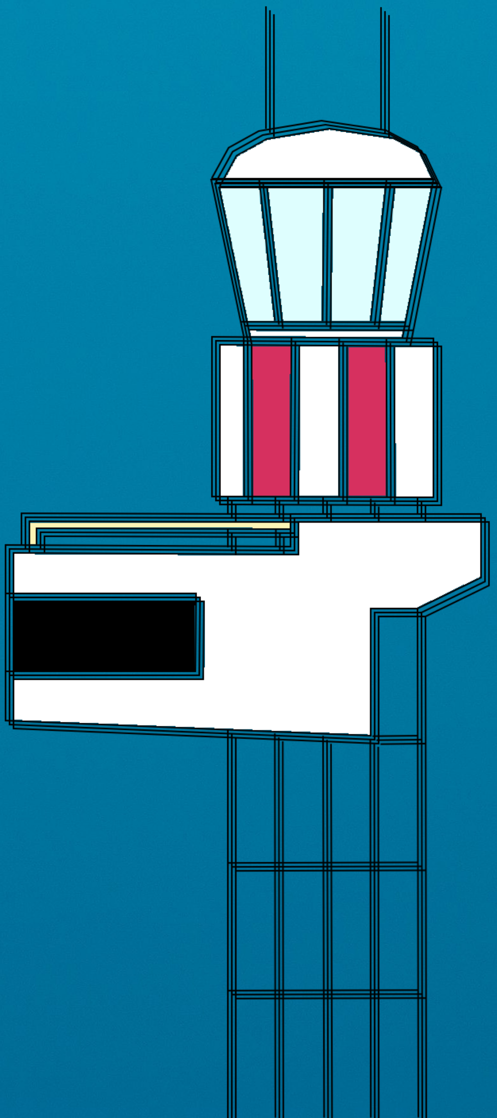
Predicting Flight Delay Distributions

A Machine Learning-Based Approach at a Regional Airport

S.C.M Dutrieux



Predicting Flight Delay Distributions

A Machine Learning-Based Approach at a Regional Airport

by

S.C.M. Dutrieux

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday, February 22, 2021 at 10:00 AM.

Student number:	4139615	
Project duration:	February 2020 – February 2021	
Thesis committee:	Dr. M.D. Pavel	TU Delft, Chair
	Dr. F. Avallone	TU Delft, Examiner
	Dr. M.A. Mitici	TU Delft, Supervisor

Acknowledgements

This report presents the findings of my Master thesis research for my Aerospace Engineering degree at Delft University of Technology, with a specialization in Air Transport and Operations. Doing research on flight delay predictions at a time when so much of the aviation industry was brought to a halt felt pretty strange at times. However, I do believe that this research will be relevant in the near future when we finally get to fly places again.

With the unpredictability of 2020, it was especially important to get the steady supervision that Dr. Mihaela Mitici and PhD student Mike Zoutendijk brought to this project. I am grateful to Mihaela for providing structure, giving very helpful feedback, and really making the time for constructive discussions. I would also very much like to thank Mike for the help with clearly defining problems and formulas. Thank you both so much.

Finally, I would like to thank my parents Susan and Richard, my sister Rosa and my brother Julian, and my girlfriend (and cover designer) Iris for all their love and support and the endless cups of coffee.

Sarah Dutrieux
Amsterdam, January 2021

Contents

List of Figures

List of Tables

Nomenclature

List of abbreviations

1D	One Day
1D	One Month
ADS-B	Automatic Dependent Surveillance-Broadcast
AUC	Area Under the ROC Curve
CDF	Cumulative Distribution Function
CRPS	Continuous Ranked Probability Score
DNN	Deep Neural Network
DOM	Day Of Month
DOW	Day Of Week
DOY	Day Of Year
DRPS	Discrete Ranked Probability Score
DT	Decision Tree
ECDF	Empirical Cumulative Distribution Function
EFE	Exclusive Feature Elimination
EIN	Eindhoven Airport
FAA	Federal Aviation Administration
FGAP	Flight-to-Gate Assignment Problem
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FSOS	Fraction of Samples within One Standard deviation
GOSS	Gradient Based One Side Sampling
LightGBM	Light Gradient-Boosted Machine
LP	Linear Program
MAE	Mean Absolute Error
METAR	Meteorological Aerodrome Report
MSE	Mean Squared Error
PDF	Probability Density Function
RF	Random Forest

RFE	Recursive Feature Elimination
RMSE	Root Mean Squared Error
ROC	Receiver Operating Curve
RTM	Rotterdam The Hague Airport
SD	Standard Deviation
SHAP	SHapley Additive exPlanations
SMOTE	Synthetic Minority Oversampling Technique
STA	Scheduled Time of Arrival
STD	Scheduled Time of Departure
TN	True Negative
TP	True Positive
TPR	True Positive Rate

List of symbols

α_i	the set of predicted mixture coefficients $\alpha_{i,l}$ for each Gaussian l of flight i
$\bar{\eta}$	the average efficiency score
$\bar{\rho}$	the average robustness score
$\bar{y}_i$	the estimated mean of the delay of flight i
β	the current maximum number of flights assigned at one gate
$^{\circ}F$	degree Fahrenheit
η	the efficiency score of a flight-to-gate schedule
$\hat{y}_{i,j}$	the delay prediction of estimator j for flight i
$\hat{y}_{i,k}$	the delay prediction of run k for flight i
μ_i	the set of predicted means $\mu_{i,l}$ for each Gaussian l of flight i
ϕ_i	Shapley value
ρ	the robustness score of a flight-to-gate schedule
σ_i	the set of predicted standard deviations $\sigma_{i,l}$ for each Gaussian l of flight i
B	the set of all bins of a histogram
b	a bin of a histogram
$c_{i,j}$	the cost of assigning flight i to gate j
$C_{j,t}$	the set of all possible combinations with at least two flights i from $F_{j,t}$
D	the set of test days d
d	a single test day
e	the number of estimators in the RF

F	the set of all features
$f(p_{i,t}, r)$	a scaling function
$F(x)$	the cumulative density function
f_1	the f_1 score
f_s	the expected output of the model for input feature set S
$F_{j,t}$	the set of flights i assigned to gate j at time t
g	the number of Gaussians used for the Mixture Density Model
$G_{j,t}$	the number of flights scheduled at gate j at time t with presence probability $p_{i,t} \geq 0.5$
g_{pres}	the presence probability function
h	the number of runs of the Dropout Network
$H_{j,t}$	the set of flights i present at gate j at time t
K	the set of time steps
k	the total number of time steps
M	the set of gates available at the airport
m	the total number of available gates at the airport
N	the set of flights to be scheduled
n	the total number of flights to be scheduled
n_s	the number of samples
$o_{i,k}$	a variable that indicates whether bin k is larger than the actual value
$p_{i,t}$	the presence probability of flight i at time t
p_{max}	the maximum allowable presence probability for a second flight
$q_{i,k}$	the cumulative probability of sample set i at bin k
r	the maximum allowed overlap probability
$r_{true_{j,t}}$	the true overlap probability that is calculated for each gate j and time step t
S	a subset of F without feature i
s^2	the estimated variance
S_i	a feature's encoded value
s_{avg}	the average standard deviation
$s_{i,t}$	a variable that indicates whether flight i has a non-zero probability to be present at time step t
T	the set of test flights
t_{cycle}	the time span of one complete cycle
t_{time}	a time
$v_{j,t}$	a variable that indicates whether at least one flight has a presence probability of at least 0.5 for gate j and time t

$w_{j,t}$ a variable that indicates whether there is a conflict at gate j at time t

X_i a feature's categorical value

X_{arr} a random variable that indicates the time step at which flight i arrives at the airport

X_{dep} a random variable that indicates the time step at which flight i departs from the airport

$x_{i,j,t}$ a decision variable which indicates whether flight i is assigned to gate j at time step t

Y a feature's target value

y_i the actual delay of flight i

z_i a variable that indicates whether prediction is within one standard deviation of the actual value

Introduction

Accurate flight delay predictions are important for many stakeholders throughout the aviation industry, including airports, airlines, and passengers. As a result, numerous researches have attempted to predict flight delays as accurately as possible. Although many are rather successful, and classification accuracies above 80% are not uncommon, there is not much variation in the prediction target; almost all researches aim to predict the flight delay as either a class or in minutes. While that might be sufficient information for some applications, this research expects that certain airport operation optimization models could be improved by including an indication of how certain the model is about a specific flight delay prediction, ideally in the form of a complete probability distribution. Another underexposed topic in existing literature is how well flight delays can be predicted at regional airports. While larger airports might be the larger stakeholders, this research expects that many regional airports could also benefit from accurate predictions.

The main research objective of this thesis is to gain insight into the possibility and potential effect of accurately predicting flight delay probability distributions with machine learning algorithms at a regional airport. To achieve this, the problem is divided into three parts. Since most existing flight delay studies evolve around large, international airports, the first sub-goal of this thesis is to apply the same machine learning-based binary classifiers to a regional airport. The second sub-goal is to evolve flight delay predictions from point estimates to probability distributions by applying probabilistic machine learning algorithms, a novelty in the field of flight delay predictions. The third and final sub-goal is to investigate the potential effect of the predicted flight delay distribution by incorporating them into an existing Flight-to-Gate Assignment Problem.

This thesis research is conducted at the Air Transport and Operations department of the Aerospace Engineering faculty of Delft University of Technology. Although this department collaborates with Rotterdam The Hague Airport, the airport is not directly involved in the development of this thesis. All data comes from publicly available sources. The research project is unique for its attempt to form a bridge between two research fields; the field of flight delay predictions and the field of scheduling problems. Its results have the potential to improve airport operations, including the scheduling of flights to gates, which could be beneficial for airports, airlines and passengers. The applied methodology also has the potential to improve logistical operations outside the aviation industry, such as parcel delivery routing or the scheduling of other modes of transportation.

This thesis report is organized as follows. Firstly, the scientific paper is presented in Part I. The second part re-states the literature study, which has previously been graded under a different course name. The final part consists of detailed work supporting the paper.

I

Scientific Paper

Machine learning-based predictions of flight delay distributions at a regional airport

Sarah Dutrieux [*]

Delft University of Technology, Delft, The Netherlands

Abstract

In an effort to improve an airport operation optimization model, this research investigates the possibility of predicting probability distributions of flight delays with machine learning algorithms. The research is centered around Rotterdam The Hague Airport, a regional airport in the Netherlands. The first objective is to test how well machine learning classifiers can predict whether a flight will be delayed for a regional airport. This results in accuracies of around 70%, while taking precision and recall into account. The second objective is to predict the probability distributions of flight delays, for which three models are selected: a modified Random Forest Regressor, a Mixture Density Network and a Dropout Network. The main finding is that accurately predicting distinctive delay probability distributions for individual flights is possible. As a final objective, the predicted flight delay distributions are incorporated into an existing Flight-to-Gate Assignment Problem. It is found that this improves the robustness of the resulting schedules, although associated with a small reduction in their efficiency. The overall conclusion of this research is that machine learning-based prediction of flight delay distributions is possible, sufficiently accurate, and can improve at least one airport operation optimization problem. Further research will have to show whether this approach can be extended to other airports, other aviation optimization problems, or even optimization problems in other research areas.

1 Introduction

Throughout the aviation industry, there are many stakeholders who can benefit from accurate flight delay predictions. Airports, who are optimizing the efficient use of their existing capacity to accommodate for the growing air travel demand. Airlines, who aim to minimize the propagation effect of disruptions in their schedule. And last but not least, the passengers, who prefer to know about delays as soon as possible. As a result, numerous flight delay researches have been performed within the field of air transport and operations.

Simultaneously, numerous researches investigate how the operational processes of airports and airlines can be optimized. These optimizations, such as airport surface traffic optimization, aircraft routing optimization, and airline crew scheduling, all involve arriving and departing aircrafts, for which delays are inevitable. Subsequently, a number of these airport operation optimization models take uncertainty into account. However, they rarely take full advantage of research findings in the field of flight delay predictions. The hypothesis of this paper is that certain airport operation optimization models could be improved by directly incorporating an extensive flight delay prediction model.

To prove the hypothesis, this paper attempts to improve the Flight-to-Gate Assignment Problem (FGAP) model as defined by van Schaijk and Visser (2017). This model, however, requires the flight delay predictions to be expressed as a probability distribution, which exposes a first limitation of the current state-of-the-art in the field of flight delay predictions. Most of the flight delay studies apply machine learning algorithms to predict whether or not a flight will arrive or depart within 15 minutes of its scheduled time, which corresponds to the delay reporting system of the U.S. Federal Aviation Administration (FAA). Several studies attempt to predict the delay in minutes. But while the majority of these studies properly state the overall performance accuracy of their models, they do not provide a confidence interval for the delay predictions of individual flights, let alone a complete probability distribution.

A second limitation of the current state-of-the-art in the field of flight delay predictions is that the majority of the researches targets large, international airports with large, and available, historical flight delay datasets. A smaller, regional airport inherently has a smaller dataset, which might be challenging when applying machine learning algorithms. Furthermore, a regional airport likely serves an airline mixture with more low cost carriers, who might have less resources available to solve delaying situations. Flight-to-gate scheduling, however, is relevant for all airports that serve passenger airlines, which includes regional airports.

[*] Msc Student, Air Transport and Operations, Faculty of Aerospace Engineering, Delft University of Technology

Combining the potential improvement in airport operations and the limitations of current flight delay studies leads to the main objective of this paper: *investigating whether it possible and potentially beneficial to accurately predict the probability distribution of flight delays with machine learning algorithms at a regional airport.* The first research objective is to predict flight delays at a regional airport, by applying two established machine learning classifiers that are currently used for larger airports. The second objective is to investigate whether it is possible to accurately predict flight delay probability distributions. In order to do this, three machine learning algorithms that have successfully estimated probability distributions in other research fields are introduced to the field of flight delay predictions. The final objective is to show how enhancing flight delay predictions with a probability distribution can be beneficial for airport operations optimization by incorporating the newly predicted delay distributions into an existing flight-to-gate assignment problem.

The remainder of this paper is divided as follows. Section 2 presents a literature review that elaborates on the current state-of-the-art in the field of flight delay predictions. A description of the data available for this research is given in section 3. Section 4 describes the methodology used for the prediction of flight delays and their distributions. The results are presented in section 5. In section 6, a case study is conducted to examine the potential impact of the predicted flight delay distributions. Finally, the conclusions and recommendations of this research are given in section 7.

2 Literature Review

Over the past two decades, many different approaches have been applied to the flight delay prediction problem. One of the earlier studies by Mueller and Chatterij (2002) approaches the flight delay problem by fitting different distributions to historical data. Although not strictly a prediction, the research concludes that arrival flights are best modeled by a Normal distribution, while a Poisson distribution best describes the departure flights. In (Xu et al., 2005), the problem is modeled as a Bayesian network, which requires the conditional probabilities between states to be known. Klein (2010) recognizes the importance of the weather and incorporates weather forecasts and observations in a multi-linear regression.

The beginning of the previous decade highlights the start of a new trend in the field of delay predictions: machine learning techniques. A large contribution is made by Rebollo and Balakrishnan (2014), who are among the first to apply a binary Random Forest (RF) classifier to the flight delay problem and achieve an average accuracy of 80%. Both the method and the accuracy often function as a benchmark in successive research.

Since tree-based models are relatively easy to interpret, almost all studies nowadays still incorporate at least one bagging or boosting extension of the Decision Tree (DT) model. However, in order to achieve more accurate results, many recent studies also include more complex models. A promising addition to the boosting algorithms is LightGBM, which allows for leaf-wise growth of a sequentially updated decision tree and outperforms all other models in both (Lambelho et al., 2020) and (Shao et al., 2019). Neural networks, a deep learning approach that requires many data points, are introduced to flight delay predictions in (Kim et al., 2016) and (Khanmohammadi et al., 2016). Another approach is the construction of a two stage model that first determines whether there is a delay, followed by how much delay (Thiagarajan et al., 2017). Even more recently, Yu et al. (2019) have developed a combination of a Deep Belief Network and a Support Vector Regression.

For the input features, the trend of increasing complexity over time does not hold as strongly, although some researches specifically focus on the effects of innovative features. The minimum input requirement for predicting flight delays are the features related to flight schedule of the historical flight delay data, such as the origin, destination, and scheduled time of departure. In (Choi et al., 2016), weather forecast features are added to the prediction model. However, most researches that follow continue to use the actual, observed weather in the form of METAR data, which is easier to obtain. Interestingly enough, none of those researches identifies METAR as one of the most important features, even though de Neufville and Odoni (2013) and Mueller and Chatterij (2002) state that weather is often reported as the cause of delay. Finally, Yu et al. (2019) and Chen and Li (2019) emphasize the benefits of knowing the delay of the previous flight executed by a specific aircraft.

The increased complexity of models and features is rewarded; the two stage model of Thiagarajan et al. (2017) results in an accuracy of 94.35%, Yu et al. (2019) report that 99.3% of the predicted delay minutes are within 25 minutes of the actual value, and the research of Chen and Li (2019) leads to a (relaxed) accuracy of 92.7%. A critical note, however, is placed at the fact that not all of these specific studies, and similar ones, explicitly state their false negatives and false positives. This information is essential for the interpretation of these results, given that the dataset of flight delays is highly imbalanced. Furthermore, it should be noted from a practical perspective that the most accurate results are from predictions very shortly before the departure of the flight. For certain operational optimization solutions, this might be too late. Nevertheless, this literature review on flight delay predictions shows that very high accuracies can be achieved. Rather than aiming to improve them even further, this research focuses on two underexposed and potentially innovative topics within the field of flight delay predictions: regional airports and probabilistic forecasting.

Regional airports

Most existing flight delay researches revolve around large, international airports. Although these airports are large stakeholders, with large and available datasets, there are also many regional airports that could benefit from accurate predictions. With respect to flight delay predictions at an international airport, a regional airport differs in two ways. It has a different mixture of airlines, i.e. one with more low cost carriers, and it has a smaller historical flight dataset.

An assumption associated with low cost carriers is that they might have less resources available to absorb delays within their schedule, resulting in more difficult-to-predict flight delays. However, both Horiguchi et al. (2017) and McCarthy et al. (2019) explicitly focus on low cost carriers and their prediction results are only slightly inferior to the results of large airports. With a prediction horizon of one day in advance, Horiguchi et al. (2017) achieve an AUC score of 0.647 which is only marginally lower that the 0.68 achieved in (Choi et al., 2016), a study at 45 major airports, likely serving a large share of legacy carriers.

The smaller database that is inherently associated with a regional airport makes machine learning-based predictions more challenging and prone to overfitting, but not impossible. In (McCarthy et al., 2019), the transfer learning framework of Moon and Carbonell (2017) is used to enlarge a training dataset of a smaller low cost carrier with data from a much larger airline. With all other parameters remaining the same, the RMSE reduced from 10.2 to 9.2 when transfer learning was applied. Furthermore, Gui et al. (2020) show a good classification accuracy with a dataset of only 5,761 flights and the application of undersampling to counter the imbalanced dataset. Overall it is concluded that the main concerns associated with applying existing delay prediction models to regional airports, are not necessarily limiting factors according to literature.

Probabilistic forecasting

The second identified gap in literature relates to the target of the flight delay predictions. All of the encountered delay studies that involve machine learning algorithms aim to predict the delay in either minutes or a delay class. Almost none of them consider the importance and potential of adding probabilities to the estimate, let alone including an entire probability density function. The motivation for probabilistic forecasting therefore comes from promising studies in other research fields. Examples are (Zhu and Laptev, 2017), where neural networks with dropout are applied for the prediction of time series uncertainty, and (Vossen et al., 2018), which applies a mixture density network to predict power load peaks in energy networks. This research aims to introduce similar techniques to the field of flight delay predictions.

3 Data description

Conducting this research requires two types of data; data of the airport itself and data of its historical flights. Both are described in the following sections.

Rotterdam The Hague Airport

This research is centered around Rotterdam The Hague Airport (RTM), a regional airport located in the Netherlands. It serves flights to destinations across Europe, which is illustrated in figure 2(a). This figure is obtained by plotting the geographical coordinates of the airports in the available flight dataset. The five airports that occur most frequently are marked with their IATA code. As listed in table 1, RTM currently serves around two million passengers a year, divided over eight different airlines. Furthermore, the airport has one runway and one recently renovated terminal, which opened in December 2020. The new terminal has 11 gates, which is three more than the old terminal had. Since this research is based on flights that arrived or departed prior to December 2020, it is based on the old terminal. Its layout, including a marking of the eight gates, is given in figure 1.

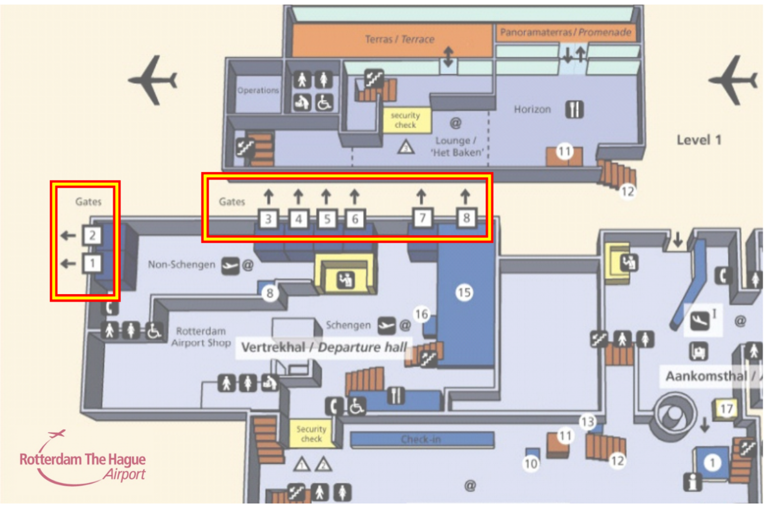
	RTM
Gates	8
Terminals	1
Runways	1 (06/24; 2200m)
Airlines	8
Passengers 2019	2,133,976

Table 1: Information Rotterdam The Hague Airport [1]

	RTM
Number of flights	34,678
Arrivals	17,317
Departures	17,361
Time period	May 2017 - Feb 2020

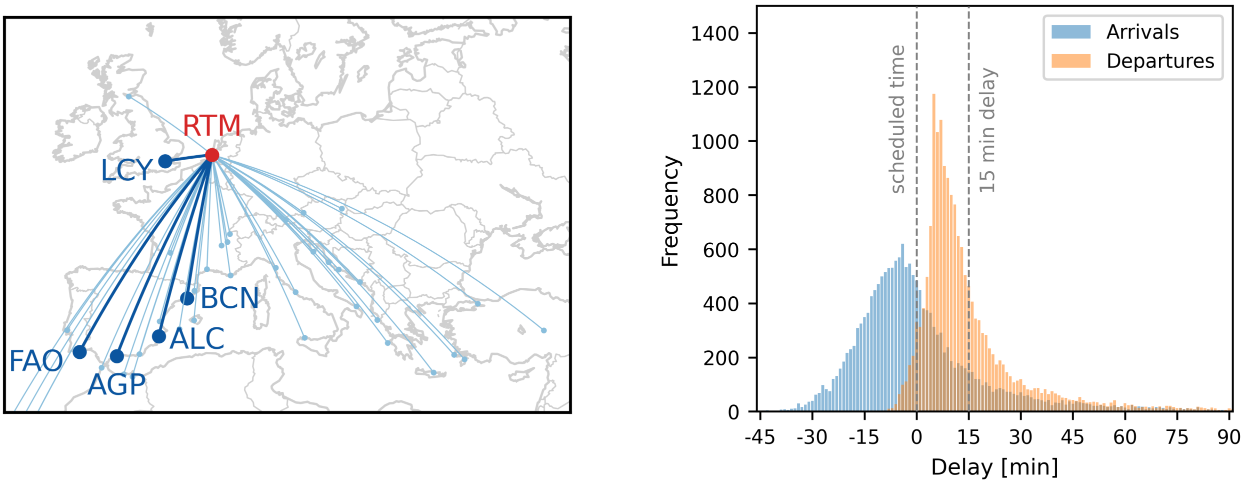
Table 2: Information available data

[1] https://www.rotterdamthehagueairport.nl



Source: based on an original map of Rotterdam The Hague Airport[1]

Figure 1: Layout of the terminal of RTM prior to 2020



(a) Flight network

(b) Historical arrival and departure delay

Figure 2: Visualization of the available RTM data

Available data and features

This research is based on 34,678 historical passenger flights that arrived or departed from RTM airport in the period between May 2017 and February 2020, as listed in table 2. The historical arrival delays and departure delays in minutes are illustrated individually in figure 2(b). These histograms show that the arrival flights tend to arrive before their scheduled time, while the departing flights tend to depart slightly late.

All data are publicly available and collected from two main sources: Flightradar24[2] and the Meteorological Aerodrome Report (METAR) database of Iowa State University [3]. The first is a real-time global flight tracking service that collects its data with ADS-B receivers. For each tracked flight it provides flight schedule information, such as the origin, destination, airline, scheduled departure/arrival times , and the actual time of departure and arrival. The second source provides information regarding the weather in the form of the METAR weather code of the World Meteorological Organization. These codes are provided by airports and weather stations and are typically updated twice an hour.

A number of features cannot be found directly in one of the two sources above, but are derived from the schedule. The first feature in this category is the distance, which can be estimated with the geographical

[2] https://www.flightradar24.com/

[3] https://mesonet.agron.iastate.edu/request/download.phtml

coordinates of the origin and destination airport. The second feature is the scheduled flight time, which is the difference between the scheduled time of departure and arrival. The final feature is the number of other flights scheduled at the airport in the time window of an hour before to an hour after the flight in question. An overview of all available and considered features can be found in table 3.

Source	**Features**
FlightRadar24[2]	aircraft type, airline, country of destination, country of origin, day of week, day of year, destination airport, flight number, month, origin airport, Scheduled Time of Arrival (STA), Scheduled Time of Departure (STD), week, year
METAR database[3]	for both the destination and origin: dew point temperature, pressure altimeter, temperature, wind speed
derived from schedule	distance, scheduled flight time, other flights scheduled at airport

Table 3: All available features

4 Methodology of flight delay prediction

The methodology of this research that is used for the prediction of flight delays, can roughly be divided into three parts; the encoding and selection of features, the binary classification of delays, and the prediction of flight delay distributions. All three are described in detail in the following sections.

4.1 Features and data

As stated in the data description, this paper is completely based on data from publicly available sources. Before the features in table 3 can be used in the predictions models, the data have to be encoded, selected, and balanced first.

Encoding of categorical features

Since a number of the available features described above are categorical, while machine learning algorithms only accept numerical features, feature encoding is required. This research uses three types of encoding, the first being the use of geographical coordinates. For all features that represent a certain location, e.g. the origin airport and country, the current label of the feature is replaced with its corresponding geographical coordinates.

The second type of encoding is trigonometric encoding, which adds cyclical information to time-based features by projecting them on a unit circle. This is done with the following two formulas:

$$sin\left(\frac{2\pi t_{time}}{t_{cycle}}\right) \qquad \text{and} \qquad cos\left(\frac{2\pi t_{time}}{t_{cycle}}\right) \tag{1}$$

Here, t_{time} represents the time to be converted and t_{cycle} the time span of one complete cycle, for example 365 days or 24 hours. With this type of encoding, a single time-based feature is divided into two partial features; one representing the cosine part and the other representing the sine part.

The final type of encoding is target encoding. When target encoding is used in a classification problem, the feature's current label is replaced with the probability of the target being 1, given the feature's categorical value. This can be formulated more formally as follows:

$$X_i \rightarrow S_i \cong P(Y|X = X_i) \tag{2}$$

where X_i is the feature's current categorical value, S_i the new encoding and Y the binary target variable with value 1. When target encoding is used in a regression problem with a continuous target, the encoding analogously becomes the expected value of the target given the categorical value. Again, more formally formulated this becomes:

$$X_i \rightarrow S_i \cong E(Y|X = X_i) \tag{3}$$

In this case Y represents the continuous target variable. The encoding used in the final feature selection is included in tables 4 and 5.

Feature selection process

The main reason to deploy a feature selection is to avoid multicollinearity; the phenomenon that two or more explanatory variables are highly correlated to each other. For some machine learning algorithms, in particular regression models that assume all variables to be independent, multicollinearity reduces the performance. A proper feature selection contributes to prevent this.

Another reason to apply feature selection is that not all features are appropriate for every application. The availability of weather related features strongly depends on the time between the prediction and the observation, also known as the prediction horizon. In a real life application, the model would be based on weather forecasts. However, for this research only actual weather observations, in the form of METAR data, are available. For predictions with a horizon of at most one day (1D), it is assumed that the weather forecasts would be accurate enough to approximate these forecasts with the actual METAR data. In fact, METAR data could be interpreted as perfect weather forecasts. For longer prediction horizons, for example one month (1M), this assumption does not hold. Instead, these predictions are based on the multiple year daily average of the METAR data.

Furthermore, there is distinction between arriving flights and departure flights; not all features have to be equally useful for both. To accommodate for these differences, this flight delay research distinguishes between four different study groups: 1) 1D arrivals, 2) 1D departures, 3) 1M arrivals, and 4) 1M departures. For each of these groups the appropriate feature set is selected with a two step process.

Step 1: Pearson correlation matrix

The first step is a selection based on the Pearson correlation matrix of all features available for the study group. When two features have a correlation coefficient of 0.8 or higher at least one of the two features is removed, a decision that is based on domain knowledge. More details of this selection procedure can be found in appendix A of the associated thesis, where both the complete and resulting correlation matrices are presented.

Step 2: Recursive Feature Elimination

The second step is the Recursive Feature Elimination (RFE) method as described in detail in (Granitto et al., 2006). In summary this is an algorithm that eliminates redundant features by systematically running the model with different subsets and comparing the performance. Although thorough, this approach is also rather time consuming. Nevertheless, this research runs the model twice for verification purposes; once with the Random Forest model and once with LightGBM. Only features that are eliminated in both cases are removed from the final feature set, which results in the removal of the feature 'airline' for all study groups. The final feature sets for the arrivals and departures are listed in table 4 and 5 respectively.

Horizon	Feature	Type	Encoding
Any	Origin airport	C	Target, geographical coordinates
	Country of origin	C	Target
	Aircraft type	C	Target
	Distance [km]	N	-
	Scheduled time of departure (STD)	N	Trigonometric
	Year	N	-
	Day of Year (DOY)	N	Trigonometric*
	Day of Month (DOM)	N	Trigonometric*
	Day of Week (DOW)	N	Trigonometric*
	Other flights scheduled at RTM	N	-
1-Month	Average temperature [°F] at origin/destination	N	-
	Average pressure altimeter [cm] at origin/destination	N	-
	Average wind speed [$knots$] at origin/destination	N	-
1-Day	Temperature [°F] at origin/destination	N	-
	Wind speed [$knots$] at origin/destination	N	-
	Pressure [cm] at origin/destination	N	-

* These features are also included without trigonometric encoding

Table 4: Feature set for arrival flight delay predictions

Horizon	Feature	Type	Encoding
Any	Destination airport	C	Target, geographical coordinates
	Country of destination	C	Target
	Aircraft type	C	Target
	Distance [km]	N	-
	Scheduled time of departure (STD)	N	Trigonometric
	Year	N	-
	Day of Year (DOY)	N	Trigonometric*
	Day of Month (DOM)	N	Trigonometric*
	Day of Week (DOW)	N	Trigonometric*
	Other flights scheduled at RTM	N	-
1-Month	Average temperature [°F] at origin airport	N	-
	Average wind speed [$knots$] at origin airport	N	-
	Average Pressure [cm] at origin airport	N	-
1-Day	Temperature [°F] at origin airport	N	-
	Wind speed [$knots$] at origin airport	N	-
	Pressure [cm] at origin airport	N	-

[*] These features are also included without trigonometric encoding

Table 5: Feature set for departure flight delay predictions

Imbalanced dataset

When machine learning algorithms are applied to a classification problem, they often produce the best test results when trained on a well-balanced dataset, meaning that all classes are represented equally. When the training dataset is highly imbalanced, the model is trained with a bias towards the majority class, leading to missclassification of the minority class in the test case. As can be seen in figure 2(b), dividing the historical flight data of this research into the target groups 'less than 15 minutes delayed' and 'delayed by 15 minutes or more', leads to such an imbalanced (training) dataset.

A solution to counter this bias is to even the number of samples in each class by oversampling or undersampling the training data. Since the latter has the disadvantage of potentially removing valuable information, the commonly used oversampling method Synthetic Minority Oversampling Technique (SMOTE)(Chawla et al., 2002) is selected. In order to generate more samples of the minority class, SMOTE first draws a line between a random instance of the minority class and one of its randomly selected k-nearest neighbors. It then randomly identifies a point on this line as a new sample in the minority class and continues this process until both (or all) classes have the same sample size.

4.2 Binary classification

Once the features are selected and the dataset is balanced, binary classification models are applied. The aim is to predict whether flights will arrive (or depart) within 15 minutes of their scheduled time of arrival (or departure). In order to achieve this, this paper introduces classifiers that have previously been successful in flight delay studies at larger airports, together with appropriate performance metrics and a model-agnostic interpretability method to identify important features.

Classification models

To enable verification, this research uses two different machine learning classification models the Random Forest Classifier and the Light Gradient-Boosted Machine. A short description of both models is given in the following two sections. For a detailed description of the hyperparameter tuning process and resulting settings, this article refers to appendix B of the associated thesis.

Random Forest Classifier

The first model selected is the widely used Random Forest (RF) Classifier, an algorithm that originates from (Breiman, 2001). In principle this model is a collection of Decision Tree (DT) classifiers. Each tree in the collection is based on a bootstrap sample of the training data, meaning that the sample is drawn uniformly and with replacement. For classification, the RF determines its prediction by taking the majority vote of the test results of each individual tree. Analogously, when applied to a regression problem it determines its output by taking the average. This procedure of sampling and assembling is also known as bootstrap aggregating or bagging.

Light Gradient-Boosted Machine

The second model is LightGBM, which was recently developed by Ke et al. (2017) and stands for Light Gradient-Boosting Machine. Similar to the RF, LightGBM is also an ensemble technique based on the DT algorithm. The main difference however is that this is a boosting algorithm, meaning that it is trained by by improving the decision tree in sequential steps. At each step, a random sample is used to construct a new tree, based on reducing the classification error of the previous tree. LightGBM stands out from other gradient boosting algorithms by applying leaf-wise growing of the tree instead of level-wise growing, which allows the trees to be more complex and more accurate. Furthermore, it introduces two new concepts to reduce the number of features and data samples, and therefore the computational time. The first one is Gradient Based One Side Sampling (GOSS), which selects the samples with large gradients and randomly downsamples features with smaller gradients, to reduce the number of data points with as little as possible information loss. The second is Exclusive Feature Elimination (EFE), which reduces the number of features by bundling features that are mutually exclusive, i.e. are never simultaneously zero.

Performance metrics for classification

		Actual	
		Positive	**Negative**
Predicted	**Positive**	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Table 6: Confusion matrix

$$accuracy = \frac{TP + TN}{total} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$precision = \frac{TP}{TP + FP} \tag{6}$$

The most intuitive and commonly used metric to express the performance of a classification algorithm is the *accuracy* in equation 4, which is the fraction of correctly identified instances. In principle it holds that the higher the score the better, with the score bound between 0 and 1. Accuracy alone, however, is not sufficient when the dataset is highly imbalanced. It is also important to take the false positives and false negatives into account. Therefore, two additional metrics are used as well; the *recall* and the *precision* in equation 5 and 6 respectively. The *recall* is a measurement of how many of the actual positive instances are also classified as positive. The *precision* represents how many of the predicted positives are also actually positive. Again, both metrics are bound between 0 and 1, and a higher score is preferred for both.

It is important to note that precision and recall are a trade-off; increasing either will decrease the other. This research assumes that the costs for falsely predicting a delay and falsely not predicting a delay are equal, which makes both metrics equally important. To find the optimal balance point between the two, the f_1 score is used. This score combines both metrics in a single one by taking their harmonic mean:

$$f_1 = \frac{2 * precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \tag{7}$$

Another widely used performance metric is the Area Under the Receiver Operating Curve (ROC)(AUC). The *ROC* is defined as follows:

$$ROC = \frac{TPR}{FPR} = \frac{\frac{TP}{TP+FN}}{\frac{FP}{FP+TN}} \tag{8}$$

The integrated AUC always falls between 0 and 1, and the closer to 1 the better. It should be observed, however, that randomly guessing the class results in a 0.5 AUC score. A prediction model should ideally perform better than this.

Model-agnostic interpretability method

While the use of more complex machine learning models might lead to more accurate predictions, it comes at the expense of explainability and interpretability. As opposed to for example a regression model, it is impossible to interpret the direction and impact of each individual feature directly from machine learning models. Therefore, model-agnostic interpretability methods are introduced. This research uses the SHAP (SHapley Additive exPlanations) values approach, as proposed by Lundberg and Lee (2017). The approach is based on Shapley values ϕ_i, which are calculated as follows (Lundberg and Lee, 2017):

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!} [f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)] \tag{9}$$

with F the set of all features, $S \subseteq F \setminus \{i\}$ a subset of the set of all features except feature i, and f_S the expected output of the model for input feature set S. In other words, they represent the average contribution of a feature to the prediction, considering all permutations of the available features. The SHAP value approach calculates these values for all, or a sample of, the observations. For each observation, the summation of the SHAP values of all features is equal to the difference between the prediction of the model and the base value, which is the value obtained when predicting without any features. A higher absolute SHAP value indicates a higher contribution of the feature, the sign indicates the direction of that contribution.

4.3 Prediction of distributions

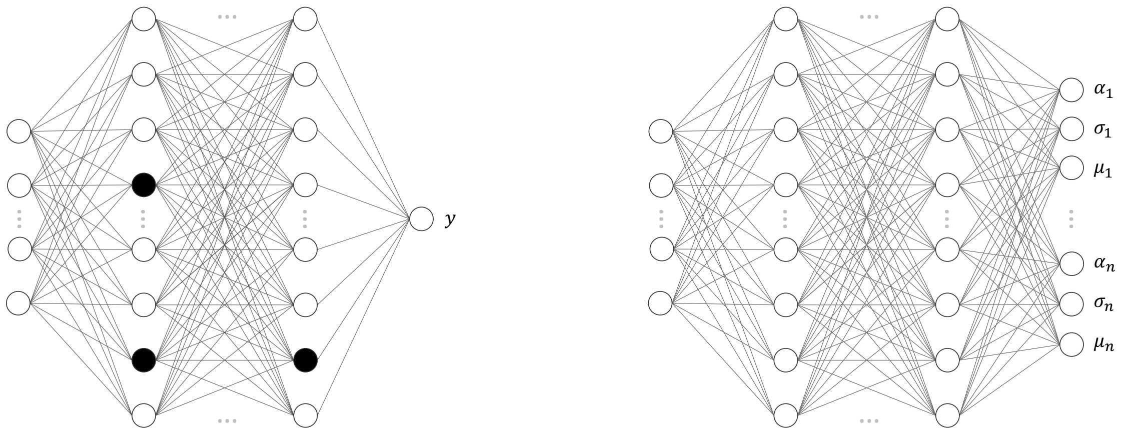
Following the binary classification, this section introduces the methodology for the prediction of flight delay distributions. Since this is a novelty in the field of flight delay predictions, three potential models are selected, accompanied with a set of performance metrics.

Probabilistic forecasting models

The probabilistic forecasting part of this research applies the Random Forest Regressor, the Dropout Network and the Mixture Density Network. A short description of each model is given in the following sections. For a detailed description of the hyperparameter tuning process and resulting settings, this article again refers to appendix B of the associated thesis.

Random Forest Regressor
The first model selected is a modified version of the Random Forest (RF) Regressor as described in (Breiman, 2001). The RF Regressor is a tree-based ensemble learning method that uses the same bagging principle as previously described for the RF Classifier. Different than the classifier variant that is designed to predict a class, the regressor is built to predict a continuous numerical output. In the standard version of the RF Regressor, the outcomes of all individual trees are averaged to generate one single prediction. In this modified version of the RF Regressor, the outcomes of the individual trees are converted to a probability histogram.

Dropout Network
The second selected model is the Dropout Network, as defined by Gal and Ghahramani (2016). In neural networks, the term dropout refers to randomly deactivating certain neurons of the network. It is typically used in the training phase of a model as a regularization method to prevent overfitting. This model uses dropout in the prediction phase of a Deep Neural Network (DNN) to create a Bayesian approximation of the predicted delay. The general architecture of this network is presented in figure 3(a), where the shaded circles illustrate one example of a random dropout setting. By predicting the same input many times, each time with a different random dropout, a probability histogram can be obtained.



(a) Dropout Network

(b) Mixture Density Network

Figure 3: Neural Networks used for probabilistic forecasting

Mixture Density Network
The third and final model is the Mixture Density Network, as defined in (Bishop, 1994). This network aims to estimate a probabilistic density function by extending a standard Deep Neural Network with Gaussian mixture model. In order to achieve this, the output layer of the neural network consists of sets of tree neurons, each set representing a Gaussian function. The first neuron of the set estimates the mixture coefficient, the second neuron the standard deviation and the third the mean. The general architecture of this network is given in figure 3(b). The mixture model that follows the output layer constructs a probability density function based on the mixture coefficients, means, and standard deviations estimated by the network. Given that the resulting density is a mixture of Gaussians, it can approximate the shape of any smooth function, as accurately as the number of Gaussians, in this case determined by the number of neuron sets, allow.

Estimated mean and variance

Most of the performance metrics associated with the probabilistic forecasting models are based on two values; the estimated mean $\bar{y}_i$ and variance s_i^2 of the estimated probability histogram or density function. When applied to flight delay predictions, the RR Regressor and Dropout Network both produce a probability histogram for each flight in the test set. The mean and variance of these histograms can be estimated with the commonly used functions listed in equation set 10 - 11 for the RF Regressor and 12 - 13 for the Dropout Network.

Random Forest Regressor

e : the number of estimators in the RF
$\hat{y}_{i,j}$: the delay prediction of estimator j for flight i
T : the set of test flights

$$\bar{y}_i = \frac{1}{e}\sum_{j=1}^{e} \hat{y}_{i,j}, \qquad \forall i \in T \qquad (10)$$

$$s_i^2 = \frac{1}{e}\sum_{j=1}^{e} \left(\hat{y}_{i,j} - \bar{y}_i\right)^2, \qquad \forall i \in T \qquad (11)$$

Dropout Network

h : the number of runs of the Dropout Network
$\hat{y}_{i,k}$: the delay prediction of run k for flight i
T : the set of test flights

$$\bar{y}_i = \frac{1}{h}\sum_{j=1}^{h} \hat{y}_{i,k}, \qquad \forall i \in T \qquad (12)$$

$$s_i^2 = \frac{1}{h}\sum_{j=1}^{h} \left(\hat{y}_{i,k} - \bar{y}_i\right)^2, \qquad \forall i \in T \qquad (13)$$

When the Mixture Density Network is applied to the flight delay prediction problem, it estimates a probability density function for each of the test flights. Each of these density functions is represented by a set of means, standard deviations and mixture coefficients. According to the original publication of this model (Bishop, 1994), the overall mean and variance of the complete density function can be estimated with equations 14 - 15.

Mixture Density Network

g : the number of Gaussians used for the Mixture Density Model
α_i : the set of predicted mixture coefficients $\alpha_{i,l}$ for each Gaussian l of flight i
μ_i : the set of predicted means $\mu_{i,l}$ for each Gaussian l of flight i
σ_i : the set of predicted standard deviations $\sigma_{i,l}$ for each Gaussian l of flight i
T : the set of test flights

$$\bar{y}_i = \sum_{l=1}^{g} \alpha_{i,l}\, \mu_{i,l}, \qquad \forall i \in T \qquad (14)$$

$$s_i^2 = \sum_{l=1}^{g} \alpha_{i,l} \left(\sigma_{i,l} + \|\mu_{i,l} - \bar{y}_i\|^2\right), \qquad \forall i \in T \qquad (15)$$

Performance metrics for probabilistic forecasting

The performance metrics for probabilistic forecasting can be divided into three groups; metrics for the associated estimated mean, metrics regarding the estimated variance, and metrics that take the complete density or histogram into account. The following sections elaborate on each group separately.

Metrics for point estimates
The first group consists of metrics that are commonly used for single point predictions. This research uses the ones listed in equation set 16 - 19, where $\bar{y}_i$ is the previously explained estimated mean of the delay of flight i,

y_i the actual delay of flight i, and n_s the total number of samples in the test flight set. The Root Mean Squared Error (RMSE), Mean Squared Error (MSE), Mean Absolute Error (MAE), and the Max Error all measure a variant of the error, meaning a lower value is considered better. The metrics are expressed in the same unit as the target variable of the prediction.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_s}(\bar{y}_i - y_i)^2}{n_s}} \tag{16}$$

$$MSE = \frac{\sum_{i=1}^{n_s}(\bar{y}_i - y_i)^2}{n_s} \tag{17}$$

$$MAE = \frac{\sum_{i=1}^{n_s}|\bar{y}_i - y_i|}{n_s} \tag{18}$$

$$\text{Max Error} = max\left(|\bar{y}_i - y_i|\right) \tag{19}$$

Metrics for standard deviations

Secondly, this research uses metrics for the standard deviation of the estimation. Ideally, the standard deviation of the prediction should be as small as possible, while still covering the actual value. To quantify this trade-off, two metrics are used. The first one is the average standard deviation, which is based on the previously explained estimated variance s_i^2 of all flight samples i and is calculated as follows:

$$s_{avg} = \frac{1}{n_s}\sum_{i=1}^{n_s}\sqrt{s_i^2} \tag{20}$$

The second metric counts how often the actual value lies within one absolute standard deviation of the prediction. To calculate this, an additional variable z_i is introduced that indicates whether this holds for each sample flight i. The Fraction of Samples within One Standard deviation (FSOS) can then be calculated by taking the sum over all sample flights i and dividing it by the total number of samples n_s:

$$z_i = \begin{cases} 1, & \text{if } \left(\bar{y}_i - \sqrt{s_i^2}\right) \leq y_i \leq \left(\bar{y}_i + \sqrt{s_i^2}\right), \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

$$\text{FSOS} = \frac{1}{n_s}\sum_{i=1}^{n_s} z_i \tag{22}$$

Metrics for predicted distributions

The final category of performance metrics for probabilistic forecasting takes into account the entire predicted density or histogram. The first metric is the Continuous Ranked Probability Score (CRPS). This score originates from the field of weather forecasts and is defined as follows (Hersbach, 2000):

$$CRPS_i = \int_{-\infty}^{\infty}(F_i(y) - \mathbf{1}(y - y_i))^2 dy \tag{23}$$

The underlying idea of this score is to model the true value y_i as a Heaviside step function $\mathbf{1}(y - y_i)$. The difference between the cumulative density $F_i(y)$ of the predicted function and the corresponding step function, is then squared and integrated to quantify how well the prediction and actual value correspond. The score can be approximated for the samples of the histograms with the analogous Discrete Probability Ranking Score (DRPS):

$$DRPS_i = \frac{1}{B}\sum_{b=1}^{B}(q_{i,b} - o_{i,b})^2 \tag{24}$$

$$o_{i,b} = \begin{cases} 1, & \text{if } b \geq y_i \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

Here, $q_{i,b}$ is the cumulative probability of sample set i at bin b. Similar to the Heaviside step function, $o_{i,b}$ is zero for all bins below the actual value, and one for all bins larger than the actual value. The set of all bins of the histogram is represented by B.

For both scores it holds that the more similar the cumulative probability function is to the actual value modeled as a step input, the lower the score becomes. The minimum score is zero, which is achieved if the algorithm predicts the exact value with a confidence interval width of zero, i.e. the perfect prediction. The further the point estimate of the curve is away from the true value, and the wider the confidence interval, the higher the CRPS or DRPS. The performance of the entire probabilistic forecasting algorithm can be measured by taking the average score over all test samples.

5 Results of flight delay prediction

By applying the methodology of the previous section to the available data of Rotterdam The Hague Airport, two main results are obtained: the results of the binary classification, including an indication of the most important features, and the results of the probabilistic predictions. Together they cover the first two goals of this paper.

5.1 Classification results

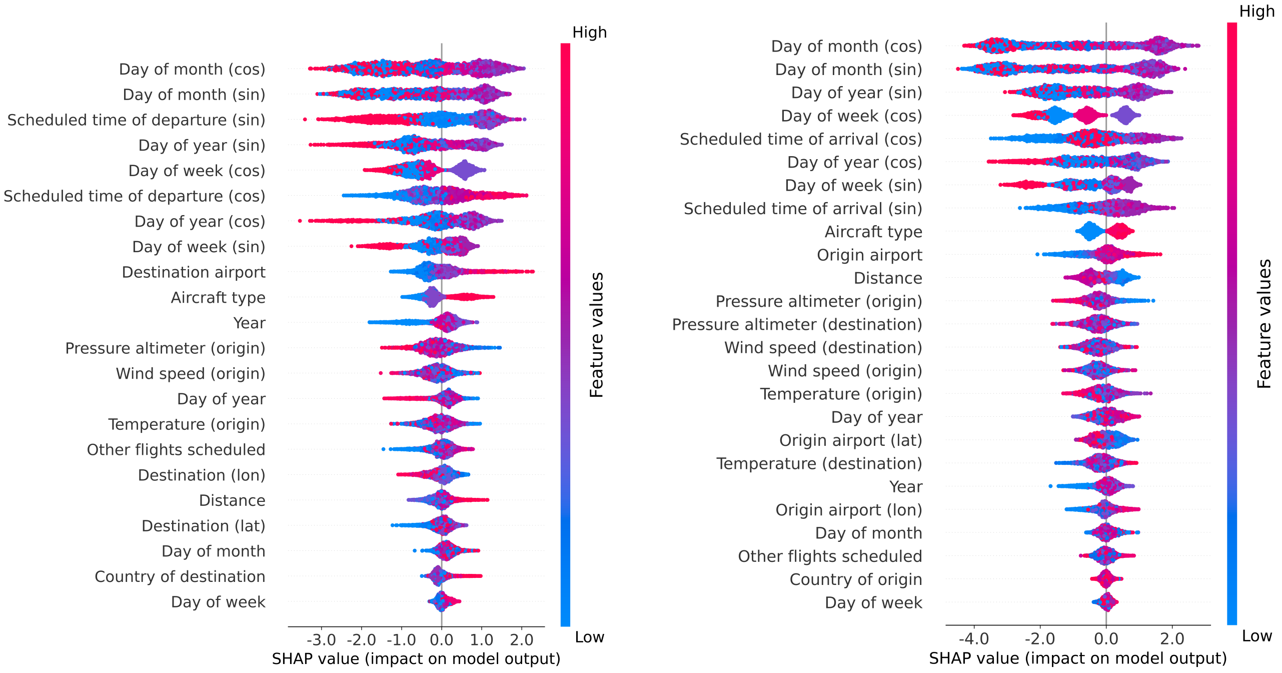
The goal of the first part of this paper is to apply established machine learning classifiers to a regional airport, with the aim to predict whether or not a flight will be delayed by at least 15 minutes. An overview of the results of this binary classification is presented in table 7. All these results are based on a 5-fold prediction. This means that the data are split into five groups, each consisting of 20% of the data. For each group, a prediction is made with a model that is trained on the remaining 80% of the data. The mean and standard deviation for each metric in result table 7 are based on the resulting five predictions.

The first thing to note is that the LightGBM model outperforms the RF Classifier in terms of accuracy. Nevertheless, the results are sufficiently close to interpret them as a verification of both models. All accuracies of the LightGBM model are above 0.7, which is not much inferior to similar studies at large, international airports. As stated earlier, this research considers the recall and precision equally important. The target is to have both scores above 0.5, as this indicates that there are more true positives than false negatives (recall) and more true positives than false positives (precision). It can be seen in table 7 that this is the case for all predictions of the departure delay. For the predictions of the arrival delay, these scores are lower than 0.5. This difference can be explained by the distribution of their historical data, which is illustrated in figure 2(b). Given that the split for the delay classification is 15 minutes, the arrival delay dataset has a higher imbalance ratio than the departure delay dataset. Apparently, oversampling the minority class was not sufficient to counter the majority class bias for the arrivals. Logically, this difference in recall and precision is also seen in the f_1 score. Nevertheless, the AUC score is greater than 0.5 for all cases, which means that all models perform better than a randomly guessing classifier.

A final thing to note is that there is not too much difference between the two different prediction horizons; the predictions that are made a month (1M) in advance are very similar to the predictions of a day (1D) in advance. As stated previously, the only difference between the two horizons is the type of weather features. The 1D predictions use the actually observed METAR weather data, whereas the 1M predictions use an aggregated daily average based on several years of METAR data. Since the differences in performance metrics are small, and the METAR weather forecasts are more easily obtained in real life applications, the remainder of this research focuses on the 1D predictions. It is taken into account that using actually observed weather is impossible in real life predictions. However, this research assumes that the weather forecasts made one day in advance are sufficiently close to the actual weather to justify this approximation.

		Random Forest Classifier		LightGBM	
Classifier	Metric	Mean	std	Mean	std
1M Departures	**accuracy**	**0.674**	$\mathbf{6.9 \times 10^{-3}}$	**0.707**	$\mathbf{7.5 \times 10^{-3}}$
	precision	0.530	1.1×10^{-2}	0.592	8.7×10^{-3}
	recall	0.590	1.9×10^{-2}	0.520	1.8×10^{-2}
	f_1	0.558	1.3×10^{-2}	0.554	1.1×10^{-2}
	AUC	0.655	9.2×10^{-3}	0.664	7.5×10^{-3}
1M Arrivals	**accuracy**	**0.690**	$\mathbf{1.1 \times 10^{-2}}$	**0.797**	$\mathbf{5.8 \times 10^{-3}}$
	precision	0.294	2.3×10^{-2}	0.391	1.7×10^{-2}
	recall	0.527	2.0×10^{-2}	0.255	6.2×10^{-3}
	f_1	0.377	2.0×10^{-2}	0.308	4.0×10^{-3}
	AUC	0.626	1.1×10^{-2}	0.584	2.0×10^{-3}
1D Departures	**accuracy**	**0.674**	$\mathbf{8.8 \times 10^{-3}}$	**0.704**	$\mathbf{9.6 \times 10^{-3}}$
	precision	0.530	1.6×10^{-2}	0.586	2.5×10^{-2}
	recall	0.587	6.3×10^{-3}	0.518	1.7×10^{-2}
	f_1	0.557	9.2×10^{-3}	0.550	1.7×10^{-2}
	AUC	0.654	6.7×10^{-3}	0.661	1.1×10^{-2}
1D Arrivals	**accuracy**	**0.715**	$9.5\times\mathbf{10^{-3}}$	**0.800**	$\mathbf{4.6 \times 10^{-3}}$
	precision	0.292	1.6×10^{-2}	0.402	3.6×10^{-2}
	recall	0.425	6.4×10^{-2}	0.255	1.0×10^{-2}
	f_1	0.344	2.8×10^{-2}	0.312	1.6×10^{-2}
	AUC	0.602	2.2×10^{-2}	0.587	6.1×10^{-3}

Table 7: Results binary classification



(a) 1D departures

(b) 1D arrivals

Figure 4: SHAP values

Feature importance

To gain insight into the contribution of individual features, the model-agnostic SHAP method is used. The summary plots of the SHAP values of the best performing model, the LightGBM model, are given for the 1D arrivals and 1D departures in figure 4(a) and 4(b) respectively. The features in this plot are listed in order of importance, the most important one being at the top. Each dot in the plot represents the SHAP value of a feature for a single observation. The color of the dot indicates the original feature value of the observation. The SHAP values, which for LightGBM are expressed in log-odd units, show the size and direction of the impact of the feature on the prediction. Combining the two gives a correlation. For example, the SHAP plot in figure 4(b) shows for the feature 'Distance' that most observations with a high value (indicated by color) have a medium-sized, negative impact (indicated by the SHAP value) on the expected delay.

Overall, the summary plots show that for both test groups the time-related features, such as the day of the month/year/week and the scheduled time of departure/arrival, contribute most to the prediction. For both the arrivals and the departures, the first non time-related feature is the aircraft type. Most of the weather-related features belong to the lower half of the feature ranking for both the departures and the arrivals. This is noteworthy since extreme weather is an often reported cause of delay. A possible explanation for this difference is that the currently used METAR weather reports of the arrival and departure airport do not capture the en-route weather situation. Although noteworthy, these results are not unexpected; they correspond to the small differences between the 1D and 1M predictions discussed in the previous section.

5.2 Probabilistic forecasting results

The second goal of this research is to investigate the possibility of accurately predicting flight delay distributions with machine learning algorithms. For each of the three models applied, the results are listed in table 8. Similar to the binary classification, the mean and standard deviation are based on a 5-fold prediction. As stated in the methodology, the predicted flight delay distributions are compared by three groups of metrics; metrics for a point estimate, metrics for a confidence interval, and metrics for a complete probability distribution.

The first four metrics in table 8, which are the RMSE, MSE, MAE, and max error, belong to the point estimate metric group and are all expressed in minutes. All four are based on the difference between the expected delay, which is derived from the probability distribution, and the actual delay. This means that a lower score is preferred. The main difference between the RMSE and the MAE is that the RMSE squares the differences before averaging them, which emphasizes the contribution of the larger differences. For the prediction of flight delays, it seems reasonable to assume that large deviations have much more negative impact than small deviations; a deviation of a minute is considered a good result, whereas a deviations of an hour might have costly consequences. The RMSE scores in table 8 show that the standard deviation of the prediction errors is around 25 minutes for all three models, while the maximum error scores show that the prediction errors can be over 400 minutes. Although these values are rather high, it should be emphasized that predicting the delay in minutes is not a standalone goal of this research.

The second group of metrics consists of two metrics; the s_{avg}, which is the average estimated standard deviation, and the FSOS, which is the fraction of test flights for which the actual delay lies within one standard deviation of the predicted value. Ideally, the FSOS is as high as possible, while the s_{avg} is as low as possible. However, as stated earlier, these two metrics are a trade-off. This becomes clear when comparing the results of the Mixture Density Network and the Dropout Network in table 8. For the 1D departures, the Mixture Density Network has a high FSOS score (0.921), but also a high s_{avg} (23.203). For the same departures, the s_{avg} (5.365) of the Dropout Network is much lower, but so is the FSOS score (0.316). Similar scores are found for the 1D arrivals.

To determine which of the models of this research has the best probabilistic forecasting performance, the final and decisive metric is the CRPS, which takes the entire distribution into account. As explained in the methodology, a lower CRPS means that the prediction is closer to the perfect prediction. Despite the fact that the RF Regressor model neither has the best s_{avg}, nor the best FSOS, it does have the best CRPS score for both the 1D departures and the 1D arrivals. The second best performing model is the Mixture Density Network, whose CRPS score is only slightly greater. The overall least performing model is the Dropout Network. Since predicting probability distributions is a novelty in the field of flight delay predictions, it is difficult to put the obtained CRPS scores into a broader perspective to evaluate the accuracy of the models. Nevertheless, it can be concluded that the metric values of the RF Regressor and the Mixture Density Network are very close to each other, which is a first step in the verification of both models.

Another observation is that all models perform better for the departures than for the arrivals, which is similar to the behavior of the binary classification models. For the classification models, this was explained by the difference in the imbalance ratio of the historical flight datasets. For the probabilistic forecasting, there is no imbalance ratio, as the target is continuous. Nevertheless, the historical flight delay data distributions of the arrivals and the departures are still different. As can be seen in figure 2(b), the arrival flights more frequently arrive prior to their scheduled time, which means that the dataset contains more negative delays. Furthermore, the spread of the arrival delay histogram appears to be wider, indicating more variation in the delays to be predicted. This aligns with the conclusion that the arrival flights are more difficult to predict.

These metric scores, however, are not the only important measurements of performance. In order to improve airport operation optimization models, it is important that the delay prediction models are also able to estimate distinctive distributions for individual flights. If the flight delay model predicts the same delay distribution for every flight, the results might not be very useful for the airport operation optimization models. In that case, there might be simpler solutions for the optimization models to account for uncertainty, for example adding a time buffer to each scheduled flight. To test whether the delay prediction models are also able to predict distinctive distributions, a number of the predicted flight delay distributions are visualized in the following section.

		RF Regressor		**Mixture Network**		**Dropout Network**	
Classifier	**Metric**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**
1D Departures	RMSE	24.634	9.0×10^{-1}	25.436	9.7×10^{-1}	25.998	9.4×10^{-1}
	MSE	607.643	4.5×10^{1}	647.944	5.0×10^{1}	676.749	4.9×10^{1}
	MAE	12.556	2.7×10^{-1}	12.868	2.2×10^{-1}	12.645	2.9×10^{-1}
	max. err.	428.071	5.5×10^{1}	431.517	6.0×10^{1}	433.595	6.1×10^{1}
	s_{avg}	16.438	1.2×10^{-1}	23.203	5.6×10^{-1}	5.365	4.9×10^{-1}
	FSOS	0.837	4.8×10^{-3}	0.921	5.3×10^{-3}	0.316	3.2×10^{-2}
	CRPS	**8.643**	$\mathbf{2.5 \times 10^{-1}}$	**9.059**	$\mathbf{2.5 \times 10^{-1}}$	**10.448**	$\mathbf{3.5 \times 10^{-1}}$
1D Arrivals	RMSE	26.141	1.2×10^{0}	26.982	1.2×10^{0}	27.967	1.1×10^{0}
	MSE	684.674	6.0×10^{1}	729.547	6.6×10^{1}	783.335	6.0×10^{1}
	MAE	15.188	2.1×10^{-1}	15.632	2.4×10^{-1}	16.625	2.0×10^{-1}
	max. err.	411.908	5.8×10^{1}	414.152	5.6×10^{1}	411.650	5.3×10^{1}
	s_{avg}	19.356	2.4×10^{-1}	25.133	4.6×10^{-1}	3.253	3.8×10^{-1}
	FOS	0.782	5.2×10^{-3}	0.870	6.6×10^{-3}	0.141	1.7×10^{-2}
	CRPS	**10.702**	$\mathbf{2.5 \times 10^{-1}}$	**11.223**	$\mathbf{3.4 \times 10^{-1}}$	**15.031**	$\mathbf{2.3 \times 10^{-1}}$

Table 8: Results probabilistic forecasting

Visualization of the predicted flight delay distributions

To visualize the differences and similarities between the distribution predictions made by each of the models, four arriving and four departing flights are selected and listed in tables 9 and 10 respectively. The flights are selected in such a way that they cover a mixture of airlines, origins, aircraft, and scheduled times. The models, all trained on the same 80% of the available data, each predict the delay distribution of the selected flights. The resulting probability functions and histograms are presented in figure 5. For each flight, the predicted distribution, the associated expected delay, and the actual value are plotted. The y-axis represents the probability, while the x-axis represents the minutes of delay with respect to the Scheduled Time of Arrival (STA) or the Scheduled Time of Departure (STD), which are located at the zero minute time mark.

The most important visible result is that all models are able to predict distinctive distributions for each of the flights. Both the shape of the predicted distribution and the associated expected delay vary throughout the tested flights. A more narrow distribution can be interpreted as a more certain prediction, a wider curve indicates more uncertainty. The previously discussed differences in s_{avg} and FSOS between the Dropout Network and Mixture Density Network are clearly visible for these eight example flights, especially when comparing the results of the 1D arrivals. The distributions predicted by the Dropout Network are more narrow (i.e. a lower s_{avg}), but as a result the actual values are more often outside one standard deviation of the expected value (i.e. a lower FSOS). The opposite is true for the Mixture Density Network.

Figure 5 also shows that the Mixture Density Network and the RF Regressor do not only produce very similar metric scores, they also produce very similar delay distributions for individual flights. Although the RF Regressor predicts a probability histogram, as visualized in 5(a)-5(b), while the Mixture Density Network predicts a probability function, visualized in 5(c)-5(d), the similarity in their general shape is clearly visible. Considering that these results are obtained with two very different models, one with a tree-based framework and one with a neural network base, the similarity in scores and distribution shapes is interpreted as a verification of both models. The metric scores and the predicted distributions of the Dropout Network are less similar to the results of the other two models. However, it should be noted that finding the perfect hyperparameter settings is outside the scope of this research. Perhaps further optimizing the settings could decrease the difference with the other two models, and the difference between its 1D departures and 1D arrivals predictions.

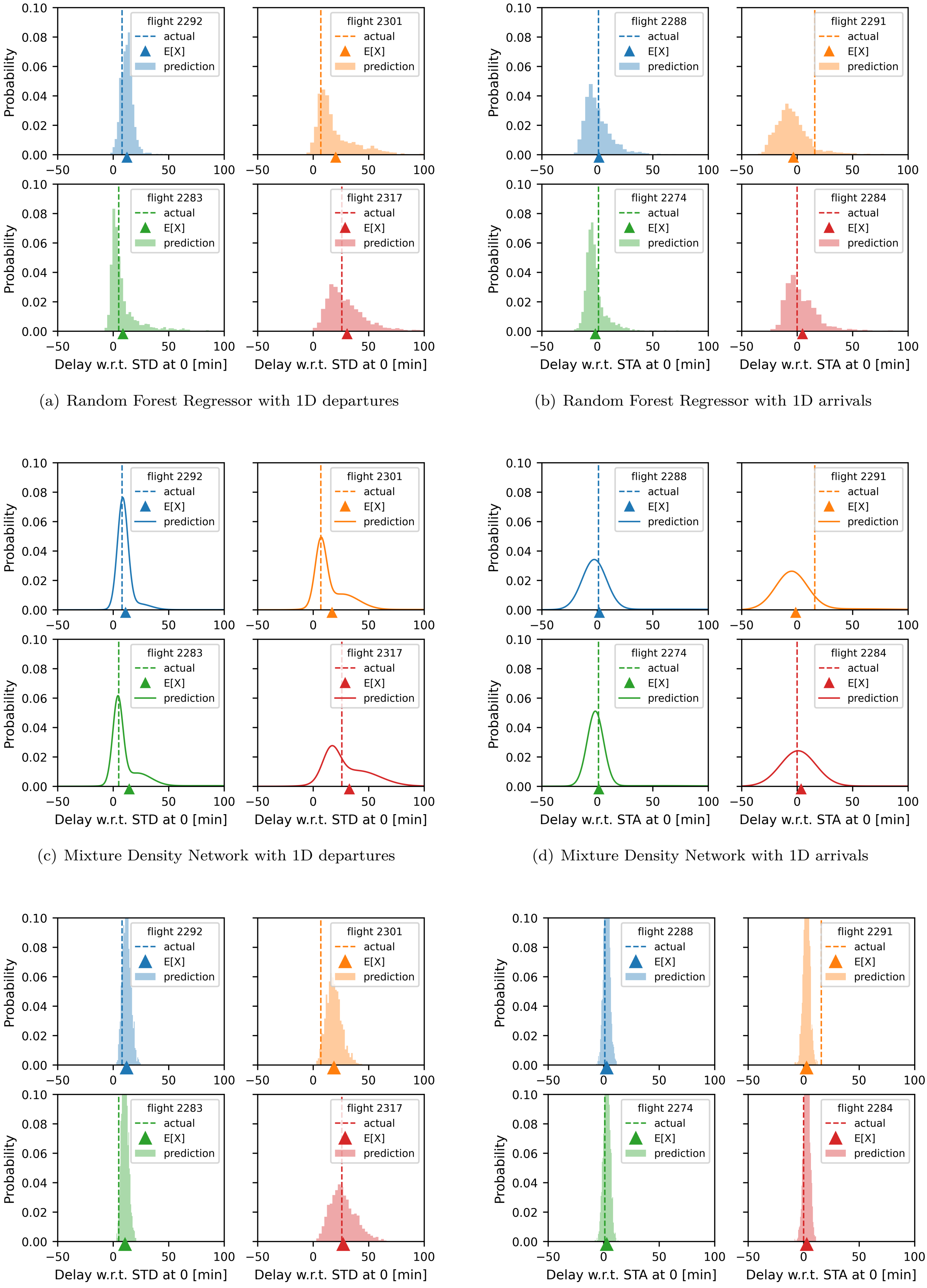
The overall result of the probabilistic flight delay forecasting, is that it shows that it is possible to accurately predict flight delay distributions with machine learning algorithms. The RF Regressor outperforms the other two models, and its average standard deviation of less than 20 minutes appears to be sufficiently small for the usage in airport operation optimization problems. The remainder of this paper investigates whether this is true, and whether the predictions can actually improve the airport operation optimization model.

Legend	STA	Flight number	Airline	Origin	Destination	Aircraft
2288	2019-05-05 11:15:00	HV5068	Transavia	GRO	RTM	B738
2291	2019-05-05 22:35:00	HV5008	Transavia	DBV	RTM	B737
2274	2019-05-02 15:05:00	BA4455	British Airways	LCY	RTM	E190
2284	2019-05-04 13:00:00	PC1261	Pegasus Airlines	SAW	RTM	A20N

Table 9: Example arrival flights

Legend	STD	Flight number	Airline	Origin	Destination	Aircraft
2292	2019-05-03 06:55:00	HV6061	Transavia	RTM	BCN	B737
2301	2019-05-03 18:40:00	HV5293	Transavia	RTM	VIE	B737
2283	2019-05-01 10:50:00	BA4454	British Airways	RTM	LCY	E190
2317	2019-05-06 14:35:00	PC1262	Pegasus Airlines	RTM	SAW	A20N

Table 10: Example departure flights



(a) Random Forest Regressor with 1D departures

(b) Random Forest Regressor with 1D arrivals

(c) Mixture Density Network with 1D departures

(d) Mixture Density Network with 1D arrivals

(e) Dropout Network with 1D departures

(f) Dropout Network with 1D arrivals (with EIN)

Figure 5: Impression of the results of the probabilistic forecasts

6 Description of the Case Studies

The final part of this research is dedicated to the incorporation of the predicted flight delay distributions into an airport operation optimization model. The selected airport operation is the Flight-to-Gate Assignment Problem (FGAP), a challenge faced by both regional and international airports.

6.1 Additional methodology Flight-to-Gate Assignment Problem

Since the goal of this paper is to show the effect of incorporating flight delay predictions, rather than creating the optimal FGAP model, the additional methodology is assembled accordingly. It starts with a description of the standard FGAP model, which will function as a benchmark. The adjustments needed to convert this model into a probabilistic FGAP model, which includes the delay predictions, follow afterwards.

Standard FGAP model

The usual approach within the aviation industry is to model the Flight-to-Gate Assignment Problem (FGAP) as a Linear Problem (LP). This research follows the definition of van Schaijk and Visser (2017) with modifications of L'Ortye (2019). It starts with the following set of definitions:

$$\begin{aligned}
N &: \text{the set of flights to be scheduled}\\
M &: \text{the set of gates available at the airport}\\
K &: \text{the set of time steps}\\
n &: \text{the total number of flights to be scheduled}\\
m &: \text{the total number of available gates at the airport}\\
k &: \text{the total number of time steps}\\
c_{i,j} &: \text{the cost of assigning flight } i \text{ to gate } j
\end{aligned}$$

It is important to emphasize that a flight in this context is defined from the airport's perspective; it is a certain aircraft that is present at the airport for a certain amount of time. This research aims to schedule single days, each day divided into time steps of five minutes. This results in a total number of time steps k of 288. Furthermore, it assumes that all gates at RTM airport are available, which means that m is set to 8. It also assumes that there is no preference between the gates, therefore all $c_{i,j}$ have a value of 1. The number of flights n depends on the day to be scheduled. Additionally to the set above, a variable is introduced that indicates for each flight i whether it has a positive presence probability at time step i:

$$s_{i,t} = \begin{cases} 1, & \text{if flight } i \text{ has a non-zero probability to be} \\ & \text{present at time step } t, \\ 0, & \text{otherwise.} \end{cases} \qquad \text{for } i \in N, \text{and } t \in K \tag{26}$$

In this standard, deterministic version of the FGAP model, the presence probability is either 1 when a flight is scheduled to be present at the airport, or 0 when it is not. The binary decision variables of this Linear Program are defined as follows:

$$x_{i,j,t} = \begin{cases} 1, & \text{if flight } i \text{ is assigned to gate } j \text{ at time step } t, \\ 0, & \text{otherwise.} \end{cases} \qquad \text{for } i \in N, j \in M, \text{and } t \in K \tag{27}$$

The objective of the Linear Program is to minimize the costs of the assignment, summed over all flights, time steps and gates:

$$\min \quad \mathcal{Z} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{t=1}^{k} c_{i,j} x_{i,j,t} \tag{28a}$$

s.t.

$$\sum_{j=1}^{m} s_{i,t} x_{i,j,t} = s_{i,t}, \qquad \forall i \in N,\ \forall t \in K, \tag{28b}$$

$$\sum_{i=1}^{n} s_{i,t} x_{i,j,t} \leq 1, \qquad \forall j \in M,\ \forall t \in K, \tag{28c}$$

$$s_{i,t} x_{i,j,t+1} - s_{i,t+1} x_{i,j,t} = 0, \qquad \forall i \in N,\ \forall t \in K \tag{28d}$$

The standard Linear Program has three constraints. The first, stated in 28b, is to ensure that each flight is assigned to a gate for all time steps where its presence probability is positive. The purpose of the second constraint is to ensure that only one flight gets assigned to a certain gate in a certain time step. The final constraint ensures that once a certain flight is assigned to a certain gate, it remains assigned to the same gate for all following time steps where the flight is scheduled to be present.

Probabilistic FGAP model

A major assumption in the standard FGAP is that the presence probability of each flight is known in advance and fixed; the possibility of delays is not taken into account. As a solution, the research of van Schaijk and Visser (2017) proposes to replace constraint 28c with a new constraint that considers the presence of a certain flight as a probability rather than a binary:

$$\sum_{i=1}^{n} f(p_{i,t}, r) p_{i,t} x_{i,j,t} \leq 1, \qquad \forall j \in M,\ \forall t \in K \tag{29}$$

Instead of allowing only a single flight to be scheduled at a certain gate in a certain time step, this constraint allows multiple flights, under the condition that the probability of multiple flights being present simultaneously is below a certain level. This level of maximum allowed overlap probability is set by input parameter r, which can be decomposed as follows:

$$r = p_{i,t} \cdot p_{max} \tag{30}$$

where, $p_{i,t}$ is the presence probability of flight i at time t, which is discussed in detail in the following section. The other variable, p_{max}, is the maximum allowable presence probability for a second flight. To incorporate this into the FGAP constraint, where all terms are summed instead of multiplied, the presence probabilities have to be scaled first. The scaling function $f(p_{i,t}, r)$ needs to fulfill the following condition:

$$f(p_{i,t}, r) \cdot p_{i,t} + f(p_{i,t}, r) p_{max} = 1 \tag{31}$$

which, in combination with rewriting equation 30 for p_{max}, leads to:

$$f(p_{i,t}, r) = \frac{p_{i,t}}{r + p_{i,t}^2} \tag{32}$$

Presence probability curve

The probabilistic FGAP model uses a constraint that is based on presence probability $p_{i,t}$; the probability that flight i is present at the apron at time t. To acquire this probability for each time step, a presence probability curve is constructed for each flight. In the work of of van Schaijk and Visser (2017), this presence probability is a relatively simple estimation, based on two parameters of the historical flights. This research proposes a new method for the construction of the probability presence function, which is based on the previously estimated flight delay distributions. As stated earlier, a flight in the FGAP model is a certain aircraft that is present at the airport for a certain amount of time. Each flight is characterized by two events; its arrival at the airport and its departure from the airport. For the probabilistic FGAP model it is assumed that the two events are independent. To determine the probability that flight i is present, the following two random variables are introduced:

X_{arr} : a random variable that indicates the time step at which flight i arrives at the airport

X_{dep} : a random variable that indicates the time step at which flight i departs from the airport

The Cumulative Distribution Functions (CDF) that represent whether a flight arrives (for X_{arr}) or departs (for X_{dep}) before time step t are defined as follows:

$$F_{X_{arr}}(t) = P(X_{arr} \leq t) \tag{33}$$

$$F_{X_{dep}}(t) = P(X_{dep} \leq t) \tag{34}$$

The corresponding Probability Density Functions (PDF) are:

$$f_{X_{arr}}(t) = \frac{d(F_{X_{arr}} \leq t)}{dt} \tag{35}$$

$$f_{X_{dep}}(t) = \frac{d(F_{X_{dep}} \leq t)}{dt} \tag{36}$$

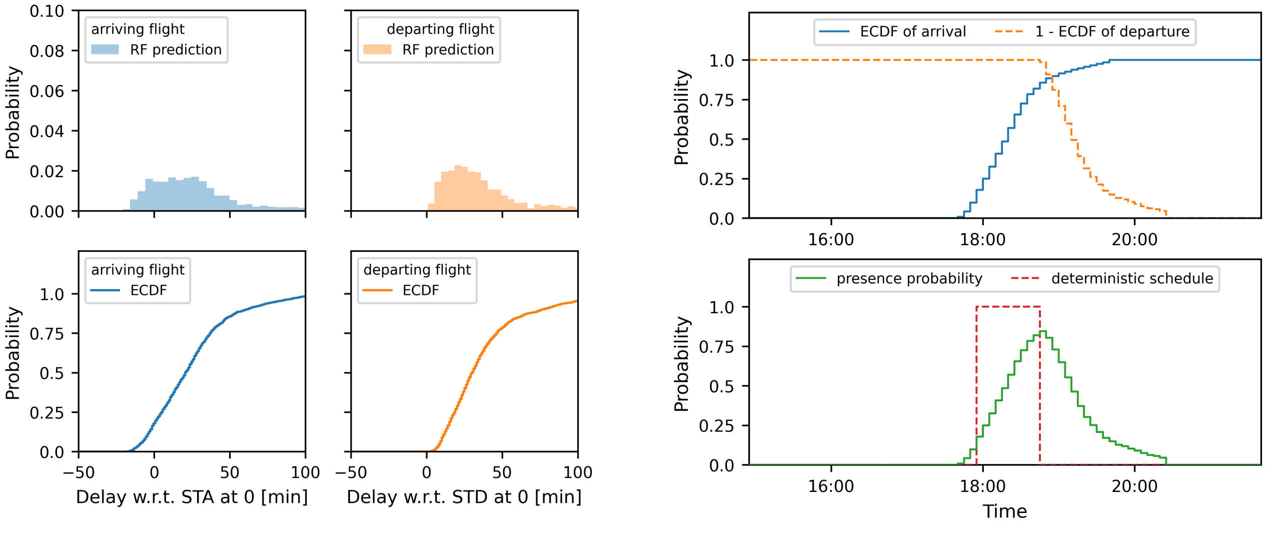
As shown in section 5, the probability density functions can be approximated with machine learning algorithms. By taking the cumulative of these approximations, the cumulative density functions can be approximated. An example is given in figure 6(a). Here, the upper two subfigures show the delay distributions of an arrival flight and its subsequent departing flight. Both are predicted by the modified Random Forest (RF) Regressor, which was previously identified as the best performing model. The delay predictions are estimated against the scheduled arrival or departure times, which are centered at zero. The lower two subfigures show their corresponding Empirical Cumulative Distribution Functions (ECDF).

To be in the state 'present at the airport', a flight has to fulfill two requirements: 1) the aircraft considered has arrived at the airport and 2) it has not yet departed. Since it is assumed for this model that these events are independent, the requirements lead to the following presence probability function:

$$g_{pres} = F_{X_{arr}} \cdot (1 - F_{X_{dep}}) \tag{37}$$

An example is given in figure 6(b). In the upper subfigure, the ECDF of the arrival and departure are placed on the same timeline, with their zero minute delay points aligned at their respective scheduled times of arrival or departure. Since the presence probability function is based on the probability that the flight is not yet departed, the ECDF of the departure is converted to one minus the ECDF, an approximation of $1 - F_{X_{dep}}$. In the example in figure 6, the aircraft of the illustrated flight arrives and departs on the same day. When a flight stays at the airport overnight, the flight either misses an arrival or departure delay prediction, since this research schedules for individual days only. If this is the case, the absent CDF approximations are replaced with a unit step input. For a flight that departs after staying overnight, this step input for the approximation of $F_{X_{arr}}$ is placed at an hour before the scheduled time of departure. For a flight that arrives at the airport and stays overnight, $F_{X_{dep}}$ is approximated with a step input at an hour after the scheduled arrival time.

The lower subfigure of figure 6(b) shows the approximation of the complete presence probability function as defined in equation 37. The time between the scheduled time of arrival and scheduled time of departure, as used in the deterministic FGAP schedule, is also indicated. For this particular example, it can be seen that the presence probability curve is wider, and centered around a later time, than its deterministic schedule counterpart. By constructing a presence probability curve for each flight in the schedule, the corresponding $p_{i,t}$ can be determined for every time step.



(a) Approximated arrival and departure PDF and CDF

(b) Presence probability function

Figure 6: Constructing the presence probability function

Re-assignment process

A limitation of the probabilistic FGAP is that the scaling function in equation 32 is based on a maximum of two flights overlapping. However, both in theory and in practice this number could be greater. When this is the case, the scaling function decreases the presence probabilities $p_{i,t}$ in probabilistic constraint 29 too much. As as result, the constraint may accept a greater overlap probability than the maximum specified by input parameter r. To counter any overlap probability violations, this research uses an iterative process, similar to the one presented by L'Ortye (2019). The general idea consists of the following steps:

Step 1: Make a flight-to-gate assignment with the probabilistic FGAP model

Step 2: Check if there are any overlap probability violations

(a) If there are violations: add a constraint to the FGAP model to reduce the number of flights allowed to be scheduled at a gate at the same time and repeat the previous steps

(b) If there are no violations: the schedule is approved

To formalize step 2, the following sets of flights are introduced:

$F_{j,t}$: the set of flights i assigned to gate j at time t

$C_{j,t}$: the set of all possible combinations with at least two flights i from $F_{j,t}$

To give an example, if flight 1, 2, and 3 are all assigned to gate j at time step t, $F_{j,t}$ becomes $\{1, 2, 3\}$ and $D_{j,t}$ becomes $\{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$. To calculate the true overlap probability, two complementary probabilities are taken into account; the probability that a flight is present $p_{i,t}$ and the probability that it is not present, denoted by $1 - p_{i,t}$. The flights listed in a combination in $C_{j,t}$ are assumed to be present, the missing flights are assumed to be not present. For each combination in $C_{j,t}$, the presence probability is calculated by multiplying the corresponding probabilities. By adding the results of all combinations, the true overlap probability is calculated for each gate j and time step t. This can be written more formally as:

$$r_{true_{j,t}} = \sum_{c\, \in C_{j,t}} \prod_{i\, \in F_{j,t}} p_{i,t} \mathbb{1}_{i \in c} + (1 - p_{i,t}) \mathbb{1}_{i \notin C}, \quad \forall j \in M,\ \forall t \in K \tag{38}$$

where the indicator $\mathbb{1}$ is one if the sub-scripted statement is true and zero otherwise. A violation takes place if any of the true overlap probabilities is larger than input overlap probability r. As stated in step 2(a), the solution to overlap probability violations is to reduce the maximum number of flights allowed to be assigned to a gate at a certain time. In order to do this, the current maximum number of flights assigned at a gate is calculated first:

$$\beta = \max\{ \sum_{i=1}^{n} x_{i,j,t}, \quad \forall j \in M,\ \forall t \in K\ \} \tag{39}$$

To reduce this maximum, this research proposes the following additional constraint:

$$\sum_{i=1}^{n} x_{i,j,t} \leq \beta - 1, \qquad\qquad \forall j \in M,\ \forall t \in K \tag{40}$$

This constraint ensures that the maximum number of flights in the following iteration of solving the FGAP model, is at least one less than the current maximum. All steps of solving the FGAP, checking for violations and reducing the maximum number of flights, are repeated until the schedule no longer contains any violations.

FGAP performance metrics

To measure the performance of the probabilistic FGAP model, which includes the predicted flight delay distributions, the resulting schedules are compared to the schedules obtained with the standard FGAP model. A good flight-to-gate schedule is robust enough to withstand disruptions without too much re-scheduling. However, a good schedule should also maximize the available capacity. Assigning only one flight per gate per day might be very robust, but it is not efficient. To quantify this trade-off between robustness and efficiency, two metrics are introduced.

Efficiency - average number of scheduled occupied time slots

To express the efficiency of a flight-to-gate schedule, the average number of time slots that are scheduled to be occupied is calculated. A time slot is defined as a certain time step at a certain gate. As can be seen in figure 6(b), the presence probability curve is often much wider than the difference between the scheduled arrival and departure time. To allow for a better comparison between the standard and the probabilistic FGAP model, the lower presence probabilities are not taken into account when determining the scheduled occupation. Specifically, only the time steps where a flight is more likely to be present ($p_{i,t} \geq 0.5$) than not, are counted towards the efficiency metric. This concept is illustrated in figure 7(a). To express the metric more formally, the following set is defined:

$G_{j,t}$: the number of flights scheduled at gate j at time t with presence probability $p_{i,t} \geq 0.5$

A time slot is considered occupied if it at least one of the flights assigned to it has a presence probability of at least 0.5. This is captured with the following indicator:

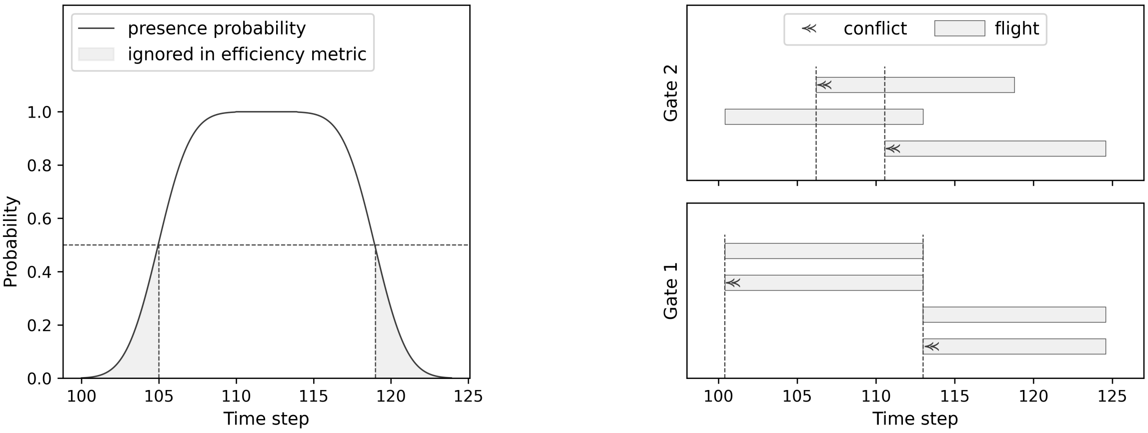
$$v_{j,t} = \begin{cases} 1, & \text{if } G_{j,t} \geq 1 \\ 0, & \text{otherwise.} \end{cases} \qquad \text{for } j \in M, \text{and } t \in N \tag{41}$$

To calculate the efficiency score η for an entire flight-to-gate schedule, $v_{j,t}$ is summed over all time slots, i.e. all time steps and gates:

$$\eta = \sum_{j=1}^{m} \sum_{t=1}^{k} v_{j,t} \tag{42}$$

Finally, the efficiency performances of the standard FGAP and probabilistic FGAP model are compared by taking the average of the efficiency metric over multiple test days:

$$\bar{\eta} = \frac{\sum \eta_d}{\|D\|}, \quad \text{for each test day } d \in D \tag{43}$$



(a) Efficiency metric

(b) Robustness metric

Figure 7: FGAP metrics support

Robustness - average number of conflicts
In this research, the robustness of a flight-to-gate assignment is measured by the number of conflicts that arise when executing the schedule. In this context, a conflict is defined as a situation where upon the arrival of a flight at the airport its assigned gate is already occupied. Since this situation would always require some form of re-scheduling, regardless of how long the overlap at the gate is, the metric measures the frequency of occurrence rather the duration of the overlap. Different than the efficiency metric, which is based on the assigned gate and the *scheduled* times of arrival and departure, this metric is based on the assigned gate and the *actual* time of arrival and departure of all flights in the schedule.

To illustrate which conflict scenarios the metric should capture, two examples are given in figure 7(b). The upper subfigure shows that a conflict should be defined per time step. For a newly arriving aircraft it does not matter how many other flights are already present at the gate, it should count as one new conflict. Following the same reasoning, the lower subfigure shows that two flights arriving at the gate at the exact same time should also only count as one conflict. Furthermore, the lower subfigure shows that simply counting the flights per time step, or calculating the differences between time steps, does not capture all conflicts. Even though the number of flights at gate 1 remains exactly the same between time steps 110 and 115, there is a new conflict. Capturing all these different conflict scenarios with one robustness metric, starts with defining the following flight set:

$$H_{j,t} : \text{the set of flights } i \text{ present at gate } j \text{ at time } t$$

Since a conflict only arises when the number of flights at the gate is at least two, the first requirement of a conflict is $\|H_{j,t}\| \geq 2$. The second requirement is based on the interpretation that a new conflict can only arise when a new flight arrives at the gate. Therefore the set of flights present at the time of the conflict cannot be a subset of the flights present at the previous time step, i.e. $H_{j,t} \not\subseteq H_{j,t-1}$. Combining these two requirement leads to the following conflict indicating variable:

$$w_{j,t} = \begin{cases} 1, & \text{if } \|H_{j,t}\| \geq 2 \text{ and } H_{j,t} \not\subseteq H_{j,t-1} \\ 0, & \text{otherwise.} \end{cases} \qquad \text{for } j \in M, \text{and } t \in N \tag{44}$$

To calculate the robustness score ρ for an entire flight-to-gate schedule, $w_{j,t}$ is summed over all time slots, i.e. all time steps and gates:

$$\rho = \sum_{j=1}^{m} \sum_{t=1}^{k} w_{j,t} \tag{45}$$

To compare how well the standard and probabilistic FGAP models perform in terms of robustness, the average of the robustness metric over multiple test days is calculated:

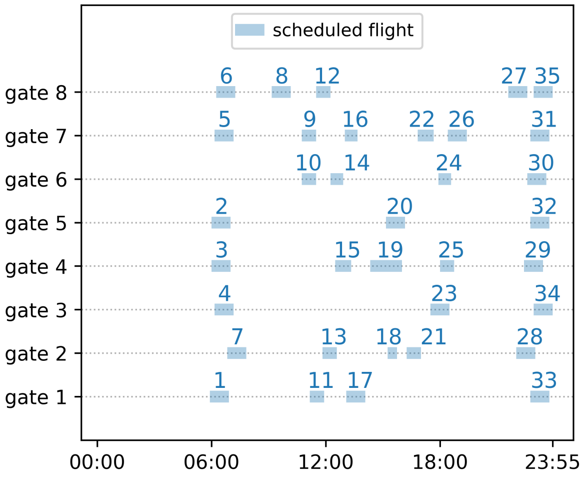
$$\bar{\rho} = \frac{\sum \rho_d}{\|D\|}, \quad \text{for each test day } d \in D \tag{46}$$

6.2 Case study results

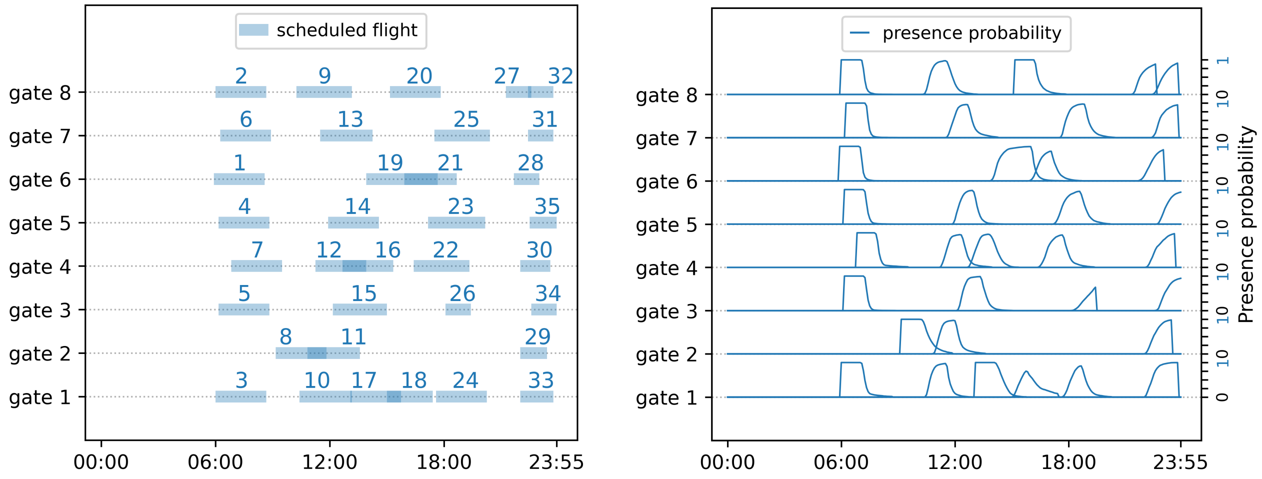
The final goal of this research is to quantify the effect of incorporating the predicted flight delay distributions into the FGAP model. The underlying hypothesis is that this incorporation leads to an improvement in terms of the robustness and/or the efficiency of the resulting flight-to-gate schedules. In order to test this, both the standard FGAP model and the probabilistic FGAP model are applied to the same test days. Their performances are measured and compared by calculating the efficiency and robustness of the resulting flight-to-gate schedules. The case study is based on the same RTM data as previously described. The selected test days are the days between the 1^{st} of July and the 31^{st} of August in 2019. Both models are trained on all available data prior to the test date.

Resulting flight-to-gate schedules

In principle, a flight-to-gate schedule is a list of which flights are assigned to which gates at which time steps. Figure 8 gives an example of the flight-to-gate schedules obtained with both FGAP models. The example is based on the flights at RTM on the 14^{th} of July, 2019, and an overlap probability r of 0.1. Subfigures 8(a) and 8(b) present the resulting schedules, each showing the time of presence (with $p_{i,t} > 0$) and the assigned gate of all flights. The probabilistic schedule in 8(b) is visibly fuller than the deterministic schedule in 8(a), despite the fact that a certain overlap between flights is allowed here. To verify that the maximum overlap probability is indeed not exceeded, the presence probabilities of the fights, assigned according to the probabilistic schedule, are given in figure 8(c).



(a) Schedule obtained with the standard FGAP model



(b) Schedule obtained with the probabilistic FGAP model

(c) Presence probabilities

Figure 8: Example of flight-to-gate schedules obtained with the standard and probabilistic ($r = 0.1$) FGAP model. More information about the flights in this example can be found in appendix D of the thesis.

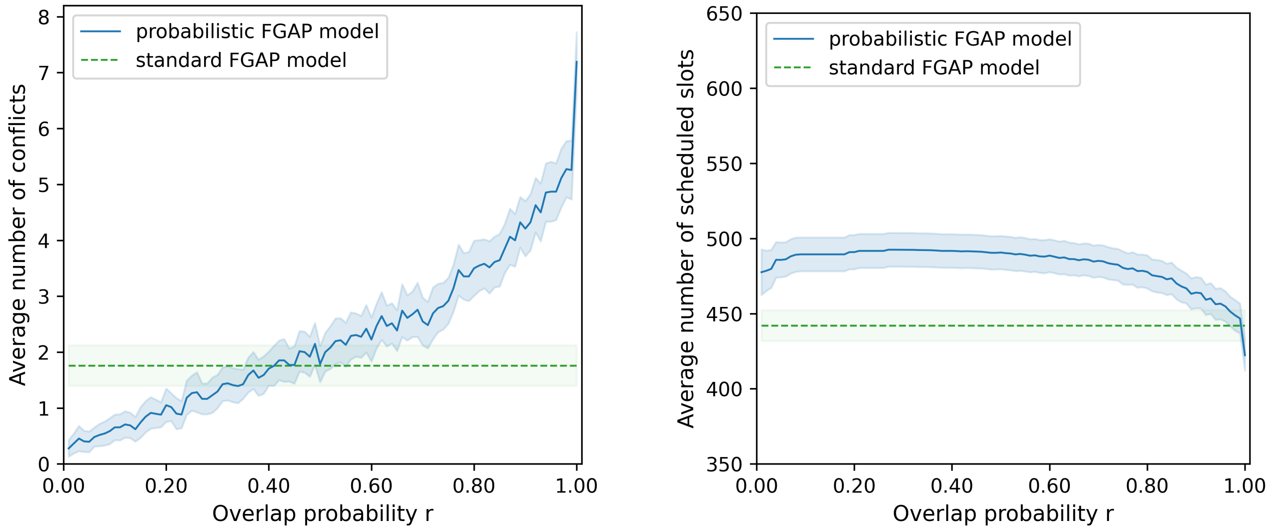
Evaluation of gate schedules

To measure the performance of both FGAP models, the models are applied to the same set of test days. An important factor for the results of the probabilistic FGAP model, is input parameter r, which sets the maximum allowed overlap probability. Since its value strongly depends on the preference of the user, all values for r in the range $\{0.01, 0.02, ...1\}$ are tested for every day in the test set. For the standard, deterministic FGAP model, r is irrelevant, which means that the same schedule is obtained for every r. The resulting robustness and efficiency metric are averaged over all test days for each value of r. The variation in r and the prediction for multiple test dates are simultaneously interpreted as a sensitivity analysis for the probabilistic FGAP model.

In figure 9(a), the average number of conflicts, accompanied with an indication of the standard deviation within the test group, is given per r for both FGAP models. As expected, r does not influence the standard FGAP model, which explains the constant number of conflicts. For the lower values of r, the probabilistic schedule outperforms the standard schedule in terms of conflicts, i.e. in terms of robustness. The reverse is true for the higher values of r. This makes intuitive sense; if the schedule allows for a higher overlap probability, more actual overlapping conflicts can be expected. For this particular combination of test days and airport, the probabilistic FGAP model outperforms the standard model for values of r between 0.01 and 0.4. It appears that the lower the overlap r, the lower the average number of conflicts. It should be noted, however, that there is a minimum value of r required to find a feasible solution with the probabilistic FGAP model. This minimum value depends on the test day; days with many flights and/or wide presence probability distributions might require a larger minimum r than other days.

The improvement in robustness does come at the cost of a reduction in efficiency, as can be seen in figure 9(b). Here, the average number of scheduled slots is presented as a function of overlap probability r. For the entire range of r where the probabilistic FGAP model outperforms the standard model in terms of conflicts, which is between 0.01 and 0.4, the number of scheduled occupied slots in the probabilistic schedule is greater than in the deterministic equivalent. Only for overlap probabilities close to 1, both models have the same average number of scheduled to be occupied slots. The small dip in the average number of scheduled occupied slots for values of r between 0.01 and 0.05, initially seems contradicting, as less overlap would typically lead to assigning more time slots. The reduction is explained by the fact that for these low values of r the fuller test days become infeasible, which eliminates them from the test sample. Effectively, this reduces the average number of scheduled occupied slots for the lowest values of r.

Even when ignoring the lowest values of r, as they are not suitable for every test case and their average number of scheduled slots is likely underestimated, there is still a large range of r values for which the probabilistic FGAP model outperforms the standard model. For the values of r between 0.05 and 0.4, the average number of conflicts is reduced by potentially more than 50%, at the cost of an increase of roughly 11% in the average number of scheduled occupied slots. Ultimately, the level of acceptable overlap, and the trade-off between robustness and efficiency, would be determined by the end user of the models at the airport. For this research, the main result of this case study is that it is possible to improve the robustness of flight-to-gate schedules, albeit at the cost of efficiency, by incorporating flight delay distribution predictions into a standard FGAP model.



(a) Robustness metric

(b) Efficiency metric

Figure 9: Evaluating the performance of the FGAP models

7 Conclusions

The aim of this research paper is to investigate whether it is possible and potentially beneficial to accurately predict the probability distribution of flight delays with machine learning algorithms at a regional airport. This leads to the following insights.

First of all, it is found that the machine learning classifiers that are often used for the prediction of flight delays at large airports, also perform satisfactory at a regional airport. The best performing model is the LightGBM model, which achieves an accuracy for departure delays of around 0.7, with a recall and precision above 0.5. The accuracy is greater for the arrival delays, but their recall and precision are below 0.5, which is a result of the high imbalance ratio of the arrival data. Altogether, it is concluded that the model performs as expected, and that the fact that this airport is regional is not necessarily a limiting factor for accurate delay classification. Another conclusion is that there is not much difference between a prediction horizon of one month and a prediction horizon of a day. This corresponds with the finding that the most important features are related to the flight schedule, which is usually already made six months in advance, and is therefore available for both prediction horizons. Although this similarity in performance suggests that there might be even better (weather related) features available for the 1D horizon, which was expected to outperform the 1M horizon features, the benefit of this similarity is that the models and results presented in this paper are representative for different strategical planning phases.

The second part of this research has shown that it is possible to accurately predict distinctive delay distributions for individual flights. The modified Random Forest Regressor gives the best results, followed closely by the Mixture Density Network. Since predicting the probability density function of flight delays is rather new, it is difficult to put their performance into a broader perspective. However, the fact that these two models produce very similar results in terms of both the overall performance metrics as well as the shape of the individual flight delay distributions, even though both models are very different, is interpreted as a verification of the results.

In the final part of this paper, the resulting delay distributions are incorporated into an existing Flight-to-Gate Assignment Problem to investigate whether the predicted delay distributions are able to improve an airport operation optimization problem. In this particular case study, incorporating the flight delay distributions into the FGAP model can lead to a reduction in the average number of conflicts of more than 50%, at the expense of increasing the average number of scheduled slots by roughly 11%. Differently stated, the case study shows that the resulting gate schedules are more robust but less efficient. Although it is acknowledged that these results are case specific, this paper shows that it is possible to improve an airport operation optimization problem by implementing predicted delay distributions.

The overall conclusion of this paper is that accurately predicting flight delay distributions is possible and has the potential to improve at least one airport operation optimization model by constructing one integrated model. Since the results of the binary classification of delays were reasonably similar for regional and international airports, it can be expected that the same holds for the probability distribution predictions. Not only can this research be extended to larger airports, integrating predicted delay probability distributions might be beneficial for many more logistical applications, both within and beyond the aviation industry.

Recommendations

For further research, this paper proposes the following steps. First of all, the aim of this paper is to show the possibility and potential of combining a probabilistic flight delay model and an airport operations optimization. Although this includes the selection of appropriate features, models, and hyperparameter settings, finding the optimal features, models and settings falls outside the scope of this research. This paper does not dismiss the possibility of even better models, features and hyperparameter settings. In fact, the small difference between the results of the 1D and 1M feature sets already suggested that there might exist better weather features. This means that it is expected that the results of this paper can be further improved, which is the first recommendation when the resulting probabilistic FGAP model would be applied in practice.

The second recommendation is to extend this research to large, international airports. Not only are they expected to have a larger database, which might lead to even more accurate delay predictions, they also have a different flight schedule. Compared to a regional airport, international airports generally have longer turnaround times and more flights per day. For the regional airport in this paper, the predicted presence probability functions were almost always wider than the length of the turnaround time in the schedule, which made the probabilistic FGAP schedules almost automatically less efficient in terms of the number of scheduled to be occupied slots. Perhaps this is different for schedules with longer turnaround times. Not only is it expected that international airports have different robustness and efficiency scores, as these scores are always strongly dependent on the case study, their fuller schedules and longer turnaround times might even enable an improvement in efficiency.

A third recommendation is to investigate the effect of incorporating the probabilistic flight delay models into other airport, or airline, operation optimization models. As stated in the introduction, many airport and airline operations, such as airport surface traffic optimization, aircraft routing optimization, and airline crew scheduling, involve the arrival and departure of flights. It is expected that at least some of them could also be improved by integrating predicted flight delay distributions. As a matter of fact, the methodology of integrating predicted delay distributions into an operational optimization model might also be applicable to other logistics fields, such as the delivery of parcels or other modes of transportation. Therefore, the final recommendation is to explore opportunities outside the aviation industry.

References

Bishop, C. (1994). Mixture density networks. Working paper, Aston University.

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

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chen, J. and Li, M. (2019). Chained predictions of flight delay using machine learning. *AIAA Scitech 2019 Forum*, page 1661.

Choi, S., Kim, Y., Briceno, S., and Mavris, D. (2016). Prediction of weather-induced airline delays based on machine learning algorithms. *IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6.

de Neufville, R. and Odoni, A. (2013). *Airport Systems.* McGraw-Hill Education, New York City, New York.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *Proceedings of The 33rd International Conference on Machine Learning*, 48:1050–1059.

Granitto, P. M., Furlanello, C., Biasioli, F., and Gasperi, F. (2006). Recursive feature elimination with random forest for ptr-ms analysis of agroindustrial products. *Chemometrics and Intelligent Laboratory Systems*, 83(2):83–90.

Gui, G., Liu, F., Sun, J., Yang, J., Zhou, Z., and Zhao, D. (2020). Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on vehicular technology*, 69(1):140–150.

Hersbach, H. (2000). Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting*, 15:559–570.

Horiguchi, Y., baba, Y., Kashima, H., Suzuki, M., Kayahara, H., and Maeno, J. (2017). Predicting fuel consumption and flight delays for low-cost airlines. *Proceedings of the Twenty-Ninth AAAI Conference on Innovative Applications*, pages 4686–4693.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *31st Conference on Neural Information Processing Systems (NIPS)*, 1:3147–3155.

Khanmohammadi, S., Tutun, S., and Kucuk, Y. (2016). A new multilevel input layer artificial neural network for predicting flight delays at jfk airport. *Procedia Computer Science*, 95:237 244.

Kim, Y., Choi, S., Briceno, S., and Dimitri, M. (2016). A deep learning approach to flight delay prediction. *IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6.

Klein, A. (2010). Airport delay prediction using weather-impacted traffic index (WITI) model. *Digital Avionics Systems Conference (DASC), IEEE/AIAA 29th*, pages 2–B.

Lambelho, M., Mitici, M., Pickup, S., and Marsden, A. (2020). Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions. *Journal of Air Transport Management*, 82.

L'Ortye, J. (2019). Robust flight-to-gate assignment planning with airside and landside constraints. Master's thesis, Delft University of Technology.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *31st Conference on Neural Information Processing Systems (NIPS).*

McCarthy, N., Karzand, M., and Lecue, F. (2019). Amsterdam to dublin eventually delayed? lstm and transfer learning for predicting delays of low cost airlines. *The Thirty-First AAAI Conference on Innovative Applications of Artificial Intelligence (IAAI-19).*

Moon, S. and Carbonell, J. (2017). Completely heterogeneous transfer learning with attention - what and what not to transfer. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*, pages 2508–2514.

Mueller, E. and Chatterij, G. (2002). Analysis of aircraft arrival and departure delay characteristics. *IAA aircraft technology, integration and operations (ATIO) conference.*

Rebollo, J. and Balakrishnan, H. (2014). Characterization and prediction of air traffic delays. *Transportation research part C: Emerging technologies*, 44:231–241.

Shao, W., Prabowo, A., Zhao, S., Tan, S., Koniusz, P., Chan, J., Hei, X., Feest, B., and Salim, M. (2019). Flight delay prediction using airport situational awareness map. *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.*

Thiagarajan, B., Srinivasan, L., Sharma, A., Sreekanthan, D., and Vijayaraghavan, V. (2017). A machine learning approach for prediction of on-time performance of flights. *IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, pages 1–6.

van Schaijk, O. and Visser, H. (2017). Robust flight-to-gate assignment using flight presence probabilities. *Transportation Planning and Technology*, 40(8):928–945.

Vossen, J., Feron, B., and Monti, A. (2018). Probabilistic forecasting of household electrical load using artificial neural networks. *IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6.

Xu, N., Donohue, G., Laskey, K. B., and Chen, C.-H. (2005). Estimation of delay propagation in the national aviation system using bayesian networks. *6th USA/Europe Air Traffic Management Research and Development Seminar.*

Yu, B., Guo, Z., Asian, B., Wang, H., and Chen, G. (2019). Flight delay prediction for commercial air transport: A deep learning approach. *Transportation Research Part E*, 125:203–221.

Zhu, L. and Laptev, N. (2017). Deep and confident prediction for time series at uber. *2017 IEEE International Conference on Data Mining Workshops*, pages 103–110.

II

Literature Study
previously graded under AE4020

1 Introduction

The ultimate goal of this research is to incorporate an elaborate flight delay prediction model into an airport operation optimization model. Numerous researches have been performed in the field of delay predictions and operations optimization separately, but the potential benefit of integrating one into the other directly is an underexposed topic in existing literature.

This research recognizes the potential value of combining the two models; more extensive flight delay prediction with probability intervals could lead to more robust and efficient operations. Given that the global aviation industry has been growing for most of the past decade, this is valuable knowledge for both international and regional airports. In fact, more insight in flight delay predictions at regional airports is valuable knowledge on its own.

The aim of this report is to understand the current state-of-the-art in the field of flight delay predictions. In order to bridge the gap to optimization problems, it is important to first identify the usual approach and possible improvements in existing flight delay literature. Furthermore, this report aims to understand how the flight delay predictions can be incorporated in airport operations optimization problems in more detail.

The structure of this literature report is as follows. Chapter 2 describes the project in more detail, including the formulated research questions and objectives. Chapter 3, the main part of this report, presents a thorough literature review that should function as a guide for answering all research questions upon its completion. In the final chapter, chapter 4, the main conclusions of this literature study are given.

2 Project description

Motivation and background

Throughout the aviation industry there are many stakeholders involved when it comes to accurate flight delay predictions. Airports, who are optimizing the efficient use of their existing capacity to accommodate for the growing air travel demand. Airlines, who aim to minimize the propagation effect of disruptions in their schedule. And last but not least, the passengers, who prefer to know about delays as soon as possible.

As a result, numerous flight delay researches have been performed within the field of air transport and operations. Most studies aim to predict whether or not a flight will arrive or depart within 15 minutes of its scheduled time, which corresponds to the delay reporting system of the U.S. Federal Aviation Administration (FAA). Furthermore, most researches are centered around large, international airports. A number of different methods have been explored; statistical methods, machine learning methods such as boosting and bagging decision trees, and most recently neural networks with overall rather positive results. Accuracy's of above 80%, such as in (Rebollo and Balakrishnan 2014), are not uncommon.

This research focuses on extending the current state-of-the-art in flight delay predictions in two ways. Firstly, it is targeted at Rotterdam the Hague Airport, which is a regional airport; an underexposed topic in existing literature. Secondly, it aims to expand flight delay predictions with a probability density function, a novelty in the field that is potentially beneficial for airport operation optimizations. The combination of these two identified improvement points lead to the ultimate goal of this thesis: incorporating a probabilistic flight delay prediction model into an existing gate assignment optimizer for a regional airport.

Research questions

The main research question of this research is:

MQ: Is it possible to accurately predict the probability distribution of delays of individual flights at a regional airport with machine learning algorithms?

To answer the main question, three sub-questions are defined. The first sub-question aims to answer how well the flight delay prediction models from existing literature, previously mostly applied to international airports, perform at a regional airport. Following the majority of the previous studies, the flight delay problem is initially considered as a binary classification problem, where a flight is either delayed or not. The underlying assumption is that if the performance of the binary classifiers are insufficient, models with more complicated prediction targets, such as the minutes of delays, will not perform better.

Q1: How well can a binary classifier predict flight delays at a regional airport in terms of accuracy, precision and recall?

(a) *How do the performances of two binary classifiers, a Random Forest and a LightGBM model, compare in terms of accuracy, precision and recall when applied to the same historical flight dataset of a regional airport?*

(b) *Which input features are most important for predicting arrival and departing flight delays as identified by the models?*

(c) *Does adding historical flight data of comparable airports to the training set improve the predictions by the concept of transfer learning?*

The second sub-question investigates the possibility of predicting the probability density function of an individual flight. Since this is a novelty within the field of flight delay predictions, the model and performance metrics selection is less straightforward and part of the research.

Q2: Is it possible to predict the conditional probability density function of the delay of an individual flight with machine learning algorithms?

(a) *Which machine learning algorithms are able to predict a probability distribution?*

(b) *How should the performance of a probabilistic forecasting algorithm be measured?*

(c) *How well do these algorithms perform for a regional airport in terms of the aforementioned performance metrics?*

Assuming that the second sub-question leads to positive results, the final sub-question relates to how expressing a flight delay prediction as a probability density function rather than a point estimate can improve actual airport operations optimizations.

Q3: What are the effects of incorporating a probabilistic forecasting model in an existing Flight-to-Gate Assignment Problem (FGAP) in terms of efficiency and robustness of the resulting schedule?

Research objective

The main research objective of this thesis is to gain insight into the possibility and effect of evolving flight delay predictions from point estimates to probabilistic forecasts for a regional airport by developing a machine-learning based probabilistic flight delay prediction model and incorporating it into an existing Flight-to-Gate Assignment Problem.

To achieve this, the problem is divided into three parts. Since most existing flight delay studies evolve around large, international airports, the first sub-goal is to predict flight delays at a regional airport by applying similar machine-learning based binary classifiers to historical data of Rotterdam the Hague Airport. The second sub-goal is to evolve flight delay predictions from point estimates to probability density functions by applying probabilistic forecasting methods to the same historical data, a novelty in the field of flight delay predictions. The final sub-goal is to tests the effect of representing flight delays predictions by probability density functions by incorporating the results of the probabilistic flight delay prediction model into an existing Flight-to-Gate Assignment Problem.

3
Literature review

With the project description established, this chapter presents the literature review. Its goal is to identify the most influential researchers, their usual approaches and the current state of the art in the field of flight delay and cancellation predictions. Once completed, it should function as a guide to answer the research questions defined in the previous chapter.

The remainder of this chapter is divided as follows. Section 3.1 provides an overview of research developments within the field and fields closely related. The usual approach for predicting flight delays and cancellations is defined in section 3.2. In section 3.3 the possibility to evolve from the usual prediction methods to probabilistic forecasting is investigated. Finally, section 3.4 presents a case study to illustrate how probability density forecasts for flight delays can support airport operation optimization problems.

3.1. Previous research to flight delay predictions

Over the years, many different approaches have been applied to the flight delay prediction problem. This section provides a general overview of the most relevant researches before elaborating on their selected features, methodology and results in section 3.2.

One of the earlier studies, (Mueller and Chatterij 2002), approaches the flight delay problem by fitting different distributions to historical data. Although not strictly a prediction, the research concludes that arrival flights are best modeled by Normal distribution, while a Poisson distribution best describes the departure flights. In (Xu et al. 2005) the problem is modeled as a Bayesian network, which requires the conditional probabilities between states to be known. (Klein 2010) recognizes the importance of weather and incorporates weather forecasts and observations in a multi-linear regression.

The beginning of the previous decade highlights the start of a new trend in the field of delay predictions: machine learning techniques. A significant contribution is made by (Rebollo and Balakrishnan 2014), which is amongst the first to apply a binary Random Forest (RF) classifier to the flight delay problem and achieves an average accuracy of 80%. Both the method and the accuracy often function as a benchmark in successive research; almost all delay researches that follow that are based on any tree-structured algorithm also include the Random Forest.

With the progress of time, the complexity of the tree-based models increases as well. The binary classification is occasionally extended to a multi-class, leading to the argument that missclassification should be associating with different costs for different classes in both (Alonso and Loureiro 2015) and (Choi et al. 2017). Boosting extensions of decision trees are introduced in (Choi et al. 2016) and (Manna et al. 2017). Within this category of tree-based models, the latest promising addition is LightGBM, which is used in both (Lambelho et al. 2020) and (Shao et al. 2019).

Besides tree-based models, Neural Networks are introduced to flight delay predictions in (Kim et al. 2016) and (Khanmohammadi et al. 2016). A complementary approach is the construction of a two stage model that first determines whether there is a delay, followed by how much delay, as presented in (Thiagarajan et al. 2017). Even more recently (Yu et al. 2019) has developed a combination of a Deep Belief Network and a Support Vector Regression. A detailed overview of the development of the applied models can be found in table 3.2.

For the features the trend of increasing complexity over time does not hold as strongly, although some research specifically focus on the effects of new features. In (Choi et al. 2016), weather forecast features are added to the prediction model. Most researches afterwards however continue to use the actual observed weather, which is easier to obtain. (Chen and Li 2019) emphasizes the benefits of knowing that the previous flight is delayed by feeding this feature into a propagation model. It can be concluded from table 3.1 that this feature trend holds throughout 2019.

3.1.1. Identified gaps in literature

As will be discussed in detail in section 3.2.6, the increased complexity of models and features is rewarded; accuracies above 90% are reported. Despite all positive developments in the field of flight delay predictions, some topics are still left for investigation. This research will focus on the following two.

Regional airports

Most researches evolve around large, international airports. Although these are large stakeholders with very large datasets, there are also many regional airports that could benefit from accurate predictions. The fact that a regional airport mainly serves low cost carriers which might be more difficult to predict, does not have to be an obstruction. Both (Horiguchi et al. 2017) and (McCarthy et al. 2019) explicitly focus on low cost carriers and their results are not significantly inferior to the results of large airports. The smaller database associated with a regional airport makes predictions more challenging but not impossible. (McCarthy et al. 2019) shows that is possible to use data from larger airports through the concept of transfer learning, and (Gui et al. 2020) shows high classifying accuracy for a dataset that is smaller than the dataset of the average regional airport. Therefore, the first potential contribution of this research is applying existing flight delay knowledge to a regional airport.

Probabilistic forecasting

The second identified gap in literature relates to the target of flight delay predictions. Almost (if not) all of the researches that involve machine learning algorithms aim to predict the delay in either minutes or a delay class. Almost none of them consider the importance and potential benefit of adding probabilities to the estimate, and none of them predict an entire probability density function. Certain optimization problems however, for example the flight-to-gate assignment, could benefit from probabilistic forecasting. Therefore, the second potential contribution of this research is introducing probabilistic forecasting to the field of flight delay predictions.

3.2. A binary classifier for flight delays and cancellations

Following the general literature overview in the previous section, this section presents literature evidence to define the usual flight delay prediction approach in detail. Upon completion it will function as a baseline for the methodology part of this research. This section first discusses typical challenges encountered, such as feature and model selection, and ends with an overview of case studies and achieved results.

3.2.1. Feature collection, encoding and selection

The first step in defining a usual approach is to understand the required data in detail. For each of the previously identified relevant researches, the prediction horizon, features, encoding types and selection method are summarized in table 3.1. Each topic is individually examined in the following sections.

Features

According to (Neufville and Odoni 2013), the causes of flight delays can be divided into five groups; weather, reactionary delay (i.e. the late arrival of a previous flight), airline, the airspace system and security. With the latter two being too complex to incorporate in this research, this literature review investigates how previous research has incorporated features of the following groups; the flight schedule, the actual weather, weather forecast and the late arrival of the previous flight. A detailed overview of the specific features can be found in table α.1.

As expected, all flight delay studies encountered in table 3.1 use flight schedule features including the origin, destination, airline, and scheduled arrival and departure times. A large number of studies incorporate weather features, but only a couple integrate weather forecasts. The logical explanation to discard weather forecasts is two-folded. Foremost it is difficult to obtain weather forecasts and secondly, actual weather, which is essentially a perfect weather forecast, will result in a better performance. The latter is confirmed by (Choi et al. 2016), where it is stated that *'the predictions with forecast were much worse than the predictions with the actual weather'*, a result that will be further discussed in section 3.2.6. Considering that in real-life applications only forecasts are available, this difference should be kept in mind.

The final category, the late arrival of a previous flight, is incorporated in roughly half of the studies. This corresponds with the idea that the feature is useful, but that the links between subsequent flights can be difficult to obtain for large airports. For smaller airports it might be easier to obtain the links, however it could reduce the smaller database even further if not all flights have a qualified predecessor. Nevertheless, considering the benefit of improving the predictions, this research aims to include the arrival time of the previous flight as a feature.

Horizon

Directly related to the features is the prediction horizon, which is the time between the prediction and the actual observation. The longer the prediction is made in advance, the less features are available and vice versa. More time in advance leads to more time to incorporate the predictions in strategical planning, more features leads to potentially better predictions, meaning a trade-off has to be made.

As can be seen in table 3.1, none of the encountered studies have a prediction horizon larger than 6 month, and all studies include the flight schedule as a feature. This can be explained by the fact that flight schedules are the minimum requirement to predict the delay or cancellations of individual flights and they are usually published 6 months in advance. Weather features are only accurate when the horizon is at most a couple days, and to know the delay of the previous flight, that flight has to land first, resulting in a prediction horizon of at most a couple hours.

Two researches, (Lambelho et al. 2020) and (Horiguchi et al. 2017), choose for a prediction horizon of several months with less features, all others choose the shorter prediction horizon with more features. Since this research recognizes the benefits of both options, it will select two distinct feature sets; one suitable for a prediction horizon of one month and another suitable for predictions on the same day.

Reference	Horizon	Features				Encoding	Selection
		FS	AW	WF	LA *		
(Gui et al. 2020)	0 days	✓	✓	✗	✗	enumeration, mapping letters to numbers	-
(Lambelho et al. 2020)	6 months	✓	✗	✗	✗	target, trigonometry, geographical	RFE
(Chen and Li 2019)	0 days	✓	✓	✗	✓	one-hot	RFE
(McCarthy et al. 2019)	0 days	✓	✗	✗	✓	one-hot	-
(Shao et al. 2019)	4 hours	✓	✓	✗	✓	one-hot	PCA
(Yu et al. 2019)	2 hours	✓	✓	✗	✓	one-hot	correlation
(Choi et al. 2017)	0 days	✓	✓	✗	✓	one-hot, normalization	p-value
(Horiguchi et al. 2017)	1 day - 5 months	✓	✗	✗	✗	one-hot, trigonometry, normalization	depending on horizon
(Manna et al. 2017)	0 days	✓	✗	✗	✗	enumeration, normalization	correlation
(Thiagarajan et al. 2017)	0 days	✓	✓	✗	✗	ordinal, normalization	-
(Choi et al. 2016)	0/1/5 days	✓	✓	✓	✓	enumeration, normalization	-
(Khanmohammadi et al. 2016)	hours	✓	✗	✗	✓	ordinal, normalization	-
(Kim et al. 2016)	0 days	✓	✓	✗	✗	-	by the model
(Alonso and Loureiro 2015)	0 days	✓	✓	✗	✓	geographical, binary, ordinal, aircraft length	by the model
(Rebollo and Balakrishnan 2014)	2 hours	✓	✓	✗	✗	-	p-values
(Klein 2010)	6 hours	✓	✓	✓	✗	-	-
(Xu et al. 2005)	1:45 hours	✓	✓	✓	✗	estimated conditional probabilities	experts, R^2
(Mueller and Chatterij 2002)	-	✓	✗	✗	✗	-	-

* FS = Flight Schedule AW = Actual Weather WF = Weather Forecast LA = Late Arrival

Table 3.1: Literature overview of features, encoding and selection

Categorical feature encoding

As can be seen in table $\alpha.1$, a significant portion of the potential features, e.g. the origin, destination and airline, is categorical. Many machine learning algorithms work better (or only) with numerical features, meaning that the categorical ones have to be encoded. Some basic but frequently occurring encoding techniques are one-hot and ordinal encoding, but they each come with disadvantages. One-hot, where each feature is extended to a separate binary feature for each option, has the potential to make the problem very large. Ordinal encoding, also described in articles as the enumeration of features, assumes that one categorical feature is somehow better than others, which is often not the case.

In several articles alternative encoding methods are proposed that are all considered possibilities for this research. (Alonso and Loureiro 2015) suggests the use of geographical coordinates to encode airport categories. (Lambelho et al. 2020) and (Horiguchi et al. 2017) emphasize on the importance of encoding cyclical features with trigonometric functions:

$$sin\left(\frac{2\pi t}{t_{cycle}}\right) \quad \text{and} \quad cos\left(\frac{2\pi t}{t_{cycle}}\right) \tag{3.1}$$

(Lambelho et al. 2020) also presents the concept of target encoding, where each categorical features is encoded with the probability of a delay, given that specific option of the feature:

$$X_i \rightarrow S_i \cong P(Y|X = X_i) \tag{3.2}$$

Finally, six of the researches apply normalization to their numerical features to scale them between zero and one, with the argument that it improves numerical stability and reduces training time. In principle, neural networks and random forests do not require this scaling, but it will be considered for other algorithms if needed.

Feature selection

Multicollinearity is the phenomenon that more than two explanatory variables are highly correlated to each other. For some machine learning algorithms, in particular regression models that assume all variables to be independent, this reduces the performance.

Based on the literature overview in table 3.1, two main options are identified to actively counter multicollinearity. The first one is a Pearson correlation matrix, where the correlations between all pairs of features are plotted. Based on this table one of two highly correlated variables is removed, a clear and quick approach. The second method is Recursive Feature Elimination (RFE), described in (Granitto et al. 2006), which is an algorithm that eliminates redundant features by systematically running the model with different subsets and comparing the performance. Although thorough, this approach is time consuming. For this research the clearness of the Pearson correlation matrix is preferred, but if a more extensive algorithm turns out to be required, the RFE is considered next.

3.2.2. Classification models

Now that all feature related topics are discussed, this section will investigate which models suit the first phase of this research. Table 3.2 provides an overview of the methodology used in relevant literature. It should be noted that (Klein 2010), (Xu et al. 2005) and (Mueller and Chatterij 2002) consider a statistical approach, network model and probabilistic fitting respectively, which do not strictly fall in the machine learning category. They are included for completeness, but their models are not considered appropriate for this research.

As can be seen in the overview, over half of the selected researches consider the flight delay problem as a classification problem with either a multi-class or a binary target. The other half estimates the delay in actual minutes, meaning they approach the flight delay as a regression problem. The researches (Thiagarajan et al. 2017) and (Kim et al. 2016) combine the two approaches in a two stage model; the first stage predicts whether or not a flight is delayed, the second stage estimates the minutes if the first phase indicates a delay. Only one of the selected flight delay researches also takes cancellations into account, which is (Lambelho et al. 2020). Although the main focus of this research is the prediction of flight delays, this confirms the idea that the same models can be applied to predict cancellations, which motivates this research to do the same.

Reference	Target	C*	Models	Imbalanced Data	Feat. Importance
(Gui et al. 2020)	Multi-class (4)	×	LSTM, RF	undersampling	-
(Lambelho et al. 2020)	Binary	✓	LightGBM, MLP, RF	class weights	SHAP values
(Chen and Li 2019)	Multi-class (15)	×	RF, Delay Propagation Model	SMOTE	
(McCarthy et al. 2019)	Minutes	×	LSTM, Transfer Learning	-	-
(Shao et al. 2019)	Minutes	×	LightGBM, LR, MLP, SVR	-	change in RMSE
(Yu et al. 2019)	Minutes	×	Deep Belief Network + SVR	-	change in MSE
(Choi et al. 2017)	Binary	×	AdaBoost, DT, k-NN, RF	costing sampling	-
(Horiguchi et al. 2017)	Binary	×	DNN, RF, XGBoost	-	by the model
(Manna et al. 2017)	Minutes	×	Gradient Boosting	-	Pearson Correlation
(Thiagarajan et al. 2017)	Minutes, Binary	×	AdaBoost, DNN, Extra-trees, Gradient Boosting, MLP, RF	SMOTE	-
(Choi et al. 2016)	Binary	×	AdaBoost, DT, k-NN, RF	SMOTE	change in ROC
(Khanmohammadi et al. 2016)	Minutes	×	ANN with multi-level input	-	-
(Kim et al. 2016)	Binary (>15/30min)	×	LSTM	-	-
(Alonso and Loureiro 2015)	Multi-class (5)	×	Binomial Trees with unimodal output	-	-
(Rebollo and Balakrishnan 2014)	Binary (>60min)	×	RF	oversampling	by the model
(Klein 2010)	Minutes	×	Multiple LR	-	-
(Xu et al. 2005)	Multi-class (5)	×	Bayesian Network	only delayed flights	R^2 for different horizons
(Mueller and Chatterij 2002)	PDFs	×	Fitting Distributions	-	frequency of occurrence

* C = Cancellation

Table 3.2: Literature overview of methodology

Based on the overview it can be concluded that most classification problems apply at least one bagging or boosting extension of the Decision Tree (DT) algorithm. The most popular one is the well known Random Forest (RF), a bagging extension of the DT, which is used in almost all classification researches listed. Other popular options are different types of boosting extensions of the DT. (Choi et al. 2016) and (Thiagarajan et al. 2017) use AdaBoost, (Horiguchi et al. 2017) uses XGBoost and (Shao et al. 2019) and (Lambelho et al. 2020) apply the recently developed LightGBM.

For this research both a bagging (RF) and a boosting (LightGBM) extension of the DT algorithm are selected, to accommodate for comparison and validation. The next two sections describe each model in more detail.

Random Forest

The first model selected is the commonly used Random Forest (RF) classifier, an algorithm that originates from (Breiman 2001). In principle this model is a collection of Decision Tree (DT) classifiers. Each tree in the collection is based on a bootstrap sample of the training data, meaning the sample is drawn uniformly and with replacement. For a classification the RF determines its prediction by taking the majority vote of the test results of each individual tree. Analogously, when applied to a regression problem it determines its output by taking the average. This procedure of sampling and assembling is also known as bootstrap aggregating or bagging.

One of the benefits of a Random Forest is that it is less prone to overfitting, as the result of adding more trees converges by the law of large numbers. Given that this research focuses on smaller, regional airports, with relatively small databases, this might be a particularly useful characteristic. Another benefit is that a decision tree ranks features by default, as it places more important features higher in the tree. This will be useful when answering the research question regarding feature importance. Finally, the model is relatively easy to interpret, which makes it a solid benchmark for more complicated algorithms such as neural networks, which are used in the second phase of this research.

LightGBM

The second model is LightGBM, which was recently developed by (Ke et al. 2017) and stands for Light Gradient-Boosting Machine. Similar to the RF, LightGBM is also an ensemble technique based on the DT algorithm. The main difference however is that this is a boosting algorithm, meaning that it is trained by improving the decision tree in sequential steps. At each step, a random sample is used to construct a new tree, based on reducing the classification error of the previous tree.

The most time consuming part of any gradient boosting algorithm is determining the splitting point of each feature. LightGBM introduces two new concepts to reduce the number of features and data samples and therefore decrease the computational time. Gradient Based One Side Sampling (GOSS) selects the samples with large gradients and randomly downsamples features with smaller gradients, to reduce the number of data points with as little as possible information loss. Exclusive Feature Elimination (EFE) reduces the number of features by bundling features that are mutually exclusive, i.e. are never simultaneously zero.

LightGBM also stands out from other gradient boosting algorithms by applying leaf-wise growing of the tree instead of level-wise growing, which allows the trees to be more complex and more accurate. In both flight delay researches that use LightGBM, i.e. (Shao et al. 2019) and (Lambelho et al. 2020), it is indeed the most accurate algorithm. The disadvantage of LightGBM is that it complexity makes it prone to overfitting on small datasets. Given that this research focuses on regional airports, this is an important point of attention. The expected challenges of a small dataset are further discussed in 3.2.6.

3.2.3. Imbalanced dataset

Another point of attention is the high Imbalance Ratio (IR) of the dataset. Since most flights arrive and depart within 15 minutes of their scheduled departure and arrival times, and even more flights are operated as opposed to cancelled, the dataset for this research is highly imbalanced. This is unfavorable since many machine learning algorithms are designed to maximize the accuracy and reduce the error. As a result, in case of a highly imbalanced dataset those algorithms will be trained with a bias towards the majority class, leading to missclassification of the minority class.

Based on the flight delay related literature in table 3.2, two options are identified to accommodate for the high IR of the dataset. The first option is to even the number of samples in each class by oversampling or undersampling. Since the latter has the disadvantage of potentially losing valuable information, a commonly used oversampling method is the Synthetic Minority Oversampling Technique (SMOTE)(Chawla et al. 2002). In order to generate more samples of the minority class, SMOTE first draws a line between a random instance of the minority class and one of its randomly selected k-nearest neighbors. It then randomly identifies a point on this line as a new sample in the minority class and continues this process until both (or all) classes have the same sample size.

The second option is to make the model cost sensitive by increasing the weight of miss-classifying the minority class in the loss function. In (Lambelho et al. 2020), these weights are set to maximize the f1-score; the harmonic mean of the recall and precision which are explained in the following section.

3.2.4. Performance metrics for classification algorithms

		actual	
		positive	negative
predicted	positive	True Positive (TP)	False Positive (FP)
	negative	False Negative (FN)	True Negative (TN)

Table 3.3: Confusion matrix

$$accuracy = \frac{TP + TN}{total} \tag{3.3}$$

$$recall = \frac{TP}{TP + FN} \tag{3.4}$$

$$precision = \frac{TP}{TP + FP} \tag{3.5}$$

The most intuitive and a commonly used metric to express the performance of a classification algorithm is the *accuracy* in equation 3.3, which is the fraction of correctly identified instances. In principle, the higher the score the better, where the score is bound between 0 and 1. As explained in the previous section, accuracy alone is not sufficient when using highly imbalanced data. Therefore, two additional metrics are often taken into account as well; the *recall* and the *precision* in equation 3.4 and 3.5 respectively. The *recall* is a representation of how many of the actual positive instances are also identified as positive. The *precision* represents how many of the predicted positives are also actual positives. Again, both metrics are bound between 0 and 1, and a higher score is preferred for both.

It is important to note that precision and recall are a trade-off; increasing either will decrease the other. When both metrics are equally important, and the goal is to find the optimal balance point, the f1 score can be optimized. It combines both metrics in a single one by taking their harmonic mean:

$$F_1 = \frac{2 * precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \tag{3.6}$$

Another widely used performance metric is the Area Under the Receiver Operating Curve (ROC)(AUC). The *ROC* is defined as follows:

$$ROC = \frac{TPR}{FPR} = \frac{\frac{TP}{TP+FN}}{\frac{FP}{FP+TN}} \tag{3.7}$$

The integrated AUC falls between 0 and 1, and the closer to 1 the better. It should be observed that a random classifier already achieves a 0.5 AUC. A summary of the performance results found in literature is presented in table 3.5 and discussed in section 3.2.6.

3.2.5. Feature importance

Not only is this research interested in how well a classifier performs, it is also interested in identifying the most important features for flight delay and cancellation predictions at regional airports. Several methods are found in literature and summarized in table 3.2. Some models, such as a decision tree, inherently identify the most important features as they are represented by the highest splitting nodes in the tree. In the researches (Shao et al. 2019), (Yu et al. 2019) and (Choi et al. 2016), the effect of the adding and removing certain groups of features, such as weather or late arrivals, is quantified by the change in performance of the model.

The most complete method to quantify feature importance is found in (Lambelho et al. 2020). The SHAP (SHapley Additive exPlanations) values approach, proposed by (Lundberg and Lee 2017), is based on shapley values and represents the contribution of each feature for the expected outcome, averaged over all permutations of feature order. A higher absolute SHAP indicates a higher contribution, the sign indicates the direction. This research prefers this method for two reasons. Firstly, it is applicable to many different machine learning algorithms. Secondly, it allows for very informative visualizations.

Five of the selected articles explicitly rank the features by importance. The identified important features for flight delay predictions are summarized in table 3.4 and highlighted in bold in appendix table α.1. Most of the features identified are only available on the day of departure. Both (Chen and Li 2019) and (Yu et al. 2019) emphasize the importance of the late arrival of a previous flight, a feature that could be constructed for this research as well. Amongst the features that are available longer in advance, the influence of the scheduled departure time is significant in both (Lambelho et al. 2020) and (Horiguchi et al. 2017).

A noticeably absent feature is the direct weather information in the form of METAR data, even though both (Neufville and Odoni 2013) and (Mueller and Chatterij 2002) address that weather is often the cause of a delay. Although out of the five articles listed only (Shao et al. 2019) considers METAR in the first place, it will be interesting to see if the presumption that other features are more important than weather features also holds in this research.

Reference	Horizon < 24 hours	Horizon ≥ 1 days
(Lambelho 2019)		arrival ATFM delay, airline, **hour**, seats
(Chen and Li 2019)	departure delay group, **late aircraft delay**	
(Shao et al. 2019)	airport GPS trajectories	
(Yu et al. 2019)	air route situation, (actual) airport crowdedness, **delay of previous flight**	
(Horiguchi et al. 2017)	scheduled fuel on board, number of passengers	**scheduled departure minute of day**

Table 3.4: Important features

Reference	Model[1]	Horizon[2]	Airport(s) or *Airline(s)*	Data points	Performance[3]
(Gui et al. 2020)	C_4	0D	all flights in the area	5,761	accuracy= $\{90.2\%(2), 81.4\%(3), 70.0\%(4)\}$, > 40 min. error in 27% of the cases
(Lambelho et al. 2020)	C_2	6M	LHR	2.3 million	accuracy > 0.75 for delays, **accuracy** > **0.98** for cancellations
(Chen and Li 2019)	C_{15}	0D	ORD	1 year of flights	relaxed **accuracy = 0.92669**, accuracy = 0.86727
(McCarthy et al. 2019)	R	0D	*Small and large* *European LCC*	24,000 (small), 340,000 (large)	+ transfer learning: RMSE = 10.2, − transfer learning: RMSE = 9.2
(Shao et al. 2019)	R	4H	LAX	2 months of flights	RMSE = 37, accuracy around 18 min
(Yu et al. 2019)	R	2H	PEK	1 year of flights	99.3 % of the predicted values are within 25 min deviation from the actual value
(Choi et al. 2017)	C_2	0D	45 major airports	1-2 million	accuracy = $\{82.8\%, 75.5\%, 65.3\%\}$ for cost ratio $\{1:1, 1:5, 1:10\}$

(Horiguchi et al. 2017)	C_2	1D-5M	*Peach Aviation*
			(Asian LCC)
				54,000	for 5 months: AUC <0.6
					for 1 week: AUC <0.6,
					for 1 day: AUC = 0.647

(Manna et al. 2017)	R	0D	70 busy US airports	2,175,534	arrivals: R^2 = 92.3%, departures: R^2 = 94.9%
(Thiagarajan et al. 2017)	C_2/R	0D	15 major US airports	3.2 million	C: **accuracy = 94.35**%, R: MSE = 26.36, R^2=0.985
(Choi et al. 2016)	C_2	0D-5D	45 major airports	2 million	0 days: accuracy = 80.36%, AUC = 0.68, 5 days: accuracy = 26.79%
(Khanmohammadi et al. 2016)	R	H	JFK	1 month of flights	RMSE = 0.1366
(Kim et al. 2016)	C_2	0D	10 major US airports	5.5 years of flights	stage 1: accuracy ~ 90%, stage 2: accuracy~ 87%
(Alonso and Loureiro 2015)	C_5	0D	Porto Airport	26,189	r_{int} = 0.7 (network), r_{int} = 0.66 (trees)
(Rebollo and Balakrishnan 2014)	C_2/R	2H	100 most delayed US OD pairs	2 years of flights	C: avg. accuracy = 81%, R: avg. median error = 20.9 min
(Klein 2010)	R	6H	ORD	1 year of data	accuracy = 80-85%
(Xu et al. 2005)	C_5	1:45H	ORD, LGA, ATL	3 months of flights	error rate = 19.1 %
(Mueller and Chatterij 2002)	-	-	10 major US airports	21-days of flights	fit errors for different distributions

[1] C_x = Classification$_{classes}$ R = Regression [2] H = Hours D = Days M = Months

[3] The highest accuracies for classification models are shown in bold

Table 3.5: Literature overview of case studies

3.2.6. Case studies

The final part of this section is dedicated to the case studies performed in the selected literature. As highlighted in table 3.5, the highest accuracy for flight delay classification is 94.35%, obtained with a Gradient Boosting model for arrival delays in (Thiagarajan et al. 2017). It should be noted that this result is achieved for a binary classification with a 0 day prediction horizon.

Increasing the number of classes generally reduces the accuracy. This corresponds to the results of (Gui et al. 2020), where the accuracy reduces from 90.2% for two classes to 70.0% for four classes. (Chen and Li 2019), which uses 15 delay classes, argues that missclassification in an adjacent class is still a relatively good result and includes them in the relaxed accuracy. This relaxation leads to an accuracy of 92.67%, the second highest accuracy for flight delay of the selected articles.

Another factor that generally influences the performance is the prediction horizon. As the results of (Choi et al. 2016) show, increasing the horizon from 0 days to 5 days leads to an accuracy reduction from 80.36% to 26.79%. This performance is particularly poor considering there are only two classes, meaning a random guess would give an accuracy of 50%. A similar trend is observed in (Horiguchi et al. 2017), where the poor prediction result leads to the conclusion that there is not enough information available 5 months prior to the day of the flight. This makes the 98% accuracy for cancellation predictions with a 6 months horizon in (Lambelho et al. 2020) even more remarkable. A possible explanation for this high accuracy could be the high imbalance ratio of the cancellations dataset; however the article does take this ratio into account and the resulting f_1 score of 0.6 is significantly better than the 0 that a naive approach with zero true positives would give.

Regional airports and small datasets

Almost all case studies involve one or more major airports, most often airports in the United States. There are several advantages associated with these airports; the databases for their on-time performance and cancellations already exist, are available and most importantly they contain a lot of data points. In principle, more data leads to better model performance. The lack of large databases is an inherent and major challenge in the prediction of flight delays at regional airports; foremost they simply serve less flights and secondly, adding much older data is not a desired solution as the circumstances of those flights might be very different than the circumstances of today's target flights.

As anticipated, none of the articles found specifically focus on regional airports. A relatively small dataset is used in (Alonso and Loureiro 2015), but Porto Airport is still an international airport and the chosen performance metric is difficult to compare to other results. (Gui et al. 2020) uses the smallest database with only 5,761 flights, all collected within the range of a certain ADS-B receiver. Their classification results, in particular the binary classification accuracy of 90.2%, are well in the range of studies at larger airports, which is a promising outlook for this research.

Two of the case studies focus specifically on flight delays of Low Cost Carriers (LCC). Although they do not particularly focus on regional airports, they are still very relevant considering regional airports mainly serve LCCs. The first research, (Horiguchi et al. 2017), uses a dataset of 54,000 flights from an Asian LCC. Recalling that a randomly guessing algorithm achieves an AUC of 0.5, the results for a prediction horizon of 5 months or 1 week are below par. The result for predictions with a one day window however are not far below the results of the large database study in (Choi et al. 2016); an AUC of 0.647 versus an AUC of 0.68. This is again a promising result for this research.

Transfer learning

Perhaps the most interesting approach to the small dataset challenge is found in the second research that investigates the delay behaviour of LCC. (McCarthy et al. 2019) applies the concept of transfer learning to train the prediction model for a small LCC on a dataset that includes data from another, much larger airline. Specifically, the research applies the heterogeneous feature framework of (Moon and Carbonell 2017), which allows for only partially overlapping the training and target feature sets. The results consistently show that enhancing models with transfer learning reduces the RMSE and thus improves the prediction.

This research will apply the theory of transfer learning in two variants. The first will enhance the training set with data from Eindhoven Airport (EIN), a similar, regional airport. The second option involves adding data from Amsterdam Airport Schiphol (AMS) a closely located, large, international airport. The benefit of using EIN is that the airports are most similar, the advantage of AMS is that it has a large dataset. Both results will be benchmarked against the model without transfer learning.

3.3. A probabilistic forecasting approach for flight delays

The previous section presented the usual approaches in the field of flight delay predictions and discussed the challenges of applying them to regional airports. It also indirectly showed that most (if not all) of the existing literature focuses on machine learning delay predictions with a point estimate as outcome. Some applications however would benefit from an accurate delay probability estimate, ideally a complete conditional probability density function. This section investigates how this problem is approached in other fields.

3.3.1. Probabilistic machine learning algorithms

The first step is to identify which models are suitable for the predictions of delay probability density functions of individual flights. Some earlier studies, such as (Mueller and Chatterij 2002), fit different standard distributions to historical flight data and select the one that fits best. It should be emphasized that these probability density function estimates are fittings and not predictions by the definition of this research.

As examples in the following sections will show, the prediction of probability densities is a topic of interest in the fields of electricity networks, weather forecasts and speech recognition. Although there are more complicated variations available, this research will apply the following three models for a step-wise and verifiable evolving from the point estimates in 3.2 to the probabilistic forecasting in the second phase.

Multi-class Random Forest

The first model selected is the Multi-class Random Forest. In principle this is the same RF model as described in 3.2.2, only with the addition of more possible classes for the prediction. As explained before, the RF is ensemble technique that uses the majority vote of many decision trees to determine its prediction.

An intuitive and easily interpretable approach to turn these into probabilities is by taking all votes and determine the ratio of occurrence for each individual class. This idea is supported by (Niculescu-Mizil and Caruana 2006), which confirms that bagging gives non-biased probabilities. The number of classes should be set high enough to turn the votes into a probability histogram, but low enough to allow sufficient data points for each class. When the result is successful, this model functions as a bridge between the first and second phase of this research and provides a benchmark for the next two models.

Soft-max Regression Network

Although the multi-classification approach is a nice starting point, it is assumed that models specifically designed to predict probabilities perform even better. This leads to the field of neural networks. The first network discussed is the Soft-max Regression Network, which is the combination of a Deep Neural Network (DNN) and a Soft-max Regression output layer, as presented in (Jiang et al. 2018) and illustrated in figure 3.1a.

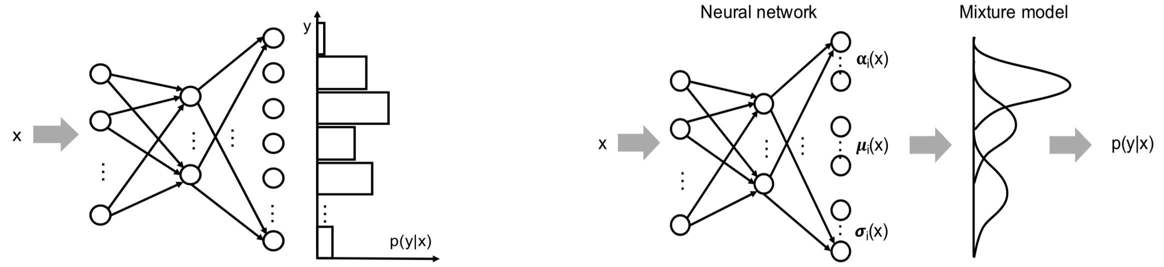
Each neuron in the output layer of the DNN represents one of the classes that the output can be assigned to. The additional output layer uses soft-max regression, also known as a multi-class logistic regression, to transform these outcomes to probabilities. The result is a probability histogram where each bin has a value between 0 and 1 and the total sums to 1.

Since both outcomes are probability histograms, the results of this model can easily be compared with the results of the previously described Multi-class Random Forest. Simultaneously, it is based on a DNN, which is also used in the next model. It is therefore a valuable second step towards the prediction of a complete probability density function.

Mixture Density Network

The third and final model is the Mixture Density Network, as defined in (Bishop 1994) and illustrated in figure 3.1b. Similar to the Soft-max Regression Network it is based on a standard DNN with a probabilistic extension, this time a mixture model. The outputs neurons DNN can be divided into sets of three. Each set represent a Gaussian function, where one neuron represents the mean and one the standard deviation. The third neuron represents the mixture coefficient of the Gaussian. The mixture model constructs a probability density function based on the mixture coefficients, means and standard deviations estimated by the network.

Two examples of the mixture density models in practice are (Vossen et al. 2018), which applies it to predict power load peaks in an energy network, and (Raeis et al. 2019), which applies it to queuing theory. Although the resulting density functions are very similar to the goal of this research, it should again be emphasized that DNN requires very large datasets. It is expected that the success of this approach largely depends on the success of the previously discussed transfer learning technique.



(a) Soft-max Regression

(b) Mixture Density Network

Figure 3.1: Probabilistic Neural Networks (Vossen et al. 2018)

3.3.2. Performance metrics for probabilistic forecasting algorithms

To determine the performance of the predicted probability histograms and density functions, additional performance metrics are required. Three types of scores will be taken into account, each described in one of the following sections.

Performance metrics for point estimates

Although these models are chosen for their ability to estimate probabilities, the point estimate that can be derived from them still has to be sufficiently accurate. Also, quantifying the performance of the estimate allows for comparison with other flight delay researches. For the Multi-class Random Forest and Soft-max Regression Network the previously described classification metrics in section 3.2.4 can be applied. For the Mixture Density Networks the difference between the predicted predicted point estimate and actual delay can be quantified with the following well-known metrics, typically used for regressors:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{3.8}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(\bar{y}_i - y_i)^2} \tag{3.9}$$

$$MAE = \frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{n} \tag{3.10}$$

Prediction Interval (PI) metrics

Some researches, for example (Xie et al. 2020), focus on predicting confidence intervals with Neural Networks directly, rather than first estimating a complete probability distribution. Although this research aims to predict a probability density function rather than a confidence interval, the interval is closely related. Therefore, the following prediction interval metrics are also suitable for this research:

$$PICP = \frac{1}{n}\sum_{i=1}^{n} c_i \tag{3.11}$$

$$PINAW = \frac{1}{nR}\sum_{i=1}^{n}(U_i - L_i) \tag{3.12}$$

$$CWC = PINAW(1 + \gamma(PICP)e^{-\eta(PICP-\mu)}) \tag{3.13}$$

The Prediction Interval Coverage Probability (PICP) represents how often the estimated interval contains the actual target. The value is bounded between 0 and 1 and in principle it holds that a higher value is preferred. At the same time, a confidence interval should be as narrow as possible, as it indicates the certainty of the prediction. The Prediction Interval Normalized Average Width (PINAW) sums the differences between the estimated upper and lower bounds, and normalizes it with the sum of the naive estimate of the confidence intervals, which is the range of the target interval. Since there is a trade-off between a high PICP and low PINAW, the Coverage Width-based Criterion (CWC) is introduced as a metric to indicate models with the best balance between the two.

Continuous Ranked Probability Score (CRPS)

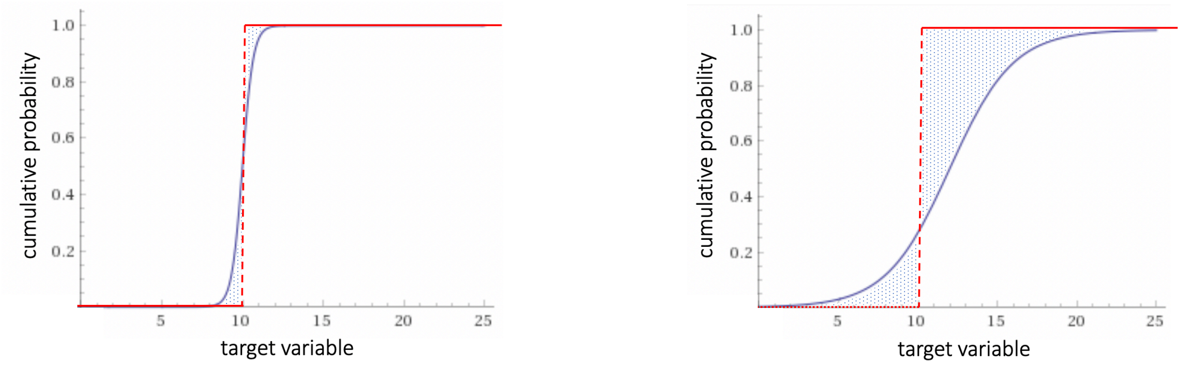
The third category of performance metric encountered in literature, is the category of the Continuous Ranked Probability Score (CRPS). This score originates from the field of weather forecasts (Hersbach 2000) and distinguishes itself by scoring the entire shape of the curve. An example of this score in combination with a Mixture Density Network can be found in (Vossen et al. 2018), a study that estimates power loads in electricity networks. Besides the continuous score, there also exists a Discrete Probability Ranking Score (DRPS). The two are defined as follows:

$$CRPS = \int_{-\infty}^{\infty} (F(y) - 1(\hat{y} - y))^2 dy \tag{3.14}$$

$$DRPS = \frac{1}{K} \sum_{k=1}^{K} (p_k - o_k)^2 \tag{3.15}$$

The underlying idea for both scores is to model the true value as a Heaviside step function, as indicated by $\mathbf{1}(y - \hat{y})$ and o_k. The difference between the cumulative density $F(x)$ and the step input is then squared and integrated in the continuous score. Analogously, the difference between the step input and cumulative of the probability bins p_k is squared and summed in the discrete form.

As illustrated for the CRPS in figure 3.2, the more similar the curve is to the Heaviside step function, in this example plotted at a true target value of 10, the smaller the difference between the two. A smaller difference leads to lower probability ranking scores. The minimum score is zero, which is achieved if the algorithm predicts the exact value with a confidence interval width of zero, the "perfect" prediction. The further the point estimate of the curve moves away from the true target and the wider the confidence interval, the higher the CRPS or DRPS. The performance of the entire probabilistic forecasting algorithm can be measured by taking the average score over all test instances.



(a) Narrow probability curve

(b) Wide probability curve

Figure 3.2: The effect of the shape of the probability curve on the CRPS

3.4. Integrating probabilistic delay forecasts into optimization problems

The final section of this literature review investigates how probabilistic delay forecasting can be incorporated into an optimization problem. The underlying assumption is that an adequate estimation of the probability of a flight delay can positively contribute to strategical planning. This theory will be tested on an existing Flight-to-Gate Assignment Problem.

3.4.1. A Flight-to-Gate Assignment Problem (FGAP)

One of the challenges that all airports face is how to assign all flights to the available gates. The assignment directly translates to the effective capacity of the airport. The closer the sequential flights are scheduled after each other, the more flights each existing gate can serve in a day. However, as studied in the previous sections, flights often divert from their scheduled arrival and departure times. When the flights are scheduled too tightly, the schedule is not robust enough to absorb these delays, leading to traffic congestion at the apron. Summarized, the assignment of flights to gates is a constant trade-off between the efficient use of capacity and the robustness against delays.

The usual approach within the aviation industry is to treat the Flight-to-Gate Assignment Problem (FGAP) as a Linear Problem (LP). Following the definition in (Schaijk and Visser 2017), the objective function of the basic multiple slot FGAP can be defined as follows:

$$\min \left[Z = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=1}^{k} c_{ij} x_{ijt} \right] \qquad \text{for } i \in N, j \in M \text{ and } t \in K \tag{3.16}$$

where N is the set of scheduled flights, M the set of gates and K the set of time slots with n, m and k as total number respectively. Furthermore, c_{ij} is the cost of assigning flight i to gate j for a single time slot and x_{ijt} the binary decision variable for assigning that flight at time slot t. It is subjected to the following set of constraints:

$$\sum_{j=1}^{m} s_{it} x_{ijt} = 1 \qquad \text{for } i \in N \text{ and } t \in K \tag{3.17}$$

$$\sum_{i=1}^{n} s_{it} x_{ijt} \leq 1 \qquad \text{for } j \in M \text{ and } t \in K \tag{3.18}$$

$$s_{it} x_{ijt+1} - s_{it+1} x_{ijt} = 0 \qquad \text{for } i \in N, j \in M \text{ and } t \in K \tag{3.19}$$

with s_{it} the binary presence coefficient of flight i at time slot t. Constraint 3.17 ensures that each present flight is assigned to a gate, constraint 3.18 makes sure that there is only one flight assigned to a gate for a certain time slot and constraint 3.19 ensures that a flight remains assigned to the same gate in the subsequent time slot.

Presence probability

A major assumption in the basic FGAP is that the presence s_{it} of each flight is known in advance and is fixed. The possibility of delays and cancellations are not taken into account. Some researches extend the model by including a certain penalty or margin for flight delays in the cost function. A different and innovative approach is presented in (LOrtye 2019) and (Schaijk and Visser 2017). Both researches account for potential delays by replacing constraint 3.18 with a new constraint that considers the presence of a certain flight as a probability rather than a binary:

$$\sum_{i=1}^{n} f(p_{it}, r) p_{it} x_{ijt} \leq 1 \qquad \text{for } j \in M \text{ and } t \in K \tag{3.20}$$

$$\text{with} \qquad f(p_{it}, r) = \frac{p_{it}}{r + p_{it}^2} \tag{3.21}$$

Here, r is the input parameter for the maximum allowed probability overlap and p_{it} the probability that flight i is present at the apron at time t. To determine the value of p_{it}, presence probability curves are constructed. Given a certain turnaround, the presence probability curve consists of the cumulative functions of two delay probability densities: one around the Scheduled Time of Arrival (STA) of the incoming flight and one around the Scheduled Time of Departure (STD) of the outgoing flight. This concept is visualized in figure 3.3.

In short the cumulative probability distributions around the STA and STD are constructed by grouping all flights based on either their airline or region. For each possible option the researches construct a presence probability curve by determining how many of the total flights are present at a certain time interval from their scheduled arrival or departure time. This method is described as a linear regression, although it appears to be closer to a fitting.

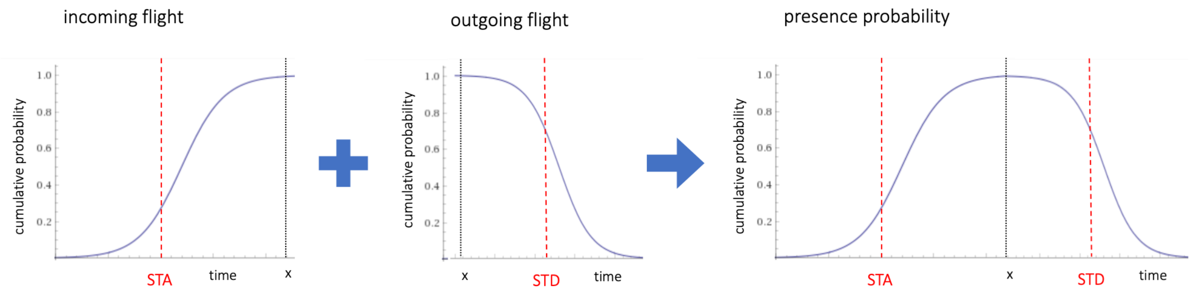


Figure 3.3: Construction of a presence probability curve

An example of the effect of the new probabilistic constraint on the scheduling of two sequential flights is shown in 3.4. Here the blue curves are the discussed presence probabilities, indicated by p_i in constraint 3.20. The green line indicates the probability that the two will overlap, which is constrained by input parameter r in constraint 3.20. By allowing a certain presence overlap the existing gates can be used more efficiently compared to the deterministic approach.

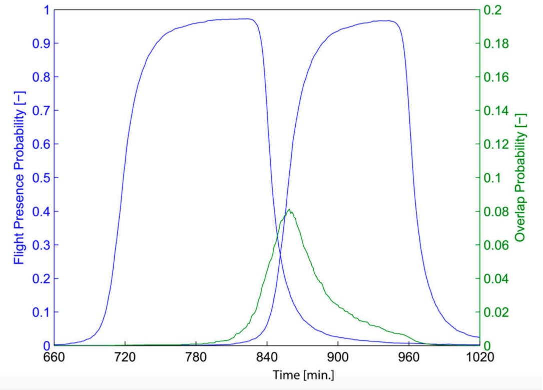


Figure 3.4: Example of two sequential presence probabilities (Schaijk and Visser 2017)

Proposed innovation

Despite the fact that the results of both researches are positive, a critical note should be placed at the construction of the presence probability curves around the STA and the STD. These fittings are a rather simplistic approach; they only distinct the flights with two factors (airline and region) and they are based on historical data fitting instead of actual predictions. The authors of both studies acknowledge this. As stated in (Schaijk and Visser 2017): *'Future research will also need to focus on the development of an improved regression model to predict flight presence probabilities distributions'*.

This research aims to improve the predictions of the flight presence probabilities distributions around the STA and STD by introducing machine learning algorithms. In section 3.3, three machine learning algorithms have been identified that are able to predict a probability histogram, or even a complete probability density of flight delays. Assuming that the algorithms are successful, a probability density or histogram centering around a peak at approximately 0 minutes delay is expected, predicted for individual flights. By taking the cumulative of this density or histogram and placing the 0 minute delay point at the scheduled time of arrival, the cumulative presence of the incoming flight in figure 3.3 is derived completely by machine learning. A similar approach can be applied to the outgoing flight.

This proposed innovation allows the complete presence curve to be an actual prediction, based on more than two input features. To verify whether these newly predicted probability functions perform well, two flight to gate assignments will be generated. One that is based on the deterministic approach that handles flight presence as a binary, and one schedule that includes the flight delay probability prediction just described. By simulating arrival and departure times of the scheduled flights, according to their delay distributions, the two schedules are compared on their efficiency (i.e. how tight flights are scheduled) and robustness (i.e. the ability to absorb delays).

4 Conclusion

As stated in the introduction, the first goal of this report is to understand the state of the art in the field of flight delay predictions. After an extensive literature review, it can be concluded that most binary flight delay classification problems apply either a bagging or boosting extension of the Decision Tree model or a Neural Network. In combination with flight schedule features and possibly weather and/or reactionary features, achieved accuracies range from around 80% to accuracies above 90%. It should be noted that these good results are almost always achieved for short, same day, prediction horizons.

One limitation of the existing literature is that almost all case studies in literature evolve around large international airports. None of the encountered researches specifically focus on a regional airport. This is understandable considering that regional airports cannot provide the large databases required for certain machine learning algorithms. Nevertheless, regional airports could also benefit from better flight delay predictions. A potential solution is the theory of transfer learning, which allows smaller airports to train their model on additional data from either a very similar airport or a much larger airport with more data.

Another major limitation of the existing literature is the prediction target, which is always a point estimate in the form of a delay class or the number of minutes. None of the articles focus on estimating a confidence interval or probability density function, even though airport operations optimization problems could benefit from it. A potential solution is found in the fields of weather forecasting and power load estimations. Adding a Soft-max Regression or Mixture Density extension to a Deep Neural Network should allow for the prediction of a probability histogram or density respectively for the delay of an individual flight.

The second goal stated in the introduction is to identify in detail how these probabilistic flight delay predictions can be incorporated in operation optimizations. A potential airport operation optimization problem is an existing Flight-to-Gate Assignment Problem (FGAP). This linear problem currently incorporates presence probability functions that are based on a rather simplistic regression method. By replacing these simplistic probability functions with probability densities predicted by machine learning algorithms, the overall performance of the schedule is expected to be improved.

The overall conclusion of this report is that despite the identified challenges, integrating a probabilistic flight delay forecast into an airport operation optimization problem is theoretically valuable and possible. Further research has to verify whether the theories and assumption derived in this literature review work in practice.

Bibliography

[1] H. Alonso and A. Loureiro. “Predicting Flight Departure Delay at Porto Airport: A Preliminary Study”. In: *Proceedings of the 7th International Joint Conference on Computational Intelligence (IJCCI 2015)* 3 (2015), pp. 93–98.

[2] C.M. Bishop. *Mixture Density Networks.* Aston University, Birmingham, 1994.

[3] L. Breiman. “Random Forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.

[4] N. V. Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[5] J. Chen and M Li. “Chained Predictions of Flight Delay Using Machine Learning”. In: *AIAA Scitech 2019 Forum* (2019), p. 1661.

[6] S. Choi et al. “Cost-sensitive Prediction of Airline Delays Using Machine Learning”. In: *IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)* (2017), pp. 1–8.

[7] S. Choi et al. “Prediction of Weather-induced Airline Delays Based on Machine Learning Algorithms”. In: *IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)* (2016), pp. 1–6.

[8] P. M. Granitto et al. “Recursive feature elimination with random forest for PTR-MS analysis of agroindustrial products”. In: *Chemometrics and Intelligent Laboratory Systems* 83.2 (2006), pp. 83–90.

[9] G. Gui et al. “Flight Delay Prediction Based on Aviation Big Data and Machine Learning”. In: *IEEE Transactions on vehicular technology* 69.1 (2020), pp. 140–150.

[10] H. Hersbach. “Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems”. In: *Weather and Forecasting* 15 (2000), pp. 559–570.

[11] Y. Horiguchi et al. “Predicting Fuel Consumption and Flight Delays for Low-Cost Airlines”. In: *Proceedings of the Twenty-Ninth AAAI Conference on Innovative Applications* (2017), pp. 4686–4693.

[12] M. Jiang et al. “Text classification based on deep belief network and softmax regression”. In: *Neural Computating* & *Applications* 29 (2018), pp. 61–70.

[13] G. Ke et al. “LightGBM: A Highly Efficient Gradient Boosting Decision Tree”. In: *31st Conference on Neural Information Processing Systems (NIPS)* 1 (2017), pp. 3147–3155.

[14] S. Khanmohammadi, S. Tutun, and Y. Kucuk. “A New Multilevel Input Layer Artificial Neural Network for Predicting Flight Delays at JFK Airport”. In: *Procedia Computer Science* 95 (2016), pp. 237–244.

[15] Y.J. Kim et al. “A Deep Learning Approach to Flight Delay Prediction”. In: *IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)* (2016), pp. 1–6.

[16] A. Klein. “Airport delay prediction using weather-impacted traffic index (WITI) model”. In: *Digital Avionics Systems Conference (DASC), IEEE/AIAA 29th* (2010), 2–B.

[17] J. LOrtye. “Robust Flight-to-Gate Assignment Planning with Airside and Landside Constraints”. Delft University of Technology, 2019.

[18] M. Lambelho. “Ranking pre-season flight schedules at an airport using a machine learning approach”. Delft University of Technology, 2019.

[19] M. Lambelho et al. “Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions”. In: *Journal of Air Transport Management* 82 (2020).

[20] S.M. Lundberg and S-I Lee. “A Unified Approach to Interpreting Model Predictions”. In: *31st Conference on Neural Information Processing Systems (NIPS)* (2017).

[21] S. Manna et al. “A Statistical Approach to Predict Flight Delay Using Gradient Boosted Decision Tree”. In: *2017 International Conference on Computational Intelligence in Data Science (ICCIDS)* (2017), pp. 1–5.

[22] N. McCarthy, M. Karzand, and F. Lecue. “Amsterdam to Dublin Eventually Delayed? LSTM and Transfer Learning for Predicting Delays of Low Cost Airlines”. In: *The Thirty-First AAAI Conference on Innovative Applications of Artificial Intelligence (IAAI-19)* (2019).

[23] S. Moon and Jaime Carbonell. “Completely Heterogeneous Transfer Learning with Attention - What And What Not To Transfer”. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI.* 2017, pp. 2508–2514.

[24] E.R. Mueller and G.B. Chatterij. “Analysis of aircraft arrival and departure delay characteristics”. In: *IAA aircraft technology, integration and operations (ATIO) conference* (2002).

[25] R de Neufville and A. Odoni. *Airport Systems.* New York City, New York: McGraw-Hill Education, 2013.

[26] A. Niculescu-Mizil and R. Caruana. “Predicting Good Probabilities With Supervised Learning”. In: *Proceedings of the 22nd International Conference on Machine Learning* (2006), pp. 625–632.

[27] M. Raeis, A. Tizghadam, and A. Leon-Garcia. “Real-Time Prediction of Delay Distribution in Service Systems using Mixture Density Networks”. In: *15th International Conference on Network and Service Management (CNSM)* (2019), pp. 1–6.

[28] J.J. Rebollo and H. Balakrishnan. “Characterization and prediction of air traffic delays”. In: *Transportation research part C: Emerging technologies* 44 (2014), pp. 231–241.

[29] O.R.P. van Schaijk and H.G. Visser. “Robust flight-to-gate assignment using flight presence probabilities”. In: *Transportation Planning and Technology* 40.8 (2017), pp. 928–945.

[30] W. Shao et al. “Flight Delay Prediction using Airport Situational Awareness Map”. In: *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (2019).

[31] B. Thiagarajan et al. “A Machine Learning Approach for Prediction of On-time Performance of Flights”. In: *IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)* (2017), pp. 1–6.

[32] J. Vossen, B. Feron, and A. Monti. “Probabilistic Forecasting of Household Electrical Load Using Artificial Neural Networks”. In: *IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)* (2018), pp. 1–6.

[33] T. Xie, G. Peng, and H. Wang. “Interval Construction and Optimization for Mechanical Property Forecasting with Improved Neural Networks”. In: *Advances in Computational Intelligence Systems* (2020), pp. 223–234.

[34] N. Xu et al. “Estimation of delay propagation in the national aviation system using Bayesian networks”. In: *6th USA/Europe Air Traffic Management Research and Development Seminar* (2005).

[35] B. Yu et al. “Flight delay prediction for commercial air transport: A deep learning approach”. In: *Transportation Research Part E* 125 (2019), pp. 203–221.

α

Literature study features overview

Reference	Flight schedule features	Weather features	Other features
(Gui et al. 2020)	airport name, day of month, day of week, flight number, ICAO code, origin, destination, month, scheduled arrival/departure time, season	weather condition, wind direction, wind speed	traffic flow of air route
(Lambelho et al. 2020)	aircraft, **airline**, airport, country, day of month, day of week, day of year, distance, **hour**, month, **seats**, year, terminal		**arrival ATFM delay**
(Chen and Li 2019)	day of month, day of week, scheduled arrival/departure time, scheduled elapsed time	dew point temperature, dry and wet bulb temperature, hourly visibility, present weather type, relative humidity, station pressure, wind speed and gust	**arrival delay group** , **LAAD group**
(McCarthy et al. 2019)	actual arrival/departure time, destination airport, flight date, origin airport, time to next scheduled departure, scheduled arrival time, scheduled departure time	-	arrival time previous flight, turnaround time, passenger and bags information
(Shao et al. 2019)	call-sign, scheduled arrival/departure time	daily weather observations: temperature, humidity, wind directions, wind speed, air pressure	**airport GPS trajectories**, cause of delay (5 cat), scheduled/actual arrival time for each aircrafts previous flight
(Yu et al. 2019)	aircraft capacity, flight terminal, number of passengers, origin or pass-by flight	**air route situation**	airline properties, **airport crowdedness**, closing time of gate, **delay of previous flight**, gap between check-in time and scheduled departure time, ready time of shuttles or jet bridge
(Choi et al. 2017)	day of month, day of week, destination, origin, month, quarter of year, scheduled arrival/departure time (local)	METAR	arrival delay indicator

(Horiguchi et al. 2017)	arrival airport, day of week, departure year, flight air frame id, month, **scheduled arrival/departure time**, year	-	reservation data (for 1 week horizon), estimated en-route time, **passenger data**, **scheduled fuel**, standby position ID (all for 1 day horizon)
(Manna et al. 2017)	actual time of departure, carrier, day of week, destination, origin, scheduled departure time	-	-
(Thiagarajan et al. 2017)	airline ID, day of month, destination airport ID, origin airport ID, flight Number, month, quarter of year, scheduled arrival/departure time, year	cloud cover, dew point, humidity, precipitation, pressure, temperature, time of observation, visibility, weather code, wind (chill, direction, gust, speed)	-
(Choi et al. 2016)	day of month, day of week, destination, origin, month, quarter of year, scheduled arrival/departure time (local)	METAR (training), forecast (prediction)	arrival delay indicator
(Khanmohammadi et al. 2016)	actual arrival/departure time, day of month, day of week, ID code of origin, scheduled arrival/departure time	-	reason for arrival delay (carrier, weather, NAS, security, late arrival), delay at JFK, delay at origin airport
(Kim et al. 2016)	date, day of week, destination, month, origin, season, scheduled arrival/departure time	daily average METAR, hourly METAR	daily delay status OD
(Alonso and Loureiro 2015)	aircraft type, airline, day, destination, hour, month, predicted weekday, origin	meteorological conditions	aircraft parking stand, arrival delay (in minutes), ground operation time in minutes, take-off runway
(Rebollo and Balakrishnan 2014)	aircraft tail number, carrier codes, destination, month of year, origin, season, scheduled and actual gate in and wheels-off time, time of day	NAS delay state, type of delay day, and previous days type (all include weather)	-
(Klein 2010)	scheduled traffic	Weather Information and Traffic Index (WITI), WITI-FA forecasts (in categories en-route, terminal and queuing)	-
(Xu et al. 2005)	flight number, time	VMC or IMC (binary), actual and predicted over 1:45 h	airport arrival cancellations, airport arrival and departure delay, departure delay previous airport to this airport
(Mueller and Chatterij 2002)	airport code, actual arrival/departure time, date of departure, identification code, scheduled arrival/departure time	-	-

[1] The features in bold are identified by the research as most important for flight delay predictions

Table α.1: Literature overview: all features

III

Supporting work

A

Additional results binary classification

A.1. Confusion matrices

The metric scores in table 7 of the scientific paper are based upon the confusion matrices in tables A.1 - A.8.

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	2216.2	356.0
	Delayed	630.4	260.8

Table A.1: Results random forest - 1D arrivals

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	2614.6	459.4
	Delayed	232.0	157.4

Table A.2: Results LightGBM - 1D arrivals

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	1629.4	500.0
	Delayed	631.2	711.6

Table A.3: Results random forest - 1D departures

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	1816.0	583.8
	Delayed	444.6	627.8

Table A.4: Results LightGBM - 1D departures

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	2066.4	292.4
	Delayed	783.4	325.0

Table A.5: Results random forest - 1M arrivals

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	2605.0	460.2
	Delayed	244.8	157.2

Table A.6: Results LightGBM - 1M arrivals

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	1626.0	496.4
	Delayed	635.0	715.6

Table A.7: Results random forest - 1M departures

		Predicted	
		Not delayed	Delayed
Actual	Not delayed	1825.6	581.4
	Delayed	435.4	630.6

Table A.8: Results LightGBM - 1M departures

A.2. Pearson correlation matrices - all features

The Pearson correlation matrices, based on all available features, are given in figure A1 and A2 for the arriving and departing flights respectively, and for a prediction horizon of one day (1D). The correlations of the features available at a month (1M) in advance are very similar. The selection procedure intends to remove features with an absolute correlation of at least 0.8, which are indicated in the matrices with their value.

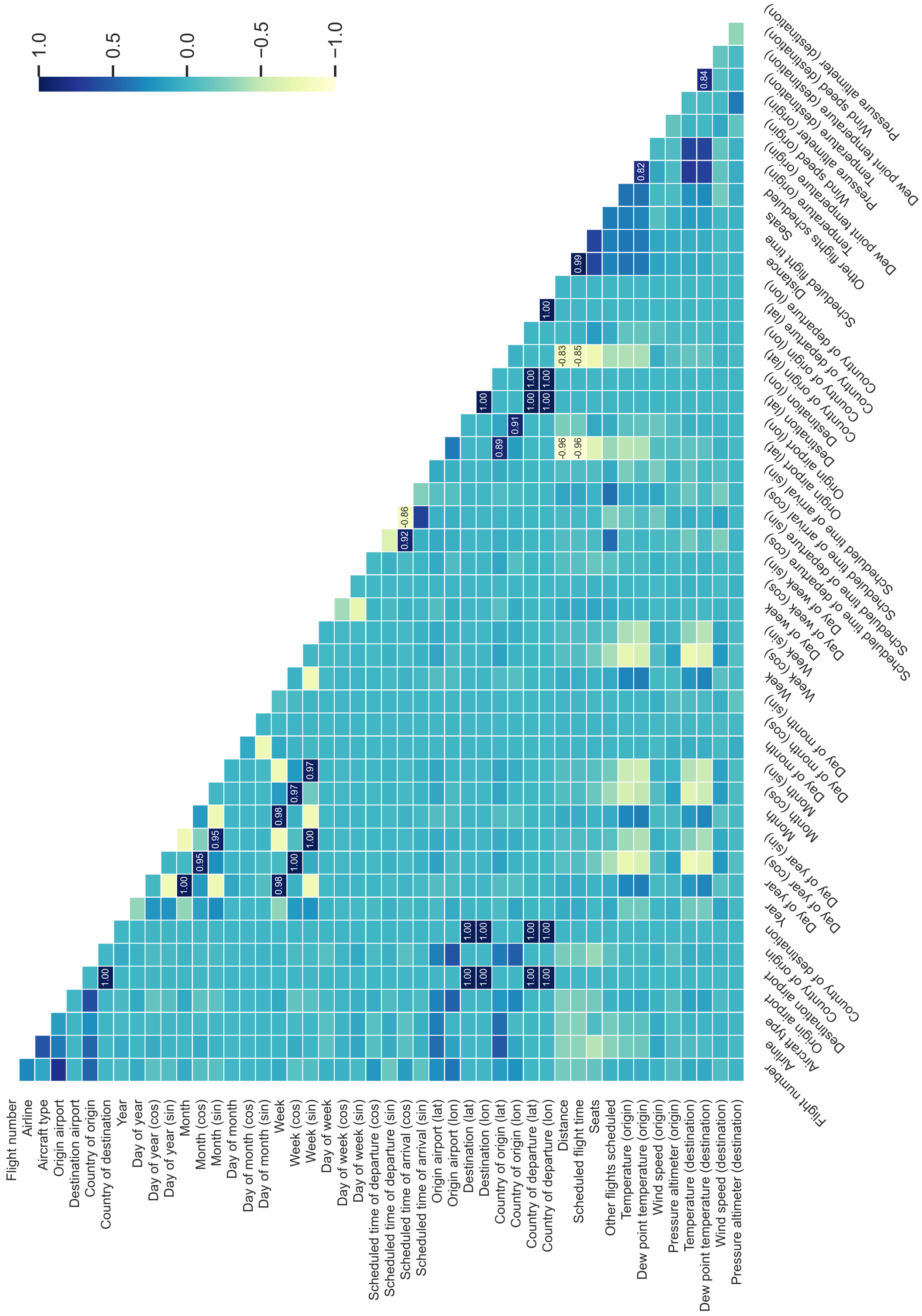


Figure A.1: Pearson correlation matrix 1D arrivals - all available features

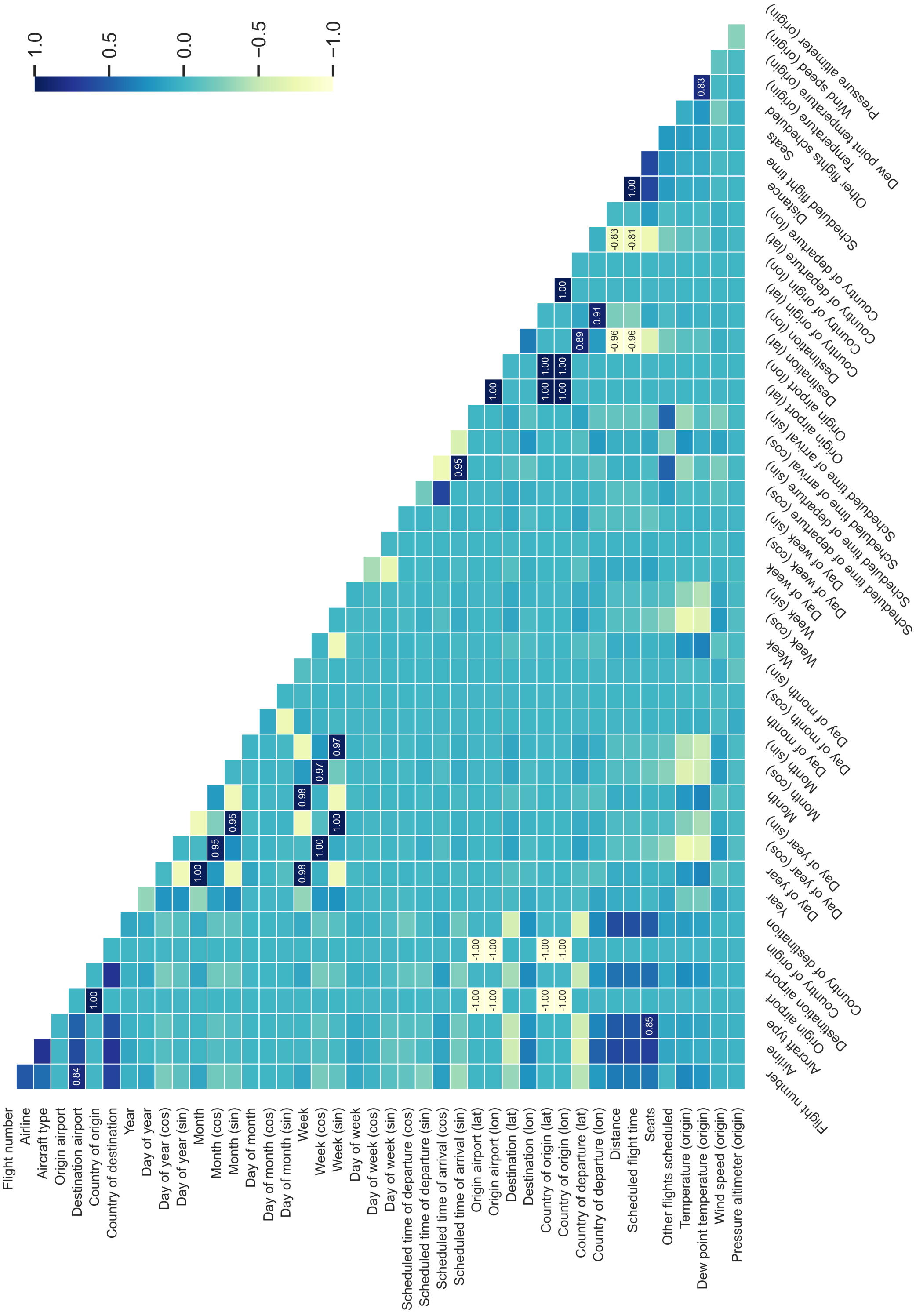


Figure A.2: Pearson correlation matrix 1D departures - all available features

A.3. Pearson correlation matrices - selected features

After the selection procedure described in the research paper, the 1D correlation matrices are reduced to the matrices in figure A.3 and A.4. Note that features that consist of two components, such as longitude and latitude or cosine and sine, are only removed if both features are highly correlated with a third one, which explains the occasional score above 0.8 or below -0.8.

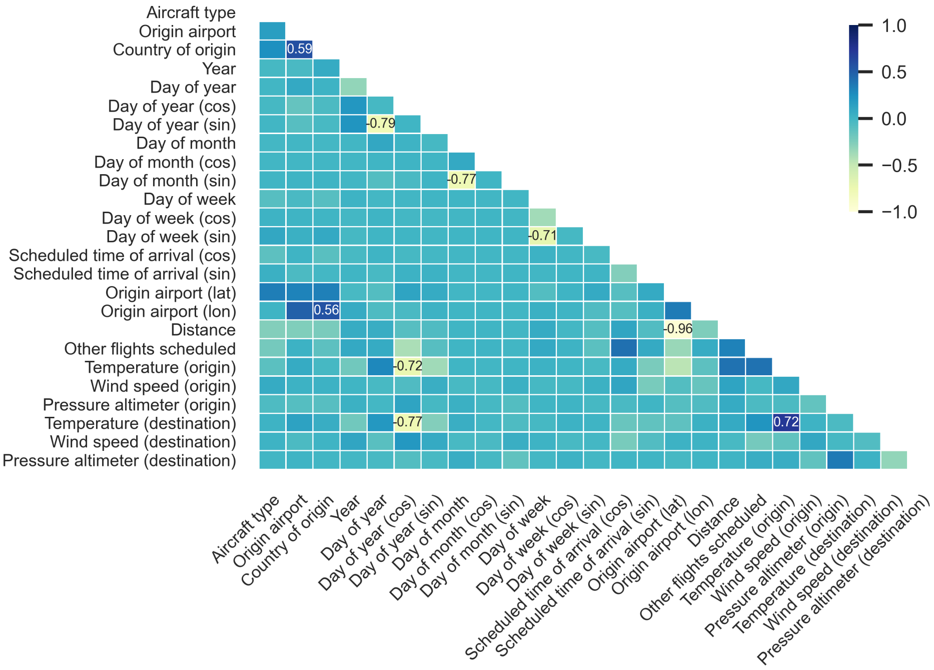


Figure A.3: Pearson correlation matrix 1D arrivals - selected features

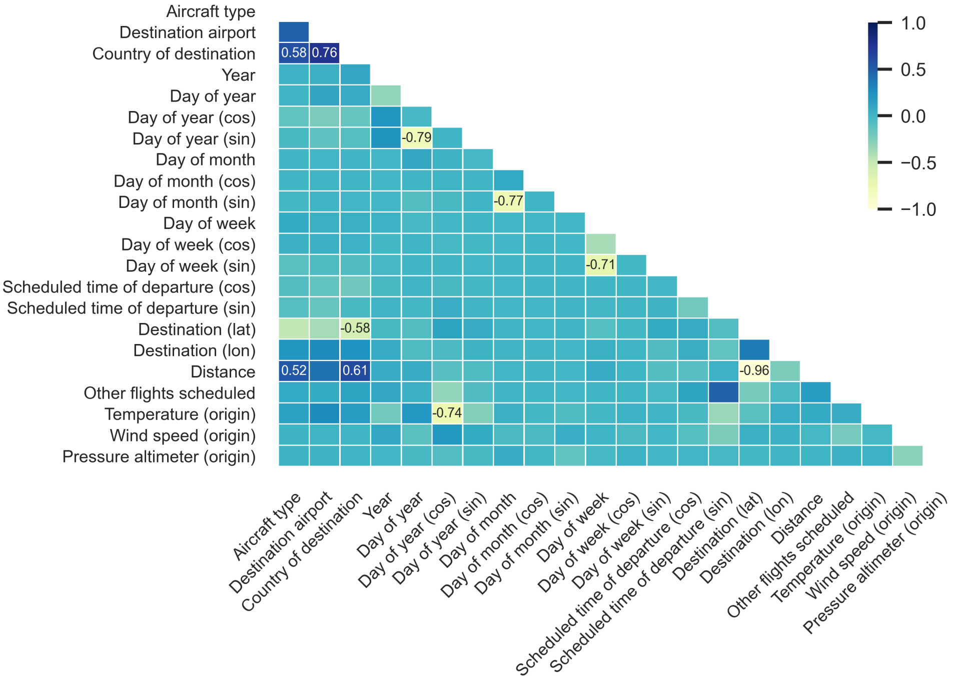


Figure A.4: Pearson correlation matrix 1D departures - selected features

B
Hyperparameter tuning

The aim of hyperparameter tuning is to find appropriate settings for each of the machine learning models described in the research paper. Since the data of this research are highly imbalanced, the tuning focuses on improving and balancing the recall and precision, in the form of optimizing the f1 score. It should be emphasized, however, that the goal of this tuning process is to improve the default settings. Finding the perfect settings is a time consuming task that falls outside the scope of this research.

Tuning process

To tune the models, three options are considered: an extensive grid search, a random search, and a Bayesian optimization-based grid search. All methods are based on a certain parameter search space, which specifies the possible values of each parameter, as determined by the user. In an extensive grid search, all possible combinations of parameter settings from the search space are tested, and the best performing setting is selected. Although thorough, this process is very time consuming. The second option is to randomly draw a number of parameter settings from the search grid and select the best performing one. This method is more time efficient, but less thorough.

The third possibility is a Bayesian optimization-based grid search, executed with the `hyperopt` package in python. Different than the extensive and random grid search, this model updates the parameter settings sequentially by using the results of the previous settings to determine the next setting. In order to do this, all parameters are represented by a probability distribution. This research uses the continuous uniform distribution, which is defined as:

$$f(x) = \begin{cases} \frac{1}{b-a}, & \text{for } a \leq x \leq b. \\ 0, & \text{otherwise.} \end{cases} \tag{B.1}$$

or its discrete variant, depending on the parameter. Furthermore, the approach requires an objective function, which in this case is the negative f_1 score, as the objective is always minimized. For verification purposes, all machine learning models used in this research are tuned twice; once with a random search and with a Bayesian search. The best performing setting is selected, where a distinction is made between the settings for the arrivals and for the departures.

Search space

The search spaces selected for the machine learning models used in this research, are listed in tables B.1 - B.5.

Parameter	Probability distribution
Estimators	Discrete Uniform (10, 2500)
Max depth	Discrete Uniform (3, 70)
Min samples leaf	Discrete Uniform (1, 100)
Max features	Uniform (0.3, 1)
Criterion	[Gini, Entropy]

Table B.1: Parameter search space RF Classifier

Parameter	Probability distribution
Estimators	Discrete Uniform (10, 5000)
Max depth	Discrete Uniform (3, 60)
Min child weight	Uniform (0.01, 200)
Subsample	Uniform (0.4, 1)
Number of leaves	Discrete Uniform (8, 900)
Learning rate	Uniform (0.0001, 0.3)

Table B.2: Parameter search space LightGBM

Parameter	Probability distribution
Estimators	Discrete Uniform (10, 2500)
Max depth	Discrete Uniform (3, 70)
Min samples leaf	Discrete Uniform (1, 100)
Max features	Uniform (0.3, 1)
Criterion	[MSE, MAE]

Table B.3: Parameter search space RF Regressor

Parameter	Probability distribution
Gaussians	Discrete Uniform (1, 10)
Neurons	Discrete Uniform (10, 250)
Hidden layers	Discrete Uniform (2, 4)
Dropout rate	Uniform (0.01, 0.3)
Learning rate	Uniform (1 x 10^{-6}, 0.3)
Activation	[sigmoid, tanh, ReLU]
Optimizer	[Adagrad, Adadelta, Adam]
Batch size	[10, 100, 500, 1000, 2000]

Table B.4: Parameter search space Mixture Density Network

Parameter	Probability distribution
Neurons	Discrete Uniform (10, 250)
Hidden layers	Discrete Uniform (2, 4)
Dropout rate	Uniform (0.01, 0.9)
Learning rate	Uniform (1 x 10^{-6}, 0.3)
Activation	[sigmoid, tanh, ReLU]
Optimizer	[Adagrad, Adadelta, Adam]
Batch size	[10, 100, 500, 1000, 2000]

Table B.5: Parameter search space Dropout Network

Hyperparameter settings

The search spaces selected for the machine learning models used in this research, are listed in tables B.6 - B.10.

	Features	Estimators	Max depth	Min samples leaf	Max Features	Criterion
Departures	22	625	5	1	0.73	entropy
Arrivals	25	1680	5	60	0.37	gini

Table B.6: Hyperparameter settings RF Classifier

	Features	Estimators	Max depth	Min child weight	Subsample	Number of leaves	Learning rate
Departures	22	2535	28	62	0.76	96	0.14
Arrivals	25	4876	44	128	0.86	499	0.28

Table B.7: Hyperparameter settings LightGBM

	Features	Estimators	Max depth	Min samples leaf	Max Features	Criterion
Departures	22	1975	60	3	0.36	MSE
Arrivals	25	1975	60	3	0.36	MSE

Table B.8: Hyperparameter settings RF Regressor

	Features	Gauss.[1]	Neurons	Hidden layers	Dropout rate	Learning rate	Activa-tion	Opti-mizer	Batch size
Departures	22	6	100→100	2	0.05	0.0001	ReLU	Adam	100
Arrivals	25	6	100→100	2	0.05	0.0001	ReLU	Adam	100

[1] Gauss.: number of Gaussians

Table B.9: Hyperparameter settings Mixture Density Network

	Features	Est.[1]	Neurons	Hidden layers	Dropout rate	Learning rate	Activa-tion	Opti-mizer	Batch size
Departures	22	1975	14→14	2	0.479	0.085	ReLU	Adadelta	100
Arrivals	25	1975	43→43	3	0.838	0.092	tanh	Adam	100

[1] Est.: number of estimates

Table B.10: Hyperparameter settings Dropout Network

Tuning evaluation

An import setting for the Neural Networks is the number of epochs, which is the number of times that the entire dataset is passed through the network. Setting the value too low might result in an underfitted model, setting the value too high might result in an overfitted one. Rather than setting the number beforehand, this research applies early stopping. The model is validated on 20% of the training data. If the validation test results do not improve for five iterations in a row, the training of the model is stopped. The resulting numbers of epochs are listed in table B.11. This table also gives the running times of the delay distribution predicting models. The evaluation shows that the Mixture Density network is the fastest one, followed by the Dropout Neural Network and finally the RF Regressor.

Model	Max epoch 1D Departures	Max epoch 1D Arrivals	Run time (5-folds)
RF Regressor	-	-	~ 60 min.
Mixture Density Network	321	229	~ 10 min.
Dropout Network	414	67	~ 30 min.

Table B.11: Epochs and running times of the delay distribution predicting models

C

Transfer learning

Anticipating a shortage of historical data, this research attempted to enlarge the training dataset of Rotterdam The Hague Airport by adding historical data of Eindhoven Airport (EIN), a similar regional airport that is also located in the Netherlands. This concept of adding data from one subject to the training data of another, is known as transfer learning. For this particular study, transfer learning was not beneficial. Adding EIN data to the training dataset of RTM did not improve the performance of any model by much, as can be seen in tables C.1 and C.2. All results in the research paper are based on RTM data only. The results of the training set with EIN data, are added to this appendix for comparison and completeness.

C.1. Binary classification results

		RF Classifier		**LightGBM**	
Classifier	**Metric**	**Mean**	**SD**	**Mean**	**SD**
1M Departures	**accuracy**	**0.681**	**7.4×10^{-3}**	**0.709**	**4.5×10^{-3}**
	precision	0.547	1.2×10^{-2}	0.596	1.2×10^{-2}
	recall	0.510	2.7×10^{-2}	0.520	1.2×10^{-2}
	f1	0.528	1.4×10^{-2}	0.555	3.5×10^{-3}
	AUC	0.642	9.1×10^{-3}	0.666	2.4×10^{-3}
1M Arrivals	**accuracy**	**0.657**	**9.5×10^{-3}**	**0.798**	**8.1×10^{-3}**
	precision	0.268	1.7×10^{-2}	0.401	4.2×10^{-3}
	recall	0.535	1.6×10^{-2}	0.268	1.7×10^{-2}
	f1	0.357	1.6×10^{-2}	0.321	1.2×10^{-3}
	AUC	0.609	7.9×10^{-3}	0.591	6.6×10^{-3}
1D Departures	**accuracy**	**0.682**	**5.2×10^{-3}**	**0.708**	**8.0×10^{-3}**
	precision	0.551	2.5×10^{-2}	0.595	3.0×10^{-2}
	recall	0.502	2.7×10^{-2}	0.517	1.2×10^{-2}
	f1	0.524	8.3×10^{-3}	0.553	1.6×10^{-2}
	AUC	0.641	2.3×10^{-3}	0.664	9.5×10^{-3}
1D Arrivals	**accuracy**	**0.703**	**7.7×10^{-3}**	**0.803**	**7.0×10^{-3}**
	precision	0.278	2.4×10^{-2}	0.415	3.5×10^{-2}
	recall	0.415	1.0×10^{-2}	0.259	1.2×10^{-2}
	f1	0.332	1.7×10^{-2}	0.319	1.8×10^{-2}
	AUC	0.590	5.9×10^{-3}	0.590	7.5×10^{-3}

Table C.1: Results binary classification with EIN data

C.2. Probabilistic forecasting results

		RF Regressor		Mixture Network		Dropout Network	
Classifier	**Metric**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**
1D Departures	RMSE	24.565	8.7×10^{-1}	25.412	9.9×10^{-1}	25.837	1.0×10^{0}
	MSE	604.222	4.3×10^{1}	646.740	5.1×10^{1}	668.560	5.3×10^{1}
	MAE	12.603	2.5×10^{-1}	12.721	3.1×10^{-1}	13.678	1.9×10^{-1}
	R^2	0.141	1.5×10^{-2}	0.081	1.1×10^{-2}	0.050	2.5×10^{-3}
	max. err.	426.285	5.4×10^{1}	431.709	6.1×10^{1}	431.172	6.1×10^{1}
	s_{avg}	16.656	9.7×10^{-2}	24.266	3.4×10^{-1}	4.687	5.8×10^{-2}
	FOS	0.841	4.2×10^{-3}	0.922	1.6×10^{-3}	0.208	1.1×10^{-2}
	CRPS	**8.638**	$\mathbf{2.3 \times 10^{-1}}$	**9.028**	$\mathbf{2.3 \times 10^{-1}}$	**11.522**	$\mathbf{1.8 \times 10^{-1}}$
1D Arrivals	RMSE	25.936	1.2×10^{0}	26.764	1.2×10^{0}	28.051	1.1×10^{0}
	MSE	674.002	5.9×10^{1}	717.679	6.3×10^{1}	788.116	6.2×10^{1}
	MAE	14.975	2.5×10^{-1}	15.262	3.2×10^{-1}	15.956	3.3×10^{-1}
	R^2	0.139	1.5×10^{-2}	0.083	9.7×10^{-3}	-0.007	4.4×10^{-3}
	max. err.	395.482	6.2×10^{1}	414.829	5.4×10^{1}	414.013	5.4×10^{1}
	s_{avg}	19.221	1.8×10^{-1}	24.323	4.5×10^{-1}	3.068	2.8×10^{-1}
	FOS	0.785	6.0×10^{-3}	0.861	1.0×10^{-2}	0.150	1.7×10^{-2}
	CRPS	**10.587**	$\mathbf{2.8 \times 10^{-1}}$	**11.057**	$\mathbf{3.2 \times 10^{-1}}$	**14.477**	$\mathbf{3.2 \times 10^{-1}}$

Table C.2: Results probabilistic forecasting with EIN data

D

Example flights information

The research paper presents two examples of the results, based on actual flights from the database. This chapter present more detailed information about the selected flights. Section D.1 lists the flights used for figure 5 of the paper, section D.2 presents a list of the flight pairs used for figure 7.

D.1. Example flights probabilistic forecasting results

Legend	STA	Flight number	Airline	Origin	Destination	Aircraft
2288	2019-05-05 11:15:00	HV5068	Transavia	GRO	RTM	B738
2291	2019-05-05 22:35:00	HV5008	Transavia	DBV	RTM	B737
2274	2019-05-02 15:05:00	BA4455	British Airways	LCY	RTM	E190
2284	2019-05-04 13:00:00	PC1261	Pegasus Airlines	SAW	RTM	A20N

Table D.1: An overview of the arriving flights used for figure 5 of the scientific paper

Legend	STD	Flight number	Airline	Origin	Destination	Aircraft
2292	2019-05-03 06:55:00	HV6061	Transavia	RTM	BCN	B737
2301	2019-05-03 18:40:00	HV5293	Transavia	RTM	VIE	B737
2283	2019-05-01 10:50:00	BA4454	British Airways	RTM	LCY	E190
2317	2019-05-06 14:35:00	PC1262	Pegasus Airlines	RTM	SAW	A20N

Table D.2: An overview of the departing flights used for figure 5 of the scientific paper

D.2. Example flights gate scheduling

	Arrival							Departure					
Flight	STA	Number	Airline	Aircraft	Origin	Destination	Registration	STD	Number	Airline	Aircraft	Origin	Destination
1	-	-	-	-	-	-	-	06:55	HV5051	Transavia	B737	RTM	ALC
2	-	-	-	-	-	-	-	07:00	HV6259	Transavia	B738	RTM	SPU
3	-	-	-	-	-	-	-	07:00	HV5021	Transavia	B737	RTM	AGP
4	-	-	-	-	-	-	-	07:10	HV6081	Transavia	B737	RTM	EGC
5	-	-	-	-	-	-	-	07:10	HV6285	Transavia	B738	RTM	TLN
6	-	-	-	-	-	-	-	07:15	HV6441	Transavia	B737	RTM	VLC
7	-	-	-	-	-	-	-	07:50	HV6093	Transavia	B737	RTM	FAO
8	-	-	-	-	-	-	-	10:10	OR183	TUIfly	B738	RTM	RHO
9	10:45	HV6082	Transavia	B737	EGC	RTM	PH-BGO	11:30	HV5007	Transavia	B737	RTM	DBV
10	10:45	HV5690	Transavia	B738	IBZ	RTM	PH-HSM	11:30	HV6091	Transavia	B738	RTM	FAO
11	11:10	HV6192	Transavia	B738	PMI	RTM	PH-HZG	11:55	HV6191	Transavia	B738	RTM	PMI
12	11:30	HV6286	Transavia	B738	TLN	RTM	PH-HXI	12:15	HV5987	Transavia	B738	RTM	MPL
13	11:50	HV6260	Transavia	B738	SPU	RTM	PH-HZL	12:35	HV6261	Transavia	B738	RTM	SPU
14	12:15	HV5052	Transavia	B737	ALC	RTM	PH-XRY	12:55	HV6063	Transavia	B737	RTM	BCN
15	12:30	HV6442	Transavia	B737	VLC	RTM	PH-XRD	13:20	HV5121	Transavia	B737	RTM	ACE
16	13:00	PC1261	Pegasus Airlines	A20N	SAW	RTM	TC-NBK	13:40	PC1262	Pegasus Airlines	A20N	RTM	SAW
17	-	-	-	-	-	-	-	14:05	HV5369	Transavia	B737	RTM	HER
18	15:15	BA4455	British Airways	E190	LCY	RTM	G-LCYL	15:45	BA4456	British Airways	E190	RTM	LCY
19	14:20	HV6094	Transavia	B737	FAO	RTM	PH-BGL	16:00	HV5243	Transavia	B737	RTM	LIS
20	-	-	-	-	-	-	-	16:10	HV5023	Transavia	B737	RTM	AGP
21	16:15	HV5988	Transavia	B738	MPL	RTM	PH-HXI	17:00	HV5053	Transavia	B738	RTM	ALC
22	16:50	HV5008	Transavia	B737	DBV	RTM	PH-BGO	17:40	HV6037	Transavia	B737	RTM	FCO
23	17:30	HV6262	Transavia	B738	SPU	RTM	PH-HZL	18:30	HV5997	Transavia	B738	RTM	PUY
24	17:55	HV6064	Transavia	B737	BCN	RTM	PH-XRY	18:35	HV5067	Transavia	B737	RTM	GRO
25	18:00	HV6092	Transavia	B738	FAO	RTM	PH-HSM	18:45	HV5689	Transavia	B738	RTM	IBZ
26	18:25	OR184	TUIfly	B738	RHO	RTM	CS-TQU	-	-	-	-	-	-
27	21:35	BA4459	British Airways	E190	LCY	RTM	G-LCYZ	-	-	-	-	-	-
28	22:00	HV5370	Transavia	B737	HER	RTM	PH-XRV	-	-	-	-	-	-
29	22:25	HV5244	Transavia	B737	LIS	RTM	PH-BGL	-	-	-	-	-	-
30	22:35	HV5024	Transavia	B737	AGP	RTM	PH-XRC	-	-	-	-	-	-
31	22:45	HV5054	Transavia	B738	ALC	RTM	PH-HXI	-	-	-	-	-	-
32	22:45	HV5998	Transavia	B738	PUY	RTM	PH-HZL	-	-	-	-	-	-
33	22:45	HV5122	Transavia	B737	ACE	RTM	PH-XRD	-	-	-	-	-	-
34	22:55	HV6038	Transavia	B737	FCO	RTM	PH-BGO	-	-	-	-	-	-
35	22:55	HV5068	Transavia	B737	GRO	RTM	PH-XRY	-	-	-	-	-	-

Table D.3: An overview of the flights used for the example FGAP schedules in figure 7 of the scientific paper. The flights are scheduled for RTM airport on 2019-07-14.

E

Additional results FGAP

In the scientific paper, the performances of the standard and probabilistic FGAP model are compared on their robustness and efficiency. The robustness is expressed in the average number of conflicts, the efficiency in the average number of scheduled occupied slots. For different values of overlap probability r, both metrics are averaged over around 60 test days. Since all days are between the 1^{st} of July and the 31^{st} of august 2019, it is assumed that all test days are reasonably similar in terms in terms of how busy the airport is, weather conditions, the destination network, etc.. As a result, it is expected that the resulting metrics are also reasonably similar throughout the test dates.

To verify whether this holds, the resulting metrics are plotted against their timeline. In figure E.1 and E.2, the robustness metric and efficiency metric of the probabilistic FGAP model with $r = 0.9$, the probabilistic FGAP model with $r = 0.1$, and the deterministic FGAP model, are plotted for each day of the case study. Although erratic, both figures show that the metric values do not differ too much between the test dates. This becomes even more clear when filtering the results by taking the 7-day moving average, which is shown in figures E.3 and E.4 for robustness and efficiency respectively. The highest and lowest values are spread across the dates and there is no steep trend in either metric.

These time plots validate that the selected test dates are similar enough to average them, as is done in the research paper. Simultaneously, selecting different dates and different values for probability overlap r is also a first sensitivity analysis, which shows that the model works for different settings.

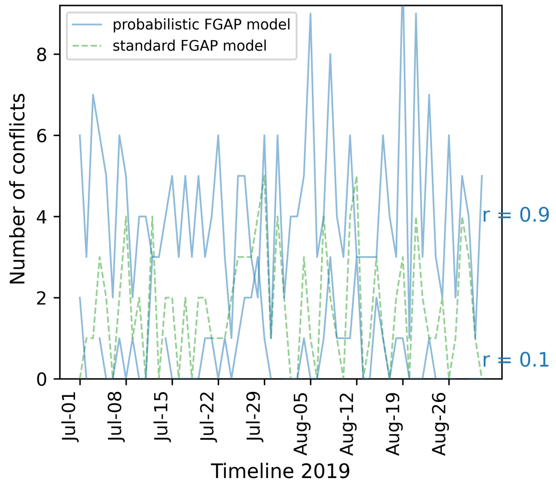


Figure E.1: Robustness metric per day

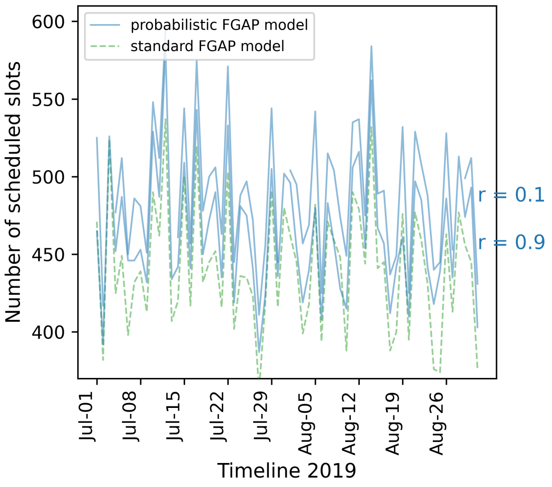


Figure E.2: Efficiency metric per day

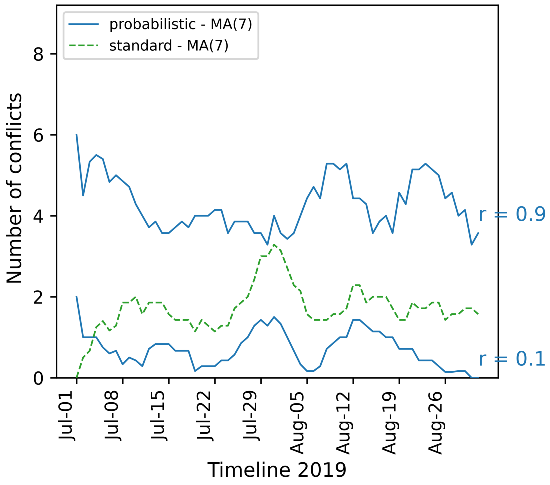


Figure E.3: Robustness metric moving average

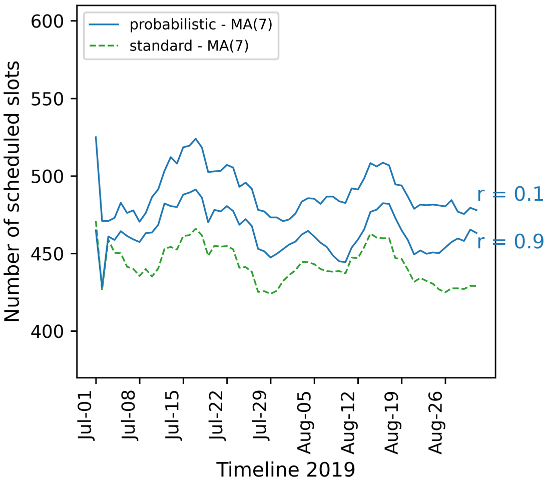


Figure E.4: Efficiency metric moving average