An Interpretable Machine Learning Approach for Predicting Ground-Handling End Time Adherence

Master of Science Thesis

L.J. Dijkstra

Technische Universiteit Delft

TU Delft knowledge & development centre Mainport & Airport

An Interpretable Machine Learning Approach for Predicting Ground-Handling End Time Adherence

by

L.J. Dijkstra

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday, August 27th at 9:30.

Student number:	4860403	
Project duration:	March 2023 – July 2024	
Thesis committee:	Dr. A. Bombelli	Chair
	Dr. O.A. Sharpanskykh	Daily Supervisor (Delft University of Technology)
	W. Schaberg, MSc	Daily Supervisor (Royal Schiphol Group)
	Dr. ir. C. Borst	External Examiner

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Acknowledgements

With the submission of this thesis, my journey as a student at the faculty of Aerospace Engineering at Delft University of Technology comes to an end. I have been interested in aviation for as long as I can remember. Over the past 6 years, I've had the opportunity to learn a lot about the technical and non-technical aspects of the aviation industry, meet and collaborate with inspiring people and develop myself both academically and personally. Finishing this degree is a dream coming true, but it has not always been easy. I am, however, very proud to present to you my thesis called: *An Interpretable Machine Learning Approach for Predicting Ground-Handling End Time Adherence.*

This thesis would have not been possible without the continuous support, guidance, expertise and feedback from my daily supervisors dr. Alexei Sharpanskykh, Wouter Schaberg (MSc) and Jos van den Berg (MSc), as well as from dr. ir. Marcel Raas. Your support helped me elevate the quality of this thesis and helped me grow professionally. I also appreciate the academic freedom you granted me, which allowed me to explore and develop my ideas and research. My sincere gratitude goes out to all of you.

I would also like to express my gratitude to my colleagues at the Schiphol Airport Operations Centre (APOC), and I would especially like to thank ing. Tim Grob, whose operational expertise in ground-handling operations has been really valuable for this research. Additionally, I would like to express my gratitude towards the Royal Schiphol Group (RSG) and Knowledge and Development Centre (KDC) for providing me the opportunity to perform this thesis in such a fun and exciting environment.

To my parents, sister and girlfriend, your unconditional love and support, patience, understanding, encouragement and distractions through the good and bad times have been fundamental to my academic achievements, for which I am eternally grateful. Obtaining my master's degree wouldn't have been possible without you, and I am very happy to celebrate this moment in my life with you.

I would also like to express my appreciation to my (study) friends for their support, encouragement, distractions and good times besides studying. The past few years have not all been smooth sailing, and I deeply appreciate the help and support you consistently offered me over the past years during turbulent times.

Lars Dijkstra
Dronten, July 2024

Contents

List of Figures

List of Tables

List of Abbreviations

A-CDM	Airport Collaborative Decision Making
AAS	Amsterdam Airport Schiphol
ABM	Agent-Based Modeling
ACARS	Aircraft Communications Addressing and Reporting System
ACGT	Actual Commence of Ground-handling Time
AdaBoost	Adaptive Boosting
AIBT	Actual In-Block Time
ALDT	Actual Landing Time
ANN	Artificial Neural Networks
ANSP	Air Navigation Service Provider
AO	Aircraft Operator
AOBT	Actual Off-Block Time
ARDT	Actual Ready Time
ASAT	Actual Start-up Approval Time
ASRT	Actual Start-up Request Time
ATC	Air Traffic Control
ATFCM	Air Traffic Flow and Capacity Management
ATFM	Air Traffic Flow Management
ATOT	Actual Take-Off Time
BNN	Bayesian Neural Network
CNN	Convolutional Neural Network
DAG	Direct Acyclic Graph
DNN	Deep Neural Network
EFD	Flight Data
EOBT	Estimated Off-Block Time
ETFMS	Enhanced Tactical Flow Management System
ETOT	Estimated Take-Off Time
FAST	Fourier Amplitude Sensitivity Test
FMP	Flow Management Position
FN	False Negative

FP	False Positive
GBDT	Gradient Boosted Decision Trees
GDP	Ground Delay Programs
GMM	Gaussian Mixture Models
GRU	Gated Recurrent Unit
HBN	Hybrid Bayesian Network
IFR	Instrument Flight Rules
JPD	Joint Probability Distribution
KNMI	Royal Netherlands Meteorological Institute
LCC	Low-Cost Carrier
LIME	Local Interpretable model-agnostic Explanations
LSI	Latent Semantic Indexing
LSTM	Long Short-Term Memory
LVNL	Luchtverkeersleiding Nederland
MAE	Mean Absolute Error
MAP	Maximum A Posteriori
MAPE	Mean Absolute Percentage Error
MARS	Multivariate Adaptive Regression Splines
MDN	Mixed Density Networks
METAR	Meteorological Aerodrome Report
MGHA	Main Ground handler
MLP	Multilayer Perceptron
MSE	Mean Squared Error
MTT	Mininum Turnaround Time
NMOC	Network Manager Operation Centre
OTP	On-Time-Performance
PCA	Principal Component Analysis
PFI	Permutation Feature Importance
PRM	Passenger with Reduced Mobility
PSAP	Process Structure Aware Prediction
ReLU	Rectified Linear Unit Function
RETD	Revised Estimated Time of Departure
RF	Random Forest
RMSE	Root Mean Squared Error

SA	Sensitivity Analysis
SAOC	Schiphol Airline Operators Committee
SHAP	SHapley Additive exPlanations
SIBT	Scheduled In-Block Time
SID	Standard Instrument Departure
SOBT	Scheduled Off-Block Time
SPNN	Serial Process Neural Network
TN	True Negative
TOBT	Target Off-Block Time
TP	True Positive
TSAT	Target Start-up Approval Time
XGBoost	eXtreme Gradient Boosting

Introduction

Over the past years, air travel demand has recovered from the impacts of the COVID-19 pandemic and is projected to grow with an annual compound growth rate of 3.6% between 2023 and 2042 [215]. This extra demand will put more stress on the already scarce airport and airspace resources. Therefore, a more efficient and robust orchestration of the components that drive air travel is required than is the case today. One of the main components that facilitates air travel is the aircraft ground-handling process. From the moment the aircraft arrives at the stand until it is ready for departure, a series of ground-handling processes unfold that prepare the aircraft for its next flight. Currently, the ground-handling process forms a black box for airports and many operational partners, such as Air Navigation Service Providers (ANSPs) and aircraft operators.

This study proposes a novel machine learning approach for ground-handling end time adherence predictions, using data from camera systems that automatically track ground-handling processes ongoing at the aircraft stand. The approach is inherently interpretable, providing transparency into the decision-making process of the model, an area of research which has not yet seen much coverage in literature. Modelling the ground-handling process using interpretable machine learning allows the operational partners to not only gain insights into how the model's predictions came about, but also gain insights into the turnaround system and how the ground-handling processes interact with the onset of delays. The model's predictions and insights can be used by the operational partners to improve decision-making and operational efficiency, and in turn, facilitate an increase in the robustness of the air traffic network and resource utilization.

This thesis report is organized as follows: In Part I, the scientific paper is presented. Part II contains the relevant Literature Study that supports the research. Finally, in Part III, some additional results are presented.

I

Scientific Paper

An Interpretable Machine Learning Approach for Predicting Ground-Handling End Time Adherence

L.J. Dijkstra*

Delft University of Technology, Delft, The Netherlands

Abstract

To accommodate the rising demand for airport resources and airspace capacity, a more efficient and robust orchestration of the components that facilitate air travel is required than is the case today. Over recent years, camera systems for automatic identification of ground-handling events were developed, providing the opportunity to automatically detect ground-handling delays as they manifest. Detecting ground-handling delays early allows the Air Navigation Service Providers (ANSPs), ground handlers and aircraft operators to reallocate their resources, facilitating more stable planning and robustness of the air traffic network. This study proposes a novel interpretable machine learning approach to predict initial ground-handling end time adherence. This approach is based on a discrete Bayesian network composed of a central network structure and modular sub-networks, representing distinct turnaround processes often found on the critical path. The Bayesian network structure and discretization were found to be adequate to model most ground-handling operations for narrow-body aircraft. The model proved to be more accurate than the ground handler in predicting initial Target Off-Block Time (TOBT) adherence at all prediction moments. At the initial TOBT-0 [minutes], a maximum in model accuracy was obtained amounting to 65.76%. In addition, the model demonstrated the capability to detect ground-handling delays earlier. At the initial TOBT-20 [minutes], an average recall of 50.4% was obtained for delays within 5 and 15 minutes. In contrast, the ground handler identified only 4.3% of small ground-handling delays at this prediction moment. The discrete Bayesian network approach is also inherently interpretable, providing transparency into the decision-making process of the model.

1 Introduction

In recent years, air travel demand recovered from the impacts of COVID-19 and is projected to surpass the 2019 level of demand for the first time since the onset of the global pandemic, with a compound annual growth rate of 3.6% between 2023 to 2042 [51]. Consequently, airlines will strive to operate more flights to fill this increasing demand for air travel, further straining the already scarce airport resources and airspace capacity. To accommodate the rising demand for airport resources and airspace capacity, a more efficient orchestration and robustness of the components that facilitate air travel is required than is the case today.

Airport Collaborative Decision Making (A-CDM) is already implemented at various airports in Europe to enhance the robustness of the air traffic network. A-CDM enhances the overall efficiency at airports and improves the Air Traffic Flow and Capacity Management (ATFCM), by tackling unharmonized processes [18, 39]. To effectuate this, A-CDM strives to improve the predictability of events and optimise the utilisation of resources. In A-CDM, operational partners such as the Air Navigation Service Providers (ANSPs), ground handlers and airlines are encouraged to work more transparently and collaboratively by sharing information on the status of a flight [50, 14]. A flight's status is defined by a milestone and describes a significant event in the planning or progress of a flight [39]. An important milestone is the Target Off-Block Time (TOBT), the time instance the aircraft operator or handling agent estimates the aircraft ground-handling process is finished and able to start-up and push-back immediately when clearance is obtained from the ANSP [13]. Based on the TOBT, Estimated Taxi-Out Time (EXOT) and runway capacity, the ANSP's Departure Manager (DMAN) creates a departure sequence from the runway. Subsequently, a Target Take-Off Time (TTOT) is issued for every flight. This time indicates when a flight is expected to begin its take-off roll. Accurate TOBTs are therefore crucial for realising the optimal departure flow. From the moment the aircraft arrives at the stand until it is ready for departure, a series of ground-handling processes unfold that prepare the aircraft for its next flight. If ground-handling cannot be completed within TOBT±5 [minutes], then the handler or aircraft operator must update the TOBT. Subsequently, the ANSP issues a new TTOT [39]. Early identification of TOBT non-compliance enables the

*MSc Student, Sustainable Air Transport, Faculty of Aerospace Engineering, Delft University of Technology

reallocation of departure slots, mitigating the disruptive effects of ground-handling delays and improving the overall departure capacity utilization [49].

In recent years, academics have already focused on modelling aircraft ground-handling delays. Although in the past probabilistic approaches were more commonplace (e.g. [15, 4, 35]), nowadays most researchers opt for advanced machine learning models (e.g. [20, 19, 17, 27]) offering higher model accuracy and enhanced abilities to capture complex data relationships. Previous work primarily focused on predicting ground-handling delays before the flight's arrival, rather than updating predictions during the turnaround based on observed events. This was likely due to the researchers having limited or no access to information regarding turnaround events, as automatic turnaround sub-process identification systems were not available yet. However, recent advancements in computing and computer vision have enabled the development of systems such as Deep Turnaround [40]. These systems open up a new area of research for predicting ground-handling delays based on real-time observations of ground-handling processes, an area largely unexplored until now.

Within this area of research, Luo et al. [27] utilized data from automatic turnaround process identification systems to predict the total aircraft ground-handling duration and confirm TOBT adherence at the final TOBT update milestone, which in A-CDM is a milestone before boarding commences. To predict TOBT adherence, Luo et al. [27] employed tree-based machine learning models. Similarly, Schiphol Group Aviation Solutions [40] uses data from automatic turnaround process identification systems for the Predicted End Ground-handling Time (PEGT) model, predicting the ground-handling end time of a flight every 5 [minutes] using a black-box machine learning model. Within the area of dynamical predictions, Beltman [6] employed Random Forest, CatBoost and Deep Neural Networks to update his belief on the flight departure delay during the turnaround. However, for this, Beltman [6] did not utilize data from automatic sub-process identification systems. In the aforementioned approaches, the interpretability of the underlying model(s) is limited, where interpretability refers to the degree to which the decision or prediction process is understandable by an expert in the field. While obtaining accurate predictions on ground-handling delays is important, understanding how the model arrived at its predictions can be considered equally important. After all, an interpretable approach can foster a user's trust in the model and allow one to gain insights into the underlying system, trading model accuracy for model interpretability and system understanding.

To be able to detect ground-handling delays as early as possible and to explore an interpretable approach in this research area, the objective of this research can be formulated as follows: *To develop and evaluate an interpretable machine learning model to predict initial ground-handling end time adherence.* In this context, the initial TOBT when the aircraft arrives at the stand is regarded as the initial ground-handling end time. To arrive at the ground-handling delay, the initial TOBT is compared with the Actual End of Ground-handling Time (AEGT). Initial ground-handling end time adherence is predicted at various moments before the initial TOBT, ranging from initial TOBT-50 to initial TOBT-0 [minutes] with a 10-minute prediction interval. A visual representation of these prediction instances is provided in Figure 1. The start at initial TOBT-50 [minutes] of the prediction horizon was influenced by the typical turnaround time for narrow-body aircraft at Schiphol of the particular aircraft operator and to provide predictions early within the turnaround process. A 10-minute prediction interval was selected to reduce the complexity and computational cost of the model.

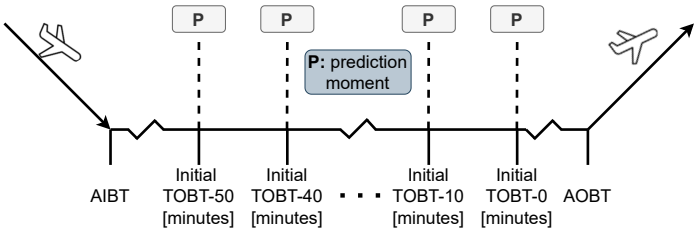

Figure 1: Visual representation of the prediction instances.

To accomplish the objective, this paper proposes a discrete Bayesian network approach that accommodates the temporal aspect of the aircraft ground handling process and supports multiple prediction moments. A Bayesian network is a probabilistic graphical model, where the underlying relationships between the variables are explicitly defined through edges. In the approach, the Bayesian network structure is split into a set of modular sub-networks representing distinct turnaround processes and a central network structure, which processes the information propagated from the modular sub-networks to predict initial TOBT adherence. This results in a novel interpretable machine learning approach for ground-handling end time adherence modelling, which provides transparency into the model's decision-making process.

The paper is organized as follows. Initially, in section 2, we present an overview of the turnaround system and the associated challenges. Following that, the Bayesian network theory, conceptual framework and implementation are elaborated on in section 3. Next, the Bayesian network structure is discussed in section 4. The model evaluation results on various Key Performance Indicators (KPIs) are presented and discussed in section 5.

A discussion on the model, among other things its limitations and broader implications, is provided in section 6. Lastly, we conclude the paper in section 7 with the main conclusions of this research and opportunities for future work.

2 Aircraft Ground-Handling

An aircraft turnaround consists of many individual ground-handling processes, which together prepare the aircraft for its next flight. Ground-handling processes commence as soon as the aircraft arrives at the stand, and may be provided by one or more service providers [41]. An overview of the main processes in aircraft ground-handling for narrow-body aircraft is presented in Figure 2.

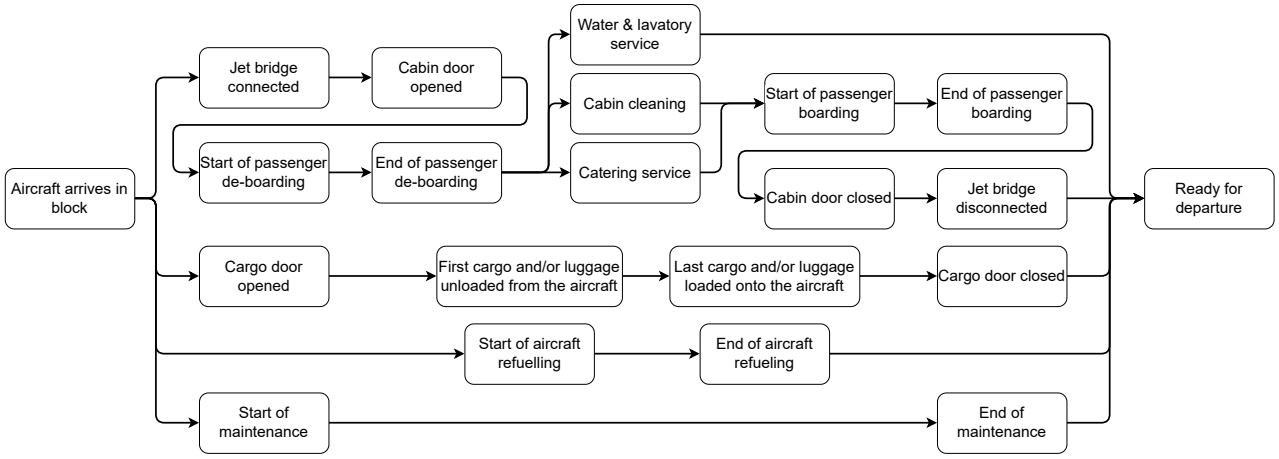

Figure 2: Main ground-handling processes for narrow-body aircraft at connected aircraft stands (based on [55]).

When a narrow-body aircraft halts into position at the stand, a jet bridge or set of stairs is connected to the aircraft. After a short handshake between the crew and staff, the passenger disembarking process starts. Parallel to this process, the cargo doors are opened, the cargo is unloaded and other processes such as maintenance start. When passenger de-boarding has finished, the aircraft cabin is cleaned, and the catering provider exchanges the in-flight catering trolleys to limit the disturbance to passengers [41]. In parallel, the fuel service provider loads the required fuel onto the aircraft for its next flight. Generally, when previous processes are finished, passenger embarking starts and the last pieces of baggage are loaded onto the aircraft. After passenger boarding is completed, the crew closes the aircraft doors and the ground staff disconnects the set of stairs or jet-bridge. When all ground-handling processes have finished and start-up clearance is obtained from ATC, a push-back truck pushes the aircraft back from its parking position so that it can commence its journey on its power.

In aircraft ground-handling, some processes are dependent on other processes due to operational, legal or technical dependencies and therefore have to be executed sequentially, as Figure 2 indicates [15, 45]. In general, aircraft ground-handling processes follow a strict chronological order due to the dependencies between processes, regulations and limitations in space in and around the aircraft [41]. However, some processes can be executed in parallel. The critical path, i.e. the longest sequence of parallel and sequential activities, defines the total aircraft ground-handling time. When activities on the critical path are delayed, the whole process is delayed [34]. Due to the complex nature of the aircraft ground-handling operations, the processes are stochastic resulting in a variable critical path [3, 4, 52]. Hence, the critical path may vary between turnarounds and no unique critical path can be defined.

The onset of ground-handling delays is not immediate. Rather, they manifest at some point within the turnaround process in one or more of the ground-handling processes. At a given moment, the ground-handling process may be on track. However, delays may potentially manifest later on, and only then the ground-handling process becomes delayed. For example, passengers may be missing during the boarding process or complications may arise in the aircraft refuelling process, delaying the ground-handling process. Next to this, as the critical path is variable and not all activities lie on the critical path, a delay in a particular process doesn't necessarily lead to a departure delay. Suppose a small delay manifests in the baggage loading and unloading process but is not part of the critical path for this flight. In this case, the projected ground-handling end time remains unaffected. Conversely, a minor delay in passenger boarding, generally found on the critical path, may already manifest a ground-handling delay.

Next to passenger boarding, the baggage loading and unloading, aircraft refuelling, cabin cleaning, catering and passenger de-boarding processes are often found on the critical path for narrow-body aircraft [41, 43, 3]. The duration of these processes is subject to many variables and uncertainties. For example, the duration of the

passenger boarding process is influenced by various variables. A summarized list of influential variables for this particular process is presented below. The many variables and uncertainties at play make accurately predicting the total duration of a process rather challenging:

- Number of outbound passengers [36, 41, 43, 44, 47]
- Airport infrastructure
- Missing (transfer) passengers [47]
- Layout of the aircraft cabin [41]
- Boarding scheme [44]
- Passengers with Reduced Mobility (PRM) [20]
- Passenger anthropometrics [41]
- Amount of hand luggage [38, 41, 42]

Moreover, not all ground-handling processes are executed in every aircraft turnaround. For example, airlines may opt to fuel at airports with lower fuel costs, thus omitting the aircraft refuelling process from the ground-handling planning at the destination airport. In this case, the savings in fuel cost outweigh the cost of carrying extra fuel on the flight to the destination airport. Next to this, for short-haul flights, airlines usually only exchange catering trolleys and clean the aircraft cabin at their hub bases. The turnaround schedule of ground-handling activities between airlines also varies. As an example, one major European Low-Cost Carrier (LCC) has opted to remove the seat pockets on the aircraft seats to reduce the duration of the cabin security check [48]. Differences in turnaround schedules inherently lead to differences in the critical path.

3 Methodology

In this section, the methodology of this research is elaborated on. First, the theory behind Bayesian networks and the conceptual framework of this thesis are elaborated on in section 3.1. Next, the implementation of the conceptual framework is discussed in section 3.2.

3.1 Bayesian Networks

This section discusses Bayesian networks, a probabilistic machine learning model, and the conceptual framework implemented in this study. First, section 3.1.1 elaborates on the theory behind Bayesian networks after which the conceptual framework on which this research was built is discussed in section 3.1.2.

3.1.1 Theory

A Bayesian network is a graphical representation of a Joint Probability Distribution (JPD) over a set of random variables and can represent knowledge about an uncertain domain [26, 53]. In a Bayesian network, the underlying system is modelled as a network of interactions between variables, where the underlying relationships between the variables are explicitly defined from root cause to the target outcome [11]. A Bayesian Network is composed of two elements: (1) a Direct Acyclic Graph (DAG) and (2) Conditional Probability Tables (CPTs) defining the strength of the relationships between the variables for all nodes in the network [25]. The DAG and CPTs define the qualitative and quantitative aspects of the network, respectively [37].

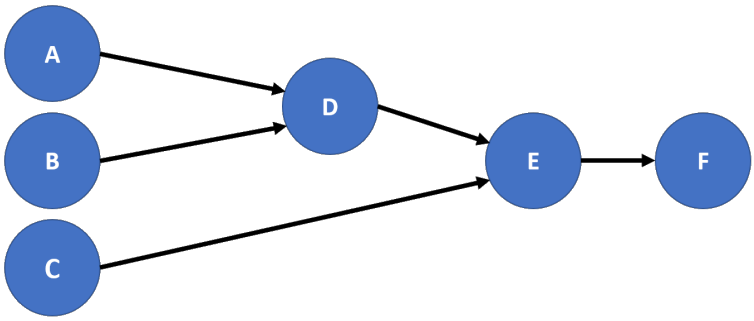

Figure 3: Example Direct Acyclic Graph (DAG).

The DAG comprises a set of nodes and directed edges, representing the random variables and the direct probabilistic dependencies between the random variables [10]. An example DAG is presented in Figure 3. A directed edge between nodes X_i and X_j denotes a statistical dependence of X_j on X_i, and a lack thereof indicates no direct dependency of X_j on X_i. In other words, the state of variable X_j depends on the state variable X_i takes [7]. Within the structure of the DAG, no loops or edges connecting the node to itself on a directed path are allowed, as this violates the acyclic nature of the DAG [53]. Bayesian networks assume conditional

independence, and the DAG encodes the set of conditional independence statements [7, 30]. Consider the two variables A and B and a third variable C. Variables A and B are conditionally independent given the state of the third variable C when knowledge of the state of A does not provide additional information on the state of B [21].

In discrete Bayesian networks, when evidence (input) is observed, the probability distribution of the observed node is updated. In turn, the posterior probability distribution of the unobserved parent and child nodes are affected, and so on, propagating through the network [25]. This evidence propagates along unblocked paths. A path from X_i to X_j can be considered blocked, or D-separated when one or more nodes in the set of interior nodes satisfy one of the following properties: (1) a collider node is considered and neither it nor one of its descendants are in the evidence node set $\mathbf{E}$ or (2) a non-collider node is considered and the node is in the set of evidence nodes $\mathbf{E}$ [7, 32]. To illustrate property (1), consider the path between A and B in Figure 3, given by $A \rightarrow D \leftarrow B$. The only interior node within this path is node D, a collider node along the path between A and B. If neither node D nor its descendants E or F are in the set of evidence nodes $\mathbf{E}$, then the path between A and B can be considered blocked. When node D or one of its descendants are in the evidence set $\mathbf{E}$, the path between A and B opens, creating a conditional dependence between the variables. To illustrate property (2), consider the path between nodes A and F, given by $A \rightarrow D \rightarrow E \rightarrow F$. Here, the interior nodes are D and F, none of which is a collider along the path. Therefore, the path between A and F can be considered open. However, A and F become D-separated when D, E or both become part of the set of evidence nodes $\mathbf{E}$. The forward and backward propagation of information allows Bayesian networks to be used for bottom-up, i.e. diagnostic, and top-down, i.e. explanatory, reasoning.

More formally, the qualitative and quantitative components of a Bayesian Network are defined as follows [16, 7, 30]. The Direct Acyclic Graph is denoted by G, representing the relationships between the random variables V. The quantitative element, the conditional probability tables, is given by the set Θ. In the set Θ, for every possible state of node X_i conditioned on the parents of X_i in G, π_i, the conditional probability of X_i given its parents $\theta_{X_i|\pi_i}$ is present. The local probability distribution of root nodes is unconditional, as root nodes have no parents. The joint probability distribution over the set of random variables V can then be described by Equation 1 [7]. Consequently, the joint probability distribution of the DAG in Figure 3 can be represented by Equation 2.

$$P(X_1, ..., X_N) = \prod_{i=1}^{n} P\left(X_i|\pi_{X_i}\right) = \prod_{i=1}^{n} \theta_{X_i|\pi_i} \tag{1}$$

$$P(A, \ldots, F) = P(F \mid E) \cdot P(E \mid C, D) \cdot P(C) \cdot P(D \mid A, B) \cdot P(A) \cdot P(B) \tag{2}$$

Variable elimination can be employed to compute the marginal distribution over a set of variables of interest, describing the probabilities of the variables of interest taking on a specific value, regardless of the values of the other variables [54]. Variables that are irrelevant to the query are eliminated by summing over all their possible values. For each variable to be eliminated, two operations are performed [24]. The first operation sums out the variable by creating a new factor that represents the sum of the values of the original factor over all possible values of the variable. Next, the factor is multiplied by the remaining factors in the network, which is subsequently normalized to represent a valid probability distribution. After all variables have been eliminated, the marginal probability of the variable(s) of interest remains. To illustrate the variable elimination algorithm, suppose that we want to eliminate variable A from Equation 2. We then first sum Equation 2 over all possible values of A, where factors without variable A remain unaffected. Variable A can is then represented by the factor $\lambda_A(B, D) = \sum_A P(D|A, B) \cdot P(A)$, which is the sum over all possible values of A. This factor is subsequently multiplied by the remaining factors in the network and normalized to represent a valid probability distribution.

3.1.2 Conceptual Framework

An aircraft turnaround is a large, complex and temporal system, which can be decomposed into a set of individual ground-handling processes. In decomposable systems, the Bayesian network structure can be broken down into network fragments [8, 9]. This enhances model interpretability, as the underlying system can be more easily conceptualised [31]. In this paper, the Bayesian network structure is split into a set of modular sub-networks, representing distinct turnaround processes, each connected to one central network structure. This structure processed the information propagated from the various network fragments and computed the distribution of the ground-handling delay classes.

The central network structure, of which a schematic is presented in Figure 4, was composed of several nodes. In this structure, the most important node was the ground-handling delay node. This node formed the centre of the structure, as the modular sub-network structures and several other nodes were all connected to this node. As a result, all information from the nodes in the complete network structure propagated towards the ground-handling delay node. One of the nodes within the central network structure represented the current time to

the initial target off-block time. This node provided the crucial temporal context to the ground-handling delay node. Next to this, a set of context nodes were also connected to the ground-handling delay node. These nodes represented differentiating elements between aircraft turnarounds. One example from this set was the initial total available turnaround time, as flights with longer initial total turnaround times often have larger buffers, increasing the likelihood of adhering to the initial TOBT.

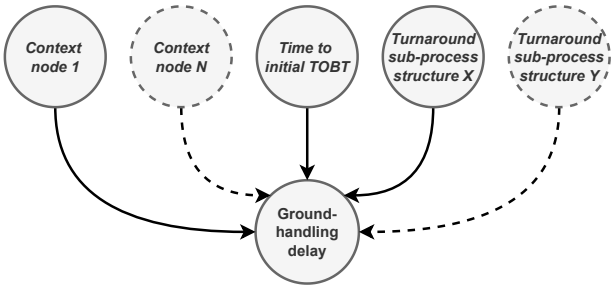

Figure 4: Schematic of the central Bayesian network structure.

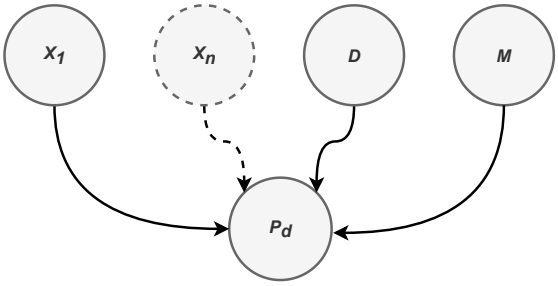

Figure 5: General model Bayesian network structure of the turnaround sub-processes.

The modular sub-network structures for the individual ground-handling processes did not require the same temporal context as the central network structure. Rather, the modular sub-network structure accommodated the various states the particular process takes on. To reduce model complexity and improve the overall understanding of the inner workings of the network structures, a uniform modular sub-network structure was implemented for the main processes in the ground-handling process often found on the critical path. This general sub-network structure is illustrated in Figure 5 and comprised of four types of nodes, namely: (1) a set of context nodes $\mathbf{X}$ closely related to the duration of the process, (2) a dynamic node $\mathbf{D}$ representing the current progression in the ground-handling process as a measure of time, (3) a milestone node $\mathbf{M}$ denoting the current state of the process and (4) the target variable $\mathbf{P_d}$, which represented the expected total time remaining in the particular process. The set of context nodes $\mathbf{X}$ and the nodes $\mathbf{D}$, $\mathbf{M}$ served as evidence (input) in the Bayesian network. Together, they served as input to the node $\mathbf{P_d}$.

To illustrate the various types of nodes, the aircraft fueling process is used as an example. This running example will also be used to explain the data discretization methodology in section 3.2.2. The model structure of the aircraft fueling process is presented in Figure 6. The figure shows that the set of context nodes $\mathbf{X}$ was formed by the *outbound flight distance* and *aircraft type* nodes. The current total time that the fueling truck was in position represented the dynamic node $\mathbf{D}$, and the milestone node $\mathbf{M}$ was illustrated by the *fueling status* node. This node denoted the status of the process, i.e. whether it has not started, is ongoing or has finished, and changed state when a certain milestone was observed. Here, a fueling truck moving out of position serves as an example milestone. The nodes served as an input to node $\mathbf{P_d}$, depicting the remaining estimated that the fueling truck will be in position during the turnaround.

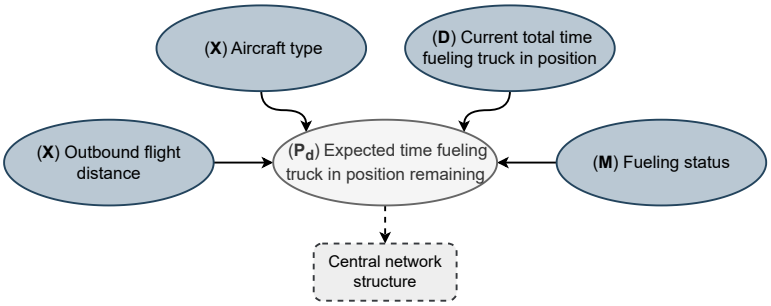

Figure 6: Model structure of the aircraft (re)fueling process. Letters indicate the type of node.

To demonstrate how the sub-network structure in Figure 6 accommodated the various states the fueling process can take on, we illustrate its inner workings through an example. When the fueling process had not commenced yet, the expected total time remaining in a sub-process was equal to the expected total duration of the process. In this case, the distribution of node $\mathbf{P_d}$ was solely determined by the states of the *aircraft type* and *outbound flight distance* nodes, set $\mathbf{X}$, as the evidence states of nodes $\mathbf{D}$ and $\mathbf{M}$ in all cases indicated that the process has not started. This made the expected distribution of the total duration independent of nodes $\mathbf{D}$ and $\mathbf{M}$, as only changes to the states of the nodes in set $\mathbf{X}$ could be affiliated to changes in node $\mathbf{P_d}$'s state. When the process commenced, the state of the dynamic node $\mathbf{D}$ and milestone node $\mathbf{M}$ changed. Here, the

milestone node **M** indicated the current status of the process, i.e. that the process is ongoing, while the current total time that the fueling truck has been in position during the turnaround served as a proxy of progression in the fueling process. The state of the set of context nodes **X** remained unchanged as they are independent of the current status of the process. Together, these nodes formed inputs to node $\mathbf{P_d}$, where the distribution of the expected time remaining that the fueling truck will be in position during the turnaround was computed. When the process had finished, the state of milestone **M** changed to a state that could only be attributed to the condition in which the process had finished, indicating that the total time remaining of the fueling truck in position had to be zero.

3.2 Model Development Pipeline

This subsection will elaborate on the model development process and the implementation of the conceptual framework, explained within the context of a machine learning pipeline.

3.2.1 Data Ingestion

For this study, data on flights arriving and departing from Amsterdam Airport Schiphol (AAS) between the 1st of July 2021 to the 17th of October 2023 was acquired from several sources. The core of the dataset was formed by detections from the Deep Turnaround system, a system developed and implemented at AAS. Deep Turnaround utilises cameras and computer vision to track over 70 turnaround events in 30 turnaround processes [40]. The system comprises two cameras on each flank of the aircraft, a schematic of which is presented in Figure 7, tracking distinct turnaround processes ongoing at the aircraft stand. Every 5 seconds, the cameras at the stand capture an image and forward this to the computer vision model [40]. From the computer vision model, a set of detections of turnaround events is obtained, containing information on the current state of the turnaround. Examples of turnaround events include a fuel truck moved into position, or the appearance of a catering truck at the aircraft stand. However, the set of detections at a specific moment in time provides little to no context on the previous evolution of the turnaround system, as detections only contain information on the current turnaround state. Therefore, detections obtained from the Deep Turnaround system were converted to time-series data.

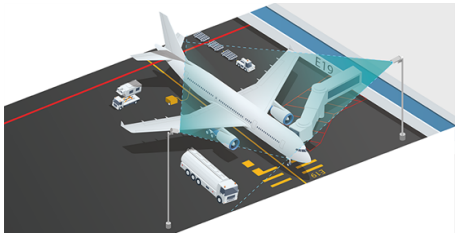

Figure 7: Schematic of the Deep Turnaround camera setup [40].

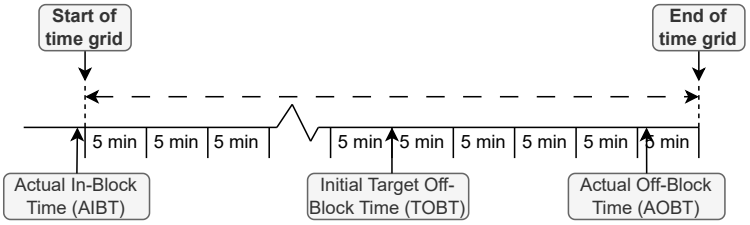

Figure 8: Example time grid for a delayed flight.

To assist in converting the computer vision detections to time-series data, a time grid was constructed with an interval of 5 minutes. The time grid for a flight was based on its initial target off-block time and spans from the Actual In-Block Time (AIBT) to the Actual Off-Block Time (AOBT), visualized in Figure 8. While prediction moments beyond the initial target off-block time were beyond the study's scope, time grid elements beyond the initial TOBT to the AOBT were included for delayed flights. These records contained information on the future evolution of the turnaround system in case ground-handling was delayed, which was utilized in the data pre-processing and feature engineering process. This means that the time grid was essentially composed of two elements: (1) the nominal time grid elements between the AIBT and initial TOBT and (2) the time grid elements between the initial TOBT and the AOBT.

As a next step, for every time grid element and every possible event, the cumulative time duration and event count, describing how often the process has started and has come to a halt, of the particular event between the AIBT and specific time grid element was computed. From this, the current total duration of a process and total event count at a particular time instance were obtained. This yielded a vertical dataset for every flight, where a flight was represented by multiple time grid elements. These records contained the cumulative time duration and event count at any time grid element throughout the flight's ground-handling process.

Data on the passenger turnaround processes was obtained from people counters, elements of the passenger flow measurement system at Schiphol. For flights for which Deep Turnaround detections were available, data from the passenger flow measurement system was processed. The flow measurement system counts the number of people passing a particular point in a particular amount of time [5]. Therefore, the flow measurement data was only available in a specific time resolution and created a new time grid. With the time grid and set of

measurements, start- and end-criterion were set up to determine the start time and end time of the turnaround passenger processes. Using these start and time grid elements, the cumulative sum of the total duration of the passenger process over the time grid elements between the start and end elements was obtained. This denoted the current total duration at a specific instance of time that the process has been ongoing.

Next, the dataset was enriched with flight information. For example, the outbound flight distance (calculated using the Haversine function), aircraft type, expected number of inbound passengers, and expected number of outbound passengers. These attributes were added to all time-grid elements of the particular flights, expanding the vertical dataset with a set of columns that were invariant with time.

3.2.2 Data pre-processing

Data pre-processing aims to clean, transform, and prepare the dataset obtained from the data ingestion step so it can be utilized in model development. For discrete Bayesian networks, the data pre-processing process consists of two elements: (1) data cleaning and (2) data discretization. These are discussed in the next paragraphs.

Data cleaning

Data cleaning is the process of fixing or removing incomplete, duplicate, incorrect, and irrelevant records. Records were considered relevant when they were within the research scope and when sufficient supporting data was available. The research scope for this thesis consisted of the elements presented below:

- Turnarounds with inbound and outbound handling;
- Turnarounds on connected aircraft ramps;
- One aircraft operator;
- Initial total available turnaround time between 25 and 180 [minutes];
- Narrow-body aircraft;

Flights that did not fall within these criteria were removed from the dataset. Hence, wide-body aircraft, multiple airlines, tow-in or tow-out turnarounds, overnight stays, and semi-connected and remote turnarounds were not considered in this research. These elements have been removed from the research scope, as they significantly differentiate turnarounds from one another and contribute to model complexity.

Due to the inherent nature of computer vision techniques, turnaround event detections are never 100% accurate. Rather, they may be subject to blank detections [27]. Flights with blank detections, the situation in which a system fails to detect an object, vehicle or process, have been removed with the help of a set of business rules presented below this paragraph. The business rules were set based on the minimum expected process duration and were verified by an expert in aircraft ground-handling. To evaluate the business rules, a flight was represented as a horizontal record which captured the information of the last time grid element. This record lies beyond the AOBT of the flight, Figure 8, and therefore contains the most information on the total process duration of the various processes. When the data in the horizontal record of a flight did not meet all business rules, the specific flight was removed from the dataset.

- During the turnaround, one or more catering trucks have been in position for at least 360 [s].
- During the turnaround, one or more beltloaders have been in position for at least 840 [s].
- During the turnaround, the fueling truck has been in position for at least 540 [s].

Data discretization

Discrete Bayesian networks are unable to deal with continuous input data and therefore require continuous data to be converted to a set of discrete mutually exclusive states. Ideally, one wants to set the boundaries of the intervals at breakpoints or regions of interest. These breakpoints may be essential to achieve model objectives or describe a significant state change [11]. For this, domain knowledge can be employed. However, in most circumstances, breakpoints cannot be defined solely with domain knowledge due to, for example, complex data patterns, precision or time constraints. In these case, various data discretization algorithms available in literature can be utilized.

The most commonly used and simplistic discretization algorithms are equal-width and equal-frequency discretization [11]. The former splits the domain between the minimum and maximum values in user-defined equal-sized intervals. However, ideally, one prioritizes high resolution in regions that encapsulate a lot of information, i.e. have many observations. Equal-frequency discretization splits the data into equal-sized intervals, where all intervals contain roughly the same number of observations. Equal-frequency discretization therefore prioritizes high resolution in regions that encapsulate a significant amount of information, ensuring that the intervals are more true to the underlying distribution. Moreover, equal-frequency discretization consistently yields a singular discretization, a characteristic the popular K-means clustering algorithm lacks. K-means selects the

initial clusters at random and therefore does not guarantee a unique clustering [23]. Next to this, K-means is more susceptible to outliers, as it is a distant-based algorithm. Therefore, due to its simplistic nature, high resolution in regions that encapsulate a lot of information, and the consistent discretization set, a combination of domain knowledge and equal-frequency discretization was selected to establish the breakpoints.

For an overview of the discretization techniques used in the discretization process for the various variables, one is referred to Supporting Work A.2. The static nodes in the set of context nodes $\mathbf{X}$ were either numerical or categorical. Numerical static variables, e.g. the expected number of outbound passengers, were generally discretized using equal-frequency discretization across 5 intervals, to accommodate sufficient resolution in the network structures and to limit model complexity. An exception to this is the ground-handling delay node, which was discretized into three states: (-inf, 5), [5, 15) and [15, inf). These states refer to a timely ground-handling execution, a small ground-handling and a significant ground-handling delay, respectively. These intervals have been selected based on operational significance and required actions. The breakpoint at 5 minutes was chosen as aircraft operators must report any change to their TOBT larger than ±5 minutes to the A-CDM system at Schiphol [39]. A second breakpoint was defined for ground-handling delays of 15 minutes or larger as when the departure time deviates more than 15 minutes from the scheduled departure time, the flight is considered delayed [12]. Categorical static variables were discretized using domain knowledge, and the number of states was discussed and verified with an expert.

The milestone nodes $\mathbf{M}$, e.g. fueling status, were discretized into three states using either the event of the particular process or the time between the current time grid element and the last time a particular event of the process was detected. Generally, one state indicated that the process has not started, one state indicated that the process is in progress and one state indicated that the process was expected to be finished.

For the nodes $\mathbf{P_d}$ and $\mathbf{D}$, a different approach was required. The mutually exclusive set of discrete states for these nodes should accommodate the states the node adopts at the start of the process, states it may adopt when the process is finished and any intermediate states representing the states the node can adopt when the process is ongoing. For these nodes, the following number of states were selected, based on the expected required resolution, interpretability, and model objectives:

- One state denoting the process has not started ($\mathbf{D}$) or has finished ($\mathbf{P_d}$).
- In general, three intermediate states were chosen to represent possible states when the process is ongoing for both nodes $\mathbf{D}$ and $\mathbf{P_d}$, based on the average total process duration and model prediction horizon. Given the short average total duration of the catering process, one intermediate state was removed, while an additional state was added for the baggage loading/unloading process.
- Two states denoting the states the node can adopt when the process has finished ($\mathbf{D}$) or has not started ($\mathbf{P_d}$).

To illustrate the methodology to obtain the aforementioned states for node $\mathbf{P_d}$, we reuse the aircraft fuelling process example introduced in section 3.1.2. Please note that throughout this example, the unit [seconds] is used in the intervals. In the example, node $\mathbf{P_d}$ denoted the expected total time remaining that the fueling truck will be in position at the aircraft stand. First, a state [0, 1) was defined, indicating that it is expected that the fueling truck will no longer be in position during the turnaround. As a next step, states the node can adopt when the process has not started, where the expected time remaining equals the total process duration, were defined. For this, the vertical dataset for every flight was reduced to one row which encapsulated information on the overall total duration of the sub-process, creating a horizontal dataset. This row was either the first row of the vertical dataset for node $\mathbf{P_d}$ or the last row for node $\mathbf{D}$, as these variables show a consistent decrease and increase in parameter value as time progresses, respectively. The resulting horizontal dataset, where every flight was represented by one record, was subsequently subjected to equal-frequency discretization across three bins. In Figure 9, the distribution of the total time spent by fueling trucks in position at the aircraft stand is visualized and accompanying bin edges obtained from equal-frequency discretization are indicated with dotted lines. From this, the last two states, [1079, 1374) and [1374, inf), were selected to represent the total distribution when the process has not commenced, facilitating room for any intermediate states in the interval [1, 1079).

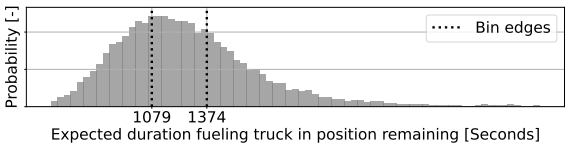

Figure 9: Distribution of the total time spent by fueling trucks in position during the turnaround, and accompanying bin edges.

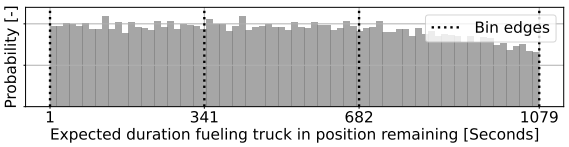

Figure 10: Distribution of the time remaining that the fueling truck will be in position while the process is ongoing.

The intermediate states in the interval [1, 1079) were obtained as follows. Figure 10 presents the distribution of the time that the fueling truck will remain in position during the rest of the turnaround, of the instances where the remaining time of the fueling truck in position was in the interval [1, 1079). To obtain the set of intermediate states that the node can adopt while the process is ongoing, the distribution was subjected to equal frequency binning across three bins. The discretization set for node $\mathbf{P_d}$ in the fueling sub-process network therefore consisted of the intervals [0, 1), [1, 341), [341, 682), [682, 1079), [1079, 1374) and [1374, inf). To illustrate the resulting temporal behaviour with this set of mutually exclusive states of node $\mathbf{P_d}$ within the aircraft fueling general sub-network structure, a visual example is provided in Figure 11.

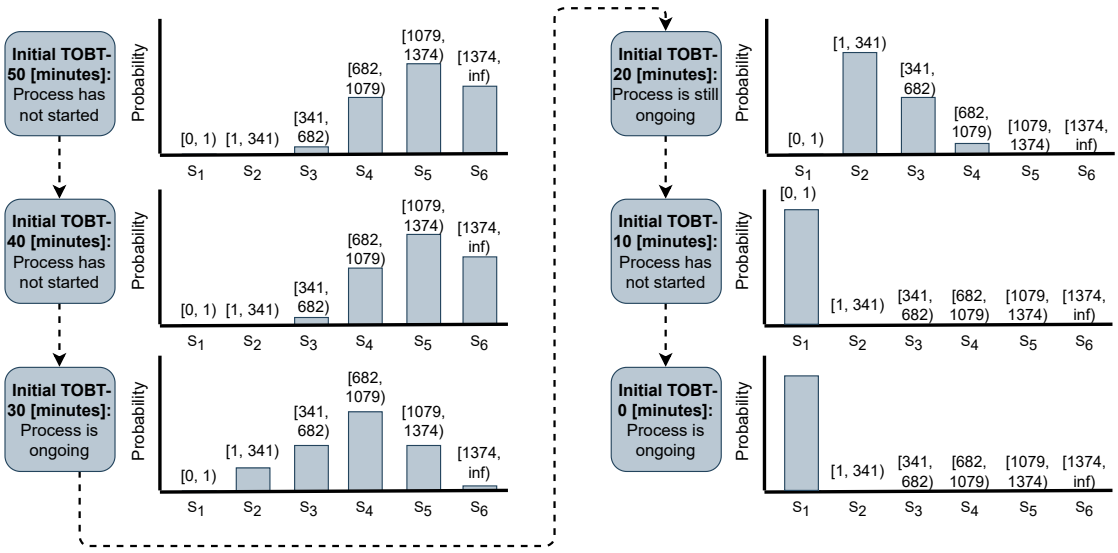

Figure 11: Example of the temporal behaviour of node $\mathbf{P_d}$ within the fuelling process sub-network structure, describing the expected total time remaining in the fuelling process. Units in seconds.

3.2.3 Feature engineering

Feature engineering involves creating new features from the available data and selecting relevant features to be used as variables in the Bayesian network model. First, for the sub-process networks, two variables with strong associations with the ground-handling process were selected, forming the set of context nodes $\mathbf{X}$ for this sub-process. Here, the number of context nodes was limited to two. This reduces computational complexity, increases model interpretability and requires fewer flights to accurately learn the model's parameters [11, 29]. Subsequently, this set of variables was verified with experts in a structured interview, guided by knowledge elicitation functions and probes from Shadbolt and Smart [46]. If the expert interview revealed additional associations with the duration of the sub-process than defined previously, then the set $\mathbf{X}$ was expanded accordingly. For the nodes, $\mathbf{D}$, $\mathbf{P_d}$ and $\mathbf{M}$, a turnaround detection for the particular sub-process was selected that has a clear meaning in the turnaround context and marked the beginning and end of a process. As discussed previously, for the central network structure, context nodes were identified and evaluated that provided crucial context to the ground-handling delay node, where crucial context is defined as nodes that differentiate turnarounds from one another.

3.2.4 Model Training & Analysis

After feature engineering, the dataset was reduced to the prediction moments and the model was constructed and evaluated using pgmpy, a Python package for probabilistic modelling [2]. From the Bayesian network, the marginal distribution of the ground-handling delay was obtained. This distribution describes the probabilities associated with the different delay classes. To convert this marginal distribution to a single point estimate, we employed Maximum A Posteriori (MAP) predictions. With MAP predictions, the most probable state is always selected from the marginal distribution, even when the differences between the classes are small. Model results were obtained using stratified K-cross validation, where the dataset was split into 5 folds. This choice aligns with the findings of Marcot and Hanea [28], who found that $k = 5$ generally suffices for large samples ($n > 5000$).

Stratified K-cross validation offers a statistically more robust estimate of the model performance relative to a training, validation and test split as it allows for assessing the robustness of the model. In stratified K-cross validation, the underlying class distribution of the strata ground-handling delay was preserved in every fold. To prevent non-i.d.d. data group leakage, i.e. the instance where flight information is spread over multiple folds,

the folds were created on a horizontal dataset where each flight was represented as one record. After the flights were randomly split over the folds per strata, the vertical dataset concerning the particular flights in a fold was retrieved. Subsequently, the model was trained on $k-1$ folds and evaluated on the remaining fold. This process was repeated for the remaining folds until all folds were used in the training set $k-1$ times and once as a test set. Subsequently, performance metrics were averaged yielding the results presented in section 5.

4 Model Description

This section elaborates on the model structures resulting from the implementation of the conceptual framework. The Direct Acyclic Graph (DAG) structure is not elaborated on as a whole but rather discussed separately for better readability. First, the central network structure is discussed in section 4.1. Next, the modular sub-networks for the passenger boarding and the baggage loading and unloading processes are elaborated on. These are discussed in section 4.2 and section 4.3, respectively. The rest of the modular sub-networks are presented in Supporting Work B.1. For the figures in this section, the accompanying legend is presented in Figure 12, to help distinguish between regular, network structures and evidence (input) nodes in the network.

Figure 12: Legend to distinguish between regular nodes, network structures and evidence (input) nodes.

4.1 Central network structure

The central network structure of the Bayesian network model is presented in Figure 13, and was composed of 3 nodes and 5 network structures. In the figure, the modular sub-networks of the processes within the research scope are represented by their target nodes. All network structures were directly connected to the ground-handling delay node, aligning with the conceptual framework described in section 3.1.2. Two nodes served as input to the central network structure, with the *time to the initial TOBT* node providing the essential temporal context to the ground handling delay node. Due to modelling approach limitations, discussed in section 6.1, the initial total turnaround time served as the sole differentiating static element between turnarounds.

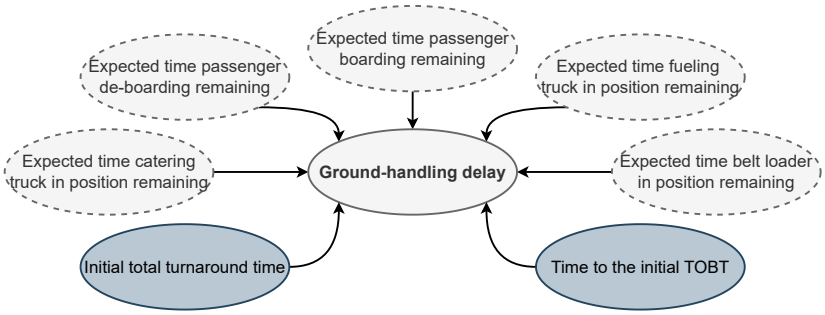

Figure 13: Central network structure of the Bayesian network model.

4.2 Passenger boarding sub-network structure

The modular sub-network in Figure 14 represented the passenger boarding process and follows the general sub-network structure proposed in section 3.1.2. The network composed 6 nodes, three of which belong to the set of context nodes **X**: (1) *expected total outbound passengers*, (2) *aircraft type* and (3) *destination type*. The latter served as a proxy for the longer boarding times caused by the increase in hand luggage, especially trolleys [38, 42]. The *current total boarding time* node served as a measure for the progress in the passenger boarding process, node **D**. The boarding status formed the milestone node **M**, and was discretized into three states: (1) not started, (2) in progress and (3) no passenger in the last 5 minutes. The latter state was assigned to the node when no passengers passed the people counter within the previous five minutes, as the true end of passenger boarding is unknown when only the sensor's data is used.

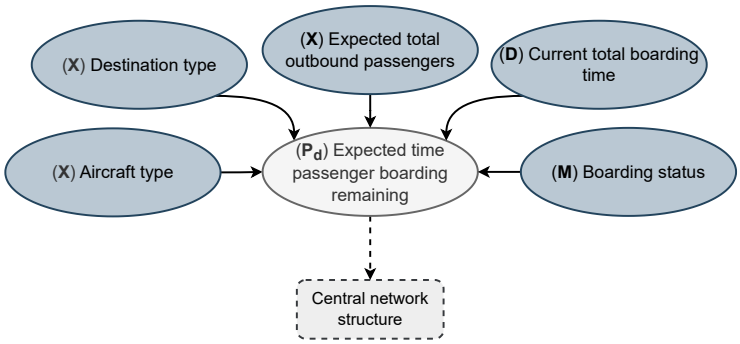

Figure 14: Bayesian sub-network of the passenger boarding process. Letters indicate the type of node.

4.3 Baggage loading and unloading sub-network structure

The sub-network structure for the baggage loading and unloading process is presented in Figure 15, and significantly differs from the general sub-network structure proposed in Figure 5. This is because the baggage loading and unloading process can be executed in various ways. For example, a handler might use two belt loaders to (un)load baggage and leave them at the stand for the entire turnaround. The handler can also use one belt loader to first unload the front and rear cargo hold, subsequently move the belt loader to another stand, and at a later moment in time return to load the baggage for the outbound flight. Any other possible combination of previous examples is also possible. The general sub-network structure and associated discretization were found unable to handle this variability and associated complexity in process execution. Therefore, the sub-network structure was split into two connected components: (1) a component representing the expected time baggage (un)loading remaining and (2) a component representing the expected time remaining that the belt loader will remain in position during the turnaround. Within the two components, elements of the general sub-network structure can be identified.

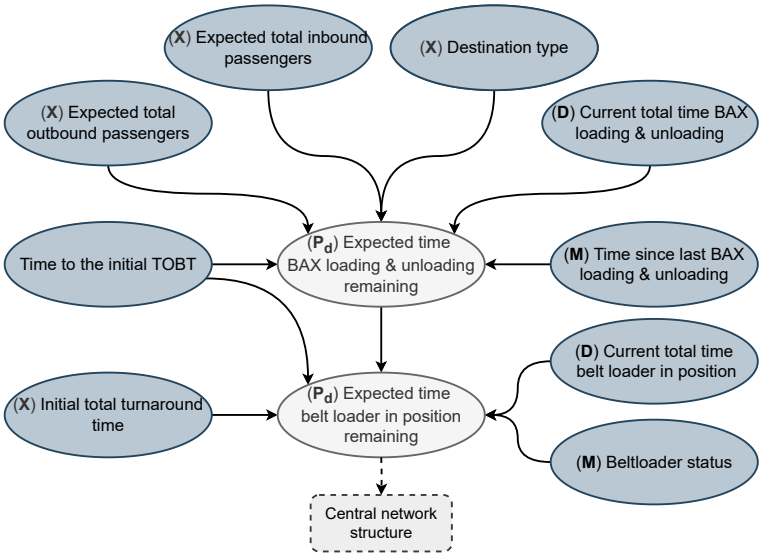

Figure 15: Bayesian sub-network of the baggage loading & unloading process. Letters indicate the type of node.

The first component computed the expected time that baggage (BAX) loading or unloading would occur during the rest of the ground-handling process, based on the number of inbound and outbound passengers, type of destination and the current observed time of baggage loading and unloading. For this component, the time since the baggage loading and unloading process last came to a halt was used as the milestone node, as there are no observable indicators that baggage loading and unloading is finished other than the closure of the cargo doors. However, cargo doors are also closed when the ground crew leaves the stand, as the aircraft may not be left unattended. Therefore, no unique turnaround event could be selected as a milestone node, and the time since the last time baggage loading and unloading came to a halt was used. As this event generally came to a true halt twice, once after baggage unloading and once after baggage loading, the network structure required additional context. After all, a significant amount of time between the last baggage loading and unloading event and the current time grid element can be associated with both the unloading and loading of baggage

being finished. This additional context was provided by the *time to initial TOBT* node, which helped the model to distinguish between the two.

The expected time remaining for baggage loading or unloading propagated to the second component. In this component, the belt loader status represented the milestone node. This node indicated whether a belt loader was not yet observed in position during the turnaround, was in position or has been observed in position and moved out of position. In the network structure, the *time to initial TOBT* was also used, serving as a measure of context, as a belt loader that moved into and out of position did not necessarily indicate that the process was finished. Therefore, the same additional temporal context as for the first component was provided to this node.

5 Results

This section elaborates on the model's ability to model ground-handling delays and the model's sensitivity to changes in discretization. The model's effectiveness in predicting ground-handling delays was assessed using a set of Key Performance Indicators (KPIs). First, the model's accuracy is elaborated on in section 5.1, and simply describes the share of correct predictions made by the model out of all predictions. In this section, the model's accuracy is also compared to the handler's accuracy, to determine the model's performance relative to the handler. However, while accuracy provides a simple measure of the overall correctness of the model, precision, recall and the F1-score offer a more nuanced insight into the model's performance. These metrics are discussed in section 5.2. Precision defines the model's ability to identify instances of a particular class correctly. Recall then refers to the fraction of instances in a class that the model correctly classified out of all instances in that class. The F1-score forms the harmonic mean of precision and recall and penalizes classifiers favouring one of the two metrics. In section 5.3, the confusion matrices over the prediction moments are presented to help understand the errors the model is making, as it summarises the model's classification results. Lastly, the model's sensitivity to changes in the discretization technique and number of bins was also assessed. The results of these interventions are elaborated on in section 5.4.

5.1 Accuracy & Predictions Over Time

The predictive accuracy of the model over the prediction horizon between TOBT-50 and TOBT-0 [minutes] is visualised in Figure 16. The figure shows that, in general, the predictive accuracy of the model increased as time progressed to a maximum of 65.76% at TOBT-0 [minutes]. This increasing behaviour and maximum at TOBT-0 [minutes] was expected, as the onset of ground-handling delays was not immediate. Rather, the ground-handling delay(s) manifested at some point of time within the turnaround process in one or more of the ground-handling processes. Consequently, the model's accuracy improved as more delays manifested themselves. Given the expected upward trend of model accuracy over the prediction horizon as more delays manifest themselves, the minimum in model accuracy at initial TOBT-40 instead of initial TOBT-50 [minutes] forms a discrepancy in the accuracy metric.

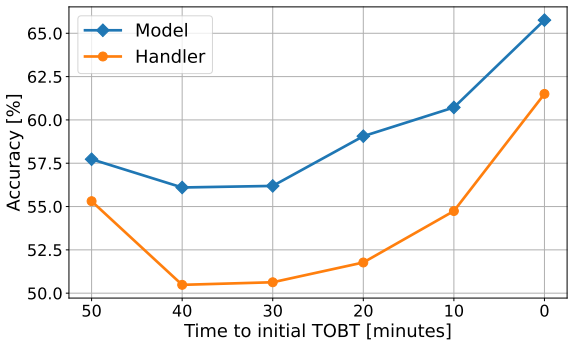

Figure 16: Predictive accuracy for model and handler.

The discrepancy in accuracy can be largely explained by the variance in the initial total available turnaround time between flights. Due to schedule differences, previous delays or other circumstances, the initial total available turnaround time varies, as Figure 17 suggests. Flights with a longer initial total turnaround time available can more effectively absorb delays [22]. Consequently, ground-handling operations for these flights are anticipated to be completed on time more frequently. Flights with an initial total turnaround time shorter than 50 [minutes] are not uncommon and were therefore accounted for in the model. However, these flights

were not included in model training and evaluation for the prediction instance initial TOBT-50 [minutes]. This created an implicit selection of flights at TOBT-50 [minutes], where only flights with an initial total available turnaround time of 50 minutes or more were included.

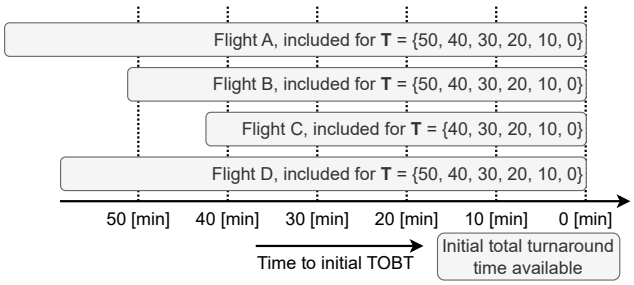

Figure 17: Implicit flight selection in training & evaluation folds. As flight C has a shorter initial total turnaround time than 50 minutes, it is not included in training & evaluation for TOBT-50 [minutes].

Given that ground-handling operations with longer initial total turnaround times tend to adhere to the initial TOBT more frequently and that the onset of ground-handling delays was not immediate, it was expected that for a prediction moment early in the ground-handling process, the model would predict timely turnarounds more often in comparison to a prediction moment later in the turnaround. Figure 18, visualizing the actual and predicted ground-handling delay classes over the prediction moments, confirms this hypothesis. The figure shows that initially, the model predicted that around 85% of the turnarounds were expected to finish in a timely manner. Over time, this share decreased as the turnaround progressed and ground-handling delays manifested. Consequently, the model increasingly predicted ground-handling delays. This is most evident for class [15, inf) in Figure 18, where from the initial TOBT-30 to the initial TOBT-0 [minutes] the share of predictions of severe ground-handling delays significantly increased.

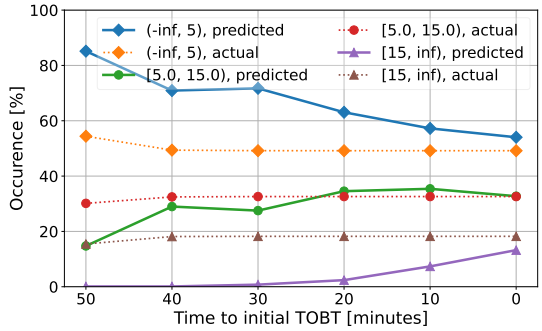

Figure 18: Actual and predicted ground-handling delay classes over time.

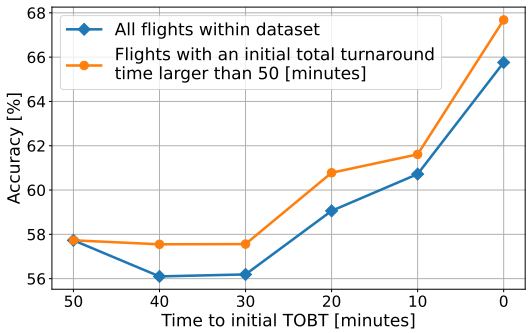

Figure 19: Predictive accuracy of the model for flights with an initial total turnaround time equal to or larger than 50 [minutes] and all flights within the dataset.

To test the hypothesis that the decrease in model accuracy between TOBT-50 and TOBT-40 [minutes] can be largely associated with an implicit selection of flights, we conducted an experiment where we eliminated all flights with an initial total turnaround time shorter than 50 minutes, e.g. flight C in Figure 17, from the evaluation folds. Figure 19 visualizes the results of this intervention and shows that the initial decrease in model accuracy between TOBT-50 and TOBT-40 [minutes] decreased. The implicit selection of flights therefore largely explains the discrepancy in accuracy at TOBT-40 [minutes]. However, a decrease of 0.18% in accuracy between TOBT-50 and TOBT-40 [minutes] can still be observed, amounting to roughly 6 flights per fold. Next to this, an increase in overall accuracy was observed after the intervention. This indicates that the model was less effective in predicting ground-handling delays for flights with a shorter initial total turnaround time as, from the moment flights with initial total turnaround times shorter than 50 minutes were included, overall model performance decreased. This was likely due to the limited presence of these flights within the dataset, as only 1 in 6 flights had an initial total available turnaround time shorter than 50 [minutes].

5.2 Precision, Recall & F1-Score

This subsection presents the model's evaluation results on the precision, recall and F1-score KPIs introduced in the introduction of this section. An overview of the results is presented in Table 1, and are discussed in the next paragraphs.

Table 1: Performance overview of the precision, recall and F1-score metrics. With ±, one standard deviation from the mean is indicated.

Time to initial TOBT [minutes]	(-inf, 5)			[5, 15)			[15, inf)		
	Precision [%]	Recall [%]	F1-Score	Precision [%]	Recall [%]	F1-Score	Precision [%]	Recall [%]	F1-Score
50	59.5 ± 1.0	93.1 ± 0.9	0.73 ± 0.007	48.0 ± 2.6	23.4 ± 0.9	0.31 ± 0.011	8.6 ± 12.8	0.1 ± 0.2	0.0 ± 0.004
40	60.1 ± 0.9	86.3 ± 0.7	0.71 ± 0.006	46.4 ± 1.1	41.4 ± 1.2	0.44 ± 0.01	38.1 ± 42.2	0.3 ± 0.4	0.01 ± 0.007
30	59.9 ± 1.2	87.3 ± 0.7	0.71 ± 0.008	46.7 ± 1.2	39.5 ± 2.5	0.43 ± 0.02	50.5 ± 9.8	2.1 ± 0.4	0.04 ± 0.008
20	65.5 ± 1.0	84.0 ± 0.9	0.74 ± 0.008	47.6 ± 1.0	50.4 ± 1.3	0.49 ± 0.01	53.9 ± 3.6	7.1 ± 0.8	0.13 ± 0.012
10	67.9 ± 0.7	79.0 ± 0.9	0.73 ± 0.002	48.1 ± 0.7	52.2 ± 0.9	0.50 ± 0.004	65.9 ± 3.6	26.6 ± 1.1	0.38 ± 0.018
0	75.2 ± 0.6	82.7 ± 0.8	0.79 ± 0.006	53.5 ± 1.0	53.8 ± 1.2	0.54 ± 0.01	57.3 ± 3.5	41.6 ± 2.9	0.48 ± 0.031

For timely turnarounds, i.e. ground-delay class (-inf, 5), precision increased from 59.5% to 75.2% between the initial TOBT-50 and the initial TOBT-0 [minutes]. This indicates that the model became more accurate in predicting timely turnarounds as time progressed. However, the average recall of the class decreased over time, indicating that the model's ability to identify timely turnarounds decreased. A minimum average recall of 79.0% was achieved at TOBT-10 [minutes], while a maximum average recall of 93.1% was obtained at TOBT-50 [minutes]. A maximum in recall at TOBT-50 [minutes] was not unexpected, as on average roughly 85% of the model's predictions at this prediction moment were timely turnarounds, as Figure 18 indicated. In general, the F1-score for (-inf, 5) showed a positive trend, indicating that the overall model performance improved as time progressed.

The model's precision for small ground-handling delays, i.e. the interval [5, 15), showed a slight upward trend as time progressed ranging from 46.4% to 53.5% between the initial TOBT-40 and the initial TOBT-0 [minutes]. Next to this, the model already identified 23.4% of small ground-handling delays at the initial TOBT-50 [minutes]. This recall metric for this class significantly improved with time, ranging from 23.4% at the initial TOBT-50 to 53.8% at the initial TOBT-0 [minutes] indicating that the model was able to detect more small ground-handling delays as they manifested. As a result, the average F1-score score for the ground-handling delay class [5, 15) showed an upward trend ranging from 0.31 [-] to 0.54 [-] between the initial TOBT-50 and initial TOBT-0 [minutes].

The average precision for severe ground-handling delays, i.e. ground-handling delays in the interval [15, inf), also showed an upward trend as time progressed. At TOBT-50 [minutes], an average minimum precision of 8.6% was achieved, while at TOBT-0 [minutes] a precision of 57.3% was obtained. A peak in precision was achieved at TOBT-10 [minutes], amounting to 65.9%. However, high variability in average precision was observed between the folds for the first prediction moments initial TOBT-50, initial TOBT-40 and initial TOBT-30 [minutes]. This can be attributed to the fact that the model seldom predicted this class for these prediction moments, as Figure 18 indicated. This was also visible in the recall metric, as at the initial TOBT-50 [minutes] an average recall of only 0.1% was achieved. As time progressed and delays manifested themselves, the recall metric improved significantly, reaching 41.6% at the initial TOBT-0 [minutes]. Given the upward trend of both the precision and recall metrics, the F1-score also saw an upward trend ranging from 0.0 [-] at the initial TOBT-50 to 0.48 [-] at the initial TOBT-0 [minutes], indicating that overall model performance increased as time progressed.

To further explore the findings that the model outperformed the handler on average accuracy in section 5.1, the precision and recall metrics were also evaluated for the handler's predictions on the ground-handling delay. For this, the TOBT at the prediction moment was considered as their prediction. An overview of the model's and handler's performance is presented in Table 2. The handler's precision for timely turnarounds, (-inf, 5), was on par with the model for the first prediction moment at the initial TOBT-50 [minutes] after which the model started outperforming the handler. For the small and severe ground-handling delay classes, the handler's precision exceeded the model's precision. However, in the identification of small and severe delays, the model significantly outperformed the handler. This is especially pronounced at the initial TOBT-20 [minutes], where for small ground-handling delays the model demonstrated an average recall of 50.4%. In comparison, the handler's average recall for this prediction moment only amounted to 4.3%. The handler's high recall for timely turnarounds and low recall for ground-handling delays suggest that the handler struggled to identify ground-handling delays, and as a result, the handler's predictions were rather conservative, often predicting few to no delay minutes. These results proved that the model could detect delays earlier and was better equipped to detect ground-handling delays as they manifested themselves than the handler.

Table 2: Performance overview of the average precision and recall of the model and handler.

Ground-Handling Delay	Metric	Model/handler	Time to initial TOBT [minutes]					
			50	**40**	**30**	**20**	**10**	**0**
(-inf, 5)	Precision [%]	Model	59.5	60.1	59.9	65.5	67.9	75.2
		Handler	55.2	50.2	50.3	51.3	54.2	61.7
	Recall [%]	Model	93.1	86.3	87.3	84.0	79.0	82.7
		Handler	99.7	99.6	99.5	99.1	97.8	94.3
[5, 15)	Precision [%]	Model	48.0	46.4	46.7	47.6	48.1	53.5
		Handler	45.6	49.6	48.2	51.7	53.1	57.0
	Recall [%]	Model	23.4	41.4	39.5	50.4	52.2	53.8
		Handler	0.9	1.2	1.7	4.3	13.2	34.2
[15, inf)	Precision [%]	Model	8.6	38.1	50.5	53.9	65.9	57.3
		Handler	75.2	74.3	75.1	74.9	74.9	75.4
	Recall [%]	Model	0.1	0.3	2.1	7.1	26.6	41.6
		Handler	5.1	5.0	6.4	8.8	13	21.9

5.3 Confusion Matrices

This subsection presents the confusion matrices for the prediction moments considered in this research. The entries in a confusion matrix represent the proportion [%] of predictions made for a specific actual, realised state. The confusion matrices for time instances initial TOBT-40, initial TOBT-20 and initial TOBT-0 [minutes] are presented in Table 3, 4 and 5 respectively. An overview of all confusion matrices is presented in Supporting Work B.3.

At the initial TOBT-40 [minutes], the model correctly predicted timely ground-handling execution in 86.3% of cases where ground-handling was executed on time. The model made most errors by predicting small ground-handling delays. For ground-handling delays within 5 to 15 minutes, the model expected a timely ground-handling execution in 58.5% of cases. In only 0.1% of cases, the model expected a severe ground-handling delay. Therefore the model correctly identified 41.4% of cases. For severe ground-handling delays, it is interesting to note that the model in 48.3% of cases already expected a ground-handling delay between 5 and 15 minutes at this particular time instance, and less often predicted a timely turnaround in comparison to class [5, 15). However, only 0.3% of flights with actual ground-handling delays longer than 15 minutes were correctly identified, indicating the model struggled to identify the severe ground-handling delays 50 minutes before the initial TOBT, with a substantial portion being misclassified as delays between 5 and 15 minutes.

Table 3: Confusion matrix for the initial TOBT-40 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	86.3	13.7	0.0
	[5, 15)	58.5	41.4	0.1
	[15, inf)	51.4	48.3	0.3

At the initial TOBT-20 [minutes], a small increase in errors made for timely turnarounds was observed as instead, the share of predictions in [5, 15) increased. However, for small and severe ground-handling delays, the model's performance increased in comparison to earlier prediction instances. Table 4 shows that in comparison to TOBT-40 [minutes], the distribution of predictions for cases of small and severe ground-handling delays shifted towards the right, indicating that more ground-handling delays were expected by the model. This reduced the number of errors made, as less often timely turnarounds were predicted. For severe ground-handling delays larger than 15 minutes, in 64.2% of cases an untimely turnaround was expected. Here, in 57.1% of turnarounds, the model expected a ground-handling delay within 5 to 15 minutes and in only 7.1% of flights the model expected a ground-handling delay larger than 15 minutes. This finding expresses that the model was rather optimistic about the future progress of the turnaround process, often underestimating the severity of the delays, and therefore struggled to accurately identify these delays at this prediction moment.

Table 4: Confusion matrix for the initial TOBT-20 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	84.0	15.7	0.3
	[5, 15)	46.6	50.4	3.0
	[15, inf)	35.8	57.1	7.1

Providing interpretation to the results in Table 5, i.e. the confusion matrix for TOBT-0 [minutes], we observe that for timely turnarounds the model makes the most errors by predicting a small ground-handling

delay, amounting to 16.1%, which is a 0.4% increase in errors in comparison to the initial TOBT-20 [minutes]. In 1.2% of predictions, the model predicted a severe ground-handling delay. For delays within 5 to 15 minutes, the model made most errors by predicting a timely turnaround, amounting to 30.8% of cases. In 15.4% of cases, a severe ground-handling delay was expected by the model. For severe ground-handling delays, we observe that in 81.7% of cases of severe ground-handling delays, the model predicted that ground-handling for the specific flight would not finish within the TOBT window. This was split between 40.1% of predictions in the interval [5, 15) and 41.6% of predictions in the interval [15, inf). In 18.3% of cases, the model predicted a timely turnaround. From this, it can be concluded that the model identified initial TOBT non-compliance well, although it struggled to correctly identify cases of severe ground-handling end time non-compliance.

Table 5: Confusion matrix for the initial TOBT-0 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	82.7	16.1	1.2
	[5, 15)	30.8	53.8	15.4
	[15, inf)	18.3	40.1	41.6

Summarising the general trend in the confusion matrices, we observe that most prediction errors are made by predicting timely turnarounds for prediction moments early in the ground-handling process. This suggests that the model was often overly optimistic about the future execution of the ground-handling process. As time progressed, the model more frequently predicted small or severe ground-handling delays. We observe that, over time, the share of predictions for a particular actual ground-handling delay class shifted towards the true state, indicating that the model adjusted its predictions as more information on the ground-handling process came in. As a result, the model expected more ground-handling delays. However, we also observe that the model tended to underestimate delays, as most errors made were found in the column left to the respective ground-handling class. The imbalance in training data could explain this conservative behaviour, as the model might have learned to lean towards predicting no or small ground-handling delays more frequently as severe ground-handling delays formed a minority class within the dataset. Next to this, the available data might have not supported the model to learn the model's parameters for this class accurately. However, on the other hand, the delay(s) could have not been fully manifested yet, leading to the rather optimistic behaviour of the model about the future progress of the ground-handling process.

5.4 Sensitivity Analysis

The discretization of continuous input data into a set of mutually exclusive states effectively forms a hyperparameter of a discrete Bayesian network model. Hence, changes to the discretization are expected to result in changes in model performance. Within the central network structure, the nodes $\mathbf{P_d}$ exert the most influence on the ground-handling delay prediction. To assess the model's sensitivity and robustness to changes in discretization, the states for nodes $\mathbf{P_d}$ were altered, where the bin edges were changed and the number of bins varied in various manners. Accordingly, the discretization of nodes $\mathbf{D}$ were also subjected to the interventions. These interventions are discussed in section 5.4.1 and 5.4.2, respectively.

5.4.1 K-Means & rounding

To assess the model's robustness to changes in bin edges, two experiments were conducted, where the number of bins of all nodes remained constant. In the first experiment, K-means discretization was applied to the two-step discretization procedure previously elaborated on in section 3.2.2. In the second experiment, the set of mutually exclusive states of nodes $\mathbf{P_d}$ and $\mathbf{D}$ were rounded to the nearest multiple of 5 [minutes]. From a model interpretability perspective, one might prefer this approach, as it enhances the readability of the intervals. When two intervals were within the same 5 [minutes] interval, one was rounded up towards the nearest instance of 5 [minutes]. As an example, for the aircraft fuelling process, the set of bin edges changed from [0, 1, 341, 682, 1079, 1374, inf] to [0, 1, 300, 600, 1200, 1500, inf].

The results of both interventions are presented in Supporting Work B.4. After evaluating the model on the revised discretization sets, no significant changes to the KPIs were observed, except for the increase in precision for [15, inf) at TOBT-50 and TOBT-40 [minutes]. However, it was already found that this metric was highly variable and as the model seldom predicts this class at these prediction moments, it is not elaborated on further. Next to this, a small increase in model accuracy was observed for the prediction moments initial TOBT-50, initial TOBT-40 and initial TOBT-30 [minutes], while a small decrease in model performance was observed for the other prediction moments. This indicates that the model was rather insensitive to small changes to the discretization set, indicating the robustness of the model.

5.4.2 Number of bins

To evaluate the model's ability to a finer granularity in discrete states, the number of bins used in the two-step discretization was also altered. This allows for a more detailed representation of the nodes, enabling the model to capture more subtle relationships between variables. However, the higher resolution comes at the cost of reduced interpretability, increased model complexity and computational cost. A high number of discrete states may also lead to model overfitting, reducing the generalization ability of the model on unseen data. Next to this, this may also contribute to data sparsity, as more data is required to accurately estimate the probabilities in the CPTs to support the finer resolution. To assess the model's ability to deal with a finer resolution in discrete states, four experiments were conducted expanding the granularity in various areas within the two-step discretization of nodes $\mathbf{P_d}$ and $\mathbf{D}$:

- **T1**: The addition of one bin to the number of bins in the total distribution.
- **T2**: The addition of two bins to the number of bins in the total distribution.
- **P1**: The addition of one bin to the number of bins representing the intermediate states.
- **T1P1**: The addition of one bin to the number of bins representing the total distribution and one bin representing the intermediate states.

The results of the experiments are presented in Supporting Work B.4. Overall, no large changes in model performance were observed, indicating that the initial binning may already have captured the relevant information in the data well. This led to negligible performance improvements with an increasing number of bins. The finer resolution in discrete states seemed to benefit the accuracy metric in some prediction instances. This was especially evident at the initial TOBT-20 and initial TOBT-10 [minutes], and can be attributed to the increase in precision and recall for ground-handling delays (-inf, 5) and [5, 15). However, an overall loss in precision was observed for the delay class [15, inf). This can be explained by data sparsity, as [15, inf) forms a minority class within the dataset. With increasing bins, the data becomes more segmented. This makes it more difficult to accurately learn the model's parameters associated with the minority class within the same dataset. The more sparse binning may therefore lead to a reduced model performance for the minority classes, as the model struggles to find patterns tied to the severe ground-handling delay. As a result, the model may be more biased towards the more dominant classes, explaining the overall increase in recall for classes (-inf, 5) and [5, 15).

6 Discussion

This section discusses the model, modelling approach and results presented in this paper. First, we provide a discussion on the structure's strengths and limitations in section 6.1. Second, we compare the modelling approach to the state-of-the-art in the field in section 6.2. Third, we assess the model's ability to provide insights into the turnaround system in section 6.3. Next, we address data limitations in this study in section 6.4. Finally, we conclude this section with a discussion of the broader implications of the study, identifying possible use cases and philosophising about how the model can contribute to the increase in robustness and efficiency of the air traffic network in section 6.5.

6.1 Model structure & results

The Bayesian network model as presented in this paper proved effective for modelling ground-handling operations at AAS for the particular aircraft operator and demonstrated a higher predictive accuracy than the handler over various KPIs. However, in its predictions, the model tended to be conservative in predicting ground-handling delays, often predicting fewer delay minutes. Moreover, the model proved to be relatively insensitive to changes in discretization. This insensitivity indicates high robustness of the modelling approach proposed in this paper, as the model remained consistent with changes in input data, and suggests that the modelling approach is not overly dependent on specific parameter settings.

The general modular sub-network structure and associated discretization proved to be adequate in describing most main ground-handling processes for narrow-body aircraft. However, it was found unable to adequately model processes that start and come to a halt more than once, such as the baggage loading and unloading process, forming a limitation of the general sub-network structure. This limits the model's ability to adequately model these processes. To incorporate multiple start-stop cycles, a new general sub-network structure for these processes is required that can deal with the associated complexity of operations.

The Bayesian network structure does not model the critical path directly. Rather, the expected time remaining in the various processes is propagated to the ground-handling delay node, which estimates the likelihood of obtaining a certain ground-handling delay. The ground-handling delay node implicitly models the critical path and in model training learns patterns that are more likely to lead to delays. This approach was considered beneficial, as the strict sequential dependencies are not as strict in practice. For example, the exchange of

catering trolleys in the aft aircraft galley does not necessarily have to start after passenger disembarking has finished, despite what Figure 2 suggests.

The network structure allows the modelling of ground-handling operations for narrow-body aircraft at various aircraft stands and airports. To tailor the model to other types of aircraft stands, a change to the set of turnaround events used in the model and the associated data is required for the passenger processes. However, one can still utilize the same modular sub-network structures with small modifications, such as the inclusion of the number of available aircraft stairs for remote handling, providing additional context to the model over the expected boarding and de-boarding duration.

The model structure implicitly assumes that all turnaround processes are executed, which is not always the case. This is especially true for airlines operating narrow-body aircraft without a base at AAS, as they often only clean the aircraft cabin and exchange catering trolleys at their hub bases. Next to this, Low-Cost Carriers (LCCs) generally also do not execute all turnaround processes in every turnaround. The model structure insufficiently accounts for these aircraft operators, and therefore an adaption to the network structure is required to accommodate other airlines.

Next to this, the modelling approach also suffers from limitations. During model development, the variable *season* was added to the central network structure to represent periods of the year where adverse weather conditions are more common and to account for the number of daily flights, e.g. via the IATA summer/winter schedules. When implemented, a decrease in overall model performance was observed, despite expert verification of the variable's relevance. This can be partially attributed to the increased complexity of adding more parents to the ground-handling delay node. Expanding the number of parents also contributes to data sparsity, as the network requires sufficient data to accurately estimate the conditional probabilities for every combination of parents, which could lead to decreased model performance if the amount of supporting data is insufficient. The variable *season* was therefore removed from the model and the set of context nodes for the central network structure was reduced to the *initial total available turnaround time* node. To accommodate additional distinguishing factors between turnarounds, changes to the central network structure are required. Potential solutions include factor aggregation or defining new nodes which summarise some aspects, limiting the number of parents of the ground-handling delay node and enhancing the model's ability to deal with additional factors of influence.

6.2 Comparison to the state-of-the-art

The ground-handling delay modelling approach proposed in this paper is inherently interpretable, providing insights into exactly how predictions came about. The network representation of the DAG facilitates model interpretability, expressing qualitative relationships of direct dependencies between variables forming a stable element of the model [37]. The networks can be built without explicit modelling skills, and can easily understood by non-technical users [11]. As a result, the underlying causal relationships were easily verified with experts in the field. The DAG also facilitates the incorporation of expert knowledge into the structure. This makes the modelling approach very adaptable, as one can tailor the DAG structures and associated discretization to obtain a desired behaviour. Next to this, Bayesian networks also explicitly express uncertainties, providing a measure of the model's confidence in a prediction, a characteristic that most black-box machine learning models lack [11]. The aforementioned characteristics form a significant advantage to modelling approaches using black-box machine learning algorithms, such as those employed by Luo et al. [27] and Schiphol Group Aviation Solutions [40]. However, generally, black-box models can capture more complex relationships between variables, leading to more accurate predictions.

In comparison to the work of Luo et al. [27], the modelling approach also facilitates updates of the current belief of TOBT adherence at various prediction moments during the turnaround, rather than at one prediction moment in the ground-handling process. This characteristic, which it shares with PEGT [40], allows for more accurate and timely decision-making, facilitating the increase in robustness of the air traffic network. However, PEGT is also able to deal with TOBT updates from the handler or aircraft operator during the ground-handling process, a characteristic that the modelling approach we present currently does not facilitate.

6.3 Obtaining insights into the turnaround system

The modelling approach presented in this paper facilitates various methods for obtaining insights into the turnaround system. These are discussed in this subsection.

The output of the discrete Bayesian network describes the probability of occurrence of the ground-handling delay classes, given the observed variables, which are affected by the information propagated by the nodes $\mathbf{P_d}$. These are in turn affected by the evidence nodes in the particular modular sub-networks. To gain insights into the turnaround system, one can adjust the input states of the evidence nodes within a particular sub-network and observe how they interact with the ground-handling delay distribution. As an example, this allows one to obtain insights into how the number of outbound passengers affects ground-handling delays and investigate

conditions that are more likely to lead to ground-handling delays. Next to this, one can also expand the set of evidence nodes. The set can be expanded with the expected time remaining in the particular process, $\mathbf{P_d}$, for one or more processes. This allows one to investigate how the particular process(es) interact with the ground-handling delay distribution when one changes the node to a specific state.

Discrete Bayesian networks also facilitate obtaining the marginal distribution of unobserved nodes. Within the context of the model proposed in this paper, this allows one to compute and gain insights into the distribution of the time remaining in one or more processes. This means that, at every prediction moment, one can compute the distribution of the expected time remaining in the ground-handling process(es) and the ground-handling delay within the same model. This feature also allows one to determine how specific (additional) input nodes interact with the total time remaining in a particular process, and investigate the influence on the ground-handling delay distribution simultaneously.

Backward inference can be employed to compute the marginal distribution of the expected time remaining in one or more ground-handling processes, given that a specific ground-handling delay class is to be achieved. This can help with understanding the turnaround system. For example, backward inference can give insights into the most likely time remaining in a particular process when a timely turnaround is expected. These then give insights into the most likely required remaining time in the various processes times to achieve a nominal turnaround.

In backward inference, the ground-handling delay node becomes an input to the model. This information propagates backwards into the nodes $\mathbf{P_d}$, as there are no blocked paths between nodes $\mathbf{P_d}$ and *ground-handling delay* node. The temporal context propagates along the path *time to the initial TOBT* $\rightarrow$ *ground-handling delay* $\leftarrow \mathbf{P_d}$. Within this path, the *ground-handling delay* node forms a collider node. This means that observing one of its parent nodes does not provide any extra information on the states of the other parent node(s). However, when a collider node becomes observed, unconditional associations between the parent nodes are established. Consider the simple network structure (*Motivation* $\rightarrow$ *Admitted to University*) and (*Grades* $\rightarrow$ *Admitted to University*) based on Morgan and Winship [32]. Observing that an individual with low motivation is admitted to the university establishes unconditional associations between *Motivation* and *Grades*, making them conditionally dependent as individuals with low motivation generally require high grades to be admitted. Similarly, observing the *ground-handling delay* node makes the nodes $\mathbf{P_d}$ and *time to the initial TOBT* conditionally dependent, allowing information to propagate between the nodes.

Analogously, conditional dependencies between the processes are established when the ground-handling delay node becomes observed. Consequently, the most likely marginal distribution of the time remaining in a particular process is also, next to the ground-handling delay node, evidence nodes in the sub-network of the particular process and evidence nodes in the central network structure, influenced by the state of other processes. Consider the end of a delayed turnaround, where all but one turnaround process is finished. Knowing that all other processes have finished and a delay is expected increases the likelihood of a (significant) amount of time remaining in the last process. The conditional dependence between processes therefore provides more precise information on the expected state of the remaining process. However, additional research is required to fully explore which specific insights can be retrieved from utilizing backward inference, and how these types of information can be applied to optimize operational efficiency and decision-making.

6.4 Data limitations

In this study, observational data was used, which always contains imperfections. Due to the inherent nature of computer vision techniques, its detections are not always 100% accurate. With the implementation of business rules, the number of blank detections was reduced. These implicitly assume that the system can detect all ground-handling processes within the study's scope. However, as ground-handling events that can be accurately detected in various operational conditions were selected, the impact of the business rules on the study's results is minimal. Noise within the dataset was retained to mimic the operational conditions of the model.

Not all factors of influence and processes could be included in this study. Cabin cleaning, a ground-handling process often found on the critical path, was not included in this study due to the lack of data on this process. Cabin cleaning takes place within the aircraft cabin and therefore its progress is not observable on the aircraft stand. Next to this, variables with a strong influence on the duration of a particular process were not always available, such as the transfer flows to and from the particular aircraft operator, limiting the model's ability to fully represent the associated complexities to the turnaround system. Moreover, the exact start and end times of the passenger processes were unknown. To filter out ground staff and noise, strict start and end criteria were set up. Consequently, this led to the loss of information on any late passengers.

The modelling approach also placed limitations on the input data. Discrete Bayesian networks require a continuous variable to be discretized into a set of mutually exclusive states in which variations are neglected. This introduces imprecision and vagueness in the model, and leads to a potential loss in statistical accuracy [11, 33]. A larger number of discrete states decreases the size of the intervals, potentially recovering some of the loss in model accuracy, but also decreases model interpretability as the number of discrete states exponentially

increases the size of the CPT [11]. Next to this, a larger number of states may require more data to accurately learn all model parameters or may result in decreased model performance for minority classes.

6.5 Implications of the study

In this subsection, the study's broader implications are elaborated from a stakeholder perspective. This includes discussing the viewpoints of the Air Navigation Service Providers (ANSPs), airports and ground handlers.

At airports with A-CDM, ANSPs rely on the ground handler's TOBTs for optimal departure flow but lack insights into the ground-handling process beyond TOBT updates. They therefore depend on accurate TOBTs for optimal departure planning. In section 5.2, we demonstrated that the modelling approach outperformed the ground handler in predictive accuracy, precision and recall and concluded that the model showed superiority to the handler in identifying ground-handling delays. This was especially pronounced for small ground-handling delays, as 20 [minutes] before the initial TOBT, the model achieved an average recall of 50.4%, while the ground handler's prediction at the particular time instance was correct in only 4.3% of cases. The model therefore significantly outperformed the ground handler in the identification of small initial TOBT non-adherence at TOBT-20 [minutes]. Early detection of TOBT non-compliance allows the ANSP to mitigate the possible disruptive effect of ground-handling delays directly, avoiding later re-planning. This leads to more stable planning and an increase in the overall departure capacity utilization as found by Snijders [49], also benefiting airports and ground handlers.

From the en-route perspective, the modelling approach may also facilitate improvements within Air Traffic Flow Management (ATFM), benefiting from the reduced uncertainty in initial TOBT adherence. As the planning becomes more stable, uncertainties in the departure planning also decrease. Consequently, the uncertainty when flights arrive in particular sectors also decreases, via more accurate predictions of the Actual Take-Off Times (ATOTs). This leads to fewer and more accurate regulations and an overall increase in airspace capacity utilization, while also enhancing safety as the airspace capacity is less often exceeded. For airports without current A-CDM implementation, the buffers around a flight's scheduled departure time can be further decreased when accurate predictions on the progress of the ground-handling process are available, leading to less Calculated Take-Off Time (CTOT) delays for these flights. Next to this, a reduced uncertainty in initial TOBT adherence also opens possibilities for long-range ATFM concepts.

The more stable planning also provides advantages to the ground-handling parties, as it helps with the resource allocation of scarce ground equipment. As an example, push-back planning is currently based on the Target Start-up Approval Time (TSAT), denoting the time the aircraft can expect start-up and push-back approval. The TSAT is a derivative of the TTOT, which in turn is a derivative of the TOBT, EXOT and runway capacity [1]. When push-back resources are scarce, the handler can prioritize turnarounds that are more likely to execute in a timely fashion, using the model's provided prediction uncertainty.

Next to this, modelling ground-handling processes allows ground-handlers and airports to learn more about the turnaround system, and how processes and events interact with the onset of delays. After all, due to the interpretable machine learning approach, the decision-making process can be retrieved.

7 Conclusions & Future Work

This study proposed an interpretable machine learning approach that uses data from automatic turnaround process identification systems to predict ground-handling end time adherence of narrow-body aircraft at various prediction moments during the turnaround. Here, the initial TOBT when the aircraft arrived at the stand was considered as the initial ground-handling end time. The approach considered a discrete Bayesian network model composed of a central network structure and a set of modular sub-network structures. A modular sub-network structure represented a distinct ground-handling process often found on the critical path, and its structure and associated discretization accommodated the various states the process can be in. A general modular sub-network was proposed which was composed of four types of nodes: (1) nodes closely related to the duration of the process, (2) a node representing the measure of progress as a measure of time, (3) a milestone event node and (4) a node representing the time remaining in a particular process. All modular sub-networks were connected to the central network structure, which was composed of the ground-handling delay node, the initial total available turnaround time as a distinguishing factor between turnarounds and a node providing temporal context to the network. This structure processed the information propagated from the sub-networks and computed the model's belief on the initial TOBT adherence.

The proposed general sub-network structure and two-step discretization proved to be adequate in modelling most main ground-handling processes. However, the general sub-network structure was found to be unable to adequately model processes with multiple start-stop cycles, such as the baggage loading & unloading process. A new structure for this process was proposed.

The model was evaluated on various Key Performance Indicators (KPIs) using stratified K-cross validation. The model demonstrated to be more accurate than the ground handler in predicting initial TOBT adherence. As time progressed, model precision increased for all ground-handling delay classes. As a result, the average accuracy rose to a maximum of 65.76% at the initial TOBT-0 [minutes]. The model also showed superiority to the handler in the identification of ground-handling delays, identifying the onset of small ground-handling delays earlier than the ground handler. This was especially pronounced at the initial TOBT-20 [minutes], where the model achieved an average recall of 50.4% for ground-handling delays between 5 and 15 minutes, while the handler's predictions at this prediction moment were only correct in 4.3% of cases. The confusion matrices showed that as time progressed, the model's predictions converged towards the true realised ground-handling delay. However, the model was found to be conservative in predicting ground-handling delays, often predicting fewer delay minutes.

We conclude that the model can facilitate a possible increase in the efficiency and robustness of the air traffic network. The model proved capable of detecting ground-handling delays at an earlier time instance than is the case today and decreases the uncertainty in the ground-handling process progress. Consequently, the ANSPs, aircraft operators, airports and ground handlers can benefit from more stable planning and an increase in overall departure capacity utilization. Next to this, these operational partners can benefit from insights that the modelling approach can facilitate.

While this study demonstrated that the discrete Bayesian network structure was able to adequately model the ground-handling progress, further research into this topic is necessary. Currently, the Bayesian network structure was constructed for narrow-body aircraft for one particular aircraft operator at AAS. Efforts could be devoted to adapting the model structure to accommodate multiple aircraft operators or different types of aircraft stands into one model. As an example, a new type of node can be implemented to distinguish between aircraft operators having a base at a particular airport, as this is generally a distinguishing factor in whether the catering process is executed. Next to this, the model can be expanded to wide-body aircraft and expanded with more processes or factors influencing the ground-handling process.

Moreover, further research is required into the central network structure and modular sub-network structures. For the central network structure, finding ways to increase the number of parents needs to be thought of. This can, for example, be done by aggregating multiple factors into a new node summarizing some aspects. Additionally, the central network structure requires an update to deal with TOBT updates to be implemented in an operational setting. Next to this, the modular sub-networks currently only accommodate processes that start and come to a halt only once, limiting the model's ability to model operational variations within a process. Therefore, an alteration to the general sub-network structure is required. The sub-network structures can also be expanded with information that was not available for this study, such as the number of missing passengers and their status. Additional research is also required to fully explore which specific insights can be retrieved from the model, and how these types of information can be applied to optimize operational efficiency and decision-making.

To conclude, we presented a novel interpretable approach to ground-handling delay modelling using a discrete Bayesian network composed of a central network structure and a set of sub-networks, representing distinct turnaround processes. The model provided transparency into the decision-making process, proved to be more accurate than the handler over all prediction moments and demonstrated superior capabilities to the ground handler in the identification of ground-handling delays. In operation, the model can contribute to more stable operational planning, benefiting the ANSPs, ground handlers and airports.

References

[1] *A-CDM Manual Schiphol Airport*. Royal Schiphol Group. Feb. 2024. URL: https://www.schiphol.nl/nl/download/b2b/1569488978/7ERl8iHeLELDtgFsnK0mGi.pdf.

[2] A. Ankan and J. Textor. *pgmpy: A Python Toolkit for Bayesian Networks*. 2023. arXiv: 2304.08639 [cs.LG].

[3] A.S. Antonio, A.A. Juan, L. Calvet, P. Fonseca i Casas, and D. Guimarans. "Using simulation to estimate critical paths and survival functions in aircraft turnaround processes". In: *2017 Winter Simulation Conference (WSC)*. 2017, pp. 3394–3403. DOI: 10.1109/WSC.2017.8248055.

[4] E. Asadi, J. Evler, H. Preis, and H. Fricke. "Coping with Uncertainties in Predicting the Aircraft Turnaround Time at Airports". In: *Operations Research Proceedings 2019*. Ed. by J.S. Neufeld, U. Buscher, R. Lasch, D. Möst, and J. Schönberger. Cham: Springer International Publishing, Sept. 2020, pp. 773–780. ISBN: 978-3-030-48439-2. DOI: 10.1007/978-3-030-48439-2_94.

[5] Schiphol Nederland B.V. *Crowd management*. https://www.schiphol.nl/en/page/data-processing-crowd-management/. Aug. 2022.

[6] M. Beltman. "Dynamically Forecasting Airline Departure Delay Probability Distributions for Individual Flights using Supervised Learning". MA thesis. Delft University of Technology, Dec. 2023. URL: http://resolver.tudelft.nl/uuid:4fba195a-4627-4121-88cf-ae8f004d01cf.

[7] I. Ben-Gal. "Bayesian Networks". In: *Encyclopedia of Statistics in Quality and Reliability*. John Wiley & Sons, Ltd, 2008. ISBN: 9780470061572. DOI: https://doi-org.tudelft.idm.oclc.org/10.1002/9780470061572.eqr089. eprint: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/pdf/10.1002/9780470061572.eqr089. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1002/9780470061572.eqr089.

[8] K. Blackmond Laskey and S.M. Mahoney. "Network Fragments: Representing Knowledge for Constructing Probabilistic Models". In: *CoRR* abs/1302.1557 (2013). arXiv: 1302.1557. URL: http://arxiv.org/abs/1302.1557.

[9] M.E. Borsuk, C.A. Stow, and K.H. Reckhow. "A Bayesian network of eutrophication models for synthesis, prediction, and uncertainty analysis". In: *Ecological Modelling* 173.2 (2004), pp. 219–239. ISSN: 0304-3800. DOI: https://doi.org/10.1016/j.ecolmodel.2003.08.020. URL: https://www.sciencedirect.com/science/article/pii/S0304380003003958.

[10] E. Charniak. "Bayesian Networks without Tears." In: *AI Magazine* 12.4 (Dec. 1991), p. 50. DOI: 10.1609/aimag.v12i4.918. URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/918.

[11] S.H. Chen and C.A. Pollino. "Good practice in Bayesian network modelling". In: *Environmental Modelling & Software* 37 (2012), pp. 134–145. URL: https://www.sciencedirect.com/science/article/abs/pii/S1364815212001041.

[12] Eurocontrol. *CODA Digest: All-Causes Delays to Air Transport in Europe - Annual 2022*. Tech. rep. Eurocontrol, Mar. 2023. URL: https://www.eurocontrol.int/sites/default/files/2022-10/eurocontrol-coda-digest-q2-2022.pdf.

[13] Eurocontrol. *TOBT - Target Off-Block Time*. 2024. URL: %5Curl%7Bhttps://ansperformance.eu/acronym/tobt/%7D%20accessed%20on%2024/05/2024..

[14] Eurocontrol. *Airport Collaborative Decision-making (A-CDM)*. URL: https://www.eurocontrol.int/concept/airport-collaborative-decision-making.

[15] H. Fricke and M. Schultz. "Delay Impacts onto Turnaround Performance - Optimal Time Buffering for Minimizing Delay Propagation". In: *USA/Europe Air Traffic Management Research and Development Seminar*. June 2009. URL: https://api.semanticscholar.org/CorpusID:28942704.

[16] N. Friedman, D. Geiger, and M. Goldszmidt. "Bayesian Network Classifiers". In: *Machine Learning* 29.2 (Nov. 1997), pp. 131–163. ISSN: 1573-0565. DOI: 10.1023/A:1007465528199. URL: https://doi.org/10.1023/A:1007465528199.

[17] Y. Gao, Z. Huyan, and F. Ju. "A Prediction Method Based on Neural Network for Flight Turnaround Time at Airport". In: *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*. Vol. 2. 2015, pp. 219–222. DOI: 10.1109/ISCID.2015.44.

[18] IATA Airline A-CDM Coordination Group. *Airport - Collaborative Decision Making (A-CDM): IATA Recommendations*. IATA. 2018. URL: https://www.iata.org/contentassets/5c1a116a6120415f87f3dadfa38859d2/iata-acdm-recommendations-v1.pdf.

[19] E. Halmesaari. "Interpretable Machine Learning for Prediction of Aircraft Turnaround Times". MA thesis. Aalto University - School of Science, Dec. 2020. URL: http://urn.fi/URN:NBN:fi:aalto-2020122056351.

[20] O.F.J. van Hassel. "Predicting the turnaround time of an aircraft: a process structure aware approach". MA thesis. Eindhoven University of Technology, Dec. 2019. URL: https://research.tue.nl/en/studentTheses/predicting-the-turnaround-time-of-an-aircraft.

[21] K. Horimoto. "Conditional Independence". In: *Encyclopedia of Systems Biology*. New York, NY: Springer New York, 2013, pp. 484–485. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_452. URL: https://doi.org/10.1007/978-1-4419-9863-7_452.

[22] M. Jetzki. "The propagation of air transport delays in Europe". MA thesis. EUROCONTROL / RWTH Aachen University, Dec. 2009.

[23] S.S. Khan and A. Ahmad. "Cluster center initialization algorithm for K-means clustering". In: *Pattern Recognition Letters* 25.11 (2004), pp. 1293–1302. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2004.04.007. URL: https://www.sciencedirect.com/science/article/pii/S0167865504000996.

[24] V. Kuleshov and S. Ermon. *Variable Elimination - GitHub Pages*. https://ermongroup.github.io/cs228-notes/inference/ve/. Accessed: 2024-06-06. 2024.

[25] C. Lacave and F. Dez. "A Review of Explanation Methods for Bayesian Networks". In: *The Knowledge Engineering Review* 17 (May 2001). DOI: 10.1017/S026988890200019X.

[26] S. Liu, J. McGree, Z. Ge, and Y. Xie. *Computational and Statistical Methods for Analysing Big Data with Applications*. Ed. by S. Liu, J. McGree, Z. Ge, and Y. Xie. San Diego: Academic Press, 2016. ISBN: 978-0-12-803732-4. DOI: https://doi.org/10.1016/B978-0-12-803732-4.00001-5. URL: https://www.sciencedirect.com/science/article/pii/B9780128037324000015.

[27] M. Luo, M. Schultz, H. Fricke, B. Desart, F. Herrema, and R. Barragán Montes. "Agent-based simulation for aircraft stand operations to predict ground time using machine learning". In: *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*. 2021, pp. 1–8. DOI: 10.1109/DASC52595.2021.9594325.

[28] B. Marcot and A. Hanea. "What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis?" In: *Computational Statistics* 36 (Sept. 2021). DOI: 10.1007/s00180-020-00999-9. URL: https://link.springer.com/article/10.1007/s00180-020-00999-9.

[29] B.G. Marcot, J.D. Steventon, G.D. Sutherland, and R.K. McCann. "Guidelines for developing and updating Bayesian belief networks applied to ecological modeling and conservation". In: *Canadian Journal of Forest Research* 36.12 (2006), pp. 3063–3074. DOI: 10.1139/x06-135. eprint: https://doi.org/10.1139/x06-135. URL: https://doi.org/10.1139/x06-135.

[30] B. Mihaljevi, C. Bielza, and P. Larrañaga. "Bayesian networks for interpretable machine learning and optimization". In: *Neurocomputing* 456 (2021), pp. 648–665. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.01.138. URL: https://www.sciencedirect.com/science/article/pii/S0925231221009644.

[31] J.L. Molina, J. Bromley, J.L. García-Aróstegui, C. Sullivan, and J. Benavente. "Integrated water resources management of overexploited hydrogeological systems using Object-Oriented Bayesian Networks". In: *Environmental Modelling & Software* 25.4 (2010), pp. 383–397. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2009.10.007. URL: https://www.sciencedirect.com/science/article/pii/S1364815209002679.

[32] S.L. Morgan and C. Winship. *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. 2nd ed. Analytical Methods for Social Research. Cambridge University Press, 2014.

[33] D. Nash and M. Hannah. "Using Monte-Carlo simulations and Bayesian Networks to quantify and demonstrate the impact of fertiliser best management practices". In: *Environmental Modelling & Software* 26.9 (2011), pp. 1079–1088. ISSN: 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2011.03.009. URL: https://www.sciencedirect.com/science/article/pii/S1364815211000843.

[34] S. Neumann. "Is the boarding process on the critical path of the airplane turn-around?" In: *European Journal of Operational Research* 277.1 (2019), pp. 128–137. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2019.02.001. URL: https://www.sciencedirect.com/science/article/pii/S0377221719301110.

[35] B. Oreschko, T. Kunze, M. Schultz, H. Fricke, V. Kumar, and L. Sherry. "Turnaround Prediction with Stochastic Process Times and Airport Specific Delay Pattern". In: *5th International Conference on Research in Airport Transportation*. June 2012. URL: https://catsr.vse.gmu.edu/pubs/Kumar_ICRAT2012.pdf.

[36] B. Oreschko, M. Schultz, J. Elflein, and H. Fricke. "Significant Turnaround Process Variations due to Airport Characteristics". In: *Air Transport & Operations Symposium 2010*. June 2010. DOI: 10.3233/978-1-60750-559-4-263.

[37] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. ISBN: 1558604790.

[38] S. Reitmann and M. Schultz. "Real-time Prediction of Aircraft Boarding". In: *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. 2018, pp. 1–9. DOI: 10.1109/DASC.2018.8569370.

[39] *Schiphol Airport CDM Operations Manual*. Royal Schiphol Group. Sept. 2019. URL: https://www.schiphol.nl/nl/download/b2b/1569488978/7ERl8iHeLELDtgFsnK0mGi.pdf.

[40] Schiphol Group Aviation Solutions. *Deep Turnaround - Predictable turnarounds and improved collaboration*. https://www.schiphol.nl/en/aviation-solutions/page/deep-turnaround/ accessed on 01/08/2023. 2023.

[41] M. Schmidt. "A review of aircraft turnaround operations and simulations". In: *Progress in Aerospace Sciences* 92 (May 2017). DOI: 10.1016/j.paerosci.2017.05.002.

[42] M. Schultz. "Fast Aircraft Turnaround Enabled by Reliable Passenger Boarding". In: *Aerospace (EISSN 2226-4310)* 5 (Jan. 2018), p. 8. DOI: 10.3390/aerospace5010008.

[43] M. Schultz and H. Fricke. "Improving Aircraft Turnaround Reliability". In: *ICRAT - International Conferences on Research in Air Transportation*. June 2008.

[44] M. Schultz, T. Kunze, and H. Fricke. "Boarding on the critical path of the turnaround". In: *Proceedings of the 10th USA/Europe Air Traffic Management Research and Development Seminar, ATM 2013*. June 2013.

[45] M. Schultz, T. Kunze, B. Oreschko, and H. Fricke. "Microscopic Process Modelling for Efficient Aircraft Turnaround Management". In: *International Air Transport and Operations Symposium*. July 2013. URL: https://api.semanticscholar.org/CorpusID:198160711.

[46] N.R. Shadbolt and P.R. Smart. "Knowledge Elicitation: Methods, Tools and Techniques". In: *Evaluation of Human Work*. 4. CRC Press, May 2015.

[47] J. Skorupski and M. Wierzbisk. "A method to evaluate the time of waiting for a late passenger". In: *Journal of Air Transport Management* 47 (2015), pp. 79–89. ISSN: 0969-6997. DOI: https://doi.org/10.1016/j.jairtraman.2015.05.001. URL: https://www.sciencedirect.com/science/article/pii/S0969699715000563.

[48] O. Smith. *Ryanair unveils new game-changer seats but why wont it give us pockets?* Aug. 2017. URL: https://www.telegraph.co.uk/travel/comment/ryanair-new-planes-seat-back-pockets/.

[49] D.C. Snijders. "Investigating the effect on departure capacity of changing Target Off-Block Time uncertainty". MA thesis. Delft University of Technology, Feb. 2024.

[50] Eurocontrol Airport CDM Team. *THE MANUAL - Airport CDM Implementation*. Tech. rep. Eurocontrol, Mar. 2017. URL: https://www.eurocontrol.int/sites/default/files/publication/files/airport-cdm-manual-2017.PDF.

[51] ACI World. "The future of global air travel". In: *Airport World* 29.1 (Feb. 2024), pp. 18–19.

[52] C.L. Wu. "Monitoring Aircraft Turnaround Operations Framework Development, Application and Implications for Airline Operations". In: *Transportation Planning and Technology* 31.2 (2008), pp. 215–228. DOI: 10.1080/03081060801948233. URL: https://doi.org/10.1080/03081060801948233.

[53] X.S. Yang. In: *Introduction to Algorithms for Data Mining and Machine Learning*. Academic Press, 2019. ISBN: 978-0-12-817216-2. DOI: https://doi.org/10.1016/B978-0-12-817216-2.00008-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128172162000089.

[54] N.L. Zhang and D.L. Poole. "Exploiting Causal Independence in Bayesian Network Inference". In: *CoRR* cs.AI/9612101 (1996). URL: https://arxiv.org/abs/cs/9612101.

[55] C. Zhe, L. Yang, L. Yifan, X. Gaoyang, and Z. Yi. "Time Prediction of Off-Block Model Based on Serial Process Neural Network". In: *2019 IEEE 1st International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. 2019, pp. 120–124. DOI: 10.1109/ICCASIT48058.2019.8973174.

II

Literature Study
Previously graded under AE4020

1 Introduction

The aviation network is a complex and interconnected system, which requires the activities that drive air travel to finish a timely manner to ensure that flights depart on time. Untimely departures lead to disruptions in the rest of the aviation network, and subsequently may lead to inconvenience for passengers and financial losses for the aviation parties involved. Following the definition set by Eurocontrol [50], a timely departure refers to a flight departs earlier or within 15 minutes of its scheduled departure time. In Europe in 2022, only 68.6% of flights departed on time [50]. In recent years, predicting whether, and if so, the extent to which, an aircraft is delayed has become increasingly important in the field as the aviation industry aims to improve the punctuality of departing flights and to operate more sustainable and efficiently. While obtaining an accurate prediction of flight delays is important, understanding how a prediction came about is equally important. After all, this yields insights into the underlying processes and influential factors that affect the on-time-performance of flights, the root causes, and also fosters a users trust in the model's predictions. Subsequently, measures can be established to boost the efficiency of the departure process. The main focus of this literature study is to obtain a set of models that are accurate in predicting flight departure delays, as well as provide insights into the underlying complex system.

A proper literature review is of paramount importance in order to obtain a critical summary of existing knowledge and research in a field of interest. After all, without a thorough understanding of the elements, a researcher cannot perform proper research. When examiners review a dissertation that is not up to expectations, they assume the rest of the dissertation might be flawed as well [20]. In conducting a literature review, it is quintessential to define in advance what knowledge one aims to build. For this, one can formulate literature research questions, in line with the framework proposed by Randolph [157]. Please find the literature research questions for this literature study and a justification on them below.

(i) *What ground operation processes are most influential on the on-time-performance of departing flights?*
With this question, the aim is to identify ground operation processes which are most prone to be sources of ground operation delays.

   (a) *What processes make up aircraft ground operations?*
   This subquestion aims to identify the various processes involved in aircraft ground operations. When one has a thorough understanding of the processes involved, it becomes easier to analyse their impact on the on-time-performance of departing flights.

   (b) *What factors have an influence on the duration of the ground operation processes under (a)?*
   This subquestion aims to lay out the factors that affect the duration of ground operation processes. This helps one gain insights into the factors that drive these processes.

   (c) *Which of the factors under (b) may have a significant influence on the process duration?*
   This subquestion aims to gain a deeper understanding of the relative influence of the factors identified in (b) to determine the driving factors of the process duration.

(ii) *What other elements, processes and/or factors may have a (significant) influence on the on-time-performance of departing flights?*
This question focuses on the identification of other elements, processes and/or factors that may contribute to flight delay while the aircraft is on ground.

(iii) *What is the state-of-the-art in academic research in the area of ground operations, focusing on predicting the delay and/or on-time-performance of departing flights?*
This question aims to determine the state-of-the-art of the research in the field and requires a thorough analysis of relevant academic papers. From this, one can identify gaps in research and subsequently allows one to determine the contribution of this research to the field.

(a) *What relevant academic research papers can be identified in this area of research?*
This subquestion aims to identify relevant academic papers that discuss aircraft ground operations and predict flight departure delays and/or the on-time-performance of departing flights to gain insights into previous research and obtain a sense of the status quo in the field.

(b) *What scope did the researchers in the field consider for their research?* This subquestion aims to identify the scope of the relevant academic research papers, which gives an understanding of the processes and/or factors the researches in the field have taken into account and also which they have excluded.

(c) *What methods and/or models have the researchers in the field used to obtain their results?*
This subquestion aims to give an understanding of the methods and/or models the researchers in the field have used to obtain their results, which in term helps to identify trends in model selection and possibly allows to get a first estimate of the applicability of their findings to the problem at hand.

(d) *What data have the researches in the field used to obtain their results?*
Gaining insights into the sources and quality of data the researchers have used helps one to determine the reliability of their results. The quality of the model results are directly dependent the quality of the model input and therefore the model is only as good as the quality of the inputs provided (garbage in yields garbage out, after all [163]).

(iv) *What model or combination of models are appropriate to predict flight departure delays in an interpretable and/or explainable manner?*
This question aims to identify appropriate machine learning models to predict flight departure delays, and also provide insights into the inner workings (i.e. decision process) of the model.

(a) *What ante-hoc interpretable machine learning models are suited to efficiently and accurately predict flight departure delays?*
This subquestion aims to identify suitable machine learning models that are inherently transparent allowing for easier understand and interpretation of the model output, and can efficiently, adequately and accurately predict flight departure delays.

(b) *What black-box machine learning models are suited to efficiently and accurately predict flight departure delays?*
This subquestion aims to identify black-box models that do not possess the inherent interpretability that the models under (a) offer as they are more complex. In term, black-box models offer higher predictive performance than the models considered under (a).

(c) *What methods are available to add interpretability and/or explainability to the models considered under (b)?*
This subquestion aims to identify methods to enhance the interpretability and/or explainability of the black-box models considered under (b) in order to provide insights into the driving factors behind the model prediction.

Summarising the above, the purpose of this literature review is to identify relevant ground operation processes and/or factors affecting the on-time-performance of flights, evaluate the state-of-the-art in academic research in predicting flight delays and to identify methods to predict flight departure delays in an interpretable and/or explainable manner in order to extract insights into the underlying system.

This report is structured as follows. First, in chapter 2, the A-CDM implementation at Amsterdam Airport Schiphol (AAS) and the factors that drive the on-time-performance of departing flights are elaborated on. Second, an overview of the state-of-the-art in the field of flight departure delay prediction is presented in chapter 3. Third, chapter 4 discusses a selection of machine learning techniques used by researchers in the field, as well introduces new machine learning techniques which may be appropriate to solve the problem at hand. Fourth, chapter 5 explores explainability frameworks to provide insights into the decision process

of a machine learning model. Last, chapter 7 concludes the report, and chapter 6 elaborates on the research proposal.

2

Exploring A-CDM at Schiphol Airport and Factors Influencing OTP of Flights

Air transportation is a vital component in facilitating global connectivity. For this, the on-time-performance of flights is of paramount importance. With the ever increasing demand for air travel, making sure all flights arrive and depart on time is a major challenge. From an airport perspective, this requires careful coordination with the operational partners in the aviation sector such as the ground handlers, airlines and Air Navigation Service Providers (ANSPs). Moreover, this requires the removal of inefficiencies from the ground processes that make air travel possible in order to operate at maximum efficiency and accomplish timely departures. However, it is quintessential to first analyse the factors that have an influence on these processes to gain insights into the underlying complex system.

This chapter aims to elaborate on the Airport Collaborative Decision Making (A-CDM) implementation at Amsterdam Airport Schiphol (AAS), which encourages the operational partners in the aviation sector to work more transparently and collaboratively [51]. This is elaborated on in section 2.1. Next to this, this chapter aims to lay out the initial research scope, the processes that drive air travel, and the factors that may have a negative influence on the duration of these processes and subsequently the on-time-performance of departing flights. These are presented in section 2.2.

2.1. Airport Collaborative Decision Making (A-CDM) at Schiphol

At Amsterdam Airport Schiphol (AAS), Airport Collaborative Decision Making (A-CDM) was first implemented locally in 2015 and in 2018 connected to the European network managed by the Eurocontrol Network Manager Operation Centre (NMOC) [75]. A-CDM has the objective to increase the overall efficiency at airports by tackling the unharmonized processes and to improve the Air Traffic Flow and Capacity Management (ATFCM) [74, 171]. To achieve this, the aim is to reduce delays, improve the predictability of events (e.g. end of ground handling time) and optimise the utilisation of resources [195]. For this, A-CDM encourages the airport partners to work more transparently and collaboratively [51]. Here, operational partners share accurate information on the flight status, schedules and aircraft handling process [128]. Operational partners at Schiphol include but are not limited to: (1) Amsterdam Airport Schiphol, (2) Luchtverkeersleiding Nederland (LVNL), (3) KLM Royal Dutch Airlines, (4) the Royal Netherlands Meteorological Institute (KNMI), (5) the aircraft ground handling parties and (6) the Schiphol Airline Operators Committee (SAOC) [75].

The implementation of A-CDM at AAS provides a useful source of data in light of this literature study. In A-CDM, a milestone approach is utilised. Here, milestones are considered significant events in the planning or progress of a flight [171]. The A-CDM milestones prescribed by Eurocontrol are presented in Figure 2.1. At every milestone, the milestones downstream are updated leading to increase in the accuracy in which the future progress can be predicted. These milestones are updated by the operational partners and therefore no single entity or organisation updates provides the updates for all milestones [171]. Here, it is realistic to assume that the quality of these updates may vary as the data quality sometimes is "agreed locally" [195].

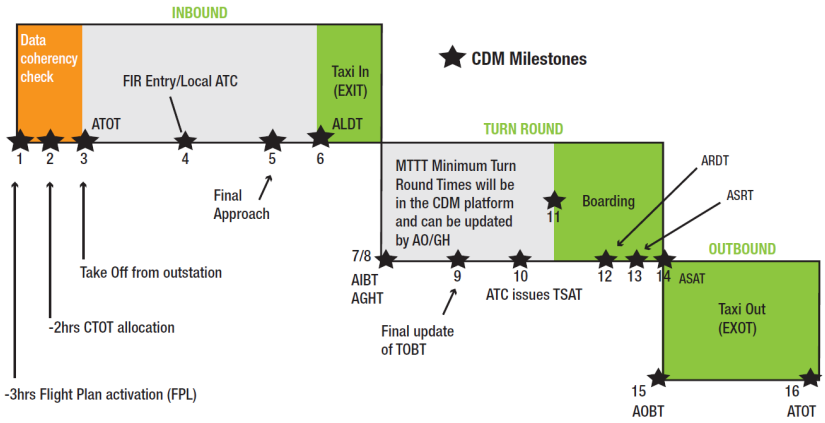

Figure 2.1: Collaborative Decision Making milestone event sequence [195]

The milestones of primary interest in Figure 2.1 are milestones 6 to 16, namely the milestones between the Actual Landing Time (ALDT) and the Actual Take-Off Time (ATOT). The milestones in between are further elaborated on in this paragraph, based on [195, 171]. Milestones 7 and 8 denote the instances the aircraft comes to a complete stop at the stand, Actual In Block Time (AIBT), and when ground handling commences, Actual Commence of Ground Handling (ACGT), respectively. Here, note that is assumed that ACGT commences at AIBT as prescribed in the A-CDM implementation manual [195]. Milestone 9 requires the Main Ground Handler (MGHA) or aircraft operator to provide the most accurate prediction of the instance the aircraft is ready to leave the aircraft stand (Target Off-Block Time (TOBT)). Following this milestone, Air Traffic Control (ATC) issues the Target Start-up Approval Time (TSAT) (milestone 10). Milestone 11 denotes the instance boarding commences. Next, milestones 12 defines the moment in time when the doors is closed, jet bridge and/or stairs are removed, push-back truck is connected and the aircraft is ready to taxi when instructions from the tower are received. Milestones 13 and 14 are updated at the instances start up is requested by the pilots, Actual Start up Request Time (ASRT), and approved by ATC (Actual Start up Approval Time (ASAT)), respectively. Lastly, milestone 15 denotes the instance the aircraft vacates the aircraft stand, Actual Off Block Time (AOBT). An overview of the origin and priority of the milestones updates is presented in Table 2.1. Next, the influential factors on the timing of milestone events 6 to 16 are presented in section 2.2.

Table 2.1: Overview of the operational status, origin & priority of the updates, timing and data quality for the relevant milestone events (information from [195, 171])

Milestone events	**Operational status**	**Origin & priority**	**Timing**	**Data quality**
Milestone 6 (ALDT)	LANDED	ATC system or ACARS	Directly available	$\pm 1 minute$
Milestone 7 (AIBT)	IN-BLOCK	ACARS or automated docking systems or ATC systems or manual input	Directly available	$\pm 1 minute$
Milestone 8 (ACGT)	IN-BLOCK	Airport Operator (AO) or MGHA	Directly available	$\pm 1 minute$
Milestone 9	IN-BLOCK	Airport Operator (AO) or MGHA	t minutes before EOBT	Agreed locally
Milestone 10	SEQUENCED	ATC	t minutes before EOBT	Agreed locally
Milestone 11	BOARDING	Automatic airport system or manual input from AO or MGHA	Directly available	$\pm 1 minute$
Milestone 12 (ARDT)	READY	Airport Operator (AO) or MGHA	Directly available	$\pm 1 minute$
Milestone 13 (ASRT)	READY	ATC	Directly available	$\pm 1 minute$
Milestone 14 (ASAT)	READY	ATC	Directly available	$\pm 1 minute$
Milestone 15 (AOBT)	OFF-BLOCK	ACARS or automated docking systems or ATC systems or manual input	Directly available	$\pm 1 minute$
Milestone 16 (ATOT)	DEPARTED	ATC system or ACARS	Directly available	$\pm 1 minute$

2.2. Ground Operation Factors influencing on the OTP of flights

Maintaining On-Time-Performance (OTP) is of paramount importance in order to operate as efficiently as possible. This requires a series of activities that occur between a flight's arrival and departure to be completed in a timely manner. In this section, the aim is to present an overview of the series of activities and the, often interdependent, factors at play in these activities to ensure a timely turnaround. Four ground phases can be distinguished. Namely, taxi-in, ground handling, start-up/pushback and taxi-out. This yields the initial research scope as presented in Figure 2.2.

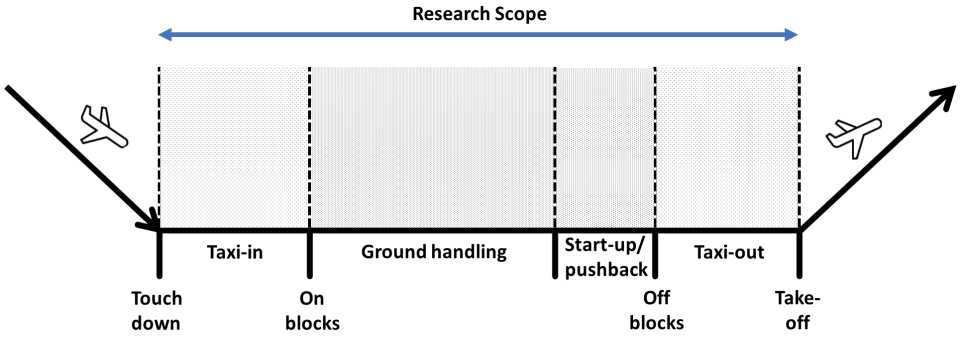

Figure 2.2: Initial research scope (figure based on work from Van Hassel [81])

This section is structured as follows. First in subsection 2.2.1, arguably the most important, the series of ground handling activities during aircraft turnaround are elaborated on. Second, delay caused by load connection, e.g. late arrival of crew, transfer passengers or the aircraft are discussed in subsection 2.2.2. Third, weather impacts on the on-time-performance of departing flights are presented in subsection 2.2.3. Fourth, Air Traffic Control (ATC) factors that may influence the on-time-performance negatively are discussed in subsection 2.2.4. Last, in subsection 2.2.5, the influence of aircraft taxi operations on the On-Time-Performance (OTP) are elaborated on.

2.2.1. Influences of ground handling on the OTP of flights

The aircraft turnaround consists of many processes which together aim to prepare the aircraft for its next flight. In this literature review, the turnaround time is defined as the time between the moment the aircraft comes to a complete stop at the stand (Actual In Block Time (AIBT)) and when all necessary turnaround processes have finished and the aircraft has left the stand (Actual Off Block Time (AOBT)). During the turnaround process, among other things, boarding, refuelling and baggage (un)loading take place. A schematic overview of a flight support process is shown in Figure 2.3.

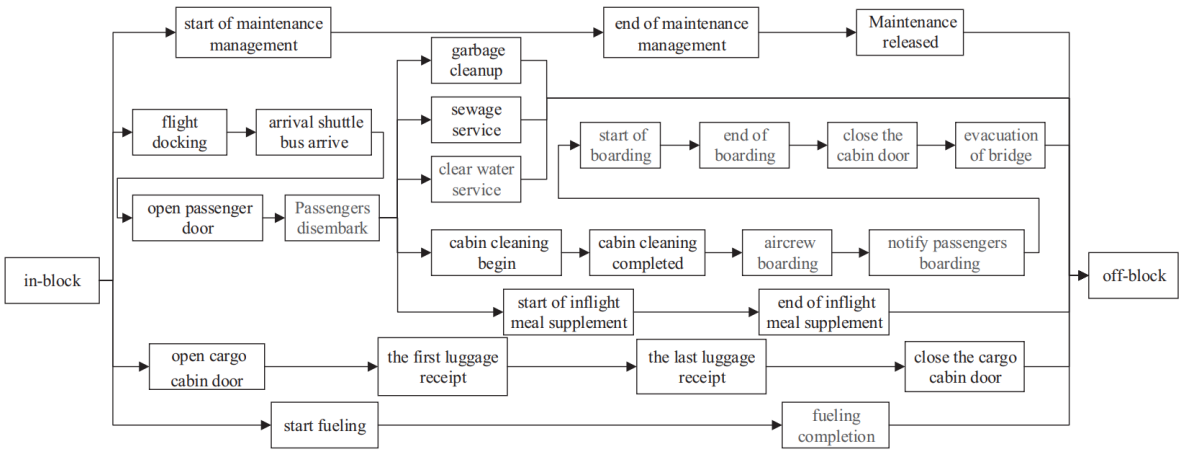

Figure 2.3: An example of a flight support process [224]

In the flight support process, some processes are dependent on other processes due to technical, legal or operational dependencies meaning that the initial process has to be completed before the start of the next process, e.g. boarding can only start when cleaning of the aircraft has finished [59, 176]. Hence, these can be considered as sequential activities [4]. Other processes may be executed in parallel, such as boarding and baggage loading. The longest path, measured in units of time, of parallel and sequential events yields the shortest turnaround time and is called the critical path [148]. This implies that when processes on the critical path are delayed, the whole process is delayed [143]. Given the complexity of the operations the processes are stochastic in nature and therefore the critical path is variable [4, 6, 147, 216].

Boarding is the act of loading passengers onto the aircraft and is often on the critical path of the turnaround [148, 172, 174, 175, 181]. Boarding can only commence when the jet bridge or stair is connected. The boarding process duration is influenced by various variables. The most prominent variable is the number of passengers. As the number of passengers on a flight increases, the boarding time also increases [65, 158, 175]. The layout of the aircraft cabin and number of jet bridges/stairs also influence the boarding rate [172]. Some carriers make use of boarding schemes, prioritising certain individuals or groups. These boarding schemes in term influence the boarding duration [175]. On most airports, when the aircraft is on a remote stand, the passengers are transferred from the terminal to the aircraft which requires the busses to arrive and depart in a timely manner. Untimely arrival and departure of busses consequently yields a longer boarding times. Next to this, Passengers with a reduced mobility (PRMs) require more assistance and may slow down the boarding process [81]. Moreover, the willingness and/or the ability to adhere to the boarding procedure plays a role, as it may influence the boarding rate [158, 173]. Another significant factor contributing to the increase in boarding duration is the increasing amount of hand luggage, especially trolleys [158, 172, 173]. No narrow-body aircraft has sufficient overhead bin capacity to stow a hand luggage trolley for every passenger on a fully booked flight [26]. When the total volume of passenger hand luggage exceeds the total overhead bin capacity, then the remaining hand luggage is loaded into the cargo hold further delaying the flight support process. In addition, the passenger anthropometrics also may influence the boarding duration [172]. Lastly, missing or belated passengers passengers may contribute to the overall boarding duration. In the case of missing passenger(s), their luggage has to be removed from the aircraft. However, the time one waits for the missing passengers (if at all) before one proceeds with the next activities in the turnaround process, consequently leaving the missing passengers behind, is proprietary information and not the same for all airlines [181].

Passenger de-boarding is the act of unloading passengers from the aircraft. The process starts as soon as the jet bridge or stair is connected and the crew has finished all preparations for passenger disembark. Next to boarding, de-boarding is often part of the critical path [172, 174]. The number of passengers has a direct influence on the de-boarding process duration [65]. Many other variables, also present in the boarding process, may influence its duration. For example, the willingness and/or ability to adhere to the de-boarding procedure as well as the amount of hand luggage [158, 173, 172].

Cleaning of the aircraft cabin is a typical member of the critical path for short haul flights and is required to meet passenger expectations [4, 172, 174]. Cabin cleaning may consist of processes such as rubbish removal, dirt removal or in light of the COVID-19 pandemic, disinfection. The duration of the cabin cleaning process in an aircraft turnaround depends on which cabin cleaning service(s) the ground handler offers and the airline has selected [176]. For example, Menzies Aviation[1], a ground handler on Schiphol, offers three different cabin cleaning solutions.

Luggage and cargo (un)loading are also part of the turnaround process. The amount of luggage and cargo varies from flight to flight, contributing to the stochastic nature of the turnaround process [172]. Loading and unloading can be considered as a typical process member of the critical path and the amount of cargo and luggage directly influences the turnaround duration [174, 172]. Before the luggage and cargo can be offloaded and/or loaded, one or more belt loaders and/or high loaders need to be correctly positioned relative to the aircraft. The duration of the process is dependent on the type and amount of cargo and luggage. In order to meet the turnaround schedule, the belt and/or high loaders, luggage and cargo should arrive at the aircraft in a timely manner. This in term requires the airport baggage handling chain to operate without delay.

Next to this, the skill level of ground handling staff influences the turnaround performance [149, 6]. It can be

[1] https://menziesaviation.com/services/ground-handling/

assumed that the respective ground handling company handling the aircraft turnaround also has an effect on the turnaround performance as it is expected that there are differences in resources the ground handling companies are able to allocate towards aircraft turnarounds. Moreover, it can be assumed that some ground handling companies are able to handle schedule disruptions better than others, especially given that as of writing there are 6 main ground handling parties at Schiphol [119].

Time of day also may influence the turnaround duration [65]. During peak moments, less resources and manpower are available to allocate towards the turnaround process of a specific flight. Next to this, an untimely resource scheduling and negligence of manpower may disrupt the turnaround process, resulting in departure delays [229].

During flight, foods and beverages are handed out or available for purchase. These are replenished during the flight support process by a caterer. This replenishment of foods and beverages may directly influence the turnaround duration [172, 174, 224]. The process can be split in various elements, such as timely arrival of the catering supplies (and subsequently positioning the catering truck correctly with respect to the aircraft) or (un)loading of the foods and beverages. All of which influence the duration of the catering process, and subsequently the turnaround duration.

Commercial aircraft require fuel to fly from their origin to destinations, and burn this fuel along the way. Hence, aircraft (re)fuel during the flight support process. In short-haul operations, fueling is often part of the critical path and is generally executed in parallel with the catering and cleaning process [172, 174]. The refuelling time is directly related to the distance to the destination (i.e. required fuel quantity) [6, 81]. In order to start fueling on time, this requires the fuel truck to arrive, position itself and connect to the aircraft in a timely manner. Various strict fire precautions are in place to prevent fueling to lead to unsafe conditions. These precautions subsequently also affect the duration of the fueling process.

Aircraft types also have an effect on the turnaround duration [6, 175, 172, 59]. Airport schedules show a variance of more than 100% in the ground time of different aircraft types [59]. Widebody aircraft may have a significantly longer ground time. This is due to the fact that the individual process times may take longer. For example, the catering process takes often takes longer on widebody aircraft with respect to narrowbody aircraft since widebodies generally fly longer and to farther destinations than narrowbodies and therefore require more supplies. Widebody aircraft may also be more susceptible to customs and immigration controls due to their more distant destinations, increasing the turnaround process duration [4].

Moreover, line maintenance (A-check) may be executed during the turnaround process [47, 224]. This may be due to unforeseen circumstances, such as defective parts, but can also be scheduled activities such as routine and/or additional engineering inspections. During line maintenance, the aircraft is not taken out of service and the aircraft remains in its operating environment (i.e. aircraft stand) [73]. The main contributor to line maintenance delay is a poor logistic process, mainly availability of spares, followed by unscheduled maintenance defects [134].

The business model between the airlines varies, and subsequently may influence the turnaround duration [6, 172]. One of the objectives of Low Cost Carriers (LCC) is to achieve maximum aircraft utilization. Therefore, they perform strictly the necessary set of tasks and may even opt to omit catering from the turnaround process [4]. LCCs often have tight turnaround schedules with respect to traditional carriers, and allocate less buffer time. Hence, the on-time performance between traditional carriers and LCCs varies as the former is able to better absorb delays [95]. Consequently, LCCs show a lower percentage of on-time flights with respect to traditional carriers [114]. Airlines establish dynamic buffer strategies in order to limit the consequence of deviations in turnaround time and maximize the possibility to adhere to the flight schedule [176, 126]. Some airlines (e.g. hub-and-spoke carriers) may opt to increase their buffer times even further at their hubs, as they have often have a high volume of connecting passengers [148]. Increasing the buffer size allows them to mitigate the risk of a passenger missing their connection. However, Fricke and Schultz [59] found no systematic pattern in buffer strategies and state that the efficiency of buffer strategies is mostly dependent on the information available at the time the disruption occurs and the airlines operating experience.

When one of these factors results in a turnaround delay and belated flight departure, its subsequent flight

might also be delayed when the delay cannot be absorbed. Such cases are classified as reactionary delays, and are elaborated on in subsection 2.2.2.

2.2.2. Influence of Reactionary Delays on the OTP of flights

Reactionary delay refers to the causes of load connection, e.g. late arrival of aircraft, belated or missing (cabin) crew and/or passengers, check-in errors or operations control [173]. In 2022, roughly 48% of the European delays originated from the belated arrival of aircraft and crew, translating to 8.0 minutes per flight [50]. The influence of reactionary delays with respect to the total delay on the turnaround duration is therefore significant [172, 176, 143, 6]. This is also the case for delays related to airline and turnaround operations. In 2022, these amounted to 4.9 minutes per flight [50].

When an aircraft arrives late, the flight support process cannot commence on time. Whether the aircraft is able to leave in a timely manner depends on whether sufficient buffer time is allocated by the airline to absorb this delay. If the aircraft cannot depart in a timely manner or the delay cannot be absorbed in flight (e.g. by flying at a higher cruise speed), the delay propagates to subsequent flights of the aircraft. Morning sequences of reactionary delays have a higher impact and magnitude than those later in the day as the delay propagates along more subsequent flights [95]. As already touched on in subsection 2.2.1, the ability to absorb root delays varies along the airlines. Traditional hub-and-spoke carriers are able to absorb a significant amount of the delay in the turnaround phases, while LCCs generally absorb most delay in the block-to-block phase [95]. This makes them prone to primary delays and hence, given that it is not always possible to absorb delays in the block-to-block phase, the reactionary delay usually increases over subsequent flights.

Next to this, if an aircraft with connecting passengers on board arrives considerably late, this delay may propagate through the entire aviation network as the connecting passengers are more likely to miss their connecting flights. Airlines may opt to wait for the passengers on their connecting flights, which subsequently delays these flights leading to a cascade of delays. The same is true for belated crew when they are assigned to subsequent flights.

Next to turnaround delays, reactionary delays may also be the result of disturbances due to adverse weather. This is discussed in the next subsection, subsection 2.2.3.

2.2.3. Weather impact on the OTP in Air Travel

Adverse weather conditions such as thunderstorms, poor visibility, low cloud ceiling and strong winds, may disrupt airport operations [138]. Consequently, the adverse weather conditions reduce the airports capacity, limiting the number of arrivals and departures. This may induce additional delays or lead to cancelled flights [176, 47]. In term, these may disrupt the carriers future operations through reactionary delays, already touched upon in subsection 2.2.2.

At Amsterdam Airport Schiphol, when a thunderstorm is present within a radius of 5 [km], the flight support process has to come to a halt in order to guarantee the safety of the workers on the platforms [146]. This means that refuelling, baggage (un)loading and other turnaround processes come to a complete stop. Subsequently, this adds delay to the scheduled flights. Moreover, strong winds may also yield dangerous situations. For example, very strong winds are able to lift light aircraft into the air or tip them over [199].

Next to this, cold weather conditions may lead to the need for deicing in order to assure safe flight [176, 47]. Deicing considers the removal of snow, ice or frost from the surfaces of an aircraft. Typically, this process consists of two steps [145]. However, it is not always the case that both steps need to be performed. It is difficult to know in advance if deicing operations have to take place, and also to determine which step(s) should be taken into account [79]. The first step considers with the removal of frost and ice using a warm liquid substance (type I fluid), such as a buoyant glycol mix. The second step considers with the prevention ice and frost forming on the aircraft, using a thicker fluid (type II-IV fluid). It should be noted that this prevention is only temporary, as the fluid wears off over time [145]. At AAS, deicing operations take place at the aircraft stands (at-the-gate deicing) and on remote stands depending on the deicing handler [203].

As mentioned previously, adverse weather conditions reduce the available capacity on an airport. This is true for both the origin and destination of a flight. If there are capacity problems at the destination airport, the air-

port may request Air Traffic Flow Management (ATFM) regulations, limiting the inbound flow. This, and other Air Traffic Control (ATC) factors affecting the timely departure of flights are discussed in subsection 2.2.4.

2.2.4. Air Traffic Control factors influencing OTP

When the turnaround process is finished and the aircraft is ready to depart, i.e. all ground handling activities finished (including the removal of handling equipment and jet bridges/stairs) and the doors are closed [171], the pilots may request start-up and push-back approval from ground control. Push-back is required since an aircraft is unable, or is not allowed, to leave the respective aircraft stand under its own power [176]. However, before they can request approval, first the flight needs to have obtained clearance from the delivery controller. The delivery controller provides the Instrument Flight Rules (IFR) clearance, expected runway, the clearance limit, Standard Instrument Departure (SID) and aircraft identification (SSR-code) [113, 48].

At Amsterdam Airport Schiphol (AAS), the delivery controller position may also be combined with the outbound planner position[2]. When the aircraft is ready to leave the stand, the pilots contact the outbound planner. The outbound planner will verify if the flight is in its Target Start-up Approval Time (TSAT) window. If so, the outbound planner will set a Revised Estimated Time of Departure (RETD) to help tower planning [48]. If a flight contacts the outbound planner before the TSAT window, then the outbound planner will ask the pilots to stand by till the TSAT window opens. If the aircraft is at a push-back stand, the outbound planner will request the pilots to contact ground control for start-up and push-back approval. If parked at a stand that allows taxi-out, then the outbound planner may provide start-up clearance and requests the pilot to contact ground control.

In the departure sequence, ground controllers provide start-up (if still applicable) and push-back clearances and aim to route the aircraft from the aircraft stand to the runway in an as safe and efficient as possible manner. They also aim to achieve an efficient departure flow on the runways, by sequencing departing aircraft at the runways for an efficient departure flow [48]. At the runway, control is handed over to the tower controllers, responsible for providing aircraft with their take-off clearance and/or other instructions.

Departure delays can also arise from Air Traffic Flow Management (ATFM) regulations (or alternatively: Ground Delay Programs (GDP)). ATFM measures may be instated by the local Flow Management Position (FMP) (in Europe) upon request from the ATC centers in case expected traffic demand exceeds the available capacity in these centers and/or airports [49]. In this case, aircraft that are expected to arrive at the respective sectors and/or destination airports when insufficient capacity is available, are delayed on ground on their origin airport in order to limit airborne holding [1]. The goal of ATFM delay is to regulate the flow of traffic through the downstream sectors.

While a ground controller is able to act influence on the total taxi duration, other factors also affect the total taxi duration of an inbound or outbound flight. This quickly elaborated on in subsection 2.2.5.

2.2.5. Taxi operations influences on the OTP of flights

As aircraft are not able nor allowed to directly take-off from their stands, taxiing is required from the aircraft stand to the runway (and vice versa for arrivals). Here, the total taxi distance is of primary interest, as this is the primary driver of the total taxi time. The total taxi distance is dependent on the airport surface layout, assigned gate and runway configuration. Next to this, ground controllers have an influence on the total taxi distance and time as they determine the route of the aircraft between the gate and runway. When the airport surface is congested, aircraft may be asked to hold at their positions or take alternative, possibly longer, routes[2]. Next to this, various other variables are at play. For example, weather conditions, aircraft type, airport specific factors (e.g. operational procedures) and pilot behavior [78].

[2]Information obtained from listening to Amsterdam Live Air Traffic Control, https://www.liveatc.net/search/?icao=EHAM.

3

Predicting Flight Departure Delays: the State-of-the-Art in the Field

Predicting whether a flight departs late is a complex and challenging problem, as delays can stem from a wide variety of process and factors (chapter 2). Over the last decade, predicting whether a flight departs late has drawn the attention from the academic research community, partly due to the increase in the available data (e.g. via A-CDM), rise of advanced machine learning models that are able to capture complex systems and the increase in computational power. In this chapter, a selection of papers in the field of flight departure delay are briefly summarised and assessed in section 3.1, with the goal of finding answers to question (iii) in chapter 1. Next, section 3.2 presents a discussion on the selection of papers and identifies research gaps.

3.1. Relevant literature in the field of flight departure delays

This section elaborates on relevant academic papers in the area of predicting flight departure delays. These papers are summarised and assessed in Table 3.1. When one or more authors are affiliated with research institutes actively publishing in the field, a superscript with their affiliation is added. A discussion on the overview is presented in the next section, 3.2.

Table 3.1: Overview of the state-of-the-art in predicting flight departure delays

Reference	Model foundation(s)	Short summary & remarks
[81]	Multilayer Perceptron (MLP), Random Forest (RF), Critical Path Method	In collaboration with an aircraft ground handler, Van Hassel [81] proposes a Process Structure Aware Prediction (PSAP) approach to predict the taxi-in and turnaround duration for Boeing 737 flights of a major European LCC at Eindhoven Airport in an interpretable manner. In the PSAP approach, the turnaround process is split into a set of activities of which the cycle time is predicted. For this, Van Hassel [81] considers two algorithms: (1) Random Forest and (2) Multilayer Perceptron (MLP) (a fully connected feedforward ANN [66]). In the PSAP framework, Van Hassel [81] found that the performance can be considered equal between the models. In an aggregated approach, when the process structure is not explicitly defined, MLP proved to be more capable in estimating the turnaround duration. However, Van Hassel [81] only considers a handful of processes and factors in his research. For example, transfer passengers are not included in the model, nor is aircraft catering and cleaning considered. It is therefore uncertain how well Van Hassels [79] findings translate to the problem at hand.
[137]	Johnson-SU distributions, Artificial Neural Networks (ANN)	Mori [137] argues that the TOBT accuracy is not constant between turnarounds and therefore proposes to predict the statistical distribution of the TOBT. For this, Mori [137] makes use of Artificial Neural Networks (ANN) trained on the TOBT update history of flights from a French A-CDM airport to estimate the parameters of a Johnson-SU distribution. In the ANN, only five features are considered: (1) time to latest TOBT, (2) time since last TOBT, (3) number of TOBT updates, (4) airline and (5) the Scheduled Off-Block Time (SOBT). No other features are considered in the model, but the model shows potential in using the TOBT update history to obtain a more accurate TOBT prediction.

Continued on next page

Table 3.1 – Continued from previous page

Reference	Model foundation(s)	Short summary & remarks
[79]	eXtreme Gradient Boosting (XGBoost), SHapley Additive exPlanations (SHAP)	In collaboration with a small (non-hub) airport and a Scandinavian full service carrier, Halmesaari [79] presents an explainable aggregated machine learning approach to predict the aircraft ground handling process duration. The approach consists of two steps: (1) obtaining a prediction of the turnaround process duration using a gradient boosted tree-ensemble regression model, XGBoost, and (2) extracting explanations from the model using a post-hoc explanatory framework, SHAP. One of the limitations in Halmesaari's [79] study is the lack of data of available data. Although Halmesaari [79] found that in most cases the turnaround duration can be described by only a few variables, in cases where the turnaround is significantly longer than the scheduled duration the available data is not able to describe this discrepancy. Next to this, the airport in Halmesaari's [79] work has special characteristics. For example, as the airport is relatively close to Helsinki (hub of the collaborating airline), aircraft often do not refuel on said airport. Given the size and (special) characteristics of the airport considered in Halmesaari's work [79], it can not be said with certainty whether Halmesaari's findings translate well to other airports.
[65]	Artificial Neural Networks (ANN)	Gao, Huyan, and Ju [65] utilise Artificial Neural Networks (ANN) to predict the flight turnaround time of flights at a large international hub airport in China. In the ANN model only 7 features are considered, namely: (1) aircraft stand (to account for distance for support vehicles), (2) aircraft type, (3) type of flight (domestic or international, accounting for immigration and custom inspections), (4) aircraft ground handler, (5) flight arrival time (to account for high turnaround demand), (6) number of arriving passengers and (7) number of departing passengers. With this low number of variables, the model was able to find the general trend in the data. However, while Gao, Huyan, and Ju [65] provide a first order estimate of the applicability of ANNs in predicting the turnaround duration, the results show that more features are required to obtain accurate results.
[96]	Latent Semantic Indexing (LSI), Convolutional Neural Network (CNN)	Jian et al. [96] consider a two-stage model to predict the off-block time. In the first stage, the model input dimensions are reduced using the Latent Semantic Indexing (LSI) algorithm, capturing only the most important features. Next, this result is forwarded to a Convolutional Neural Network (CNN), an ANN that adopts local connection and weight sharing [96]. On average, the two-stage model shows higher predictive performance than fully connected ANNs in the error ranges of ±5 [min], ± 10 [min] and ±15 [min]. However, Jian et al. [96] consider mostly features that are only available during the aircraft turnaround (e.g. opening and closing of cargo hold, start and end time of boarding). Therefore, Jian et al. [96] aim to predict the flight departure delay during the flight turnaround, taking into account the ground handling progress and updating this prediction in various intervals.
[39][a]	Gradient Boosted Decision Trees (GBDT), Artificial Neural Networks (ANN), SHapley Additive exPlanations (SHAP)	Dalmau et al. [39] aim to obtain a better estimation of the Estimated Take-Off Time (ETOT) than is extracted from the Enhanced Tactical Flow Management System (ETFMS) Flight Data (EFD), as they deem it to be too inaccurate for their needs. In order to obtain a more accurate estimation of the ETOT, Dalmau et al. [39] utilise Gradient Boosted Decision Trees (GBDT) and ANNs trained on Enhanced Tactical Flow Management System (ETFMS) and weather data, and compare the performance against the ETFMS. Next to this, Dalmau et al. [39] apply the SHAP framework to obtain a features importance. In the model, the turnaround is not explicitly modelled, but rather taken into account implicitly in the ETFMS data (e.g. timestamps from processes after the turnaround has finished). However, Dalmau et al. [39] note that the prediction accuracy can be significantly improved when the features on the turnaround process are incorporated in the model.
[59][b]	Weibull Distributions, Gamma Distributions, Critical Path Method	Fricke and Schultz [59] present a statistical approach to determine the turnaround process duration. In the model, the turnaround process is split into five sub-processes, namely: (1) de-boarding, (2) cabin cleaning, (3) catering, (4) fueling and (5) boarding. For every process, a Weibull or Gamma distribution (whichever describes the process best) is fit using operational data from a regional airport in the United States. Next, in a similar fashion to [81], the critical path method (i.e. correctly taking into account the sequential and parallel dependencies between turnaround events) is used to obtain total turnaround process duration. The model is rather simplistic, however. In the model, only a handful of processes are considered, neglecting the influence of key turnaround process such as baggage and cargo (un)loading. Next to this, the statistical approach suffers from some significant limitations (e.g. the inability to deal with categorical values) limiting its applicability to the problem posed in this literature study.

Continued on next page

Table 3.1 – Continued from previous page

Reference	Model foundation(s)	Short summary & remarks
[6][b]	Weibull Distributions, Gamma Distributions, Analytical Convolution	Asadi et al. [6], on the basis of work in [59], propose a novel analytical convolution method to predict the TOBT of a flight, taking into account uncertainties in the turnaround process. The turnaround process is split into various sub-processes. For each process, the authors use and, in some cases, (re)parameterize the Weibull or Gamma distributions obtained in [59]. Next, using analytical convolution the stochastic process times are obtained. Taking into account the parallel and sequential dependencies between the various processes, Asadi et al. [6] apply analytical convolution to obtain the Estimated Off-Block Time (EOBT). However, as the model is based on the work in [59], the model suffers from the same limitations.
[147][b]	Weibull Distributions, Gaussian Distributions, Critical Path Method	Oreschko et al. [147] build upon the method proposed in [59], and expand the model to incorporate factors such as aircraft type, airline, flight distance and incoming delay to more accurately predict the duration of the turnaround process. While this contributes to the accuracy of the model in a positive way, it remains uncertain whether the model is able to accurately describe the complex nature of the turnaround process.
[224]	Serial Process Neural Network (SPNN)	Zhe et al. [224] present a serial process node ANN approach to estimate the duration of the flight support process, where each node is only executed after the previous node showing resemblance to the PSAP approach presented by Van Hassel in [81]. In the model, three parallel processes are defined: (1) cabin service, (2) baggage/cargo (un)loading and (3) maintenance. As passenger boarding can only commence once fueling is finished, Zhe et al. [224] consider aircraft fueling to be part of the cabin service process. In the model, a serial process ANN is trained for every parallel process on data from a large hub airport in the south of China. Subsequently, the results of these three models serve as an input for another ANN which predicts the total in-block time based on the predicted duration of the three processes. As a comparison, Zhe et al. [224] train a fully connected "regular" ANN. The serial process node ANN approach shows more stable results and a small increase in the predictive capabilities in the error range of 2-minutes, while a negligible larger error is obtained for errors within 5-minutes. For higher delays, Zhe et al. [224] note that the serial process node ANN performs worse than the fully connected ANN.
[225][c]	Artificial Neural Networks (ANN)	Zhou et al. [225] present a deep learning approach using Gated Recurrent Units (GRU) to predict the departure time of a flight. Zhou et al. [225] train the model on data from a spoke airport in eastern China, comparable in size and passenger numbers to London Luton. In the model, Zhou et al. [225] take the following into account: basic flight information (e.g. actual landing time, actual departure time), airport factors (e.g. number of flights arriving and departing), weather and airline factors. This indicates that variables directly influencing the turnaround process are not taken into account. With respect to standard ANNs and Long Short-Term Memory (LSTM) neural networks, Zhou et al. [225] find that neural networks with GRUs show higher predictive performance on their data sets.
[117]	Graph Convolutional Imputation Neural Networks, Bidirectional Sliding	In their model, Liu et al. [117] use data from a large hub airport in the south of China, which has been collected from A-CDM, video analysis, RFID sensors and manual recording. However, they note that the flight ground service data may consist of abnormalities or missing data due to the sensor equipment or manual input errors. In order to mitigate the error associated with these factors, Liu et al. [117] present a graph convolutional neural network imputation and bidirectional sliding mechanism to deal with missing values and/or data suffering from abnormalities. Next to this, Liu et al. [117] prove that this model approach adds more robustness and accuracy to the model, and therefore has a higher predictive power than alternatives.
[228][d]	Mixed Density Networks (MDN), Random Forest (RF)	Zoutendijk and Mitici [228] present an approach to obtain the probabilistic distributions of the flight delay of individual flights using machine learning techniques, in a similar fashion to Mori [137]. However, in contrast to Mori [137], Zoutendijk and Mitici [228] do not determine the parameters of one specific distribution. Rather, Zoutendijk and Mitici [228] take two different approaches. The first considers with the use of Mixed Density Networks (MDN), a combination of ANNs and Gaussian Mixture Models (GMM). Consequently, the flight delay probability distribution for an individual flight becomes a weighted sum of Gaussian distributions. Next to this, Zoutendijk and Mitici [228] utilise the Random Forest (RF) algorithm. Here, as every tree in the forest yields a model output, the probability distribution can be obtained from the contributions of individual trees.

Continued on next page

Table 3.1 – Continued from previous page

Reference	Model foundation(s)	Short summary & remarks
[229]	Artificial Neural Networks (ANN)	Zuo et al. [229] propose an ANN approach optimised by the genetic algorithm, a randomised search method based on the biological principles of evolution, in order to limit the instability and improve the accuracy of ANNs. For this, Zuo et al. [229] utilise the genetic algorithm in the training phase of the model to optimize the parameters of the ANN in an attempt to increase the robustness of the model. In the paper, two models are trained on ground support data from a large hub airport. The first ANN model aims to predict TOBT before flight arrival, considering four input features. In the second ANN nine features are considered, with the aim of predicting the TOBT when the aircraft arrives in-block. Unfortunately, Zuo et al. [229] do not disclose these specific features. Nonetheless, Zuo et al. [229] find that the departure time of a flight (and the corresponding level of congestion on an airport) influences the accuracy of the TOBT. Next to this, Zuo et al. [229] show the superiority of state-of-the-art machine learning models over empirical statistical methods.
[196]	Extra-Trees, Random Forest (RF), Adaptive Boosting (AdaBoost), Gradient Boosting, Multilayer Perceptron (MLP), Deep Neural Networks (DNN)	Thiagarajan et al. [196] present a unique two-stage machine learning model approach for the prediction of the on-time-performance of flights. The first stage aims to binary classify the occurrence of flight delays. This result is fed into the second stage, a machine learning regressor, to predict the duration of the delay. Thiagarajan et al. [196] compare various machine learning models trained on historical data from US domestic airline on-time-performance and weather data for both stages, with the aim of identifying the highest performing selection of models. Thiagarajan et al. [196] find that the combination of the gradient boosted classifier and extra-trees regressor show the highest accuracy. However, while Thiagarajan et al. [196] take into account the effects of weather on flight departure delay, they do not consider any features related to the aircraft turnaround process. It is therefore uncertain whether their findings in model selection translate well to the problem at hand.
[122][a, b]	eXtreme Gradient Boosting (XGBoost), Random Forest (RF), Decision Trees, Agent-Based Modeling (ABM), SHapley Additive exPlanations (SHAP)	In their work, Luo et al. [122] aim to forecast the duration of the turnaround process (i.e. regression) and confirm aircraft off-block adherence (i.e. classification) in two separate models. Next to this, Luo et al. [122] note that the automatic identification of turnaround subprocesses by computer vision techniques are prone to false detections and blank detection. Therefore, to provide a complete data-set alternative to the actual data provided by an European airport, an Agent-Based Modeling (ABM) model is constructed that simulates the turnaround (sub-)processes based on actual data and domain knowledge. Unfortunately, due to data confidentiality, no comments are made on the performance of the ABM model. To forecast the duration of the turnaround process and confirm off-block adherence, Luo et al. [122] train various tree based models on the synthetic and available data. As a side result, Luo et al. [122] find that splitting the (sub-)process durations in separate features (e.g. first PAX in, last PAX in) yields higher model performance.
[121][a, b]	Linear Regression, Decision Trees, Random Forest (RF), Principal Component Analysis (PCA)	Following the work in [122], Luo et al. [121] present a data-driven fusion of turnaround subprocesses to make a prediction on the aircraft ground handling duration. To reduce the variables at hand, Luo et al. [121] use Principal Component Analysis (PCA), a mathematical dimensionality reduction method such that the large data-set can be characterised by a smaller number of variables. Next, the fusion model aims to integrate the sequential information on the (sub-)processes into the regression model. As regression models, Luo et al. [121] consider linear regression, decision trees and random forests to predict the duration of the turnaround process.
[7][b]	Fuzzy Logic, Critical Path Method	Asadi and Fricke [7] employ fuzzy logic to predict the turnaround time of a flight. First, Asadi and Fricke [7] transform the probability distributions of (sub-)processes (e.g. as done in [59, 6]) into a cumulative density function, which is mathematically equivalent to the fuzzy membership function (section 4.4). Next, Asadi and Fricke [7] combine fuzzy logic and the critical path method to make an estimate of the turnaround time of a flight, taking into account the main turnaround processes. However, the approach suffers from some significant limitations. For example, only triangular and trapezoidal fuzzy membership are considered in the model, yielding an inherent loss of accuracy, as not every cumulative density function can be accurately described by such functions.

[a] Eurocontrol
[b] Air Transport Technologies and Logistics, Dresden University of Technology
[c] Department of Traffic and Transportation, Nanjing University of Aeronautics and Astronautics
[d] Faculty of Aerospace Engineering, Delft University of Technology

3.2. Discussion on the State-of-the-Art in the Field & Research Gaps

From the overview in Table 3.1, various conclusions can be drawn. The overview shows that in the last decade, researchers in the field have opted primarily for machine learning algorithms to capture the complex under-

lying system. As model foundations, most researchers opted for (derivatives of) ANNs and tree-ensembles. These models, in itself, provide little insights into the underlying processes as the interpretability of these models is low. In term, this indicates that researchers in the field are more interested in the accuracy of their predictions, rather than insights into the underlying system, as they trade accuracy for interpretability. Next to this, Table 3.1 shows that little attempt is made to extract the driving factors from the underlying system post-hoc.

Some researchers, e.g. Van Hassel [81] and Zhe et al. [224], split the turnaround process into smaller sub-processes and explicitly model the relationships between them. Subsequently, they make an estimation of the cycle time of the various processes and evaluate the critical path, yielding the total turnaround duration. However, in this attempt to add interpretability to the model, the driving factors behind the duration of these sub-processes remain unknown, as no attempt is made to gain insights into these processes as this requires turnaround process data (e.g. obtained via camera's and/or manual input). Most researchers do not have this data available to them, and therefore opt to implicitly model the turnaround duration via features that are not directly related with the turnaround process.

Therefore, various research gaps can be identified. For example, in the studies in Table 3.1, little incentive is made to obtain insights into the turnaround process and how this process relates to other processes, nor are researchers actively aiming to identify the root causes of delays from the turnaround process. Researchers mainly focus on prediction accuracy, by employing state-of-the-art machine learning models, significantly limiting the interpretability of the model. Next to this, often a limited scope is considered. None of the studies consider a wide variety of factors at once, e.g. factors such as the performance of individual ground handlers or the influence of the passenger process are generally not accounted for at the same time.

Additional scoping by narrowing down the research area is required to keep the number of variables manageable. After all, a significant number of variables affecting the on-time-performance of flights presented are subsection 2.2.1-2.2.5. This additional scoping also keeps the research relevant and focused. The refined research scope is presented in Figure 3.1, and shows that taxi-in and taxi-out have been removed from the research scope, as well as start-up/push-back. Researchers in the field have devoted little attention to identifying the driving processes and factors in flight turnaround delay prediction, leaving a significant research gap. Therefore, taxi-in, taxi-out and start-up are opted to be removed. This to prevent the research from becoming too broad and unfocused, and contribute to the gap in knowledge.

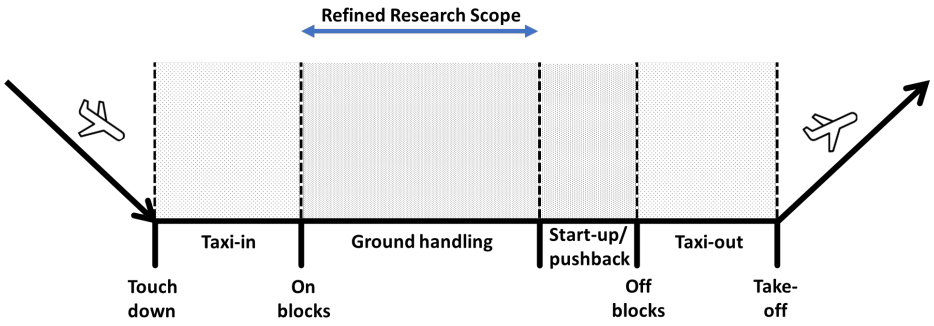

Figure 3.1: Refined research scope (figure based on work from Van Hassel [81])

4

Predicting Turnaround Delays: Machine Learning Techniques

Machine learning algorithms allow computers to learn the general trend and/or patterns from a set of data without the need of explicitly modeling the underlying (complex) systems. In this chapter, only supervised learning techniques are considered. Supervised machine learning algorithms consider two steps: (1) training and (2) inference [101]. In the training phase, the machine learning model is trained on historical data. In this phase, the algorithm sets its internal rules, weights and other parameters to make predictions while taking into account the hyperparameters set by the user (e.g. maximum depth of a tree). The second phase, inference, considers with extracting predictions from the trained model when subjected to a set of input features. Relevant example model outputs are continuous numeric, binary, multi-class or probability distributions (e.g. in [228]).

This chapter considers among other things with supervised learning algorithms, and aims to provide explanations on, and insights into the models used in chapter 3. Next to this, the chapter presents other machine learning algorithms which show potential in solving the problem at hand. This chapter is structured as follows. First, in section 4.1, a comment is made on the interpretability and explainability of machine learning models. Second, ante-hoc explainable machine learning models are elaborated on in section 4.2. Third, Bayesian Networks are explained and discussed in section 4.3. Fourth, a machine learning algorithm that does not rely on the absolute truth and can deal with linguistic terms, fuzzy logic, is elaborated on in section 4.4. Fifth, black-box machine learning models are discussed in section 4.5. Last, in section 4.6, model validation techniques are discussed.

4.1. A comment on the Interpretability & Explainability of models

In literature, authors making claims about the interpretability of models often do not define interpretability, nor state why it is of high importance in some applications (e.g. in [161]). Lipton [115] identifies two possible causes for this: (1) the definition is universally agreed upon and nobody had the opportunity to set it in stone or (2) the term interpretability is ill-defined. Lipton [115] found the latter case to be true, and other critical literature agrees [42]. As a result, there is no clear definition or motive for the concept of interpretability and it may refer to more than one concept [115, 69]. Therefore, in order to compare the interpretability of models qualitatively, a formal and consistent proxy of interpretability is required.

Doshi-Velez and Kim [42] define interpretability in the context of explainable machine learning as follows: *"the ability to explain or to present in understandable terms to a human"*. In [132], interpretability is defined as the degree to which a human can understand the cause of a decision made by the model, given the context of the model. According to Kim, Khanna, and Koyejo [103], interpretable models are models that allow one to correctly and efficiently predict the model's results. In this literature study, interpretability is defined as follows: *interpretability, as a passive characteristic of a model, defines the degree to which the inner workings, i.e. the decision or prediction process, of the model are understandable by an expert in the field.* This means that no additional tools are required to understand the models inner workings, and requires the models inner

workings to be transparent. Different dimensions of interpretability exist [42]. For example, a model may be globally and locally interpretable. In case of the former, the user is able to understand the whole logic of the model leading to all possible model outputs. Local interpretability entails that a user is only able to predict the model output for a single case as the model may contain too many variables for a human to comprehend the whole logic. After all, the interpretability degrades with the number of input features [93].

Interpretability is often used interchangeably with explainability (e.g. in [132]), while literature suggests there are arguments to distinguish between the two [69]. In this literature study, interpretability and explainability are not considered as interchangeable. Instead, explainability is defined as follows: *explainability defines the degree to which the decision or prediction process of the model can be clarified or detailed post-hoc in human terms by experts in the field.* Here, experts may consider the use of additional tools. Hence, an explainable model is one from which (parts of) the decision or prediction process of the model can be extracted post-hoc by an expert in the field and his tools, and does not require the model to be ante-hoc interpretable.

In the upcoming (sub)sections, various machine learning algorithms are discussed. Please note that the main sections have been ordered based on their simulatability by a human, starting with simulatable ante-hoc interpretable models in section 4.2. Next, Bayesian Networks and Fuzzy Logic, more sophisticated but less simulatable ante-hoc intepretable systems, are discussed in section 4.3 and section 4.4 respectively. Last, non-simulatable black-box machine learning algorithms are discussed in section 4.5.

4.2. Simulatable Ante-hoc Interpretable Machine Learning Models

In this section, a selection of ante-hoc interpretable machine learning models are elaborated on. Ante-hoc interpretable machine learning models offer interpretability (primarily transparency) at the cost of accuracy, and subsequently are less complex as their black-box counterparts, to be discussed in section 4.5. This section will elaborate on three ante-hoc interpretable machine learning models. First, in subsection 4.2.1, multiple linear regression is elaborated on. Second, subsection 4.2.2 goes into detail about logistic regression. Third, subsection 4.2.3 discusses decision trees. Fourth, Naive Bayes, of which its foundations lay in Bayes theorem, are elaborated on in subsection 4.2.4. Please note that models such as K-nearest neighbours or decision rules are not discussed in this section. This is because it is expected that when these models are subjected to a highly complex system, the inherent transparency of such models is lost due to the sheer number of variables at hand [93].

4.2.1. Multiple Linear Regression

Multiple Linear Regression aims to predict the target variable using a bias term, set of weights w and set of features x. The target variable is the weighted sum of the various feature values and their respective weights and a bias term, as the model assumes linear dependence or approximate linear behavior between the input features and target variable [93, 99, 8, 82]. Therefore, as the target variable is simply the linear sum of the elements in the input vector $\boldsymbol{X}$ and weights vector $\boldsymbol{W}$, the interpretation of the model is rather straightforward. The mathematical formulation is presented in Equation 4.1 [82].

$$\hat{f}(X) = w_0 + \sum_{i=1}^{p} x_i w_i \tag{4.1}$$

Here, w_o is the bias term, x_i a feature value from input vector $\boldsymbol{X}$ and w_i the corresponding weight for feature x_i from the set of weights $\boldsymbol{W}$. The mathematical formulation shows that multiple linear regression models are not able to capture feature interactions, as it is simply the sum of the weighted feature values. Therefore, multiple linear regression models require the features in input vector $\boldsymbol{X}$ to be non-correlated. However, interaction terms or the use of regression splines may overcome this limitation [135]. For example, one might define x_i as the product of x_j and x_k. The model remains linear in its parameters while the terms itself might not be linear, as it is still a weighted sum of the mathematical parameters [82]. The inclusion of higher order terms also does not violate this property, allows for the modelling of non-linear systems and often leads to better prediction performance in the region of interest [94]. However, it should be noted that an increase in the polynomial degree may yield worse performance than lower degree polynomials, as the model tends to overfit on the training data. Furthermore, higher degree polynomials also become unstable near the boundaries of the training data [43].

The set of weights $\boldsymbol{W}$ is often obtained from ordinary least squares [82, 135]. Ordinary least squares aims to minimize the sum of squared errors between $\hat{f}$ and the training data (i.e. observations, required to be independent), given by a set of input variables and their respective observation y.

Multiple linear regression provides an inherently transparent model, as the weights and relationships can be directly observed and are easily understood [135]. Moreover, multiple linear regression is able to model non-monotonic relationships. Here, an increasing monotonic function satisfies $\hat{f}(x_i) \leq \hat{f}(x_{i+1})$, while for decreasing monotonic functions $\hat{f}(x_i) \geq \hat{f}(x_{i+1})$ holds. However, the model suffers from some key downfalls. Linear regression is unable to deal with non-linear complex problems and does not provide the best explanations [192, 116]. Next to this, the model describes a function over the entire domain, while for some parts of the domain this might not be the perfect fit. This downfall can be mitigated by implementing Multivariate Adaptive Regression Splines (MARS), an expansion of the recursive partitioning regression strategy and additive modeling approach presented by [136, 24] and [63] respectively [60].

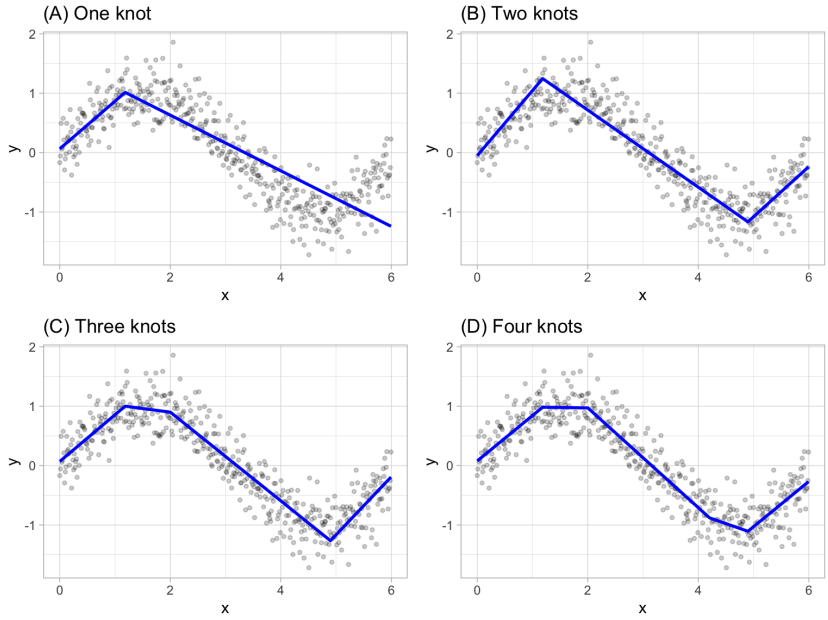

Figure 4.1: Visual example of the forward stage of the MARS algorithm [19]

The general idea of MARS is as follows [19, 82, 60]. MARS first constructs a basic function over the entire domain. Next, the algorithm splits the domain into two sub-domains. Here, MARS splits the domain on cutpoints (knots), where the largest decrease in training error (e.g. MSE, R^2) can be achieved. This process is repeated until a minimum increase in model performance is observed or the maximum number of knots M is obtained. On every cutpoint, a basis function is constructed for either side of the knot. A visual example of this process is presented in Figure 4.1. This stage, also called the forward stage, therefore considers with discovering a set of simple piecewise linear functions over the domain.

The result of the forward stage may be prone to overfitting on the training data [60]. Therefore, a backward stage is considered. In this stage, one by one every basis function is removed from the model and the remaining model is evaluated. The term that yields the lowest increase residual squared error is removed from the model [19, 82]. The removal of redundant terms simplifies MARS's output, increasing the interpretability of the model.

MARS is well suited for problems with a large number of variables, is able to deal with correlated features, performs automated feature selection and is able to deal with non-linear complex problems [82]. Correlated features do not necessarily hurt model performance, but can make model interpretation difficult [19]. When considering two highly correlated features, MARS selects only one of these two variables. This is due to the fact that the second feature will contribute significantly less to the overall objective. Consequently, this de-

creases the applicability of MARS as it is assumed that some input features will be highly correlated which MARS is not able to adequately identify and subsequently may yield wrong or inaccurate conclusions. Next to this, multiple linear regression and MARS require all features to be instantiated in order to obtain a prediction [162].

Multiple linear regression and MARS yield continuous model outputs, which may not be appropriate to all problems. The classification counterpart of these models is logistic regression, and is discussed in the next section.

4.2.2. Logistic Regression

In contrary to what its name suggests, logistic regression is only applicable to classification problems [10]. In "standard" binary classification using logistic regression, the model output is the probability P of a certain input set to belong to class X [193]. The complement, i.e. $1-P$, is the probability of the set of input features to belong to class Y. Logistic regression shows resemblance to linear regression, discussed in the previous subsection. Here, the primary difference lays in the model output as linear regression's output range is unbounded, while logistic regression's output ranges between 0 and 1. At the foundation of the model lies the logistic function, presented in Equation 4.2 [135],

$$\text{Logit}(\eta) = \frac{1}{1+e^{-\eta}} \tag{4.2}$$

where η is equal to the right hand side of Equation 4.1, i.e. $\sum_{i=1}^{p} x_i w_i$, and Logit(η) the probability that the set of input values belongs to class X. Here, the logit is defined as the natural log of the odds, i.e. the probability of being classified in class X divided by the probability of being classified in class Y [10, 193]. Equation 4.2 transfers the input values to the domain [0,1]. Here, the input variables do not require to be discrete, as logistic regression is able to deal with continuous, discrete, dichotomous or a mix of these variables [193]. Next to this, multinomial logistic regression expands logistic regression to multi-class classification, and is therefore able to model scenarios where more than two discrete model outputs are possible [45]. Several optimized implementations are available, such as from Scikit-learn [190, 68].

In model training for "standard" logistic regression, the objective is to assign high probabilities to instances of X and low probabilities for instances of Y [68]. This requires the best combination of linear coefficients in η, in order to maximise the probability that an instance is classified correctly. To estimate the set of coefficients, various methods are available in literature. One of these methods is the maximum likelihood estimation, an iterative process which aims to find the most probable set of weights to obtain the sample instances [193, 86]. First, all weights are assigned arbitrary values. Next, the algorithm aims to determine the direction and size of change in coefficients such that a better estimate of the observed instances is obtained. Subsequently, this model is evaluated and the process of determining the direction and size of change in coefficients is repeated, until a stopping criterion is met.

One of the main advantages of logistic regression is that it does not require linear relationships between the input and output features, due to the nonlinear transformation to the odds ratio [10]. Next to this, several methods are available in literature to obtain insights into the relevancy of the input features and to find a feature's contribution to the model outcome [193]. Moreover, logistic regression does not make assumptions on the distributions of the input features and is able to deal with a mix of input variables. Logistic regression models are *"one of the most interpretable linear machine learning models for certain class of events"* [44]. However, the algorithm may require post-hoc explanations, especially when the results are to be explained to a non-expert audience [8]. This is partly due to the more complex nature of the nonlinear mathematical operations, as the interpretation of weights becomes non-additive but multiplicative instead [135]. Next to this, logistic regression assumes linear dependency between the model output and input features, and requires little to none multicollinearity between the input values [164]. Furthermore, logistic regression shows low to average performance with respect to other ante-hoc interpretable and more sophisticated machine learning approaches in literature [194].

While multiple linear regression and logistic regression are suited for regression and classification respectively, some machine learning algorithms are able to do both while remaining simulatable. An example of which are decision trees, elaborated on in subsection 4.2.3.

4.2.3. Decision Trees

Decision Trees are hierarchical supervised learning algorithms, and can be considered as rule-based systems [116]. These systems can be applied to both classification and regression problems [192]. The basic idea is to construct a binary tree where the features are split multiple times on a cut-off threshold (e.g. smaller, equal, larger than threshold value X) at each internal node [93, 99]. This process stops when a leaf node is reached and a model output is obtained. Here, a leaf node is defined as a node that has no child nodes. The path between the root node, i.e. the start node of the tree, and a leaf node describes how a decision or prediction came about [93]. This makes decision trees transparent in the way they provide their predictions.

A decision tree can be described by a set of rules, which are interpretable by humans. The most common form being *if-then* rules, which describe a set of conditions to be met for the outcome to be true [76]. This desirable property makes decision trees interpretable as one, when subjected to a decision tree which is not disproportionately large, is able to explore and simulate the workings of the model and obtain a prediction given a set of input features. Therefore, decision trees show high simulatability and are utilized in decision making problems due to the transparency and understandability that they offer [8].

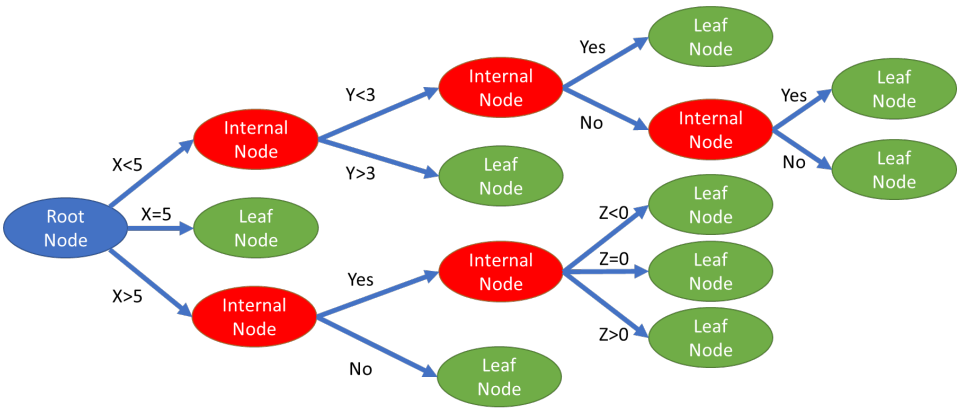

Figure 4.2: Example graphical representation of a decision tree

When considering a decision tree as a set of rules only, the relevancy of features may not become evident. Here, the graphical presentation of decision trees provides is of use [76]. An example graphical representation of a decision tree is presented in Figure 4.2. In this representation, the most relevant features are near the tip of the tree-like structure and features with less relevancy are located at the bottom. This is possible due to the internal feature selection in the decision tree algorithm, which aims to identify the most relevant features and exclude irrelevant features [82, 61]. Therefore, the hierarchical position in a tree provides insights into the relevancy of the feature [93, 76].

Decision trees are able to deal with non-linear relationships between the in- and output vectors, as well as correlations between the features and are considered as highly interpretable models in literature [69, 82]. This relationship cannot be expressed mathematically, however [93]. Moreover, as the number of levels in a tree grows, the interpretability of the model decreases [116, 93]. Next to this, different trees may be generated for the same problem and small changes in input may yield large changes in model, as decision trees lack smoothness [93, 82].

The performance of decision trees relative to other models in literature is low [116, 82, 61]. This is mainly due to the poor generalization properties (i.e. how well the model performs on data it has not seen before), as a result of overfitting [9, 82]. The low performance of decision trees can be overcome by aggregating the the prediction of smaller trees trained on different subsets from the data-set or by connecting decision trees end-to-end. This will be elaborated on in subsection 4.5.2.

4.2.4. Naive Bayes

The simplest form of a Bayesian Network classifier is naive Bayes [130]. At the core of the Naive Bayes classifier lies Bayes theorem (Equation 4.4). Naive Bayes assumes the features are conditionally independent, i.e. a

particular feature in a class is uncorrelated to any other feature in the class [16, 64, 169]. For example, an apple can be identified by the following features: (1) taste, (2) color and (3) shape. Naive Bayes assumes that all these features do not depend on each other and individually contribute to the classification whether the fruit is an apple or not. A result of this naive assumption is that the joint probability becomes the product of individual probabilities, as presented in Equation 4.3 [64]. An example network structure is shown in Figure 4.3.

$$P(X_1, X_2, ..., X_n|y_j) = \prod_{i=1}^{K} P(X_n|y_j) \tag{4.3}$$

The stochastic independence of features makes the Naive Bayes method rather scalable and useful for large data-sets, where the complexity is only $O(n)$ [169]. The assumption of conditional independence of features seems rather unrealistic in the context of this literature study. Nonetheless, the consensus in literature is that naive Bayes delivers competitive classification accuracy [210]. However, this should be taken with a grain of salt, as naive Bayes is outperformed in flight delay predictions by other machine learning classification algorithms in [227], [90] and [219].

Naive Bayes can also be adapted such that it can be applied to regression problems. Here, the distribution of the target value is modelled using kernel density estimators. However, it has not proven itself to be a proper regression method, as its performance is very poor [56]. This is likely due to the independence assumptions between the random variables.

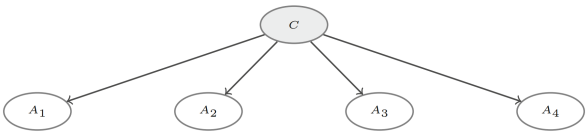

Figure 4.3: An example naive Bayes network structure [169]

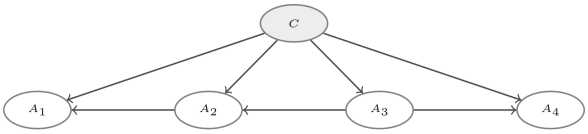

Figure 4.4: An example tree-augmented naive Bayes network structure [169]

Other forms of and/or adaptations to the naive Bayes classifier exist. Tree-augmented naive Bayes relaxes the conditional independence assumption by means of a tree structure connecting the features [210]. Here, one feature has only the class as a parent and the rest of the features have two parents. Namely, the class variable and another feature, as Figure 4.4 shows. The implementation of the tree structure aims to reduce the bias of the naive Bayes classifier [169].

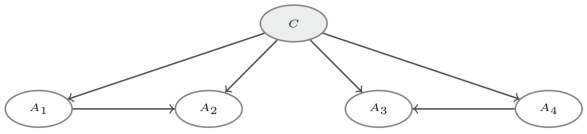

Figure 4.5: An example forest-augmented naive Bayes network structure [169]

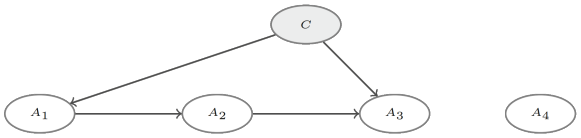

Figure 4.6: An example extended tree-augmented naive Bayes network structure [169]

An improvement of tree-augmented naive Bayes is forest-augmented naive Bayes, as presented in Figure 4.5. Here, a forest is constructed from a set of disjoint trees [169]. However, in both tree-augmented and forest-augmented naive Bayes the relevancy of features is not taken into account. Every feature is connected to the class variable, without checking whether this connection is justified. Extended tree-augmented naive Bayes aims to overcome this problem, where it allows features to have the following connections: (1) a connection to the class variable, (2) the class variable and a feature, (3) a feature and (4) no parent [169]. An example network structure of the extended tree-augmented naive Bayes is presented in Figure 4.6.

Naive Bayes classifiers are simple probabilistic classifiers and the independence assumptions make them rather interpretable. After all, the contribution of a variable towards a prediction can be observed from the conditional probability [135]. However, they show relatively poor performance in flight delay classification and do not allow one to efficiently and accurately explore the causal relationships between a large number

of variables. A more sophisticated approach is offered by Bayesian Networks, which will be elaborated on in section 4.3.

4.3. Bayesian Networks

This section will elaborate on Bayesian Networks, graphical models which are among other things utilized for obtaining a posterior distribution of a random variable based on evidence (i.e. known values of other random variables) [131]. Bayesian Networks are a powerful tool for inference, as it allows for abductive inference, i.e. finding an explanation for the set of observed random variables [31]. This property allowed Bayesian Networks to find their use in medicine in the early days. This was helped by the fact that Bayesian Networks are intuitively understandable and allow expert knowledge to be incorporated [189]. In the years after, Bayesian Networks became widely accepted models for reasoning with uncertainty [212].

This section is structured as follows. First, a mathematical introduction to classical Bayesian Networks is presented in subsection 4.3.1. Second, other relevant forms of Bayesian Networks are elaborated on in subsection 4.3.2. Next, subsection 4.3.3 and subsection 4.3.4 discuss structure and parameter learning in Bayesian Networks, respectively.

4.3.1. Theoretical framework of Bayesian Networks

Bayesian networks are probabilistic graphical models used to represent knowledge about an uncertain domain and are based on Bayes theorem, presented in Equation 4.4 [12, 218, 118]. They allow one to compute the Joint Probability Distribution (JPD) over a set of random variables and aim to model a situation in which causality plays a role but where the understanding of a system is incomplete, therefore resorting to a probabilistic approach [153, 31]. One element of a Bayesian network is the Directed Acyclic Graph (DAG), which in term is made up of a set of connected nodes (vertices) and directed edges. These represent the random variables and direct probabilistic dependencies between the random variables respectively [31].

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.4}$$

An example DAG is presented in Figure 4.7. In directed acyclic graphs, nodes are often represented by circles and the dependencies between the variables with directed edges. Within the DAG, no loop or edges connecting a node to itself are allowed as this would violate the acyclic nature of these graphs [218]. A directed edge from variable X_i to variable X_j represents a statistical dependence of X_j on X_i, i.e. the value of variable X_j depends on what value X_i takes [12]. Here, variable X_i is considered as the parent of X_j and X_j as the child of X_i. More generally, the parents of node X_i are all nodes with direct arcs pointing towards X_i and the children of X_i are all nodes which are connected via outgoing arcs from X_i [131]. The set of nodes that can be reached on a direct path from node X_i are regarded as the descendants of X_i. In similar fashion, the set of nodes from which node X_i can be reached are considered as the ancestors of X_i [72]. Returning to the DAG example presented in Figure 4.7, here D can be considered a descendant of both A and B. Therefore A and B are ancestors of D. The set of nodes that can be reached on a direct path from A and B contains [D, E, F], while for node C this only consists of [E, F].

Bayesian networks represent conditional independence statements, where a node is independent of its non-descendants given the state of the parent nodes [12, 64, 131]. When the state of the parents of node X_i is known, then the node becomes independent of all non-descendants, i.e. it is only dependent on its parents and its own descendants. Random variables (X, Y) are conditionally independent given another random variable (Z) if Equation 4.5 holds. Here, random variables X and Y are conditionally independent if for any value $Z = z$ and the probability of x is not affected when the value of $Y = y$ is known [131]. In more simple terms, given some value of variable $Z = z$ knowing the value of $Y = y$ does not give any more information about the value x random variable X takes. Returning to Figure 4.7, the conditional independence statements imply that when one has an interest in random variable F and its parent E is known, then F becomes independent of its non-descendants.

$$P(x|y,z) = P(x|z) \quad \forall\, x, y, z \text{ values of X, Y, Z} \tag{4.5}$$

More specifically, a random variable X_i is dependent on variable X_j given set of evidence nodes E if there is a d-connecting path between X_i and X_j given the set of nodes E. In order for a path to be d-connecting, every

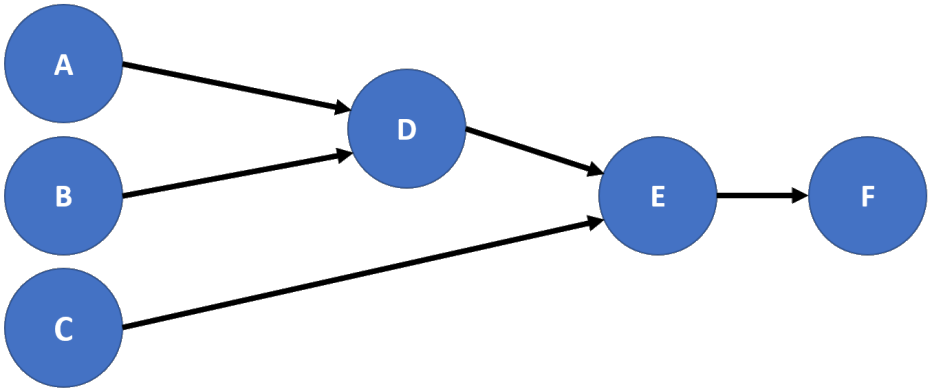

Figure 4.7: Example Directed Acyclic Graph

interior node in the path between X_i and X_j should satisfy the following properties: (1) the node is linear or diverging and not a member of the set of evidence nodes E or (2) the node is converging and the node or one of its descendants is in the set of evidence nodes E [31]. In literature, the definition of d-separation is more common. Considering two random variables X_i and X_j and a set of evidence nodes E, then E d-separates X_i from X_j if every node in the set of interior nodes satisfies one of the following properties: (1) a converging node is considered and nor it nor one of its descendants is in the evidence node set E or (2) a non-converging node is considered and the node is in the set of evidence nodes E [153]. When an interior node satisfies either condition, the path is considered to be blocked.

In traditional probability, when considering only binary variables, one requires $2^N - 1$ parameters in order to define the complete joint probability distribution [31]. Describing a JPD for a medium to large set of parameters becomes computationally expensive and significantly decreases the interpretability of the JPD [131]. The conditional independence feature of Bayesian networks decreases the number of parameters required to define the JPD of the network, which in term makes obtaining a posterior probability given the evidence less cumbersome and decreases the computational effort required [123, 12].

A Bayesian network also consists of a quantitative part [131]. The strength of the dependencies are quantified by conditional probabilities [153]. The local conditional probability is represented by a conditional probability table, one for each variable, which contains the feasible values of a child node for each combination of values of its parents [12, 64].

More formally, Bayesian networks are defined as follows [64, 12, 131]. A Bayesian network B consists of two elements, namely an annotated Directed Acyclic Graph G and the set of parameters Θ. The system aims to represent a Joint Probability Distribution over a set of random variables V. The nodes in G represent the random variables $[X_0, ..., X_n]$ and the directed edges the dependencies between variables. Here, the variables are independent of its non-descendants given its parents in G. The quantitative element consists of a set of parameters in the network, namely $\theta_{x_i|\pi_i} = P_B(x_i|\pi_i)$ for each realization x_i of X_i conditioned on π_i. Here, the term π_i denotes the set of parents of X_i in G and $\theta_{x_i|\pi_i}$ the conditional probability of X_i given its parents. The local probability distribution of root nodes, which have no parents, are unconditional. For example, A, B and C in Figure 4.7 have no parents and are therefore considered as root nodes, making them unconditional. With both elements of the Bayesian network B described, the Joint Probability Distribution over the set of random variables V can be computed with Equation 4.6 [12]. Hence, one requires the prior probabilities of all root nodes and the conditional probabilities of all non-root nodes given all possible states of their parents to obtain the JPD [31].

$$P_B(X_1, ..., X_N) = \prod_{i=1}^{n} P_B\left(X_i|\pi_{X_i}\right) = \prod_{i=1}^{n} \theta_{X_i|\pi_i} \tag{4.6}$$

As an example, the JPD of the DAG in Figure 4.7 can be computed as follows. Working from right to left, the first term one encounters is random variable F. Its only parent is node E, yielding $P_B(F|E)$ (or alternatively $\theta_{F|E}$). The next term is E, with two parents C and D, yielding $P_B(E|C,D)$ (i.e. $\theta_{E|C,D}$). In similar fashion, one

computes $P_B(D|A,B)$ for node D. As nodes A, B and C are root nodes, $P_B(X_i|\pi_i) = P_B(X_i)$. Combining previous results yields the result as presented in Equation 4.7 for standard, discrete Bayesian Networks. Relevant adaptions to the discrete Bayesian Networks are shortly elaborated on in subsection 4.3.2.

$$\begin{aligned} P_B(A,B,C,D,E,F) &= P_B(A) \cdot P_B(B) \cdot P_B(C) \cdot P_B(D|A,B) \cdot P_B(E|C,D) \cdot P_B(F|E) \\ &= \theta_A \cdot \theta_B \cdot \theta_C \cdot \theta_{D|A,B} \cdot \theta_{E|C,D} \cdot \theta_{F|E} \end{aligned} \tag{4.7}$$

4.3.2. Other forms of Bayesian Networks

Other forms of Bayesian Networks are proposed in literature. Relevant forms will be elaborated on in this subsection. It should be noted that this subsection only focuses on the high-level characteristics, and therefore extensions of these forms are not considered in this literature study. It should be noted that these forms of Bayesian Networks are more complex, and therefore not all structure and parameter learning algorithms are suited for these models. An overview of the general methodology behind structure and parameter algorithms is presented in subsection 4.3.3 and subsection 4.3.4, respectively.

Dynamic Bayesian Networks

Traditional Bayesian Networks are static and therefore only represent the probabilistic relationships between the random variables at a specific moment in time [142]. Dynamic Bayesian Networks are an extension of the traditional Bayesian Network which takes into account the temporal characteristic [38]. Dynamic Bayesian Networks are able to represent how random variable X_i is related to its own value and the values of other random variables at points earlier in time. For this, only the state of the network at time $t-1$ is required to describe the state at time t (also known as the Markov property) [142]. When a set of stochastic and time-varying random variables is considered, then Dynamic Bayesian Networks perform better than the traditional Bayesian Networks [150]. A mathematical formulation for dynamic Bayesian Networks is presented in [142].

Gaussian Bayesian Networks

In Gaussian Bayesian Networks, all random variables are considered to be continuous and the conditional probability distributions are linear Gaussians [106]. As a result, alternative representations of multivariate Gaussian distributions are Gaussian Bayesian Networks [85].

Hybrid Bayesian Networks

In most real-world scenarios, data is a combination of both discrete and continuous random variables. Hybrid Bayesian Networks (HBN) are able to deal with both discrete and continuous random variables in the model. Here, continuous random variables have discrete and/or continuous parents (assuming that the random variable has a linear Gaussian distribution), while discrete random variables can only have discrete random variables as parents [141].

Continuous time Bayesian Networks

Continuous time Bayesian Networks are an extension of the classical Bayesian Networks where the state of the nodes (i.e. random variables) evolves continuously over time [188]. Hence, the state of a node's parents influence the evolution of the node. Here, edges can be cyclic [205]. This form of Bayesian Networks are especially useful for modeling systems that evolve over time and involve continuous time events.

4.3.3. Structure learning in Bayesian Networks

The structure of the Directed Acyclic Graph can be constructed from expert knowledge in case the number of variables is limited. When the number of variables increases the construction of the DAG by hand may become rather cumbersome and one may resort to the implementation of structure learning algorithms. These algorithms aim to learn the approximate or exact structure of the DAG on the basis of data. Although the structure learning problem is NP-hard, various approximate and exact learning algorithms exist in literature [35]. Three general approaches can be distinguished, namely: (1) score-based, (2) constraint-based and (3) hybrid algorithms [71]. However, the DAG G is only one of two elements of which define a Bayesian Network. Parameter learning allows one to estimate the set Θ (i.e. the set of conditional probability distributions), given the graph G obtained from structure learning [178].

Score-based Bayesian network structure learning

Score-based Bayesian Network structure learning comprises of two steps: (1) defining a search strategy which in term aims to generate a path to follow in the search space defined by all possible graphs and (2) evaluation of each explored graph in the defined search space by use of an objective function [104, 37]. The former defines how the algorithm traverses the search space, and also how the search space can be reduced. The latter aims to assess the "goodness-of-fit" of the considered graphs, where the graph that shows the highest "goodness-of-fit" is the preferred graph [37, 178].

Many algorithms do not guarantee to return the best fitting graph from the search space and return therefore may return a sub-optimal graph structure [104]. These algorithms are considered as approximate algorithms, as there is uncertainty in whether the globally optimal graph structure is returned. Exact algorithms on the other hand always return, when subjected to the objective function, the highest scoring graph in the search space. Therefore, for exact algorithms, when defining the search strategy, one should be in particular careful to not remove the optimal solution from the search space.

Two categories in objective functions can be distinguished: (1) Bayesian scores and (2) informatic-theoretic scores. The former allows for the incorporation of prior knowledge and focuses on the "goodness-of-fit" [104]. The latter considers model complexity next to the "goodness-of-fit" in the objective function to prevent model overfitting.

Constraint-based Bayesian network structure learning

Constraint-based learning methods use conditional independence tests between the variables at hand, with the goal of obtaining a set of edge constraints and their orientation for the graph [107, 37, 178]. Therefore, this always yields a graph that is consistent with the provided data [104]. In simple terms, constraint-based structure learning algorithms aim to test for conditional dependence and independence in the data, and then find a class of networks that best explains these results [106].

Hybrid Bayesian network structure learning

In most forms, hybrid Bayesian Network structure learning are a combination of constraint-based and score-based approaches [104]. These approaches contain two steps: (1) restriction of the search space and (2) obtaining the graph with the maximum local or global score [178]. Here, constraint-based approaches are utilized to restrict the search space. Within this restricted search space, score-based algorithms subsequently find the graph with highest local or global score. This form is often referred to as restrict/maximise hybrid algorithms [104].

A more detailed explanation of most forms of hybrid approaches is as follows [178]. Considering a set of random variables X, the first step consists of finding a set of candidate parents for node X_i using conditional independence tests. The next step, assuming that the number of elements in the set of candidate parents is small with respect to the number of elements in the set of random variables X, is to obtain the network structure in the reduced and more regular search space subjected to the constraint that the parents of random variable X_i should be in the set of candidate parents.

Discussion

Many Bayesian network structure learning approaches exist. An overview of the evolution of Bayesian Network structure learning algorithms is presented in Figure 4.8. Among other things, the figure shows the class a structure learning approach belongs to, whether it is approximate or exact and whether it is able to deal with latent variables or not. Moreover, for hybrid algorithms, the figure also shows whether it is a restrict/maximise approach in nature.

Scanagatta, Salmerón, and Stella [169] note that the standard approaches to structure learning require the data-sets to be complete, as only a few methods are able to deal with incomplete data. Hence, it is quintessential to assess whether a data-set is complete before one aims to select a structure learning algorithm.

It should be noted that a structure learning algorithm for classical (and static) Bayesian Networks may not applicable to every Bayesian Network form discussed in subsection 4.3.2. For example, in Dynamic Bayesian Networks, a temporal dimension is added, which makes it rather complex to explore the structure [38]. Other

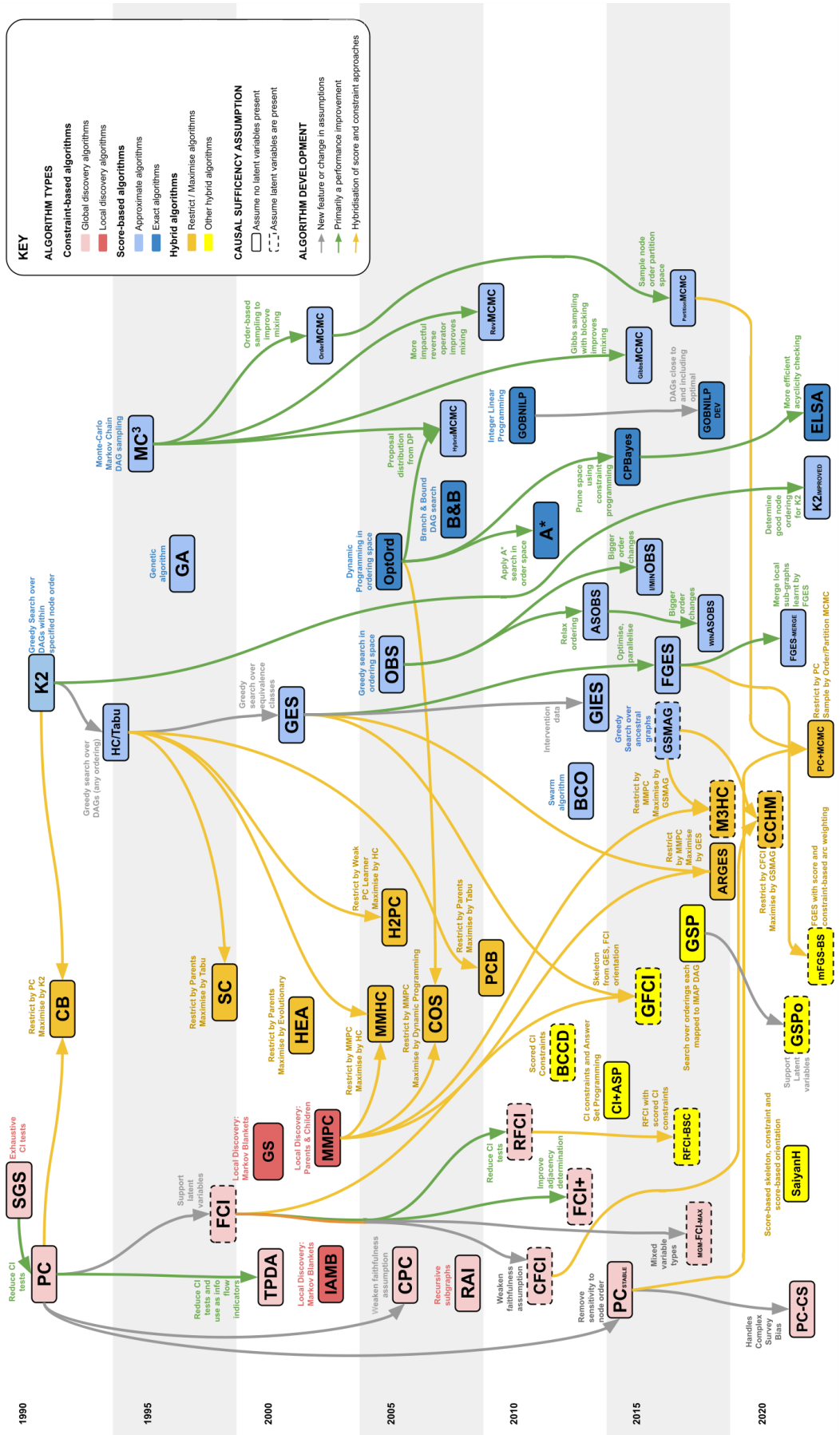

Figure 4.8: Overview of the evolution of Bayesian Network structure learning algorithms across all approaches to structure learning [104]

algorithms, not present in Figure 4.8, may be able to deal with this in a better way. Most algorithms also assume the random variables to be discrete, and are not able to directly deal with continuous variables [169]. Therefore, special attention is required in the selection of structure learning algorithms.

The main concern in constraint-based approaches is the implementation of experts' knowledge into the statistical independence tests [71]. Moreover, constraint-based approaches are sensitive to failures in individual independence tests, where one failure may have adverse effects on the construction of the graph structure [106]. Score-based approaches are less sensitive as they consider the whole structure at once, and do not dependent on individual independence tests. Here, score-based approaches are able to make a better trade-off between the "cost" of adding an edge and the dependency between the variables [106]. However, score-based approaches may not always be as efficient as constraint-based approaches.

Scutari, Graafland, and Gutiérrez [178] found that in the cases that they subjected the hybrid structure learning approaches to, the obtained results were not more accurate than constraint-based approaches. Moreover, in simulated and real-world complex data, constraint-based algorithms show a lower accuracy with respect to score-based algorithms while not offering computational efficiency benefits.

With the DAG, the set of parameters Θ that describe the strength of the dependencies between the variables can be obtained using parameter learning algorithms. This procedure is elaborated on in subsection 4.3.4.

4.3.4. Parameter learning in Bayesian Networks

The aim of parameter learning is to obtain an estimate of the conditional probability tables. After all, these tables describe the causal relationship among the variables. Here, we can distinguish two types of parameter learning: (1) approaches that require the data-sets to be complete and (2) approaches that consider data-sets to be incomplete. Here, a data-set is considered complete if for every observation n the observed value for all random variables $X_i \in \boldsymbol{X}$ (i.e. nodes in the DAG) is available. In literature, various methods are proposed, such as constraint-based parameter learning [197, 226, 53, 200], models based on expectation-maximization [222, 209], a combination of the former two [112], a combination with other methods (e.g. [191] combines expectation-maximization and genetic algorithm to converge to global instead of local optima) or new concepts such as [201, 124].

One significant disadvantage of standard Bayesian Networks is the need for discretization of continuous data. This means that all continuous variables, i.e. numerical values (e.g. number of PAX), need to be discretized such that they can serve as an input for the Bayesian Networks. In term, this yields a loss in accuracy as information is lost in the discretization process. Next to this, standard Bayesian Networks are also not able to deal with subjective features. In the next section, Fuzzy Logic is discussed. Fuzzy Logic is able to deal with subjective input features as it considers features to be part of multiple sets when no strict boundaries can be set.

4.4. Fuzzy Logic Systems

Ordinary set theory considers elements to either be part of (1) or excluded from (0) set X. This means that there is no partial truth, the elements cannot be part of both sets. In many real-life problems, it is not always clear whether an element is part of set X or not, as no clear boundaries can be set [220, 7]. For example, someone who's never seen an aircraft before might consider the first aircraft they see to be very large, while another who's travelling by plane on a daily basis reckons it is not that large and only medium in size. Here, boundaries on which to classify the airliners based on size vary and can be rather vague. Computers are not able to interpret this, as they operate on binary classification [7].

Fuzzy logic aims to mimic the way that humans analyse problems and make decisions based on vague and/or subjective values, not relying on the absolute truth [177]. This allows fuzzy logic to deal with terms as very old, old, young and very young [100]. In fuzzy logic, next to being fully part or excluded from set X, there also exists a partial truth. Here, an element is attributed a value in the interval $(0,1)$, denoting that it is partially part of a fuzzy set [220, 177, 100]. The extent to which an element x is part of a fuzzy set X is determined by the membership function $\mu_X(x)$. The mathematically possible outcomes of the membership function are summarised in Equation 4.8 [25].

$$\mu_X(x) = \begin{cases} 1 & \text{if } x \text{ is a full member of } X \\ \in [0,1] & \text{if } x \text{ is a partial member of } X, \\ 0 & \text{if } x \text{ is not a member of } X. \end{cases} \tag{4.8}$$

Fuzzy logic consists of the steps presented in Figure 4.9. The first step considers with fuzzification, the process where a crisp input is converted into a fuzzy value using the membership functions [100, 177]. This is elaborated on in subsection 4.4.1. The second step considers with inference, to be discussed in subsection 4.4.2. Next, the defuzzification process is elaborated on in subsection 4.4.3 and a discussion on fuzzy logic systems is presented in subsection 4.4.4.

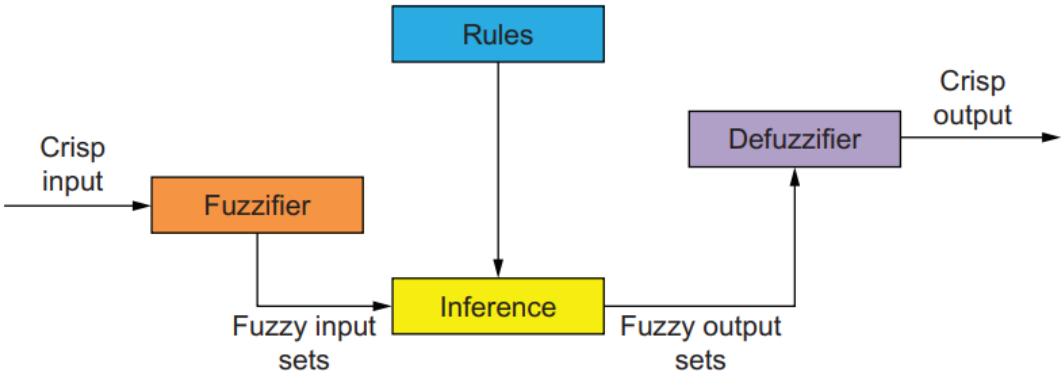

Figure 4.9: Elements of a fuzzy logic system [100]

4.4.1. Fuzzy Membership Functions

Various membership functions can be distinguished, such as triangular, Gaussian, trapezoidal, Bell-shaped, two-sided Gaussian and singleton functions [223]. Here, the triangular and trapezoidal membership functions have the most straightforward arithmetic operations [7]. Some membership functions are explained into greater detail below.

- **Triangular membership function:** the triangular membership function is described by three parameters. Namely: a and c denote the left and right bound respectively, and b defines the location of maximum height. Mathematically, it is formulated as follows [11, 223]:

$$\mu_X(x) = \max\left\{\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right\}$$

- **Trapezoidal membership function:** the trapezoidal membership function is described by four parameters. The lower and upper bound are given by a and d, and the location of the corner points is given by b and c given $a < b < c < d$. This can be mathematically formulated as follows [223]:

$$\mu_X(x) = \max\left\{\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right\}$$

  Note that when $b = c$, the triangular membership function is obtained.
- **Gaussian membership functions:** the Gaussian membership functions, namely symmetric and two-sided Gaussian, are based on the width of the curve σ and distance from the origin c. In a two-sided Gaussian membership function, the left-side curve is described by (σ_1, c_1) and the right-side curve by (σ_2, c_2). Here, $c_1 < c_2$ such that the maximum value does not exceed one [223]. Moreover, the two-sided Gaussian membership function takes the value of 1 in the interval $[c_1, c_2]$. Mathematically, a Gaussian membership function is formulated as follows:

$$\mu_X(x) = \exp\left(\frac{-(x-c)^2}{2\sigma^2}\right)$$

- **Bell-shaped membership functions:** have symmetrical shapes and are described by the parameters a, b and c which are the width of the curve, an often positive parameter and the center of the curve [223]. The mathematical expression is presented below.

$$\mu_X(x) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}}$$

- **Singleton membership functions:** is a special membership function and comparable to the unit impulse. The singleton membership function is one at $x = c$, and zero at the rest of the locations. This can be formulated as follows [36, 98]:

$$\mu_X(x) = \begin{cases} 1, & \text{if } x = c \\ 0, & \text{otherwise} \end{cases}$$

4.4.2. Fuzzy Inference

The second step, inference, aims to generate fuzzy decisions based on rules in the rule base and assigns the values to the output vector [177]. The rules in the rule base are if-then rules and describe the quantitative relationship between the variables in linguistic terms [98]. The rules are often based on experience, but can in some cases be obtained from machine learning or empirical approaches [100]. A rule can be described by its antecedent and its consequent [98]. A generic example is presented below. As the example shows, logical operators (e.g. AND, OR and NOT) can also be added to the antecedent [100].

$$\textbf{if} \underbrace{X_1 \text{ is } a \textbf{ and } X_2 \text{ is } b}_{\text{Antecedent}} \textbf{ then } \underbrace{Y_1 \text{ is } c}_{\text{Consequent}}$$

Here, it holds that if the antecedent is true to some extent, then the consequence is true to that same extent [98]. Moreover, it is possible to add more elements to the consequence. For example, one may opt to add another output Y_2 is d to the example. If more consequents are added, they are all subjected to and affected by the same antecedents [98].

4.4.3. Fuzzy Inference Systems

Three fuzzy inference systems can be distinguished, namely: (1) Mamdani-type systems, (2) Takagi-Sugeno-type systems and (3) Tsukamoto-type systems. They all have in common that the antecedents follow the same form, but the consequent element varies between the types and they vary in the way outputs come about [36, 98]. The Tsukamoto-type system will not be considered in this literature study, as they are less transparent than Mamdani- and Takagi-Sugeno-type systems [159]. In Mamdani-type systems, the output membership function is expected to output a fuzzy set [98]. For every output variable, a fuzzy set is generated which requires to be defuzzified in order to return a crisp value. In Takagi-Sugeno-type systems, the output membership functions are linear or constant [98].

The last step, defuzzification, considers with the translation from the fuzzy output to a single crisp value [100]. There are many defuzzification methods. A few applicable to the Mamdani-type inference systems are (1) Centre of Sums, (2) Center of Gravity, (3) Weighted Average Method and (4) maxima methods. Takagi-Sugeno-type inference systems do not use output membership function, and therefore do not require defuzzification. Results are obtained using weighted averages of the consequent [151].

4.4.4. Discussion on Fuzzy Systems

Fuzzy logic is a proper modelling tool to deal with qualitative terms, vagueness and expert knowledge [156]. Fuzzy logic systems in literature are often employed in control theory, e.g. for traffic junctions [152] or pH-control of flowing waste water [83]. Yet the practical application of fuzzy logic systems to aircraft turnaround problems is limited to only one academic paper: [7].

Mamdani inference systems have as an advantage that they are inherently capable of handling non-linear relationships between the model in- and output [151]. Takagi-Sugeno-type systems have this same advantage and do this via the interpolation of multiple linear systems [98]. However, Takagi-Sugeno inference systems have less expressive power and interpretable rule consequents with respect to Mamdani inference systems [151]. Therefore, they are less relevant in the context of this literature study. However, Kalogirou [98] and Pandey, Kushwaha, and Kumar [151] do not comment on how interpretability is defined. It is assumed that these authors relate the linguistic terms in the rule base to interpretability, which is rather simplistic reasoning [87]. Rule bases for accurate models may consist of many rules, limiting ones ability to fully understand the model. Here, Hüllermeier [87] draws an analogy with decision trees, where small trees are interpretable

while larger and more accurate trees lose interpretability due to the increase in number of nodes.

Differences in interpretability exist between fuzzy logic models where the rules are based on knowledge and extracted from data. The models based on the former are rather small, often made up of only a few rules formulated by an expert to formalize the relationship between variables of interest [87, 88]. Combined with the expert knowledge that generated these rules, a fuzzy logic model based on manually defined rules is rather interpretable. However, this comes at a high labour cost, even when a small number variables is considered. For models where the rules are extracted from the data-set, interpretability is lost rather quickly as the rules are computer generated which may influence its semantic interpretation [87]. When a lot of rules are generated, this basically transforms the model to a look-up table [2]. This large set of rules decreases the interpretability of the Fuzzy Logic model, and transform the system into a black-box (section 4.5).

4.5. Black-box Machine Learning Models

This section will elaborate on Artificial Neural Networks (ANN) and tree-ensemble machine learning algorithms, both widely regarded in literature as black-box models. This is due to their inherent lack of transparency, as the inner workings and decision making processes are not easily understood. However, at the cost of interpretability, they offer higher predictive capabilities and are able to capture more complex underlying systems. First, in subsection 4.5.1, Artificial Neural Networks (ANN)s are elaborated on. Second, in subsection 4.5.2, tree-ensemble based machine learning algorithms are discussed.

4.5.1. Artificial Neural Networks (ANN)

Artificial Neural Networks (ANN) inspiration originates from the information processing mechanisms present in biological nervous systems, especially from elements that consider with the information processing mechanisms in the human brain [17]. In particular, the nerve cells, called neurons [101]. ANNs, first introduced in 1943 in [127], aim to model the way human brain processes information [68]. While they are now at the core of deep learning, this has not always been the case as the computational power of computers was low and only small quantities of data were available.

Artificial Neural Networks consist of a set of interconnected nodes, which represent the neurons present in a human brain. Each of these nodes learns a simple function, i.e. mapping the input to a single output. When combining the nodes in the network, a more complex function can be described due to the emergence of interactions between these simple nodes [101]. In ANNs, three types of layers can be distinguished. Namely, (1) an input layer, (2) one or more hidden layers and (3) an output layer [204, 180]. A graphical representation is presented in Figure 4.10.

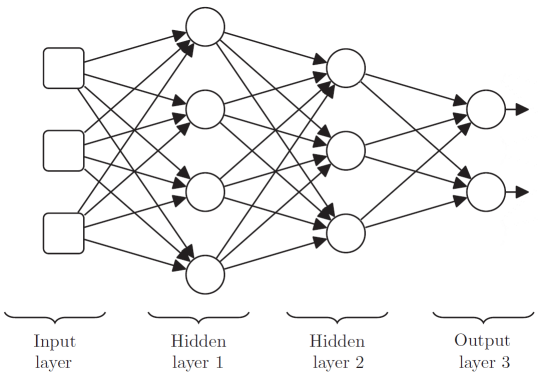

Figure 4.10: Schematic representation of Artificial Neural Networks (adapted from [101])

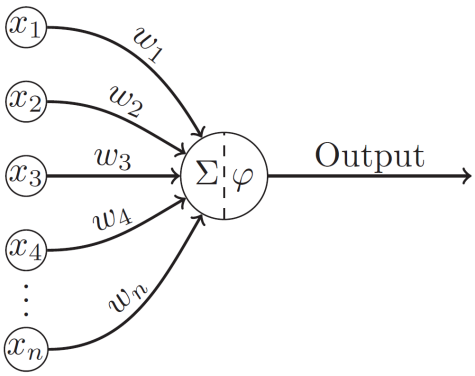

Figure 4.11: Schematic representation of the internal working principles of an artificial neuron [101]

In the input layer, the network takes in the set of variables $\boldsymbol{X}$. The nodes in this layer do not process information, but solely output the input they are fed [68]. From the first hidden layer onwards, information is processed in artificial neurons according to the principle visualised in Figure 4.11. This process consists of two stages. First, the inputs to the artificial neuron $[x_1, ..., x_n]$ are multiplied with a weight w_i after which often a bias term is added, which is analogous for the firing threshold in a biological neuron [17]. Typically, this

is represented by a bias neuron x_0, who's corresponding weight w_0 is always 1 [68]. The result of the weighted sum (the "z" value") is passed to the second stage, a non-linear activation function φ [17]. These functions allow the ANNs to capture non-linear and complex behavior of a system. Various activation functions exist. For example, hyperbolic tangent functions, logistic functions, Rectified Linear Unit Functions (ReLU) and sigmoid functions [68, 101, 70]. The result of the activation function is then passed along to the next layer in the ANN. In this manner, the information traverses through the layers of the network until the output layer is reached.

Hyperparameter Learning

ANNs allow for the use of different activation functions in different layers of the network, in order to maximise the accuracy of the model output. Within layers, generally activation functions of the same type are used [101]. The activation functions are a design parameter, set by the user (i.e. hyperparameter) before training commences. One can test various activation functions and evaluate their performance, or select the most common in literature.

The number of hidden layers in ANNs define the depth of the network. While Figure 4.10 only shows two hidden layers with four and three nodes respectively, the number of hidden layers in the network and nodes in these layers are hyperparameters set by the user before training commences [204]. Other examples of hyperparameters in ANNs are batch size, strength of weight regularization, learning rate and dropout [129, 14].

The set of hyperparameters influences the performance of the model, and therefore requires special attention. Finding the optimum set of hyperparameters is still an active field of research [204]. Various methods have already been proposed in literature, an example of which is grid search [14]. In grid search, for every hyperparameter, a set of possible hyperparameter values is defined. Next, the performance of the ANN is evaluated for every possible combinations of hyperparameters. The combination of hyperparameters that yields the best model performance is selected. The simplicity of grid search makes it simple to implement and parallelize. However, evaluating model performance for every combination of hyperparameters makes grid search rather time-consuming and computationally expensive [183]. Next to this, grid search does not perform well in practice [14].

In random search, hyperparameters are sampled from the same configuration space as defined by grid search and then evaluated on performance [14]. This means that not all combinations of hyperparameters are evaluated, making the method less computationally expensive and time-consuming than grid search. Next to this, the approach remains rather simplistic, easy to implement and parallelize. However, the method has proven to be unreliable and is rather inefficient in finding the hyperparameter optimum [55, 13]. Fortunately, in literature, other hyperparameter optimization algorithms are proposed which outperform grid and random search, such as sequential hyper-parameter optimization algorithms [13], Bayesian optimization [67], Hyperband [111] and BOHB: a combination of Bayesian optimization and Hyberband [52]. However, given the main goals of the literature study, these are not elaborated on.

Model training

The aim of the training process is to adjust the internal weights, such that the error between the trained model and the true system output is minimal [204]. The general procedure of training feedforward ANNs in more simple terms is as follows [101, 82, 204, 144, 46, 68]. The training process commences with the assignment of random values near zero to all internal weights. Next, the training set is presented to the model and forwarded through the layers until a model output is obtained (forward pass). This estimate is then compared to the true system output and the difference is quantified using a loss function C (e.g. squared errors). The next step considers with computing the gradients of the loss function with respect to the weights and biases layer by layer (backward pass), i.e. compute $\partial C/\partial w$ and $\partial C/\partial b$. These gradients are utilised to update the weights and biases in the network, and point towards the direction of steepest increase in loss function. Updating the weights can be, among other algorithms, done by the gradient descent algorithm. The process (forward and backward pass, updating the weights) is repeated iteratively until the error reaches a set minimum. Optionally, one may adjust the hyperparameters or implement procedures to improve the model performance or to prevent overfitting (e.g. via regularisation) on the training data.

Discussion

One of the main advantages of ANNs is that they allow one to model highly non-linear and complex relationships between the input features, given that the depth (i.e. number of hidden layers and nodes) of the model is sufficient. ANNs transform the input space through a cascade of non-linear transformations, which allow the model to capture the non-linear and complex relationships between the variables [204, 28]. An insufficient number of layers and/or nodes may yield unexpected and inaccurate results, as the model is not able to capture non-linear behaviors (underfit) [82]. On the other hand, too many hidden layers increases the capacity of the model, but decreases the generalization performance as the model overfits on the training data [125, 70, 82]. Generally, ANNs have the tendency to overfit rather than underfit [101]. In literature various methods (e.g. regularization and dropout) have been proposed which aim to mitigate this efficiently [70, 207].

In literature, a significant amount of attention has been devoted to ANNs in the last decade as ANNs have demonstrated state-of-the-art performances across several fields of research and offer high processing speeds [204, 17]. A share of the field's attention is devoted to the development of more sophisticated training algorithms, as training of the network is considered to be computationally costly [109]. These algorithms aim to decrease the training time and the computational expense of training the network, as well as to improve model performance and mitigate overfitting.

The transformation of the input space through a cascade of non-linear transformations together with the use of activation functions significantly limit the interpretability of the model. In literature, the consensus is that ANNs are considered to be black-boxes, as they lack transparency and are difficult to trust [99, 68]. The latter is strengthened by the fact that standard ANNs do not provide a measure of uncertainty of the model output [32, 70]. An adaptation to standard ANN, Bayesian Neural Networks (BNN), are able to provide such measure of uncertainty.

In standard ANNs, model weights and biases are considered constant and the model input random variables. The model weights and biases in classical training are often obtained from Maximum A Posteriori (MAP) optimisation [18, 213]. In Bayesian Neural Networks (BNN), model weights and biases are treated as random variables with prior distributions [70]. BNNs aim to learn a posterior probability distribution of the weights and biases using Bayes theorem (Equation 4.4), given the set of training data [214, 70]. The main benefit of this approach is that BNNs can evaluate the faithfulness and provide a measure of uncertainty of the models output [28]. Here, BNNs are able to differentiate between epistemic (i.e. lack of knowledge) and aleatoric uncertainty (i.e. due to the randomness of the observed events) [32, 125]. Standard ANNs are not able to capture this, as standard ANNs aim to find a constant set of weights and biases. One other major benefit is that BNNs represent regularized versions of standard ANN, which makes them less prone for overfitting [206]. Next to this, expert knowledge or prior information can be incorporated into BNNs [207, 125].

There are also limitations to BNNs, however. One of them being the scalability of the model, as BNNs can be computationally demanding and require large computational power to compute the posterior [125, 70]. This is also why, in the early days of machine learning, BNNs were not readily adopted. However, since then, advancements have been made to substantially increase the scalability of BNNs in literature [207, 125]. Next to this, BNNs are not inherently transparent and therefore require post-hoc explanations from for example Local Interpretable Model-agnostic Explanations (LIME) or SHapley Additive exPlanations (SHAP) [160, 120].

4.5.2. Tree-ensemble methods

Tree-ensemble methods are methods based on the aggregation of decision trees, which have already been discussed in subsection 4.2.3. The aggregation of trees makes the model rather opaque and subsequently less interpretable due to the loss of decision trees' inherent transparency. Therefore, the tree-ensemble methods discussed in this subsection are considered as black-box machine learning models.

Random Forests

Random forests utilize bagging, i.e. the act of generating multiple versions of a predictor and training them on random sample subsets of the data-set, to obtain a set of independent decision trees that can operate as an ensemble [22, 68]. Here, sampling is performed with replacement, i.e. a sample can occur more than once as samples are considered to be independent. The main idea is to average a large number noisy but unbiased

models to reduce the variance, yielding better generalization performance [9, 82, 23]. Every independent decision tree generated in the bagging process is identically distributed, yielding that the expectation of any tree is the same as the expectation of the set of trees. This also yields that the bias of an ensemble of trees is the same as an individual decision tree from this ensemble [82]. It should be noted that the independent decision trees trained on a subset of the data-set have a higher bias than one trained on the original data-set. However, the aggregation of model results yields a lower bias and variance for the ensemble of trees. The net result is that the ensemble of trees has a similar bias to a decision tree trained on the original data-set, but with a lower variance [68].

The random forest algorithm obtains it classification results by means of voting. Here, the independent decision trees will form a committee and cast a vote (i.e. class prediction). The class prediction with the most votes from the independent decision tree yields the random forest model output [68]. In regression problems, the average of the independent predictions from the set of generated decision trees yields the model output.

Adaptive Boosting (AdaBoost)

Adaptive Boosting (AdaBoost) was the first practical application of boosting (first proposed in [57]), where the main idea is to obtain a highly accurate model by combining many base-learners (in this context: decision trees) [33]. Boosting aims to build an ensemble of decision trees sequentially where the trees are connected end-to-end. At each iteration, a new decision tree is trained that attempts to reduce the error in model output of the whole ensemble [133, 68, 140]. A prediction is obtained by a weighted majority vote, of which the weights are adjusted between iterations [82]. The AdaBoost algorithm is primarily designed for binary classification problems [170]. In literature, adaptations have been proposed to make the algorithm suitable for multi-class and regression problems, such as AdaBoost.R(T) [186, 57].

eXtreme Gradient Boosting (XGBoost)

eXtreme Gradient Boosting (XGBoost) utilises gradient boosting, an adaption to boosting discussed previously. In gradient boosting, the aim is to maximise the correlation between the negative gradient of the loss function of the whole ensemble and the base learner (e.g. decision trees) [140]. This gradient allows the algorithm to focus on increasing its performance more efficiently, as it points towards the region in which more model performance can be obtained. At every iteration i, a new base learner f_i is trained which aims to predict the error of the current ensemble (pseudo response) [46]. Next, this trained base learner f_i is first multiplied by a shrinkage factor ν and subsequently added to the ensemble F_i such that the error of the ensemble is decreased [62].

At XGBoosts foundation lays Equation 4.9 [33]. Here, l is a differentiable convex loss function with a sum of two terms as input. The first aims to quantify the difference between the true state y_i and prediction about $\hat{y}_i$ at the latest full iteration $j-1$, and the second term adds to this f_j, the contribution of the trained tree structure at iteration j (i.e. the corrective term) given set $\mathbf{x}_i$. $\Omega(f_j)$ is a regularisation term, which aims to penalise the complexity of function f_j as to promote the use of simple functions. The first term in the regularisation term, γT, penalises the number of leaves T [133]. The second, $\frac{1}{2}\lambda\|w\|^2$ penalises the attribution of extreme weights. The terms γ and λ are hyperparameters, set by the user [133].

$$\mathscr{L}^{(j)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(j-1)} + f_j(\mathbf{x}_i)\right) + \Omega(f_j), \quad \text{where } \Omega\left(f_j\right) = \gamma T + \frac{1}{2}\lambda\|w\|^2 \tag{4.9}$$

The regularized objective presented in Equation 4.9 has the goal to mitigate model overfitting [33]. Next to this, in XGBoost, contains more measures to improve its generalisation performance. The shrinkage factor, analogous to the learning rate in ANNs, reduces the influence of individual base learners which in term leaves room to subsequent trees to further decrease the model error [33]. One could also implement column feature subsampling in order to mitigate overfitting. However, this is currently not supported by open-source packages [33].

Discussion

Tree-ensembles have several advantages over their individual counterparts. For example, tree-ensembles overcome the poor performance of individual decision trees and add better robustness and generalisation properties to the model [82, 8]. However, as already touched upon in the introduction of this section, the aggregation of trees makes the model rather opaque and less transparent. Consequently, this makes it more

difficult to understand model behaviour [44]. The workings of some models can be better explained than others. For example, the workings of tree-ensembles based on bagging are more easily understood than those based on gradient boosting, as gradient boosting is more complex. Next to this, XGBoost provides a generalised gradient boosting implementation, as in its fundamentals use is made of regularised loss function which takes into account the complexity of the base-learners. This makes understanding the inner workings of the model rather cumbersome. This inherent loss of transparency forces users to use post-hoc explanation techniques in order to get an estimate of the working principles [44, 8].

When comparing the tree-ensembles discussed in this section and other tree-ensembles available in literature, XGBoost has some key advantages. Namely, XGBoost is significantly more computationally efficient, has a better scalability (most importantly: ability to parallelize model training), is able to efficiently deal with missing values in a data-set, shows state-of-the-art results when subjected to a wide range of problems and shows competitive performance with respect to Artificial Neural Networks (ANN) [33]. However, this model performance is only achieved given a set of optimised hyperparameters. Various methods exist in literature for hyperparameter optimisation for tree-ensembles, some of which are also applicable to ANNs such as grid search or Bayesian optimisation. When only considering grid search, random search and Bayesian optimisation, with the latter a more optimal set of hyperparameters is obtained [155].

A model is only when it properly captures the complex system which it aims to model. To determine whether the model captures the system at hand to a sufficient extent, validation is performed. Validation and its associated performance metrics are discussed in the next section, section 4.6.

4.6. Model validation

Assessing the performance of a machine learning model is of paramount importance to determine the usefulness in the domain of application as, after all, the goal is to construct and train a machine learning model such that captures the behavior of a system and is able to make accurate predictions (i.e. within some specified tolerance) in the domain of application [221, 211]. One of the key aspects in evaluating model performance is assessing the generalization, i.e. how well it performs on input data which the machine learning model has not seen before. Quantifying model performance using performance metrics aids in determining the relative performance of the model with respect to the complex system of which it aims to capture the behavior.

This section is structured as follows. First, in subsection 4.6.1, K-cross validation will be elaborated on. Second and third, subsection 4.6.2 and subsection 4.6.3 will discuss the performance metrics applicable to classification and regression problems respectively. Next to this, sensitivity analysis can also be part of the model validation process [102]. This will however not be discussed in this section, but be elaborated on in section 5.1.

4.6.1. K-Cross validation

When sufficient data is available one may opt to allocate 70-80% of the training data towards the training of the machine learning model, and the other 20-30% towards validating the model. When data is scarce, or when one aims to prevent overfitting and desires to estimate its generalisation abilities, one can make use of K-cross validation [15]. Here, it should be noted that K-cross validation does not guarantee equal or higher model performance, as the authors of [68] demonstrate. In K-cross validation the available data is split into K roughly equal-sized chunks (folds), typically ranging from 5 to 10 folds [82]. Then, for K iterations, the machine learning model is trained on $K-1$ data chunks, where the performance metrics are evaluated on the remaining chunk of data (the test set). In every iteration, a different test set is used and hence every iteration is the model is trained on a slightly different data-set. Aggregating the results yields an overall assessment of the machine learning models performance [82]. Next to the fact that it may not guarantee equal or higher performance, K-cross validation is also computationally expensive as it requires retraining the model several times [68].

4.6.2. Performance metrics for classification models

From classification models, a non-continues response is obtained with little intrinsic meaning [108]. For example, binary classification models yield 0 and 1, indicating whether the target variable is part of a specific class or not. After the evaluation of a classification model, one might have an interest in the number of

correctly classified samples (True Positive (TP) and True Negative (TN)) and the number of wrongly classified samples (False Positive (FP) and False Negative (FN)). With these results, one can evaluate the accuracy of the model, i.e. the overall portion of correctly classified samples [204]. This can be computed with Equation 4.10 and is the total number of correctly classified samples divided by the total number of samples.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{4.10}$$

The error rate can subsequently be computed by subtracting the accuracy from 1. However, these metrics are highly sensitive to changes in data and are also can also yield deceiving results [84]. For example (from [84]), considering a set of 100 samples of which 5 are of type 0 and 95 of type 1. Naively classifying every sample as type 1 (i.e. TP = 95, FP = 5 and TN = FN = 0) yields an accuracy of 95%. However, in this case, 0 cases of type 0 are correctly classified, yet still an accuracy of 95% is obtained. Next to this, accuracy shows inconsistencies in performance representations as it is sensitive to data distributions [84].

Precision refers to the total number correctly classified positive samples, divided by the total number of samples classified positive. This is mathematically formulated in Equation 4.11 [154, 84].

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{4.11}$$

Recall (or: specificity) defines the share of positively classified samples divided by total number of positive samples, and can be computed with Equation 4.12 [108, 154]. Precision and recall share an inverse relationship, where only precision is sensitive to data distributions [84].

$$\text{Recall} = \frac{TP}{TP+FN} \tag{4.12}$$

Precision and recall serve as an input for the F-measure, or alternatively F_1, which is a measure of the effectiveness of classification given a weighted ratio of recall and precision. This ratio is set by the user, via the β term in Equation 4.13, yielding more insights into the functionality of the classifier [84]. The interval of possible F_1 scores ranges from [0, 1], where a score of 1 is attributed to models which correctly classify all samples. However, the F_1 score is less interpretable than the performance metrics discussed previously, as a value between one and zero does not have a clear description [108].

$$F_1 = \frac{(1+\beta^2)\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\beta^2\cdot\text{Recall}} \tag{4.13}$$

4.6.3. Performance metrics for regression models

Arguably one of the most famous performance metrics is the coefficient of determination R^2, a "goodness-of-fit" metric, mathematically formulated in Equation 4.14 [34, 29, 110]. The coefficient of determination describes the proportion of the variance in the target variable, given a set of input variables [34, 166]. The interval of possible R^2 values ranges from $-\infty$ to 1. A negative value of R^2 is obtained when the regression model does not follow the trend of the data, or in other words, a horizontal line is a better fit. A value near 1 is obtained when the model shows an almost perfect relationship with the data [168].

$$R^2 = 1 - \frac{\sum_{i=1}^{m}\left(\hat{f}(x_i) - f(x_i)\right)^2}{\sum_{i=1}^{m}\left(\bar{f}(x_i) - f(x_i)\right)^2} \tag{4.14}$$

However, R^2 has several limitations. For example, R^2 is inherently biased, is insufficient as a performance metric and is misleading [110]. Next to this, the applicability to non-linear problems is up to debate (e.g. conflicting views in [34] and [166]) and considered as an incorrect measure of predictive accuracy [166]. Moreover, the coefficient of determination is not capable of adequately dealing with outliers [34].

Mean Squared Error (MSE) measures the average squared distance between the model output and the true value from the data-set. Therefore, MSE is able adequately deal with outliers, as it attributes high weights to such predictions [34]. Possible MSE values are in the interval $[0,\infty]$, where higher values are attributed to models that show large discrepancies between the model output and true values from the data-set. An advantage of MSE is the fact that its units are expressed in the squared units of the target variable [208].

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{f}(x_i) - f(x_i)\right)^2 \tag{4.15}$$

Root Mean Squared Error (RMSE) is the square root of the Mean Squared Error, of which its mathematical representation is shown in Equation 4.16 [34]. Therefore, the units are expressed in the units of the target variable. However, MSE (and consequently RMSE) suffer from shortcomings and weak performance [208]. Next to this, RMSE was found to be unreliable [5]. Although, this has been disputed in [30]. While able to adequately deal with outliers, this also represents one of its weaknesses as the metric might be skewed due to only a handful of outliers [34].

$$\text{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\hat{f}(x_i) - f(x_i)\right)^2} \tag{4.16}$$

The Mean Absolute Error (MAE) performance metric, presented in Equation 4.17, attributes the same weight to all differences between the model output and true value from the data-set [30]. This makes MAE less prone to be skewed by outliers with respect to MSE [34]. However, as discussed in the preceding paragraphs, this can also be considered a disadvantage. Next to this, as is the case for RMSE, MAEs units are expressed in the units of the target variable, which makes the metric more natural to understand.

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}\left|f(x_i) - \hat{f}(x_i)\right| \tag{4.17}$$

The Mean Absolute Percentage Error (MAPE), Equation 4.18, is also less sensitive to outliers than MSE. It provides an intuitive interpretation in terms of the relative error and it has been argued that MAPE is suitable for forecasting applications when large quantities of data are available [40]. However, MAPE is biased towards low forecasts due to the nature of its denominator (i.e. when $f(x_i) \to 0$, MAPE $\to \infty$). This makes them unsuitable for evaluating model performance when large discrepancies are to be expected [34].

$$\text{MAPE} = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{f(x_i) - \hat{f}(x_i)}{f(x_i)}\right| \tag{4.18}$$

Summarising, all performance metrics discussed in this section have their advantages and disadvantages. It is therefore not surprising that no consensus has been reached on a single, unified performance metric to evaluate machine learning models [34].

5

Model-Agnostic Explainability Frameworks for Machine Learning

In recent years, machine learning models have evolved and shown the ability to capture more complex underlying systems. However, this comes at the cost of a loss in interpretability, limiting a users ability to understand the inner workings and/or decision process of the machine learning model. Consequently, this reduces a users trust in the model, as one does not understand how the predictions came about. Post-hoc explainability frameworks aim to provide insights into the inner workings or the importance of features in a prediction, and aim to restore a users trust into the predictions made by a black-box machine learning model. This chapter will elaborate on model-agnostic post-hoc explanatory frameworks (with the exception of decompositional methods in section 5.3). Model-agnostic frameworks consider the machine learning model to be a black-box, and are therefore applicable to all models considered in chapter 4.

This chapter is structured as follows. First, in section 5.1, local and global sensitivity analysis are discussed. Second, Permutation Feature Importance (PFI) is elaborated on in section 5.2. Third, section 5.3 delves into rule-based extraction frameworks. Fourth and fifth, Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP) are elaborated on in section 5.4 and section 5.5 respectively.

5.1. Sensitivity Analysis (SA)

Sensitivity Analysis (SA) is a method aimed to measure the change in the model output based on a change in input variable(s), providing information on the inner workings of a black-box model [187, 202]. Local and global sensitivity measures exist. Local sensitivity analyses have the aim of providing insights into the behavior of the black-box model around prediction point $\hat{x}$. Global sensitivity analysis analyzes the black-box function response to a change in the entire input space [202, 187].

5.1.1. Local sensitivity analysis

The most simple version of the sensitivity analysis is the one-at-a-time (or nominal range) method and is the most popular in top journals [21, 54]. The main idea of the one-at-a-time method is that given black-box model input space $[x_1, x_2, ..., x_n]$ one analyses the model output to a change in one input variable while the other input variables are kept constant [58]. The mathematical formulation is presented in Equation 5.1 [21].

$$\Delta_i \hat{f} = \hat{f}\left(x_i + \Delta x_i, \mathbf{x}^0_{\sim i}\right) - \hat{f}\left(\mathbf{x}^0_{\sim i}\right) \tag{5.1}$$

Here, $\hat{f}$ denotes the black-box model, $\mathbf{x}^0$ the base case (i.e. the input feature values of $[x_1, x_2, ..., x_n]$ at $\hat{x}$), $x_i + \Delta x_i$ a change to input feature value x_i by Δ_i, $\mathbf{x}^0_{\sim i}$ the input vector without x_i and $\Delta_i \hat{f}$ the change in black-box model response due to the change Δx_i in input feature value x_i.

From Equation 5.1 the sensitivity of the black-box model to a change in input variable x_i is obtained. Therefore, this process has to be repeated for all variables x_i in the input vector in order to obtain the sensitivity of

all variables at the considered prediction point $\hat{x}$ [21, 165]. Note that the extent of the change Δx_i along the input variables should be consistent (e.g. 10% of the base case value, one standard deviation [21, 80]) in order to compare their effects on the model output.

The effects of a change in variables $[x_1, x_2, ..., x_n]$ can be visualised using a Tornado diagram, of which an example is presented in Figure 5.1. The figure shows the change in black-box model output when subjected to a positive and negative change in input feature x_i. In the Tornado diagram, the input feature changes which yield the highest change in black-box model output are located at the top of the diagram.

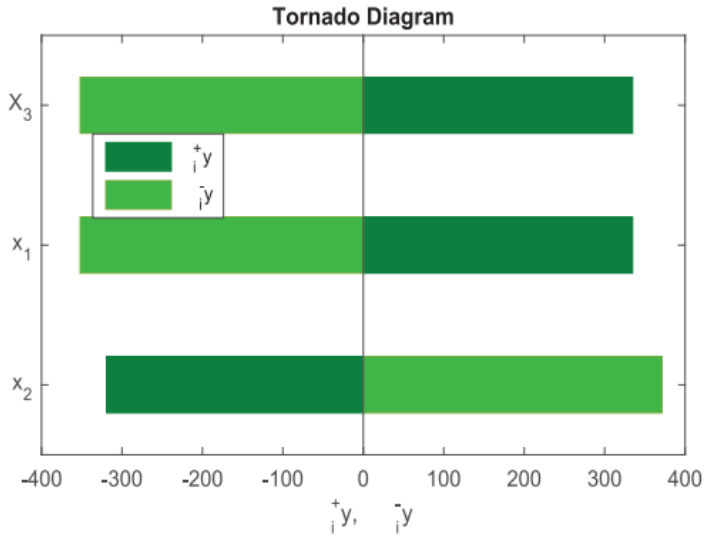

Figure 5.1: Example Tornado diagram [21]

Both the Tornado diagram and the one-at-a-time approach do not take into account interactions between features [21]. Moreover, the results are most valid when the sensitivity analysis is done on a linear model [58]. Next to this, the one-at-a-time method is inefficient when the input vector is large and only a few inputs are influential [165]. This makes the one-at-a-time technique less attractive as a model-agnostic analysis technique as feature interactions and highly non-linear systems are considered in this literature study. Frey and Patil [58] note that conditional sensitivity analysis may be an alternative to the one-at-a-time approach as it accounts for correlation between inputs or non-linear interactions in the model response. However, this method has its limitations and computationally expensive when a relatively large input vector is considered [58]. Next to this, it still does not account for interactions between features. A global sensitivity analysis, subsection 5.1.2, is able to capture these interactions.

5.1.2. Global sensitivity analysis

A global sensitivity analysis takes into account the interactions between the feature inputs as it considers a change in the entire input space, yielding more insights into the relevancy of features in the model. [58]. Here, scatterplots provide a simple and visual impression of the relative importance of features [165]. However, visual inspection of the scatterplots may be rather cumbersome, especially when a significant number of input features is considered [21]. Next to this, Iooss and Lemaître [92] note that scatterplots do not capture all interaction effects between the model inputs. This does not render scatterplots useless, as scatterplots may serve as an initial step in a broader sensitivity analysis [105].

More sophisticated and quantative approaches are offered by Sobol's indices. Sobol's indices are based on the variance of input features on the model output, and aims to quantify the variance contribution of an individual input variable on the variance of the model output [198, 21]. Sobol's indices can be represented by a sum of elementary functions and is mathematically formulated in Equation 5.2 [92, 185].

$$\hat{f}(\mathbf{x}) = \hat{f}_0 + \sum_{i=1}^{d} \hat{f}_i(x_i) + \sum_{i<j}^{d} \hat{f}_{ij}(x_i, x_j) + \cdots + \hat{f}_{1,2,\ldots,d}(x_1, x_2, \ldots, x_d) \tag{5.2}$$

Here, it is assumed that the inputs are independent [21]. Therefore, the variance can be decoupled and by substituting $Y = \hat{f}(\mathbf{x})$ the variance of Y is obtained, presented in Equation 5.3 [165],

$$\mathrm{Var}(Y) = \sum_{i=1}^{d} \mathscr{D}_i(Y) + \sum_{i<j}^{d} \mathscr{D}_{ij}(Y) + \cdots + \mathscr{D}_{ij\ldots d}(Y) \tag{5.3}$$

where $\mathscr{D}$ is given by Equation 5.4 [92].

$$\begin{aligned} \mathscr{D}_i(Y) &= \mathrm{Var}[E[Y \mid x_i]] \\ \mathscr{D}_{ij}(Y) &= \mathrm{Var}\left[\left(E \mid x_i, x_j\right)\right] - \mathscr{D}_i(Y) - \mathscr{D}_j(Y) \\ \mathscr{D}_{ij\ldots d}(Y) &= \ldots \end{aligned} \tag{5.4}$$

Next, the Sobol' indices can be obtained from Equation 5.5 [92]. Here, the first order Sobol' indices represent the individual contributions of input features on the output variance [165]. In similar fashion, second and higher order Sobol' indices represent the share of the output variance due to the interactions between input features [198, 165].

$$S_i = \frac{\mathscr{D}_i(Y)}{\mathrm{Var}(Y)}, \quad S_{ij} = \frac{\mathscr{D}_i j(Y)}{\mathrm{Var}(Y)}, \quad \ldots \tag{5.5}$$

Naturally, the first and higher order Sobol' indices sum up to one, as presented in Equation 5.6 [165].

$$\sum_{i=1}^{n} S_i + \sum_{i=1}^{d} \sum_{j>i}^{d} S_{ij} + \sum_{i=1}^{d} \sum_{j>i}^{d} \sum_{k>j}^{d} S_{ijk} + \cdots + S_{ijk\ldots d} = 1 \tag{5.6}$$

The total contribution of x_i, including all interaction effects, can be computed by summing the terms in which x_i is considered. When considering the three factor (i.e. dimensions) model example from Saltelli et al. [165], the total variance contribution of variable 1 to the total output variance (i.e. the direct contribution and the contribution of higher order interactions) can be computed with Equation 5.7. The total contribution of all variables can be computed with $S_T = \sum_{i=1}^{d} S_{T_i}$, for which holds that $S_T \geq 1$ [21].

$$S_{T_1} = \underbrace{S_1}_{1^{\text{st}}\text{ order}} + \underbrace{S_{12} + S_{13}}_{2^{\text{nd}}\text{ order}} + \underbrace{S_{123}}_{3^{\text{rd}}\text{ order}} \tag{5.7}$$

The total number of indices grows exponentially with the number of dimension, by $2^d - 1$ [21]. The calculation Sobol' indices requires therefore many model evaluations. As a result, Sobol's method is fairly computationally expensive. In case model evaluation is too costly, more global sensitivity analysis methods are available in literature such as Extended Fourier Amplitude Sensitivity Test (FAST) which are less computationally expensive [58].

5.2. Permutation Feature Importance (PFI)

Permutation Feature Importance (PFI) aims to evaluate the importance of features in a machine learning model, by measuring the increase in model error after a feature's values in the validation set have been permuted [23]. This breaks the relationship between the feature and the target variable [135]. If shuffling of a feature's values yields a higher model error, then this feature contributed to the model prediction. When shuffling of a feature's values does not yield a higher model error, then the model ignores this feature, indicating that this feature does not contribute to the prediction made by the model. When all features have been permuted and evaluated, comparing the increases in model error due to the permutation of the feature's values yields the relative importance of a feature in the model.

PFI relates to local Sensitivity Analysis (SA), discussed in section 5.1. However, there are some key differences between the two. For example, permutation feature importance randomly shuffles the values of a particular feature, while in a sensitivity analysis changes are made to the input variables. This means that, in PFI,

a permuted feature's values will not be outside the numerical domain the model has already seen in model training. This is not necessarily the case in sensitivity analyses. Next to this, PFI and SA do not have the same goal. PFI aims to rank the features on their importance in the predictions, while SA aims to determine the sensitivity to a disturbance in input variables.

PFI offers one the ability to obtain an importance ranking of features in a trained model in a rather simplistic computationally inexpensive manner, providing insights into the features that drive a model's behavior. In the process, PFI takes into account the interaction and main feature effects, since shuffling of a feature's values destroys the relationship between the specific feature and other features in the model [135]. This can be considered as an advantage but also as a disadvantage, since an increase in model error does not solely capture the effect of a feature, but also the interaction effects between the specific feature and the rest of the features. The importance of a specific feature can therefore be overestimated [135]. Next to this, correlated features may yield a misleading feature ranking. In the case of correlated features, a correlated feature's importance is split between the set of correlated features [135]. This decreases the relative importance of the features in the set of correlated features with respect other features in the model, which in term may lead to misleading conclusions.

5.3. Rule-extraction

Rule-extraction, a post-hoc explanatory framework, aims to obtain a set of rules from more complex machine learning models and are able to provide very interpretable rules for complex models. Therefore, they serve as a simpler proxy to understanding the model [116, 8]. These rules aim to approximate the internal reasoning, i.e. the relationships between the input features and target variable(s), of machine learning models [167]. In term, these relationships provide insights into the behavior of the model. In literature, rule extraction algorithms can be divided into two main approaches: (1) decompositional and (2) pedagogical [3, 77]. The former approach considers the model as a black-box, and obtains a set of rules that describe the global behavior of the model. This approach is therefore model-agnostic. The latter is model specific, as the approach analyses the individual components of the model (e.g. activity of hidden nodes [116]) to obtain a set of rules from the model [27].

Given the sheer number of rule-extraction algorithms available in literature (small overview presented in [76]), the two approaches are only discussed in very broad lines. In pedagogical approaches, rules are obtained from the model with a rule-based surrogate model [77]. The surrogate model aims to mimic the black-box model locally, requiring the surrogate model to be a good fit of the black-box model locally (i.e. show high fidelity) [116, 97]. As surrogate model, any machine learning model can be employed that is ante-hoc interpretable [97]. Here, note that this does not entail that every model provides the same explanations. Pedagogical approaches are most common in literature [91]. However, they suffer from some limitations. For example, there are no guarantees that the surrogate model will be faithful to the black-box model. In contrast, decompositional methods show perfect faithfulness to the model [97]. However, also decompositional algorithms show limitations. For example, the scalability and comprehensibility of decompositional methods is limited [97].

5.4. Local Interpretable Model-agnostic Explanations (LIME)

Local Interpretable Model-agnostic Explanations (LIME) aims to provide insights into the reasoning behind individual black-box predictions [160]. LIME trains a transparent and interpretable model locally by tweaking the feature values and observing the result of these changes on the model output [89, 161]. The behavior of the surrogate model around prediction $\hat{x}$ should be locally faithful to how the black-box model behaves in the vicinity of prediction $\hat{x}$ [160]. This means that LIME does not provide global explanations, but only in smaller regions of interest in close proximity of prediction $\hat{x}$. Ribeiro, Singh, and Guestrin [160] note that local fidelity does not imply global fidelity, and that features that are important globally may be of less importance locally and vice versa.

The mathematical formulation of the model is as follows [160, 161, 135]. Consider a set of interpretable models $g \in G$ (e.g. linear models or decision trees), where a model can provide explanation using visual or textual artifacts. As the interpretability of the models $g \in G$ varies, the creators of LIME let $\Omega(g)$ be the measure of complexity of the model. In case of decision trees, this may be the depth of the tree, while for linear models

the number of non-zero weights can be considered [160]. Moreover, LIME considers π_x as a proximity measure between prediction $\hat{x}$ obtained from black-box model $\hat{f}$ ($\mathbb{R}^d \rightarrow \mathbb{R}$) and an instance z. The unfaithfulness of g in approximating $\hat{f}$ in the local region bounded by π_x is defined as $\mathscr{L}\left(\hat{f}, g, \pi_x\right)$. The goal of LIME is to minimize the measure of unfaithfulness in approximating the black-box function $\hat{f}$ (i.e. $\mathscr{L}\left(f, g, \pi_x\right)$), and the measure of complexity, $\Omega(g)$, to ensure local fidelity and interpretability. This is mathematically formulated in Equation 5.8 [160, 161, 135]. Here, $\xi(\hat{x})$ is the explanation obtained by LIME.

$$\xi(\hat{x}) = \underset{g \in G}{\operatorname{argmin}}\, \mathscr{L}\left(\hat{f}, g, \pi_x\right) + \Omega(g) \tag{5.8}$$

Before the fidelity $\mathscr{L}$ (or loss function) can be estimated, first a set of perturbed samples ($x' \in X'$) around $\hat{x}$ is to be generated, the black-box model $\hat{f}$ be evaluated and the perturbed samples to be weighted according to proximity measure π_x [161, 182]. These serve as an input for the loss function, mathematically presented in Equation 5.9 [160]. Here x' denotes a perturbed sample, an element of the set of perturbed samples X'.

$$\mathscr{L}\left(\hat{f}, g, \pi_x\right) = \sum_{x' \in X'} \left[\hat{f}\left(x'\right) - g\left(x'\right)\right]^2 \pi_x\left(x'\right) \tag{5.9}$$

Weighting of the proximity measure follows from Equation 5.10 [182]. Here, D denotes some distance function (e.g. cosine distance, Euclidean norm) with width σ.

$$\pi_x = \exp(-D(x,z)^2/\sigma^2) \tag{5.10}$$

As the first term of Equation 5.8 can now be computed, this leaves the complexity $\Omega(g)$ of the surrogate model to be computed in order to obtain explanation $\xi(\hat{x})$. In practice, the complexity $\Omega(g)$ is set by the user, for example by selecting the maximum number of nodes in a decision tree or the maximum number of features in a linear regression model [135].

An example explanation from LIME is presented in Figure 5.2 [160]. Here, the black-box function $\hat{f}$ is visualised by the blue and pink background and the prediction $\hat{x}$ is represented by the bold red cross. The other, smaller crosses represent the generated perturbed samples. The size of the cross expresses its measure of proximity to prediction $\hat{x}$. The figure shows that the learned explanation, represented by the dashed line, is locally faithful to the black-box model $\hat{f}$.

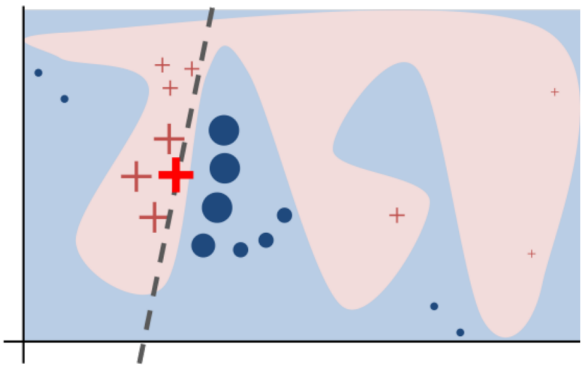

Figure 5.2: Example explanation LIME [160]

One of the main advantages of LIME is the flexibility that it offers [135]. With LIME, one can easily change the underlying machine learning model as LIME considers the trained machine learning models as black-boxes. Therefore, LIME is independent of the selected machine-learning model. Moreover, LIME gives one the ability to use a more interpretable machine learning model as local surrogate model, without using this model for the predictions. Next to this, LIME works for tabular data, text and images. However, arguably most important, is that LIME is able to provide a first estimate of the fidelity measure (i.e. how well the surrogate model approximates the black-box function at the considered point of interest $\hat{x}$) from fidelity function $\mathscr{L}$ [135, 160].

LIME also has its disadvantages. In its current form, data points are sampled from a Gaussian distribution, ignoring the correlation between features [135]. This also entails that some sampled data points may be a

combination of infeasible feature values, on which the surrogate model is trained. Next to this, in the end, the user is responsible for the compromise between the complexity of the surrogate model and interpretability. Moreover, it is possible that LIME does not always provide the same explanations as the perturbed samples on which the surrogate model is trained vary in every sampling process [135]. This may lower one's trust in the explanations provided by LIME.

Next, SHAP is discussed in section 5.5. SHAP is also an additive feature attribution method, and its explanations are build on a common foundation such that its explanations are consistent, a feature that LIME lacks.

5.5. SHapley Additive exPlanations (SHAP)

SHapley Additive exPlanations (SHAP) mathematical foundations are based on coalitional game theory [9]. To be more specific, on Shapley values. In cooperative game theory, Shapley values describe the marginal contribution of every player in the game towards the end result. In the context of this literature study, features are considered as players and the model output is considered as the payout. Therefore, the Shapley value of a certain feature X is its average marginal contribution across all possible coalitions of features [135]. The mathematical formulation to calculate Shapley values is presented in Equation 5.11.

$$\phi_i(\hat{f},\hat{x}) = \sum_{z' \subseteq X'} \frac{|z'|!\,(M-|z'|-1)!}{M!} \left[\hat{f}_{\hat{x}}(z') - \hat{f}_{\hat{x}}(z' \setminus i)\right] \tag{5.11}$$

Here, ϕ_i denotes the Shapley value of feature i at prediction point $\hat{x}$ and M the total number of features. To calculate the Shapley value for feature i, one iterates over all possible subsets z' (i.e. coalitions, all permutations of the input features) as to account for the interaction between feature values [41]. It should be noted that in almost all subsets $z' \in X'$ one or more input features of the black-box model is excluded. Considering feature i, the black-box model is evaluated with ($\hat{f}_x(z')$) and without feature i ($\hat{f}(z' \setminus i)$) in subset z'. The difference between the black-box model outputs yields the contribution of feature i towards the payout in subset z'. Next, this result is weighted with the number of features in the subset (fraction in Equation 5.11). Here, the contribution of feature i is attributed a higher weight in case many features are already included. On the other hand, small correlations are also attributed higher weights, as to isolate the effect of the features on the model output [41].

In the process of calculating the Shapley values, the input vector of the black-box model remains constant. When a subset z' with less elements than the input vector is considered, the missing elements are replaced by random values from the features in the trained data-set as to sample out the relevancy of these features [41].

In essence, both LIME and SHAP are additive feature attribution methods [120]. Shapley values satisfy three properties: (1) local accuracy, (2) missingness and (3) consistency. Whether these properties are satisfied by LIME, depends on the loss function $\mathcal{L}$, proximity measure π_x and complexity measure Ω [120]. LIME obtains the parameters for these functions heuristically [160]. In term, this violates the local accuracy and consistency properties, which, when subjected to certain circumstances, might lead to unintuitive behavior [120].

In Kernel-SHAP, LIME equations 5.9 and 5.10 are adapted such that they satisfy the local accuracy and consistency properties. In Equation 5.12 and Equation 5.13 respectively, the loss function and proximity measure equations that satisfy the Shapley value properties are presented. Here, $|z'|$ denotes the number of non-zero elements in z' and it is assumed that $g(z')$ follows a linear form. Moreover, complexity $\Omega(g)$ is set to zero, eliminating the term from Equation 5.8. Therefore, Shapley values can be obtained from a weighted linear regression in a model-agnostic manner [120].

$$\mathcal{L}(\hat{f}, g, \pi_{x'}) = \sum_{z' \in Z} \left[\hat{f}(h_x^{-1}(z')) - g(z')\right]^2 \pi_{x'}(z') \tag{5.12}$$

$$\pi_{x'}(z') = \frac{(M-1)}{(M \text{ choose } |z'|)\,|z'|\,(M-|z'|)} \tag{5.13}$$

Shapley values, as part of cooperative game theory, have a strong theoretical foundation [135]. Moreover, SHAP explanations are consistent [184]. This is an advantage with respect to LIME, where local explanations may be inconsistent as they lack a common foundation [135]. This consistency property has as an added

benefit that one can make up global explanations from local explanations [41]. One obtains a global explanation by averaging the Shapley values for all instances in the data-set used to train the black-box model [187]. Here, SHAP not only provides the relative importance, but also the direction of change on the model output (i.e. target variable(s)). This cannot be done with LIME, as it is solely a local explanation method [160].

However, SHAP also has its disadvantages. Kernel-SHAP, the model-agnostic version of SHAP, is computationally expensive when Shapley values for many instances are computed [9, 135, 139, 187]. This makes Kernel-SHAP less attractive. However, a more computationally efficient SHAP framework exists for tree-ensemble methods, TreeSHAP [217]. Moreover, SHAP assumes features to be independent [9]. In the context of this literature study, it is highly likely that this assumption is violated leading to more weight being attributed towards data-points that are unlikely [135]. The extent of violation of this assumption is unclear, however.

6

Research Proposal

This chapter elaborates on the research proposal, following the work presented in this literature study. First, the research objective is presented in section 6.1. Second, the set of research questions that together aim to build the knowledge to achieve the research objective is presented in section 6.2.

6.1. Research Objective

A research objective sets the goal of a study, and subsequently defines the purpose of a research project. The analysis of the state-of-the-art in flight departure delays in section 3.1 shows that, in the last decade, researchers in the field have devoted little attention to obtain insights into the underlying processes of an aircraft turnaround when predicting flight departure delays. Instead, they have opted for machine learning models with limited interpretability. In term, this yields higher model accuracy. However, these models are not able to describe their decision making process, nor the relationships between the input variables. This limits one's trust in a model, and subsequently does not allow one to identify bottlenecks in an aircraft turnaround and discover the causalities between factors and processes. Next to this, none of the researchers in the field have proposed a model that is able to evaluate the scheduled ground handling end time[1] adherence at intervals in an aircraft turnaround, in an interpretable and/or explainable manner. This is, as noted in section 3.2, presumably due to the lack of data available to the researchers in the field. After all, developing such model requires data from sources such as camera's in combination with advanced computer vision and manual inputs on turnaround events, leaving this area of research relatively uncovered. This research aims to traverse these uncovered grounds, and its associated research objective is presented below. The next subsection, section 6.2, breaks down the research objective into specific questions which relate to how the objective can be achieved.

Research Objective

To develop and evaluate an interpretable and/or explainable machine learning model to predict scheduled ground handling end time adherence at intervals at Amsterdam Airport Schiphol

6.2. Research Questions

Research questions aim to keep the research focused, by breaking down the research objective into specific questions related to how the research objective can be achieved. Together, the research objective and research questions help to define the explicit contribution and relevance of a research to the field. The research questions associated with this research are presented below this paragraph.

(i) **What key factors have a significant influence on a timely execution of the aircraft ground handling processes for flights departing from Amsterdam Airport Schiphol (AAS)?**

(a) *What underlying processes and factors influence the ground handling duration at AAS?*

[1] Available ground handling time is defined as follows. For on-time inbound flights: Scheduled In-Block Time (SIBT)-scheduled TOBT. Delayed inbounds: Minimum Turnaround Time (MTT).

(b) *How do external factors, e.g. weather, airport infrastructure and belated arrival of aircraft, interact with the underlying processes and factors under (a)?*

(c) *What is the relationship between the various ground handling processes and/or factors considered under (a) at AAS?*

(ii) **How can the scheduled end of ground handling time adherence be predicted using interpretable and/or explainable machine learning?**

(a) *How can historical data, obtained from e.g. A-CDM and Deep Turnaround, be utilised in model development?*

(b) *How can the various factors and variables, e.g. time-of-day, ground handler performance, aircraft type and airline business model, be incorporated into the model?*

(c) *How can the causalities between the variables be modelled?*

(d) *To what extent can the various processes and factors influencing the aircraft ground handling process duration at AAS be integrated in the model?*

(iii) **How well does the model reflect the aircraft ground handling performance at AAS?**

(a) *What metrics and performance indicators are available and appropriate to assess the performance of the model?*

(b) *To what extent do the model assumptions influence the performance of the model?*

(c) *What differences in ground handling performance can be observed between the model and ground handling operations at AAS?*

(d) *How well is the model able to capture the underlying processes and factors of the system?*

(iv) **How can the model contribute to increasing the efficiency in the aviation network?**

(a) *What operational partners benefit from the models insights?*

(b) *How can the models predictions and insights be utilised by the operational partners during the aircraft ground handling process of a flight?*

(c) *How can insights provided by the model be leveraged to identify bottlenecks in the turnaround process?*

(d) *How can the model contribute to the predictability of flights in the aviation network?*

(v) **How well does the model provide insights into the underlying processes and/or factors that drive the untimeliness of ground handling at AAS?**

(a) *How well can the inner workings and/or decision process of the model be understood?*

(b) *How well can the model provide explanations on the output of the model?*

7

Conclusion

The aim of this report is to obtain a critical summary of existing knowledge and research in the field of predicting flight departure delays. Various literature research questions were defined to help build this knowledge and critical summary. The first question aims to identify relevant ground operation processes and/or factors affecting the on-time-performance of flights. The next question dealt with the analysis of the state-of-the-art in academic research in predicting flight delays. Subsequently, research gaps were identified. The last literature research question aimed to analyse and assess machine learning models and explanatory frameworks.

Various processes and factors were identified that affect the on-time-performance of flights and the ground operation time. In ground operations, four phases can be distinguished, namely: (1) taxi-in, (2) ground handling, (3) start-up/push-back and (4) taxi-out. The most notable process is the aircraft turnaround process, defined as the time between AIBT and AOBT. The aircraft turnaround process consists of sequential and parallel activities. Here, the shortest path, measured in units of time, of sequential and parallel activities yields the turnaround time and is defined as the critical path. The most common process members of the critical path are aircraft boarding and deboarding. Given the stochastic nature of these and other turnaround processes, the critical path is variable adding to the complexity of the underlying system.

Predicting whether a flight departs late has drawn the attention from the academic research community in the last decade. Partly due to the increase of available data and computational power, but also due to the rise of advanced machine learning algorithms. An analysis of the state-of-the-art in the field shows that most researchers in the field employ complex black-box machine learning models such as Artificial Neural Networks (ANN) and tree-ensembles to capture the complex underlying system, significantly limiting the interpretability. At the cost of reduced interpretability, these models are able to capture the underlying system best and yield the highest model performance. However, little attempt is made to obtain insights into the turnaround process, i.e. how this process relates to other processes or into the root causes of turnaround delays while predicting flight delays.

Various machine learning models are available in literature. Two types of models can be distinguished: (1) ante-hoc interpretable and (2) black-box machine learning models. The former are often more simplistic machine learning models, and offer inherent transparency, contributing to the interpretability of the model. The latter trade-off interpretability for model performance, limiting one's ability to discover its decision making process and extract insights from the model. The best interpretability is offered by Bayesian Networks, mainly due to the Directed Acyclic Graph (DAG) that describes the causalities between the input features. On the other hand, highest model accuracy is offered by (derivatives of) ANNs and eXtreme Gradient Boosting (XGBoost).

Black-box models require post-hoc explanatory frameworks to extract insights into the turnaround process from the model. Various post-hoc explanatory frameworks were elaborated on. The most sophisticated frameworks are offered by LIME and SHAP. However, based on the assessment, post-hoc explanatory frameworks cannot fully explain a model's inner workings and/or decision making process.

III

Supporting work

A

Variable description & discretization

This appendix provides an overview of the variables and the associated discretization. These are provided in section A.1 and A.2, respectively.

A.1. Variable description

This section describes the variables used in the model. A list is provided in Table A.1

Table A.1: Description of the variables used in the model

Variable	Standard/ Evidence	Description
Ground-handling delay	Standard	Variable representing the extent of the ground-handling delay.
Initial total turnaround time	Evidence	Node distinguishing turnarounds from one another based on the initial total turnaround time available.
Time to the initial TOBT	Evidence	Node providing temporal context to the ground-handling delay node, representing the time until the initial TOBT.
Aircraft type	Evidence	Variable representing the particular aircraft type involved in the turnaround.
Destination type	Evidence	Variable representing various types of destinations, as to account for the number of trolleys and luggage on board.
Outbound flight distance	Evidence	Variable representing the outbound flight distance [km] from Schiphol, calculated using the Haversine function.
Expected total inbound passengers	Evidence	Variable representing the total number of inbound passengers.
Expected total outbound passengers	Evidence	Variable representing the total number of outbound passengers.
Expected time passenger de-boarding remaining	Standard	Variable representing the (expected) duration remaining in the passenger de-boarding process.
Current total de-boarding time	Evidence	Variable representing the progress in the de-boarding process, as a measure of time.
De-boarding status	Evidence	Variable representing the current status of the de-boarding process.
Expected time fueling truck in position remaining	Standard	Variable representing the (expected) duration remaining that the fueling truck is in position during the turnaround.

Table A.1 continued from previous page

Variable	Standard/ Evidence	Description
Current total time fueling truck in position	Evidence	Variable representing progress in the fueling process, as a measure of time that the fueling truck was in position.
Fueling status	Evidence	Variable representing the current status of the fueling process.
Expected time catering truck in position remaining	Standard	Variable representing the (expected) duration remaining that the catering truck is in position during the turnaround. Here, in position indicates that the catering truck is correctly oriented and has completed its ascent.
Current total time catering truck in position	Evidence	Variable representing progress in the catering process, as a measure of time that the catering truck was in position (see previous item).
Catering status	Evidence	Variable representing the current status of the catering process.
Number of PAX on inbound flight legs	Evidence	Variable representing the number of PAX on the two inbound flight legs (e.g. AMS - LGW - Turnaround).
Number of PAX on outbound flight legs	Evidence	Variable representing the number of PAX on the two outbound flight legs (e.g. Turnaround - LGW - AMS).
Expected time BAX loading and unloading remaining	Standard	Variable representing the (expected) duration remaining baggage loading and unloading.
Current total time BAX loading and unloading	Evidence	Variable representing progress in the baggage loading and unloading process, as a measure of time that baggage loading and unloading has been observed during the turnaround.
Time since last BAX loading and unloading	Evidence	Variable representing the total time elapsed between the last time baggage loading and unloading was observed and the current time grid element.
Expected time belt loader in position remaining	Standard	Variable representing the (expected) duration remaining that the belt loader is in position during the turnaround.
Current total time belt loader in position	Evidence	Variable representing progress in the belt loader process, as a measure of time that the belt loader was in position.
Belt loader status	Evidence	Variable representing the current status of the belt loader process.
Expected time passenger deboarding remaining	Standard	Variable representing the (expected) duration remaining in the passenger boarding process.
Current total boarding time	Evidence	Variable representing the progress in the boarding process, as a measure of time.
Boarding status	Evidence	Variable representing the current status of the boarding process.

A.2. Model discretization

This appendix provides an overview of the discretization of the Bayesian network model. This overview is presented in Table A.2. For strategy, the following abbreviations are used:

- **DK**: Domain knowledge
- **EF**: (Regular) equal frequency discretization
- **TS**: Two-step equal frequency discretization

- **TSLE**: Time since the last detection of a certain event
- **EC**: Event-count

Table A.2: Overview of the cardinality, discretization strategy and exact states of the variables in the model.

Variable	Cardinality	Unit	Strategy	States
Ground-handling delay	3	[minutes]	DK	(-inf, 5), [5, 15), [15, inf)
Initial total turnaround time	4	[minutes]	DK + EF	[25, 40), [40, 50), [50, 73), [73, inf)
Time to the initial TOBT	6	[minutes]	DK	Initial TOBT-50, Initial TOBT-40, Initial TOBT-30, Initial TOBT-20, Initial TOBT-10, Initial TOBT-0
Aircraft type	8	[-]	DK	Aircraft A, Aircraft B, Aircraft C, Aircraft D, Aircraft E, Aircraft F, Aircraft G, Aircraft H
Destination type	3	[-]	DK	Business, Leisure, Mix
Outbound flight distance	5	[km]	EF	[0, 593), [593, 732), [732, 1101), [1101, 1461), [1461, inf)
Expected total inbound passengers	5	[-]	EF	[0, 124), [124, 148), [148, 166), [166, 175), [175, inf)
Expected total outbound passengers	5	[-]	EF	[0, 124), [124, 148), [148, 166), [166, 175), [175, inf)
Expected time passenger de-boarding remaining	6	[minutes]	TSEF	[0, 1), [1, 3), [3, 6), [6, 10), [10, 13), [13, inf)
Current total de-boarding time	6	[minutes]	TSEF	[0, 1), [1, 6), [6, 9), [9, 12), [12, 14), [14, inf)
De-boarding status	3	[-]	TSLE	Not started, In progress, No PAX last 5 min
Expected time fueling truck in position remaining	6	[seconds]	TSEF	[0, 1), [1, 341), [341, 682), [682, 1079), [1079, 1374), [1374, inf)
Current total time fueling truck in position	6	[seconds]	TSEF	[0, 1), [1, 347), [347, 691), [691, 1090), [1090, 1385), [1385, inf)
Fueling status	3	[-]	EC	Not started, In progress, Finished
Expected time catering truck in position remaining	5	[seconds]	TSEF	[0, 1), [1, 354), [354, 755), [755, 970), [970, inf)
Current total time catering truck in position	5	[seconds]	TSEF	[0, 1), [1, 357), [357, 756), [756, 971), [971, inf)
Catering status	3	[-]	EC	Not started, In progress, Finished
Number of PAX on inbound flight legs	5	[-]	EF	[0, 247), [247, 287), [287, 317), [317, 339), [339, inf)
Number of PAX on outbound flight legs	5	[-]	EF	[0, 251), [251, 292), [292, 320), [320, 342), [342, inf)
Expected time BAX loading and unloading remaining	6	[seconds]	TSEF	[0, 1), [1, 285), [285, 582), [582, 939), [939, 1305), [1305, inf)
Current total time BAX loading and unloading	6	[seconds]	TSEF	[0, 1), [1, 288), [288, 585), [585, 944), [944, 1310), [1310, inf)
Time since last BAX loading and unloading	7	[seconds]	DK + EF	[-1, 0), [0, 1), [1, 139), [139, 377), [377, 705), [705, 1199), [1199, inf]
Expected time belt loader in position remaining	7	[seconds]	TSEF	[0, 1), [1, 548), [548, 1094), [1094, 1664), [1664, 2336), [2336, 3058), [3058, inf)
Current total time belt loader in position	7	[seconds]	TSEF	[0, 1), [1, 549), [549, 1101), [1101, 1681), [1681, 2375), [2375, 3093), [3093, inf)
Belt loader status	3	[-]	EC	Not started, Connected, Disconnected

Table A.2 continued from previous page

Variable	Cardinality	Unit	Strategy	States
Expected time passenger boarding remaining	6	[minutes]	TSEF	[0, 1), [1, 5), [5, 10), [10, 15), [15, 20), [20, inf)
Current total boarding time	6	[minutes]	TSEF	[0, 1), [1, 5), [5, 10), [10, 15), [15, 20), [20, inf)
Boarding status	3	[-]	TSLE	Not started, In progress, No PAX last 5 min

B

Model Description & Results

This section presents the full model structure and additional results. First, the full model structure is presented in section B.1. Next, the precision and recall metrics are visualized in section B.2. section B.3 presents the confusion matrices for all prediction moments between the initial TOBT-50 to initial TOBT-0 [minutes]. Last, the results from the sensitivity analysis are presented in section B.4.

B.1. Model structure

This section presents the Direct Acyclic Graph (DAG) structure used in this study. For readability reasons, the DAG structure has been split over multiple figures. A legend is provided in Figure B.1 to distinguish between regular, network structures and evidence (input) nodes.

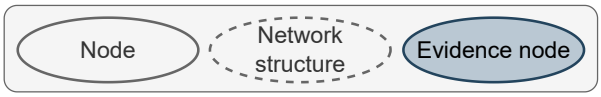

Figure B.1: Legend to distinguish between regular nodes, network structures and evidence (input) nodes.

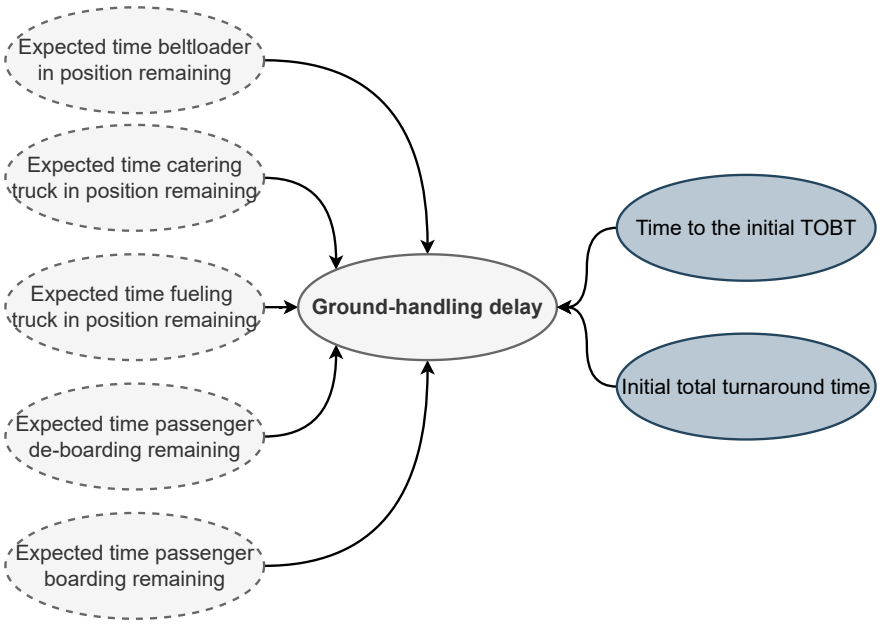

Figure B.2: Central network structure of the Bayesian network model.

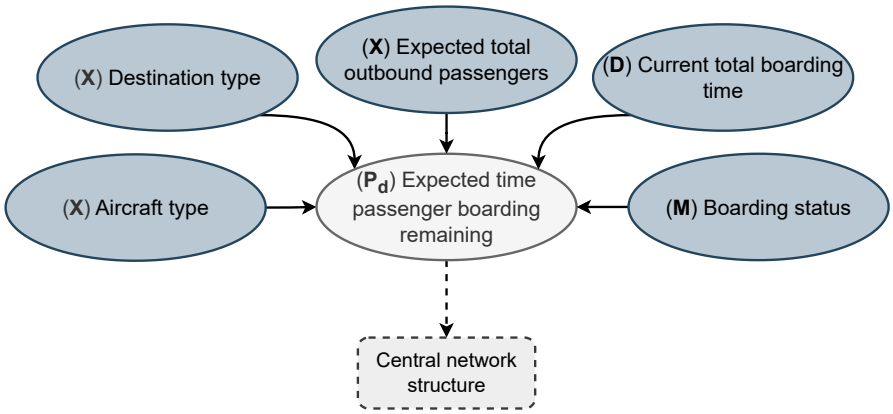

Figure B.3: Bayesian sub-network of the passenger boarding process. Letters indicate the type of node.

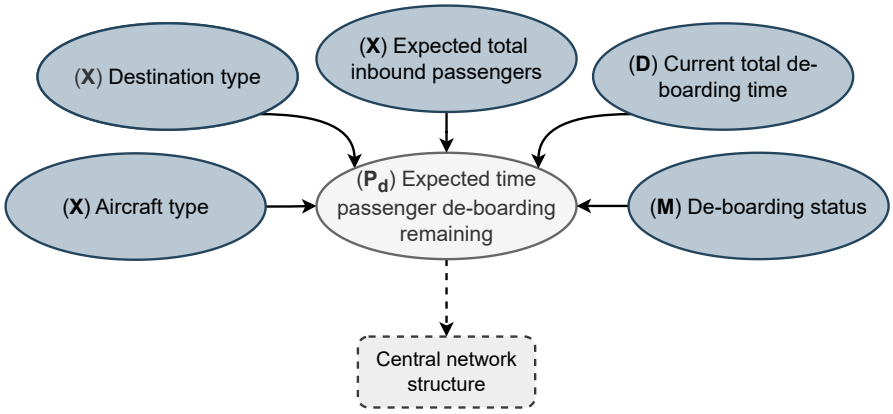

Figure B.4: Bayesian sub-network of the passenger de-boarding process. Letters indicate the type of node.

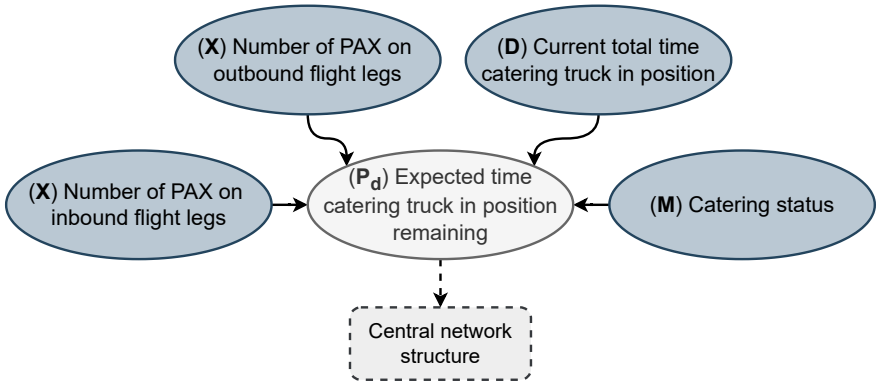

Figure B.5: Bayesian sub-network of the aircraft catering process. Letters indicate the type of node.

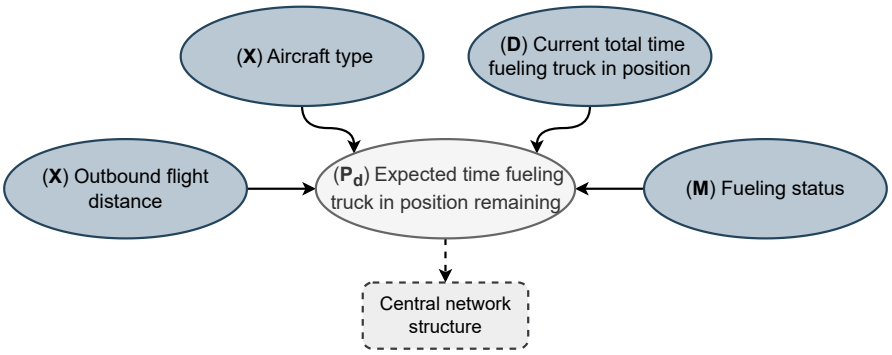

Figure B.6: Bayesian sub-network of the aircraft re-fueling process. Letters indicate the type of node.

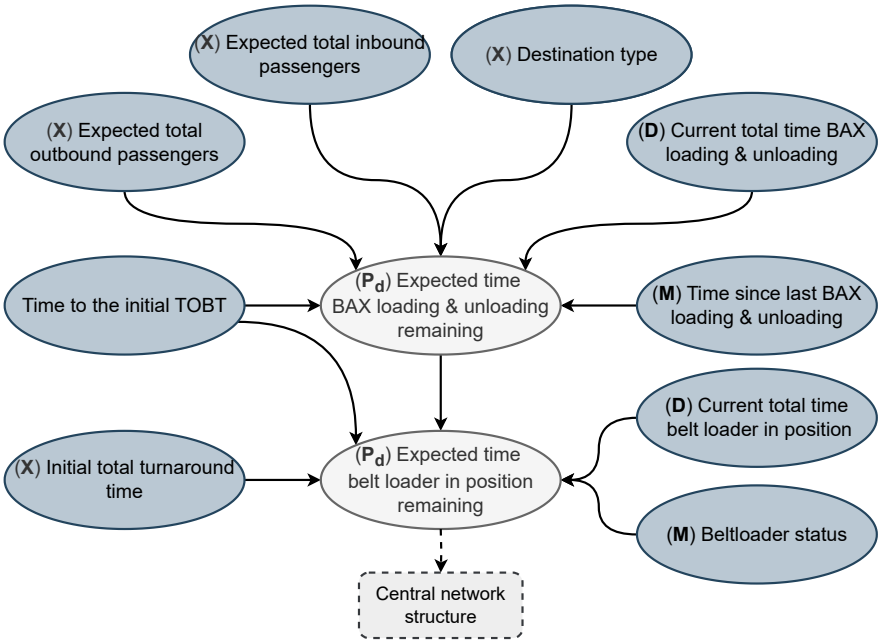

Figure B.7: Bayesian sub-network of the baggage loading & unloading process. Letters indicate the type of node.

B.2. Precision and Recall

This section visualizes the precision and recall metrics for the model and handler.

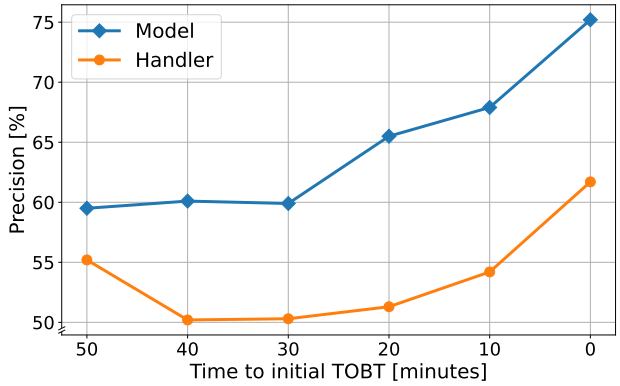

Figure B.8: Precision metric for model and handler, for ground-handling delay class (-inf, 5).

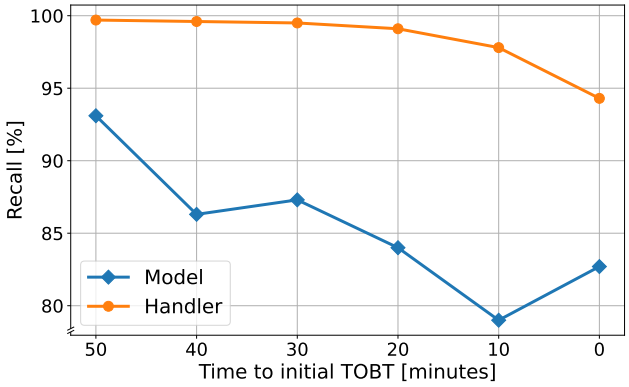

Figure B.9: Recall metric for model and handler, for ground-handling delay class (-inf, 5).

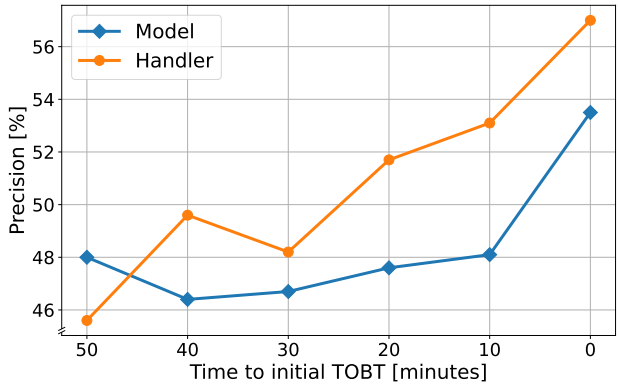

Figure B.10: Precision metric for model and handler, for ground-handling delay class [5, 15).

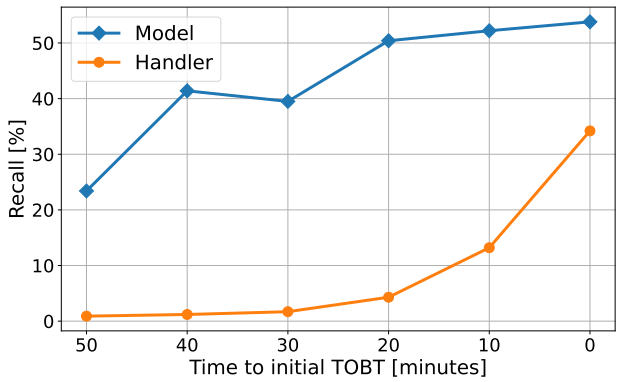

Figure B.11: Recall metric for model and handler, for ground-handling delay class [5, 15).

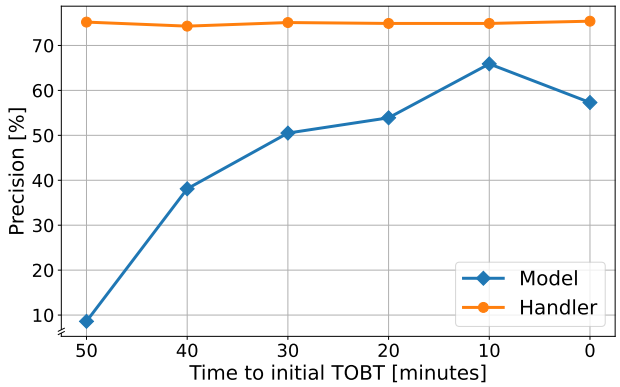

Figure B.12: Precision metric for model and handler, for ground-handling delay class [15, inf).

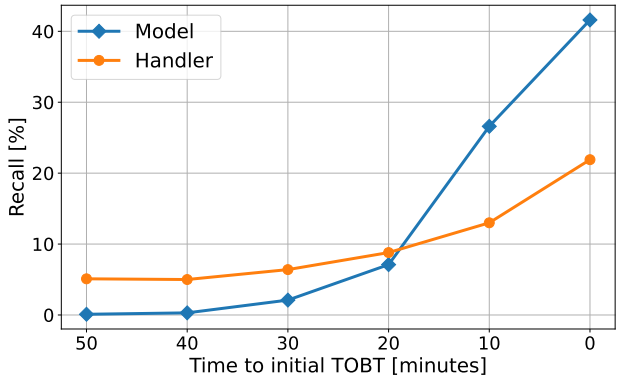

Figure B.13: Recall metric for model and handler, for ground-handling delay class [15, inf).

B.3. Confusion matrices

In this section, all confusion matrices between TOBT-50 and TOBT-0 [minutes] are presented.

Table B.1: Confusion matrix for the initial TOBT-50 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	93.1	6.9	0.1
	[5, 15)	76.4	23.4	0.1
	[15, inf)	74.3	25.6	0.1

Table B.2: Confusion matrix for the initial TOBT-40 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	86.3	13.7	0.0
	[5, 15)	58.5	41.4	0.1
	[15, inf)	51.4	48.3	0.3

Table B.3: Confusion matrix for the initial TOBT-30 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	87.3	12.5	0.2
	[5, 15)	59.6	39.5	0.8
	[15, inf)	51.3	46.6	2.1

Table B.4: Confusion matrix for the initial TOBT-20 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	84.0	15.7	0.3
	[5, 15)	46.6	50.4	3.0
	[15, inf)	35.8	57.1	7.1

Table B.5: Confusion matrix for the initial TOBT-10 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	79.0	20.4	0.6
	[5, 15)	41.0	52.2	6.8
	[15, inf)	27.5	45.9	26.6

Table B.6: Confusion matrix for the initial TOBT-0 [minutes].

		Predicted states [%]		
		(-inf, 5)	[5, 15)	[15, inf)
Actual states	(-inf, 5)	82.7	16.1	1.2
	[5, 15)	30.8	53.8	15.4
	[15, inf)	18.3	40.1	41.6

B.4. Sensitivity Analysis

This section presents the sensitivity analysis results.

Table B.7: Sensitivity analysis for rounded bins and applying K-means in the two-step discretization.

Ground Handling Delay	Metric w.r.t. baseline	Model	Time to initial TOBT [minutes]					
			50	40	30	20	10	0
N/A	$\Delta_{Accuracy}$ [%]	K-Means	0.12	0.0	0.11	-0.45	0.05	-0.35
		Rounded	0.14	-0.01	0.12	-0.5	-0.1	-0.29
(-inf, 5)	$\Delta_{Precision}$ [%]	K-Means	-0.2	-0.2	0.0	0.0	0.6	0.0
		Rounded	-0.1	-0.2	0.0	-0.1	0.5	0.0
	Δ_{Recall} [%]	K-Means	0.8	0.3	-0.3	-0.6	-0.7	-0.3
		Rounded	0.8	0.4	-0.3	-0.7	-0.6	-0.4
[5, 15)	$\Delta_{Precision}$ [%]	K-Means	-25.7	-5.6	-6.9	2.5	4.1	-1.2
		Rounded	0.8	0.2	0.4	-0.9	-0.4	-0.2
	Δ_{Recall} [%]	K-Means	-1.1	-0.6	0.3	-0.3	0.0	-1.5
		Rounded	-1.0	-0.9	0.5	-0.3	-0.3	-1.0
[15, inf)	$\Delta_{Precision}$ [%]	K-Means	21.4	15.1	2.7	-1.1	-3.0	-1.7
		Rounded	21.4	19.6	2.7	-0.6	-2.8	-1.2
	Δ_{Recall} [%]	K-Means	0.0	0.1	0.7	-0.2	2.2	1.6
		Rounded	0.0	0.3	0.4	-0.2	1.7	1.1

To assess the model's ability to deal with a finer resolution in discrete states, four experiments were conducted expanding the granularity in various areas within the two-step discretization of nodes $\mathbf{P_d}$ and **D**:

- **T1**: The addition of one bin to the number of bins in the total distribution.
- **T2**: The addition of two bins to the number of bins in the total distribution.
- **P1**: The addition of one bin to the number of bins representing the intermediate states.
- **T1P1**: The addition of one bin to the number of bins representing the total distribution and one bin representing the intermediate states.

Table B.8: Sensitivity analysis results of expanding the number of bins for nodes $\mathbf{P_d}$ and **D**.

Ground Handling Delay	Metric w.r.t. baseline	Model	Time to initial TOBT [minutes]					
			50	40	30	20	10	0
N/A	$\Delta_{Accuracy}$ [%]	T1	-0.05	0.02	-0.05	0.41	0.3	-0.19
		T2	-0.3	0.11	-0.09	0.35	0.43	-0.23
		P1	-0.23	0.05	-0.01	0.12	0.78	-0.24
		T1P1	-0.19	0.04	0.01	0.31	0.51	-0.22
(-inf, 5)	$\Delta_{Precision}$ [%]	T1	-0.2	0.0	-0.1	0.3	0.3	0.0
		T2	-0.7	-0.4	0.0	0.3	0.4	0.0
		P1	-0.5	-0.2	0.1	0.1	0.5	0.0
		T1P1	-0.5	-0.3	0.0	0.3	0.5	0.0
	Δ_{Recall} [%]	T1	0.2	-0.1	0.1	0.1	0.2	-0.1
		T2	1.0	1.4	-0.1	0.6	0.7	0.3
		P1	0.5	0.8	-0.2	0.5	0.8	-0.3
		T1P1	0.8	1.1	-0.1	0.7	0.7	-0.3
[5, 15)	$\Delta_{Precision}$ [%]	T1	0.3	0.0	0.1	0.6	0.4	-0.2
		T2	0.5	0.6	0.0	0.5	0.6	-0.2
		P1	0.0	0.3	0.1	0.1	1.0	-0.3
		T1P1	0.4	0.3	0.1	0.4	0.6	-0.1
	Δ_{Recall} [%]	T1	-0.6	0.0	0.3	0.2	0.3	-0.1
		T2	-2.8	-1.7	0.5	0.1	0.1	0.1
		P1	-1.7	-1.1	0.6	-0.1	0.5	0.3
		T1P1	-2.1	-1.5	0.6	0.2	0.1	0.3
[15, inf)	$\Delta_{Precision}$ [%]	T1	-1.9	1.9	-10.6	1.7	-0.8	-0.7
		T2	4.3	-6.7	-6.3	-0.3	-0.9	-0.5
		P1	10.9	-2.0	-8.6	-1.5	0.4	-0.4
		T1P1	24.3	-3.1	-3.0	-1.2	-0.9	-0.8
	Δ_{Recall} [%]	T1	-0.1	-0.1	-1.0	1.6	0.6	-0.7
		T2	0.0	-0.1	-1.2	0.2	0.3	-1.3
		P1	0.0	0.0	-0.7	-0.6	1.5	-1.3
		T1P1	0.0	-0.1	-0.8	-0.4	0.9	-1.0

C

Knowledge Elicitation

This appendix presents the knowledge elicitation interview questions, based on the knowledge elicitation questions from Shadbolt and Smart [179]. In the structured interview, the set of questions below was used for every turnaround process within the scope of this research.

- **Probe F.1:** Can you [in one or two minutes] talk me through [process under consideration]?
  Function F.1: Provides an overview of the [process under consideration] concept, tasks and structure of the [process under consideration].

- **Probe F.2:** When [in the turnaround process] would you execute [process under consideration]?
  Function F.2: Reveals the temporal "location" of the [process under consideration] and possible dependencies between turnaround processes.

- **Probe F.3:** Is [temporal location of process under consideration] always the case? [Why is it not always the case?]
  Function F.3: Reveals the generality of the temporal location of the process.

- **Probe F.4:** What do you reckon the "nominal" duration of [process under consideration] is? [What factors affect the nominal duration?] [What if it were not the case that [nominal duration]?] [What event defines the end of the process?]
  Function F.4: Asks the expert to assess the "nominal" duration of [process under consideration]. Teases the expert to define the "nominal" duration and its generalizability, subsequently revealing underlying precursors. After all, there is no "nominal" between turnarounds. "Nominal" is defined by a set of precursors and rules, and the interview aims to extract them from the expert.

- **Probe F.5:** Is/are [precursor(s) from the previous question] always the case?
  Function F.5: Assesses the generality of the [in previous question(s) obtained] precursor(s).

- **Probe F.6:** What would be the minimum process duration time for [process under consideration]? [When is this the case?]
  Function F.6: Extracting more precursors, validating business rules.

References

[1] Federal Aviation Administration (FAA). “Ground Delay Programs”. In: *Traffic Management National, Center, and Terminal*. Nov. 2022.

[2] P. Albertos, A. Sala, and M. Olivares. “Fuzzy Logic Controllers. Methodology, Advantages and Drawbacks”. In: *X Congreso Español sobre Tecnologías y Lógica Fuzzy*. Sept. 2000. DOI: 10.13140/RG.2.1.2512.6164.

[3] R. Andrews, J. Diederich, and A.B. Tickle. “Survey and critique of techniques for extracting rules from trained artificial neural networks”. In: *Knowledge-Based Systems* 8.6 (1995). Knowledge-based neural networks, pp. 373–389. ISSN: 0950-7051. DOI: https://doi.org/10.1016/0950-7051(96)81920-4. URL: https://www.sciencedirect.com/science/article/pii/0950705196819204.

[4] A.S. Antonio, A.A. Juan, L. Calvet, P. Fonseca i Casas, and D. Guimarans. “Using simulation to estimate critical paths and survival functions in aircraft turnaround processes”. In: *2017 Winter Simulation Conference (WSC)*. 2017, pp. 3394–3403. DOI: 10.1109/WSC.2017.8248055.

[5] J.S. Armstrong and F. Collopy. “Error measures for generalizing about forecasting methods: Empirical comparisons”. In: *International Journal of Forecasting* 8.1 (1992), pp. 69–80. ISSN: 0169-2070. DOI: https://doi.org/10.1016/0169-2070(92)90008-W. URL: https://www.sciencedirect.com/science/article/pii/016920709290008W.

[6] E. Asadi, J. Evler, H. Preis, and H. Fricke. “Coping with Uncertainties in Predicting the Aircraft Turnaround Time at Airports”. In: *Operations Research Proceedings 2019*. Ed. by J.S. Neufeld, U. Buscher, R. Lasch, D. Möst, and J. Schönberger. Cham: Springer International Publishing, Sept. 2020, pp. 773–780. ISBN: 978-3-030-48439-2. DOI: 10.1007/978-3-030-48439-2_94.

[7] E. Asadi and H. Fricke. “Aircraft Total Turnaround Time Estimation using Fuzzy Critical Path Method”. In: *Journal of Project Management* 7 (Apr. 2022), pp. 241–254. DOI: 10.5267/j.jpm.2022.4.001.

[8] A. Barredo Arrieta et al. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information Fusion* 58 (2020), pp. 82–115. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2019.12.012. URL: https://www.sciencedirect.com/science/article/pii/S1566253519308103.

[9] V. Belle and I. Papantonis. “Principles and Practice of Explainable Machine Learning”. In: *Frontiers in Big Data* 4 (2021). ISSN: 2624-909X. DOI: 10.3389/fdata.2021.688969. URL: https://www.frontiersin.org/articles/10.3389/fdata.2021.688969.

[10] H. Belyadi and A. Haghighat. “Chapter 5 - Supervised learning”. In: *Machine Learning Guide for Oil and Gas Using Python*. Gulf Professional Publishing, 2021, pp. 169–295. ISBN: 978-0-12-821929-4. DOI: https://doi.org/10.1016/B978-0-12-821929-4.00004-4. URL: https://www.sciencedirect.com/science/article/pii/B9780128219294000044.

[11] N. Ben Yahia, N. Bellamine Ben Saoud, and H. Ben Ghezala. “Integrating fuzzy case-based reasoning and particle swarm optimization to support decision making”. In: *International Journal of Computer Science Issues* 9 (May 2012).

[12] I. Ben-Gal. “Bayesian Networks”. In: *Encyclopedia of Statistics in Quality and Reliability*. John Wiley & Sons, Ltd, 2008. ISBN: 9780470061572. DOI: https://doi-org.tudelft.idm.oclc.org/10.1002/9780470061572.eqr089. eprint: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/pdf/10.1002/9780470061572.eqr089. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1002/9780470061572.eqr089.

[13] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. “Algorithms for Hyper-Parameter Optimization”. In: NIPS’11. Granada, Spain: Curran Associates Inc., 2011, pp. 2546–2554. ISBN: 9781618395993.

[14] J. Bergstra and Y. Bengio. “Random Search for Hyper-Parameter Optimization”. In: *Journal of Machine Learning Research* 13.10 (Feb. 2012), pp. 281–305. ISSN: 1532-4435.

[15] D. Berrar. "Cross-Validation". In: *Encyclopedia of Bioinformatics and Computational Biology*. Ed. by S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach. Oxford: Academic Press, 2019, pp. 542–545. ISBN: 978-0-12-811432-2. DOI: https://doi.org/10.1016/B978-0-12-809633-8.20349-X. URL: https://www.sciencedirect.com/science/article/pii/B978012809633820349X.

[16] C. Bielza and P. Larrañaga. "Discrete Bayesian Network Classifiers: A Survey". In: *ACM Computing Surveys* 47.1 (July 2014). ISSN: 0360-0300. DOI: 10.1145/2576868. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/2576868.

[17] C.M. Bishop. "Neural networks and their applications". In: *Review of Scientific Instruments* 65.6 (June 1994), pp. 1803–1832. ISSN: 0034-6748. DOI: 10.1063/1.1144830. eprint: https://pubs.aip.org/aip/rsi/article-pdf/65/6/1803/8387807/1803_1_online.pdf. URL: https://doi.org/10.1063/1.1144830.

[18] C.M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[19] B. Boehmke and B. Greenwell. *Hands-On Machine Learning with R*. first. Vol. 1. Chapman & Hall, Nov. 2019. ISBN: 9781138495685.

[20] D.N. Boote and P. Beile. "Scholars Before Researchers: On the Centrality of the Dissertation Literature Review in Research Preparation". In: *Educational Researcher* 34.6 (2005), pp. 3–15. DOI: 10.3102/0013189X034006003. eprint: https://doi.org/10.3102/0013189X034006003. URL: https://doi.org/10.3102/0013189X034006003.

[21] E. Borgonovo and E. Plischke. "Sensitivity analysis: A review of recent advances". In: *European Journal of Operational Research* 248.3 (2016), pp. 869–887. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2015.06.032. URL: https://www.sciencedirect.com/science/article/pii/S0377221715005469.

[22] L. Breiman. "Bagging predictors". In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. ISSN: 1573-0565. DOI: 10.1007/BF00058655. URL: https://doi.org/10.1007/BF00058655.

[23] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324.

[24] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification And Regression Trees*. Chapman & Hall, Oct. 1984. ISBN: 9780412048418.

[25] D. Brinkman. "Knowledge-based Control Systems for Re-entry Vehicles". MA thesis. Delft University of Technology, Jan. 2017. URL: http://resolver.tudelft.nl/uuid:b77b4495-d42b-46be-9a9c-306f49bfbe32.

[26] X.M. van der Broek. "Solving KLM's hand luggage problem". MA thesis. Delft University of Technology, Feb. 2015. URL: http://resolver.tudelft.nl/uuid:f4730c6e-7c83-4d6b-a990-4a72b3493318.

[27] S. Burkhardt, J. Brugger, N. Wagner, Z. Ahmadi, K. Kersting, and S. Kramer. "Rule Extraction From Binary Neural Networks With Convolutional Rules for Model Validation". In: *Frontiers in Artificial Intelligence* 4 (2021). ISSN: 2624-8212. DOI: 10.3389/frai.2021.642263. URL: https://www.frontiersin.org/articles/10.3389/frai.2021.642263.

[28] K. Bykov et al. *Explaining Bayesian Neural Networks*. 2021. arXiv: 2108.10346 [cs.LG].

[29] C.A. Cameron and F.A.G. Windmeijer. "An R-squared measure of goodness of fit for some common nonlinear regression models". In: *Journal of Econometrics* 77.2 (1997), pp. 329–342. ISSN: 0304-4076. DOI: https://doi.org/10.1016/S0304-4076(96)01818-0. URL: https://www.sciencedirect.com/science/article/pii/S0304407696018180.

[30] T. Chai and R.R. Draxler. "Root mean square error (RMSE) or mean absolute error (MAE)? Arguments against avoiding RMSE in the literature". In: *Geoscientific Model Development* 7.3 (2014), pp. 1247–1250. DOI: 10.5194/gmd-7-1247-2014. URL: https://gmd.copernicus.org/articles/7/1247/2014/.

[31] E. Charniak. "Bayesian Networks without Tears." In: *AI Magazine* 12.4 (Dec. 1991), p. 50. DOI: 10.1609/aimag.v12i4.918. URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/918.

[32] T. Charnock, L. Perreault-Levasseur, and F. Lanusse. *Bayesian Neural Networks*. 2020. arXiv: 2006.01490 [stat.ML].

[33] T. Chen and C. Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Aug. 2016. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145%5C%2F2939672.2939785.

[34] D. Chicco, M. J. Warrens, and G. Jurman. "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation". In: *PeerJ Computer Science* 7 (July 2021). URL: https://peerj.com/articles/cs-623/.

[35] D.M. Chickering. "Learning Bayesian Networks is NP-Complete". In: *Learning from Data: Artificial Intelligence and Statistics V*. Learning from Data: Artificial Intelligence and Statistics V. Springer-Verlag, Jan. 1996, pp. 121–130. URL: https://www.microsoft.com/en-us/research/publication/learning-bayesian-networks-is-np-complete/.

[36] CodeCrucks. *What is fuzzy membership function A complete guide*. https://codecrucks.com/what-is-fuzzy-membership-function-complete-guide/#:~:text=Singleton%20membership%20function%3A,the%20impulse%20function%20as%20shown. accessed on 17/04/2023. Aug. 2021.

[37] A.C. Constantinou. *Evaluating structure learning algorithms with a balanced scoring function*. 2020. arXiv: 1905.12666 [cs.LG].

[38] J. Dai and J. Ren. "Unsupervised Evolutionary Algorithm for Dynamic Bayesian Network Structure Learning". In: *Advanced Methodologies for Bayesian Networks*. Ed. by J. Suzuki and M. Ueno. Springer International Publishing, 2015, pp. 136–151. ISBN: 978-3-319-28379-1.

[39] R. Dalmau, S. Belkoura, H. Naessens, F. Ballerini, and S. Wagnick. "Improving the predictability of take-off times with Machine Learning: a case study for the Maastricht upper area control centre area of responsibility". In: *9th SESAR Innovation Days*. Dec. 2019.

[40] A. de Myttenaere, B. Golden, B. Le Grand, and F. Rossi. "Mean Absolute Percentage Error for regression models". In: *Neurocomputing* 192 (2016). Advances in artificial neural networks, machine learning and computational intelligence, pp. 38–48. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2015.12.114. URL: https://www.sciencedirect.com/science/article/pii/S0925231216003325.

[41] DeepFindr. *Explainable AI explained! | #4 SHAP*. https://www.youtube.com/watch?v=9haIOplEIGM accessed on 13/04/2023. Mar. 2021.

[42] F. Doshi-Velez and B. Kim. *Towards A Rigorous Science of Interpretable Machine Learning*. 2017. arXiv: 1702.08608 [stat.ML].

[43] M. Drury. *A Comparison of Basis Expansions in Regression*. http://madrury.github.io/jekyll/update/statistics/2017/08/04/basis-expansions.html accessed on 11/05/2023. Aug. 2017.

[44] R. Dwivedi et al. "Explainable AI (XAI): Core Ideas, Techniques, and Solutions". In: *ACM Computer Survey* 55.9 (Jan. 2023). ISSN: 0360-0300. DOI: 10.1145/3561048. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3561048.

[45] T.W. Edgar and D.O. Manz. "Chapter 4 - Exploratory Study". In: *Research Methods for Cyber Security*. Syngress, 2017, pp. 95–130. ISBN: 978-0-12-805349-2. DOI: https://doi.org/10.1016/B978-0-12-805349-2.00004-2. URL: https://www.sciencedirect.com/science/article/pii/B9780128053492000042.

[46] Google Machine Learning Education. *Machine Learning Crash Course*. https://developers.google.com/machine-learning/crash-course/ml-intro accessed on 16/05/2023. July 2022.

[47] T. Elangovan, M. Raheem, and A. Arshad. "Predictive Analytics for Airline Departure Delays". In: *Journal of Critical Reviews* 7 (Aug. 2020), pp. 4034–4039. DOI: 10.31838/jcr.07.11.549.

[48] F. Eriksson and Rik S. *Optimisation Outbound Cluster - A study on the usability of digital communication means in the Schiphol outbound process*. Tech. rep. Moving Dot, Jan. 2017.

[49] Eurocontrol. *Airport ATFM Delay*. https://www.eurocontrol.int/prudata/dashboard/metadata/airport-atfm-delay/ accessed on 29/03/2023. Aug. 2021.

[50] Eurocontrol. *CODA Digest: All-Causes Delays to Air Transport in Europe - Annual 2022*. Tech. rep. Eurocontrol, Mar. 2023. URL: https://www.eurocontrol.int/sites/default/files/2022-10/eurocontrol-coda-digest-q2-2022.pdf.

[51] Eurocontrol. *Airport Collaborative Decision-making (A-CDM)*. URL: https://www.eurocontrol.int/concept/airport-collaborative-decision-making.

[52] S. Falkner, A. Klein, and F. Hutter. *BOHB: Robust and Efficient Hyperparameter Optimization at Scale*. 2018. arXiv: 1807.01774 [cs.LG].

[53] A. Feelders and L.C. van der Gaag. "Learning Bayesian network parameters under order constraints". In: *International Journal of Approximate Reasoning* 42.1 (2006). PGM04, pp. 37–53. ISSN: 0888-613X. DOI: https://doi.org/10.1016/j.ijar.2005.10.003. URL: https://www.sciencedirect.com/science/article/pii/S0888613X05000630.

[54] F. Ferretti, A. Saltelli, and S. Tarantola. "Trends in sensitivity analysis practice in the last decade". In: *Science of The Total Environment* 568 (2016), pp. 666–670. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2016.02.133. URL: https://www.sciencedirect.com/science/article/pii/S0048969716303448.

[55] M. Feurer and F. Hutter. "Hyperparameter Optimization". In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by F. Hutter, L. Kotthoff, and J. Vanschoren. Cham: Springer International Publishing, 2019, pp. 3–33. ISBN: 978-3-030-05318-5. DOI: 10.1007/978-3-030-05318-5_1. URL: https://doi.org/10.1007/978-3-030-05318-5_1.

[56] E. Frank, L. Trigg, G. Holmes, I. Witten, and W. Aha. "Naive Bayes for Regression". In: *Machine Learning* (July 2001).

[57] Y. Freund and R.E. Schapire. "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting". In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139. ISSN: 0022-0000. DOI: https://doi.org/10.1006/jcss.1997.1504. URL: https://www.sciencedirect.com/science/article/pii/S002200009791504X.

[58] C.H. Frey and S.R. Patil. "Identification and Review of Sensitivity Analysis Methods". In: *Risk Analysis* 22.3 (2002), pp. 553–578. DOI: https://doi.org/10.1111/0272-4332.00039. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/0272-4332.00039. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/0272-4332.00039.

[59] H. Fricke and M. Schultz. "Delay Impacts onto Turnaround Performance - Optimal Time Buffering for Minimizing Delay Propagation". In: *USA/Europe Air Traffic Management Research and Development Seminar*. June 2009. URL: https://api.semanticscholar.org/CorpusID:28942704.

[60] J.H. Friedman. "Multivariate Adaptive Regression Splines". In: *The Annals of Statistics* 19.1 (1991), pp. 1–67. DOI: 10.1214/aos/1176347963. URL: https://doi.org/10.1214/aos/1176347963.

[61] J.H. Friedman. "Greedy function approximation: A gradient boosting machine." In: *The Annals of Statistics* 29.5 (2001), pp. 1189–1232. DOI: 10.1214/aos/1013203451. URL: https://doi.org/10.1214/aos/1013203451.

[62] J.H. Friedman. "Stochastic gradient boosting". In: *Computational Statistics & Data Analysis* 38.4 (2002). Nonlinear Methods and Data Mining, pp. 367–378. ISSN: 0167-9473. DOI: https://doi.org/10.1016/S0167-9473(01)00065-2. URL: https://www.sciencedirect.com/science/article/pii/S0167947301000652.

[63] J.H. Friedman and B.W. Silverman. "Flexible Parsimonious Smoothing and Additive Modeling". In: *Technometrics* 31.1 (1989), pp. 3–21. ISSN: 00401706. URL: http://www.jstor.org/stable/1270359 (visited on 05/11/2023).

[64] N. Friedman, D. Geiger, and M. Goldszmidt. "Bayesian Network Classifiers". In: *Machine Learning* 29.2 (Nov. 1997), pp. 131–163. ISSN: 1573-0565. DOI: 10.1023/A:1007465528199. URL: https://doi.org/10.1023/A:1007465528199.

[65] Y. Gao, Z. Huyan, and F. Ju. "A Prediction Method Based on Neural Network for Flight Turnaround Time at Airport". In: *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*. Vol. 2. 2015, pp. 219–222. DOI: 10.1109/ISCID.2015.44.

[66] M.W. Gardner and S.R. Dorling. “Artificial neural networks (the multilayer perceptron)a review of applications in the atmospheric sciences”. In: *Atmospheric Environment* 32.14 (1998), pp. 2627–2636. ISSN: 1352-2310. DOI: https://doi.org/10.1016/S1352-2310(97)00447-0. URL: https://www.sciencedirect.com/science/article/pii/S1352231097004470.

[67] M.A. Gelbart, J. Snoek, and R.P. Adams. *Bayesian Optimization with Unknown Constraints*. 2014. arXiv: 1403.5607 [stat.ML].

[68] A. Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O’Reilly Media, Incorporated, 2019. ISBN: 9781492032649. URL: https://books.google.nl/books?id=OCS1twEACAAJ.

[69] L.H. Gilpin, D. Bau, B.Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. *Explaining Explanations: An Overview of Interpretability of Machine Learning*. 2019. arXiv: 1806.00069 [cs.AI].

[70] E. Goan and C. Fookes. “Bayesian Neural Networks: An Introduction and Survey”. In: *Case Studies in Applied Bayesian Data Science*. Springer International Publishing, 2020, pp. 45–87. DOI: 10.1007/978-3-030-42553-1_3. URL: https://doi.org/10.1007%5C%2F978-3-030-42553-1_3.

[71] C. Gonzales, A. Journe, and A. Mabrouk. “Constraint-Based Bayesian Network Structure Learning using Uncertain Experts Knowledge”. In: *The International FLAIRS Conference Proceedings* 34 (Apr. 2021). DOI: 10.32473/flairs.v34i1.128453. URL: https://journals.flvc.org/FLAIRS/article/view/128453.

[72] T.L. Griffiths and A. Yuille. “A primer on probabilistic inference”. In: *The Probabilistic Mind: Prospects for Bayesian cognitive science*. Oxford University Press, Mar. 2008. ISBN: 9780199216093. DOI: 10.1093/acprof:oso/9780199216093.003.0002. eprint: https://academic.oup.com/book/0/chapter/153457642/chapter-ag-pdf/44965324/book_8069_section_153457642.ag.pdf. URL: https://doi.org/10.1093/acprof:oso/9780199216093.003.0002.

[73] Dviation Group. *The Difference Between Line, Base and Component Maintenance*. https://blog.dviation.com/2018/11/14/the-difference-between-line-base-and-component-maintenance/ accessed on 29/03/2023. Nov. 2018.

[74] IATA Airline A-CDM Coordination Group. *Airport - Collaborative Decision Making (A-CDM): IATA Recommendations*. IATA. 2018. URL: https://www.iata.org/contentassets/5c1a116a6120415f87f3dadfa38859d2/iata-acdm-recommendations-v1.pdf.

[75] Royal Schiphol Group. *Collaborative Decision Making*. https://www.schiphol.nl/en/operations/page/cdm/ accessed on 22/06/2023.

[76] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. “A Survey of Methods for Explaining Black Box Models”. In: *ACM Computer Survey* 51.5 (Aug. 2018). ISSN: 0360-0300. DOI: 10.1145/3236009. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3236009.

[77] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, D. Pedreschi, and F. Giannotti. *A Survey Of Methods For Explaining Black Box Models*. 2018. arXiv: 1802.01933 [cs.CY].

[78] A. Habib. *Cathay Pacific warns pilots about taxiing at slower speeds*. https://simpleflying.com/cathay-pacific-warns-pilots-taxiing-slower-speeds/ accessed on 21/06/2023. May 2023.

[79] E. Halmesaari. “Interpretable Machine Learning for Prediction of Aircraft Turnaround Times”. MA thesis. Aalto University - School of Science, Dec. 2020. URL: http://urn.fi/URN:NBN:fi:aalto-2020122056351.

[80] D.M. Hamby. “A comparison of sensitivity analysis techniques”. In: *Health Physics* 68.2 (Feb. 1995), pp. 195–204.

[81] O.F.J. van Hassel. “Predicting the turnaround time of an aircraft: a process structure aware approach”. MA thesis. Eindhoven University of Technology, Dec. 2019. URL: https://research.tue.nl/en/studentTheses/predicting-the-turnaround-time-of-an-aircraft.

[82] T. Hastie, R. Tibshirani, and J. Friedman. In: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer New York, 2009. ISBN: 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7_1. URL: https://doi.org/10.1007/978-0-387-84858-7_1.

[83] G. Hayward and V. Davidson. “Fuzzy logic applications”. In: *Analyst* 128.11 (Nov. 2003), pp. 1304–1306.

[84] H. He and E.A. Garcia. “Learning from Imbalanced Data”. In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (2009), pp. 1263–1284. DOI: 10.1109/TKDE.2008.239.

[85] Helene. *An Introduction to Gaussian Bayesian Networks*. https://helenedk.medium.com/an-introduction-to-gaussian-bayesian-networks-4eeed3d8e6e0 accessed on 21/04/2023. Feb. 2022.

[86] J.J. Hox. *Multilevel analysis: Techniques and applications*. 2nd ed. Quantitative Methodology. Routledge, Jan. 2010.

[87] E. Hüllermeier. “Does machine learning need fuzzy logic?” In: *Fuzzy Sets and Systems* 281 (2015). Special Issue Celebrating the 50th Anniversary of Fuzzy Sets, pp. 292–299. ISSN: 0165-0114. DOI: https://doi.org/10.1016/j.fss.2015.09.001. URL: https://www.sciencedirect.com/science/article/pii/S0165011415004133.

[88] E. Hüllermeier. *From knowledge-based to data-driven modeling of fuzzy rule-based systems: A critical reflection*. Dec. 2017. arXiv: 1712.00646 [cs.AI].

[89] L. Hulstaert. *Understanding model predictions with LIME*. https://towardsdatascience.com/understanding-model-predictions-with-lime-a582fdff3a3b accessed on 11/04/2023. July 2018.

[90] J. Huo, K.L. Keung, C.K.M. Lee, K.K.H. Ng, and K.C. Li. “The Prediction of Flight Delay: Big Data-driven Machine Learning Approach”. In: *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. 2020, pp. 190–194. DOI: 10.1109/IEEM45057.2020.9309919.

[91] J. Huysmans, B. Baesens, and J. Vanthienen. “Using Rule Extraction to Improve the Comprehensibility of Predictive Models”. In: *SSRN Electronic Journal* (Jan. 2006). DOI: 10.2139/ssrn.961358.

[92] B. Iooss and P. Lemaître. *A review on global sensitivity analysis methods*. 2014. arXiv: 1404.2405 [math.ST].

[93] S.R. Islam, W. Eberle, S.K. Ghafoor, and M. Ahmed. *Explainable Artificial Intelligence Approaches: A Survey*. 2021. arXiv: 2101.09429 [cs.AI].

[94] G. James, D. Witten, T. Hastie, and R. Tibshirani. In: *An Introduction to Statistical Learning: with Applications in R*. New York, NY: Springer US, 2021. ISBN: 978-1-0716-1418-1. DOI: 10.1007/978-1-0716-1418-1_3. URL: https://doi.org/10.1007/978-1-0716-1418-1_3.

[95] M. Jetzki. “The propagation of air transport delays in Europe”. MA thesis. EUROCONTROL / RWTH Aachen University, Dec. 2009.

[96] M. Jian, D. Zhengyang, L. Yang, L. Dingliang, D. Dingyu, and Z. Yang. “A Method of Flight Off-Block Time Prediction Based on LSI-CNN Model”. In: *2019 IEEE 1st International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. 2019, pp. 95–99. DOI: 10.1109/ICCASIT48058.2019.8973223.

[97] U. Johansson, C. Sönströd, T. Löfström, and H. Boström. “Rule extraction with guarantees from regression models”. In: *Pattern Recognition* 126 (2022), p. 108554. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2022.108554. URL: https://www.sciencedirect.com/science/article/pii/S0031320322000358.

[98] S.A. Kalogirou. “Chapter eleven - Designing and Modeling Solar Energy Systems”. In: *Solar Energy Engineering*. Boston: Academic Press, 2009, pp. 553–664. ISBN: 978-0-12-374501-9. DOI: https://doi.org/10.1016/B978-0-12-374501-9.00011-X. URL: https://www.sciencedirect.com/science/article/pii/B978012374501900011X.

[99] U. Kamath and J. Liu. In: *Explainable Artificial Intelligence: An Introduction to Interpretable Machine Learning*. Cham: Springer International Publishing, 2021. ISBN: 978-3-030-83356-5. DOI: 10.1007/978-3-030-83356-5_1. URL: https://doi.org/10.1007/978-3-030-83356-5_1.

[100] E. Kayacan and M.A. Khanesar. “Chapter 2 - Fundamentals of Type-1 Fuzzy Logic Theory”. In: *Fuzzy Neural Networks for Real Time Control Applications*. Butterworth-Heinemann, 2016, pp. 13–24. ISBN: 978-0-12-802687-8. DOI: https://doi.org/10.1016/B978-0-12-802687-8.00002-5. URL: https://www.sciencedirect.com/science/article/pii/B9780128026878000025.

[101] J. Kelleher. *Deep Learning*. Jan. 2019. ISBN: 9780262354899. DOI: 10.7551/mitpress/11171.001.0001.

[102] L.A. Kerr and D.R. Goethel. “Chapter Twenty One - Simulation Modeling as a Tool for Synthesis of Stock Identification Information”. In: *Stock Identification Methods (Second Edition)*. Second Edition. San Diego: Academic Press, 2014, pp. 501–533. ISBN: 978-0-12-397003-9. DOI: https://doi.org/10.1016/B978-0-12-397003-9.00021-7. URL: https://www.sciencedirect.com/science/article/pii/B9780123970039000217.

[103] B. Kim, R. Khanna, and O. Koyejo. “Examples Are Not Enough, Learn to Criticize! Criticism for Interpretability”. In: *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 2288–2296. ISBN: 9781510838819.

[104] N.K. Kitson, A.C. Constantinou, Z. Guo, Y. Liu, and K. Chobtham. “A survey of Bayesian Network structure learning”. In: *Artificial Intelligence Review* (Jan. 2023). ISSN: 1573-7462. DOI: 10.1007/s10462-022-10351-w. URL: https://doi.org/10.1007/s10462-022-10351-w.

[105] J.P.C. Kleijnen and J.C. Helton. “Statistical analyses of scatterplots to identify important factors in large-scale simulations, 2: robustness of techniques”. In: *Reliability Engineering & System Safety* 65.2 (1999), pp. 187–197. ISSN: 0951-8320. DOI: https://doi.org/10.1016/S0951-8320(98)00090-8. URL: https://www.sciencedirect.com/science/article/pii/S0951832098000908.

[106] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. ISBN: 0262013193.

[107] V. Kuleshov and S. Ermon. *Structure learning for Bayesian networks*. Online. https://ermongroup.github.io/cs228-notes/learning/structure/ accessed on 25/04/2023. Mar. 2022.

[108] W.F. Lamberti. “3 - An overview of explainable and interpretable AI”. In: *AI Assurance*. Ed. by Feras A. Batarseh and Laura J. Freeman. Academic Press, 2023, pp. 55–123. ISBN: 978-0-323-91919-7. DOI: https://doi.org/10.1016/B978-0-32-391919-7.00015-9. URL: https://www.sciencedirect.com/science/article/pii/B9780323919197000159.

[109] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. “Exploring Strategies for Training Deep Neural Networks”. In: *Journal of Machine Learning Research* 1 (Jan. 2009). DOI: 10.1145/1577069.1577070.

[110] J. Li. “Assessing the accuracy of predictive models for numerical data: Not r nor r2, why not? Then what?” In: *PLOS ONE* 12.8 (Aug. 2017), pp. 1–16. DOI: 10.1371/journal.pone.0183250. URL: https://doi.org/10.1371/journal.pone.0183250.

[111] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. *Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization*. 2018. arXiv: 1603.06560 [cs.LG].

[112] W. Liao and Q. Ji. “Learning Bayesian network parameters under incomplete data with domain knowledge”. In: *Pattern Recognition* 42.11 (2009), pp. 3046–3056. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2009.04.006. URL: https://www.sciencedirect.com/science/article/pii/S0031320309001472.

[113] IVAO Documentation Library. *Clearance Delivery*. https://wiki.ivao.aero/en/home/divisions/xu/atc/positions/delivery accessed on 30/03/2023. Dec. 2022.

[114] OAG Aviation Worldwide Limited. *Punctuality League Report 2019 - On time performance for airlines and airports on full year data 2019*. 2020.

[115] Z.C. Lipton. *The Mythos of Model Interpretability*. 2017. arXiv: 1606.03490 [cs.LG].

[116] P.J.G. Lisboa, S. Saralajew, A. Vellido, R. Fernández-Domenech, and T. Villmann. “The coming of age of interpretable and explainable machine learning models”. In: *Neurocomputing* 535 (2023), pp. 25–39. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2023.02.040. URL: https://www.sciencedirect.com/science/article/pii/S0925231223001893.

[117] C. Liu et al. “Airport flight ground service time prediction with missing data using graph convolutional neural network imputation and bidirectional sliding mechanism”. In: *Applied Soft Computing* 133 (2023), p. 109941. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2022.109941. URL: https://www.sciencedirect.com/science/article/pii/S1568494622009905.

[118] S. Liu, J. McGree, Z. Ge, and Y. Xie. *Computational and Statistical Methods for Analysing Big Data with Applications*. Ed. by S. Liu, J. McGree, Z. Ge, and Y. Xie. San Diego: Academic Press, 2016. ISBN: 978-0-12-803732-4. DOI: https://doi.org/10.1016/B978-0-12-803732-4.00001-5. URL: https://www.sciencedirect.com/science/article/pii/B9780128037324000015.

[119] Luchtvaartnieuws. *Voor de Zomer Voorlopig Besluit Over Minder Grondafhandelaars Schiphol*. https://www.luchtvaartnieuws.nl/nieuws/categorie/3/airports/voor-de-zomer-voorlopig-besluit-over-minder-grondafhandelaars-schiphol accessed on 28/03/2023. Mar. 2023.

[120] S.M. Lundberg and S.I. Lee. "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

[121] M. Luo, M. Schultz, H. Fricke, and B. Desart. "Data-driven fusion of turnaround sub-processes to predict aircraft ground time". In: *ATRS World Conference*. Aug. 2022.

[122] M. Luo, M. Schultz, H. Fricke, B. Desart, F. Herrema, and R. Barragán Montes. "Agent-based simulation for aircraft stand operations to predict ground time using machine learning". In: *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*. 2021, pp. 1–8. DOI: 10.1109/DASC52595.2021.9594325.

[123] D.J.C. MacKay. *Information Theory, Inference & Learning Algorithms*. USA: Cambridge University Press, 2002. ISBN: 0521642981.

[124] A.L. Madsen, Michael Lang, Uffe B. Kjærulff, and Frank Jensen. "The Hugin Tool for Learning Bayesian Networks". In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Ed. by T.D. Nielsen and N.L. Zhang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 594–605. ISBN: 978-3-540-45062-7.

[125] M. Magris and A. Iosifidis. "Bayesian learning for neural networks: an algorithmic survey". In: *Artificial Intelligence Review* (Mar. 2023). ISSN: 1573-7462. DOI: 10.1007/s10462-023-10443-1. URL: https://doi.org/10.1007/s10462-023-10443-1.

[126] P. Malighetti, C. Morlotti, R. Redondi, and S. Paleari. "The turnaround tactic and on-time performance: Implications for airlines' efficiency". In: *Research in Transportation Business & Management* 46 (2023). Themed Volume on Transport Efficiency, p. 100874. ISSN: 2210-5395. DOI: https://doi.org/10.1016/j.rtbm.2022.100874. URL: https://www.sciencedirect.com/science/article/pii/S2210539522000955.

[127] W.S. McCulloch and W. Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *Bulletin of Mathematical Biophysics* 5.4 (Dec. 1943), pp. 115–133.

[128] Royal Schiphol Group Mediarelaties. *Schiphol connected to European network through CDM*. https://news.schiphol.com/schiphol-connected-to-european-network-through-cdm/ accessed on 22/06/2023. May 2018.

[129] H. Mendoza et al. "Towards Automatically-Tuned Deep Neural Networks". In: *Automated Machine Learning: Methods, Systems, Challenges*. Ed. by F. Hutter, L. Kotthoff, and J. Vanschoren. Cham: Springer International Publishing, 2019, pp. 135–149. ISBN: 978-3-030-05318-5. DOI: 10.1007/978-3-030-05318-5_7. URL: https://doi.org/10.1007/978-3-030-05318-5_7.

[130] B. Mihaljevic, M.C. Bielza Lozoya, and P.M. Larrañaga Múgica. "bnclassify: Learning Bayesian Network Classifiers". In: *R JOURNAL* 10.2 (2018), pp. 455–468. ISSN: 2073-4859. DOI: 10.32614/rj-2018-073. URL: https://journal.r-project.org/archive/2018/RJ-2018-073/index.html.

[131] B. Mihaljevi, C. Bielza, and P. Larrañaga. "Bayesian networks for interpretable machine learning and optimization". In: *Neurocomputing* 456 (2021), pp. 648–665. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.01.138. URL: https://www.sciencedirect.com/science/article/pii/S0925231221009644.

[132] T. Miller. *Explanation in Artificial Intelligence: Insights from the Social Sciences*. 2018. arXiv: 1706.07269 [cs.AI].

[133] R. Mitchell and E. Frank. "Accelerating the XGBoost algorithm using GPU computing". In: *PeerJ Computer Science* 3 (July 2017).

[134] T.J. Mofokeng and A. Marnewick. "Factors contributing to delays regarding aircraft during A-check maintenance". In: *2017 IEEE Technology & Engineering Management Conference (TEMSCON)*. 2017, pp. 185–190. DOI: 10.1109/TEMSCON.2017.7998375.

[135] C. Molnar. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2nd ed. LeanPub, Mar. 2023.

[136] J.N. Morgan and J.A. Sonquist. “Problems in the Analysis of Survey Data, and a Proposal”. In: *Journal of the American Statistical Association* 58.302 (1963), pp. 415–434. ISSN: 01621459. URL: http://www.jstor.org/stable/2283276 (visited on 05/11/2023).

[137] R. Mori. “Off-block Time Prediction Using Operators Prediction History”. In: *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*. 2021, pp. 1–7. DOI: 10.1109/DASC52595.2021.9594354.

[138] A. Mukherjee, S.R. Grabbe, and B. Sridhar. “Predicting Ground Delay Program At An Airport Based On Meteorological Conditions”. In: *14th AIAA Aviation Technology, Integration, and Operations Conference*. DOI: 10.2514/6.2014-2713. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2014-2713. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2014-2713.

[139] M.Z. Naser. “An engineer’s guide to eXplainable Artificial Intelligence and Interpretable Machine Learning: Navigating causality, forced goodness, and the false perception of inference”. In: *Automation in Construction* 129 (2021). ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2021.103821. URL: https://www.sciencedirect.com/science/article/pii/S0926580521002727.

[140] A. Natekin and A. Knoll. “Gradient boosting machines, a tutorial”. In: *Frontiers in Neurorobotics* 7 (2013). ISSN: 1662-5218. DOI: 10.3389/fnbot.2013.00021. URL: https://www.frontiersin.org/articles/10.3389/fnbot.2013.00021.

[141] K.B. Naveen and P.V. Kumar. “Learning Parameters for Hybrid Bayesian Network”. In: *International Conference on Mobile Computing and Sustainable Informatics*. Ed. by J.S. Raj. Cham: Springer International Publishing, 2021, pp. 255–260. ISBN: 978-3-030-49795-8.

[142] R.E. Neapolitan and X. Jiang. *Probabilistic methods for financial and marketing informatics*. Morgan Kaufmann, May 2014. ISBN: 978-0-12-370477-1. DOI: https://doi.org/10.1016/B978-0-12-370477-1.X5016-6.

[143] S. Neumann. “Is the boarding process on the critical path of the airplane turn-around?” In: *European Journal of Operational Research* 277.1 (2019), pp. 128–137. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2019.02.001. URL: https://www.sciencedirect.com/science/article/pii/S0377221719301110.

[144] M. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015. ISBN: 978-3319944623.

[145] A. Norin, T.A. Granberg, D. Yuan, and P. Värbrand. “Airport logistics A case study of the turn-around process”. In: *Journal of Air Transport Management* 20 (2012). Notes, pp. 31–34. ISSN: 0969-6997. DOI: https://doi.org/10.1016/j.jairtraman.2011.10.008. URL: https://www.sciencedirect.com/science/article/pii/S0969699711001037.

[146] HSE Office. *Zakboek Safety & Security*. Tech. rep. Amsterdam Airport Schiphol, Jan. 2023.

[147] B. Oreschko, T. Kunze, M. Schultz, H. Fricke, V. Kumar, and L. Sherry. “Turnaround Prediction with Stochastic Process Times and Airport Specific Delay Pattern”. In: *5th International Conference on Research in Airport Transportation*. June 2012. URL: https://catsr.vse.gmu.edu/pubs/Kumar_ICRAT2012.pdf.

[148] B. Oreschko, M. Schultz, J. Elflein, and H. Fricke. “Significant Turnaround Process Variations due to Airport Characteristics”. In: *Air Transport & Operations Symposium 2010*. June 2010. DOI: 10.3233/978-1-60750-559-4-263.

[149] B. Oreschko, M. Schultz, and H. Fricke. “Skill Analysis of Ground Handling Staff and Delay Impacts for Turnaround Modeling”. In: *Air Transport and Operations*. Jan. 2011. DOI: 10.3233/978-1-60750-812-0-310.

[150] M.A. Palacios-Alonso, C.A. Brizuela, and L.E. Sucar. “Evolutionary Learning of Dynamic Naive Bayesian Classifiers”. In: *Journal of Automated Reasoning* 45.1 (June 2010), pp. 21–37.

[151] D.C. Pandey, G.S. Kushwaha, and S. Kumar. “Mamdani fuzzy rule-based models for psychological research”. In: *SN Applied Sciences* 2.5 (Apr. 2020), p. 913. ISSN: 2523-3971. DOI: 10.1007/s42452-020-2726-z. URL: https://doi.org/10.1007/s42452-020-2726-z.

[152] C.P. Pappis and E.H. Mamdani. “A Fuzzy Logic Controller for a Trafc Junction”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 7.10 (1977), pp. 707–717. DOI: 10.1109/TSMC.1977.4309605.

[153] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. ISBN: 1558604790.

[154] D.M.W. Powers. *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. 2020. arXiv: 2010.16061 [cs.LG].

[155] S. Putatunda and K. Rama. "A Comparative Analysis of Hyperopt as Against Other Approaches for Hyper-Parameter Optimization of XGBoost". In: *Proceedings of the 2018 International Conference on Signal Processing and Machine Learning*. SPML '18. Shanghai, China: Association for Computing Machinery, 2018, pp. 6–10. ISBN: 9781450366052. DOI: 10.1145/3297067.3297080. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3297067.3297080.

[156] M.A. Quddus, R.B. Noland, and W.Y. Ochieng. "A High Accuracy Fuzzy Logic Based Map Matching Algorithm for Road Transport". In: *Journal of Intelligent Transportation Systems* 10.3 (2006), pp. 103–115. DOI: 10.1080/15472450600793560. eprint: https://doi.org/10.1080/15472450600793560. URL: https://doi.org/10.1080/15472450600793560.

[157] J.J. Randolph. "A Guide to Writing the Dissertation Literature Review." In: *Practical Assessment, Research and Evaluation* 14 (2009), p. 13. DOI: https://doi.org/10.7275/b0az-8t74.

[158] S. Reitmann and M. Schultz. "Real-time Prediction of Aircraft Boarding". In: *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. 2018, pp. 1–9. DOI: 10.1109/DASC.2018.8569370.

[159] ResearchHubs. *Tsukamoto Fuzzy Model*. https://researchhubs.com/post/engineering/fuzzy-system/tsukamoto-fuzzy-model.html accessed on 17/04/2023. 2015.

[160] M.T. Ribeiro, S. Singh, and C. Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. DOI: https://doi.org/10.48550/arXiv.1602.04938. arXiv: 1602.04938 [cs.LG].

[161] M.T. Ribeiro, S. Singh, and C. Guestrin. *Model-Agnostic Interpretability of Machine Learning*. 2016. arXiv: 1606.05386 [stat.ML].

[162] R.F. Ropero, P. Aguilera, A. Fernández, and R. Rumí. "Regression using hybrid Bayesian networks: Modelling landscapesocioeconomy relationships". In: *Environmental Modelling & Software* 57 (July 2014). DOI: 10.1016/j.envsoft.2014.02.016.

[163] L.T. Rose and K.W. Fischer. "Garbage In, Garbage Out: Having Useful Data Is Everything". In: *Measurement: Interdisciplinary Research and Perspectives* 9.4 (2011), pp. 222–226. DOI: 10.1080/15366367.2011.632338. eprint: https://doi.org/10.1080/15366367.2011.632338. URL: https://doi.org/10.1080/15366367.2011.632338.

[164] A.R. Rout. *Advantages and Disadvantages of Logistic Regression*. Jan. 2023. URL: %5Curl%7Bhttps://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/%7D%20accessed%20on%2018/07/2023.

[165] A. Saltelli et al. "Experimental Designs". In: *Global Sensitivity Analysis. The Primer*. John Wiley & Sons, Ltd, 2007. Chap. 2, pp. 53–107. ISBN: 9780470725184. DOI: https://doi-org.tudelft.idm.oclc.org/10.1002/9780470725184.ch2. eprint: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/pdf/10.1002/9780470725184.ch2. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1002/9780470725184.ch2.

[166] R.L. Sapra. "Using R2 with caution". In: *Current Medicine Research and Practice* 4.3 (2014), pp. 130–134. ISSN: 2352-0817. DOI: https://doi.org/10.1016/j.cmrp.2014.06.002. URL: https://www.sciencedirect.com/science/article/pii/S2352081714000646.

[167] M. Sato and H. Tsukimoto. "Rule extraction from neural networks via decision tree induction". In: *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222)*. Vol. 3. 2001, pp. 1870–1875. DOI: 10.1109/IJCNN.2001.938448.

[168] L.J. Saunders, R.A. Russell, and D.P. Crabb. "The Coefficient of Determination: What Determines a Useful R 2 Statistic?" In: *Investigative Ophthalmology & Visual Science* 53.11 (Oct. 2012), pp. 6830–6832. ISSN: 1552-5783. DOI: 10.1167/iovs.12-10598. eprint: https://arvojournals.org/arvo/content_public/journal/iovs/932976/i1552-5783-53-11-6830.pdf. URL: https://doi.org/10.1167/iovs.12-10598.

[169] M. Scanagatta, A. Salmerón, and F. Stella. "A survey on Bayesian network structure learning from data". In: *Progress in Artificial Intelligence* 8.4 (Dec. 2019), pp. 425–439. ISSN: 2192-6360. DOI: 10.1007/s13748-019-00194-y. URL: https://doi.org/10.1007/s13748-019-00194-y.

[170] R.E. Schapire. “Explaining AdaBoost”. In: *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*. Ed. by B. Schölkopf, Z. Luo, and V. Vovk. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 37–52. ISBN: 978-3-642-41136-6. DOI: 10.1007/978-3-642-41136-6_5. URL: https://doi.org/10.1007/978-3-642-41136-6_5.

[171] *Schiphol Airport CDM Operations Manual*. Royal Schiphol Group. Sept. 2019. URL: https://www.schiphol.nl/nl/download/b2b/1569488978/7ERl8iHeLELDtgFsnK0mGi.pdf.

[172] M. Schmidt. “A review of aircraft turnaround operations and simulations”. In: *Progress in Aerospace Sciences* 92 (May 2017). DOI: 10.1016/j.paerosci.2017.05.002.

[173] M. Schultz. “Fast Aircraft Turnaround Enabled by Reliable Passenger Boarding”. In: *Aerospace (EISSN 2226-4310)* 5 (Jan. 2018), p. 8. DOI: 10.3390/aerospace5010008.

[174] M. Schultz and H. Fricke. “Improving Aircraft Turnaround Reliability”. In: *ICRAT - International Conferences on Research in Air Transportation*. June 2008.

[175] M. Schultz, T. Kunze, and H. Fricke. “Boarding on the critical path of the turnaround”. In: *Proceedings of the 10th USA/Europe Air Traffic Management Research and Development Seminar, ATM 2013*. June 2013.

[176] M. Schultz, T. Kunze, B. Oreschko, and H. Fricke. “Microscopic Process Modelling for Efficient Aircraft Turnaround Management”. In: *International Air Transport and Operations Symposium*. July 2013. URL: https://api.semanticscholar.org/CorpusID:198160711.

[177] G. Scott. *Fuzzy Logic: Definition, Meaning, Examples and History*. https://www.investopedia.com/terms/f/fuzzy-logic.asp#:~:text=Fuzzy%20logic%20is%20an%20approach,an%20array%20of%20accurate%20conclusions. accessed on 17/04/2023. Apr. 2023.

[178] M. Scutari, C.E. Graafland, and J.M. Gutiérrez. “Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms”. In: *International Journal of Approximate Reasoning* 115 (2019), pp. 235–253. ISSN: 0888-613X. DOI: https://doi.org/10.1016/j.ijar.2019.10.003. URL: https://www.sciencedirect.com/science/article/pii/S0888613X19301434.

[179] N.R. Shadbolt and P.R. Smart. “Knowledge Elicitation: Methods, Tools and Techniques”. In: *Evaluation of Human Work*. 4. CRC Press, May 2015.

[180] I.N. da Silva, D. Hernane Spatti, R. Andrade Flauzino, L.H.B. Liboni, and S.F. dos Reis Alves. “Artificial Neural Network Architectures and Training Processes”. In: *Artificial Neural Networks : A Practical Course*. Cham: Springer International Publishing, 2017, pp. 21–28. ISBN: 978-3-319-43162-8. DOI: 10.1007/978-3-319-43162-8_2. URL: https://doi.org/10.1007/978-3-319-43162-8_2.

[181] J. Skorupski and M. Wierzbisk. “A method to evaluate the time of waiting for a late passenger”. In: *Journal of Air Transport Management* 47 (2015), pp. 79–89. ISSN: 0969-6997. DOI: https://doi.org/10.1016/j.jairtraman.2015.05.001. URL: https://www.sciencedirect.com/science/article/pii/S0969699715000563.

[182] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju. *Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods*. 2020. arXiv: 1911.02508 [cs.LG].

[183] L.N. Smith. *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*. 2018. arXiv: 1803.09820 [cs.LG].

[184] A. Soberón. *Interpreting Machine Learning Models Using LIME and SHAP*. https://svitla.com/blog/interpreting-machine-learning-models-lime-and-shap#:~:text=LIME%20generates%20a%20perturbed%20dataset,while%20SHAP%20requires%20multiple%20observations accessed on 13/04/2023. Dec. 2022.

[185] I.M. Sobol. “Sensitivity Estimates for Nonlinear Mathematical Models”. In: *Mathematical Modelling and Computational Experiments* 4 (1993), pp. 407–414.

[186] D.P. Solomatine and D.L. Shrestha. “AdaBoost.RT: a boosting algorithm for regression problems”. In: *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*. Vol. 2. 2004, pp. 1163–1168. DOI: 10.1109/IJCNN.2004.1380102.

[187] B. van Stein, E. Raponi, Z. Sadeghi, N. Bouman, R.C.H.J. Van Ham, and T. Bäck. “A Comparison of Global Sensitivity Analysis Methods for Explainable AI With an Application in Genomic Prediction”. In: *IEEE Access* 10 (2022), pp. 103364–103381. DOI: 10.1109/ACCESS.2022.3210175.

[188] F. Stella and Y. Amer. “Continuous time Bayesian network classifiers”. In: *Journal of Biomedical Informatics* 45.6 (2012), pp. 1108–1119. ISSN: 1532-0464. DOI: https://doi.org/10.1016/j.jbi.2012.07.002. URL: https://www.sciencedirect.com/science/article/pii/S1532046412000998.

[189] T. Stephenson. “An Introduction to Bayesian Network Theory and Usage”. In: *IDIAP* (Jan. 2000).

[190] A. Subasi. “Chapter 3 - Machine learning techniques”. In: *Practical Machine Learning for Data Analysis Using Python*. Academic Press, 2020, pp. 91–202. ISBN: 978-0-12-821379-7. DOI: https://doi.org/10.1016/B978-0-12-821379-7.00003-5. URL: https://www.sciencedirect.com/science/article/pii/B9780128213797000035.

[191] P.K. Sundararajan and O.J. Mengshoel. “A Genetic Algorithm for Learning Parameters in Bayesian Networks using Expectation Maximization”. In: *Proceedings of the Eighth International Conference on Probabilistic Graphical Models*. Ed. by A. Antonucci, G. Corani, and C.P. Campos. Vol. 52. Proceedings of Machine Learning Research. Lugano, Switzerland: PMLR, Sept. 2016, pp. 511–522. URL: https://proceedings.mlr.press/v52/sundararajan16.html.

[192] R. Susmita. “A Quick Review of Machine Learning Algorithms”. In: *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. 2019, pp. 35–39. DOI: 10.1109/COMITCon.2019.8862451.

[193] B.G. Tabachnick and L.S. Fidell. *Using multivariate statistics*. 7th ed. Upper Saddle River, NJ: Pearson, July 2018. ISBN: 9780134790541.

[194] Y. Tang. “Airline Flight Delay Prediction Using Machine Learning Models”. In: *Proceedings of the 2021 5th International Conference on E-Business and Internet*. ICEBI ’21. Singapore, Singapore: Association for Computing Machinery, 2022, pp. 151–154. ISBN: 9781450385657. DOI: 10.1145/3497701.3497725. URL: https://doi.org/10.1145/3497701.3497725.

[195] Eurocontrol Airport CDM Team. *THE MANUAL - Airport CDM Implementation*. Tech. rep. Eurocontrol, Mar. 2017. URL: https://www.eurocontrol.int/sites/default/files/publication/files/airport-cdm-manual-2017.PDF.

[196] B. Thiagarajan, L. Srinivasan, A.V. Sharma, D. Sreekanthan, and V. Vijayaraghavan. “A machine learning approach for prediction of on-time performance of flights”. In: *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*. 2017, pp. 1–6. DOI: 10.1109/DASC.2017.8102138.

[197] Y. Tong and Q. Ji. “Learning Bayesian Networks with qualitative constraints”. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587368.

[198] M. Tosin, A.M.A. Côrtes, and A. Cunha. “A Tutorial on Sobol’ Global Sensitivity Analysis Applied to Biological Models”. In: *Networks in Systems Biology: Applications for Disease Modeling*. Ed. by F.A.B. da Silva, N. Carels, M. Trindade dos Santos, and F.J.P. Lopes. Cham: Springer International Publishing, 2020, pp. 93–118. ISBN: 978-3-030-51862-2. DOI: 10.1007/978-3-030-51862-2_6. URL: https://doi.org/10.1007/978-3-030-51862-2_6.

[199] Tronair. *How an airport ground crew can prepare for serious storms*. https://www.tronair.com/resources/airport-ground-crew-storm-operations/ accessed on 21/06/2023. 2023.

[200] S. Tschiatschek and F. Pernkopf. “Parameter Learning of Bayesian Network Classifiers Under Computational Constraints”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by A. Appice, P.P. Rodrigues, V. Santos Costa, C. Soares, J. Gama, and A. Jorge. Cham: Springer International Publishing, 2015, pp. 86–101. ISBN: 978-3-319-23528-8.

[201] G. Van den Broeck, K. Mohan, A. Choi, and J. Pearl. *Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data*. 2014. arXiv: 1411.7014 [cs.LG].

[202] T. Van Steenkiste, J. van der Herten, I. Couckuyt, and T. Dhaene. “Data-Efficient Sensitivity Analysis with Surrogate Modeling”. In: *Uncertainty Modeling for Engineering Applications*. Ed. by F. Canavero. Cham: Springer International Publishing, 2019, pp. 55–69. ISBN: 978-3-030-04870-9. DOI: 10.1007/978-3-030-04870-9_4. URL: https://doi.org/10.1007/978-3-030-04870-9_4.

[203] R.S. Verboon. “Deicing Scheduling”. http://resolver.tudelft.nl/uuid:e384c95b-8cdb-4dc4-aab5-e85c6acadb6c. MA thesis. Delft University of Technology, June 2016.

[204] S. Vieira, W.H.L. Pinaya, and A. Mechelli. “Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications”. In: *Neuroscience & Biobehavioral Reviews* 74.Pt A (Mar. 2017).

[205] C. Villa-Blanco, P. Larrañaga, and C. Bielza. “Multidimensional continuous time Bayesian network classifiers”. In: *International Journal of Intelligent Systems* 36.12 (2021), pp. 7839–7866. DOI: https://doi-org.tudelft.idm.oclc.org/10.1002/int.22611. eprint: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/pdf/10.1002/int.22611. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1002/int.22611.

[206] M. Vladimirova, J. Verbeek, P. Mesejo, and J. Arbel. *Understanding Priors in Bayesian Neural Networks at the Unit Level*. 2019. arXiv: 1810.05193 [stat.ML].

[207] H. Wang and D.Y. Yeung. “A Survey on Bayesian Deep Learning”. In: *ACM Computer Survey* 53.5 (Sept. 2020). ISSN: 0360-0300. DOI: 10.1145/3409383. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3409383.

[208] Z. Wang and A.C. Bovik. “Mean squared error: Love it or leave it? A new look at signal fidelity measures”. In: *IEEE signal processing magazine* 26.1 (2009), pp. 98–117.

[209] S. Wasserkrug, R. Marinescu, S. Zeltyn, E. Shindin, and Y.A. Feldman. “Learning the Parameters of Bayesian Networks from Uncertain Data”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.13 (May 2021), pp. 12190–12197. DOI: 10.1609/aaai.v35i13.17447. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17447.

[210] G.I. Webb. “Naïve Bayes”. In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by C. Sammut and G.I. Webb. Boston, MA: Springer US, 2016, pp. 1–2. ISBN: 978-1-4899-7502-7. DOI: 10.1007/978-1-4899-7502-7_581-1. URL: https://doi.org/10.1007/978-1-4899-7502-7_581-1.

[211] T. Wiegand, F. Jeltsch, I. Hanski, and V. Grimm. “Using pattern-oriented modeling for revealing hidden information: A key for reconciling ecological theory and application”. In: *Oikos* 100 (Feb. 2003), pp. 209–222. DOI: 10.1034/j.1600-0706.2003.12027.x.

[212] W. Wiegerinck, B. Kappen, and W. Burgers. “Bayesian Networks for Expert Systems: Theory and Practical Applications”. In: *Interactive Collaborative Information Systems*. Ed. by R. Babuka and F.C.A. Groen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 547–578. ISBN: 978-3-642-11688-9. DOI: 10.1007/978-3-642-11688-9_20. URL: https://doi.org/10.1007/978-3-642-11688-9_20.

[213] A.G. Wilson. *The Case for Bayesian Deep Learning*. 2020. arXiv: 2001.10995 [cs.LG].

[214] Y. Wong. *Why You Should Use Bayesian Network*. https://towardsdatascience.com/why-you-should-use-bayesian-neural-network-aaf76732c150#:~:text=What%20is%20Bayesian%20Neural%20Network,that%20best%20fit%20the%20data. accessed on 16/05/2023. Oct. 2021.

[215] ACI World. “The future of global air travel”. In: *Airport World* 29.1 (Feb. 2024), pp. 18–19.

[216] C.L. Wu. “Monitoring Aircraft Turnaround Operations Framework Development, Application and Implications for Airline Operations”. In: *Transportation Planning and Technology* 31.2 (2008), pp. 215–228. DOI: 10.1080/03081060801948233. URL: https://doi.org/10.1080/03081060801948233.

[217] J. Yang. *Fast TreeSHAP: Accelerating SHAP Value Computation for Trees*. 2022. arXiv: 2109.09847 [cs.LG].

[218] X.S. Yang. In: *Introduction to Algorithms for Data Mining and Machine Learning*. Academic Press, 2019. ISBN: 978-0-12-817216-2. DOI: https://doi.org/10.1016/B978-0-12-817216-2.00008-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128172162000089.

[219] C.Y. Yiu, K.K.H. Ng, K.C. Kwok, W. Tung Lee, and H.T. Mo. “Flight delay predictions and the study of its causal factors using machine learning algorithms”. In: *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. 2021, pp. 179–183. DOI: 10.1109/ICCASIT53235.2021.9633571.

[220] L.A. Zadeh. “Fuzzy sets”. In: *Information and Control* 8.3 (1965), pp. 338–353. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(65)90241-X. URL: https://www.sciencedirect.com/science/article/pii/S001999586590241X.

[221] B.P. Zeigler, A. Muzy, and E. Kofman. In: *Theory of Modeling and Simulation (Third Edition)*. Third Edition. Academic Press, 2019, pp. 3–25. ISBN: 978-0-12-813370-5. DOI: https://doi.org/10.1016/B978-0-12-813370-5.00009-2. URL: https://www.sciencedirect.com/science/article/pii/B9780128133705000092.

[222] S.Z. Zhang, Z.N. Zhang, N.H. Yang, J.Y. Zhang, and X.K. Wang. “An improved EM algorithm for Bayesian networks parameter learning”. In: *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*. Vol. 3. 2004, 1503–1508 vol.3. DOI: 10.1109/ICMLC.2004.1382011.

[223] J. Zhao and B.K. Bose. “Evaluation of membership functions for fuzzy logic controlled induction motor drive”. In: *IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02*. Vol. 1. 2002, 229–234 vol.1. DOI: 10.1109/IECON.2002.1187512.

[224] C. Zhe, L. Yang, L. Yifan, X. Gaoyang, and Z. Yi. “Time Prediction of Off-Block Model Based on Serial Process Neural Network”. In: *2019 IEEE 1st International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. 2019, pp. 120–124. DOI: 10.1109/ICCASIT48058.2019.8973174.

[225] H. Zhou, W. Li, Z. Jiang, F. Cai, and Y. Xue. “Flight Departure Time Prediction Based on Deep Learning”. In: *Aerospace* 9 (July 2022), p. 394. DOI: 10.3390/aerospace9070394.

[226] Y. Zhou, N. Fenton, and C. Zhu. “An empirical study of Bayesian network parameter learning with monotonic influence constraints”. In: *Decision Support Systems* 87 (2016), pp. 69–79. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.dss.2016.05.001. URL: https://www.sciencedirect.com/science/article/pii/S0167923616300744.

[227] L. Zonglei, W. Jiandong, and Z. Guansheng. “A New Method to Alarm Large Scale of Flights Delay Based on Machine Learning”. In: *2008 International Symposium on Knowledge Acquisition and Modeling*. 2008, pp. 589–592. DOI: 10.1109/KAM.2008.18.

[228] M. Zoutendijk and M. Mitici. “Probabilistic flight delay predictions using machine learning and applications to the flight-to-gate assignment problem”. In: *Aerospace* 8.6 (May 2021), p. 152. DOI: https://doi.org/10.3390/aerospace8060152.

[229] H. Zuo, Y. Ding, X. Sang, and T. Xu. “Flight Off-Block Time Prediction Based on BP Neural Network Optimized by Genetic Algorithm”. In: *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*. 2021, pp. 1–6. DOI: 10.1109/ICCASIT53235.2021.9633492.