |
| 65 | 2 | 0,02 | 0,85 | 41,28% | 42,76% | 28,44% | 38,54% | 40,03% | 47,40 | 167 | 12 | 0,001 | 0,9 | 19,57% | 9,53% | 10,63% | 11,42% | 17,05% | 13,81 |
| 136 | 1 | 0,005 | 0,85 | 41,28% | 40,99% | 27,90% | 37,52% | 40,03% | 130,70 | 126 | 128 | 0,005 | 0,95 | 18,72% | 7,37% | 9,01% | 9,85% | 16,06% | 6,90 |
| 116 | 64 | 0,005 | 0,99 | 41,28% | 37,04% | 28,36% | 35,52% | 40,02% | 7,60 | 134 | 64 | 0,005 | 0,9 | 18,72% | 7,04% | 8,36% | 9,73% | 16,01% | 7,56 |
| 23 | 12 | 0,05 | 0,9 | 40,85% | 42,58% | 32,75% | 37,92% | 39,72% | 14,01 | 153 | 128 | 0,001 | 0,99 | 18,72% | 9,96% | 8,82% | 10,01% | 15,99% | 6,93 |
| 115 | 32 | 0,005 | 0,99 | 40,85% | 38,80% | 28,38% | 36,21% | 39,63% | 9,05 | 47 | 2 | 0,02 | 0,95 | 17,87% | 8,44% | 7,89% | 10,13% | 15,17% | 55,86 |
| 114 | 16 | 0,005 | 0,99 | 40,85% | 39,43% | 27,56% | 38,14% | 39,59% | 12,01 | 143 | 64 | 0,005 | 0,85 | 17,87% | 6,40% | 7,58% | 8,50% | 14,99% | 7,58 |
| 59 | 12 | 0,02 | 0,9 | 40,85% | 37,70% | 28,41% | 35,97% | 39,56% | 13,75 | 160 | 32 | 0,001 | 0,95 | 17,45% | 7,64% | 7,20% | 9,38% | 14,63% | 9,08 |
| 103 | 8 | 0,01 | 0,85 | 40,43% | 42,71% | 29,39% | 35,00% | 39,21% | 17,12 | 176 | 12 | 0,001 | 0,85 | 17,45% | 6,50% | 6,95% | 8,81% | 14,56% | 13,73 |
| 119 | 2 | 0,005 | 0,95 | 40,43% | 40,33% | 25,38% | 36,32% | 39,12% | 51,20 | 99 | 128 | 0,01 | 0,9 | 17,02% | 7,08% | 8,41% | 9,47% | 14,31% | 7,40 |
| 91 | 1 | 0,01 | 0,9 | 40,00% | 48,02% | 29,99% | 37,76% | 38,94% | 137,53 | 168 | 16 | 0,001 | 0,9 | 17,02% | 5,94% | 6,64% | 8,17% | 14,19% | 12,09 |
| 58 | 8 | 0,02 | 0,9 | 40,00% | 44,80% | 27,99% | 37,53% | 38,80% | 17,45 | 29 | 2 | 0,05 | 0,85 | 15,32% | 12,06% | 8,29% | 11,36% | 13,52% | 50,40 |
| 146 | 2 | 0,001 | 0,99 | 40,00% | 36,25% | 28,59% | 35,46% | 38,74% | 46,52 | 40 | 8 | 0,02 | 0,99 | 16,60% | 6,19% | 7,52% | 8,17% | 13,51% | 16,38 |
| 60 | 16 | 0,02 | 0,9 | 40,00% | 38,61% | 26,84% | 35,34% | 38,67% | 13,02 | 108 | 128 | 0,01 | 0,85 | 16,17% | 4,74% | 6,95% | 7,00% | 13,08% | 6,85 |
| 66 | 4 | 0,02 | 0,85 | 39,57% | 38,35% | 29,71% | 36,28% | 38,42% | 26,51 | 135 | 128 | 0,005 | 0,9 | 16,17% | 4,38% | 5,53% | 6,48% | 12,85% | 6,71 |
| 14 | 12 | 0,05 | 0,95 | 39,57% | 39,40% | 25,83% | 36,11% | 38,35% | 14,36 | 177 | 16 | 0,001 | 0,85 | 14,89% | 4,54% | 5,39% | 6,21% | 11,71% | 12,01 |
| 93 | 4 | 0,01 | 0,9 | 39,57% | 41,52% | 26,48% | 37,22% | 38,34% | 25,93 | 161 | 64 | 0,001 | 0,95 | 14,47% | 4,37% | 5,01% | 6,26% | 11,03% | 7,58 |
| 102 | 4 | 0,01 | 0,85 | 39,57% | 41,07% | 27,60% | 35,54% | 38,33% | 27,13 | 179 | 64 | 0,001 | 0,85 | 14,47% | 2,87% | 4,94% | 4,55% | 10,92% | 7,58 |
| 76 | 8 | 0,01 | 0,99 | 39,57% | 45,80% | 25,83% | 37,49% | 38,28% | 18,12 | 82 | 1 | 0,01 | 0,95 | 12,77% | 4,90% | 6,03% | 6,54% | 10,67% | 137,06 |
| 155 | 2 | 0,001 | 0,95 | 39,57% | 38,20% | 27,57% | 34,04% | 38,28% | 47,01 | 170 | 64 | 0,001 | 0,9 | 13,19% | 5,68% | 4,88% | 6,89% | 10,08% | 7,62 |
| 163 | 1 | 0,001 | 0,9 | 39,57% | 37,04% | 25,82% | 33,74% | 38,13% | 131,39 | 169 | 32 | 0,001 | 0,9 | 13,62% | 3,56% | 4,44% | 5,21% | 10,06% | 9,19 |
| 8 | 64 | 0,05 | 0,99 | 39,15% | 43,47% | 30,08% | 36,68% | 38,03% | 7,93 | 55 | 1 | 0,02 | 0,9 | 12,77% | 8,13% | 6,90% | 8,14% | 9,97% | 135,80 |
| 129 | 4 | 0,005 | 0,9 | 39,15% | 41,48% | 26,96% | 37,41% | 37,92% | 27,25 | 144 | 128 | 0,005 | 0,85 | 13,19% | 2,66% | 4,37% | 4,17% | 9,74% | 6,58 |
| 95 | 12 | 0,01 | 0,9 | 39,15% | 38,81% | 26,94% | 34,97% | 37,87% | 14,76 | 178 | 32 | 0,001 | 0,85 | 13,19% | 2,74% | 4,75% | 4,27% | 9,61% | 9,06 |
| 130 | 8 | 0,005 | 0,9 | 39,15% | 33,15% | 26,73% | 32,52% | 37,68% | 17,59 | 110 | 2 | 0,005 | 0,99 | 11,06% | 1,41% | 3,80% | 2,47% | 7,38% | 47,95 |
| 148 | 8 | 0,001 | 0,99 | 38,30% | 43,20% | 29,64% | 33,51% | 37,14% | 19,56 | 162 | 128 | 0,001 | 0,95 | 9,36% | 2,39% | 3,02% | 2,91% | 5,68% | 6,98 |
| 32 | 12 | 0,05 | 0,85 | 38,30% | 45,62% | 26,23% | 38,15% | 37,06% | 14,26 | 180 | 128 | 0,001 | 0,85 | 8,94% | 1,29% | 3,16% | 2,16% | 5,07% | 6,75 |
| 22 | 8 | 0,05 | 0,9 | 38,30% | 38,53% | 27,29% | 35,47% | 37,03% | 17,62 | 75 | 4 | 0,01 | 0,99 | 7,23% | 1,71% | 3,75% | 2,62% | 5,04% | 28,01 |
| 94 | 8 | 0,01 | 0,9 | 38,30% | 41,41% | 29,80% | 34,60% | 37,02% | 18,25 | 171 | 128 | 0,001 | 0,9 | 6,81% | 0,63% | 2,43% | 1,14% | 3,44% | 6,81 |
| 53 | 64 | 0,02 | 0,95 | 38,30% | 42,70% | 25,60% | 34,87% | 36,92% | 8,20 | 6 | 16 | 0,05 | 0,99 | 5,11% | 0,28% | 1,27% | 0,54% | 0,34% | 12,39 |
| 17 | 64 | 0,05 | 0,95 | 37,87% | 38,71% | 25,50% | 34,13% | 36,50% | 8,02 | 20 | 2 | 0,05 | 0,9 | 2,13% | 0,05% | 1,27% | 0,10% | 0,09% | 51,18 |
| 121 | 8 | 0,005 | 0,95 | 37,87% | 38,67% | 26,61% | 34,51% | 36,49% | 17,31 | 5 | 12 | 0,05 | 0,99 | 2,13% | 0,05% | 1,25% | 0,09% | 0,05% | 14,07 |
| 61 | 32 | 0,02 | 0,9 | 37,87% | 39,51% | 25,68% | 32,90% | 36,48% | 10,11 | 1 | 1 | 0,05 | 0,99 | 0,43% | 0,18% | 1,27% | 0,36% | 0,00% | 135,82 |
| 87 | 16 | 0,01 | 0,95 | 37,45% | 40,57% | 27,63% | 33,52% | 36,20% | 12,05 | 2 | 2 | 0,05 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 48,82 |
| 7 | 32 | 0,05 | 0,99 | 37,45% | 37,00% | 23,86% | 33,20% | 36,09% | 9,38 | 3 | 4 | 0,05 | 0,99 | 5,11% | 0,26% | 1,27% | 0,50% | 0,00% | 27,37 |
| 26 | 64 | 0,05 | 0,9 | 37,45% | 35,36% | 23,29% | 33,67% | 36,01% | 7,61 | 4 | 8 | 0,05 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 17,43 |
| 100 | 1 | 0,01 | 0,85 | 37,02% | 37,00% | 24,97% | 34,19% | 35,81% | 133,61 | 10 | 1 | 0,05 | 0,95 | 4,68% | 0,22% | 1,27% | 0,42% | 0,00% | 140,94 |
| 123 | 16 | 0,005 | 0,95 | 37,02% | 35,16% | 26,69% | 32,09% | 35,71% | 11,77 | 11 | 2 | 0,05 | 0,95 | 4,68% | 0,22% | 1,27% | 0,42% | 0,00% | 48,44 |
| 13 | 8 | 0,05 | 0,95 | 37,02% | 38,86% | 23,95% | 34,46% | 35,67% | 17,95 | 12 | 4 | 0,05 | 0,95 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 27,96 |
| 69 | 16 | 0,02 | 0,85 | 37,02% | 32,98% | 25,19% | 32,28% | 35,62% | 12,61 | 19 | 1 | 0,05 | 0,9 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 133,54 |
| 64 | 1 | 0,02 | 0,85 | 37,02% | 35,47% | 24,97% | 31,96% | 35,61% | 136,85 | 28 | 1 | 0,05 | 0,85 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 137,78 |
| 172 | 1 | 0,001 | 0,85 | 36,60% | 31,16% | 25,62% | 30,14% | 35,14% | 133,99 | 37 | 1 | 0,02 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 138,06 |
| 41 | 12 | 0,02 | 0,99 | 36,17% | 45,09% | 24,87% | 34,09% | 34,92% | 13,63 | 38 | 2 | 0,02 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 49,56 |
| 42 | 16 | 0,02 | 0,99 | 36,17% | 39,08% | 24,46% | 34,62% | 34,89% | 12,73 | 39 | 4 | 0,02 | 0,99 | 5,11% | 0,26% | 1,27% | 0,50% | 0,00% | 27,36 |
| 56 | 2 | 0,02 | 0,9 | 36,17% | 38,22% | 26,60% | 33,19% | 34,87% | 57,39 | 46 | 1 | 0,02 | 0,95 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 147,61 |
| 124 | 12 | 0,01 | 0,85 | 36,17% | 32,49% | 26,18% | 31,09% | 34,82% | 13,78 | 73 | 1 | 0,01 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 140,86 |
| 150 | 16 | 0,001 | 0,99 | 36,17% | 29,83% | 24,53% | 29,34% | 34,77% | 11,86 | 74 | 2 | 0,01 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 48,86 |
| 164 | 2 | 0,001 | 0,9 | 36,17% | 31,26% | 23,84% | 30,31% | 34,68% | 47,47 | 109 | 1 | 0,005 | 0,99 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 132,95 |

Figure C.2: Performance metric scores of all unique SGD parameter settings (based on 5-fold cross-validation).

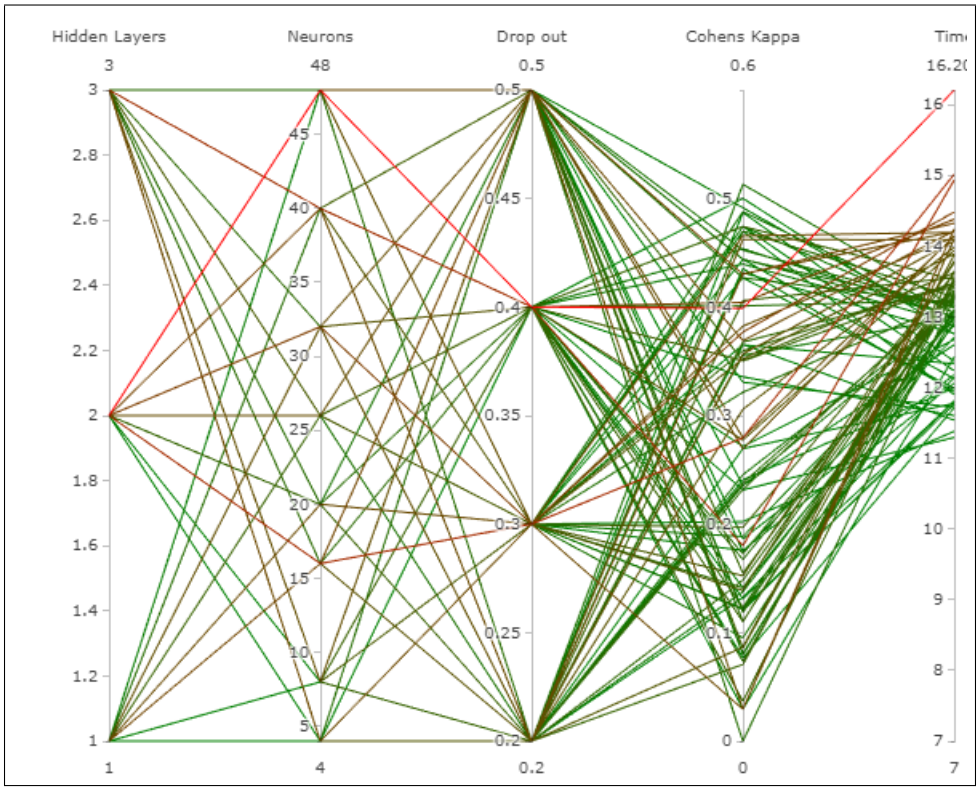



Figure C.3: Parallel Coordinates Plot of the Architecture hyper-parameter tuning experiment.

| Model | Layers | Neurons | Drop out | Accuracy | Precision | Recall | F1 Measure | Cohens Kappa | Computation Time |
|---|---|---|---|---|---|---|---|---|---|
| 17 | 1 | 26 | 0,2 | 52,34% | 52,63% | 36,64% | 50,09% | 51,30% | 13,15 |
| 31 | 1 | 48 | 0,4 | 51,06% | 52,73% | 34,47% | 49,00% | 50,01% | 12,75 |
| 32 | 1 | 48 | 0,5 | 50,21% | 50,75% | 37,88% | 47,66% | 49,14% | 12,85 |
| 22 | 1 | 32 | 0,3 | 49,79% | 56,67% | 35,10% | 48,39% | 48,78% | 13,05 |
| 29 | 1 | 48 | 0,2 | 49,79% | 51,31% | 34,39% | 47,12% | 48,70% | 12,37 |
| 27 | 1 | 40 | 0,4 | 48,51% | 51,71% | 32,57% | 47,34% | 47,43% | 12,24 |
| 18 | 1 | 26 | 0,3 | 48,51% | 51,84% | 35,51% | 46,39% | 47,43% | 12,98 |
| 19 | 1 | 26 | 0,4 | 48,51% | 48,75% | 34,75% | 45,38% | 47,42% | 13,52 |
| 25 | 1 | 40 | 0,2 | 48,09% | 49,31% | 34,60% | 46,11% | 46,99% | 13,28 |
| 10 | 1 | 16 | 0,3 | 48,09% | 46,50% | 34,13% | 43,72% | 46,99% | 11,90 |
| 14 | 1 | 20 | 0,3 | 47,66% | 51,29% | 35,40% | 45,04% | 46,56% | 14,19 |
| 21 | 1 | 32 | 0,2 | 47,23% | 53,16% | 33,52% | 46,18% | 46,22% | 14,09 |
| 23 | 1 | 32 | 0,4 | 46,38% | 51,65% | 34,62% | 45,05% | 45,37% | 13,00 |
| 20 | 1 | 26 | 0,5 | 46,38% | 50,70% | 34,59% | 44,89% | 45,24% | 13,13 |
| 16 | 1 | 20 | 0,5 | 45,53% | 45,86% | 34,44% | 42,87% | 44,41% | 13,01 |
| 30 | 1 | 48 | 0,3 | 45,53% | 49,33% | 33,57% | 44,02% | 44,39% | 12,12 |
| 24 | 1 | 32 | 0,5 | 45,53% | 49,90% | 30,66% | 43,69% | 44,38% | 13,14 |
| 15 | 1 | 20 | 0,4 | 45,11% | 44,39% | 30,43% | 41,99% | 43,95% | 12,91 |
| 9 | 1 | 16 | 0,2 | 44,68% | 48,96% | 31,15% | 43,41% | 43,53% | 11,94 |
| 61 | 2 | 48 | 0,2 | 44,26% | 45,71% | 29,52% | 41,23% | 43,16% | 12,76 |
| 57 | 2 | 40 | 0,2 | 44,26% | 41,98% | 28,67% | 40,42% | 43,06% | 14,20 |
| 12 | 1 | 16 | 0,5 | 44,26% | 40,79% | 32,32% | 39,43% | 43,02% | 14,32 |
| 26 | 1 | 40 | 0,3 | 43,83% | 51,45% | 30,12% | 41,39% | 42,70% | 11,99 |
| 28 | 1 | 40 | 0,5 | 43,83% | 48,16% | 30,73% | 41,76% | 42,69% | 13,18 |
| 13 | 1 | 20 | 0,2 | 41,70% | 48,50% | 29,25% | 40,57% | 40,59% | 13,56 |
| 59 | 2 | 40 | 0,4 | 41,70% | 37,95% | 29,33% | 35,70% | 40,42% | 14,38 |
| 11 | 1 | 16 | 0,4 | 41,28% | 39,12% | 32,44% | 36,23% | 40,10% | 13,23 |
| 63 | 2 | 48 | 0,4 | 41,28% | 33,51% | 24,99% | 34,46% | 39,89% | 16,21 |
| 62 | 2 | 48 | 0,3 | 39,57% | 39,00% | 25,51% | 36,46% | 38,24% | 14,13 |
| 54 | 2 | 32 | 0,3 | 39,15% | 34,35% | 25,44% | 32,70% | 37,69% | 14,48 |
| 64 | 2 | 48 | 0,5 | 38,30% | 33,27% | 23,27% | 31,95% | 36,90% | 14,22 |
| 45 | 2 | 20 | 0,2 | 38,30% | 29,89% | 22,80% | 29,51% | 36,77% | 13,43 |
| 6 | 1 | 8 | 0,3 | 37,87% | 35,57% | 25,82% | 34,21% | 36,54% | 11,54 |
| 5 | 1 | 8 | 0,2 | 37,87% | 32,51% | 24,01% | 31,91% | 36,51% | 12,34 |
| 46 | 2 | 20 | 0,3 | 37,45% | 29,68% | 24,37% | 28,87% | 35,95% | 13,48 |
| 53 | 2 | 32 | 0,2 | 37,02% | 28,95% | 24,07% | 30,27% | 35,68% | 14,07 |
| 49 | 2 | 26 | 0,2 | 37,02% | 37,25% | 23,54% | 33,69% | 35,55% | 12,99 |
| 58 | 2 | 40 | 0,3 | 36,60% | 33,93% | 24,50% | 32,13% | 35,22% | 13,89 |
| 51 | 2 | 26 | 0,4 | 36,60% | 26,63% | 23,68% | 28,13% | 35,06% | 13,62 |
| 60 | 2 | 40 | 0,5 | 36,60% | 25,33% | 21,59% | 27,18% | 35,02% | 13,47 |
| 7 | 1 | 8 | 0,4 | 34,89% | 31,76% | 22,68% | 30,10% | 33,56% | 11,60 |
| 8 | 1 | 8 | 0,5 | 34,47% | 29,83% | 24,48% | 29,91% | 33,07% | 11,72 |
| 50 | 2 | 26 | 0,3 | 33,62% | 26,22% | 24,42% | 25,87% | 32,15% | 13,21 |
| 41 | 2 | 16 | 0,2 | 32,34% | 24,78% | 19,61% | 26,33% | 30,57% | 13,98 |
| 56 | 2 | 32 | 0,5 | 29,79% | 21,59% | 19,93% | 21,66% | 27,96% | 14,14 |
| 42 | 2 | 16 | 0,3 | 29,79% | 17,06% | 17,85% | 19,57% | 27,95% | 15,02 |
| 55 | 2 | 32 | 0,4 | 29,36% | 20,74% | 17,32% | 21,43% | 27,74% | 13,94 |
| 47 | 2 | 20 | 0,4 | 28,94% | 18,20% | 15,52% | 19,14% | 27,15% | 12,68 |
| 1 | 1 | 4 | 0,2 | 28,51% | 17,99% | 18,71% | 20,17% | 26,96% | 11,78 |
| 52 | 2 | 26 | 0,5 | 28,51% | 13,19% | 21,32% | 16,97% | 26,91% | 14,15 |
| 77 | 3 | 20 | 0,2 | 26,38% | 14,99% | 15,31% | 16,51% | 24,55% | 13,09 |
| 37 | 2 | 8 | 0,2 | 25,96% | 17,07% | 14,29% | 16,86% | 24,08% | 12,00 |
| 93 | 3 | 48 | 0,2 | 25,96% | 13,14% | 13,49% | 16,11% | 23,79% | 13,30 |
| 81 | 3 | 26 | 0,2 | 25,53% | 14,82% | 15,04% | 17,10% | 23,57% | 13,08 |
| 4 | 1 | 4 | 0,5 | 25,11% | 18,50% | 18,38% | 18,09% | 23,56% | 12,61 |
| 85 | 3 | 32 | 0,2 | 25,11% | 15,85% | 13,63% | 15,33% | 23,20% | 13,36 |
| 3 | 1 | 4 | 0,4 | 24,68% | 21,16% | 14,00% | 19,68% | 23,15% | 11,29 |
| 2 | 1 | 4 | 0,3 | 22,13% | 10,97% | 13,22% | 13,98% | 20,23% | 11,38 |
| 90 | 3 | 40 | 0,3 | 22,13% | 10,30% | 10,28% | 10,64% | 19,70% | 13,61 |
| 94 | 3 | 48 | 0,3 | 20,85% | 10,10% | 10,26% | 12,42% | 18,85% | 12,97 |
| 40 | 2 | 8 | 0,5 | 21,28% | 6,46% | 9,25% | 9,55% | 18,64% | 12,18 |
| 91 | 3 | 40 | 0,4 | 20,43% | 6,24% | 9,75% | 9,19% | 18,00% | 14,93 |
| 38 | 2 | 8 | 0,3 | 20,00% | 6,27% | 9,44% | 8,94% | 17,57% | 12,06 |
| 48 | 2 | 20 | 0,5 | 19,57% | 9,22% | 11,75% | 10,17% | 17,47% | 12,71 |
| 87 | 3 | 32 | 0,4 | 19,15% | 6,95% | 11,33% | 9,22% | 17,25% | 13,27 |
| 43 | 2 | 16 | 0,4 | 19,15% | 6,78% | 9,97% | 9,57% | 16,73% | 13,28 |
| 44 | 2 | 16 | 0,5 | 18,72% | 6,39% | 7,36% | 8,98% | 15,47% | 13,47 |
| 82 | 3 | 26 | 0,3 | 17,45% | 8,48% | 9,31% | 9,88% | 15,24% | 13,76 |
| 89 | 3 | 40 | 0,2 | 16,60% | 10,91% | 12,08% | 11,14% | 14,83% | 13,42 |
| 95 | 3 | 48 | 0,4 | 16,60% | 5,57% | 8,69% | 7,19% | 14,13% | 12,89 |
| 86 | 3 | 32 | 0,3 | 16,17% | 5,24% | 10,46% | 7,64% | 14,13% | 13,38 |
| 70 | 3 | 8 | 0,3 | 17,02% | 3,33% | 6,08% | 5,51% | 13,89% | 13,79 |
| 73 | 3 | 16 | 0,2 | 16,60% | 4,88% | 7,00% | 7,03% | 13,77% | 13,01 |
| 33 | 2 | 4 | 0,2 | 16,17% | 4,09% | 5,83% | 6,10% | 13,24% | 12,14 |
| 39 | 2 | 8 | 0,4 | 15,74% | 4,20% | 7,17% | 6,38% | 13,13% | 11,80 |
| 79 | 3 | 20 | 0,4 | 14,47% | 2,57% | 7,59% | 4,25% | 12,33% | 13,31 |
| 34 | 2 | 4 | 0,3 | 15,32% | 2,98% | 5,06% | 4,93% | 12,20% | 11,77 |
| 74 | 3 | 16 | 0,3 | 14,89% | 3,00% | 6,33% | 4,90% | 12,13% | 13,06 |
| 36 | 2 | 4 | 0,5 | 15,32% | 2,64% | 5,06% | 4,46% | 12,00% | 11,81 |
| 88 | 3 | 32 | 0,5 | 15,32% | 2,51% | 5,06% | 4,28% | 11,86% | 13,29 |
| 96 | 3 | 48 | 0,5 | 14,47% | 2,49% | 4,73% | 4,24% | 11,03% | 13,33 |
| 83 | 3 | 26 | 0,4 | 13,62% | 4,21% | 6,08% | 5,98% | 11,00% | 13,50 |
| 78 | 3 | 20 | 0,3 | 13,19% | 1,99% | 5,06% | 3,41% | 9,52% | 13,10 |
| 65 | 3 | 4 | 0,2 | 11,49% | 2,18% | 4,30% | 3,56% | 8,66% | 13,88 |
| 92 | 3 | 40 | 0,5 | 11,91% | 2,00% | 3,80% | 3,40% | 8,58% | 13,65 |
| 76 | 3 | 16 | 0,5 | 11,91% | 1,58% | 3,80% | 2,76% | 8,08% | 12,93 |
| 80 | 3 | 20 | 0,5 | 11,91% | 1,54% | 3,80% | 2,70% | 8,00% | 13,10 |
| 84 | 3 | 26 | 0,5 | 11,91% | 1,59% | 3,80% | 2,80% | 7,97% | 13,00 |
| 35 | 2 | 4 | 0,4 | 10,64% | 1,68% | 3,80% | 2,85% | 7,68% | 11,72 |
| 75 | 3 | 16 | 0,4 | 11,06% | 1,42% | 3,80% | 2,49% | 7,35% | 13,34 |
| 69 | 3 | 8 | 0,2 | 8,94% | 2,67% | 5,13% | 3,53% | 7,06% | 13,52 |
| 71 | 3 | 8 | 0,4 | 6,81% | 0,93% | 2,74% | 1,59% | 3,65% | 13,55 |
| 67 | 3 | 4 | 0,4 | 6,81% | 0,47% | 2,53% | 0,87% | 2,95% | 12,43 |
| 72 | 3 | 8 | 0,5 | 6,81% | 0,47% | 2,53% | 0,87% | 2,95% | 14,10 |
| 66 | 3 | 4 | 0,3 | 6,81% | 0,47% | 2,53% | 0,87% | 2,94% | 14,24 |
| 68 | 3 | 4 | 0,5 | 2,13% | 0,05% | 1,27% | 0,09% | 0,00% | 13,27 |

Figure C.4: Performance metric scores of all unique architecture parameter settings (based on 5-fold cross-validation).

### C.1.2 Decision Tree

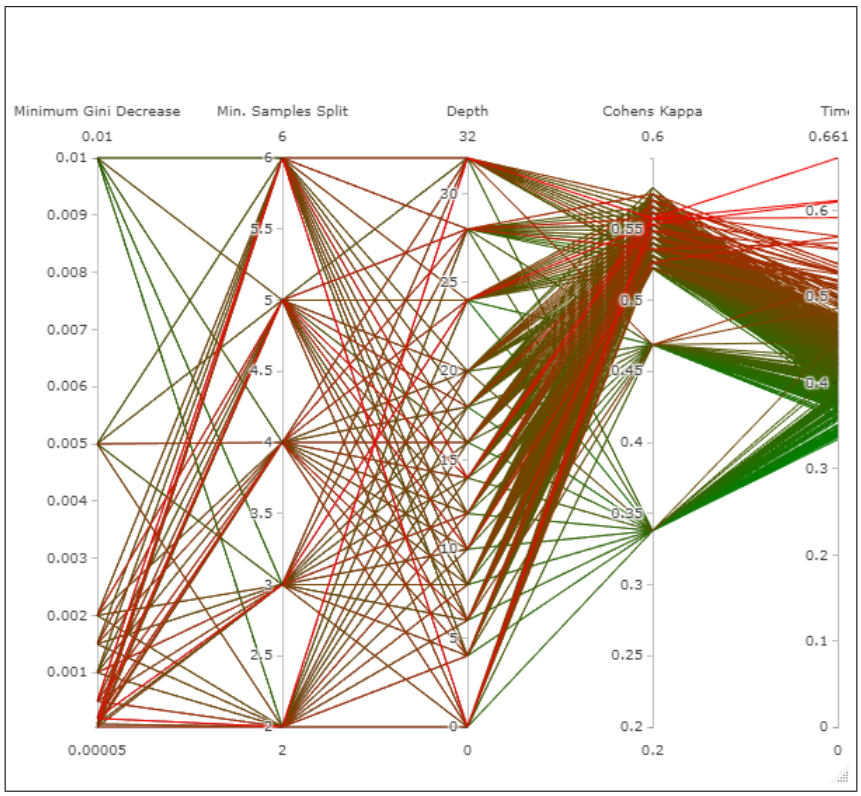



Figure C.5: Parallel Coordinates Plot of the Decision Tree hyper-parameter tuning experiment.

| Gini Decrease | Minimum Samples | Depth | Accuracy | Recall | Specificity | Precision | F1 measure | Cohens kappa | False Positive Rate | Computation Time |
|---|---|---|---|---|---|---|---|---|---|---|
| 0,00050 | 3 | 8 | 99,08% | 47,27% | 99,53% | 47,68% | 47,47% | 57,91% | 0,47% | 0,45 |
| 0,00050 | 3 | 20 | 99,08% | 47,27% | 99,53% | 47,48% | 47,37% | 57,91% | 0,47% | 0,43 |
| 0,00100 | 3 | 8 | 99,07% | 46,01% | 99,53% | 46,84% | 46,42% | 57,48% | 0,47% | 0,47 |
| 0,00050 | 2 | 6 | 99,06% | 47,66% | 99,52% | 47,92% | 47,79% | 57,48% | 0,48% | 0,46 |
| 0,00050 | 2 | 16 | 99,08% | 45,65% | 99,53% | 45,46% | 45,55% | 57,48% | 0,47% | 0,44 |
| 0,00100 | 3 | 28 | 99,06% | 46,52% | 99,52% | 47,91% | 47,20% | 57,48% | 0,48% | 0,42 |
| 0,00050 | 4 | 10 | 99,06% | 47,24% | 99,52% | 47,09% | 47,16% | 57,48% | 0,48% | 0,43 |
| 0,00050 | 4 | 14 | 99,06% | 47,24% | 99,52% | 47,45% | 47,34% | 57,47% | 0,48% | 0,43 |
| 0,00050 | 4 | 20 | 99,07% | 46,71% | 99,53% | 46,83% | 46,77% | 57,47% | 0,47% | 0,42 |
| 0,00020 | 3 | 16 | 99,07% | 45,97% | 99,53% | 44,90% | 45,43% | 57,47% | 0,47% | 0,45 |
| 0,00050 | 4 | None | 99,05% | 47,77% | 99,52% | 47,70% | 47,74% | 57,47% | 0,48% | 0,47 |
| 0,00050 | 4 | 28 | 99,05% | 47,77% | 99,52% | 47,70% | 47,74% | 57,47% | 0,48% | 0,54 |
| 0,00020 | 2 | 8 | 99,06% | 46,49% | 99,52% | 45,93% | 46,21% | 57,47% | 0,48% | 0,43 |
| 0,00050 | 2 | 20 | 99,06% | 46,44% | 99,52% | 46,46% | 46,45% | 57,47% | 0,48% | 0,42 |
| 0,00020 | 2 | 18 | 99,06% | 46,72% | 99,52% | 45,27% | 45,99% | 57,47% | 0,48% | 0,43 |
| 0,00020 | 2 | 4 | 99,06% | 46,49% | 99,52% | 46,19% | 46,34% | 57,47% | 0,48% | 0,42 |
| 0,00020 | 3 | 28 | 99,06% | 45,74% | 99,52% | 43,76% | 44,73% | 57,46% | 0,48% | 0,50 |
| 0,00100 | 3 | 6 | 99,06% | 45,87% | 99,52% | 46,24% | 46,05% | 57,06% | 0,48% | 0,44 |
| 0,00100 | 3 | 12 | 99,05% | 46,38% | 99,52% | 47,32% | 46,85% | 57,06% | 0,48% | 0,54 |
| 0,00100 | 3 | 10 | 99,04% | 46,91% | 99,51% | 47,75% | 47,32% | 57,05% | 0,49% | 0,46 |
| 0,00100 | 3 | 4 | 99,04% | 46,91% | 99,51% | 47,81% | 47,35% | 57,05% | 0,49% | 0,43 |
| 0,00100 | 2 | 10 | 99,03% | 47,45% | 99,51% | 48,85% | 48,14% | 57,05% | 0,49% | 0,43 |
| 0,00100 | 2 | 14 | 99,05% | 45,96% | 99,52% | 46,84% | 46,40% | 57,05% | 0,48% | 0,47 |
| 0,00050 | 2 | 4 | 99,05% | 47,10% | 99,52% | 47,26% | 47,18% | 57,05% | 0,48% | 0,53 |
| 0,00100 | 4 | 24 | 99,05% | 45,96% | 99,52% | 46,24% | 46,10% | 57,05% | 0,48% | 0,42 |
| 0,00050 | 3 | 32 | 99,05% | 47,05% | 99,52% | 47,00% | 47,02% | 57,05% | 0,48% | 0,46 |
| 0,00100 | 4 | 6 | 99,04% | 46,48% | 99,51% | 47,22% | 46,85% | 57,05% | 0,49% | 0,43 |
| 0,00100 | 3 | 32 | 99,04% | 45,91% | 99,51% | 46,06% | 45,99% | 57,05% | 0,49% | 0,42 |
| 0,00050 | 4 | 4 | 99,04% | 47,63% | 99,51% | 47,99% | 47,81% | 57,04% | 0,49% | 0,47 |
| 0,00020 | 4 | 18 | 99,04% | 47,39% | 99,51% | 46,57% | 46,98% | 57,04% | 0,49% | 0,42 |
| 0,00050 | 2 | 12 | 99,05% | 46,53% | 99,52% | 45,95% | 46,24% | 57,04% | 0,48% | 0,43 |
| 0,00100 | 2 | 6 | 99,02% | 47,76% | 99,50% | 49,21% | 48,47% | 57,04% | 0,50% | 0,42 |
| 0,00100 | 3 | 16 | 99,04% | 45,91% | 99,51% | 46,12% | 46,02% | 57,04% | 0,49% | 0,44 |
| 0,00100 | 4 | 12 | 99,02% | 47,56% | 99,50% | 47,72% | 47,64% | 57,04% | 0,50% | 0,43 |
| 0,00100 | 4 | 18 | 99,04% | 45,91% | 99,51% | 45,32% | 45,62% | 57,04% | 0,49% | 0,41 |
| 0,00020 | 3 | 8 | 99,05% | 46,35% | 99,52% | 45,55% | 45,95% | 57,04% | 0,48% | 0,45 |
| 0,00020 | 3 | 18 | 99,07% | 44,37% | 99,53% | 43,95% | 44,16% | 57,04% | 0,47% | 0,42 |
| 0,00020 | 2 | 10 | 99,05% | 45,93% | 99,52% | 45,32% | 45,62% | 57,04% | 0,48% | 0,43 |
| 0,00050 | 2 | 8 | 99,05% | 45,88% | 99,52% | 45,34% | 45,61% | 57,04% | 0,48% | 0,43 |
| 0,00020 | 2 | 16 | 99,05% | 46,16% | 99,52% | 44,79% | 45,46% | 57,04% | 0,48% | 0,47 |
| 0,00020 | 3 | 32 | 99,06% | 45,09% | 99,52% | 43,55% | 44,31% | 57,04% | 0,48% | 0,50 |
| 0,00020 | 2 | 6 | 99,04% | 45,88% | 99,51% | 44,41% | 45,13% | 57,04% | 0,49% | 0,45 |
| 0,00050 | 5 | 10 | 99,03% | 47,70% | 99,51% | 46,42% | 47,05% | 57,03% | 0,49% | 0,43 |
| 0,00020 | 2 | 24 | 99,05% | 45,36% | 99,52% | 44,52% | 44,94% | 57,03% | 0,48% | 0,45 |
| 0,00020 | 3 | 12 | 99,04% | 45,88% | 99,51% | 44,25% | 45,05% | 57,03% | 0,49% | 0,44 |
| 0,00050 | 3 | 12 | 99,06% | 46,34% | 99,52% | 46,97% | 46,65% | 56,63% | 0,48% | 0,44 |
| 0,00020 | 4 | None | 99,04% | 46,72% | 99,51% | 45,88% | 46,30% | 56,62% | 0,49% | 0,48 |
| 0,00050 | 3 | 6 | 99,05% | 46,39% | 99,52% | 45,81% | 46,10% | 56,62% | 0,48% | 0,44 |
| 0,00050 | 3 | 18 | 99,04% | 46,91% | 99,51% | 46,98% | 46,95% | 56,62% | 0,49% | 0,42 |
| 0,00050 | 3 | 24 | 99,04% | 46,35% | 99,51% | 45,59% | 45,97% | 56,62% | 0,49% | 0,42 |
| 0,00100 | 2 | 4 | 99,03% | 45,77% | 99,51% | 46,67% | 46,22% | 56,62% | 0,49% | 0,42 |
| 0,00100 | 2 | 20 | 99,04% | 44,84% | 99,51% | 45,17% | 45,00% | 56,62% | 0,49% | 0,42 |
| 0,00100 | 4 | 4 | 99,02% | 46,30% | 99,50% | 45,86% | 46,08% | 56,62% | 0,50% | 0,42 |
| 0,00010 | 3 | 14 | 99,06% | 45,28% | 99,52% | 45,30% | 45,29% | 56,61% | 0,48% | 0,47 |
| 0,00100 | 3 | 20 | 99,02% | 46,30% | 99,50% | 47,13% | 46,71% | 56,61% | 0,50% | 0,41 |
| 0,00100 | 4 | 20 | 99,01% | 47,42% | 99,50% | 47,64% | 47,53% | 56,61% | 0,50% | 0,41 |
| 0,00010 | 2 | 12 | 99,04% | 46,49% | 99,51% | 46,86% | 46,67% | 56,61% | 0,49% | 0,49 |
| 0,00100 | 2 | 16 | 99,04% | 45,02% | 99,51% | 45,43% | 45,23% | 56,61% | 0,49% | 0,44 |
| 0,00020 | 4 | 8 | 99,04% | 45,74% | 99,51% | 44,12% | 44,92% | 56,61% | 0,49% | 0,44 |
| 0,00050 | 2 | 14 | 99,04% | 46,49% | 99,51% | 45,88% | 46,18% | 56,61% | 0,49% | 0,47 |
| 0,00050 | 2 | 28 | 99,03% | 46,50% | 99,51% | 45,98% | 46,24% | 56,61% | 0,49% | 0,44 |
| 0,00010 | 3 | 16 | 99,04% | 46,30% | 99,51% | 45,98% | 46,14% | 56,61% | 0,49% | 0,45 |
| 0,00010 | 2 | 24 | 99,05% | 45,79% | 99,52% | 45,30% | 45,54% | 56,61% | 0,48% | 0,44 |
| 0,00050 | 3 | 14 | 99,04% | 45,93% | 99,51% | 45,29% | 45,61% | 56,61% | 0,49% | 0,47 |
| 0,00050 | 2 | 32 | 99,03% | 47,21% | 99,51% | 46,30% | 46,75% | 56,61% | 0,49% | 0,46 |
| 0,00100 | 2 | 28 | 99,03% | 45,54% | 99,51% | 45,88% | 45,70% | 56,61% | 0,49% | 0,42 |
| 0,00100 | 4 | 10 | 99,03% | 45,35% | 99,51% | 44,87% | 45,11% | 56,61% | 0,49% | 0,42 |
| 0,00020 | 2 | 20 | 99,06% | 44,55% | 99,52% | 43,68% | 44,11% | 56,61% | 0,48% | 0,48 |
| 0,00100 | 5 | 6 | 99,03% | 45,87% | 99,51% | 44,49% | 45,17% | 56,61% | 0,49% | 0,45 |
| 0,00100 | 5 | 10 | 99,02% | 46,39% | 99,50% | 45,39% | 45,88% | 56,61% | 0,50% | 0,43 |
| 0,00020 | 4 | 24 | 99,04% | 45,74% | 99,51% | 43,62% | 44,65% | 56,61% | 0,49% | 0,53 |
| 0,00100 | 5 | 4 | 99,02% | 45,82% | 99,50% | 44,62% | 45,21% | 56,61% | 0,50% | 0,41 |
| 0,00100 | 2 | 12 | 99,00% | 46,95% | 99,49% | 46,51% | 46,73% | 56,61% | 0,51% | 0,43 |
| 0,00100 | 4 | 32 | 99,02% | 45,87% | 99,50% | 44,70% | 45,27% | 56,61% | 0,50% | 0,43 |
| 0,00100 | 6 | 8 | 99,00% | 47,58% | 99,49% | 47,47% | 47,53% | 56,59% | 0,51% | 0,42 |
| 0,00100 | 6 | None | 99,00% | 47,58% | 99,49% | 46,32% | 46,94% | 56,59% | 0,51% | 0,42 |
| 0,00050 | 6 | 10 | 99,01% | 47,19% | 99,50% | 45,86% | 46,51% | 56,59% | 0,50% | 0,41 |
| 0,00100 | 2 | 8 | 99,04% | 45,59% | 99,52% | 47,26% | 46,41% | 56,20% | 0,48% | 0,47 |
| 0,00100 | 3 | None | 99,04% | 45,59% | 99,52% | 46,65% | 46,11% | 56,20% | 0,48% | 0,42 |
| 0,00050 | 3 | None | 99,02% | 47,77% | 99,50% | 48,70% | 48,23% | 56,19% | 0,50% | 0,49 |
| 0,00100 | 2 | None | 99,04% | 44,94% | 99,52% | 45,95% | 45,44% | 56,19% | 0,48% | 0,42 |
| 0,00020 | 3 | None | 99,06% | 45,28% | 99,52% | 44,80% | 45,04% | 56,19% | 0,48% | 0,44 |
| 0,00050 | 2 | 18 | 99,04% | 45,83% | 99,51% | 45,44% | 45,64% | 56,19% | 0,49% | 0,43 |
| 0,00100 | 4 | None | 99,03% | 44,70% | 99,51% | 45,02% | 44,86% | 56,19% | 0,49% | 0,42 |
| 0,00010 | 3 | None | 99,04% | 45,65% | 99,51% | 44,88% | 45,26% | 56,18% | 0,49% | 0,45 |
| 0,00010 | 3 | 12 | 99,04% | 45,65% | 99,51% | 45,21% | 45,43% | 56,18% | 0,49% | 0,42 |
| 0,00020 | 4 | 32 | 99,03% | 45,60% | 99,51% | 44,46% | 45,02% | 56,18% | 0,49% | 0,45 |
| 0,00020 | 4 | 16 | 99,03% | 45,60% | 99,51% | 44,27% | 44,93% | 56,18% | 0,49% | 0,44 |
| 0,00050 | 5 | 14 | 99,03% | 46,35% | 99,51% | 45,47% | 45,91% | 56,18% | 0,49% | 0,43 |
| 0,00005 | 3 | 24 | 99,03% | 46,35% | 99,51% | 45,98% | 46,16% | 56,18% | 0,49% | 0,43 |
| 0,00010 | 3 | 4 | 99,06% | 44,92% | 99,52% | 44,80% | 44,86% | 56,18% | 0,48% | 0,43 |
| 0,00010 | 3 | 6 | 99,02% | 46,87% | 99,50% | 46,50% | 46,69% | 56,18% | 0,50% | 0,46 |
| 0,00020 | 4 | 10 | 99,03% | 45,60% | 99,51% | 44,18% | 44,88% | 56,18% | 0,49% | 0,42 |
| 0,00050 | 3 | 4 | 99,03% | 45,79% | 99,51% | 44,64% | 45,21% | 56,18% | 0,49% | 0,50 |
| 0,00020 | 3 | 20 | 99,03% | 45,32% | 99,51% | 44,06% | 44,68% | 56,18% | 0,49% | 0,50 |
| 0,00020 | 4 | 4 | 99,01% | 46,65% | 99,50% | 45,10% | 45,86% | 56,18% | 0,50% | 0,41 |
| 0,00005 | 3 | 4 | 99,04% | 45,42% | 99,51% | 45,19% | 45,30% | 56,18% | 0,49% | 0,45 |
| 0,00050 | 5 | 24 | 99,01% | 47,41% | 99,50% | 46,31% | 46,86% | 56,18% | 0,50% | 0,43 |
| 0,00005 | 3 | 28 | 99,03% | 45,93% | 99,51% | 45,27% | 45,59% | 56,18% | 0,49% | 0,43 |
| 0,00005 | 2 | None | 99,01% | 46,98% | 99,50% | 46,35% | 46,66% | 56,17% | 0,50% | 0,51 |
| 0,00020 | 5 | 4 | 99,01% | 46,93% | 99,50% | 45,22% | 46,06% | 56,17% | 0,50% | 0,51 |

Figure C.6: Performance metric scores of all unique DT parameter settings (based on 5-fold cross-validation).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0,00020 | 2 | 14 | 99,03% | 44,66% | 99,51% | 43,05% | 43,84% | 56,17% | 0,49% | 0,43 |
| 0,00005 | 3 | 12 | 99,02% | 46,45% | 99,50% | 45,63% | 46,04% | 56,17% | 0,50% | 0,41 |
| 0,00020 | 5 | 16 | 99,02% | 46,40% | 99,50% | 44,76% | 45,57% | 56,17% | 0,50% | 0,43 |
| 0,00050 | 5 | 8 | 99,02% | 46,40% | 99,50% | 44,57% | 45,47% | 56,17% | 0,50% | 0,43 |
| 0,00050 | 5 | 4 | 99,02% | 46,40% | 99,50% | 44,57% | 45,47% | 56,17% | 0,50% | 0,45 |
| 0,00050 | 6 | 12 | 99,02% | 46,97% | 99,50% | 46,24% | 46,60% | 56,17% | 0,50% | 0,46 |
| 0,00050 | 6 | 6 | 99,01% | 47,51% | 99,50% | 46,77% | 47,14% | 56,17% | 0,50% | 0,41 |
| 0,00100 | 5 | 16 | 99,00% | 45,77% | 99,49% | 44,17% | 44,96% | 56,17% | 0,51% | 0,43 |
| 0,00100 | 5 | 24 | 99,00% | 45,77% | 99,49% | 44,17% | 44,96% | 56,17% | 0,51% | 0,42 |
| 0,00100 | 5 | 14 | 99,00% | 45,77% | 99,49% | 44,73% | 45,25% | 56,17% | 0,51% | 0,42 |
| 0,00100 | 5 | 20 | 99,01% | 45,24% | 99,50% | 44,04% | 44,64% | 56,17% | 0,50% | 0,43 |
| 0,00050 | 5 | 16 | 99,02% | 46,40% | 99,50% | 44,17% | 45,26% | 56,17% | 0,50% | 0,42 |
| 0,00100 | 5 | None | 99,00% | 45,77% | 99,49% | 43,96% | 44,85% | 56,17% | 0,51% | 0,45 |
| 0,00100 | 6 | 4 | 99,01% | 45,34% | 99,50% | 44,68% | 45,01% | 56,16% | 0,50% | 0,41 |
| 0,00020 | 6 | None | 99,01% | 47,03% | 99,50% | 45,71% | 46,36% | 56,16% | 0,50% | 0,42 |
| 0,00100 | 6 | 24 | 99,00% | 45,87% | 99,49% | 44,61% | 45,23% | 56,16% | 0,51% | 0,41 |
| 0,00050 | 6 | 8 | 99,01% | 47,03% | 99,50% | 45,52% | 46,26% | 56,16% | 0,50% | 0,42 |
| 0,00050 | 6 | 32 | 99,01% | 47,03% | 99,50% | 45,33% | 46,17% | 56,16% | 0,50% | 0,42 |
| 0,00100 | 6 | 32 | 98,99% | 46,41% | 99,49% | 45,14% | 45,76% | 56,16% | 0,51% | 0,40 |
| 0,00020 | 6 | 4 | 99,01% | 47,03% | 99,50% | 45,14% | 46,07% | 56,16% | 0,50% | 0,43 |
| 0,00020 | 6 | 14 | 99,01% | 47,03% | 99,50% | 44,76% | 45,87% | 56,16% | 0,50% | 0,57 |
| 0,00100 | 3 | 14 | 99,02% | 45,96% | 99,50% | 47,09% | 46,52% | 55,77% | 0,50% | 0,42 |
| 0,00020 | 4 | 12 | 99,02% | 46,44% | 99,50% | 46,18% | 46,31% | 55,76% | 0,50% | 0,45 |
| 0,00050 | 3 | 28 | 99,02% | 47,05% | 99,50% | 46,58% | 46,81% | 55,76% | 0,50% | 0,44 |
| 0,00020 | 4 | 28 | 99,04% | 45,92% | 99,51% | 45,46% | 45,69% | 55,76% | 0,49% | 0,56 |
| 0,00050 | 3 | 10 | 99,04% | 45,60% | 99,51% | 45,14% | 45,37% | 55,76% | 0,49% | 0,47 |
| 0,00100 | 3 | 18 | 99,01% | 45,49% | 99,50% | 45,95% | 45,72% | 55,76% | 0,50% | 0,42 |
| 0,00020 | 4 | 6 | 99,01% | 46,97% | 99,50% | 46,38% | 46,67% | 55,76% | 0,50% | 0,47 |
| 0,00050 | 3 | 16 | 99,02% | 46,63% | 99,50% | 45,67% | 46,14% | 55,76% | 0,50% | 0,45 |
| 0,00005 | 4 | 4 | 99,01% | 46,83% | 99,50% | 46,59% | 46,71% | 55,76% | 0,50% | 0,45 |
| 0,00010 | 4 | 16 | 99,02% | 46,30% | 99,50% | 46,04% | 46,17% | 55,76% | 0,50% | 0,53 |
| 0,00010 | 4 | 24 | 99,02% | 46,30% | 99,50% | 45,48% | 45,89% | 55,76% | 0,50% | 0,48 |
| 0,00100 | 4 | 16 | 98,99% | 47,13% | 99,49% | 47,52% | 47,32% | 55,76% | 0,51% | 0,46 |
| 0,00010 | 4 | 18 | 99,02% | 46,30% | 99,50% | 46,12% | 46,21% | 55,76% | 0,50% | 0,43 |
| 0,00050 | 4 | 18 | 99,00% | 47,75% | 99,49% | 47,44% | 47,59% | 55,76% | 0,51% | 0,47 |
| 0,00010 | 4 | 10 | 99,02% | 46,30% | 99,50% | 44,99% | 45,64% | 55,75% | 0,50% | 0,43 |
| 0,00100 | 5 | 12 | 98,99% | 47,08% | 99,49% | 45,37% | 46,21% | 55,75% | 0,51% | 0,42 |
| 0,00005 | 2 | 18 | 99,02% | 45,60% | 99,50% | 43,58% | 44,57% | 55,75% | 0,50% | 0,46 |
| 0,00005 | 3 | None | 99,05% | 43,82% | 99,52% | 43,47% | 43,64% | 55,75% | 0,48% | 0,44 |
| 0,00020 | 2 | 28 | 99,02% | 44,52% | 99,50% | 42,62% | 43,55% | 55,75% | 0,50% | 0,46 |
| 0,00020 | 2 | 32 | 99,04% | 45,00% | 99,51% | 44,38% | 44,69% | 55,75% | 0,49% | 0,66 |
| 0,00050 | 5 | 20 | 99,01% | 46,73% | 99,50% | 45,18% | 45,94% | 55,75% | 0,50% | 0,42 |
| 0,00005 | 2 | 4 | 99,02% | 45,79% | 99,50% | 45,60% | 45,69% | 55,75% | 0,50% | 0,48 |
| 0,00010 | 3 | 28 | 99,04% | 44,54% | 99,51% | 43,75% | 44,14% | 55,75% | 0,49% | 0,43 |
| 0,00010 | 2 | 32 | 99,04% | 44,86% | 99,51% | 44,30% | 44,58% | 55,75% | 0,49% | 0,46 |
| 0,00020 | 2 | None | 99,01% | 45,26% | 99,50% | 43,46% | 44,34% | 55,75% | 0,50% | 0,47 |
| 0,00005 | 2 | 12 | 99,02% | 44,80% | 99,50% | 44,15% | 44,47% | 55,74% | 0,50% | 0,53 |
| 0,00010 | 2 | 18 | 99,04% | 45,05% | 99,51% | 45,02% | 45,03% | 55,74% | 0,49% | 0,43 |
| 0,00050 | 4 | 24 | 99,02% | 45,22% | 99,50% | 44,08% | 44,64% | 55,74% | 0,50% | 0,42 |
| 0,00050 | 6 | 14 | 99,01% | 46,83% | 99,50% | 46,18% | 46,50% | 55,74% | 0,50% | 0,44 |
| 0,00100 | 6 | 20 | 99,00% | 46,20% | 99,49% | 46,11% | 46,16% | 55,74% | 0,51% | 0,40 |
| 0,00005 | 3 | 14 | 99,04% | 44,30% | 99,51% | 43,89% | 44,10% | 55,74% | 0,49% | 0,42 |
| 0,00010 | 2 | 4 | 99,04% | 44,12% | 99,51% | 42,34% | 43,21% | 55,74% | 0,49% | 0,45 |
| 0,00050 | 4 | 8 | 99,01% | 45,74% | 99,50% | 44,37% | 45,04% | 55,74% | 0,50% | 0,42 |
| 0,00005 | 3 | 32 | 99,02% | 44,80% | 99,50% | 43,45% | 44,12% | 55,74% | 0,50% | 0,42 |
| 0,00050 | 5 | 12 | 99,02% | 44,76% | 99,50% | 42,76% | 43,73% | 55,74% | 0,50% | 0,47 |
| 0,00010 | 2 | 10 | 99,00% | 45,64% | 99,49% | 43,39% | 44,49% | 55,74% | 0,51% | 0,47 |
| 0,00020 | 5 | 14 | 99,02% | 44,76% | 99,50% | 42,57% | 43,64% | 55,74% | 0,50% | 0,59 |
| 0,00020 | 6 | 16 | 99,01% | 45,36% | 99,50% | 43,30% | 44,31% | 55,73% | 0,50% | 0,46 |
| 0,00050 | 6 | None | 99,01% | 45,36% | 99,50% | 43,30% | 44,31% | 55,73% | 0,50% | 0,61 |
| 0,00020 | 6 | 20 | 98,98% | 46,96% | 99,48% | 44,62% | 45,76% | 55,72% | 0,52% | 0,46 |
| 0,00100 | 2 | 18 | 99,03% | 44,89% | 99,51% | 46,07% | 45,47% | 55,34% | 0,49% | 0,44 |
| 0,00050 | 2 | None | 99,02% | 46,49% | 99,50% | 46,86% | 46,67% | 55,33% | 0,50% | 0,49 |
| 0,00100 | 3 | 24 | 99,03% | 44,34% | 99,51% | 44,96% | 44,65% | 55,33% | 0,49% | 0,42 |
| 0,00020 | 4 | 14 | 99,04% | 44,32% | 99,51% | 43,45% | 43,88% | 55,33% | 0,49% | 0,51 |
| 0,00010 | 4 | 20 | 99,03% | 45,65% | 99,51% | 45,44% | 45,54% | 55,33% | 0,49% | 0,45 |
| 0,00100 | 4 | 8 | 99,02% | 44,42% | 99,50% | 44,37% | 44,39% | 55,33% | 0,50% | 0,42 |
| 0,00005 | 4 | None | 99,02% | 46,16% | 99,50% | 45,84% | 46,00% | 55,33% | 0,50% | 0,43 |
| 0,00005 | 4 | 24 | 99,02% | 46,16% | 99,50% | 45,84% | 46,00% | 55,33% | 0,50% | 0,43 |
| 0,00020 | 3 | 24 | 99,04% | 43,82% | 99,51% | 43,56% | 43,69% | 55,33% | 0,49% | 0,61 |
| 0,00005 | 4 | 28 | 98,99% | 46,65% | 99,49% | 45,11% | 45,87% | 55,33% | 0,51% | 0,42 |
| 0,00010 | 4 | 32 | 98,99% | 47,22% | 99,49% | 46,24% | 46,73% | 55,33% | 0,51% | 0,45 |
| 0,00010 | 4 | 14 | 99,00% | 46,68% | 99,49% | 46,34% | 46,51% | 55,32% | 0,51% | 0,45 |
| 0,00010 | 4 | 6 | 99,02% | 45,18% | 99,50% | 44,36% | 44,76% | 55,32% | 0,50% | 0,48 |
| 0,00010 | 3 | 32 | 98,99% | 46,93% | 99,49% | 46,73% | 46,83% | 55,32% | 0,51% | 0,44 |
| 0,00010 | 4 | 12 | 99,02% | 45,18% | 99,50% | 44,36% | 44,77% | 55,32% | 0,50% | 0,45 |
| 0,00005 | 2 | 8 | 99,04% | 44,00% | 99,51% | 43,74% | 43,87% | 55,32% | 0,49% | 0,51 |
| 0,00010 | 2 | 8 | 99,02% | 45,41% | 99,50% | 46,13% | 45,77% | 55,32% | 0,50% | 0,44 |
| 0,00010 | 4 | 8 | 99,02% | 45,18% | 99,50% | 44,24% | 44,71% | 55,32% | 0,50% | 0,47 |
| 0,00005 | 4 | 12 | 99,02% | 45,18% | 99,50% | 43,98% | 44,57% | 55,32% | 0,50% | 0,44 |
| 0,00005 | 4 | 32 | 99,00% | 45,69% | 99,49% | 44,75% | 45,21% | 55,32% | 0,51% | 0,42 |
| 0,00050 | 6 | 28 | 98,99% | 47,65% | 99,49% | 47,22% | 47,44% | 55,32% | 0,51% | 0,41 |
| 0,00020 | 3 | 4 | 99,03% | 44,30% | 99,51% | 43,06% | 43,67% | 55,32% | 0,49% | 0,45 |
| 0,00020 | 3 | 10 | 99,00% | 45,31% | 99,49% | 43,66% | 44,47% | 55,32% | 0,51% | 0,44 |
| 0,00020 | 2 | 12 | 99,03% | 43,89% | 99,51% | 42,46% | 43,16% | 55,32% | 0,49% | 0,45 |
| 0,00005 | 4 | 10 | 99,00% | 45,69% | 99,49% | 44,73% | 45,21% | 55,32% | 0,51% | 0,48 |
| 0,00020 | 3 | 6 | 99,02% | 44,80% | 99,50% | 44,20% | 44,50% | 55,32% | 0,50% | 0,45 |
| 0,00005 | 3 | 18 | 99,03% | 44,17% | 99,51% | 43,56% | 43,86% | 55,32% | 0,49% | 0,45 |
| 0,00010 | 2 | 16 | 99,02% | 45,22% | 99,50% | 44,46% | 44,84% | 55,32% | 0,50% | 0,46 |
| 0,00020 | 3 | 14 | 99,00% | 45,31% | 99,49% | 44,11% | 44,71% | 55,32% | 0,51% | 0,43 |
| 0,00020 | 5 | 20 | 99,02% | 45,60% | 99,50% | 43,98% | 44,77% | 55,32% | 0,50% | 0,49 |
| 0,00010 | 3 | 8 | 99,03% | 44,17% | 99,51% | 43,73% | 43,94% | 55,32% | 0,49% | 0,43 |
| 0,00010 | 2 | 20 | 99,03% | 43,93% | 99,51% | 43,07% | 43,50% | 55,32% | 0,49% | 0,42 |
| 0,00010 | 3 | 20 | 99,03% | 44,17% | 99,51% | 43,54% | 43,85% | 55,32% | 0,49% | 0,42 |
| 0,00150 | 4 | 24 | 98,99% | 46,21% | 99,49% | 45,89% | 46,05% | 55,31% | 0,51% | 0,42 |
| 0,00005 | 2 | 14 | 99,03% | 43,75% | 99,51% | 42,28% | 43,00% | 55,31% | 0,49% | 0,44 |
| 0,00005 | 3 | 20 | 99,02% | 44,66% | 99,50% | 44,24% | 44,45% | 55,31% | 0,50% | 0,43 |
| 0,00020 | 5 | 12 | 98,99% | 46,65% | 99,49% | 44,36% | 45,47% | 55,31% | 0,51% | 0,48 |
| 0,00050 | 5 | 18 | 99,00% | 46,12% | 99,49% | 44,46% | 45,27% | 55,31% | 0,51% | 0,42 |
| 0,00100 | 6 | 18 | 98,99% | 46,06% | 99,49% | 46,06% | 46,06% | 55,31% | 0,51% | 0,41 |

Figure C.7: Continued – Performance metric scores of all unique DT parameter settings (based on 5-fold cross-validation).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0,00020 | 5 | 10 | 99,02% | 45,60% | 99,50% | 43,23% | 44,38% | 55,31% | 0,50% | 0,47 |
| 0,00100 | 5 | 8 | 98,98% | 45,48% | 99,48% | 43,88% | 44,66% | 55,31% | 0,52% | 0,42 |
| 0,00100 | 5 | 28 | 98,98% | 45,48% | 99,48% | 43,88% | 44,66% | 55,31% | 0,52% | 0,42 |
| 0,00150 | 2 | 12 | 98,98% | 46,75% | 99,48% | 45,85% | 46,29% | 55,31% | 0,52% | 0,43 |
| 0,00005 | 2 | 28 | 99,02% | 44,47% | 99,50% | 43,17% | 43,81% | 55,31% | 0,50% | 0,42 |
| 0,00020 | 5 | None | 99,00% | 46,12% | 99,49% | 43,72% | 44,89% | 55,31% | 0,51% | 0,44 |
| 0,00050 | 5 | 6 | 99,00% | 45,55% | 99,49% | 42,96% | 44,22% | 55,31% | 0,51% | 0,42 |
| 0,00100 | 6 | 14 | 98,98% | 46,59% | 99,48% | 46,59% | 46,59% | 55,31% | 0,52% | 0,42 |
| 0,00100 | 6 | 16 | 98,98% | 46,59% | 99,48% | 46,59% | 46,59% | 55,31% | 0,52% | 0,42 |
| 0,00150 | 2 | None | 98,98% | 46,75% | 99,48% | 45,26% | 45,99% | 55,31% | 0,52% | 0,42 |
| 0,00150 | 3 | 10 | 98,98% | 46,75% | 99,48% | 45,26% | 45,99% | 55,31% | 0,52% | 0,44 |
| 0,00150 | 3 | 12 | 98,98% | 46,75% | 99,48% | 46,41% | 46,58% | 55,31% | 0,52% | 0,44 |
| 0,00150 | 2 | 18 | 98,98% | 46,75% | 99,48% | 46,41% | 46,58% | 55,31% | 0,52% | 0,46 |
| 0,00150 | 2 | 28 | 98,98% | 46,75% | 99,48% | 45,26% | 45,99% | 55,31% | 0,52% | 0,44 |
| 0,00150 | 3 | 28 | 98,98% | 46,75% | 99,48% | 45,26% | 45,99% | 55,31% | 0,52% | 0,46 |
| 0,00005 | 2 | 20 | 99,02% | 44,66% | 99,50% | 43,64% | 44,14% | 55,31% | 0,50% | 0,49 |
| 0,00100 | 5 | 18 | 98,97% | 46,02% | 99,48% | 44,38% | 45,18% | 55,31% | 0,52% | 0,42 |
| 0,00005 | 2 | 6 | 99,03% | 43,98% | 99,50% | 42,41% | 43,18% | 55,31% | 0,50% | 0,46 |
| 0,00050 | 5 | None | 98,98% | 47,19% | 99,48% | 45,10% | 46,12% | 55,31% | 0,52% | 0,50 |
| 0,00150 | 4 | 18 | 98,97% | 47,30% | 99,48% | 45,78% | 46,53% | 55,31% | 0,52% | 0,43 |
| 0,00005 | 2 | 32 | 98,99% | 45,50% | 99,49% | 43,86% | 44,66% | 55,31% | 0,51% | 0,42 |
| 0,00100 | 6 | 28 | 98,99% | 45,05% | 99,49% | 44,39% | 44,72% | 55,31% | 0,51% | 0,42 |
| 0,00020 | 6 | 24 | 99,00% | 46,21% | 99,49% | 44,72% | 45,45% | 55,31% | 0,51% | 0,45 |
| 0,00050 | 6 | 18 | 98,99% | 46,74% | 99,49% | 45,24% | 45,98% | 55,31% | 0,51% | 0,42 |
| 0,00100 | 6 | 10 | 98,98% | 45,58% | 99,48% | 44,32% | 44,94% | 55,31% | 0,52% | 0,41 |
| 0,00100 | 6 | 12 | 98,98% | 45,58% | 99,48% | 44,32% | 44,94% | 55,31% | 0,52% | 0,43 |
| 0,00050 | 6 | 16 | 98,99% | 46,17% | 99,49% | 43,89% | 45,00% | 55,30% | 0,51% | 0,42 |
| 0,00050 | 6 | 4 | 98,98% | 47,29% | 99,48% | 45,18% | 46,21% | 55,30% | 0,52% | 0,42 |
| 0,00100 | 2 | 24 | 99,01% | 45,02% | 99,50% | 46,34% | 45,67% | 54,91% | 0,50% | 0,43 |
| 0,00005 | 3 | 6 | 99,02% | 45,00% | 99,50% | 45,06% | 45,03% | 54,90% | 0,50% | 0,45 |
| 0,00020 | 4 | 20 | 98,99% | 45,69% | 99,49% | 44,60% | 45,14% | 54,90% | 0,51% | 0,47 |
| 0,00050 | 4 | 12 | 99,01% | 45,93% | 99,49% | 45,83% | 45,88% | 54,90% | 0,51% | 0,44 |
| 0,00050 | 4 | 6 | 99,01% | 45,93% | 99,49% | 45,64% | 45,78% | 54,90% | 0,51% | 0,56 |
| 0,00050 | 4 | 16 | 99,01% | 45,93% | 99,49% | 45,27% | 45,59% | 54,90% | 0,51% | 0,44 |
| 0,00010 | 4 | 28 | 99,02% | 44,54% | 99,50% | 43,72% | 44,12% | 54,90% | 0,50% | 0,42 |
| 0,00010 | 4 | 4 | 99,01% | 45,04% | 99,49% | 44,02% | 44,52% | 54,89% | 0,51% | 0,51 |
| 0,00050 | 4 | 32 | 98,99% | 46,45% | 99,49% | 46,16% | 46,30% | 54,89% | 0,51% | 0,45 |
| 0,00100 | 4 | 28 | 98,99% | 44,78% | 99,49% | 44,54% | 44,66% | 54,89% | 0,51% | 0,43 |
| 0,00005 | 2 | 16 | 99,02% | 44,26% | 99,50% | 43,02% | 43,63% | 54,89% | 0,50% | 0,47 |
| 0,00010 | 3 | 24 | 99,01% | 45,74% | 99,49% | 45,48% | 45,61% | 54,89% | 0,51% | 0,46 |
| 0,00100 | 4 | 14 | 98,97% | 45,82% | 99,48% | 45,17% | 45,49% | 54,89% | 0,52% | 0,47 |
| 0,00005 | 4 | 14 | 98,99% | 45,55% | 99,49% | 44,46% | 45,00% | 54,89% | 0,51% | 0,41 |
| 0,00005 | 4 | 20 | 98,98% | 46,07% | 99,48% | 45,15% | 45,61% | 54,89% | 0,52% | 0,42 |
| 0,00010 | 5 | 4 | 98,98% | 46,36% | 99,48% | 45,03% | 45,68% | 54,88% | 0,52% | 0,47 |
| 0,00100 | 5 | 32 | 98,97% | 45,33% | 99,48% | 43,63% | 44,46% | 54,88% | 0,52% | 0,42 |
| 0,00050 | 6 | 20 | 98,98% | 46,50% | 99,48% | 45,11% | 45,80% | 54,88% | 0,52% | 0,42 |
| 0,00005 | 5 | 6 | 98,98% | 46,36% | 99,48% | 44,70% | 45,51% | 54,88% | 0,52% | 0,46 |
| 0,00150 | 3 | 24 | 98,98% | 45,06% | 99,48% | 44,57% | 44,82% | 54,88% | 0,52% | 0,45 |
| 0,00150 | 4 | None | 98,98% | 45,06% | 99,48% | 44,00% | 44,52% | 54,88% | 0,52% | 0,52 |
| 0,00020 | 5 | 18 | 99,01% | 44,47% | 99,49% | 41,92% | 43,16% | 54,88% | 0,51% | 0,53 |
| 0,00150 | 2 | 16 | 98,97% | 45,59% | 99,48% | 44,51% | 45,04% | 54,88% | 0,52% | 0,44 |
| 0,00020 | 5 | 8 | 98,99% | 44,98% | 99,49% | 42,39% | 43,65% | 54,88% | 0,51% | 0,45 |
| 0,00150 | 5 | 14 | 98,97% | 46,12% | 99,48% | 43,85% | 44,96% | 54,88% | 0,52% | 0,45 |
| 0,00010 | 6 | 6 | 98,98% | 46,45% | 99,48% | 45,47% | 45,96% | 54,88% | 0,52% | 0,42 |
| 0,00150 | 4 | 8 | 98,96% | 46,12% | 99,47% | 45,01% | 45,56% | 54,88% | 0,53% | 0,41 |
| 0,00150 | 4 | 14 | 98,98% | 45,06% | 99,48% | 45,35% | 45,20% | 54,88% | 0,52% | 0,44 |
| 0,00150 | 3 | 18 | 98,98% | 45,06% | 99,48% | 45,35% | 45,20% | 54,88% | 0,52% | 0,46 |
| 0,00150 | 2 | 24 | 98,98% | 45,06% | 99,48% | 45,35% | 45,20% | 54,88% | 0,52% | 0,43 |
| 0,00150 | 3 | 32 | 98,98% | 45,06% | 99,48% | 45,35% | 45,20% | 54,88% | 0,52% | 0,41 |
| 0,00005 | 6 | 4 | 98,99% | 45,93% | 99,49% | 44,57% | 45,24% | 54,87% | 0,51% | 0,44 |
| 0,00150 | 3 | None | 98,98% | 45,06% | 99,48% | 44,77% | 44,92% | 54,87% | 0,52% | 0,42 |
| 0,00150 | 4 | 4 | 98,97% | 45,59% | 99,48% | 45,29% | 45,44% | 54,87% | 0,52% | 0,42 |
| 0,00150 | 4 | 12 | 98,97% | 45,59% | 99,48% | 45,29% | 45,44% | 54,87% | 0,52% | 0,42 |
| 0,00150 | 4 | 16 | 98,97% | 45,59% | 99,48% | 45,29% | 45,44% | 54,87% | 0,52% | 0,42 |
| 0,00150 | 3 | 20 | 98,97% | 45,59% | 99,48% | 45,29% | 45,44% | 54,87% | 0,52% | 0,43 |
| 0,00150 | 4 | 32 | 98,97% | 45,59% | 99,48% | 45,29% | 45,44% | 54,87% | 0,52% | 0,46 |
| 0,00150 | 5 | 32 | 98,96% | 46,66% | 99,47% | 44,35% | 45,48% | 54,87% | 0,53% | 0,42 |
| 0,00010 | 6 | 12 | 98,97% | 47,00% | 99,48% | 46,17% | 46,58% | 54,87% | 0,52% | 0,47 |
| 0,00150 | 4 | 6 | 98,97% | 45,59% | 99,48% | 44,71% | 45,14% | 54,87% | 0,52% | 0,41 |
| 0,00150 | 2 | 8 | 98,97% | 45,59% | 99,48% | 44,71% | 45,14% | 54,87% | 0,52% | 0,41 |
| 0,00150 | 4 | 10 | 98,97% | 45,59% | 99,48% | 44,71% | 45,14% | 54,87% | 0,52% | 0,41 |
| 0,00150 | 2 | 32 | 98,97% | 45,59% | 99,48% | 45,86% | 45,72% | 54,87% | 0,52% | 0,41 |
| 0,00020 | 6 | 6 | 98,99% | 45,07% | 99,49% | 43,21% | 44,12% | 54,87% | 0,51% | 0,42 |
| 0,00010 | 6 | 32 | 98,96% | 47,55% | 99,47% | 46,32% | 46,93% | 54,87% | 0,53% | 0,45 |
| 0,00050 | 6 | 24 | 98,98% | 45,59% | 99,48% | 43,70% | 44,63% | 54,87% | 0,52% | 0,41 |
| 0,00005 | 4 | 16 | 98,98% | 45,83% | 99,48% | 45,03% | 45,43% | 54,47% | 0,52% | 0,43 |
| 0,00010 | 3 | 10 | 99,01% | 44,86% | 99,50% | 45,53% | 45,19% | 54,47% | 0,50% | 0,45 |
| 0,00010 | 3 | 18 | 99,01% | 44,86% | 99,50% | 45,46% | 45,16% | 54,47% | 0,50% | 0,42 |
| 0,00100 | 2 | 32 | 99,00% | 43,90% | 99,49% | 44,54% | 44,22% | 54,47% | 0,51% | 0,43 |
| 0,00050 | 5 | 28 | 98,98% | 46,26% | 99,48% | 44,54% | 45,38% | 54,46% | 0,52% | 0,42 |
| 0,00050 | 2 | 10 | 99,00% | 44,99% | 99,49% | 44,99% | 44,99% | 54,46% | 0,51% | 0,43 |
| 0,00050 | 2 | 24 | 99,01% | 44,07% | 99,50% | 43,50% | 43,78% | 54,46% | 0,50% | 0,44 |
| 0,00050 | 5 | 32 | 98,98% | 45,31% | 99,48% | 43,98% | 44,63% | 54,46% | 0,52% | 0,43 |
| 0,00010 | 2 | 6 | 98,98% | 44,89% | 99,48% | 44,04% | 44,46% | 54,46% | 0,52% | 0,45 |
| 0,00100 | 6 | 6 | 98,96% | 46,30% | 99,47% | 45,94% | 46,12% | 54,46% | 0,53% | 0,46 |
| 0,00005 | 3 | 16 | 99,00% | 44,38% | 99,49% | 44,01% | 44,20% | 54,45% | 0,51% | 0,45 |
| 0,00010 | 5 | 6 | 98,98% | 44,70% | 99,48% | 42,98% | 43,82% | 54,45% | 0,52% | 0,45 |
| 0,00010 | 5 | None | 99,00% | 44,19% | 99,49% | 42,38% | 43,27% | 54,45% | 0,51% | 0,50 |
| 0,00005 | 5 | 14 | 98,98% | 44,70% | 99,48% | 42,81% | 43,73% | 54,45% | 0,52% | 0,41 |
| 0,00010 | 2 | 14 | 99,00% | 43,73% | 99,49% | 42,25% | 42,98% | 54,45% | 0,51% | 0,46 |
| 0,00005 | 5 | 16 | 98,98% | 44,70% | 99,48% | 42,49% | 43,56% | 54,45% | 0,52% | 0,43 |
| 0,00010 | 5 | 14 | 99,00% | 44,19% | 99,49% | 42,20% | 43,17% | 54,45% | 0,51% | 0,44 |
| 0,00150 | 6 | 6 | 98,95% | 46,37% | 99,47% | 44,17% | 45,24% | 54,44% | 0,53% | 0,41 |
| 0,00150 | 6 | 32 | 98,95% | 46,37% | 99,47% | 44,17% | 45,24% | 54,44% | 0,53% | 0,46 |
| 0,00020 | 6 | 18 | 98,98% | 44,93% | 99,48% | 42,96% | 43,92% | 54,44% | 0,52% | 0,45 |
| 0,00150 | 5 | 6 | 98,97% | 44,44% | 99,48% | 42,66% | 43,53% | 54,44% | 0,52% | 0,42 |
| 0,00150 | 6 | 10 | 98,94% | 46,92% | 99,46% | 44,68% | 45,77% | 54,44% | 0,54% | 0,40 |
| 0,00150 | 5 | 24 | 98,97% | 44,44% | 99,48% | 42,66% | 43,53% | 54,44% | 0,52% | 0,43 |
| 0,00020 | 6 | 8 | 98,97% | 45,45% | 99,48% | 43,45% | 44,43% | 54,44% | 0,52% | 0,41 |
| 0,00150 | 5 | 18 | 98,96% | 44,96% | 99,47% | 42,57% | 43,73% | 54,44% | 0,53% | 0,52 |

Figure C.8: Continued – Performance metric scores of all unique DT parameter settings (based on 5-fold cross-validation).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0,00150 | 3 | 6 | 98,97% | 43,91% | 99,48% | 44,02% | 43,97% | 54,44% | 0,52% | 0,41 |
| 0,00150 | 2 | 10 | 98,97% | 43,91% | 99,48% | 44,02% | 43,97% | 54,44% | 0,52% | 0,41 |
| 0,00150 | 4 | 28 | 98,97% | 43,91% | 99,48% | 44,02% | 43,97% | 54,44% | 0,52% | 0,45 |
| 0,00150 | 2 | 20 | 98,97% | 43,91% | 99,48% | 43,45% | 43,68% | 54,44% | 0,52% | 0,47 |
| 0,00150 | 2 | 4 | 98,96% | 44,42% | 99,47% | 43,95% | 44,19% | 54,44% | 0,53% | 0,42 |
| 0,00150 | 3 | 4 | 98,96% | 44,42% | 99,47% | 43,95% | 44,19% | 54,44% | 0,53% | 0,42 |
| 0,00005 | 6 | 6 | 98,97% | 45,31% | 99,48% | 43,80% | 44,54% | 54,44% | 0,52% | 0,49 |
| 0,00150 | 2 | 14 | 98,96% | 44,42% | 99,47% | 43,94% | 44,18% | 54,44% | 0,53% | 0,43 |
| 0,00010 | 6 | 24 | 98,96% | 45,83% | 99,47% | 44,24% | 45,02% | 54,43% | 0,53% | 0,50 |
| 0,00005 | 6 | 20 | 98,96% | 45,83% | 99,47% | 43,90% | 44,85% | 54,43% | 0,53% | 0,42 |
| 0,00010 | 6 | 28 | 98,96% | 45,83% | 99,47% | 43,92% | 44,86% | 54,43% | 0,53% | 0,42 |
| 0,00005 | 2 | 10 | 99,01% | 43,82% | 99,50% | 43,99% | 43,90% | 54,04% | 0,50% | 0,45 |
| 0,00005 | 4 | 6 | 98,97% | 45,26% | 99,48% | 44,79% | 45,03% | 54,04% | 0,52% | 0,42 |
| 0,00005 | 4 | 18 | 98,97% | 45,26% | 99,48% | 43,94% | 44,59% | 54,04% | 0,52% | 0,43 |
| 0,00020 | 5 | 6 | 98,99% | 45,18% | 99,49% | 44,04% | 44,60% | 54,03% | 0,51% | 0,44 |
| 0,00010 | 4 | None | 98,99% | 44,76% | 99,49% | 43,40% | 44,07% | 54,03% | 0,51% | 0,46 |
| 0,00150 | 2 | 6 | 98,96% | 45,78% | 99,47% | 45,70% | 45,74% | 54,03% | 0,53% | 0,41 |
| 0,00020 | 5 | 24 | 98,97% | 45,69% | 99,48% | 43,98% | 44,82% | 54,03% | 0,52% | 0,47 |
| 0,00150 | 3 | 8 | 98,96% | 45,78% | 99,47% | 45,13% | 45,45% | 54,03% | 0,53% | 0,42 |
| 0,00150 | 3 | 16 | 98,96% | 45,78% | 99,47% | 45,13% | 45,45% | 54,03% | 0,53% | 0,43 |
| 0,00005 | 3 | 10 | 99,00% | 43,75% | 99,49% | 43,31% | 43,53% | 54,03% | 0,51% | 0,49 |
| 0,00005 | 5 | 28 | 98,97% | 45,55% | 99,48% | 44,42% | 44,98% | 54,03% | 0,52% | 0,41 |
| 0,00005 | 5 | 8 | 98,99% | 45,04% | 99,49% | 43,79% | 44,40% | 54,03% | 0,51% | 0,43 |
| 0,00005 | 3 | 8 | 98,99% | 44,24% | 99,48% | 43,80% | 44,02% | 54,03% | 0,52% | 0,46 |
| 0,00150 | 3 | 14 | 98,95% | 46,31% | 99,47% | 45,63% | 45,97% | 54,03% | 0,53% | 0,43 |
| 0,00010 | 2 | None | 98,99% | 44,24% | 99,48% | 43,78% | 44,01% | 54,03% | 0,52% | 0,42 |
| 0,00010 | 5 | 28 | 98,96% | 46,07% | 99,47% | 44,17% | 45,10% | 54,03% | 0,53% | 0,42 |
| 0,00010 | 5 | 32 | 98,97% | 45,55% | 99,48% | 43,53% | 44,52% | 54,02% | 0,52% | 0,44 |
| 0,00020 | 6 | 12 | 98,96% | 46,31% | 99,47% | 44,54% | 45,41% | 54,02% | 0,53% | 0,43 |
| 0,00005 | 6 | 28 | 98,95% | 46,70% | 99,47% | 45,37% | 46,03% | 54,01% | 0,53% | 0,41 |
| 0,00150 | 5 | 4 | 98,96% | 44,30% | 99,47% | 43,18% | 43,73% | 54,01% | 0,53% | 0,43 |
| 0,00150 | 5 | 28 | 98,96% | 44,30% | 99,47% | 43,18% | 43,73% | 54,01% | 0,53% | 0,42 |
| 0,00150 | 6 | 8 | 98,94% | 45,19% | 99,46% | 42,87% | 44,00% | 54,01% | 0,54% | 0,41 |
| 0,00150 | 6 | 14 | 98,94% | 45,19% | 99,46% | 42,87% | 44,00% | 54,01% | 0,54% | 0,45 |
| 0,00150 | 6 | 28 | 98,94% | 45,19% | 99,46% | 42,87% | 44,00% | 54,01% | 0,54% | 0,41 |
| 0,00150 | 5 | 8 | 98,95% | 44,81% | 99,47% | 43,10% | 43,94% | 54,01% | 0,53% | 0,41 |
| 0,00005 | 4 | 8 | 98,97% | 45,12% | 99,47% | 44,70% | 44,91% | 53,61% | 0,53% | 0,43 |
| 0,00005 | 2 | 24 | 99,00% | 43,45% | 99,49% | 42,85% | 43,15% | 53,60% | 0,51% | 0,43 |
| 0,00020 | 5 | 32 | 98,99% | 43,56% | 99,49% | 41,89% | 42,71% | 53,60% | 0,51% | 0,42 |
| 0,00010 | 5 | 16 | 98,97% | 45,41% | 99,47% | 44,04% | 44,71% | 53,60% | 0,53% | 0,44 |
| 0,00005 | 5 | 10 | 98,97% | 45,41% | 99,47% | 44,04% | 44,71% | 53,60% | 0,53% | 0,48 |
| 0,00020 | 5 | 28 | 98,97% | 44,55% | 99,47% | 42,60% | 43,55% | 53,60% | 0,53% | 0,56 |
| 0,00005 | 5 | 18 | 98,98% | 43,91% | 99,48% | 42,10% | 42,99% | 53,59% | 0,52% | 0,44 |
| 0,00150 | 5 | 12 | 98,95% | 45,16% | 99,47% | 43,58% | 44,35% | 53,59% | 0,53% | 0,46 |
| 0,00150 | 4 | 20 | 98,94% | 45,15% | 99,46% | 44,31% | 44,73% | 53,59% | 0,54% | 0,44 |
| 0,00010 | 2 | 28 | 98,97% | 44,18% | 99,47% | 43,14% | 43,65% | 53,59% | 0,53% | 0,42 |
| 0,00005 | 5 | 12 | 98,97% | 44,41% | 99,47% | 42,39% | 43,38% | 53,59% | 0,53% | 0,43 |
| 0,00005 | 6 | None | 98,97% | 45,50% | 99,47% | 44,67% | 45,08% | 53,59% | 0,53% | 0,42 |
| 0,00150 | 5 | 10 | 98,93% | 46,22% | 99,46% | 43,99% | 45,08% | 53,59% | 0,54% | 0,40 |
| 0,00010 | 6 | 4 | 98,95% | 45,45% | 99,47% | 43,78% | 44,60% | 53,59% | 0,53% | 0,43 |
| 0,00005 | 6 | 32 | 98,95% | 46,02% | 99,47% | 44,80% | 45,40% | 53,59% | 0,53% | 0,52 |
| 0,00010 | 6 | 8 | 98,95% | 46,02% | 99,47% | 44,80% | 45,40% | 53,59% | 0,53% | 0,46 |
| 0,00010 | 6 | 10 | 98,95% | 46,02% | 99,47% | 44,80% | 45,40% | 53,59% | 0,53% | 0,48 |
| 0,00010 | 6 | None | 98,94% | 46,56% | 99,46% | 45,51% | 46,03% | 53,59% | 0,54% | 0,40 |
| 0,00005 | 6 | 24 | 98,97% | 44,51% | 99,47% | 42,96% | 43,72% | 53,58% | 0,53% | 0,42 |
| 0,00020 | 6 | 10 | 98,95% | 45,16% | 99,47% | 43,20% | 44,16% | 53,58% | 0,53% | 0,42 |
| 0,00020 | 6 | 28 | 98,95% | 45,16% | 99,47% | 43,39% | 44,26% | 53,58% | 0,53% | 0,53 |
| 0,00020 | 6 | 32 | 98,95% | 45,16% | 99,47% | 43,20% | 44,16% | 53,58% | 0,53% | 0,51 |
| 0,00005 | 6 | 10 | 98,97% | 44,51% | 99,47% | 43,02% | 43,75% | 53,58% | 0,53% | 0,43 |
| 0,00150 | 6 | None | 98,94% | 44,52% | 99,46% | 43,49% | 44,00% | 53,58% | 0,54% | 0,42 |
| 0,00005 | 6 | 18 | 98,94% | 45,54% | 99,46% | 43,63% | 44,57% | 53,58% | 0,54% | 0,46 |
| 0,00150 | 6 | 12 | 98,93% | 45,04% | 99,46% | 43,41% | 44,21% | 53,58% | 0,54% | 0,40 |
| 0,00150 | 6 | 18 | 98,93% | 45,04% | 99,46% | 43,41% | 44,21% | 53,58% | 0,54% | 0,41 |
| 0,00150 | 5 | None | 98,94% | 43,65% | 99,46% | 41,82% | 42,72% | 53,57% | 0,54% | 0,43 |
| 0,00150 | 5 | 16 | 98,94% | 43,65% | 99,46% | 41,82% | 42,72% | 53,57% | 0,54% | 0,41 |
| 0,00150 | 5 | 20 | 98,94% | 43,65% | 99,46% | 41,82% | 42,72% | 53,57% | 0,54% | 0,41 |
| 0,00200 | 3 | 8 | 98,90% | 44,62% | 99,44% | 44,02% | 44,32% | 53,53% | 0,56% | 0,47 |
| 0,00200 | 4 | 8 | 98,90% | 44,62% | 99,44% | 44,02% | 44,32% | 53,53% | 0,56% | 0,47 |
| 0,00200 | 2 | 24 | 98,90% | 44,62% | 99,44% | 44,02% | 44,32% | 53,53% | 0,56% | 0,41 |
| 0,00200 | 4 | 6 | 98,89% | 45,17% | 99,44% | 43,94% | 44,55% | 53,53% | 0,56% | 0,44 |
| 0,00200 | 3 | 14 | 98,89% | 45,17% | 99,44% | 43,94% | 44,55% | 53,53% | 0,56% | 0,40 |
| 0,00200 | 4 | 16 | 98,89% | 45,17% | 99,44% | 43,94% | 44,55% | 53,53% | 0,56% | 0,40 |
| 0,00200 | 3 | 18 | 98,89% | 45,17% | 99,44% | 43,94% | 44,55% | 53,53% | 0,56% | 0,43 |
| 0,00200 | 2 | 10 | 98,89% | 45,17% | 99,44% | 44,55% | 44,86% | 53,53% | 0,56% | 0,43 |
| 0,00200 | 3 | 10 | 98,89% | 45,17% | 99,44% | 44,55% | 44,86% | 53,53% | 0,56% | 0,47 |
| 0,00200 | 3 | 28 | 98,89% | 45,17% | 99,44% | 44,55% | 44,86% | 53,53% | 0,56% | 0,41 |
| 0,00200 | 3 | 16 | 98,88% | 45,73% | 99,43% | 44,49% | 45,10% | 53,53% | 0,57% | 0,40 |
| 0,00005 | 5 | 24 | 98,96% | 44,27% | 99,47% | 42,47% | 43,35% | 53,17% | 0,53% | 0,42 |
| 0,00010 | 5 | 8 | 98,97% | 43,77% | 99,48% | 42,03% | 42,88% | 53,16% | 0,52% | 0,42 |
| 0,00005 | 5 | 20 | 98,96% | 44,27% | 99,47% | 42,34% | 43,28% | 53,16% | 0,53% | 0,42 |
| 0,00005 | 6 | 12 | 98,94% | 44,87% | 99,46% | 43,59% | 44,22% | 53,16% | 0,54% | 0,42 |
| 0,00005 | 6 | 8 | 98,94% | 44,87% | 99,46% | 43,26% | 44,05% | 53,15% | 0,54% | 0,45 |
| 0,00010 | 6 | 20 | 98,93% | 45,40% | 99,46% | 43,57% | 44,47% | 53,15% | 0,54% | 0,42 |
| 0,00150 | 6 | 4 | 98,93% | 43,36% | 99,46% | 42,21% | 42,78% | 53,14% | 0,54% | 0,41 |
| 0,00150 | 6 | 16 | 98,91% | 44,39% | 99,44% | 42,60% | 43,48% | 53,14% | 0,56% | 0,41 |
| 0,00150 | 6 | 20 | 98,91% | 44,39% | 99,44% | 42,60% | 43,48% | 53,14% | 0,56% | 0,42 |
| 0,00150 | 6 | 24 | 98,91% | 44,39% | 99,44% | 42,60% | 43,48% | 53,14% | 0,56% | 0,41 |
| 0,00200 | 6 | 4 | 98,89% | 44,42% | 99,44% | 43,11% | 43,75% | 53,10% | 0,56% | 0,42 |
| 0,00200 | 6 | 6 | 98,89% | 44,42% | 99,44% | 43,11% | 43,75% | 53,10% | 0,56% | 0,45 |
| 0,00200 | 6 | 16 | 98,89% | 44,42% | 99,44% | 43,11% | 43,75% | 53,10% | 0,56% | 0,41 |
| 0,00200 | 6 | None | 98,88% | 44,97% | 99,43% | 43,63% | 44,29% | 53,10% | 0,57% | 0,47 |
| 0,00200 | 5 | 18 | 98,88% | 44,97% | 99,43% | 43,63% | 44,29% | 53,10% | 0,57% | 0,40 |
| 0,00200 | 6 | 14 | 98,87% | 45,52% | 99,42% | 43,55% | 44,51% | 53,10% | 0,58% | 0,40 |
| 0,00200 | 4 | None | 98,89% | 43,42% | 99,44% | 43,37% | 43,40% | 53,10% | 0,56% | 0,42 |
| 0,00200 | 2 | 12 | 98,89% | 43,42% | 99,44% | 43,37% | 43,40% | 53,10% | 0,56% | 0,46 |
| 0,00200 | 4 | 20 | 98,89% | 43,42% | 99,44% | 43,37% | 43,40% | 53,10% | 0,56% | 0,44 |
| 0,00200 | 2 | 32 | 98,89% | 43,42% | 99,44% | 43,37% | 43,40% | 53,10% | 0,56% | 0,42 |
| 0,00200 | 4 | 4 | 98,88% | 43,95% | 99,43% | 43,29% | 43,62% | 53,10% | 0,57% | 0,41 |
| 0,00200 | 3 | 6 | 98,88% | 43,95% | 99,43% | 43,29% | 43,62% | 53,10% | 0,57% | 0,40 |
| 0,00200 | 2 | 16 | 98,88% | 43,95% | 99,43% | 43,29% | 43,62% | 53,10% | 0,57% | 0,44 |

Figure C.9: Continued – Performance metric scores of all unique DT parameter settings (based on 5-fold cross-validation).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0,00200 | 3 | 24 | 98,88% | 43,95% | 99,43% | 43,29% | 43,62% | 53,10% | 0,57% | 0,42 |
| 0,00200 | 4 | 28 | 98,87% | 44,49% | 99,42% | 43,83% | 44,16% | 53,09% | 0,58% | 0,39 |
| 0,00200 | 2 | None | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,45 |
| 0,00200 | 3 | None | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,44 |
| 0,00200 | 2 | 6 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,40 |
| 0,00200 | 2 | 8 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,42 |
| 0,00200 | 3 | 12 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,47 |
| 0,00200 | 4 | 12 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,47 |
| 0,00200 | 3 | 20 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,40 |
| 0,00200 | 4 | 32 | 98,89% | 43,57% | 99,44% | 42,62% | 43,09% | 53,08% | 0,56% | 0,44 |
| 0,00200 | 4 | 18 | 98,88% | 44,10% | 99,43% | 42,53% | 43,30% | 53,08% | 0,57% | 0,44 |
| 0,00200 | 2 | 20 | 98,88% | 44,10% | 99,43% | 42,53% | 43,30% | 53,08% | 0,57% | 0,39 |
| 0,00005 | 5 | None | 98,95% | 45,12% | 99,46% | 44,11% | 44,61% | 52,75% | 0,54% | 0,46 |
| 0,00005 | 5 | 32 | 98,93% | 45,64% | 99,46% | 44,62% | 45,12% | 52,75% | 0,54% | 0,44 |
| 0,00010 | 5 | 18 | 98,95% | 45,12% | 99,46% | 43,98% | 44,54% | 52,74% | 0,54% | 0,44 |
| 0,00010 | 5 | 24 | 98,95% | 45,12% | 99,46% | 43,79% | 44,44% | 52,74% | 0,54% | 0,46 |
| 0,00005 | 5 | 4 | 98,93% | 45,64% | 99,46% | 44,48% | 45,05% | 52,74% | 0,54% | 0,43 |
| 0,00005 | 6 | 14 | 98,95% | 45,22% | 99,46% | 44,55% | 44,88% | 52,74% | 0,54% | 0,45 |
| 0,00005 | 6 | 16 | 98,95% | 45,22% | 99,46% | 44,55% | 44,88% | 52,74% | 0,54% | 0,49 |
| 0,00010 | 6 | 16 | 98,92% | 46,27% | 99,45% | 45,23% | 45,74% | 52,73% | 0,55% | 0,42 |
| 0,00200 | 5 | 20 | 98,88% | 43,22% | 99,43% | 42,46% | 42,84% | 52,67% | 0,57% | 0,39 |
| 0,00200 | 6 | 28 | 98,88% | 43,22% | 99,43% | 42,46% | 42,84% | 52,67% | 0,57% | 0,40 |
| 0,00200 | 5 | 12 | 98,87% | 43,75% | 99,42% | 42,37% | 43,05% | 52,67% | 0,58% | 0,43 |
| 0,00200 | 6 | 32 | 98,87% | 43,75% | 99,42% | 42,37% | 43,05% | 52,67% | 0,58% | 0,44 |
| 0,00200 | 5 | 10 | 98,87% | 43,75% | 99,42% | 42,98% | 43,36% | 52,67% | 0,58% | 0,57 |
| 0,00200 | 6 | 12 | 98,87% | 43,75% | 99,42% | 42,98% | 43,36% | 52,67% | 0,58% | 0,41 |
| 0,00200 | 5 | 14 | 98,85% | 44,29% | 99,42% | 42,89% | 43,58% | 52,66% | 0,58% | 0,39 |
| 0,00200 | 5 | 32 | 98,85% | 44,29% | 99,42% | 42,89% | 43,58% | 52,66% | 0,58% | 0,41 |
| 0,00200 | 5 | None | 98,88% | 43,37% | 99,43% | 41,71% | 42,52% | 52,65% | 0,57% | 0,50 |
| 0,00200 | 5 | 4 | 98,88% | 43,37% | 99,43% | 41,71% | 42,52% | 52,65% | 0,57% | 0,44 |
| 0,00200 | 5 | 8 | 98,88% | 43,37% | 99,43% | 41,71% | 42,52% | 52,65% | 0,57% | 0,46 |
| 0,00200 | 6 | 10 | 98,87% | 43,90% | 99,42% | 41,61% | 42,72% | 52,65% | 0,58% | 0,46 |
| 0,00200 | 6 | 20 | 98,87% | 43,90% | 99,42% | 41,61% | 42,72% | 52,65% | 0,58% | 0,40 |
| 0,00200 | 5 | 16 | 98,85% | 44,44% | 99,42% | 42,12% | 43,25% | 52,65% | 0,58% | 0,41 |
| 0,00200 | 3 | 4 | 98,88% | 42,37% | 99,43% | 41,98% | 42,17% | 52,65% | 0,57% | 0,40 |
| 0,00200 | 4 | 10 | 98,88% | 42,37% | 99,43% | 41,98% | 42,17% | 52,65% | 0,57% | 0,45 |
| 0,00200 | 4 | 24 | 98,88% | 42,37% | 99,43% | 41,98% | 42,17% | 52,65% | 0,57% | 0,39 |
| 0,00200 | 2 | 28 | 98,88% | 42,37% | 99,43% | 41,98% | 42,17% | 52,65% | 0,57% | 0,41 |
| 0,00200 | 2 | 4 | 98,87% | 42,88% | 99,42% | 41,88% | 42,37% | 52,65% | 0,58% | 0,44 |
| 0,00200 | 4 | 14 | 98,87% | 42,88% | 99,42% | 41,88% | 42,37% | 52,65% | 0,58% | 0,42 |
| 0,00200 | 3 | 32 | 98,87% | 42,88% | 99,42% | 41,88% | 42,37% | 52,65% | 0,58% | 0,41 |
| 0,00200 | 2 | 14 | 98,87% | 42,88% | 99,42% | 42,49% | 42,68% | 52,65% | 0,58% | 0,40 |
| 0,00200 | 2 | 18 | 98,85% | 43,41% | 99,42% | 42,40% | 42,90% | 52,64% | 0,58% | 0,43 |
| 0,00010 | 5 | 10 | 98,96% | 43,01% | 99,47% | 41,70% | 42,34% | 52,31% | 0,53% | 0,47 |
| 0,00010 | 5 | 20 | 98,95% | 43,49% | 99,47% | 41,93% | 42,70% | 52,31% | 0,53% | 0,45 |
| 0,00010 | 5 | 12 | 98,94% | 43,99% | 99,46% | 42,22% | 43,08% | 52,31% | 0,54% | 0,42 |
| 0,00010 | 6 | 14 | 98,92% | 44,59% | 99,45% | 43,39% | 43,98% | 52,30% | 0,55% | 0,46 |
| 0,00010 | 6 | 18 | 98,92% | 44,59% | 99,45% | 43,20% | 43,88% | 52,30% | 0,55% | 0,41 |
| 0,00200 | 6 | 18 | 98,87% | 42,16% | 99,43% | 41,07% | 41,61% | 52,22% | 0,57% | 0,42 |
| 0,00200 | 6 | 8 | 98,86% | 42,68% | 99,42% | 40,96% | 41,80% | 52,22% | 0,58% | 0,47 |
| 0,00200 | 5 | 24 | 98,86% | 42,68% | 99,42% | 40,96% | 41,80% | 52,22% | 0,58% | 0,43 |
| 0,00200 | 6 | 24 | 98,86% | 42,68% | 99,42% | 40,96% | 41,80% | 52,22% | 0,58% | 0,39 |
| 0,00200 | 5 | 28 | 98,86% | 42,68% | 99,42% | 40,96% | 41,80% | 52,22% | 0,58% | 0,40 |
| 0,00200 | 5 | 6 | 98,84% | 43,21% | 99,41% | 41,46% | 42,32% | 52,21% | 0,59% | 0,46 |
| 0,00500 | 2 | None | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 3 | None | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 4 | None | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,36 |
| 0,00500 | 5 | None | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,36 |
| 0,00500 | 6 | None | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 2 | 4 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 3 | 4 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 4 | 4 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,42 |
| 0,00500 | 5 | 4 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,45 |
| 0,00500 | 6 | 4 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 2 | 6 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,49 |
| 0,00500 | 3 | 6 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 4 | 6 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 5 | 6 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 6 | 6 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,43 |
| 0,00500 | 2 | 8 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 3 | 8 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 4 | 8 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 5 | 8 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 6 | 8 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 2 | 10 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 3 | 10 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 4 | 10 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 5 | 10 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,42 |
| 0,00500 | 6 | 10 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 2 | 12 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 3 | 12 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 4 | 12 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 5 | 12 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 6 | 12 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,36 |
| 0,00500 | 2 | 14 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 3 | 14 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,37 |
| 0,00500 | 4 | 14 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 5 | 14 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 6 | 14 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 2 | 16 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,37 |
| 0,00500 | 3 | 16 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,39 |
| 0,00500 | 4 | 16 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,42 |
| 0,00500 | 5 | 16 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 6 | 16 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 2 | 18 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 3 | 18 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,40 |
| 0,00500 | 4 | 18 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,37 |
| 0,00500 | 5 | 18 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,38 |
| 0,00500 | 6 | 18 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |
| 0,00500 | 2 | 20 | 98,73% | 32,34% | 99,35% | 23,78% | 27,41% | 46,87% | 0,65% | 0,41 |

Figure C.10: Continued – Performance metric scores of all unique DT parameter settings (based on 5-fold cross-validation).

### C.1.3 Random Forest

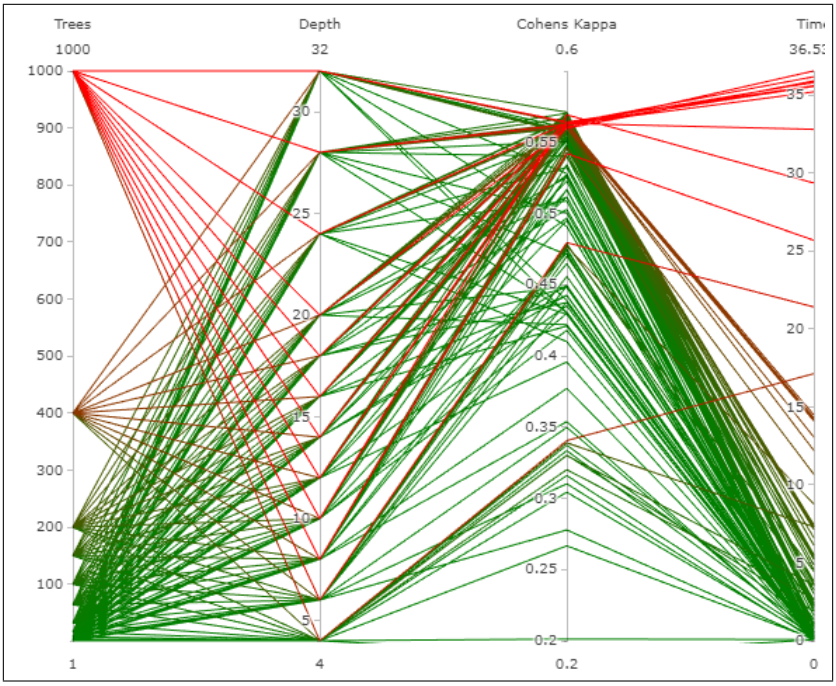



Figure C.11: Parallel Coordinates Plot of the Random Forest hyper-parameter tuning experiment.

.

| Trees | Depth | Accuracy | Recall | Specificity | Precision | F1 measure | Cohens kappa | False Positive Rate | Computation Time |
|---|---|---|---|---|---|---|---|---|---|
| 16 | 12 | 99,10% | 54,21% | 99,54% | 50,36% | 52,21% | 61,32% | 0,46% | 0,49 |
| 150 | 10 | 99,09% | 52,52% | 99,54% | 50,35% | 51,41% | 60,46% | 0,46% | 4,05 |
| 100 | 10 | 99,06% | 54,34% | 99,52% | 51,37% | 52,81% | 60,04% | 0,48% | 2,72 |
| 150 | 20 | 99,07% | 53,32% | 99,53% | 49,27% | 51,21% | 60,04% | 0,47% | 4,36 |
| 150 | 12 | 99,08% | 53,38% | 99,53% | 50,33% | 51,81% | 60,03% | 0,47% | 4,23 |
| 32 | 12 | 99,10% | 51,66% | 99,54% | 50,89% | 51,27% | 59,62% | 0,46% | 0,91 |
| 64 | 28 | 99,10% | 50,86% | 99,54% | 49,13% | 49,98% | 59,61% | 0,46% | 1,78 |
| 200 | 28 | 99,09% | 52,21% | 99,54% | 49,90% | 51,03% | 59,61% | 0,46% | 5,83 |
| 400 | 12 | 99,07% | 52,65% | 99,53% | 49,20% | 50,87% | 59,60% | 0,47% | 11,28 |
| 32 | 18 | 99,11% | 48,18% | 99,55% | 47,09% | 47,63% | 59,60% | 0,45% | 0,91 |
| 100 | 28 | 99,08% | 50,53% | 99,53% | 49,17% | 49,84% | 59,59% | 0,47% | 2,88 |
| 4 | 18 | 99,11% | 46,98% | 99,55% | 45,36% | 46,15% | 59,57% | 0,45% | 0,15 |
| 16 | 20 | 99,06% | 51,92% | 99,52% | 50,34% | 51,12% | 59,22% | 0,48% | 0,49 |
| 1000 | 32 | 99,05% | 53,75% | 99,52% | 51,85% | 52,78% | 59,20% | 0,48% | 29,69 |
| 32 | 10 | 99,06% | 51,10% | 99,52% | 50,67% | 50,88% | 59,19% | 0,48% | 0,87 |
| 16 | 32 | 99,10% | 48,94% | 99,54% | 48,30% | 48,62% | 59,18% | 0,46% | 0,48 |
| 1000 | 20 | 99,07% | 52,23% | 99,53% | 49,10% | 50,62% | 59,17% | 0,47% | 29,24 |
| 400 | 28 | 99,06% | 51,03% | 99,52% | 48,80% | 49,89% | 59,16% | 0,48% | 11,92 |
| 200 | 32 | 99,08% | 51,08% | 99,53% | 49,59% | 50,32% | 58,77% | 0,47% | 5,80 |
| 400 | 18 | 99,04% | 54,15% | 99,51% | 52,35% | 53,23% | 58,77% | 0,49% | 11,84 |
| 64 | 16 | 99,07% | 50,36% | 99,53% | 50,67% | 50,52% | 58,76% | 0,47% | 1,81 |
| 8 | 14 | 99,08% | 47,45% | 99,53% | 45,73% | 46,58% | 58,75% | 0,47% | 0,26 |
| 32 | 16 | 99,09% | 46,37% | 99,54% | 44,72% | 45,53% | 58,74% | 0,46% | 0,92 |
| 400 | 14 | 99,07% | 50,33% | 99,53% | 47,11% | 48,67% | 58,74% | 0,47% | 11,61 |
| 64 | 18 | 99,08% | 49,47% | 99,53% | 47,66% | 48,55% | 58,73% | 0,47% | 1,81 |
| 1000 | 14 | 99,07% | 49,62% | 99,53% | 46,99% | 48,27% | 58,73% | 0,47% | 29,52 |
| 1000 | 10 | 99,06% | 50,56% | 99,52% | 48,43% | 49,47% | 58,73% | 0,48% | 27,31 |
| 150 | 16 | 99,04% | 53,12% | 99,51% | 50,47% | 51,76% | 58,34% | 0,49% | 4,33 |
| 400 | 16 | 99,07% | 50,13% | 99,53% | 47,99% | 49,04% | 58,34% | 0,47% | 11,70 |
| 200 | 20 | 99,05% | 51,68% | 99,52% | 50,57% | 51,12% | 58,34% | 0,48% | 5,80 |
| 400 | 24 | 99,03% | 54,01% | 99,51% | 52,08% | 53,03% | 58,34% | 0,49% | 11,65 |
| 1000 | 18 | 99,04% | 53,27% | 99,51% | 50,61% | 51,91% | 58,33% | 0,49% | 29,50 |
| 32 | 14 | 99,06% | 48,93% | 99,52% | 47,76% | 48,34% | 58,33% | 0,48% | 0,94 |
| 32 | 32 | 99,06% | 51,19% | 99,52% | 50,92% | 51,05% | 58,33% | 0,48% | 0,94 |
| 200 | 10 | 99,04% | 51,30% | 99,51% | 48,19% | 49,70% | 58,33% | 0,49% | 5,34 |
| 400 | 10 | 99,05% | 50,61% | 99,52% | 48,69% | 49,63% | 58,32% | 0,48% | 10,82 |
| 64 | 12 | 99,03% | 52,16% | 99,50% | 49,35% | 50,72% | 58,32% | 0,50% | 1,78 |
| 100 | 16 | 99,05% | 49,56% | 99,52% | 47,70% | 48,61% | 58,31% | 0,48% | 2,90 |
| 16 | 24 | 99,03% | 50,32% | 99,51% | 45,65% | 47,87% | 58,31% | 0,49% | 0,49 |
| 100 | 20 | 99,05% | 51,50% | 99,52% | 48,24% | 49,82% | 58,31% | 0,48% | 2,90 |
| 1000 | 12 | 99,03% | 52,04% | 99,51% | 50,45% | 51,23% | 57,90% | 0,49% | 28,89 |
| 16 | 18 | 99,06% | 47,91% | 99,52% | 47,00% | 47,45% | 57,89% | 0,48% | 0,50 |
| 100 | 32 | 99,07% | 48,63% | 99,53% | 46,34% | 47,46% | 57,88% | 0,47% | 2,93 |
| 150 | 24 | 99,06% | 48,24% | 99,52% | 45,85% | 47,02% | 57,87% | 0,48% | 4,34 |
| 200 | 12 | 99,06% | 48,24% | 99,52% | 46,20% | 47,20% | 57,87% | 0,48% | 5,71 |
| 100 | 14 | 99,05% | 48,61% | 99,52% | 46,69% | 47,63% | 57,87% | 0,48% | 2,86 |
| 16 | 16 | 99,05% | 48,36% | 99,52% | 45,28% | 46,77% | 57,86% | 0,48% | 0,51 |
| 100 | 18 | 99,05% | 47,99% | 99,52% | 45,54% | 46,73% | 57,86% | 0,48% | 2,94 |
| 4 | 10 | 99,03% | 46,78% | 99,51% | 44,22% | 45,46% | 57,84% | 0,49% | 0,14 |
| 200 | 18 | 99,03% | 51,20% | 99,51% | 48,14% | 49,62% | 57,48% | 0,49% | 5,78 |
| 1000 | 16 | 99,04% | 50,61% | 99,51% | 48,99% | 49,79% | 57,47% | 0,49% | 29,32 |
| 1000 | 28 | 99,03% | 51,20% | 99,51% | 48,07% | 49,58% | 57,47% | 0,49% | 33,44 |
| 150 | 28 | 99,04% | 48,22% | 99,51% | 46,96% | 47,58% | 57,45% | 0,49% | 4,38 |
| 8 | 12 | 99,03% | 47,76% | 99,51% | 46,03% | 46,88% | 57,44% | 0,49% | 0,25 |
| 64 | 32 | 99,05% | 48,54% | 99,52% | 46,66% | 47,58% | 57,04% | 0,48% | 1,81 |
| 1000 | 24 | 99,03% | 50,23% | 99,51% | 47,56% | 48,86% | 57,04% | 0,49% | 29,47 |
| 100 | 24 | 99,03% | 49,79% | 99,51% | 47,88% | 48,82% | 57,04% | 0,49% | 2,89 |
| 16 | 28 | 99,02% | 50,58% | 99,50% | 50,30% | 50,44% | 57,04% | 0,50% | 0,50 |
| 100 | 12 | 99,02% | 49,30% | 99,50% | 47,33% | 48,29% | 57,04% | 0,50% | 2,75 |
| 400 | 20 | 99,01% | 51,84% | 99,50% | 48,31% | 50,01% | 57,03% | 0,50% | 11,86 |
| 8 | 18 | 99,04% | 44,69% | 99,51% | 44,00% | 44,34% | 57,01% | 0,49% | 0,27 |
| 64 | 24 | 99,03% | 48,19% | 99,51% | 46,16% | 47,15% | 57,00% | 0,49% | 1,80 |
| 4 | 24 | 99,04% | 45,22% | 99,51% | 42,97% | 44,07% | 57,00% | 0,49% | 0,16 |
| 8 | 28 | 99,04% | 45,56% | 99,51% | 44,64% | 45,09% | 57,00% | 0,49% | 0,28 |
| 400 | 32 | 99,03% | 47,80% | 99,51% | 44,87% | 46,29% | 57,00% | 0,49% | 11,52 |
| 100 | 8 | 99,00% | 44,89% | 99,49% | 44,89% | 44,89% | 56,95% | 0,51% | 2,45 |
| 150 | 14 | 99,01% | 49,55% | 99,50% | 46,79% | 48,13% | 56,62% | 0,50% | 4,31 |
| 32 | 28 | 99,03% | 48,24% | 99,51% | 46,25% | 47,22% | 56,60% | 0,49% | 0,93 |
| 150 | 32 | 99,02% | 49,56% | 99,50% | 46,56% | 48,01% | 56,60% | 0,50% | 4,35 |
| 200 | 16 | 99,02% | 49,56% | 99,50% | 46,01% | 47,72% | 56,60% | 0,50% | 5,90 |
| 150 | 18 | 99,00% | 51,14% | 99,49% | 47,94% | 49,49% | 56,60% | 0,51% | 4,37 |
| 16 | 14 | 99,00% | 48,62% | 99,49% | 48,82% | 48,72% | 56,60% | 0,51% | 0,49 |
| 8 | 24 | 99,03% | 46,17% | 99,51% | 46,38% | 46,27% | 56,58% | 0,49% | 0,26 |
| 8 | 20 | 99,05% | 46,02% | 99,52% | 42,88% | 44,40% | 56,58% | 0,48% | 0,26 |
| 32 | 24 | 99,03% | 45,46% | 99,51% | 44,36% | 44,90% | 56,58% | 0,49% | 0,92 |
| 64 | 10 | 99,03% | 46,53% | 99,51% | 43,73% | 45,09% | 56,57% | 0,49% | 1,73 |
| 32 | 20 | 99,01% | 47,87% | 99,50% | 44,01% | 45,86% | 56,56% | 0,50% | 0,91 |

Figure C.12: Performance metric scores of all unique RF parameter settings (based on 5-fold cross-validation).

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 4 | 14 | 99,02% | 45,67% | 99,50% | 43,59% | 44,61% | 56,56% | 0,50% | 0,15 |
| 4 | 20 | 99,03% | 46,52% | 99,51% | 42,72% | 44,54% | 56,55% | 0,49% | 0,15 |
| 16 | 10 | 99,01% | 48,69% | 99,50% | 47,85% | 48,26% | 56,18% | 0,50% | 0,45 |
| 200 | 24 | 99,02% | 49,66% | 99,50% | 46,68% | 48,13% | 56,18% | 0,50% | 5,85 |
| 8 | 10 | 99,01% | 45,76% | 99,50% | 43,43% | 44,56% | 56,15% | 0,50% | 0,25 |
| 8 | 32 | 99,01% | 47,10% | 99,50% | 46,52% | 46,81% | 56,12% | 0,50% | 0,27 |
| 32 | 8 | 98,96% | 45,97% | 99,47% | 43,16% | 44,52% | 56,09% | 0,53% | 0,83 |
| 200 | 14 | 99,02% | 46,67% | 99,50% | 44,20% | 45,40% | 55,76% | 0,50% | 5,77 |
| 64 | 20 | 99,02% | 48,07% | 99,50% | 45,50% | 46,75% | 55,75% | 0,50% | 1,85 |
| 64 | 14 | 99,01% | 45,62% | 99,50% | 45,27% | 45,45% | 55,73% | 0,50% | 1,82 |
| 4 | 16 | 99,02% | 46,51% | 99,50% | 45,41% | 45,96% | 55,73% | 0,50% | 0,16 |
| 8 | 16 | 99,01% | 48,60% | 99,50% | 50,06% | 49,32% | 55,71% | 0,50% | 0,26 |
| 4 | 32 | 98,97% | 44,95% | 99,48% | 43,01% | 43,96% | 55,26% | 0,52% | 0,15 |
| 64 | 8 | 98,98% | 42,53% | 99,48% | 41,46% | 41,99% | 55,22% | 0,52% | 1,57 |
| 8 | 8 | 99,01% | 41,99% | 99,50% | 41,02% | 41,50% | 54,87% | 0,50% | 0,22 |
| 150 | 8 | 98,95% | 42,61% | 99,46% | 39,49% | 40,99% | 54,79% | 0,54% | 3,59 |
| 400 | 8 | 98,93% | 44,77% | 99,46% | 41,63% | 43,14% | 54,79% | 0,54% | 9,62 |
| 64 | 6 | 98,95% | 37,32% | 99,46% | 31,09% | 33,92% | 54,69% | 0,54% | 1,39 |
| 2 | 28 | 98,97% | 43,13% | 99,48% | 41,95% | 42,54% | 54,38% | 0,52% | 0,10 |
| 4 | 28 | 98,95% | 46,47% | 99,47% | 44,48% | 45,45% | 54,38% | 0,53% | 0,15 |
| 4 | 12 | 99,01% | 41,61% | 99,49% | 39,44% | 40,50% | 54,37% | 0,51% | 0,15 |
| 200 | 8 | 98,95% | 41,07% | 99,46% | 40,35% | 40,71% | 54,35% | 0,54% | 4,71 |
| 1000 | 8 | 98,94% | 41,96% | 99,46% | 40,69% | 41,32% | 54,35% | 0,54% | 24,09 |
| 2 | 12 | 99,01% | 43,22% | 99,50% | 41,94% | 42,57% | 54,01% | 0,50% | 0,09 |
| 4 | 8 | 98,95% | 39,52% | 99,47% | 36,67% | 38,05% | 53,89% | 0,53% | 0,13 |
| 2 | 16 | 99,00% | 42,97% | 99,49% | 41,84% | 42,40% | 53,55% | 0,51% | 0,10 |
| 16 | 8 | 98,92% | 40,24% | 99,45% | 36,59% | 38,33% | 53,45% | 0,55% | 0,42 |
| 150 | 6 | 98,91% | 36,04% | 99,44% | 30,74% | 33,18% | 52,95% | 0,56% | 3,13 |
| 400 | 6 | 98,89% | 36,00% | 99,43% | 29,31% | 32,31% | 52,94% | 0,57% | 8,27 |
| 1000 | 6 | 98,87% | 35,70% | 99,42% | 30,26% | 32,76% | 52,07% | 0,58% | 20,85 |
| 200 | 6 | 98,85% | 36,65% | 99,41% | 31,68% | 33,98% | 51,64% | 0,59% | 4,10 |
| 1 | 32 | 98,95% | 43,89% | 99,47% | 42,54% | 43,21% | 51,46% | 0,53% | 0,08 |
| 2 | 24 | 98,92% | 41,47% | 99,45% | 39,55% | 40,49% | 51,36% | 0,55% | 0,10 |
| 32 | 6 | 98,84% | 36,25% | 99,41% | 30,30% | 33,01% | 51,21% | 0,59% | 0,70 |
| 2 | 14 | 98,88% | 45,44% | 99,43% | 43,80% | 44,60% | 50,99% | 0,57% | 0,11 |
| 2 | 8 | 98,87% | 39,28% | 99,42% | 35,12% | 37,08% | 50,90% | 0,58% | 0,09 |
| 1 | 28 | 98,88% | 44,31% | 99,43% | 41,94% | 43,09% | 50,15% | 0,57% | 0,08 |
| 2 | 32 | 98,88% | 41,62% | 99,43% | 40,65% | 41,13% | 50,11% | 0,57% | 0,10 |
| 2 | 10 | 98,86% | 40,82% | 99,42% | 36,40% | 38,49% | 50,06% | 0,58% | 0,09 |
| 100 | 6 | 98,83% | 32,06% | 99,41% | 25,88% | 28,64% | 49,90% | 0,59% | 2,11 |
| 1 | 20 | 98,93% | 37,67% | 99,46% | 35,09% | 36,33% | 49,69% | 0,54% | 0,07 |
| 2 | 18 | 98,92% | 37,44% | 99,45% | 36,61% | 37,02% | 49,28% | 0,55% | 0,11 |
| 1 | 12 | 98,84% | 41,01% | 99,41% | 40,08% | 40,54% | 48,41% | 0,59% | 0,09 |
| 1 | 16 | 98,89% | 40,01% | 99,43% | 36,95% | 38,42% | 48,39% | 0,57% | 0,08 |
| 1 | 14 | 98,85% | 36,48% | 99,42% | 36,07% | 36,28% | 48,37% | 0,58% | 0,09 |
| 16 | 6 | 98,81% | 35,29% | 99,39% | 31,45% | 33,26% | 48,25% | 0,61% | 0,37 |
| 1 | 18 | 98,91% | 32,77% | 99,45% | 32,83% | 32,80% | 47,91% | 0,55% | 0,07 |
| 2 | 20 | 98,82% | 37,54% | 99,40% | 33,17% | 35,22% | 45,79% | 0,60% | 0,10 |
| 1 | 24 | 98,78% | 36,59% | 99,38% | 35,90% | 36,24% | 45,36% | 0,62% | 0,07 |
| 1 | 10 | 98,68% | 34,67% | 99,33% | 34,84% | 34,75% | 44,97% | 0,67% | 0,09 |
| 4 | 6 | 98,70% | 30,37% | 99,34% | 24,88% | 27,35% | 44,28% | 0,66% | 0,13 |
| 8 | 6 | 98,68% | 29,35% | 99,33% | 23,25% | 25,95% | 43,42% | 0,67% | 0,20 |
| 200 | 4 | 98,61% | 26,14% | 99,29% | 17,20% | 20,75% | 42,79% | 0,71% | 3,47 |
| 400 | 4 | 98,57% | 26,01% | 99,27% | 15,59% | 19,49% | 41,85% | 0,73% | 6,93 |
| 100 | 4 | 98,55% | 24,49% | 99,26% | 13,01% | 16,99% | 41,03% | 0,74% | 1,79 |
| 1000 | 4 | 98,53% | 23,73% | 99,25% | 13,66% | 17,34% | 40,51% | 0,75% | 17,55 |
| 16 | 4 | 98,52% | 23,43% | 99,24% | 16,22% | 19,17% | 40,08% | 0,76% | 0,33 |
| 64 | 4 | 98,52% | 22,74% | 99,24% | 14,57% | 17,76% | 40,07% | 0,76% | 1,15 |
| 2 | 6 | 98,65% | 23,46% | 99,31% | 17,82% | 20,26% | 39,92% | 0,69% | 0,08 |
| 32 | 4 | 98,49% | 23,47% | 99,23% | 13,00% | 16,73% | 38,84% | 0,77% | 0,61 |
| 150 | 4 | 98,50% | 20,85% | 99,23% | 11,14% | 14,52% | 38,30% | 0,77% | 2,74 |
| 1 | 8 | 98,52% | 23,51% | 99,25% | 22,05% | 22,76% | 36,62% | 0,75% | 0,08 |
| 1 | 6 | 98,45% | 21,91% | 99,21% | 16,00% | 18,49% | 34,08% | 0,79% | 0,06 |
| 8 | 4 | 98,36% | 17,40% | 99,16% | 11,18% | 13,62% | 32,59% | 0,84% | 0,18 |
| 4 | 4 | 98,28% | 19,71% | 99,12% | 8,76% | 12,13% | 29,50% | 0,88% | 0,12 |
| 2 | 4 | 98,12% | 15,45% | 99,04% | 8,37% | 10,85% | 23,09% | 0,96% | 0,08 |
| 1 | 4 | 98,02% | 10,05% | 98,98% | 7,35% | 8,49% | 19,37% | 1,02% | 0,06 |

Figure C.13: Continued – Performance metric scores of all unique RF parameter settings (based on 5-fold cross-validation).

## C.2 Performance metrics

| Model | Accuracy | Recall | Specificity | Precision | F1 measure | Cohens kappa | False Positive Rate | Computation Time |
|---|---|---|---|---|---|---|---|---|
| Decision Tree | 99,22% | 52,53% | 99,61% | 52,73% | 52,63% | 63,58% | 0,39% | 0,52 |
| Random Forest | 99,23% | 54,49% | 99,61% | 54,30% | 50,73% | 64,03% | 0,39% | 11,94 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,86 |
| Neural Network | 47,23% | 47,23% | 99,41% | 48,72% | 43,85% | 46,24% | 0,59% | 18,05 |
| Decision Tree | 99,25% | 52,28% | 99,62% | 53,04% | 52,66% | 64,01% | 0,38% | 0,58 |
| Random Forest | 99,25% | 53,11% | 99,62% | 53,74% | 49,77% | 64,45% | 0,38% | 11,27 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,82 |
| Neural Network | 49,79% | 49,79% | 99,44% | 52,87% | 46,82% | 48,82% | 0,56% | 18,23 |
| Decision Tree | 99,24% | 53,22% | 99,62% | 53,40% | 53,31% | 64,02% | 0,38% | 0,51 |
| Random Forest | 99,25% | 53,48% | 99,62% | 53,93% | 49,98% | 64,45% | 0,38% | 11,16 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,78 |
| Neural Network | 48,94% | 48,94% | 99,43% | 52,52% | 47,22% | 47,98% | 0,57% | 17,68 |
| Decision Tree | 99,24% | 51,21% | 99,61% | 51,89% | 51,55% | 63,58% | 0,39% | 0,50 |
| Random Forest | 99,24% | 52,95% | 99,62% | 53,03% | 49,28% | 64,02% | 0,38% | 11,56 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,85 |
| Neural Network | 48,94% | 48,94% | 99,43% | 47,26% | 45,07% | 47,89% | 0,57% | 18,05 |
| Decision Tree | 99,24% | 52,68% | 99,62% | 52,70% | 52,69% | 64,02% | 0,38% | 0,53 |
| Random Forest | 99,28% | 57,19% | 99,64% | 56,60% | 53,48% | 66,62% | 0,36% | 11,61 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,91 |
| Neural Network | 48,94% | 48,94% | 99,42% | 49,41% | 45,18% | 47,89% | 0,58% | 20,25 |
| Decision Tree | 99,23% | 53,62% | 99,61% | 53,56% | 53,59% | 64,01% | 0,39% | 0,56 |
| Random Forest | 99,25% | 53,63% | 99,62% | 53,73% | 50,03% | 64,45% | 0,38% | 12,96 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,90 |
| Neural Network | 49,79% | 49,79% | 99,43% | 47,86% | 45,47% | 48,76% | 0,57% | 17,51 |
| Decision Tree | 99,24% | 52,30% | 99,62% | 51,19% | 51,74% | 64,01% | 0,38% | 0,46 |
| Random Forest | 99,22% | 54,10% | 99,61% | 53,32% | 50,29% | 63,59% | 0,39% | 11,16 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,81 |
| Neural Network | 51,06% | 51,06% | 99,45% | 48,13% | 46,33% | 50,13% | 0,55% | 17,09 |
| Decision Tree | 99,23% | 52,15% | 99,61% | 50,96% | 51,55% | 63,58% | 0,39% | 0,52 |
| Random Forest | 99,28% | 54,54% | 99,63% | 54,88% | 51,10% | 66,18% | 0,37% | 11,18 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,80 |
| Neural Network | 52,77% | 52,77% | 99,47% | 50,84% | 49,28% | 51,81% | 0,53% | 17,43 |
| Decision Tree | 99,23% | 52,69% | 99,61% | 52,71% | 52,70% | 64,01% | 0,39% | 0,46 |
| Random Forest | 99,28% | 55,12% | 99,63% | 55,41% | 51,92% | 66,17% | 0,37% | 11,16 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,81 |
| Neural Network | 51,06% | 51,06% | 99,46% | 51,57% | 48,25% | 50,07% | 0,54% | 17,56 |
| Decision Tree | 99,22% | 50,88% | 99,61% | 52,14% | 51,50% | 62,72% | 0,39% | 0,48 |
| Random Forest | 99,23% | 55,33% | 99,61% | 54,79% | 51,78% | 64,45% | 0,39% | 10,99 |
| Naïve Bayes | 98,80% | 44,12% | 99,39% | 40,67% | 42,32% | 46,28% | 0,61% | 1,80 |
| Neural Network | 53,62% | 53,62% | 99,47% | 53,89% | 49,68% | 52,74% | 0,53% | 17,02 |

Figure C.14: The performance metrics scores of each model from all 10 individual folds.

## C.3 Feature Selection Importances

### C.3.1 Classifier Specific method

Table C.1: Classifier Specific feature importance scores of each model (based on 10-fold cross-validation).

| | **Model** | | | |
|---|---|---|---|---|
| **Feature** | Decision Tree | Random Forest | Naive Bayes | Neural Network |
| *RideName* | 0,02930 | 0,05907 | 0,08455 | 0,05879 |
| *ScenarioDayNumber* | 0,01119 | 0,03244 | 0 | 0,05942 |
| *ShiftID* | 0,03672 | 0,05608 | 0,13825 | 0,09950 |
| *RideNumber* | 0,52490 | 0,13589 | 0,14970 | 0,08492 |
| *Subcontractor* | 0,11114 | 0,06827 | 0,08044 | 0,10446 |
| *ConsigneeParty_PostalCode* | 0,01703 | 0,05188 | 0 | 0,02237 |
| *ConsigneeParty_CityName* | 0,00329 | 0,03703 | 0,04411 | 0,05294 |
| *ConsigneeParty_BuildingNumber* | 0,00917 | 0,04663 | 0 | 0,02083 |
| *ConsigneeParty_CountryCode* | 0,00421 | 0,00741 | 0,04664 | 0,01106 |
| *ConsignorParty_PostalCode* | 0,04390 | 0,05924 | 0 | 0,05769 |
| *ConsignorParty_CityName* | 0,02073 | 0,04394 | 0 | 0,02009 |
| *ConsignorParty_BuildingNumber* | 0,01370 | 0,04178 | 0 | 0,00178 |
| *ConsignorParty_CountryCode* | 0 | 0,00703 | 0,04047 | 0,01093 |
| *Party_PostalCode* | 0,01619 | 0,05334 | 0 | 0,01919 |
| *Party_CityName* | 0,00675 | 0,03703 | 0,07538 | 0,03907 |
| *Party_BuildingNumber* | 0,01240 | 0,04628 | 0 | 0,01831 |
| *Party_CountryCode* | 0,00351 | 0,00714 | 0,01264 | 0,01013 |
| *Stops* | 0,04744 | 0,05577 | 0,08627 | 0,08838 |
| *Parcels* | 0,03674 | 0,05182 | 0,07090 | 0,07572 |
| *Volume* | 0,03036 | 0,05015 | 0,06159 | 0,07620 |
| *Weight* | 0,02133 | 0,05179 | 0,10905 | 0,06823 |

### C.3.2 Classifier Agnostic method

Table C.2: Permutation Importance feature scores of each model (based on 10-fold cross-validation).

| | **Model** | | | |
|---|---|---|---|---|
| **Feature** | Decision Tree | Random Forest | Naive Bayes | Neural Network |
| *RideName* | 0,4545 | 0,0740 | 0,3583 | 0,0849 |
| *ScenarioDayNumber* | 0,1711 | 0,0451 | 0,2996 | 0,1045 |
| *ShiftID* | 0,0596 | 0,0374 | 0,2349 | 0,0995 |
| *RideNumber* | 0,0185 | 0,0222 | 0,0170 | 0,0224 |
| *Subcontractor* | 0,0502 | 0,0179 | 0,0298 | 0,0762 |
| *ConsigneeParty_PostalCode* | 0,0298 | 0,0170 | 0,0860 | 0,0884 |
| *ConsigneeParty_CityName* | 0,0230 | 0,0102 | 0,0230 | 0,0594 |
| *ConsigneeParty_BuildingNumber* | 0,0196 | 0,0077 | 0 | 0,0192 |
| *ConsigneeParty_CountryCode* | 0,0281 | 0,0043 | 0 | 0,0577 |
| *ConsignorParty_PostalCode* | 0,0145 | 0,0043 | 0,0128 | 0,0588 |
| *ConsignorParty_CityName* | 0 | 0,0043 | 0,0068 | 0,0018 |
| *ConsignorParty_BuildingNumber* | 0,0477 | 0,0034 | 0,0749 | 0,0757 |
| *ConsignorParty_CountryCode* | 0,0111 | 0,0017 | 0,0187 | 0,0529 |
| *Party_PostalCode* | 0,0017 | 0,0009 | 0,0034 | 0,0111 |
| *Party_CityName* | 0,0026 | 0,0009 | 0,0034 | 0,0109 |
| *Party_BuildingNumber* | 0 | 0,0009 | 0,0179 | 0,0391 |
| *Party_CountryCode* | 0,0128 | 0 | 0,0306 | 0,0208 |
| *Stops* | 0,0111 | 0 | 0 | 0,0201 |
| *Parcels* | 0,0085 | 0 | 0,0272 | 0,0183 |
| *Volume* | 0,0017 | 0 | 0,0034 | 0,0101 |
| *Weight* | 0,0255 | 0 | 0,0621 | 0,0682 |