Predicting Probabilistic Flight Delay for Individual Flights using Machine Learning Models

Master of Science Thesis

Laurence Vorage

TU Delft

Predicting Probabilistic Flight Delay for Individual Flights using Machine Learning Models

Master of Science Thesis

by

Laurence Vorage

to obtain the degree of Master of Science
at Delft University of Technology,
to be defended publicly on Tuesday February 23rd, 2021 at 14:00h.

Student number: 4284844
Project duration: June 2020 – February 2021
Thesis committee:

Prof. Dr. D.G. Simons	TU Delft, Chair
Dr. M.A. Mitici	TU Delft, Supervisor
Dr. A. Bombelli	TU Delft, External committee member

TU Delft

Acknowledgements

The document in front of you is the work that concludes my master's degree in Aerospace Engineering at the Delft University of Technology. I have been given the opportunity to work on the future of flight delay predictions by considering them from a probabilistic standpoint using machine learning models. The goal of this research is to provide airport operators with probabilistic flight delay predictions for individual flights, such that they can be used for real-life implementations on airport operations.

During my thesis, I have been able to discover the truly fascinating world of machine learning and artificial intelligence. It has been a challenging and gratifying journey of finding the right models to predict probabilistic flight delay. I enjoyed the process of writing code, testing and debugging, to predict probabilistic flight delay on a large scale of multiple years within achievable computational efforts. I hope that my work can contribute to making aviation more pleasant-full and will result in more robust operational schedules for airports.

This research would not have been possible without my daily supervisor, Mihaela Mitici, from the department of Air Transport and Operations. I would like to thank her for her guidance and for giving me responsibility for my own research. I am grateful for the time Mihaela freed up to provide me with feedback on my work, especially since I know how busy her schedule is.

I would also like to thank Jeffrey Schäfer from Royal Schiphol Group, for providing the essential flight schedule data and expertise on flight operations at Schiphol. Our conversations about the practical side of flight delay at Schiphol were very insightful and encouraging to realize what this research could bring to the future of operational flight planning.

Furthermore, I would like to thank my former housemates of the Kraaiennest, my friends from Buurman, my study friends from Aerospace Engineering and all other friends I made in Delft which made my time as a student in Delft very enjoyable. I would like to thanks my girlfriend Tamar for her never-ending support and keeping me balanced in difficult and stressful times. Last but not least, I would like to thank my family and parents, Radboud and Anneriet as well as my brothers Stefan and Christian, who have supported me during my entire life. I am very grateful for their support and always being there for me.

Laurence Vorage
Delft, February 2021

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Meaning
A	Arrival
B	Binary feature
AMS	Amsterdam Schiphol Airport
ADS-B	Automatic Dependent Surveillance–Broadcast
AUC	Area Under Curve score
A/C	Aircraft
C	Categoricalfeature
CNN	Convolutional Neural Network
CV	Cross Validation
CWC	Coverage Width Criterion
D	Departure
DNN	Deep Neural network
FAA	Federal Aviation Administration
FGAP	Flight To Assignment Problem
FP	False Positive
FN	False Negative
GEV	Generalized Extreme Value
KNN	K Nearest Neighbours
LSTM	Long Short-Term Memory layer
MAE	Mean Absolute Error
MDN	Mixture Density Network
MDNx	Mixture Density model with x components
METAR	METeorological Aerodrome Report
MLE	Maximum Likelihood Estimation
MPIW	Mean Prediction Interval Width
MSE	Mean Squared Error
MVE	Mean Variance Estimator
N	Numerical feature
NN	Neural Network
O-D	Origin-Destination
PI	Prediction Interval
PICP	Prediction Interval Coverage Probability
RF	Random Forest
RMSE	Root Mean Squared Error

RNN	Recurring Neural Network
ROC	Receiver Operating Characteristic curve
SHAP	SHapley Additive exPlanation values
SSE	Sum of Squared Errors
SMOTE	Synthetic Minority Oversampling TEchnique
T	Trigonometric transformed feature
TB	Tree Based model
TP	True Positive
TN	True Negative
XGBoost	Extreme Gradient Boosting

KLM	KLM Royal Dutch Airlines
TRA	Transavia airline
BAW	British Airways
PEK	Beijing Capital International Airport
JFK	John F. Kennedy International Airport

List of Symbols

α	Confidence probability level [-]
α_j	Mixing coefficient of j^{th} component of MDN model [-]
D	Great circle distance [kilometers]
d_i	Delay of sample flight i [minutes]
e_i	Prediction error (residual) [-]
η	Nominal confidence level associated with PIs [-]
λ	Longitudinal coordinate [degrees]
n	Number of samples [-]
ϕ	Lateral coordinate [degrees]
ϕ_j	Conditional density function of j^{th} kernel of MDN model [-]
ϕ_{wind}	Wind direction [degrees]
p_i	Probability of sample i [-]
PI^L	Prediction Interval lower bound [minutes]
PI^U	Prediction Interval upper bound [minutes]
R	Radius of Earth [kilometers]
r_{xy}	Pearson correlation between feature x and y
ρ_τ	Tilted absolute value functions of the sample quantile [-]
$\sigma^2(\vec{x}_i)$	Predicted variance of distribution of sample i [-]
θ	Central angle between tow points on a sphere [degrees]
$\vec{x}_i$	Vector with input features for neural network of sample i [-]
$\bar{x}$	Mean of feature x
$y(\vec{x}_i)$	Predicted mean of distribution of sample i [minutes]
y_i	Actual value of sample i [minutes]
$\hat{y}_i$	Predicted value of sample i [minutes]
$\bar{y}$	Mean of feature y
v_{gust}	Wind gust speed [knots]
$v_{\perp north-south}$	Wind gust component perpendicular to the north-sourth runways of Schiphol [knots]
$z_{\alpha/2}$	Critical value of the $N(0,1)$ normal distribution for $\alpha/2$ [-]

Introduction

The continuous growth of the number of flights each year puts more and more pressure on airspace and airport capacities. The number of European flights in 2019 grew with a little over 11,000 compared to the year before, resulting in a total of 11.1 million flights. From these flights, 22.4% experienced a delay of 15 minutes or more (Eurocontrol [24]). With the continuous increase in demand for air-traffic and the limited capacities of airports and airspace, the impact of flight delay will only increase even more.

In order to ensure continuous operations at airports, operational schedules need to cope with these flight delays. Airport schedules should be robust such that they can handle delayed and early arriving flights. The impact of a flight, which does not arrive or depart at their scheduled time, on other flights should be limited as much as possible. This ensures that knock-on effects of delayed flights are reduced and that better continuous operations can be achieved. The operational schedules should be created by planning ahead with anticipation of delayed flights.

Research that improves anticipation on flight delay has seen significant growth in terms of the application of machine learning (Sternberg et al. [60]). Machine learning is a subset of artificial intelligence in which computers use data to learn how to perform tasks rather than being programmed to do them. Till now, flight delay has mainly been researched as a binary or regression problem. Classification models predict whether a flight will be delayed or not, based on a defined threshold of delay. These type of predictions lack information about what the quantity of the delayed flight will be. Regression models predict a numerical value of delay. Unfortunately, these predictions lack a certainty indication, making them less useful for decision making. In order to make flight delay predictions more useful in real-life applications, decision-makers need to have a certainty indication to base their decisions on.

This research focuses on providing a certainty indication by making probabilistic predictions of flight delay on an individual flight level with a prediction horizon of one day. The final goal of this research is to provide airport operators with probabilistic flight delay predictions for individual flights. Airport operators can use these probabilistic predictions to optimize schedules by anticipating flight delay of individual flights. This could result in more robust schedules to facilitate better continuous operations at the airport, reducing the number of delayed flights. This research has been conducted for the Aerospace Engineering faculty of Delft University of Technology with the help of Amsterdam Airport Schiphol, who provided the operational flight schedule data. The research has been conducted from June 2020 to February 2021.

The structure of this report is as follows. First, the scientific paper is presented in Part I. This paper describes the problem of flight delay, the models used to make probabilistic flight delay predictions and presents the most important results and conclusions of this research. Secondly, Part II contains the relevant Literature Study performed prior to the work presented in the scientific paper. This Literature Study has already been graded as one of the Aerospace Engineering master's degree courses. Finally, Part III contains the work that was done to support the scientific paper.

I

Scientific Paper

Predicting Probabilistic Flight Delay for Individual Flights using Machine Learning Models

Laurence Vorage
Faculty of Aerospace Engineering, Delft University of Technology, 2926 HS, Delft, The Netherlands

Abstract

To ensure continuous operations at airports, operational schedules need to be able to cope with early and late arriving/departing flights. To optimize such schedules for these events, flight delay predictions are necessary. Till now, flight delay has been studied mainly from a binary and regression standpoint. These type of predictions lack a certainty indication, which is needed by decision-makers to optimize a schedule. This research focuses on probabilistic predictions of flight delay on an individual flight level. Flight operation data from Amsterdam Schiphol airport and weather data from METAR is used to train four machine learning models that predict probabilistic flight delay distributions on a prediction horizon of one day. The first two developed models, a Dropout neural network and a Random Forest, predict empirical distributions. The other two models, a Mean Variance Estimator (MVE) and a Mixture Density Network (MDN) model, predict continuous distributions by predicting the parameters of an assumed distribution. The MVE predicts a single normal distribution, while the MDN model predicts a mixture of multiple normal distributions. Two interval-based performance metrics are suggested which assess the probabilistic predictions from different perspectives. It is concluded that arrival delay is best modelled with a MDN model with ten components, while departure delay is best modelled with only three components. Both models outperform a baseline statistical method on all considered interval widths. The MDN models outperform the other machine learning models by predicting more early arrivals and early departures correctly. The proposed models show that they can provide certainty indications of flight delay on an individual flight level, as opposed to binary and regression models. In the future, these certainty indications can assist airport operators with optimizing operational schedules, ensuring continuous operations and making ultimately air travel more pleasant.

Keywords: Flight delay, Probabilistic predictions, Machine Learning

1. Introduction

Air travel has seen continuous growth in the number of flights in the last years, causing increasing pressure on airspace and airports' capacity. In 2019, the number of flights in Europe grew by just over 11,000 compared to the year before, resulting in a total of 11.1 million flights. From these flights 22.4% experienced an arrival delay of 15 or more minutes (Eurocontrol [2020]). With the continuously increasing demand for air-traffic and the limited capacities of airports as well as airspace, flight delay will only see an increase in its impact. Research in the field of flight delay predictions has seen significant growth in applying machine learning (Sternberg et al. [2017]). Machine learning is a subset of artificial intelligence in which computers use data to learn how to perform tasks rather than being programmed to do so. Methods ranging from simple linear regression algorithms, to tree-based methods such as random forest to more complex algorithms such as deep neural networks have been applied.

Part of these algorithms aim to make binary classification predictions about (Choi et al. [2016]; Horiguchi et al. [2017]; Lambelho et al. [2020]), whether a flight will be delayed or not with respect to a certain threshold (often 15 minutes after the scheduled time of flights). The resulting predicted classes bring up the question: "with how many minutes does the model predict that a certain flight will be delayed?". If a flight is predicted to be in the delayed class, this could be a prediction of a delay of 16 minutes or a delay of 30 minutes. To obtain more insights in the actual value of the predicted delay, other algorithms approached flight delay as a regression task, where point predictions of delay are obtained (Rebollo and Balakrishnan [2014]; Yu et al. [2019]). However, the issue with a regression prediction is that it lacks any certainty indication. For two different flights, a model can predict the same delay value. For one of these predictions, the model may be certain, while for the other, the model may be uncertain. This lack of certainty indication in predictions hinders the usage of machine learning models in real-life implementations.

Karim et al. [2018] described that "a single evaluation metric is an incomplete description of most real-world task, which is the basis for all machine learning techniques". Different studies have researched the certainty aspect of machine learning predictions. Gal and Ghahramani [2016] represented model uncertainty in deep learning by using a so-called dropout process. The authors were motivated by the fact that "standard deep learning tools for regression and classification do not capture model uncertainty". Other studies trained models not to predict a single regression value, but rather the parameters of a probability distribution, which provides a direct measure of the certainty of the prediction. Nix and Andreas [1994] for example, introduced a method to predict the mean and variance of a probability distribution, given an assumed target error-distribution model. Bishop [1994] extended this idea of predicting the parameter of one assumed distribution, to predict the

parameters and weights of multiple Gaussian distributions which theoretically can model arbitrary probability distributions.

In the field of flight delay research, little research is performed concerning the probability of a prediction. Some binary classification studies like Pérez–Rodríguez et al. [2017] investigate probabilities, but from a standpoint on whether a flight will be delayed or not with a certain probability. A study by Mueller et al. [2002] modelled flight delay by a random variable that follows a statistical distribution. However, this study did not distinguish between individual flights and their respective features and modelled all flights from the same distribution.

This paper proposes and evaluates multiple machine learning models capable of making probabilistic flight delay predictions on a prediction horizon of one day. These models predict distributions, continuous or discrete, which are used to create prediction intervals. These intervals are evaluated from a mathematical and practical perspective to compare the performance of different probabilistic machine learning models. This analysis provides a means to assess the certainty of flight delay predictions in order to make them more useful to be implemented by airport operations in real-life applications. We demonstrate our methodology using 7 years of flight schedule data from early 2012 to early 2019 at Schiphol Amsterdam Aiport (AMS), merged with METAR weather data. To the best of our knowledge, this paper is the first to addresses probabilistic flight delay distribution predictions on an individual flight level.

The main contribution of this paper is that it provides flight delay predictions from a probabilistic standpoint. Until now, research in the field of flight delay using machine learning methods has focused on binary predictions and regression predictions. By considering flight delay predictions from a probabilistic standpoint, the certainty indications of these predictions give airport operators clarity and confidence in making decisions based on them. Schiphol airport and other airports could benefit from this research. It might allow them to use probabilistic flight delay predictions to increase operations efficiency at the airport by optimizing, for example, the flight to gate assignment one day before operations.

The remainder of this paper is structured as follows. Section 2. discusses the probabilistic flight delay models, including features, algorithms and performance metrics. Section 3. presents the results of the proposed machine learning approaches to which make probabilistic flight delay predictions. Section 4. discusses the results and their implications, where section 5. concludes the research and outlines future research directions.

1.1. Related work

Multiple researches have been carried out to investigate the use of machine learning algorithms to predict flight delay. These studies can roughly be divided into 3 main groups, classification studies (binary and multiclass classification), regression studies and probabilistic studies.
Firstly, the majority of studies researched flight delay as a binary classification problem. These studies differ mainly on the different types of algorithms, prediction horizons and delay thresholds used. Several studies used Random Forest algorithms (Rebollo and Balakrishnan [2014]; Choi et al. [2016]; Belcastro et al. [2016]; Gui et al. [2020]) to classify flight delay, while other studies made use of neural networks (Horiguchi et al. [2017]; Gui et al. [2020]; Lambelho et al. [2020]).

Rebollo and Balakrishnan [2014] and Belcastro et al. [2016] experimented what an increase in the threshold of the definition of delay would result into. Both researches, found that increasing the threshold of delay increased the classifier's performance, where the best performance was found when using the highest threshold value for delay.
The impact of increasing the prediction horizon, the time between the moment of prediction and the moment of the flight's execution was researched by Rebollo and Balakrishnan [2014] and Horiguchi et al. [2017]. In both researches it was found that increasing the prediction horizon results in a decrease in the classifier's performance. Horiguchi et al. [2017] verified this by changing the prediction horizon for 3 different classifiers, for which all three the performance decreases.

The implication of imbalanced data was researched by Choi et al. [2016]. The authors applied a combination of SMOTE (Synthetic Minority Oversampling TEchnique) (Chawla et al. [2002]) and random undersampling. Four binary classifiers were evaluated. All four obtained better accuracies without the sampling techniques. The authors argued that these results do not imply that applying a sampling technique is a bad choice. Chakrabarty [2019] applied the SMOTE technique to a gradient boosted method and saw the performance of their classifier improved from 0.802 to 0.857 by applying SMOTE.

Some studies extended the number of classes to more than two resulting in a multiclass classification problem (Alonso and Loureiro [2015]; Chen and Li [2019]; Gui et al. [2020]). From these researches, it is seen that the accuracy of correct predicted classes by the models is dependent on the number of intervals. Chen and Li [2019] also analyzed as well the propagation of flight delay. The authors showed that departure delay and late-arriving aircraft delays are the most important feature for delay prediction. In this study, with a prediction horizon of one day, we cannot use these features, as they are only available on the day itself once the aircraft has taken off or landed at AMS.

Secondly, some studies investigated the use of regression models to predict flight delay, where the actual value of delay in minutes was predicted, instead of a class. Tu et al. [2012] developed a smoothing-based splines model, which estimates flight departure delay distributions. The model consists of three inputs, a daily propagation, a seasonal trend and a mixture distribution to estimate the residual errors. Rebollo and Balakrishnan [2014] trained next to their Random Forest classifier, a Random Forest regression model, which was evaluated on the 100 most delayed O-D (Origin-Destination) pairs in the US. An average median test error of 27.4 minutes was found on a prediction horizon of 24 hours. The authors concluded that in general, the classification and regression problems are different as the former one requires information that helps to differentiate between high and low delay values, whereas the latter tries to predict the actual value of the delay. Yu et al. [2019] used a "noveldeep belief neural network method to mine the inner patterns of flight delays", as the authors described. One of their goals was to detect both macro and micro-level key influential factors, such as air route situation and crowdedness degree of the airport. The model's performance comes out that 99.3% of the predicted values were within a 25 minutes deviation from the observed delay value.

Thirdly, only a few studies researched flight delay from a probabilistic standpoint. Mueller et al. [2002] modelled delay by assuming that it is a random variable and follows a statistical distribution. A least-squares method was used to minimize the fit error between the raw data and the random variable. It was found that departure delay can best be modelled by a Poisson distribution, while en-route and arrival delays fit a Normal distribution better. Wu [2014] analyzed the probabilistic distribution of airport arrival and departure delays over a period of 8 months using an optimal Generalized Extreme Value (GEV) model, which is a family of continuous probability distributions. The GEV model was chosen to describe the tail characteristics of the distribution. To estimate the coefficient of the GEV model, a maximum likelihood estimation (MLE) was used. Both researches considered flight delay from a probabilistic standpoint, but modelled all flights from the same distribution and not on an individual flight level.

This paper expands the aforementioned work on flight delay prediction by developing machine learning models that predict probabilistic flight delays with a prediction horizon of one day for individual flights. These probabilistic distributions are used to create intervals with which the actual delay value will be compared to assess the performance of the predicted flight delay distributions.

1.2. Data description

The work presented in this paper is based on 2 datasets which form the basis to train, validate and test the different probabilistic models on. The first and most important dataset is the flight operational dataset from Schiphol airport (AMS). This data contains information about arrival and departure flights in the period from 25 March 2012 till 30 March 2019. This flight operational data corresponds to 7 years of data. Every year contains the summer schedule (end of March till the end of October) of around 7 months and the winter schedule (end of October till the end of March) of around 5 months. The dataset contains a total of just over 3 million flights and is structured around origin-destination (O-D) pairs. In each data entry (row), the origin or destination of the flight is always AMS. Furthermore, each row contains information elements (from now on called features) such as scheduled time, actual time, airport, airline, etc. A complete list of all features of the flight operational dataset is given in Table 1.

From all flights in the dataset, approximately 1.6% are cancelled flights, which are disregarded for this research. Flights which arrive more than 60 minutes early or arrive more than 2 hours late are also considered as outliers (approximately 1.2%) and disregarded. Freight flights (approximately 3.2%) are removed as this research focuses on predicting flight delay of passenger flights. The distribution of flight delay of the resulting flights is presented in Fig. 1 for both arrival and departure flights. It is seen that arrival and departure delay do not follow the same shape, there are more arrivals than departures before their respective scheduled times. In Fig. 1, a vertical red line is given which indicates the definition of a delayed flight. In this research a flight is defined as arrived punctual when the "flight arrives within 15 minutes or earlier than their scheduled arrival time" (Eurocontrol [2019]). This means that flights that arrive 16 minutes or more after their scheduled time are defined as delayed.

Table 1: Flight schedule and weather data with their respective features and data types (n=numerical feature, c=categorical feature, p=periodical feature)

Dataset	Features
Flight operations	Scheduled arrival/departure time^p, Actual arrival/departure time^p, Airline^c, Airport^c, Flight number^c, Number of seats^n, Taxi time^n, Schengen^c, Terminal^c, City^c, Continent^c, Country^c, Gate^c, Ramp^c, Total passenger^n, Total transfer passenger^n, Freight^n, Aircraft registration number^c, Aircraft type^c
Weather	Temperature^n, Dew point temperature^n, Relative humidity^n, Wind direction^n, Wind speed^n, Gust speed^n, Pressure^n, Visibility^n

Table 2: Arrival delay statistics

Year	Mean delay (minutes)	Standard deviation	Absolute mean delay (minutes)
All Years	0.37	21.38	14.49
2012	-1.18	19.76	13.63
2013	-2.54	19.66	13.88
2014	-0.97	19.90	13.69
2015	0.49	21.33	14.31
2016	0.5	20.99	14.19
2017	3.7	22.76	15.29
2018	0.75	22.86	15.72

Table 3: Departure delay statistics

Year	Mean delay (minutes)	Standard deviation	Absolute mean delay (minutes)
All Years	10.77	17.73	12.25
2012	8.12	16.08	10.20
2013	7.82	15.7	9.93
2014	9.25	16.34	10.96
2015	10.22	17.6	11.81
2016	10.97	17.4	12.21
2017	13.9	19.3	14.89
2018	13.06	19.08	14.11

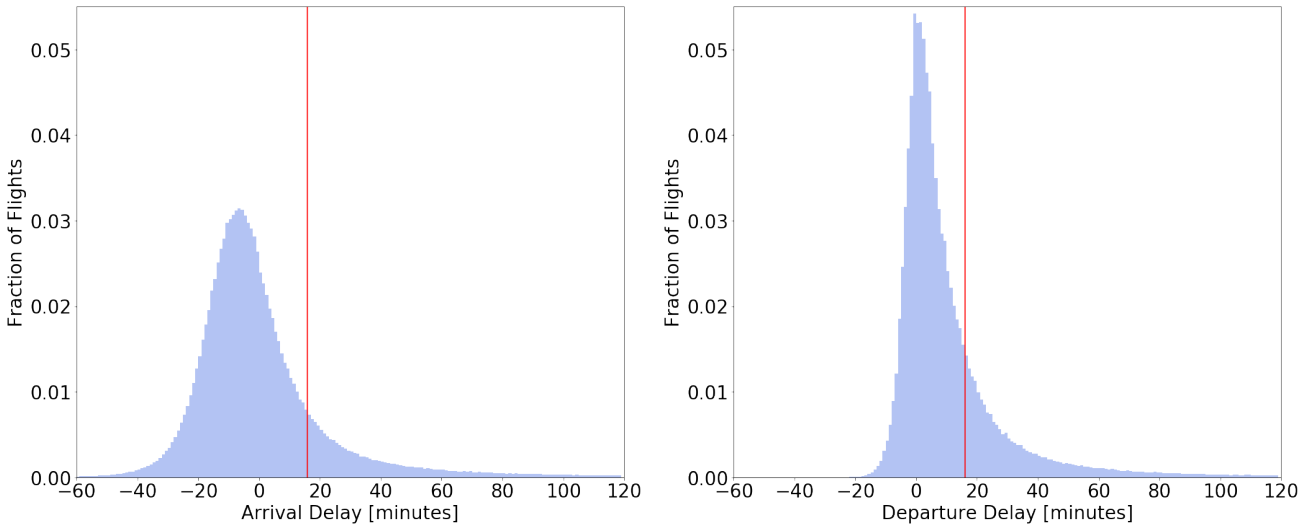


Figure 1: Delay distribution histogram for Arrival and Departure flights. The red vertical line shows the delay threshold of 16 minutes.

For both the arrival and departure distribution important statistics can be obtained which are presented in respectively Table 2 and Table 3. An important metric to keep in mind is the absolute mean delay, which is the absolute deviation from the scheduled time of arrival/departure. It could be thought of as the Mean Absolute Error (MAE) of an algorithm that would predict for every flight, the scheduled time of that flight. Both tables show that the absolute mean delay has been increasing in the last couple of years.

The second used dataset in this research contains weather data. This weather data is provided by METAR reports (IowaStateUniversity [2020]) and contains the METAR weather station's measurements at Schiphol airport. This METAR weather data consists of information on the temperature, wind direction, wind speed, gust speed, etc. (see Table 1 for full list). Weather data is retrieved for the same period (end March 2012 till end March 2019) and merged with the flight schedule data of Schiphol. In total, the METAR dataset contains over 120,000 samples, with measurements every 30 minutes.

2. Machine learning algorithms for probabilistic flight delay predictions

In this section, machine learning models are presented which predict flight delay distributions for arrival and departure flights at AMS airport. These predictions are based on operational flight and METAR weather data and assume a prediction horizon of one day, meaning that the prediction is made 1 day before the flight's day of execution. It is assumed that on a prediction horizon of one day, weather predictions equal the true weather conditions, such that actual weather data can be used. The predicted distributions of individual flights are used to obtain intervals for which their performance will be obtained. To train, validate, and test the models, 7 years of data in the period from 2012-2019 is used. In Section 2.1. the creation, encoding and selection of features are discussed, which form the models' input. Section 2.2. presents the four machine learning algorithms used to predict probabilistic flight delay. It also includes a statistical method which serves as a baseline for the machine learning algorithms to beat. In Section 2.3. the performance metrics are discussed on which the different algorithms will be compared.

2.1. Features

2.1.1. Feature creation

Extra features are created alongside the features provided in the raw data from Schiphol and METAR. These newly created features aim to provide relevant 'domain knowledge' to the model, such that it can better predict flight delay. The following numerical features were created from the scheduled arrival/departure times: day of year, day of month, day of week, minutes of day. The feature Distance was created based on the origin and destination data from the flight's operation set. Two features were created to provide information about the crowdedness at AMS airport. These are Number of flights last hour and Number of flights coming hour.

Based on the METAR weather data, three different new features were created. The feature Above 3 degrees is a binary feature, which indicates if the temperatures drop below 3 degrees, as than de-icing precautions are taken for flights taking off from AMS. AMS has 3 runways in the north-south direction, the Aalsmeerbaan, the Zwanenburglaan and the Polderbaan (Schiphol). When wind gust speeds perpendicular on these runways exceed 30 knots, they have to be closed, causing a drastic reduction in capacity for AMS. The feature Cross gust north-south runways provides the gust speed perpendicular to the north-south runways in knots. The feature Cross gust above 30 knots is a binary feature, which equals 1 when Cross gust northsouth runways is above 30 knots and equals 0 otherwise.

2.1.2. Feature encoding

The categorical features used in this research need to be encoded as machine learning algorithms can only process numerical information. Two different encoding types are used to accomplish this, binary encoding and one-hot-encoding. Binary encoding is a simple technique in which features with only binary information are encoded to either 0 or 1. For example; *Schengen* becomes 1 and *Non-Schengen* becomes 0.

For all categorical features with more than two options, the encoding technique of one-hot encoding is used. This technique creates a separate column for every unique option of a categorical feature in which the value is set to 1 only when that option of the category is present, all other options are set to 0. In Table 4 the one-hot encoding procedure is depicted for the feature Airline with the option of KLM, TRA and BAW. One-hot encoding adds significant extra columns to the model's input data (roughly 100 for airline and 200 for airport). Normally this method is not preferred as it increases the training time of algorithms, but for this research the increase in training time of the models remained reasonable (approximately 1hour for the neural networks).

An often-used alternative for one-hot encoding is target encoding, which encodes a feature with the mean target value (delay) of that category over the training set. If for example, the mean delay value of departure KLM flights is 12.4 minutes, then every KLM flight would be given that value for the airline features. This method was not chosen as it directly included specific statistical information about that airliner to the model. In this research, it was chosen to keep the models as 'general' as possible. Meaning that it would have to find and combine features which are important for delay instead of basing a prediction merely on the provided statistical values.

Periodical features, such as *month* and *wind direction*, have been transformed by trigonometric functions to account for their periodicity (Horiguchi et al. [2017]). Table 4 gives an example of periodical encoding of the feature *month* into a sine and cosine component. This ensures that the month December (12) will be followed by January (1), making them sequential for the machine learning models. Finally, to account for the different ranges of values per feature, feature scaling is used through normalization, ensuring that features have a common scale.

2.1.3. Feature selection

Three different methods are sequentially used to select features that will be used by the machine learning models. First, features are selected based on their Pearson correlation coefficient score, this score displays the linear correlation between features and the target variable (delay) and the correlation between two features. If features have a linear correlation higher

Table 4: Different encoding techniques examples

Original features			Encoded features					
Airline	Month	Schengen	One-hot-encoded			Periodic encoding		Binary encoding
			Airline -KLM	Airline -TRA	Airline -BAW	Month (sine)	Month (cosine)	Schengen -encoded
KLM	1	Yes	1	0	0	$\sin(\frac{2\pi \cdot 1}{12})$	$\cos(\frac{2\pi \cdot 1}{12})$	1
TRA	3	Yes	0	1	0	$\sin(\frac{2\pi \cdot 3}{12})$	$\cos(\frac{2\pi \cdot 3}{12})$	1
BAW	12	No	0	0	1	$\sin(\frac{2\pi \cdot 12}{12})$	$\cos(\frac{2\pi \cdot 12}{12})$	0

than 0.8 with another feature, one of them will be removed. This is done because high linear correlations between features, could lead to multicollinearity for some machine learning models. The feature that will be removed is the one which is less certain to be known exactly one day in advance or has a significant lower correlation with the target variable. For example, the features Passengers and Seats have a high correlation, meaning one should be removed. As Seats is a given value and the exact number of passenger is more difficult to be known one day in advance, the feature passengers was removed.

Because the Pearson correlation only looks at the linear correlation between features and the used algorithms in this research are non-linear methods, a second feature selection method is used. This method is called Permutation Importance and looks at how important a feature is for making predictions. Based on this information, a feature can be dropped when it has no added prediction power for the algorithm. This method works by first calculating the performance of the model on a 'normal' validation set. This score will be used as baseline performance. Now for every feature, its column with values gets randomly shuffled. With this single feature being shuffled and the other feature columns still in normal order, the validation set's performance is again computed. The difference between the baseline performance and the single feature shuffled performance indicates how important this single feature is for making good predictions for the model. If the performance decreased significantly, this means that the feature is important for making good predictions. If the performance hardly changed (or even increased), this indicates that random shuffled values of this feature don't influence the quality of the predictions. This means that this feature is not important in making predictions and therefore can be dropped. For each feature, it's column is shuffled 10 times to obtain a more unbiased value of the Permutation Importance score. Features were removed if their permutation performance score was less than 10% of the best performing feature. This relative method was chosen because it can be applied to both the arrival and departure set of features without setting a single fixed threshold value for both beforehand.

Permutation Importance is a useful method to obtain an indication of the importance of numerical and binary features for a model. As in this research, the categorical features are one-hot encoded, permutation importance is not feasible anymore. This is because of the high number of extra columns created. Removing one single option of a category is not possible without removing the whole feature. Therefore, an iterative process of removing categorical features was performed on a reference neural network. If by removing a feature, an increased or equivalent performance was obtained, then this categorical feature would be removed from the set. For categorical features containing the same sort of data, for example geographical data as airport, country or continent, a single feature was chosen which had the most performance increase. This approach was chosen because having multiple features results in significant more columns and barely resulted in an increase in performance. Table 5 shows the selected features after applying the three feature selection methods sequentially for arrival and departure flights with a prediction horizon of one day.

2.2. Probabilistic flight delay algorithms

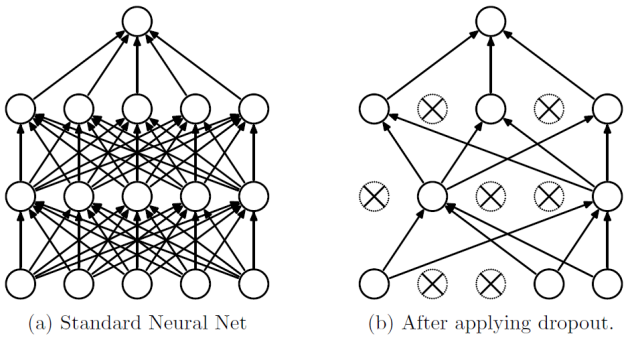
In this section the four probabilistic machine learning algorithms to predict flight delays on a prediction horizon of one day are presented: Neural Network with Dropout (Dropout), Random Forest (RF), Mean Variance Estimator (MVE) and Mixture Density Network (MDN). These four algorithms can be divided into two main groups. Firstly, algorithms which predict distributions empirically by making n predictions (Dropout and RF). Secondly, algorithms which predict distributions by predicting the parameters of an assumed distribution (MVE and MDN). Next to these four algorithms, a statistical method is presented which serves as a baseline for the performance of the other models.

Dropout (Srivastava et al. [2014]) is a simple way to prevent neural networks from overfitting and is for that reason mainly used in the training phase of a model. Its name refers to the process of randomly dropping out units (neurons) in the network. As the authors describe; "dropout prevents units from co-adapting too much, which significantly reduces overfitting and gives major improvements over other regularization methods". The choice of which unit gets dropped is random, where the dropout rate p determines the independent probability of a unit being retained. Fig. 2 visualizes how the dropout method works, where in Fig. 2a, a standard neural network is given with two hidden layers. Fig. 2b gives the same network, but now with the dropout method applied, which results in a 'thinned network'. The crossed units have been dropped out.

Normally the dropout method is applied for regularization purposes, but it can also be used to represent the uncertainty of a model, which is done by Gal and Ghahramani [2016]. In their research they used dropout to approximate Bayesian inference in deep Gaussian processes, which results in a tool to model uncertainty from existing models. To estimate the predictive mean and predictive uncertainty, a Monte Carlo method approach is used, in which N stochastic forward passes through the model are collected. These N predictions are used to obtain an empirical distribution for a single prediction of a flight. The big advantage is that uncertainty is represented without the computational complexity of training multiple models. Fig. 1 showed that arrival delay and especially departure delay do not follow an exact Gaussian distribution. They both have a heavier right tail, with more late arriving/departing flights than early arriving/departing flights. However, this method is still used as it represents the uncertainty of neural networks in an achievable computational expensive manner.

Table 5: Description of features used for probabilistic flight delay prediction on a one day prediction horizon. C=Categorical feature, B=Binary feature, N=Numerical feature, T=Trigonometric transformed periodical feature

Feature name	Feature type	Feature Description	Selected for Arrival delay	Selected for Departure delay
Airport	C	Origin/destination airport of the flight	X	X
Airline	C	Airline operating the flight	X	X
Aircraft	C	Aircraft type		
Terminal	C	Terminal assigned to the flight	X	
Country	C	Country of origin/destination airport		
Continent	C	Continent of origin/destination airport		
Distance	N	Distance between origin and destination airport (km)	X	X
Seats	N	Number of seats in the aircraft operating the flight	X	X
Schengen	B	Indicator of flight arriving/departing to Schengen area		X
Flights last hour	N	Number of scheduled flights in the last hour	X	X
Flights coming hour	N	Number of scheduled flights in the coming hour	X	X
Year	N	scheduled year of the flight		
Month	T	scheduled month of the flight		
Minutes of day	T	scheduled minutes of the day of the flight	X	X
Day of year	T	scheduled day of the year of the flight	X	X
Day of month	T	scheduled day of the month of the flight	X	X
Day of week	T	scheduled day of the week of the flight	X	X
Temperature	N	temperature at AMS (degrees)	X	X
Humidity	N	Humidity level at AMS (relative humidity %)	X	X
Temperature above 3 degrees	B	1 if temperature is above 3 degrees, 0 otherwise		X
Wind gust speed	N	Maximum wind speed at AMS (knots)	X	X
Wind direction	T	Wind direction at AMS	X	X
Visibility	N	Visibility at AMS (miles)	X	X
Cross gust north-south runways	N	Wind gust component perpendicular to the north-south runways (kntos)	X	X
Cross gust above 30 knots	B	1 if cross gust north-south runways is above 30, otherwise 0		



(a) Standard Neural Net (b) After applying dropout.

Figure 2: Visualization of the dropout method. Taken from Srivastava et al. [2014]

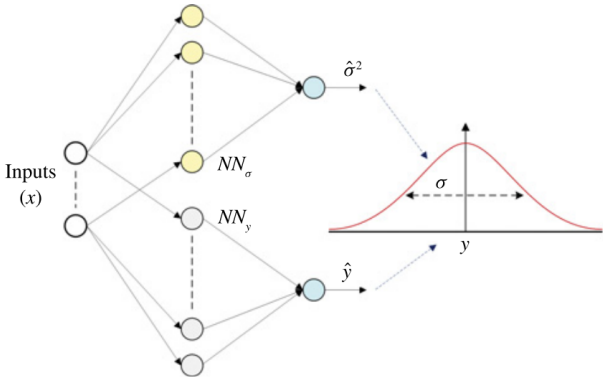


Figure 3: Architecture of the MVE method. Taken from Khosravi et al. [2011]

Random Forest (Breiman [2001]) is an ensemble method consisting out of a multitude of decision trees, which can be used both for classification and regression. The output of a RF model is the aggregation of the output of independent dissimilar decision trees. For a regression RF, the output is the mean value of all predictions of the trees. To build each decision tree a random part of the training set is used. For increased diversity of the trees, they only get to see a random fraction of the total number of features. Instead of using only the mean values of all the trees, the predictions of the trees will be used to create a distribution prediction of flight delay per individual flight.

Mean Variance Estimator (Nix and Andreas [1994]) is a method that directly estimates the mean and variance of a probability distribution. Instead of training a neural network to predict a point value, it is trained to predict the parameters of an assumed probability distribution. In this sense, a direct measure of the uncertainty of a prediction is obtained by its variance. The architecture of a typical MVE neural network is depicted in Fig. 3, where the output layer consists out of two nodes predicting $\hat{y}$ and $\hat{\sigma}^2$. Note that both outputs share the same input units, but do not share any of the hidden layer connections. This approach of splitting the hidden layers per output is not necessary as $\hat{y}$ and $\hat{\sigma}^2$ could be connected to a common large set of hidden layers, but in the experience of Nix and Andreas [1994] this split-hidden-unit architecture performs better.

Conventional neural network models use error based minimization techniques, such as Mean Absolute Error (MAE) or Mean Squared Error (MSE), during the training process. As the target variance value, $\hat{\sigma}$ is not known, this approach cannot be used. Therefore Nix and Andreas [1994] and Khosravi et al. [2011] use a Maximum Likelihood Estimation (MLE) formulation for the loss function in the training process. Based on the assumption of normally distributed errors around y_i, the conditional distribution on the input data can be written as in Eq. (1). In this process the actual value (y_i) of the target variable is known and the goal is to find the values of $\hat{y}_i$ and $\hat{\sigma}_i^2$ that maximize the probability of $\hat{y}_i = y_i$. In other words $\hat{y}_i$ and $\hat{\sigma}_i^2$ are varied for a known value of y_i, to find the distribution from which y_i comes most likely.

$$p(y_i|x_i, NN_y, NN_\sigma) = \frac{1}{\sqrt{2\pi\hat{\sigma}_i^2}} e^{-\frac{(y_i-\hat{y}_i)^2}{2\sigma_i^2}} \tag{1}$$

Mixture Density Networks (MDN) (Bishop [1994]) extend the idea of predicting the parameters of one assumed distribution, to predicting the parameters and weights of multiple distributions, which make up one distribution. Theoretically, a mixture of Gaussian functions is capable of modelling arbitrary probability densities (Bishop [1994]), if it is adequately parameterized with enough components. The probability density function of the prediction is then represented as a linear combination of kernel functions in the form of Eq. (2).

$$p(y|\mathbf{x}) = \sum_{j=1}^{m} \alpha_j(\mathbf{x})\phi_j(y|\mathbf{x}) \tag{2}$$

The number of components in the mixture is represented by *m*. The parameters $\alpha_j(\mathbf{x})$ are called the mixing coefficients, which are functions of the input **x**, which must satisfy a sum of 1. The functions $\phi_j(y|\mathbf{x})$ are the conditional density functions of the target y for the j^{th} kernel. The loss function is defined in the same way as the MVE method, but now the *m* components of the mixture need to be taken into account. An MDN model with 3 components is referred to as an MDN3 model, a model with 5 components is referred to as an MDN5 model, etc.

The statistical method is not a machine learning algorithm, it bases its predicted distribution solely on the statistical delay data of the training set. This means that the prediction for every flights is the distribution of delay of the training set. The statistical method predictions are almost identical to the delay distributions given in Fig. 1 of the arrival and departure set (due to that the majority of the flights of the full set are within the training set). The idea behind this statistical method is to serve as a baseline. This is to verify whether (complicated) machine learning algorithms outperform simple statistics in the field of probabilistic flight delay predictions.

2.3. Performance Metrics

To compare the different researched models, performance metrics need to be established. All the models predict distributions, while the actual flight's delay value is a point value. This means that straightforward metrics such as MAE and MSE cannot directly be used. They can be used by computing the mean of a predicted distribution and compare them to the actual delay value. However, this approach is not preferred, as the uncertainty of the predictions would be lost, which is precisely where this research is interested in. Two metrics are used in this research to compare the delay value with intervals. These intervals are derived from the predicted distribution of the models.

The first performance metric is called: "Actual delay in predicted interval". This metric checks whether the actual delay value lies within the model's predicted interval, which has a fixed length. Its outcome is either a 1; the actual delay value lies within the interval or a 0, the actual delay value does not lie in the predicted interval. Summing over the entire test set will give the percentage of correctly predicted intervals. The chosen intervals are based on the predicted distributions by the model. The interval is chosen, which maximizes the probability of the distribution which falls within that interval. This process of selecting the interval based on a distribution is depicted in Fig. 4 for a departure flight from KLM to Hamburg.

In Fig. 4a the predicted distribution is obtained, wherein Fig. 4b an interval with fixed length of 30 minutes is chosen. The interval with the maximum probability mass of the distribution is found to be [-11,19). In Fig. 4c the actual delay value (2 minutes) is compared whether it lies within the predicted interval or not. In this case, it lies within the interval and the actual delay value in interval metric for this flight for the 30 minutes interval is set to 1. Note that the overall performance

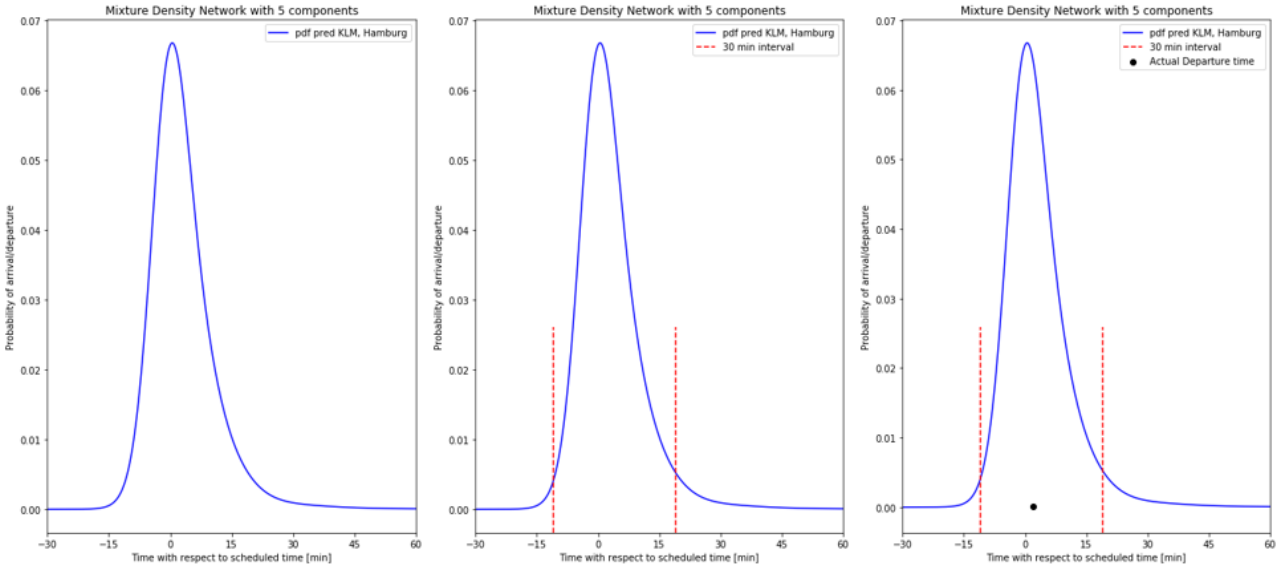


Figure 4: Actual delay in predicted interval metric

of the intervals in this method is heavily dependent on the fixed length which is chosen. Bigger intervals mean better (or equal) performance. Therefore the fixed length of these intervals will be varied to see how this influences the performances per method.

For empirical methods, the N predictions of the distribution may be centred and do not span over the full considered time axis. This results in that for larger intervals, at some point no more information is available to choose the exact interval on. If the range in which all samples lie is smaller than the considered prediction interval, than the range will be extended on both sides symmetrically to the desired length of the interval is reached. For example, if all N predictions lie in the range of 5 to 25 minutes and a 30 minutes interval is considered, the chosen interval will be extended to [0,30). Note that the predicted intervals are right-open intervals in this research. This is because the actual delay value of a flight in the original dataset is a timestamp which is rounded down to full minutes. This means that flights which arrive at 10:15:01 or at 10:15:59 are both rounded down to 10:15. The right open interval ensures that a flight from 10:16:00 and onwards does not fall within the predicted interval [10:05, 10:15).

Note that for the 'Actual delay in predicted interval' metric, the distributions are not directly used to compute the performance metric. This means that two different distributions can result in the same intervals being chosen, resulting in the same performance. This means that a distribution with a low variance can have the same performance as a distribution with a high variance.

The second performance metric is called the "Probability mass in actual delay interval". This metric looks at the predicted distribution from a more mathematical standpoint and considers the variance or spread of the distribution. The metric focuses on the quality of the distribution around the actual delay value. It is defined as the probability mass of the predicted distribution in a symmetrical interval with fixed length around the actual delay value. Fig. 5 and Fig. 6 both give a different MVE prediction of the same example flight. In both figures, the blue dotted lines indicate the 10 minutes interval around the actual delay value. For the MVE prediction in Fig. 5 only 0.31 probability mass falls within the 10 minutes interval, while for the MVE prediction in Fig. 6, 0.64 probability mass falls within the same interval. These figures visualize that both MVE distributions resulted in the same predicted interval (red dotted line), in which the actual delay value lies. But that the performance metric of Probability mass in actual delay interval, can quantify the certainty of the predicted distribution around the actual delay value and can distinguish between distribution predictions with high and low variances.

At this point, it needs to be pointed out that the used performance metrics in this research are not the same as the loss functions used to train the models. This means that the models are not trained on predicting the best quality interval, but are optimized to predict another metric. The MVE and MDN models use the negative of the log-likelihood functions as loss function. While the empirical models, Dropout and Random Forest, use the Mean Absolute Error (MAE) metric as loss function. This metric was chosen over the more standard Mean Squared Error (MSE) metric, as MAE is more robust to outliers. Because this Thesis has a limited time frame, it was not possible to write custom loss functions that can directly train based on the interval performance metrics. The models in this research are however still useful as they can serve as a baseline for further researches in the field of probabilistic flight delay distribution predictions.

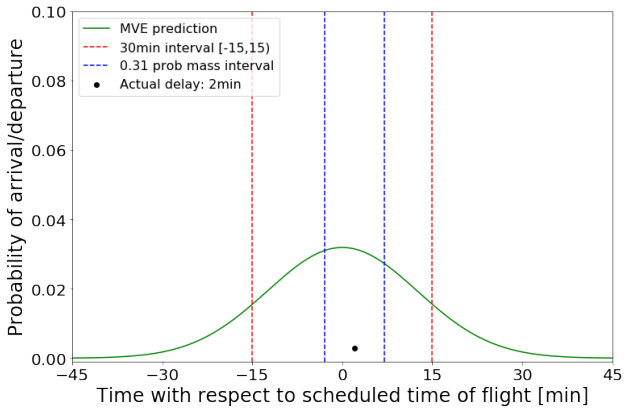


Figure 5: MVE prediction example with high variance, with both performance metrics visualized

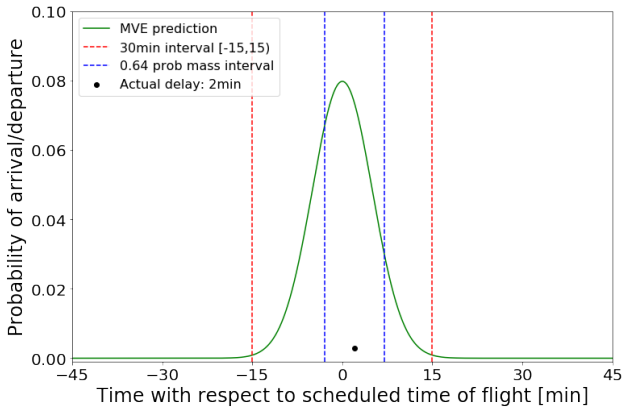


Figure 6: MVE prediction example with low variance, with both performance metrics visualized

3. Results

In this section, the results of the models are presented on the test set. All models use the same training and test set. These sets are obtained by a random split of the original prepared feature set with by a split of 80% for training and 2- % for testing. It is chosen to use a random split instead of a time series split as delay data can vary from year till year, as was seen in Table 2 and Table 3. By introducing a random split this 'year trend' of delay statistics, specific to Schiphol, is removed from the data for the models.

Before obtaining the performance of the models on the test set, the optimal hyperparameters of each model will need to be found. This is done via a k-fold cross-validation (with k=5), which takes as input the training set. In this process, the training set is randomly subdivided for 5 times into a 'smaller' training set and a validation set. This validation set is used to compare the different hyperparameters' performance to find the best-performing setting. A visualization of 5-fold cross-validation is given in Fig. 7 (in which the small green blocks form the train set and the small blue block the validation set of a fold in the cross-validation).
The created test set is solely be used to evaluate the generalized error of the final model. If the test set were used to tune the hyperparameter, a too optimistic generalized error would be obtained. This is because otherwise, the hyperparameters would have been chosen which performed best on the test set. Hastie et al. [2009] argues that for using cross-validation, "samples must be left out before any selection or filtering steps are applied".

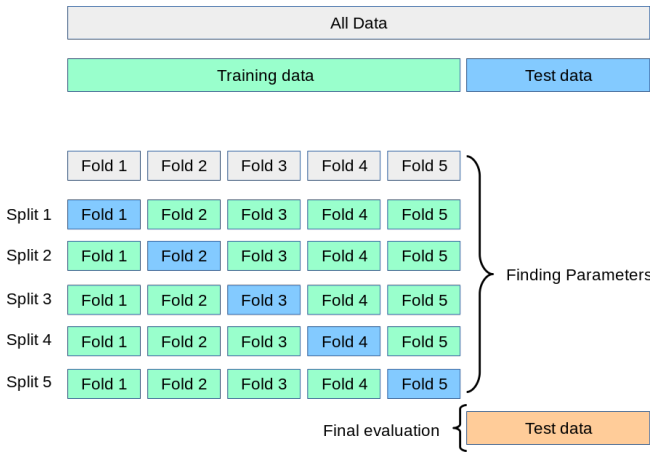


Figure 7: 5-fold cross-validation Taken from Scikit-learn developers [2017]

3.1. Hyperparameter tuning results

By using this approach of 5-fold cross-validation on the training set, the optimal hyperparameters of each of the four models are obtained. These optimal hyperparameters are displayed per model type in Table 6 till Table 9. Note that for the MDN models, the MDN5 architecture was used to find the optimal hyperparameters. These hyperparameters were then used for both the MDN3 and MDN10 model. This method was chosen instead of hyperparameter tuning each individual MDN architecture, as this would become computationally very expensive. This method also ensures that the only difference between the MDN models is the complexity defined by the number of normal components in each model.

For the implementation of a Dropout model, a reference neural network is trained, which architecture and weights are kept constant, and the predicting dropout rate is varied. Note that this dropout rate can be different than the dropout rate used in the learning process. The dropout rate in the learning process is meant to reduce overfitting of the model. The hyperparameters in Table 8 are the hyperparameters of the arrival and departure reference models. This ensures that the predicting dropout rate is the only changing variable in the predicting phase.

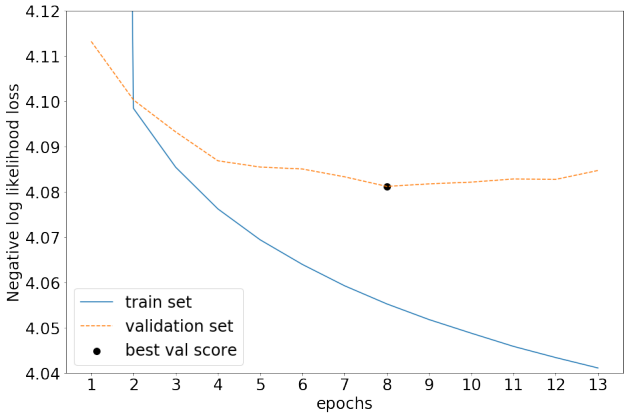


Figure 8: Early stopping loss values for arrival MDN model

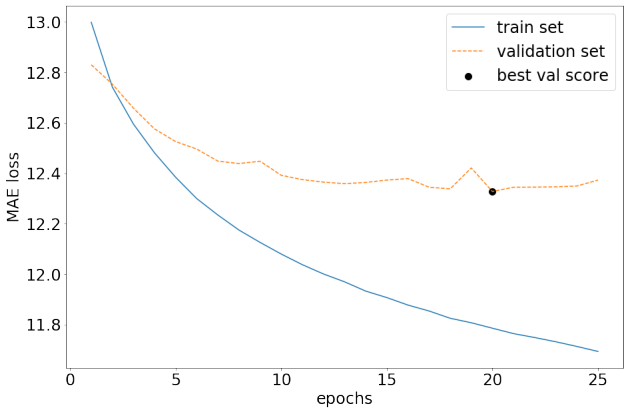


Figure 9: Early stopping loss values for arrival Dropout model

The hyperparameter of the number of trees given in Table 9 is fixed at 1000. This is done because 1000 predictions are found to be substantial enough to obtain detailed empirical distributions, but not too computationally restricting. Note that the Dropout method also makes 1000 predictions per distribution, such that both methods obtain distributions with the same level of detail.

Table 6: Hyperparameters of MVE models

MVE model	Number of layers	Number of neurons per layer	Dropout rate (learning)
Arr. delay	3	200	0.00
Dep. delay	5	200	0.00

Table 7: Hyperparameters of MDN models

MDN model	Number of layers	Number of neurons per layer	Dropout rate (learning)
Arr. delay	5	200	0.00
Dep. delay	5	250	0.00

Table 8: Hyperparameters of Dropout models

Dropout model	Number of layers	Number of neurons per layer	Dropout rate (learning)
Arr. delay	3	200	0.05
Dep. delay	5	200	0.05

Table 9: Hyperparameters of Random Forest models

RF model	Number of trees	Max depth of tree	Percentage features for each split
Arr. delay	1000	20	0.2
Dep. delay	1000	18	0.2

Training a neural network is a stochastic process in which the weights of the model are continuously changed in order to perform better on the training set. During the training process, there will be a point in which the model will stop generalizing on the data and will start learning the statistical noise of the training set. This is also the case for the neural network models used for the MVE, MDN and Dropout method. To ensure that the model doesn't start overfitting, but is generalized enough on the data an early stopping process is used. As Bishop [2006] describes it: "the goal is to achieve the best generalization performance, which suggests that training should be stopped at the point of the minimum of the validation set error".

The early stopping method applied in this research uses 50 epochs with a patience of 5 epochs (completed passes of the entire training's set through the machine learning algorithm). This means that the model will be trained on a maximum of 50 epochs. At the end of each epoch, the current model with its weights will be tested on the hold out validation set. If the performance of the model on the holdout set does not increase for 5 epochs in a row, the learning process will be stopped. If the training process of the model comes to an early stop, the weights of the model with the best performance score on the holdout validation set will be selected. Figure 8 displays the negative log-likelihood loss function on the training and holdout validation set of an MDN model. Figure 9 displays the MAE loss function on the same train and holdout validation set of a Dropout model. No model reached the limit of 50 epochs during training.

3.2. Results Arrival flights

In this section, the results of predicting flight delay for arrival flights are discussed. First, the MVE and MDN models' results will be presented, from which the best performing MDN model will be selected. Second, the Dropout model for different dropout rates will be treated. Finally, the best performing models, including the Random Forest model, will be compared to find the best overall performing algorithm for predicting probabilistic flight delay.

3.2.1. Results Arrival flights MVE and MDN methods

All MDN models have the same architecture of five layers, with 200 nodes. The only difference is the number of normal components in the predicted distribution. Figure 10 displays the Actual delay in predicted interval performance metric for interval sizes ranging from 2 minutes till 90 minutes. The statistical method is included in the plot as well to serve as a baseline. Figure 11 displays the same performance but given as deviation from the statistical method, as this better visualizes the differences in performance of the MVE and MDN methods.

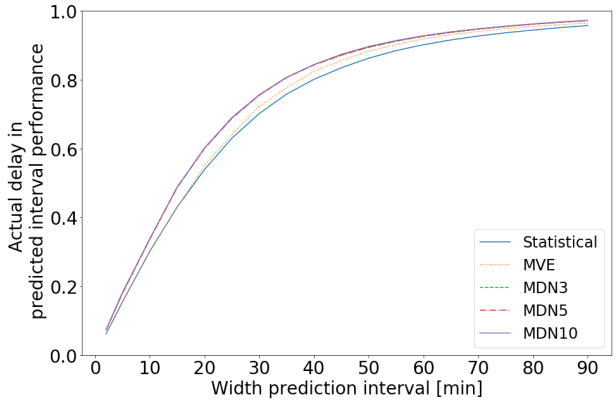


Figure 10: Actual delay in predicted interval performance for MVE and MDN arrival models

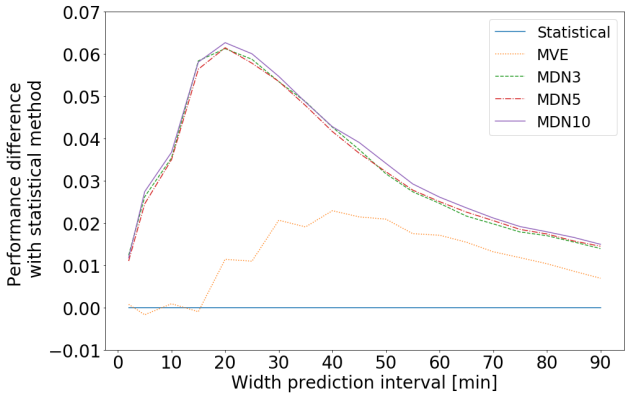


Figure 11: Actual delay in predicted interval performance relative to statistical method for MVE and MDN models

From Fig. 11 different observations can be made. First, all MDN models outperform the MVE model and the statistical method. Second, the three different MDN methods (MDN3, MDN5, MDN10) are very alike in terms of performance and follow the same general shape. The maximum difference between the statistical method and an MDN model is found for an interval of 20 minutes, where the MDN10 method outperforms the statistical method with 6.3 percent point.

The scores of the MDN models on the second performance metric, probability mass in actual delay interval, are displayed in Fig. 12 and Fig. 13. Figure 12 shows the absolute score of the models, while Fig. 13 shows the scores relative to the statistical method, to better visualize the difference in performance between the models.

From Fig. 12 it is seen that the MDN methods outperform the statistical and MVE method for all interval widths. The performance of the 3 different MDN methods are comparable, but the performance slightly increases for models with more normal components, as is seen in Fig. 13. The MDN10 model obtains the highest difference with the statistical method of 9.5 percent point for an actual delay interval of 30 minutes. The MVE method has a different performance behaviour relative to the statistical method. Its performance is similar to the statistical method till intervals of 20 minutes, whereafter it starts outperforming the statistical method.

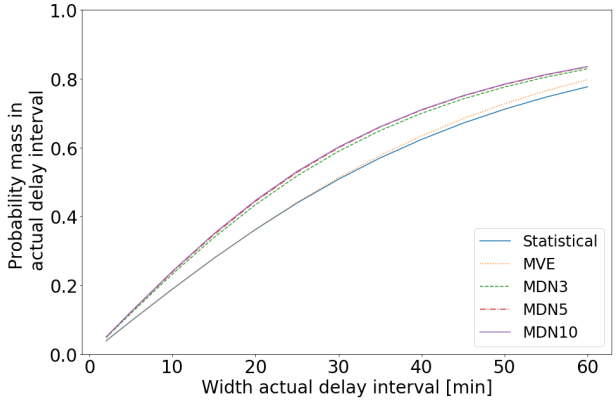


Figure 12: Probability mass in actual delay interval performance for MVE and MDN arrival models

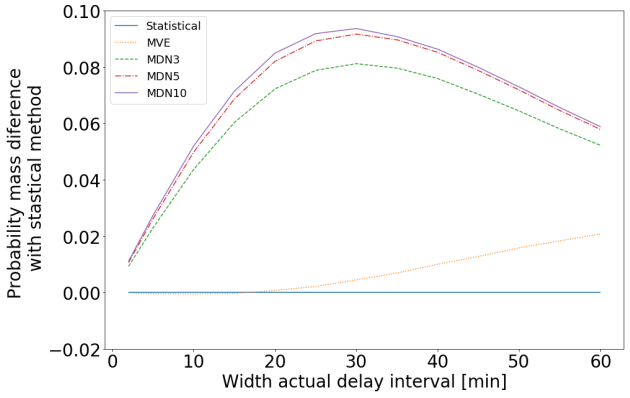


Figure 13: Probability mass performance relative to statistical method for MVE and MDN arrival models

3.2.2. Results Arrival flights Dropout methods

The performance of Actual delay in predicted intervals of the dropout models are visualized in Fig. 14 and Fig. 15, for various dropout rates in the predictions phase. Note that all dropout models make use of the same trained neural network; their only difference is a different dropout rate when making predictions. From Fig. 15 it is seen that there is no single dropout rate model which outperforms the other dropout rate model for all the considered intervals. The dropout 0.1 model performs best for intervals of 40 minutes and wider. The 0.3 and 0.5 dropout models have comparable performance till intervals of 20 minutes, where they both obtain the maximum difference with the statistical method of around 5.2 percent point. For interval larger than 20 minutes, the 0.5 dropout model outperforms the 0.3 dropout model. The largest considered dropout rate model of 0.7 under performs the other dropout models consistently, it only obtains a maximum difference with the statistical method of 3.8 percent point for an interval width of 25 minutes.

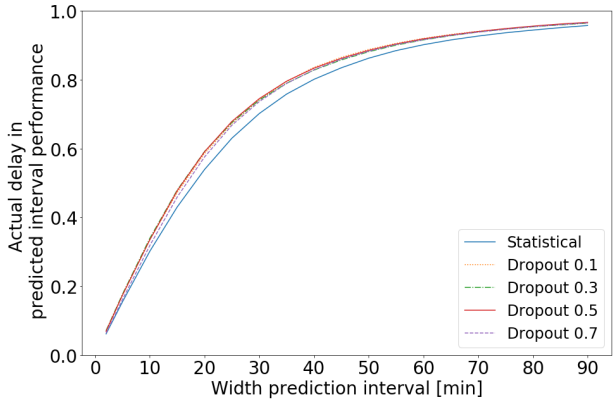


Figure 14: Actual delay in predicted interval performance for Dropout arrival models

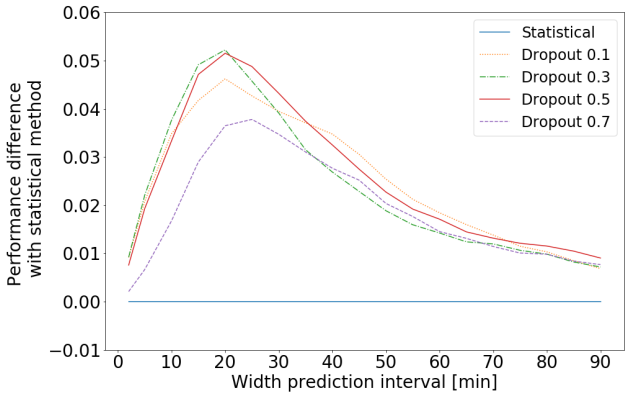


Figure 15: Actual delay in predicted interval performance relative to statistical method for Dropout models

The probability mass in actual delay interval performance of Dropout models is visualized in Fig. 16 and Fig. 17. It is clearly seen that an increase in dropout rate results in a decrease in performance. This drop in performance becomes bigger between dropout models as the dropout rate is increased. For a dropout rate of 0.7, the performance becomes even less than that of the statistical method.
The dropout 0.5 model is outperformed by the 0.1 and 0.3 model in terms of probability mass in actual delay interval, but it terms of Actual delay value in interval performance is has the best overall performance for the different interval sizes. As this is the leading metric, the dropout 0.5 model is selected as the best performing arrival dropout model.

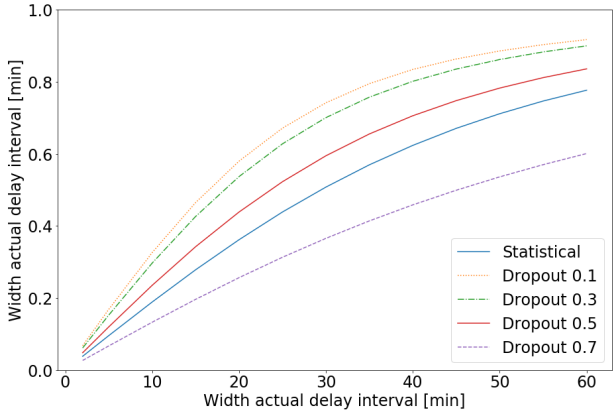


Figure 16: Probability mass in actual delay interval performance for Dropout arrival models

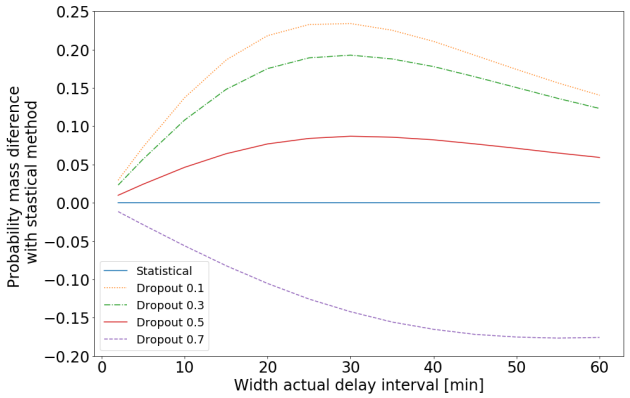


Figure 17: Probability mass performance relative to statistical method for Dropout arrival models

3.2.3. Results Arrival flights best performing methods

The following models are selected as best performing per model type: MVE, MDN10, Dropout 0.5 and Random Forest. Their actual delay in predicted interval performance are visualized in Fig. 18 and Fig. 19. The precise numerical values are given in Table 12 of appendix A.

From the relative performance to the statistical method plot in Fig. 19 it is seen that the MDN10 model outperforms all other machine learning methods for all interval widths. At an interval length of 20 minutes, the highest difference with the statistical method is obtained of 6.3 percent point. The Dropout 0.5 model is the second best performing model, which has the same general shape as the MDN10 but scores consistently lower. The Random Forest model has the same general shape as the MDN10 and Dropout models, where its maximum difference of 3.9 percent point with the statistical method is obtained for an interval width of 20 minutes. For larger intervals the performance starts to decrease relative to the statistical method. The MVE model has a different general shape than the other machine learning methods. It starts outperforming the statistical method for intervals of 15 minutes and wider, where the max difference is obtained of 2.3 percent point for an interval width of 40 minutes.

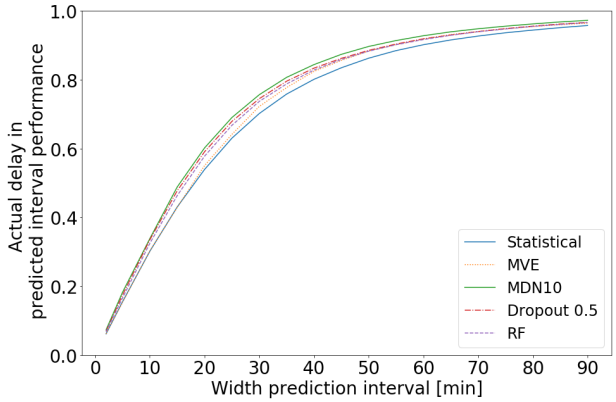


Figure 18: Actual delay in predicted interval performance for best performing arrival models

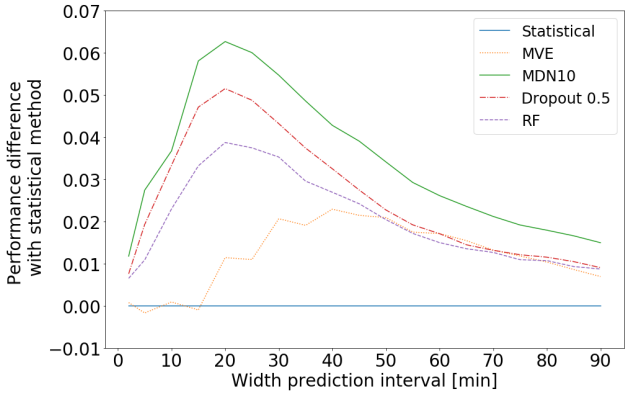


Figure 19: Actual delay in predicted interval performance relative to statistical method for best performing models

The probability mass in actual delay interval performance of the best performing models is given in Table 14 of appendix A, which are visualized in Fig. 20 and Fig. 21. From the performance relative to the statistical method in Fig. 21 it is seen that the Random Forest model is the best performing model for all actual delay intervals of width ranging from 2 till 60 minutes. For an interval of 30 minutes, the Random Forest model obtains its maximum difference with the statistical method of 0.157 probability mass. The second-best performing model is the MDN10 model, which closely followed by the Dropout 0.5 model. The MDN10 model obtains its maximum difference with the statistical method also for an interval of 30 minutes, but only with a difference of 0.092 probability mass. The MVE model is the worst performing machine learning method and only starts outperforming the statistical method slightly for intervals of 20 minutes and wider.

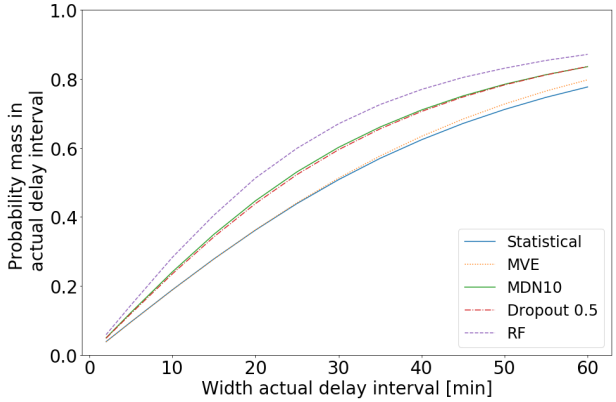


Figure 20: Probability mass in actual delay interval performance for best performing arrival models

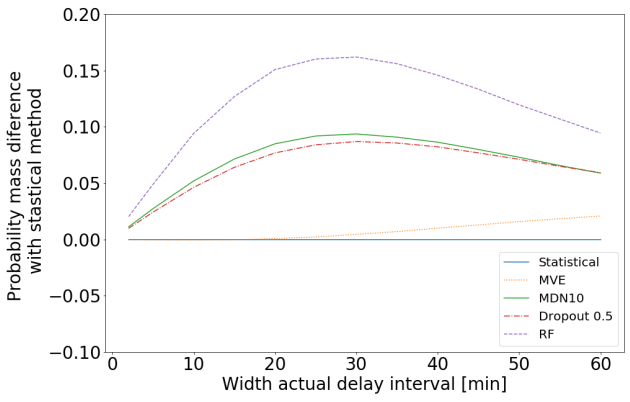


Figure 21: Probability mass performance relative to statistical method for best performing arrival models

To obtain more insight in the performance metric of Actual delay in predicted interval, the correct and false predicted flights are visualized based on their actual delay. This is done per model with a so-called 'correct-false predictions' histogram. This histogram displays all flights based on their actual delay value and indicates whether they were predicted correctly in the interval or not. For example, Fig. 22 displays the statistical method histogram for a 30 minutes interval. The statistical method predicts the same interval of [-19,11) for every flight in the test set, all flights with a delay value within this interval are predicted correctly. On the contrary, all flights which have a delay outside of this interval are predicted incorrectly.

These same 'correct-false predictions' histograms are also created for the four best performing methods, which are displayed for the MVE model in Fig. 23, for the MDN10 model in Fig. 24, for the Dropout 0.5 model in Fig. 25 and lastly for the Random Forest model in Fig. 26.

For the MDN10 model in Fig. 24, it is observed that relative to the other models the MDN10 has less false predictions for early arrivals (the red graph from -50 till 0). It has however slightly more false predictions for later arrivals (red graph from 0 till 50) compared to for example the Dropout 0.5 method in Fig. 25. Interestingly, the Random Forest model, predicted the most flights with a delay of -5 till 0 minutes correctly (red graph area between -5 and 0 is the smallest). No other machine learning method predicted this much correct flights in this interval, even the MDN10 model did not.

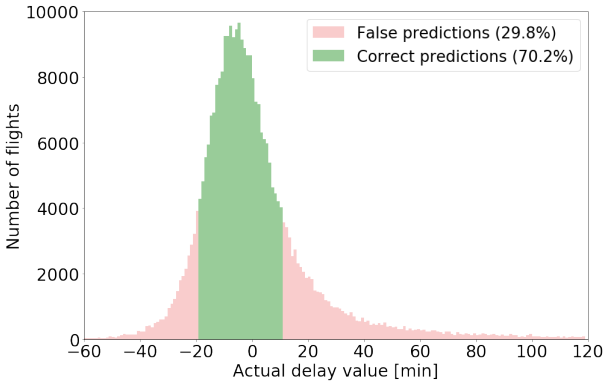


Figure 22: Correct and False predicted flights of Statistical arrival method for 30 minutes interval width

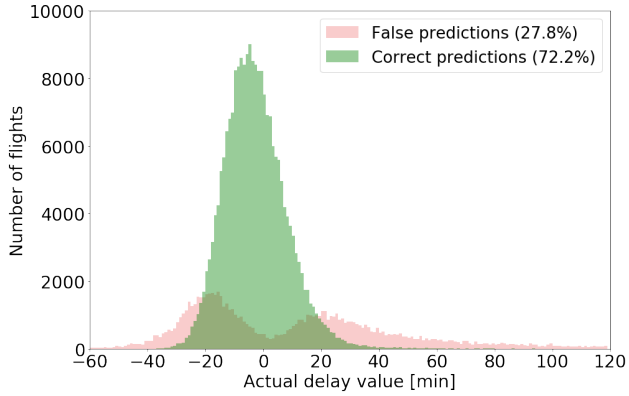


Figure 23: Correct and False predicted flights of arrival MVE model for 30 minutes interval width

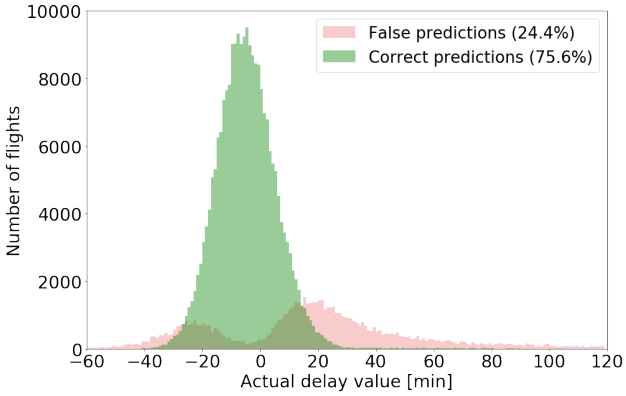


Figure 24: Correct and False predicted flights of arrival MDN10 model for 30 minutes interval width

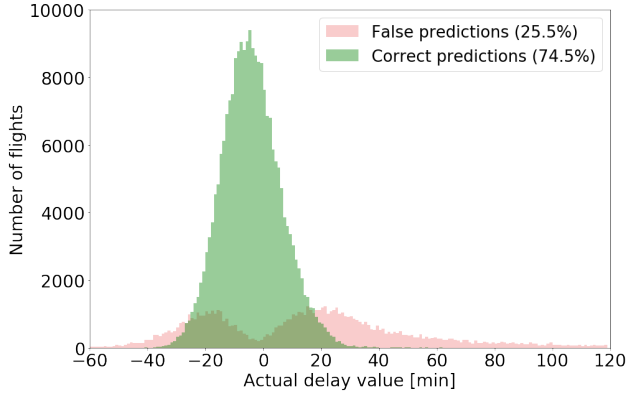


Figure 25: Correct and False predicted flights of arrival Dropout 0.5 model for 30 minutes interval width

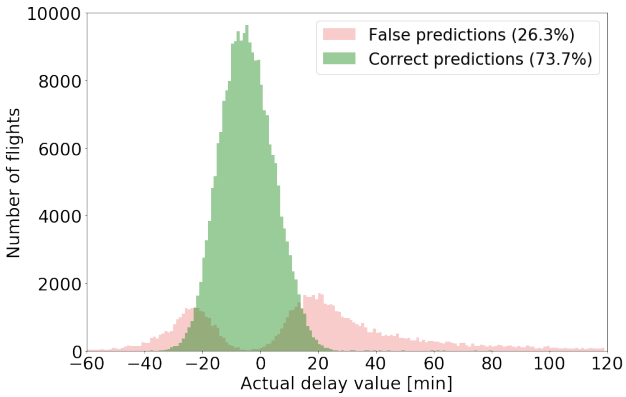


Figure 26: Correct and False predicted flights of arrival Random Forest model for 30 minutes interval width

Each model predicts a distribution, which can be used to compute a point prediction. For the MVE and MDN model, the mean of the distribution can be computed. For the Dropout model, the dropout rate is set to zero to obtain point predictions. Instead of the Random Forest trees, the prediction of the Random Forest itself is chosen, which is the mean of the predictions of all the 1000 trees. With these point predictions, the MAE and MSE error are computed for the arriving models, which are given in Table 10 (Table 11 is for departure models, which will be discussed later). Table 10 shows that all machine learning methods outperform the statistical method. The best performing method in terms of MAE score is the Dropout method with a score of 12.23 minutes. In terms of RMSE score, the MDN10 model obtains the best score of 18.78 minutes.

Table 10: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) score of point value predictions of all methods of arrival flights

	Statistical	MVE	MDN 10	Dropout rate 0.5	Random Forest
MAE	14.59	12.85	12.52	12.23	12.94
RMSE	21.42	19.07	18.78	19.05	19.87

Table 11: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) score of point value predictions of all methods of departure flights

	Statistical	MVE	MDN 3	Dropout rate 0.5	Random Forest
MAE	11.79	10.52	9.97	9.59	9.99
MSE	17.75	16.33	15.98	16.62	16.97

3.3. Results Departure flights

This section will discuss the results obtained from the departure models. The same structure is used for presenting the results as was done for arrival flights in Section 3.2.. The same type of graphs will be displayed with the absolute performance of the metrics and the performance relative to the statistical departure method.

3.3.1. Results Departure flights MVE and MDN methods

The MDN departure models all have the same architecture of five layers, with 250 nodes. They only differ in the number of normal components in their respective distributions. Fig. 27 displays the Actual delay in the predicted interval performance metric, where Fig. 28 displays the same performance but relative to the statistical departure method.

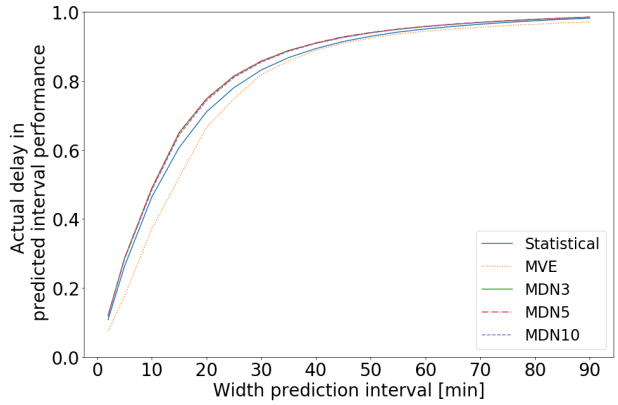


Figure 27: Actual delay in predicted interval performance for MVE and MDN departure models

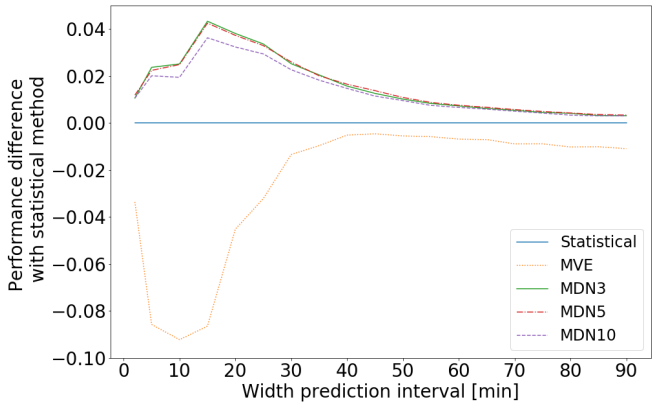


Figure 28: Actual delay in predicted interval performance relative to statistical method for MVE and MDN models

Fig. 28 shows that all MDN models outperform the statistical method. The different MDN models' results are very comparable, where only the MDN10 model scores slightly less for intervals of 5 till 40 minutes. The performances of the MDN3 and MDN5 are almost identical in terms of Actual delay in the predicted interval. The maximum difference between the statistical method and MDN models is reached at an interval of 15 minutes, where the difference is slightly above 4 percent point for both the MDN3 and MDN5 model. The MVE model never outperforms the statistical method, where its performance takes a deep dive for intervals between 5 and 20 minutes relative to the statistical method. Its worst performance of -9.2 percent point relative to the statistical method is obtained for an interval width of 10 minutes.

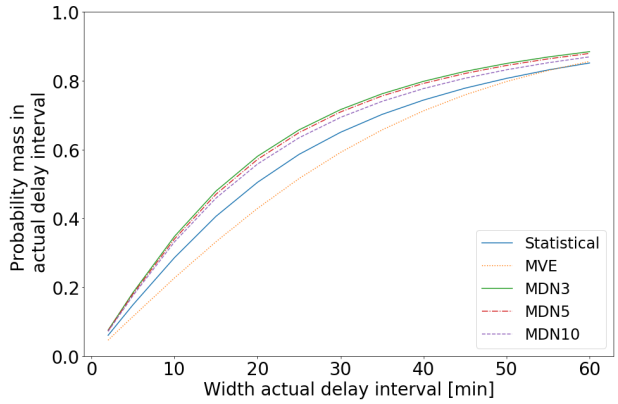


Figure 29: Probability mass in actual delay interval performance for MVE and MDN departure models

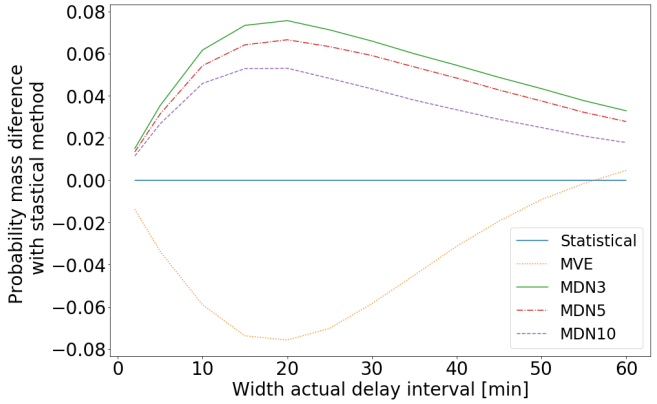


Figure 30: Probability mass performance relative to statistical method for MVE and MDN departure models

The scores of the departure MDN models on the second performance metric, probability mass in actual delay interval, are displayed in Fig. 29 and Fig. 30, which displays the score relative to the statistical method. From Fig. 29 it is seen that all three MDN models outperform the statistical method, but that the MVE model consistently performs worse than the statistical method (except for the interval of 60 minutes). The MDN3 model is the best performing model, followed by the MDN5 and then the MDN10. An increase in the number of normal components of the model results in a decrease in performance, as is seen in Fig. 30. The MDN3 model is selected as the best performing MDN model. It outperforms the other MDN models in terms of probability mass in actual delay interval. It shares the best performance in terms of actual delay in predicted interval with the MDN5 model (Fig. 28).

3.3.2. Results Departure flights Dropout methods

The performance of Actual delay in predicted intervals of the departure dropout models are visualized in Fig. 31 and Fig. 32. Note that all dropout models make use of the same reference neural network, trained on the departure train set. Their only difference is a different dropout rate when making predictions. The dropout 0.5 model is the best performing model overall for all considered interval width, as can be seen in Fig. 32. The dropout 0.3 model performs only better for intervals till 10 minutes. The dropout 0.3 and 0.5 model both have a maximum difference with the statistical method of almost 2 percent point for interval widths of respectively 10 and 20 minutes. The dropout 0.1 model only reaches a difference of 0.3 percent point with the statistical method for on interval width op 25 minutes. For the majority of interval width however, the dropout 0.1 is outperformed by the statistical method. The model with the highest dropout rate, the dropout 0.7 model, is the worst performing model. It is consistently outperformed by all other dropout models as well as the statistical method.

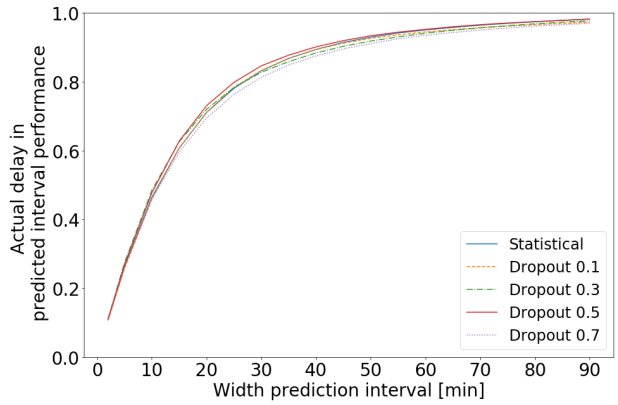


Figure 31: Actual delay in predicted interval performance for Dropout departure models

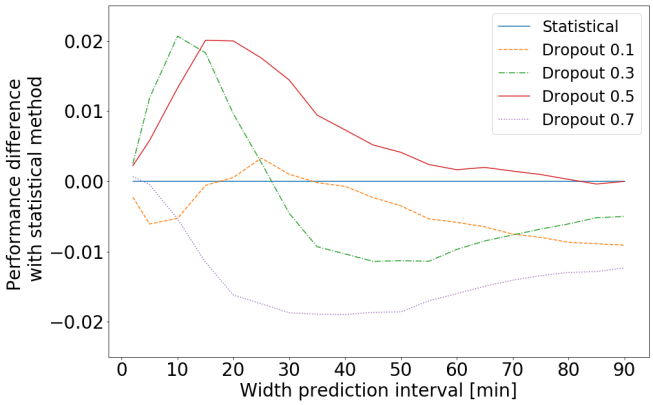


Figure 32: Actual delay in predicted interval performance relative to statistical method for Dropout models

The probability mass in actual delay interval performance of the departure Dropout models are visualized in Fig. 33 and figure Fig. 34. Fig. 34 shows that an increase in the dropout rate results in a decrease in terms of performance. Both dropout 0.1 and 0.3 model outperform the statistical method, while the dropout 0.5 model only outperforms the statistical method till interval of 30 minutes. The dropout 0.7 model never outperforms the statistical method.
The dropout 0.5 model is chosen as best performing departure dropout model, based on that it outperforms all other dropout models in terms of Actual delay value in interval performance, as was seen in Fig. 32.

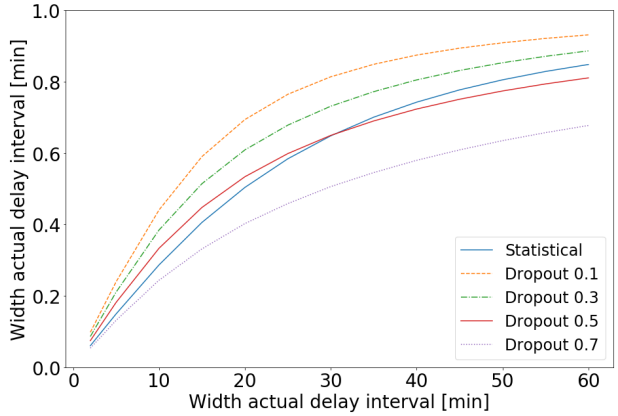


Figure 33: Probability mass in actual delay interval performance for Dropout departure models

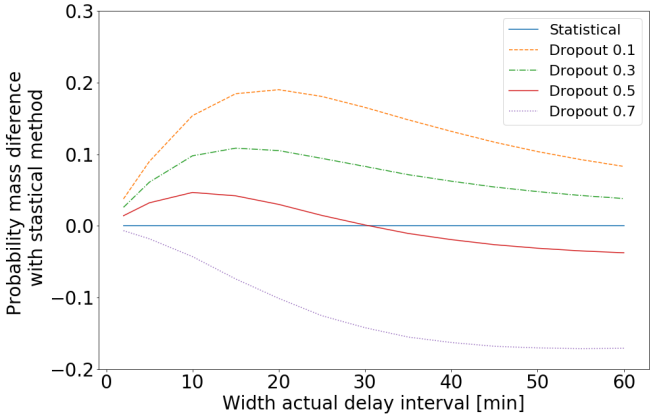


Figure 34: Probability mass performance relative to statistical method for Dropout departure models

3.3.3. Results Departure flights best performing methods

The following departure models were selected as best performing per model type: MVE, MDN3, Dropout 0.5 and Random Forest. Their Actual delay in predicted interval performance are visualized in Fig. 35 and Fig. 36. The precise numerical values are given in Table 13 of appendix A.

The relative performance to the statistical method plot in Fig. 36 shows that the MDN3 model outperforms all other machine learning methods for all interval widths. At an interval length of 15 minutes, the highest difference with the statistical method is obtained of 4.2 percent point. The Random Forest and Dropout 0.5 model have comparable performances, but the Random Forest slightly outperforms the Dropout 0.5 on most of the considered interval widths. After intervals with a width of 60 minutes the Random Forest and Dropout 0.5 model have a performance similar to that of the statistical method. The MVE method never outperforms the statistical method, where its worst performance of -9.2 percent point compared to the statistical method is obtained at an interval of 10 minutes.

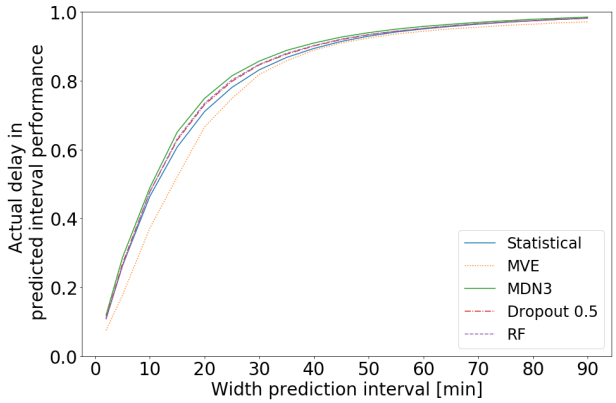


Figure 35: Actual delay in predicted interval performance for best performing departure models

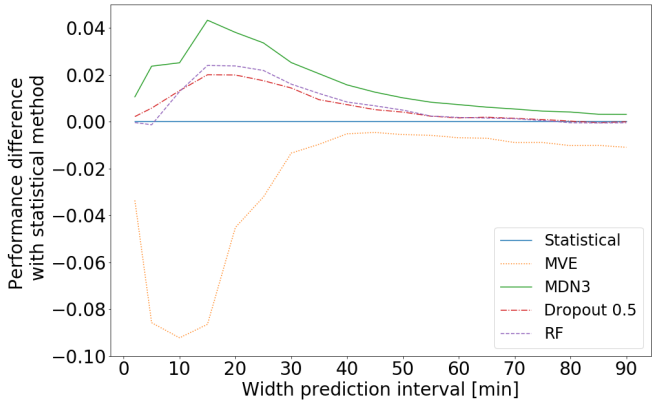


Figure 36: Actual delay in predicted interval performance relative to statistical method for best performing models

The probability mass in actual delay interval performance of the best performing departure models is given in Table 15 of appendix A, which are visualized in Fig. 37 and Fig. 38.
The performance relative to the statistical method in Fig. 38 shows that the Random Forest model is the best performing model on all actual delay intervals. For an interval of 20 minutes, the Random Forest model obtains its maximum difference with the statistical method of 0.174 probability mass. The second-best performing model is the MDN3 model, followed by the Dropout 0.5 model. The MDN3 model outperforms the statistical method for all actual delay intervals, while the Dropout 0.5 under performs the statistical method for actual delay interval of 30 minutes and wider. The MVE model is the worst performing machine learning model which only slightly outperforms the statistical method for the actual delay interval of 60 minutes.

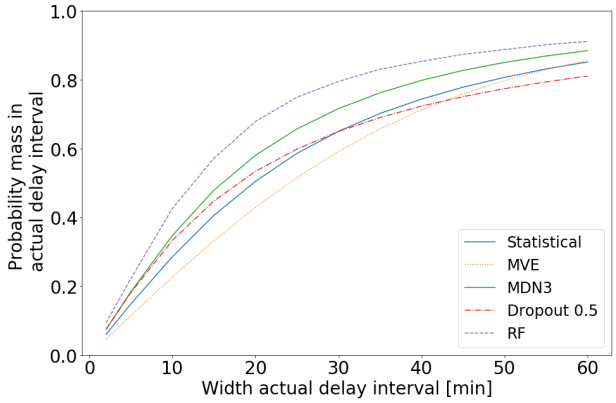


Figure 37: Probability mass in actual delay interval performance for best performing departure models

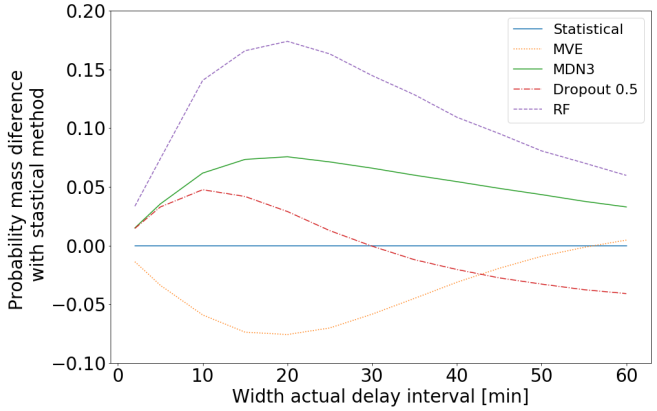


Figure 38: Probability mass performance relative to statistical method for best performing departure models

To obtain more insight into the performance metric of actual delay in the predicted interval of the departure models, the correct and false predicted flights are visualized based on their actual departure delay value. These histograms display all flights based on their actual departure delay values and whether they were predicted correctly or not. For example, in

Fig. 39 the statistical method for an interval width of 20 minutes is displayed. As the statistical method predicts the same interval of [-5,15) for every flight in the test set, all flights with delay values within this interval are predicted correctly. On the contrary, all flights which have a delay outside of this interval are predicted incorrectly.

These 'correct-false predictions' histograms, for a considered interval width of 20 minutes, are also created for the four best performing models, which are displayed for the MVE model in Fig. 40, for the MDN10 model in Fig. 41, for the Dropout model in Fig. 42 and lastly for the Random Forest model in Fig. 43. Comparing the best performing model, the MDN3 in Fig. 41, to the other models shows that relative to the other models the MDN3 has less false predictions for early departures (the red graph from -15 till 0). In terms of false predictions for later departures (red graph from 0 till 30) it has comparable performance to that of the Random Forest model in Fig. 43 and even outperforms the Dropout 0.5 model, as is seen in Fig. 42. Interestingly, the Dropout 0.5 model predicts no flights correct with a departure delay of 20 minutes or more, for the considered interval width of 20 minutes. It however predicts all flights correct, which have a departure delay of -1 till 8 minutes. No other method performs so well around the scheduled departure time as the Dropout 0.5 model. Only the Random Forest model comes close with a few false predicted flights with a departure delay of 0 till 10 minutes (red graph area between 0 and 10 in Fig. 43).

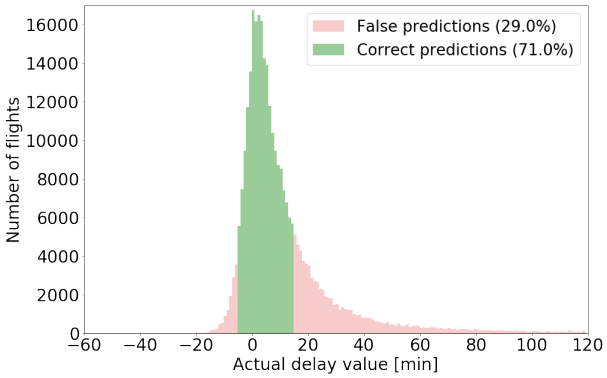


Figure 39: Correct and False predicted flights Statistical departure method for 20 minutes interval width

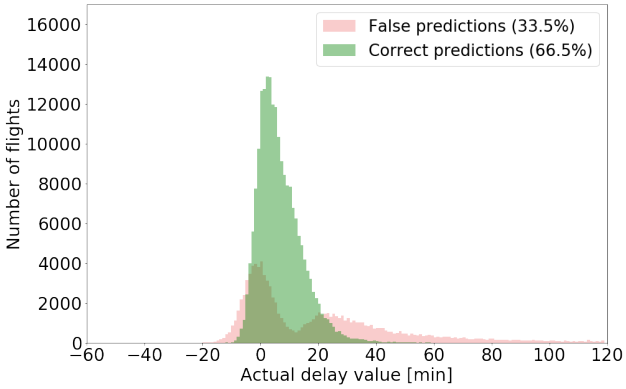


Figure 40: Correct and False predicted flights of departure MVE model for 20 minutes interval width

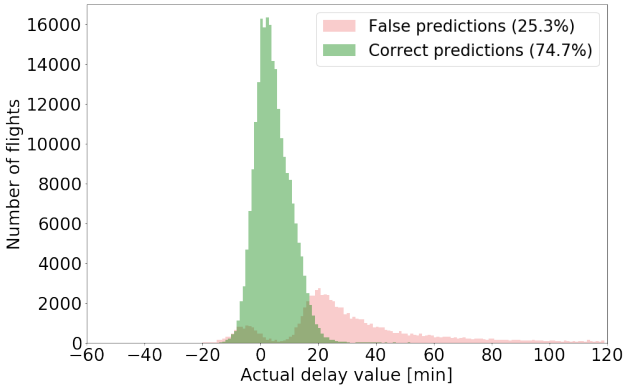


Figure 41: Correct and False predicted flights of departure MDN3 model for 20 minutes interval width

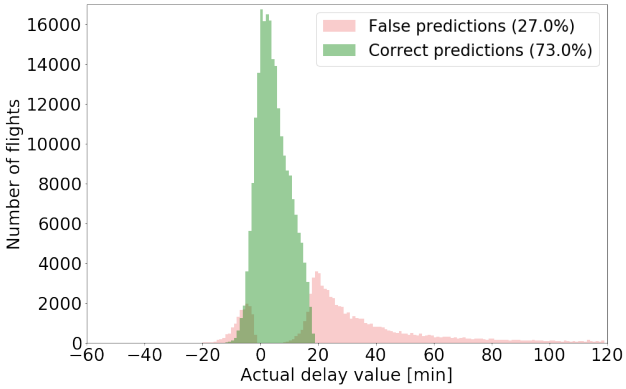


Figure 42: Correct and False predicted flights of departure Dropout 0.5 model for 20 minutes interval width

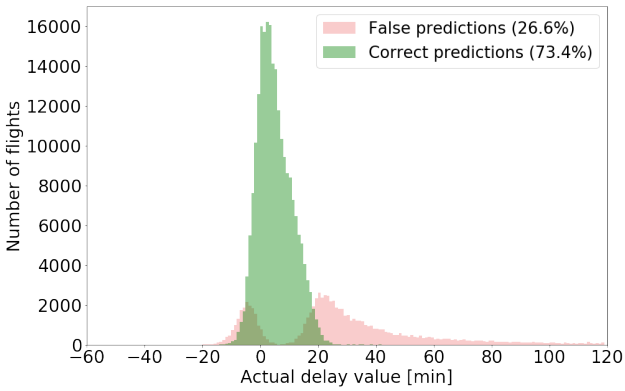


Figure 43: Correct and False predicted flights of departure Random Forest model for 20 minutes interval width

For each departure model, its distribution is transformed to a point prediction, which is used to compute the MAE and RMSE metrics, which are displayed in Table 11. The best performing method in terms of MAE score is the Dropout 0.5 method with a score of 9.59 minutes. In terms of RMSE score, the MDN3 model obtains the best score of 15.98.

4. Discussion

In this section, the key findings of this research are presented and discussed. First, the key findings of the arrival flights are discussed, followed by the key findings of the departure flights. Afterwards, the differences between continuous and empirical distributions will be discussed. Lastly, future research direction and recommendations will be given.

4.1. Key findings Arrival Flights

The first important finding in this research is that machine learning models are capable of outperforming a statistical method in predicting probabilistic flight delay of arrival flights. The best model to predict probabilistic flight delay of arrival flights is the MDN10 model. This model performs the best in terms of Actual delay in predicted intervals for all interval widths ranging from 2 to 90 minutes. It outperforms other machine learning models mainly by predicting more early arrivals correctly. This can be explained by the fact that the MDN10 is capable of modelling the general shape of delay, because it can use a mixture of 10 normal distributions.

The dropout 0.5 model is the second-best performing model in terms of Actual delay in predicted interval score. However, it scores higher in terms of MAE score, 12.23 minutes over the 12.52 of the MDN10 method. This can be explained by the fact that the reference Neural Network used for the dropout model was trained specifically to minimize the MAE error. With the optimal dropout rate of 0.5, it is capable of predicting distributions, which score well in terms of the Actual delay in predicted interval score. However, the dropout 0.5 model does not outperform the MDN10 model. In Fig. 24 and Fig. 25 the 'correct-false predictions' histograms are displayed for a prediction interval of 30 minutes. The difference in performance between the MDN10 and dropout 0.5 model is caused by the MDN10 predicting more early arrivals correct than the dropout 0.5 model.

This difference can be illustrated with Fig. 44 and Fig. 45. Both figures display the predicted distributions of an arriving flight from KLM coming from Hamburg with a scheduled arrival time of 19:40 in the evening. Fig. 44 shows the distribution prediction of the dropout 0.5 model. It is seen that the predicted distribution has a heavier right tail with more mass assigned to late arrivals. Due to this, the early arrival of -9 minutes of the flight doesn't fall within the predicted interval. Fig. 45 displays the prediction of the same flight but for the MDN10 model. The MDN10 model assigns more mass to early arrivals, compared to the dropout 0.5 model, which causes that the correct interval is predicted in which the actual delay falls. This behavior of the dropout 0.5 model, which is assigning too little mass to predict an early arrival correct is seen for a multitude of flights in the test set, which causes the lower performance compared to the MDN10 model.

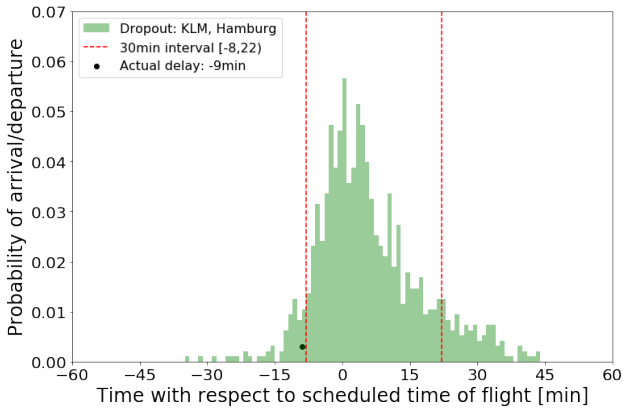


Figure 44: Example of False early arrival prediction of dropout model

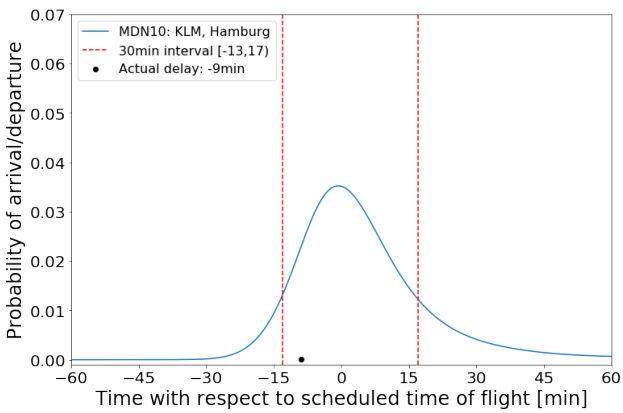


Figure 45: Example of Correct early arrival prediction of MDN10 model

The MVE model is consequentially outperformed by all the MDN models (MDN3, MDN5, MDN10) which can model flight delay as a combination of multiple normal distributions, instead of a single normal distribution. This means that a single normal distribution is not suited to represent arrival flight delay. This can be explained according to the observation at the beginning of this paper in Fig. 1, which shows that arriving flights have a 'heavier right tail' with late arriving flights. In other words, there are more late arriving flights, then early arriving flights.

However, the MVE method is still capable of outperforming the statistical method for intervals of 20 minutes and wider. This means that it is capable of learning which features influence delay and use that information to predict whether a flight will be an arriving/departing early or late. Because the MVE outputs a single normal distribution, only the mean can be used to decide which interval is chosen for the Actual delay in predicted interval metric. Due to the symmetry of a normal distribution, the shape has no added information to base the choice of this interval on.

The predicted mean of the MVE model is directly used in the calculation of the MAE metric. Comparing the MAE error of the MVE method (12.85) with the MDN10 method (12.52) shows that there is only a small difference of 0.33 minutes. This means that in terms of point predictions, the MVE does not perform significantly less than the MDN10 model. This indicates that when the shape of a distribution is needed for a performance metric, for example, the two performance metrics used in this research, the MVE performs less than models with a more general shape (MDN10) because its shape is fixed to that of a single normal distribution.

4.2. Key findings Departure Flights

The first important finding of the departure models is that the MDN3, the Dropout and RF model can outperform a statistical method. The MVE method however never outperforms the statistical method, which means that a single normal distribution is not suited to represent departure flight delay. This can be explained with Fig. 1, where the delay departure statistics show a skewed distribution with a heavier right tail with more late departures then early departures.

From Fig. 38 it is seen next to that the MVE model never outperforms the statistical method, it also took a deep dive in terms of performance of Actual delay value in predicted interval for small intervals (0 till 20 minutes). For an interval width of 10 minutes, it scored -9.2 percent point compared to the statistical method. This reduction in terms of performance of the MVE model for small intervals can be explained with Fig. 46 and Fig. 47. Both figures display the same departure flight from KLM to Gdansk departing at 20:50 from Schiphol.

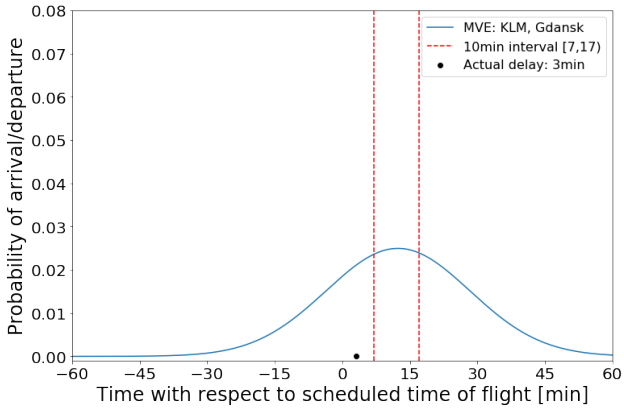


Figure 46: Example of False MVE prediction for prediction interval of 10 minutes

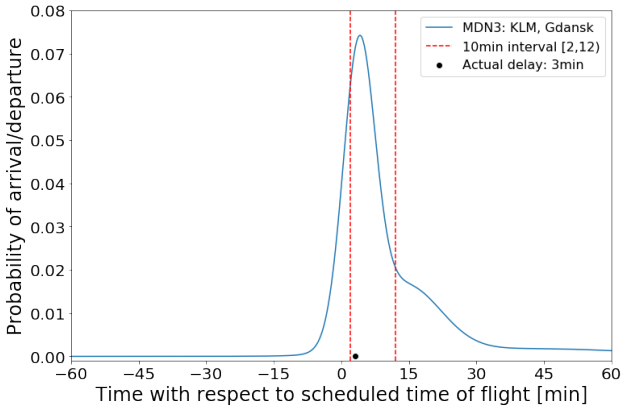


Figure 47: Example of Correct MDN3 prediction for prediction interval of 10 minutes

Fig. 46 displays the predicted distribution of the MVE model, for which an incorrect interval is chosen. Fig. 47 displays the predicted distribution of the MDN3 model, which predicts a correct interval. The MVE method predicted the wrong interval, due to the fact that its shape is fixed to a normal distribution. Which results in that it is not capable of predicting the combination of a steep descent on the left-hand side of the distribution and a heavier right tail, which the MDN3 model is capable of. The MVE model learns that there is a higher chance of a late departure flight than an early departure flight, which is seen in the departure delay statistics of Fig. 1. Therefore, the model chooses to predict the mean of the normal distribution more to the right of the scheduled time, instead of predicting close to the scheduled time of departure. In Fig. 46 for example, the predicted mean is 12 minutes. A prediction of the mean close or equalling the scheduled time would result in assigning the same probability to an early departure as a late departure, due to the normal distribution's symmetrical shape. Because early departures happen less than late departures, the MVE predicts the mean more to the right. For smaller intervals of, for example, 10 minutes, this results in delays close the scheduled time not being included. The lower bound of the predicted MVE interval starts at 7 minutes, while the lower bound of the MDN3 model starts at 2 minutes. When the interval size increases, the lower bound of the interval will move to the left, resulting in the delay values close to the scheduled time of departure being included. This causes the performance relative to the statistical method to start increasing for larger interval width. For an interval of 30 minutes, for example, the difference between the MVE and statistical method is reduced to only -1.6 percent point.

The MDN3 model is the best performing model in terms of Actual delay in predicted interval score. It outperforms other machine learning models by predicting more early departures correct compared to the other models (which is also the case for the MDN10 arrival model). It does this by modelling the general shape of departure delay as a mixture of three normal distributions. Interestingly, increasing the number of normal distributions does not result in improved performance. This indicates that MDN models with more than three normal components start overfitting on the training set, causing a decrease in the performance on the test set. This trend is also seen for the metric probability mass in the actual delay interval, which decreases for MDN models with more normal components.

Although the MDN3 model was the best performing in terms of Actual delay in predicted interval performance, the Random Forest model scored best in terms of probability mass in actual delay interval. It outperformed all other machine learning models significantly, including the MDN3 model, by almost 0.1 probability mass for an actual delay interval width of 20 minutes as was seen in Fig. 38. This is also seen for arrival models, where the Random forest scored best in terms of probability mass in actual delay interval, but was outperformed by the MDN10 and dropout 0.5 model in terms of Actual delay value in interval performance metric.

When a model scores high in terms of probability mass in the actual delay interval score, it would suggest that the model's distribution is certain about their predictions and were also correct. The model is sure about its prediction, hence a concentrated probability mass. Secondly, its prediction is correct, hence the high probability mass in the actual delay interval. It would then seem logical that a model which scores better in terms of probability mass in actual delay interval, would also score better in terms of actual delay value in interval. But this turns out not to be the case for the Random Forest model compared to the MDN models.

This behavior can be explained with examples of distribution predictions of the departure Random Forest model and the MDN3 departure model. The empirical distribution of the Random Forest model can be concentrated, as can be seen in Fig. 48 for a departure flight from easyJet going to London Luton. The distribution prediction of the same flight of the MDN3 model is given in Fig. 49. It is seen that both methods predicted the correct 30 minutes interval in which the actual delay value lies (interval with red lines). In terms of probability mass in actual delay interval (20 minutes interval with blue lines) the Random Forest model scores highest with a probability mass of 0.92, where the MDN3 model has a probability mass of only 0.58. This difference is due to the fact that the Random Forest model predicts a more concentrated distribution which is close to the actual delay value.

On the other hand, there are also examples of where this concentrated distribution results in a false predicted interval by the Random Forest model, seen in Fig. 50 for a departure flight operated by Vueling to Barcelona. Fig. 51 gives the prediction of the MDN3 model, which predicted a correct interval. Interestingly enough the Random Forest obtained a higher score for the 20 minutes actual delay interval with a probability mass of 0.58, while the MDN3 model only obtains a 0.4 probability mass score in the same interval. The Random Forest obtains concentrated predictions, which don't span the entire considered time axis. While the MDN3 model has less concentrated predictions, but span the entire considered time axis. This illustrates why a higher overall score in terms of probability mass in the actual delay interval does not necessarily mean a better performance in terms of actual delay in the predicted interval.

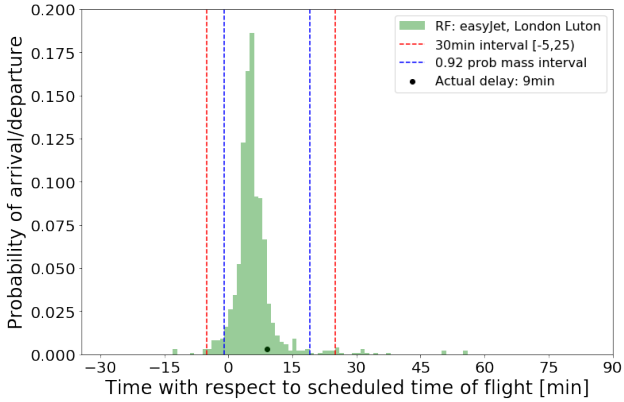


Figure 48: Example of correct, concentrated RF prediction for easyJet departure flight to London Luton

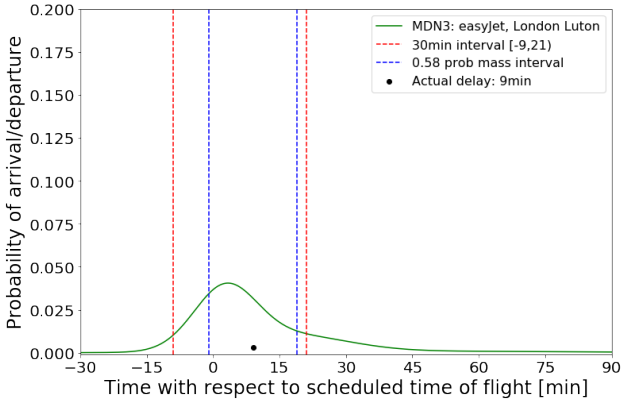


Figure 49: Example of correct, less concentrated MDN3 prediction for easyJet departure flight to London Luton

4.3. Key findings Continuous versus Empirical distributions

One of the fundamental differences between the evaluated models in this research is the difference in models which predict continuous distributions (MVE and MDN) and models that predict empirical distributions (Dropout and RF). They not only differ in the type of predictions they make, but also in the way they are trained.

The MVE and MDN method are trained using a negative log-likelihood function, which maximizes the probability that the delay of a sample flight comes from the predicted distribution. Via this way the model is trained to predict the shape of a delay distribution, instead of predicting a single point value. The empirical methods on the other hand are trained using a MAE loss function, which wants to minimize the absolute difference between the delay of the sample flight and the predicted delay value. This difference is seen in the predicted distributions as the continuous methods assign delay

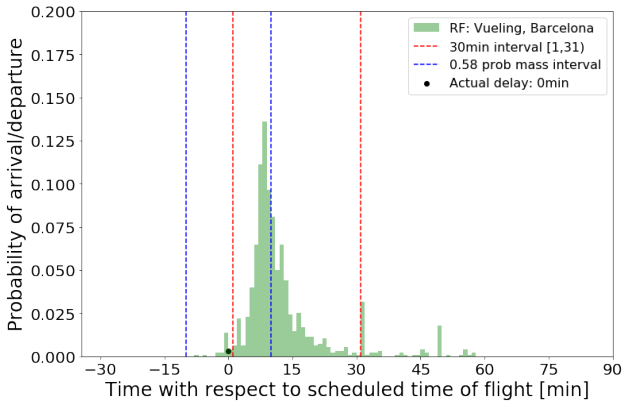


Figure 50: Example of false, concentrated RF prediction for Vueling departure flight to Barcelona

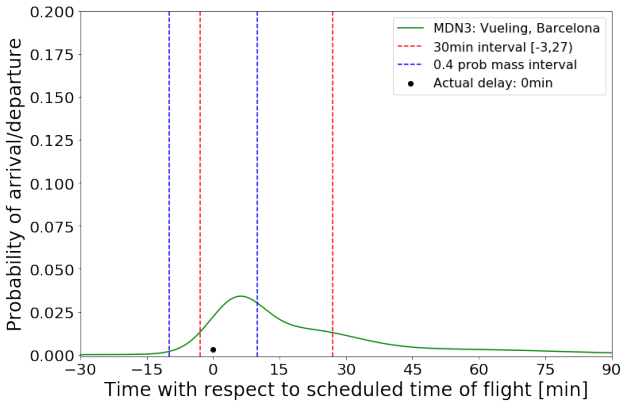


Figure 51: Example of correct, less concentrated MDN3 prediction for Vueling departure flight to Barcelona

probability mass over the full range of time steps (Fig. 49 and Fig. 51), while the empirical methods assign probability mass merely to a smaller section of the full range of time steps (Fig. 48 and Fig. 50). This is supported by the value of the average standard deviation of the predicted distributions. The average standard deviation of the departure RF model distributions is just below 10 minutes, while for the MDN3 departure model distributions it lies around 15 minutes.

The predicted point values which make up the empirical distributions are obtained by introducing noise to the model. The dropout model has random nodes being dropped in the prediction phase to obtain distributions. For the RF model, this is done during the training process by giving each tree a random subset of the training sets and a random selection of features. These empirical models are trained based on the MAE criteria, which focuses on minimizing the difference between the predicted point value and the actual delay. They are not trained specifically to predict the shape of a delay distribution as is the case for the MVE and MDN method. By introducing noise to the model to obtain distributions, an indication is given in how certain the model is over that specific point prediction. This explains why the empirical methods predict more concentrated distributions, while the predictions of the continuous methods range over the full range of the considered time axis.

This raises the question of why it is, that the continuous methods assign probability mass over the full range of time steps, but do not predict 'extreme-late' arrivals or departure (45+ minutes delayed) consistently correct. This is seen in the 'correct-false prediction' histograms of the best MDN10 arrival model in Fig. 24 and MDN3 departure model in Fig. 41. This is because the MDN models' predicted distributions never have enough probability mass around those extreme delayed values to choose such intervals. Fig. 52 gives an example of an 'extreme-late' arrival flight coming from Beijing operated by China Southern and Fig. 53 gives an example of an 'extreme-late' departure flight from KLM to Basel. Both MDN models know that there is a change of an extreme late arrival/departure, as they assign in both cases probability mass to 'extreme late' arrival/departure values. However the assigned probability mass is not enough to choose an interval in which the actual 'extreme-late' delay values lies. This indicates that the used features in this research are not powerful enough to predict those extreme late arrivals/departures consistently correct.

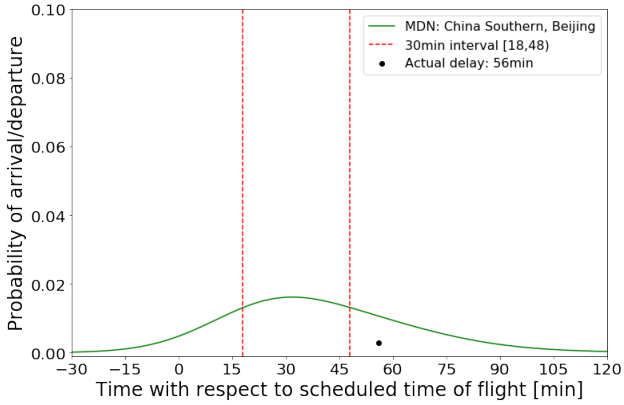


Figure 52: MDN10 prediction for 'extreme-late' arrival flight from China Southern from Beijing

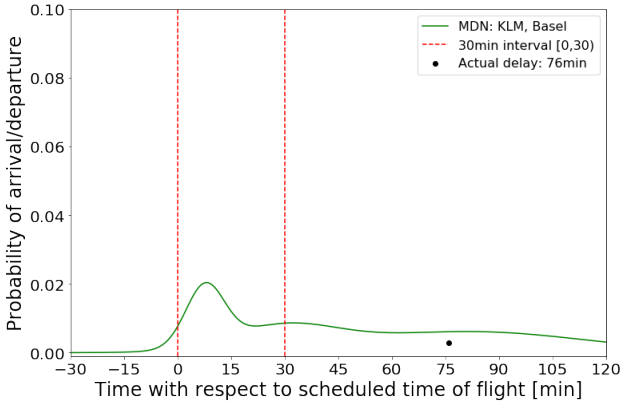


Figure 53: MDN3 prediction for 'extreme-late' departure flight from KLM to Basel

4.4. Future directions and recommendations

Based on the performed research in this thesis, directions for future work are defined to further improve the field of probabilistic flight delay predictions. First, more features should be searched for to predict extreme late arrival/departures correct with more certainty. A first suggestion would be to look into features that have predictive power for delay, such as the direction and speed of jet streams that influence the flight time of especially intercontinental flights.

A second future direction is using weather forecast data when making predictions. In this research, it is assumed that the weather forecast is accurate enough on a 1-day prediction horizon that represents the actual weather. It is important to understand the influence on the models' performance when using weather predictions, especially when using the models for real-life applications.

Additionally, the models itself can be improved in various ways. First, the loss functions used to train the models can be rewritten such that they are the same as how the model performance is evaluated. Secondly, for the empirical models, the Dropout and Random Forest models, more predictions than the 1000 used in this research per distribution can be made. This can result in more detailed distributions for individual flights which possibly span further over the considered time axis.

A future research implementation of the probabilistic flight delay distributions is to use them to optimize operational processes at an airport. A well-suited problem for this is the flight to gate assignment. van Schaijk and Visser [2017] presented a method for obtaining more robust flight-to-gate assignment schedules using so-called presence probabilities. The probabilistic flight delay distribution obtained in this research seem well suited for this method. This could showcase the usefulness of probabilistic flight delay predictions in optimizing operation processes at an airport by minimizing the number of gate conflicts.

5. Conclusion

To support the decision process of logistical operations at airports, this research has developed machine learning models to predict probabilistic fight delay of individual flights. These machine learning models are trained and tested on flight operational data from Amsterdam Schiphol airport and METAR weather data on a prediction horizon of one day. Four different models were trained and compared based on their predicted distributions with a simple statistical method. Two models were developed which predict empirical distributions, a Dropout neural network and a Random Forest model. The two other models predict continuous distributions by predicting the parameters of an assumed distribution directly. The Mean Variances Estimator predicts a single normal distribution, while the Mixture Density Network model predicts a mixture of multiple normal distributions.

Two interval-based performance metrics are suggested in this research to assess probabilistic flight delay predictions. The Actual delay value in predicted interval metric evaluates whether the actual delay value of a flight lies within a predicted interval which is chosen based on the predicted distribution of a model. The second performance metric, probability mass in actual delay interval, evaluates the probability mass of the predicted distribution in an interval around the actual delay value.

For both the arrival and departure delay, it is found that in terms of actual delay in predicted interval score, the Mixture Density Network models (MDN) perform best. Arrival delay is best modelled with an MDN model of 10 components, while departure delay is best modelled with only 3 components. Both methods outperform the statistical method on all considered interval widths. The MDN models outperform the other machine learning models by predicting more early arrivals and early departures correctly.

In terms of the probability mass in actual delay interval performance, the Dropout model scores best on both arrival and departure delay, followed by the Random Forest model. Both empirical models outperform both the continuous models for this metric. This is because empirical distributions are concentrated on a smaller section of the full considered time axis, while the continuous models assign probability mass over the full considered time axis.

The proposed probabilistic models in this research show that they can outperform a baseline statistical method on all considered interval widths, for both arrival and departure delay. Combined with the proposed performance metrics, this research can form a basis for further research on the topic of probabilistic flight delay for individual flights. The predicted distributions can be used to optimize operational processes at an airport such as the flight to gate assignment. As future work, it is suggested to extend the set of features enabling the prediction of extreme late arrivals and departures correctly with more certainty.

A Performance of best models

Table 12: Actual delay value in intervals performance for best performing arrival models

Interval width	Statistical	MVE	MDN 10	Dropout rate 0.5	Random Forest
2	0.061	0.062	0.073	0.069	0.068
5	0.155	0.153	0.182	0.174	0.166
10	0.302	0.303	0.338	0.335	0.325
15	0.430	0.429	0.488	0.477	0.463
20	0.539	0.551	0.602	0.591	0.578
25	0.630	0.641	0.690	0.679	0.668
30	0.702	0.722	0.756	0.745	0.737
35	0.758	0.777	0.807	0.796	0.788
40	0.801	0.824	0.844	0.834	0.828
45	0.835	0.856	0.874	0.862	0.859
50	0.863	0.884	0.897	0.885	0.883
55	0.885	0.902	0.914	0.904	0.902
60	0.902	0.919	0.928	0.919	0.917
65	0.916	0.931	0.939	0.930	0.929
70	0.927	0.940	0.948	0.940	0.940
75	0.936	0.948	0.956	0.949	0.947
80	0.944	0.955	0.962	0.956	0.955
85	0.951	0.960	0.968	0.962	0.961
90	0.958	0.964	0.972	0.967	0.966

Table 13: Actual delay value in intervals performance for best performing departure models

Interval width	Statistical	MVE	MDN 3	Dropout rate 0.5	Random Forest
2	0.108	0.075	0.119	0.110	0.108
5	0.262	0.177	0.286	0.268	0.261
10	0.464	0.372	0.489	0.477	0.477
15	0.607	0.521	0.651	0.627	0.631
20	0.710	0.665	0.749	0.730	0.734
25	0.781	0.748	0.814	0.798	0.802
30	0.832	0.818	0.857	0.846	0.848
35	0.868	0.858	0.888	0.877	0.880
40	0.894	0.889	0.910	0.901	0.902
45	0.914	0.910	0.927	0.919	0.921
50	0.929	0.924	0.940	0.934	0.934
55	0.942	0.936	0.950	0.944	0.944
60	0.951	0.944	0.958	0.952	0.952
65	0.958	0.951	0.964	0.960	0.959
70	0.964	0.955	0.970	0.966	0.966
75	0.970	0.961	0.974	0.971	0.970
80	0.974	0.964	0.978	0.975	0.974
85	0.978	0.968	0.982	0.978	0.978
90	0.982	0.971	0.985	0.982	0.981

Table 14: Probability mass in intervals around actual delay value for best performing arrival models

Interval width	Statistical	MVE	MDN 10	Dropout rate 0.5	Random Forest
2	0.038	0.038	0.049	0.048	0.059
5	0.096	0.095	0.123	0.120	0.145
10	0.189	0.188	0.241	0.235	0.283
15	0.278	0.278	0.350	0.342	0.405
20	0.361	0.363	0.447	0.439	0.513
25	0.438	0.441	0.531	0.523	0.599
30	0.507	0.512	0.602	0.595	0.670
35	0.569	0.577	0.661	0.655	0.726
40	0.623	0.634	0.710	0.706	0.770
45	0.670	0.684	0.751	0.748	0.804
50	0.710	0.727	0.784	0.783	0.831
55	0.745	0.765	0.812	0.812	0.854
60	0.775	0.797	0.836	0.836	0.871

Table 15: Probability mass in intervals around actual delay value for best performing departure models

Interval width	Statistical	MVE	MDN 3	Dropout rate 0.5	Random Forest
2	0.060	0.046	0.075	0.075	0.094
5	0.149	0.115	0.184	0.182	0.223
10	0.286	0.227	0.348	0.333	0.427
15	0.406	0.332	0.479	0.448	0.572
20	0.505	0.429	0.580	0.534	0.679
25	0.586	0.516	0.657	0.599	0.749
30	0.650	0.592	0.716	0.650	0.795
35	0.702	0.657	0.762	0.690	0.831
40	0.744	0.713	0.798	0.723	0.853
45	0.778	0.759	0.827	0.751	0.874
50	0.807	0.798	0.850	0.774	0.887
55	0.831	0.830	0.869	0.794	0.902
60	0.852	0.856	0.884	0.811	0.911

References

Hugo Alonso and António Loureiro. Predicting flight departure delay at porto airport: A preliminary study. In *IJCCI 2015 - Proceedings of the 7th International Joint Conference on Computational Intelligence*, volume 3, pages 93–98, 2015.

Loris Belcastro, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Using Scalable Data Mining for Predicting Flight Delays. *ACM Transactions on Intelligent Systems and Technology*, (January), 2016.

Christopher Bishop. *Pattern Recognition and Machine Learning*, volume 27. 2006.

Christopher M Bishop. Mixture Density Networks. *Neural Computing Research Group Report: NCRG/94/004 Aston University*, 1994.

Leo Breiman. Random forests. *Machine Learning 45*, pages 5–32, 2001.

Navoneel Chakrabarty. A data mining approach to flight arrival delay prediction for American airlines. *IEMECON 2019 - 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference*, pages 102–107, 2019.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE : Synthetic Minority Oversampling Technique. *Journal of Artificial Intelligence Research 16 (2002)*, pages 321–357, 2002.

Jun Chen and Meng Li. Chained predictions of flight delay using machine learning. *AIAA Scitech 2019 Forum*, 2019.

Sun Choi, Young Jin Kim, Simon Briceno, and Dimitri Mavris. Prediction of weather-induced airline delays based on machine learning algorithms. In *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, volume 2016-Decem, 2016.

Eurocontrol. Coda Digest 2019, All-Causes Delay and Cancellations to Air Transport in Europe. page 34, 2019. URL `https://www.eurocontrol.int/archive_download/all/node/11079`.

Eurocontrol. Performance Review Report (PRR 2019). 2020. URL `https://www.eurocontrol.int/publication/performance-review-report-prr-2019`.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *33rd International Conference on Machine Learning, ICML 2016*, pages 1651–1660, 2016.

Guan Gui, Fan Liu, Jinlong Sun, Jie Yang, Ziqi Zhou, and Dongxu Zhao. Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology*, 2020.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Infer- ence, and Prediction*. 2009.

Yuji Horiguchi, Yukino Baba, Hisashi Kashima, M. Suzuki, H. Kayahara, and J. Maeno. Predicting Fuel Consumption and Flight Delays for Low-Cost Airlines. *Innovative Applications of Artificial Intelligence (IAAI) Conference*, pages 4686–4693, 2017.

IowaStateUniversity. ASOS-AWOS-MEATR Data, 2020.

Abdul Karim, Avinash Mishra, M. A. Hakim Newton, and Abdul Sattar. Machine learning interpretability: A Science rather than a tool. *arXiv*, 2018.

Abbas Khosravi, Saeid Nahavandi, and Senior Member. Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances. (July 2014), 2011.

Miguel Lambelho, Mihaela Mitici, Simon Pickup, and Alan Marsden. Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions. *Journal of Air Transport Management*, 82, 1 2020.

Eric R Mueller, Gano B Chatterji, and Moffett Field. Analysis of Aircraft Arrival and Departure Delay Characteristics. *AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical*, (October):1–14, 2002.

David A. Nix and S. Weigend Andreas. Estimating the Mean and Variance of the Target Probability Distribution. *IEEE International Conference on Neural Networks (ICNN'94)*, 1994.

J. V. Pérez–Rodríguez, J. M. Pérez–Sánchez, and E. Gómez–Déniz. Modelling the asymmetric probabilistic delay of aircraft arrival. *Journal of Air Transport Management*, pages 90–98, 2017.

Juan Jose Rebollo and Hamsa Balakrishnan. Characterization and prediction of air traffic delays. *Transportation Research Part C*, pages 231–241, 2014.

Scikit-learn developers. Cross-validation: evaluating estimator performance, 2017. URL `https://scikit-learn.org/stable/modules/cross_validation.html`.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, pages 1929–1958, 2014.

Alice Sternberg, Jorge Soares, Diego Carvalho, and Eduardo Ogasawara. A Review on Flight Delay Prediction. 3 2017.

Yufeng Tu, Michael O Ball, Wolfgang S Jank, T U Yufeng, Michael O B All, and Wolfgang S J Ank. Estimating Flight Departure Delay Distributions — A Statistical Approach With Long-Term Trend and Short-Term Pattern E. 1459, 2012.

Oscar R.P. van Schaijk and Hendrikus G. Visser. Robust flight-to-gate assignment using flight presence probabilities. *Transportation Planning and Technology*, 2017.

Qian Wu. A Stochastic Characterization Based Data Mining Implementation for Airport Arrival and Departure Delay Data. (August 2012):1037–1040, 2014.

Bin Yu, Zhen Guo, and Sean Sobhan Asian. Flight delay prediction for commercial air transport: A deep learning approach. *Logistics, Part E*, (March), 2019.

II

Literature Study
previously graded under AE4020

1

Abstract

The application of machine learning in the field of flight delay predictions has seen significant growth in the last decade ([60]). The most recent advances often focus on binary predictions of flight delay by using flight, weather and flight delay related data. This is done on prediction horizons varying from a couple of hours before flight till 6 months in advance. Although these binary predictions provide better insight in flight delay, they do not provide information on what the actual delay value will be and how certain these predictions are (e.g. with an associated probability). However, probabilistic prediction methods are capable of providing distribution predictions for individual samples by obtaining n different estimations for one point or by estimating the parameters of a distribution. With these methods and data made available by Schiphol and KNMI, the following research question was defined:

Which machine learning model, trained on historical flight schedule and weather data, produced accurate flight delay distribution predictions for individual flights, on a prediction horizon of one day before flight?

2

Introduction

Flight delay is an ever-increasing problem due to the continuous growth of the number of flights each year. In 2019, the number of flights grew with a little more than 11000 in Europe compared to the year before, resulting in a total of 11.1 million flights. From these flights, 22.4% of flights experienced arrival delay of 15 or more minutes (Eurocontrol [24]). With this continuous increasing demand in air-traffic and the limited capacities of airports and the airspace, flight delay will only see an increase in its impact.

Research in the field of flight delay predictions has seen significant growth in terms of the application of machine learning (Sternberg et al. [60]). Machine learning is a subset of artificial intelligence in which computers use data to learn how to perform tasks rather than being programmed to do them. Methods which have been applied range from simple linear regression algorithms, to tree based methods such as random forest to more complex algorithm such as deep neural networks. As large datasets are made available by Schiphol and KNMI, machine learning seems an interesting approach to the problem of flight delay at Schiphol.

This paper will present the literature study as preparation for the research addressing the problem of flight delay predictions using machine learning. This literature study is focused on investigating what kind of delay predictions have been researched in literature and finding what opportunities and gaps exist in the research field. The literature gap is found by reviewing scientific papers, books and articles on state-of-the-art techniques in the field of machine learning. The aim of this literature study is to find the literature gap and provide a methodology in order to achieve the objective of the research to obtain probabilistic flight delay distributions for individual flights using machine learning.

The literature study will kick-off with a short chapter that is dedicated to describing the problem of flight delay (chapter 3) from different perspectives. Subsequently, chapter 4 will dive into the topic of flight delay predictions by first describing the fundamental concepts involved and secondly an elaborate literature overview is provided. A distinction is made in terms of what type of flight prediction (classification, regression or probabilistic) is obtained. Chapter 5 dives into the subject of machine learning methods for general probabilistic predictions. Chapter 6 treats how flight delay predictions can be integrated into the optimization of operations at an airport by giving an overview of recent studies in this field. In chapter 7 the literature gap, the research objective and the research question will be presented. Finally, chapter 8 will give the proposed methodology on how to achieve the stated research objective. The proposed methodology will serve as a guideline for the actual research that will be conducted over the coming months.

3

The problem of flight delay

Deviations in arrival time and departure time have been a problem for the aviation industry since its beginning. In Europe in 2019, 22.4% of flights experienced arrival delay of 15 or more minutes, Eurocontrol [24]. The number of flight grew with a little more than 11000 flights compared to 2018 to a total of 11.1 million in the ECAC area.

Amsterdam Airport Schiphol (AMS) uses the on-time performance of airlines as one of its key performance indicators. In 2019 the average outbound punctuality was 66.5%, which is the percentage of commercial flights that departed on time (Royal Schiphol Group [53]).

Estimated is that in 2040, Europe as a whole will have 16.2 million flights, Eurocontrol [21]. This is 53% more than 2017, which equals to an average annual growth per year of 1.9% till 2040. "Air traffic growth will be limited by the available capacity at the airports and when the capacity limits are reached, congestion at airports will increase quite rapidly which will lead to extra pressure on the network, and more delays."

As delays are a problem for both the passengers and the operators (airline and airport), early arrivals are also not beneficial. While this is good for the passenger experience, early arrival may affect operations, wherein 2019, 10.3% of flight arrived more than 15 minutes ahead of schedule (Eurocontrol [23]). Flight arriving early may experience that their gate is still in use by another flight and have to stand somewhere else. This creates extra work and complications for the allocation of the airplane performed by the gate planners.

Figure 3.1 by Deshpande and Arikan [17] summarizes in a clear way what the main segments of air travel time are and how the different flight segments, taxi-out, air time and taxi-in are related. Also is given how departure delay and arrival delay can be related to the scheduled (CRS) time and the actual departure/arrival time.

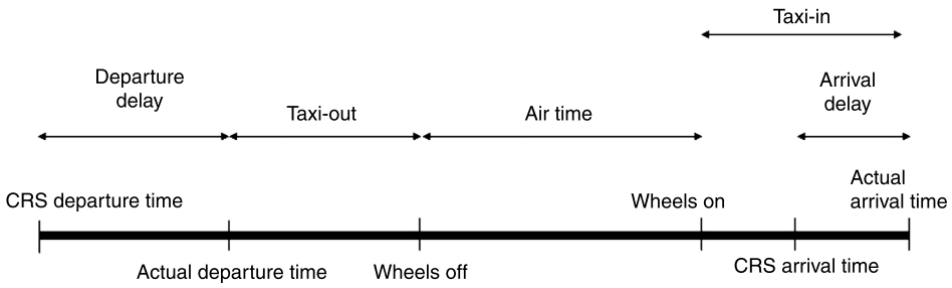


Figure 3.1: Main segments of air travel time. Adapted from Deshpande and Arikan [17]

This chapter will give a brief introduction to flight delay, where the problem is discussed from different perspectives, where in section 3.1 the cost of flight delay are discussed. Section 3.2 treats the causes of flight delay and section 3.3 discusses what can be done when flight delay is anticipated.

3.1. Cost of flight delay

Flight delays have many different financial impacts, which have been studied in different researches. In a report by Joint Economic Comittee [32] it was estimated that the yearly total cost of flight delays to the U.S. economy was as much as $41 billion. This estimate includes the operating cost to the airline of $ 19 billion and the estimated passenger delay cost of $ 12 billion.

Ball et al. [3] conducted a study on the total delay impact (TDI) in the United States. In their study they analyzed a variety of cost components caused by flight delay and indicated three major components that make up the total cost; cost to the airlines, cost to the passengers and costs from lost demand. Next to these 'direct' costs associated with flight delays, the authors pointed out a fourth, minor component, impact on GDP. Which described the increased cost experienced by other sectors due to inefficiency in the air transportation sector.

Krstić Simić and Babić [37] viewed the cost of delay from a sustainability point of view. In their research, it was found that delay may cause environmental damage. During peak hours at busy airports, arrival and departure queuing delays as well as increased taxi-in and taxi-out times induce additional fuel consumption, leading to extra gas emissions.

3.2. Causes of flight delay

The main causes of flights experiencing arrival delay of 15 or more minutes in Europe in 2019 were analyzed by Eurocontrol [23]. It was found that reactionary delay was the main delay contributor, which is the delay caused by late arrival of the aircraft or crew from a previous journey. Other causes for delay reported by airlines are from most contributing to least: Airline, ATFM En-route, ATFM Airport, Other Airport, Weather and Government.

Various researches have attributed flight delays to several causes such as airport congestion, airspace congestion, weather condition and more. Several studies (Rebollo and Balakrishnan [51], Wong and Tsai [64], Tu et al. [62]) agreed that a close relation exists between arrival delay and departure delay.
Pyrgiotis et al. [50] studies this effect of an aircraft arriving late, causing it to depart late on its next flight. A so-called 'delay propagation model' was developed and analyzed.

Other causes of flight delay were researched in different studies. Deshpande and Arikan [17] researched the impact of the scheduled block time allocated for a flight on flight delay. Wu [65] considered the role which airline ground operators play to flight delay in daily operations.

3.3. Integrating predictions into flight operations

Sternberg et al. [60] concluded in their review of flight delay prediction that "predictions are crucial during the decision-making process for every player in the air transportation system". A distinction can be made between predictions being used as insights for long-term strategical decision in the system or are used for improving the actual operations of a flight at that day.

Yu et al. [67] worked together with Beijing Capital International Airport (PEK) to develop a new practical data-driven model to accurately predict flight delays in real-time. A novel deep belief network was used which considers next to the already used macro factors (weather, seasonal effects, delay propagation and air traffic control) also micro-level factors (e.g. air route situation and crowdedness degree of the airport). The model's prediction should be made available two hours before flight departure to make it real-time, where the prediction error should be less than 30 minutes with a 98% confidence level.

Lambelho et al. [38] used flight delay predictions in the decision-making process at another time horizon, namely 6 months prior to execution. In their research, a generic assessment of strategic schedules based on flight delay and cancellation predictions was proposed. Strategic flight schedules are used by large European airports to mitigate air traffic demand-capacity imbalances. In these schedules, flights are assigned to arrival and departure slots several months prior to the execution of the actual flight.

4

Flight delay predictions

This chapter will dive into the research which has been done in literature in the area of flight delay predictions. First section 4.1 will give a brief overview and explanation of fundamental concepts which are used in flight delay prediction literature. Section 4.2 will present an elaborate overview and discussion of flight delay prediction literature, where studies are grouped and discussed based on the type of predictions: classification, regression or probabilistic.

4.1. Fundamental concepts in flight delay predictions

To better understand and compare the research findings presented in literature of flight delay predictions, some important concepts need to be explained which are used in the literature overview of section 4.2. These concepts are the following:

Definition of delay
Although the conventional definition of delay is that an event occurs after its scheduled times, in terms of flight delay different thresholds with respect to the scheduled time are used, which are defined as:

The (FAA [25]) defines flight delay in its Aviation System Performance Metrics (ASPM) as: "flights that arrive 15 minutes or more past the arrival time contained in the schedule or flight plan in effect at the time of departure to be delayed."

Eurocontrol [23], defines a flight as arrived punctual, when "flight arrives within 15 minutes or earlier than their scheduled arrival time (STA)". Meaning that flights that arrive 16 minutes or more after their scheduled time are defined as delayed.

In this Literature Study when the definition of flight delay is not specifically stated, it can be assumed that the definition of the FAA and EUROCONTROL is used for a flight being delayed when it arrives/departs 16min or more after the scheduled time of arrival/departure. When a different delay definition is used in a paper, this will be stated in ref table 1 and ref table 2 or in the discussion of the paper itself.

Prediction horizon
The prediction horizon is the quantity of time between the moment the prediction of a flight being delayed is made and the moment of that flight's execution. So, for example, a prediction horizon of one week means, that we predict whether a flight is delayed one week prior to the day of the flight's execution.

Used features
Features are the variables being used by the model in order to learn and make predictions. Features are most often obtained by processing the raw input data of the model. The data used in literature to define features in the context of flight delay predictions can be categorized in the following four main categories (see also table 4.4):

1. **Flight data**, e.g. airline, airport, airport size, aircraft, distance, seats, number of departures, etc.
2. **Time stamp data**, e.g. scheduled departure time, month, hour of day, day of the week, season, etc.
3. **Weather data**, e.g. wind direction, wind speed, visibility at the airport, temperature, precipitation, etc.
4. **Flight delay related data**. e.g. delay of arriving flight, arrival ATFM delay, crowdedness airspace, etc.

Note that features can be very dependent on the prediction horizon, in terms of whether they are available as well as the quality of the data. For example, the crowdedness of the airspace is only known on the day itself and has good quality less than an hour before the flight will arrive. The accuracy of the weather data becomes significantly better when the prediction horizon is small, e.g. a few hours before arrival.

Used Algorithm
Different algorithms can be used to make delay predictions. In the last 10-20 years, there has been an exponential increase in the number of algorithms which can be used to make predictions. Different families of algorithms are (see appendix 9 for an elaborate overview):

- Fitting flight delay data to a curve
- Classical Machine learning algorithms
- Tree Based methods (TB)
- Neural Networks (NN)

Type of prediction
Different types of prediction can be made in term of delay prediction using machine learning, where three main categories have been defined in this literature study (for more information, see appendix 9):

1. **Classification**, predicting whether the predicted target belongs to a certain category. The most simple form is binary classification, e.g. whether a flight will be delayed (1) or will be on time (0). Which is dependent on which definition is used for delay. Multi class classification deals with problems where more than 2 classes are defined. In the context of flight delay this can be for example 3 delay intervals in minutes: (-inf,-30], (-30,30), [30,inf).
2. **Regression**, predicting a target value instead of a class. The target value can be any real number, any positive number, or integer dependent on the problem. In the context of flight delay, the target value is the amount of delayed minutes or to be more precise the deviation in minutes (or seconds) from the scheduled time.
3. **Probabilistic**, predicting the target value or class with an associated probability distribution for that class or value. For classification problems, this method will result in attributing a probability to every class, accumulating to 1 when summed over all classes. For regression predictions a probability density function is obtained, when results in 1 when integrating over the range of predictions.

Performance of prediction
To be able to compare the quality of a prediction the performance of predictions need to be compared, where in order to obtain the performance of the prediction the prediction value or class is compared to the actual value or class. Dependent on the type of prediction different metrics are used (see Appendix 9 for more information):

- For Classification: **Accuracy**, **Recall**, **Precision**, **F1-score**, **ROC-curve**
- For Regression: **Mean squared error**, **Mean absolute error**
- For Probabilistic: **Confidence interval**

4.2. Literature overview

This section will present an overview of the studies which researched flight delay prediction and closely related delay prediction studies. These studies have been divided into four main groups based on the type of prediction for easier comparison. These four groups are: binary classification (4.2.1), multiclass classification (4.2.2), regression predictions (4.2.3) and probabilistic predictions (4.2.2). Each section consists out of an overview table of all the relevant reviewed papers with their key characteristics, where after every paper is briefly discussed and overall conclusion per type of prediction group are presented at the end of each subsection. Note that for the used features column when 'Flight data' is specified, this also includes 'Time stamp data', this is done to keep the tables organized as time stamp data is related to the flight data.

4.2.1. Binary classification

Table 4.1 presents an overview of all evaluated researches considering binary classification of flight delay, which are discussed in more detail individually in the paragraphs below.

Rebollo and Balakrishnan [51] proposed a Random Forest algorithm which both considers temporal and network delay states to predict departure delays 2 to 24 hours in the future. The authors evaluated their model on the 100 most delayed links in the evaluated system (National Airspace System of USA) for a prediction horizon of 2 hours at three different values of the classification delay threshold. For a 45 min threshold, the obtained accuracy is 0.788; for a 60 min threshold, accuracy is 0.809 and for the 90 min threshold, accuracy is 0.836. As the authors expected the accuracy improves as the classification threshold increases.
The authors also researched the performance of the classifier when the forecast horizon was increased (a classification threshold of 60min was used here). When evaluating the performance over the 100 most delayed links again, the authors obtained accuracies of 0.809 (2-h prediction horizon), 0.786 (4-h prediction horizon), 0.774 (6-h prediction horizon) and 0.728 (24-h prediction horizon).
The authors found that the False Negative Rate (FNR) dominates the False Positive Rate (FPR), meaning that the classifier is more likely to misclassify a high delay link than to predict high delay when in reality the delay on the OD pair is low. On the evaluated 100 links, the average FNR is 23.62% and the average FPR is 14.6%, where the FNR rate is higher than the FPR on all links. The authors attribute this behavior of the model to it focusing on the delay state of the different elements in the network and not capturing localized delays.

Choi et al. [13] primary goal was to predict airline delays caused by inclement weather conditions, where next to flight data, 12 different weather variables were used. To deal with the imbalanced data of delayed and non-delayed flight, the authors applied a combination of SMOTE (Chawla et al. [11]) and random undersampling. As can be seen from table 4.1 four different binary classifiers were evaluated, where it was found that all four performed better in terms of accuracy without the sampling techniques. The Random Forest classifier performs best both on the unsampled data (accuracy 0.834) and on the sampled data (accuracy 0.814). The authors argue that these results do not imply that applying a sampling technique is a bad choice. Classifiers are biased toward 'on-time' class when they are trained on imbalanced data, which makes it easier for a classifier to predict the 'on-time' class.
The authors also analyzed how the performance of the Random Forest classifier changed when weather forecast data (5days and 1 day in advance) were used instead of data from the actual weather. It was found that the predictive performance drastically lowered due to the uncertainty in the weather forecast. For a test-size of 56 flights, the accuracy for the 5 days forecast was 0.268, for the 1-day forecast 0.304 and for the actual weather data 0.804.

Belcastro et al. [4] used two different delay threshold to evaluate the performance of their trained Random Forest binary classifier. The model takes into consideration both flight information and weather conditions at origin airport and destination airport. It does not take into account en-route weather conditions as it is not trivial to derive the weather along a flight trajectory, as the altitude of the aircraft should also be taken into account, Takacs [61].
In the data preparation process, the authors filtered out canceled and diverted flight, as well as flights arriving late due to en-route weather, as these delays do not depend on weather information at origin/destination. To deal with the imbalance in the flight data, 78.3% on-time vs 19.9 % delayed (1.5% canceled, 0.2% diverted), a random under-sampling algorithm was used (Kotsiantis et al. [36]) to balance the classes. When evaluating the model with a delay threshold of 15 minutes the model obtained an accuracy of 0.742 and recall of 0.718, where for a threshold of 60 minutes the accuracy improves to 0.858 and recall to 0.869.

Table 4.1: Overview binary classification studies

Authors (Year)	Used algorithms	Performance prediction	Used features	Prediction horizon
Rebollo and Balakrishnan (2014)	Random Forest (TB)	Acc: 0.81	Flight data, Flight delay related data	2h - 24h
Choi et al. (2016)	Random Forest (TB) AdaBoost k-Nearest-Neighbors Decision trees (TB)	Acc: 0.834* ,Acc: 0.814 Acc: 0.832*, Acc: 0.781 Acc: 0.824*, Acc: 0.617 Acc: 0.828*, Acc: 0.770	Flight data Weather data (detailed)	few hours
Belcastro et al. (2016)	Random Forest (TB)	Acc: 0.742, Recall: 0.718	Flight data, Weather data	1 day
Ding (2017) [delayed above 30 min]	Multi Linear Regression Naive-Bayes C4.5 Decision Rree	Acc: 0.791, F1-score: 0.79 Acc: 0.702, F1-score: 0.75 Acc: 0.683, F1-score: 0.65	Flight data, Weather data, Flight delay related data	few hours
Horiguchi et al. (2017)	Random Forest (TB) XGBoost (TB) Neural Network (NN)	AUC: 0.604 (1 day) AUC: 0.634 (1 day) AUC: 0.647 (1 day)	Flight data (incl. pax. data)	1day, 1week, 5 months
Choi et al. (2017)	Random Forest (TB) AdaBoost k-Nearest-Neighbors Decision Tree (TB)	Acc: 0.828, AUC: 0.64 Acc: 0.831, AUC: 0.63 Acc: 0.806, AUC: 0.60 Acc: 0.825, AUC: 0.62	Flight data, Weather data	1 day
Perez et al. (2017)	Asymmetric Bayesion logit model	Acc: 0.9056 by alternating threshold probability	Flight data, Weather data	few hours
Chakrabarty (2019)	Gradient Boosting (TB) Gradien Boosting (SMOTE*)	Acc: 0.802, AUC: 0.71 Acc: 0.857, AUC: 0.9	Flight data	-
Gui et al. (2020)	RecurringNN (LSTM) Random Forest	Acc: 0.312(test), 0.99(train) Acc: 0.902	Flight data, Weather data, Flight delay, ADS-B data	1 hour
Lambelho and Mitici (2020)	LightGBM (TB), Multi Perceptron (NN), Random Forest (TB)	Acc: 0.794, AUC: 0.786 Acc: 0.772, AUC: 0.754 Acc: 0.771, AUC: 0.744	Flight data, Flight delay related data	6 months

Ding [18] proposed a method to model the arrival flights with a multiple linear algorithm to predict delay, where it compared this method against Naive-Bayes and a C4.5 decision tree approach. Although being regression methods, Ding [18] used them to classify the delay time predictions into the 'delay' or 'no delay' class, where a threshold of delay was used of 30 minutes. The multiple linear regression model outperformed the other two methods with an accuracy of 0.702 against respectively 0.702 and 0.683. Also in terms of precision, recall and F-score the multiple linear classifier performed better.

Horiguchi et al. [31] trained three different prediction models by using flight and passenger information to classify whether a flight will be delayed or not (delay if departure time 15 minutes behind scheduled, based on FAA criteria). Three different prediction horizons were considered, one day, one week and five months before departure date. The Deep Neural Network algorithm scored best for the one-day prediction horizon with an area under the ROC curve (AUC score) of 0.647 (with RF: 0.604, XGB: 0.634). For the prediction horizon of five months, the XGBoost method performed best with an AUC score of 0.542 (with DNN: 0.500, RF: 0.534).
In the encoding process of features, the authors used trigonometric functions on some features in order to take periodicity into account. For example, the 'departure day of year' values were transformed by the functions $sin(\frac{2\pi d}{365})$ and $cos(\frac{2\pi d}{365})$, where d is the departure day of the year value. This enables the model to understand that New Year's Day and New Year's Eve are sequential dates. This same approach was applied for the features 'scheduled departure time' and 'scheduled arrival time', but with the functions $sin(\frac{2\pi m}{1440})$ and $cos(\frac{2\pi m}{1440})$. With m being the scheduled time in minutes of the day, which is divided by 1440 as there are 24*60=1440 minutes in a day.

Choi et al. [14] proposed a cost-sensitive classifier to identify individual flight delays, where the misclassification costs of the on-time class and delayed class were analyzed. As classifiers are usually trained under the assumption that all types of misclassification costs are the same, which is in contrast with real-life problems, where the misclassification costs are asymmetric. This motivated the authors the adaptation of cost-sensitive learning as it allows for meaningful classification predictions on flight delays.
By using the 'costing' sampling method the performance of the model was evaluated by measuring the accuracy and the weighted error rate of the model for the various cost ratios between false positive errors and false negative errors. Four different types of models (see Table 4.1) were considered where the Adaboost method performed best for the 1:1 (normal) cost ratio with an accuracy of 0.828. For a cost ratio of 1:10, Random Forest outperformed the other models with an accuracy of 0.653. Interesting to note is that the accuracy of the Decision Tree and Adaboost method decreased significantly, from approximately 0.83 to 0.29, when changing the cost ratio from 1:1 to 1:10.

Pérez–Rodríguez et al. [47] presented an asymmetric logit probability model to predict the daily probabilities of aircraft arrival delay. Their models take into account statistical regularities by noting that more arrival is on time than delayed, reflecting an asymmetric pattern of behavior. The authors proposed an asymmetric Bayesian logit model, which was compared with a frequentist logit and symmetric Bayesian logit model. The proposed asymmetric Bayesian logit model obtained the best fit for the considered statistics. The obtained accuracy was 0.956, which is the same value as for the two other evaluated models, where the asymmetric Bayesian was able to identify a new delaying factor, namely distance between airports. Important to note for this study is that the threshold probability used to fit and predict a delay was the sampling frequency of delay, namely 0.237 (0.234 for the control sample), and not the conventionally seen 0.5 in binary classification.

Chakrabarty [10] proposed a Gradient Boosting classifier model, which is hyper-parameter tuned by applying a Grid Search method. It used the oversampling technique, Randomized SMOTE, Chawla et al. [11], to reduce the imbalanced data, which boosted the performance of the classifier. The accuracy increase from 0.802 to an accuracy score of 0.857 and the area under the ROC increased from 0.71 to 0.90 by using the SMOTE oversampling technique.

Gui et al. [27] explored flight delay predictions from a classification perspective for both binary and multiclass (3 and 4 classes), with classes divided by 1 hour. Next to flight schedule and weather data, they explored the use of ADS-B messages data. The ADS-B messages are used to compute the traffic flow in defined corridors by counting the number of flights per hour. For the predictions two algorithms were used; a Recurring Neural Network (RNN) with long short-term memory (LSTM) and a random forest based model.

For the RNN approach, three different network architectures based on the LSTM cells were implemented. First with a standard LSTM architecture, second a combination with a fully-connected layer and thirdly with an integrated additional dropout layer. The added dropout method with a memory depth of 3 performed best on the test set with an accuracy of 0.415. In contrast, the standard architecture with a memory depth of 7 performed best on the training set (accuracy of 0.99), but performed worst on the testing set (accuracy of 0.331). Indicating an overfitting problem, which may be due to the limited training data and the use of a random under-sampling strategy which decreases the available data size.
The random forest model obtained significantly higher prediction accuracies (0.902 vs 0.331 for the binary classification) and overcame the overfitting problem which was an issue for the RNN. The authors suggested that the generalization ability of the random forest based method is stronger than the LSTM based on the used dataset. However, they state that there are reasons to believe that the overfitting problem can be overcome by using more data for the LSTM based method.

Lambelho et al. [38] proposed a generic assessment of strategic flight schedules using predictions about arrival/departure flight delays and cancellations. Strategic flight schedules are schedules where flights are assigned to arrival/departure slots several months prior to execution, which is why the authors chose a prediction horizon of 6 months. They defined delay based on the definition of Eurocontrol [22], where flight are delayed if during execution, the flight arrives or departs 16min or more after the scheduled time.
To cross-check the classification results the authors used three different algorithms, each belonging to a different machine learning type, see table 4.1. The light GBM model performed best with an accuracy of 0.794 and AUC score of 0.786 for departure delay, and an accuracy of 0.791 and AUC score of 0.803 for arrival delay. The authors visualized the impact of features on the output by using so-called SHapley Additive exPlanation (SHAP) values, Lundberg and Lee [42]. They found that for departure delay the feature Arrival ATFM delay has the most impact, followed by the features Hour and Airline. For arrival delay classification the features Arrival ATFM Delay, Airline and Hour have the biggest impact. The feature Seats also has high importance for both arrival and departure delay classification.

From the discussed papers about binary classification of flight delay above, which are summarized in table 4.1 different conclusions can be drawn. First is is seen that compared to other type of prediction (see coming subsections), binary classification has been the main subject of interest for applying machine learning in the context of delay predictions. Random Forest has been the most frequently used method for binary classification, where the models of Pérez–Rodríguez et al. [47] and Gui et al. [27] performed the best in terms of accuracy, with a score just over 0.90. It is seen that in the last few years that the application of Neural Networks in delay prediction has seen significantly growing.

4.2.2. Multiclass classification

Some studies were interested in extending the number of classes to more than two, resulting in multi-class classification of flight delay. These studies are summarized in table 4.2.

Table 4.2: Overview multiclass classification studies

Authors (Year)	Used algorithms	Performance predictions	Used features	Prediction horizon
Alonso and Loureiro (2015)	Neural Network (NN) Tree based method (TB) [both methods: 5 classes]	r_{int}: 0.70 r_{int}: 0.66	Flight data, Weather data, Flight delay related	moment of arrival
Chen and Li (2019)	Random Forest (TB) [15 classes]	Acc: 0.867	Flight data, Weather data, Flight delay related	moment of arrival
Gui et al. (2020)	Random Forest [3 classes] Random Forest [4 classes]	Acc: 0.814 Acc: 0.700	Flight data, Weather data, Flight delay related, ADS-B messages	1 hour

Alonso and Loureiro [2] focused in their research on predicting departure delay by considering it as a multi-classification problem with the following 5 classes/intervals: (-∞,0), [0,15), (15,30], (30,60] and (60,∞). As these intervals can be viewed as naturally ordered classes, the authors treated the prediction problem as on ordinal classification task. Which motivated the usage of the so-called unimodal model, Pinto da Costa et al. [48], which takes into account the order relation between the classes. In general terms this means that the probability should decrease monotonically to the left and to the right of the interval or class where the maximum probability is attained, meaning that the distribution should be unimodal.

The authors applied the unimodal model to two machine learning algorithms, a neural network (Haykin [30]) and a tree based model (Hastie et al. [29]). In order to analyze and compare the results of the classifiers, a so-called r_{int} coefficient was used. This coefficient measures the association between the two ordinal variables, true class and predicted class. It is computed from the confusion matrix (Pinto da Costa et al. [48]), and takes values in [-1,1], where 1 is obtained when the two variables are identical and -1 when they are completely opposite. The authors motivated this choice of metric because misclassifications in this problem are not equally costly due to the ordinal nature of the problem, meaning that the misclassification error rate is not suitable. They argued that the mean squared error and the mean absolute deviation are better metrics, but decided that these two are also not appropriate as the performance assessment they provide is evidently influenced by the number of chosen classes.

The r_{int} values were calculated for both classifiers, where for the neural network r_{int} = 0.70 was obtained and for the tree based model r_{int} = 0.66. The authors concluded based on these values that a strong association between the true departure delay class and the predicted class was indicated, where the best performance was obtained by the neural network. The authors noted that the network obtained better results using only half of the predictor variables the tree method used.

Chen and Li [12] introduced an air traffic delay prediction model that combined multi-label random forest classification with a delay propagation model. An optimal feature selection process was introduced to improve the prediction performance and to make the model unbiased the synthetic minority over-sampling technique (SMOTE), Chawla et al. [11], was used to deal with the imbalanced data. The most important feature was shown to be the departure delay and late-arriving aircraft delay. The chained model was able to predict the flight delay along the same aircraft's itinerary given the initial departure delay.
For the multiclass classification delay groups were used, which were divided by 15 minutes from -2 to 12, where 15 minutes after scheduled time was used as a maximum not delayed value. The model with optimal features obtained accuracy for arrival delay of 0.867, where the model with all features scored an accuracy of 0.859. Due to the 'narrow' delay groups range of 15 minutes, a new metric called 'relaxed accuracy' was introduced, which allowed predictions to be one delay group index off. On this new metric of relaxed accuracy, the model scored 0.927 with optimal features and 0.875 with all features.

Compared to binary classification, significantly less studies focused on flight delay predictions in a multi-class setting. It is seen that the performance of the algorithms in terms of accuracy is dependent on the number of intervals (e.g. Chen and Li [12] versus Gui et al. [27]). In order to create a more 'fair' comparison metric for the difference in number of classes, Chen and Li [12] introduced the concept of 'relaxed accuracy'. This metrics focuses on the importance of flight being on time or delayed and not only on whether the correct class is predicted or not.

4.2.3. Regression predictions

Some studies investigated the use of regression models in the context of flight delay, where the actual value of delay in minutes was predicted, instead of a class. An overview of these studies is given in table 4.3, with their key characteristics.

Tu et al. [62] developed a non-parametric model based on smoothing-based splines, which estimates flight departure delay distributions which can be used to predict expected airspace congestion levels. The model consists of the three components, non-parametric methods for daily propagation and seasonal trends and in addition uses a mixture distribution to estimate the residual errors. The model was mode dynamically adaptive using a rolling horizon approach.

Table 4.3: Overview regression prediction studies

Authors (Year)	Used algorithms	Performance prediction	Used features	Prediction horizon
Tu et al. (2012)	Additative model with non parametric smooth splines	-	Seasonal trend Daily delay pattern Residual error	1 day
Rebollo and Balakrishnan (2014)	Random Forest (TB)	Median test error: 21 min	Flight data, Flight delay related data	2h - 24h
Lee and Malik (2016) [taxi-out time]	Linear Regression Support Vector Machines k-Nearest-Neighbors Random Forest Neural Network	MAE: 3.79, RMSE: 4.83 MAE: 5.40, RMSE: 6.83 MAE: 4.50, RMSE: 5.77 MAE: 3.75, RMSE: 4.82 MAE: 5.47, RMSE: 6.61	Flight data, Weather data, Traffic flow data	few hours
Yu et al. (2019)	DBN-SVR (NN) k-Nearest-Neighbors SVM Linear Regression	MAE: 8.41, RMSE: 12.65 MAE: 11.96, RMSE: 16.01 MAE: 12.04, RMSE: 16.15 MAE: 15.56, RMSE: 20.20	Flight data, Flight delay related data	2 hours

Rebollo and Balakrishnan [51] trained next to a Random Forest classifier, also a Random Forest regression model, which was trained to predict the delay time value of a flight. The performance of their regression model was evaluated using the same data set for the 100 most delayed OD pairs, where an average median test error of 20.9 min was found. The authors found that the standard deviation of these error values is low, with the 90-th percentile of the error distribution being 1.17 minutes. Also for the regression model the prediction horizon was increased, resulting in an average median test error of 23 min (4-h), 24.3 min (6-h) and 27.4 (24-h). Meaning that the average median test error increases with only 6.5 minutes as the prediction horizon increases from 2h to 24h.
The authors investigated the relationship between the classification error and regression test error. They found that although there is a strong positive correlation (0.78) between the two, some specific links perform significantly different in the classification and regression problem. The authors concluded that in general, the classification and regression problems are different in the sense that the former one requires information that helps to differentiate between high and low delay values, whereas the latter tries to predict the actual value of the delay.

Lee and Malik [39] proposed machine learning algorithms for predicting taxi-out times of departures at Charlotte airport. The algorithms were trained on a combination of flight, weather and traffic flow data. From the 5 tested machine learning algorithms, linear regression and random forest performed the best on the test set with a mean absolute error of respectively 3.79 and 3.75 minutes. Where the average taxi-out time of the test set entails 17.78 minutes with a standard deviation of 6.59 minutes.

Yu et al. [67] followed a multifactor approach for their flight delay prediction model with a novel deep belief network method to mine the inner patterns of flight delays. A support vector regression was embedded in the development of the model to perform a supervised fine-tuning. One of their goals was to detect both macro and micro level key influential factors, where micro influential factors such as air route situation and crowdedness degree of the airport were used, which directly affect flight delays at the operations level. The prediction performance of the model satisfied the key requirement set as 99.3% of the predicted values were within 25min deviation from the observed value.

For the discussed regression studies the focus has been on predicting a value for the flight delay instead of a class. A large variety of algorithms have been evaluated for predicting taxi-out delays as well as flight delays. In the study of Lee and Malik [39], the Random Forest model performed best in terms of RMSE, which was

also the best performing algorithm for binary classification. For the study of Yu et al. [67], the Neural Network scored best in terms of RMSE performance (no Random Forest method was tested). Their network was able to accurately provide real-time predictions that can be implemented for daily operation at airports.

4.2.4. Probabilistic predictions

A small number of studies investigated the use of models which give probabilistic predictions of flight delays. For example Mueller et al. [44] modeled delay by assuming that it is a random variable and that if follows a statistical distribution. In their research distributions were created which show the probability of a certain delay time for a given flight. The delay-time probability density functions were modeled using Normal and Poisson distributions with the mean and standard deviations derived from the raw data. A least squares method was used to minimize the fit error between the raw data and the model. The authors found out that departure delay can best be modelled using a Poisson distribution, while the en route and arrival delays fit a Normal distribution better.

Pérez–Rodríguez et al. [47] used probabilities for their binary classification predictions. Probabilities were used in the sense that they were given whether a flight was delayed or not based on a predefined definition of delay (often 15 minutes). Meaning probabilities were used in order to indicate uncertainty among the defined classes for the predictions.

No research has been found in literature that predicted flight delays in a probabilistic approach, where for a single flight, a probability distribution was given with respect to the scheduled time of the flight.

4.2.5. Features in delay predictions

For each research from table 4.1 till table 4.3 it has been given what kind of data was used for the features; flight data (including time stamp data), weather data and flight delay related data. Table 4.4 gives an overview of all the features used per study for creating flight delay predictions. Some studies provided information about the importance of the individual features in a ranked manner. If this is the case, the top 3 or top 5 is given in parentheses behind the features (dependent on what is given).

From table 4.4 it is seen that some feature or used in almost every research, e.g. airport and scheduled departure time. While some features are rarely used, e.g. crowdedness airport passengers and day of the year. It is seen that for all researches where feature importance is given, that when flight delay related data features are used, they are ranked as most important for the algorithm's predictions. Interesting to note is that weather data features are not present in the lists of most important features for the considered researches.

Table 4.4: Overview used features per research

Data type	Feature	Tu (2012)	Rebollo ('14)	Alonso ('15)	Choi (2016)	Belcastro(16)	Lee (2016)	Ding (2017)	Horigu ('17)	Choi (2017)	Perez (2017)	Chakra ('19)	Chen (2019)	Ye (2019)	Gui (2020)	Lambelho (2020)
Flight data	airline			x				x			x	x		x		x (3)
	airline size										x (2)			x		
	aircraft type			x			x	x						x		x
	aircraft weight class						x									
	airframe / flight ID		x						x			x			x	
	airport		x	x		x		x	x	x	x	x	x	x	x	x
	airport size										x					
	distance (km)					x	x	x			x					x
	scheduled air time					x		x	x				x			
	country			x								x				x
	domestic or international								x							
	origin or pass-by flight													x		
	seats (aircraft capacity)													x		x (4)
	terminal						x							x		x
	gate						x							x		
	parking stand			x			x							x		
	ground operation time			x			x							x		
	take-off runway		x	x			x									
	number of arrivals						x						x	x (4)		
	number of departures						x						x	x (4)		
	market concentration													x		
	interval of previous flight													x		
	crowdedness airport pax													x (5)		
	reservation information								x							
	passenger information								x					x		
Time stamp data	scheduled departure time		x	x	x	x	x	x	x	x		x	x	x	x	x
	scheduled arrival time		x		x	x		x	x	x		x	x (3)	x	x	x
	month			x	x				x	x		x			x	x
	year								x			x				x (2)
	hour of day			x												x
	day of the week			x	x				x	x	x (3)	x	x		x	x
	day of month			x	x				x	x		x	x		x	x
	day of year								x							x (5)
	quarter of year/ season				x					x		x			x	
Weather data	temperature					x	x	x					x			
	humidity					x		x					x			
	barometric pressure					x		x					x			
	wind direction angle		x		x	x	x	x		x			x		x	
	wind speed rate		x		x	x	x	x		x			x		x	
	visibility		x	x	x	x	x			x			x			
	precipitation		x		x		x			x						
	snow depth / accumulation				x					x						
	weather intensity code				x			x		x						
	weather descriptor code		x		x					x			x		x	
	precipitation / obscuration / other weather code				x	x		x		x						
	seasonal trend	x														
Flight delay related data	arrival delay (minutes)			x				x						x (2)		
	arrival delay indicator				x							x	x (2)			
	arrival ATFM delay															x (1)
	departure delay (minutes)										x (1)					
	departue delay indicator												x (1)			
	daily delay pattern	x														
	air traffic control													x (1)		
	air route situation (ADS-B)													x (3)	x	

5

Machine Learning for general probabilistic predictions

This section will treat the different types of machine learning which can be used to obtain probabilistic estimations, where two main distinctions can be made in how these probabilistic estimations are obtained, via training multiple models or via training a single model. Before discussing the models, however, the use of prediction intervals will be discussed as they can a measure for uncertainty of the prediction.

5.1. Prediction Intervals

5.1.1. Terminology

Prediction Intervals (PIs) will be used as a measure for uncertainty of a prediction. They are different than a confidence interval, as a confidence interval quantifies the uncertainty of an estimated model variable, such as the mean and standard deviation (Shrestha and Solomatine [58]). A prediction interval quantifies the uncertainty of a prediction, often consisting out of an upper and lower bound of which a future unknown value (the prediction) is expected to lie with a prescribed probability. Figure 5.1 visualizes the terminology involved with a prediction interval.

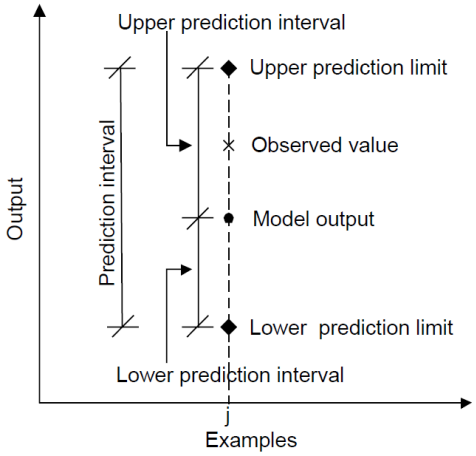


Figure 5.1: Prediction interval terminology. Taken from Shrestha and Solomatine [58]

5.1.2. Constructing Prediction Intervals

In order to construct these prediction intervals, two methods can be considered. The first method assumes an underlying distribution for the residuals, often the normal distribution. For the second method the prediction interval are obtained empirically and no underlying distribution needs to be assumed.

For the first method, it is assumed that the prediction error (residuals), e_i, follow a certain distribution. Where the most common assumed distribution is the Gaussian [6] [58], which will be used from this point onward. It is furthermore assumed that e_i is independently and identically (iid) distributed with variance σ^2 and that the distribution has the form $N(0,\sigma^2)$. Based on these assumptions the upper and lower bound of a prediction interval can be defined as:

$$PI^U = \hat{y}_i + z_{\alpha/2} \tag{5.1}$$

$$PI^L = \hat{y}_i - z_{\alpha/2} \tag{5.2}$$

where PI^U and PI^L represent the prediction interval's upper and lower bounds respectively and $z_{\alpha/2}$ is the value of the standard normal variate with a cumulative probability level of $\alpha/2$. For example for a 95% prediction interval for an assumed Gaussian distribution $z_{\alpha/2}$ has a value of 1.96. As the upper and lower bound in equation 5.1 and 5.2 are symmetric around $\hat{y}_i$, it is assumed that the prediction is unbiased.

The problem here is that σ^2 is often not known in practice and needs to be estimated from the n data samples. An unbiased estimate of σ^2 with n-p degrees of freedom, denoted by s^2, is given by formula 5.3 [16] [58], where p is the number of parameters in the model and SSE is the sum of squared errors.

$$s^2 = \frac{SSE}{n-1} = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.3}$$

For the construction of prediction intervals by the second method, no assumption of the underlying distribution of the residuals is needed. The prediction intervals are created empirically based on the number of predictions and their related quantiles by constructing an upper and lower bound. The advantage of this second method is that no underlying distribution has to be assumed. The downside however is that no 'confidence' level can be constructed based on an assumed distribution. Only an empirical 'confidence' level or probability can be calculated by taking the fraction of predictions that lie within the interval divided by the total number of predictions.

The second method works by ordering all predicted values from low to high and dependent on which prediction interval is chosen, prediction points are selected as upper and lower bound. For example, if 100 predictions are made, they are ordered and the 5th value and 95th value are selected to make the lower and upper bound of the 90% prediction interval.

For both methods it is important to carefully check if the algorithms can make accurate predictions, that is that the predictions lie somewhat in the neighbourhood of the actual value. This is of importance as it is possible that an algorithm cannot capture the complex behaviour of delayed flights or the data does not have enough predictive power. This can results in inaccurate predictions from which prediction interval can be obtained, with an associated 'confidence' level, for which the actual value does not even lie in or close to the obtained interval.

Therefore it will be checked once prediction intervals are obtained with an associated 'confidence' level, if the actual values will lie in these intervals with a certain frequency. For example, if for a certain algorithm 95% prediction intervals are obtained and only in 50% of the predictions the actual value lies within the prediction interval. This means that the algorithm does not perform satisfactory and changes need to be made by for example improving the algorithm (hyper parameter tuning or new features) or even changing to a different type of algorithm.

5.1.3. Evaluating Prediction Intervals

To assess the performance of a prediction interval two main perspectives will be considered, first from a coverage probability perspective and secondly from a width perspective as discussed in Khosravi et al. [35]. The most importance will be given to the PI coverage probability metric (PICP), which is measured by counting the number of target values covered by the constructed PI's overall the test samples (n_{test}), see equations 5.4 and 5.5.

$$PICP = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} c_i \tag{5.4}$$

$$c_i = \begin{cases} 1, & y_i \in [PI_i^L, PI_i^U] \\ 0, & y_i \notin [PI_i^L, PI_i^U] \end{cases} \tag{5.5}$$

The width performance of the interval is measured by the Mean PI width (MPIW), see equation 5.6.

$$MPIW = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (PI_i^U, PI_i^L]) \tag{5.6}$$

As both PICP and MPIW evaluate the quality of PIs from a different perspective, a combined metric is required to compare models based on a single metric. This metric will give a higher priority to PICP, as this is the most important feature of PIs. The combined metric is called the coverage-width based criterion (CWC) [35], which in given in equation 5.7 and evaluates PIs from both the coverage probability and width perspective. The two hyper parameters controlling the equation are η and γ, which can be determined by the level of confidence associated with the PIs. η corresponds to the nominal confidence level associated with PIs and can be set to 1 - α. γ is dependent on the PICP and η as can be seen in equation 5.8.

$$CWC = MPIW\left(1 + \gamma(PICP)e^{-\eta(PICP-\mu)}\right) \tag{5.7}$$

$$\gamma = \begin{cases} 1, & PICP < \mu \\ 0, & PICP \geq \mu \end{cases} \tag{5.8}$$

The authors, Khosravi et al. [35], based on the design of the CWC metric on two principles. Firstly, if PICP is less than the nominal confidence level (1-α), CWC should be large regardless of the width of the PI. Secondly, if PICP is greater than or equal to its corresponding confidence level, then MPIW should be the influential factor. The exponential term in the CWC formule penalizes the violation of the coverage probabilities. This a smooth penalty rather than a hard one, because it appropriately penalizes the degree of violation, rather than just an abrupt binary penalty [35]. The role of η is to magnify any small difference between PICP and μ. Usually, it should be selected to have a large value (10,100,300), Khosravi et al. [34].

5.2. Via training multiple models

Instead of training just a single model, multiple of the same model (identical algorithms and settings) can be trained to obtain multiple predictions for the same input, also called an ensemble of models. Any kind of model can be used to create an ensemble of these models. Figure 5.2 displays an ensemble of neural networks [35], which all predict a single value ($\hat{y}^i$) based on the same input data. From all these predictions an average prediction ($\hat{y}$) as well as an indication of the uncertainty for that prediction, often the variance ($\sigma^2_{\hat{y}}$), is given. But in order to obtain (slightly) different predictions, slightly different models are needed. Meaning that the training sets need to be slightly different for each model, otherwise the same model would be obtained. There are two main ways in how different datasets can be obtained.

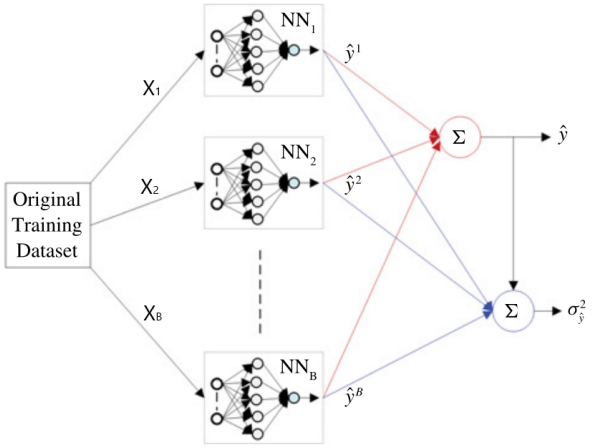


Figure 5.2: Ensemble of Neural Networks. Adapted from from Khosravi et al. [35]

The first method divides the original dataset into n pieces on which the n models can be trained. This method uses the same approach as cross-validation (explained in appendix 9), but instead of computing the same metric (e.g. accuracy) multiple times, the same prediction is made multiple times. The drawback of this method however is that in order to obtain sufficient predictions, for example, n prediction, the original dataset needs to be separated into n parts. By doing so the dataset can become too small, resulting in that the model will not be able to learn the generalized patterns of the data, meaning it will perform poorly on data it has never seen before. This concept is called overfitting (see also appendix 9) and happens when the model adjust to specifically to the training, finding patterns which do not exists, resulting in poor prediction on new data.

In order to keep the dataset sufficiently large enough, a second method can be used, namely bootstrapping (Efron [20], Bishop [6]) can be used, in which multiple data sets are created as follows. Based on an original dataset consisting out of N data point, $X = \{x_1, .., x_N\}$, a new data set X_B is created. This is done by drawing N data points at random from X with replacement. This means that some points in X may be replicated and multiple times present in X_B, whereas other points in X may be absent in X_B. This process can be repeated M times, such that M datasets of size N are created based on the original dataset. Note that X_B has the same number of data points as the original dataset X, this, however, is not necessary and any arbitrary size of the number of data points can be chosen for X_B.

With an ensemble of M models being trained on M different bootstrap datasets, M different predictions can be made based on the same input. By looking at the variability of the predictions between the trained models the statistical accuracy of a prediction can be evaluated by for example defining a prediction interval as was discussed in section 5.1.3.

5.2.1. Quantile regression

In normal regression problems, the algorithm is trained by minimizing a loss function (see appendix 9). The most familiar loss function is the squared error loss function, which is minimized in the least square regression. Different loss functions can be applied for the training of algorithms, so also the minimization of the sum of absolute residuals. In fact here the algorithm is trained for defined the median as the solution Das et al. [15]. The symmetry used for the piecewise linear absolute value function implies, that positive and negative residuals are treated the same way (if their absolute value is the same). As the median is the 0.5 quantile, by alternating the absolute function, the algorithm can be trained to predict other quantiles. In order for this, the sum of asymmetrical weighted absolute residuals should be minimized, meaning that different weights are assigned to positive and negative residuals in order to predict quantile values. This can be described mathematically, as is done in equation 5.9. Where the function ρ_τ is the tilted absolute value function, that yield the τth sample quantile as its solution ([15]). Figure 5.4 gives the solution of this method applied to a neural network solution.

$$min \sum \rho_\tau (y_i - \hat{y}_i) \tag{5.9}$$

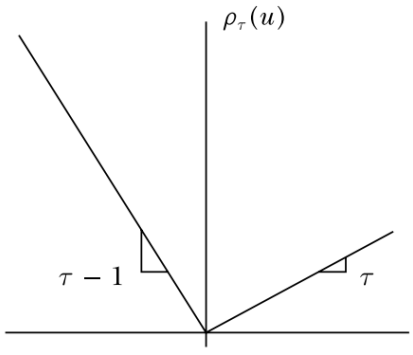


Figure 5.3: Quantile regression ρ function. Taken from Das et al. [15]

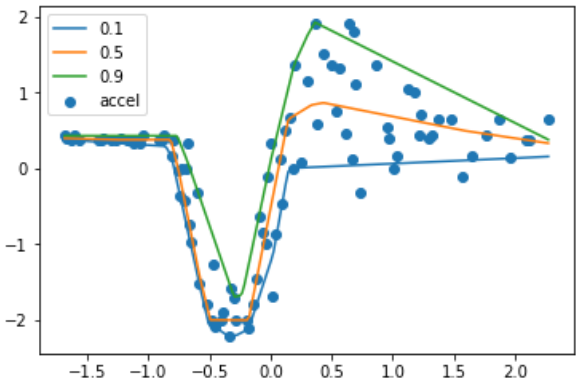
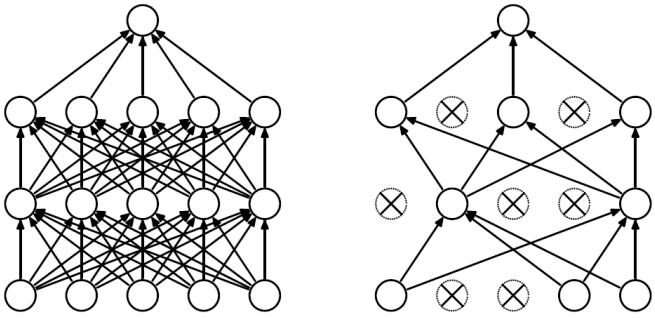


Figure 5.4: Quantile regression example of neural network. Taken from Abeywardana [1]

5.3. Via training a single model

5.3.1. Drop-out method for Neural Networks

The drop-out method is conventionally used in the training stage of a neural network, where its name refers to the process of randomly dropping out of units (neurons) in the network. Drop-out prevents units from co-adapting too much, which significantly reduces overfitting and gives major improvements over other regularization methods, Srivastava et al. [59]. By dropping a unit out, temporarily removing it from the network, along with all its incoming and outgoing connections is meant. The choice of which unit gets dropped is random, where the dropout rate determines the ratio of drop-out neurons in the hidden and/or visible layer of the network. Figure 5.5 visualizes how the dropout method works, wherein figure 5.5a, a standard neural network is given with 2 hidden layers. Figure 5.5b gives the same network, but now with the dropout method applied to result in a 'thinned network'. Crossed units have been dropped out.



(a) Standard Neural Net (b) After applying dropout.

Figure 5.5: Visualisation of the dropout method. Taken from Srivastava et al. [59]

Although the dropout method was initially designed to be used in the training process of a neural network, it can also be used to represent model uncertainty, as has been proposed by Gal and Ghahramani [26]. They developed a theoretical framework casting the dropout training method in a deep neural network as approximate Bayesian inference in deep Gaussian processes. As the author states this gives a tool to model uncertainty of neural network model. To estimate the predictive mean and predictive uncertainty, a Monte Carlo method approach is used, where N stochastic forward passes through the model results are collected. These N predictions can be used to obtain an empirical mean and variance (uncertainty) of a single prediction. Methods on how to obtain these were described in section 5.2

When making predictions of the uncertainty via the dropout method it is important to define a good dropout rate for the model. As Blake [8] explains, if the dropout rate is too large then the predictions generated will be very diverse, meaning the confidence interval will become too large. Conversely, if the dropout rate is too small then the predictions generated will be too similar, resulting in a too narrow confidence interval.In order to determine the optimal dropout rate, it is suggested to look at the percentage of the actual values which fall within each calculated confidence interval. For an optimal dropout rate, it is expected that 10% of the actual values fall within the 10% confidence interval, 20% within the 20% confidence interval and so on.

5.3.2. Mean-Variance Estimation (MVE) method using Neural Networks

All methods described until now created prediction intervals via predicting multiple actual values. Instead of training a neural network to predict an actual value, it can be trained to predict the parameters of a probability distribution. In this sense, a direct measure of the uncertainty of a prediction is given. Using the predicted distribution parameters (mean and variance), prediction intervals can be obtained by using the formula's for upper and lower bound (equations 5.1 and 5.2).

Nix and Andreas [45] introduced a method to estimate the mean and variance of the probability distribution as the target function depends on the input, given an assumed target error-distribution model. The fundamental assumption made is that the target variance is dependent on the set of inputs. The architecture of their neural network is depicted in figure 5.6, where the output layer consists out of two nodes predicting $\hat{y}$ and $\hat{\sigma}^2$. Note that both outputs share the same input units, but don't share any of the hidden layer connec-

tions. This approach of splitting the hidden layers per output is not necessary as $\hat{y}$ and $\hat{\sigma}^2$ could be connected to a common large set of hidden layers, but in the experience of Nix and Andreas [45] this split-hidden-unit architecture works better.

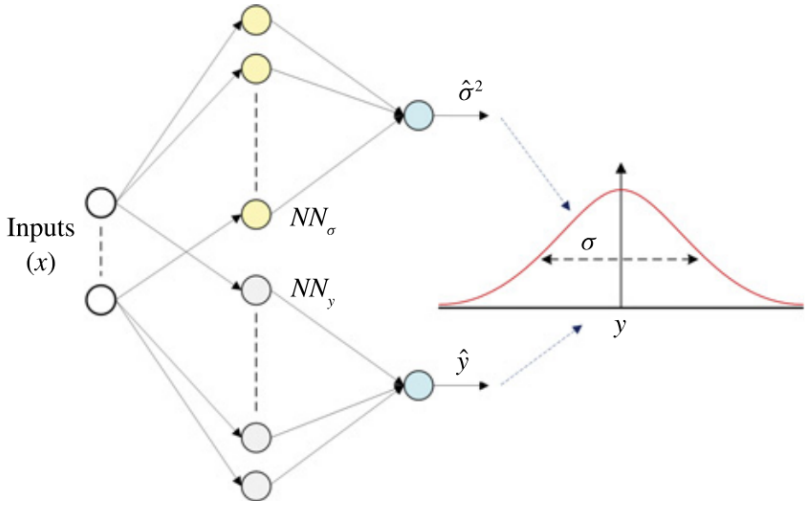


Figure 5.6: Architecture of the MVE method. Taken from Khosravi et al. [35]

In order to train the neural network, it has to be decided what general shape the predicted distribution will have. This probability distribution can be expressed as a function $f(y,\theta)$, where y represents all values the target variable can take, while Y represents the actual value and θ represents the set of parameter(s) describing the shape of the distribution. Depending on which distribution is chosen, the number of parameters is set, dictating the number of outputs. The range these outputs can take needs to defined as well. This is done by choosing the correct activation function in the last layer of the neural network (see appendix 9 for more information). Taking the normal distribution as an example, the mean ($\hat{y}$) can be any real number, which means that no range limitations are needed. The variance ($\hat{\sigma}^2$) of a normal distribution is however strictly positive, to ensure this a softplus or ReLu activation function should be used.

As the MVE method predicts a set of parameters θ of a distribution and not a value itself, a different loss function is needed to reflect this fundamental difference. Normal regression algorithms use error-based minimization which takes as input the predicted value and the actual value, for example minimizing the mean squared error. To define a suitable loss function, let's reflect on what is actually happening. When training the model, Y the actual value of the target variable is known and the goal is to find the values of θ that maximizes the probability of y=Y. In other words, the parameters of θ are varied for a known value of Y in order to find the parameters of a distribution from which Y comes most likely, which is called maximum likelihood $L(\theta|Y)$. The model searches for the curve that maximizes the probability of our data set given a set of curve parameters which have to predict.

In practice, the natural logarithm is taken from the likelihood function in order to make it better solvable for computers. By convention, loss functions are minimized and a maximization function is considered till this point, its negative is therefore taken, resulting in the final cost function given in formula 1.3.

$$C_{MVE} = \frac{1}{2}\sum_i \ln\sigma_i^2 + \frac{(y_i - \hat{y}_i)^2}{\sigma_i^2} \tag{5.10}$$

To summarize, the MVE method predicts parameters of an assumed distribution, by minimizing its negative log-likelihood function. The main advantage of this method is its simplicity as no complex derivatives and the inversion of the Hessian matrix (Khosravi et al. [35]), making is computationally cheap. Next, the methods can predict variance based on the input, where other methods assume a constant variance. The downside of the method is that a probability distribution needs to be assumed, which often is unknown in real-life problems.

Mixture Density Networks

As was discussed before the drawback of the MVE method is that a probability distribution needs to be assumed, which is often unknown in real life. Bishop [7] extended the idea of predicting the parameters of one assumed distribution, to predicting the parameters and weights of multiple distributions which make up one distribution (see schematic in figure 5.7). The method was called Mixture Density Networks (MDN) as its idea was quite straightforward: combine a deep neural network and a mixture of distributions. The DNN predicts the parameters for multiple distributions, which are then mixed by some predicted weights.

Theoretically, a mixture of Gaussian functions is capable of modeling arbitrary probability densities (Bishop [7]), if it is adequately parameterized with enough components. The probability density function of the target data is then represented as a linear combination of kernel functions in the form of equation 1.4.

$$p(y|\mathbf{x}) = \sum_{j=1}^{m} \alpha_j(\mathbf{x})\phi_j(y|\mathbf{x}) \tag{5.11}$$

where m is the number of components in the mixture. The parameters $\alpha_j(\mathbf{x})$ are called the mixing coefficients, which are functions of the input $\mathbf{x}$ and which must satisfy that there sum equals 1. The functions $\phi_j(y|\mathbf{x})$ are the conditional density functions of the target y for the j^{th} kernel for which various choices are possible. The loss function is defined in the same way as for the MVE method, but now the m components of the mixture need to be taken into account.

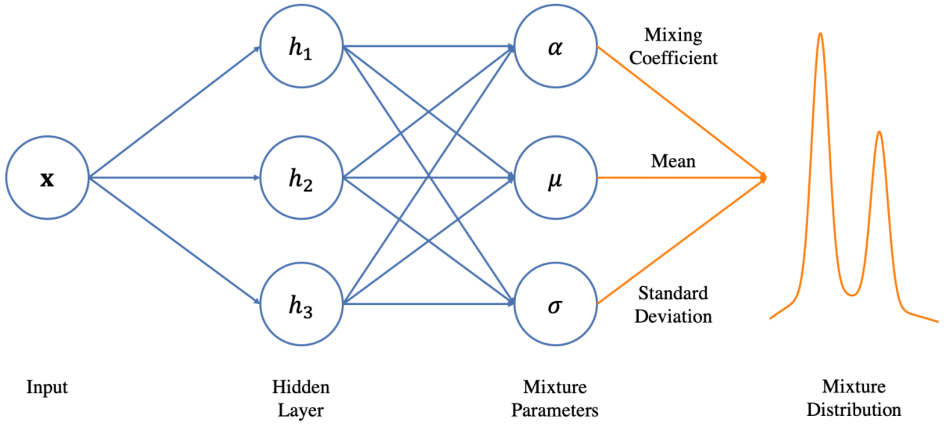


Figure 5.7: Representation of mixture density network. Taken from Borchers [9]

6

Integrating of flight delay predictions into operations optimization

This chapter treats how flight delay prediction can be integrated into operation optimization problems. First, an overview will be given of the different integration made in literature in 6.1, where after they will be discussed in more depth per research in section 6.2.

6.1. Overview literature

Table 6.1 gives an overview of the different integrations of flight delay predictions into operation optimization researched in literature. In the table type of optimization problem states what kind of problem is solved, which is most often the flight to gate assignment. Methodology for robustness states briefly how in a particular research flight delays are used and how they are integrated into the optimization process.

Table 6.1: Overview operation optimization with integration of flight delays

Authors	Year	Type of optimization problem	Methodology for robustness
Yan and Tang	2007	flight to gate assignment	stochasticiy due to the creation of n flight scenarios
Lee, Lee and Tan	2007	flight schedule	simulate airline operations under operational irregularities using SIMAIR 2.0
Seker and Noyan	2012	flight to gate assignment	uncertainty of flight arrival and departure times is incorperated by idle time as random variable
Khanmodhammadi, Chou, and Lewis	2014	aircraft landing scheduling	fuzzy decision making procedure to schedule with adaptive network to predict flight delays
Dorndorf, Jaehn and Pesch	2017	flight to gate assignment	clique partitioning problem with minimization of the expected number of gate conflicts
van Schaijk and Visser	2017	flight to gate assignment	stochastic constraints to limit gate conflict probability based on flight presence probabilities
Pternea and Haghani	2019	flight to gate assignment	flight arrival and departure delays are introduced using statistical distributions

6.2. Robuster schedules at airport

Yan and Tang [66] developed a heuristic approach to make airport gate assignments sensitive to stochastic flight delays. The framework included three components, a stochastic flight delay gate assignment model, a real-time assignment rule and two penalty adjustment methods. The stochasticity of flight delay is created by generating n flight delay scenarios, which are used to solve the gate assignment model. For the generation of n flight delay scenarios, different types of delay considered, being low, medium and high delay scenarios.

Lee et al. [40] researched how to improve the robustness of traditional flight schedule as they do not take into consideration disruptions that may arise during the actual operations. A multi-objective genetic algorithm was used where a simulation model, SIMAIR 2.0, was used to simulate airline operations under operational irregularities. By re-timing the departure times of the flight schedules via this method better on-time performance schedules were obtained.

Seker and Noyan [57] incorporated random disruptions in constructing the flight to gate assignment as they are crucial for the effectiveness of the schedule. The authors considered the uncertainty of flight arrival and departure times in the problem. A stochastic programming model was created which incorporates a robustness measure based on the number of conflicting flights, idle and buffer times. The stochasticity is incorporated by defining the idle times as a random variable making the constraints of stochastic nature. Due to this approach, the problem became computationally expensive, meaning that it became hard to solve for a standard solver as CPLEX. In order to obtain an assignment of reasonable quality within a realistic solving time, a tabu search algorithm was implemented.

Khanmohammadi et al. [33] identified that traditional aircraft landing scheduling problems are not able to capture the dynamic nature of the problem by handling the uncertainty involved. The authors introduced a systems approach to the problem based on a fuzzy decision-making procedures. Arrival flights of JFK were simulated by using randomly defined weight of criteria
Needs to be extended and improved this text part

Dorndorf et al. [19] included stochastic starting and completion times of flight activities to the flight to the gate assignment problem. The authors modeled the problem as a clique partitioning problem, which introduces a measure of robustness. As objective function, the minimization of the expected number of gate conflicts was defined. When testing the model with statistical data on arrival time distributions the author noted that “little is known about dependencies of these arrival time distribution, which could result in more accurate solutions, making them an interesting topic for future research.”

van Schaijk and Visser [63] presented a method to improve the robustness of the solution to the Flight-to-Gate Assignment Problem (FGAP) with the aim to reduce the need for gate re-planning due to unpredicted flight schedule disturbances in the daily operation at an airport. This is done by replacing the deterministic gate constraint with a stochastic gate constraint. Via this way the inherent stochastic flight delays are incorporated by ensuring that the expected gate conflict probability of two flights assigned to the same gate at the same time does not exceed a user-specified limit. Historical data of flight movements were used to predict the probabilities of flights being present at the apron. From the historical data the two most influential predictors, airline identity and origin/destination region, were used in a regression model to estimate the departure and arrival time delay cumulative probability distribution. With these probability distributions, a so-called flight presence probability was created, which served as input for the stochastic gate constraint.

Pternea and Haghani [49] proposed an integrated framework for reassigning flights to gates in case of schedule disruption. The authors developed a binary integer model to assess the success of passenger transfers. Delay is randomly introduced for arriving and departing flight using statistical distributions for two disruption scenarios. For the first scenario arrival delay follows a normal distribution $N(40,10)$ and departure delay a $N(0,5)$ distribution. In the second scenario arrival delay follows a Gamma distribution with shape and scale parameters $\Gamma(3,1)$ and departure delay follows still a normal distribution, but with parameters $N(10,10)$.

In conclusion, the flight to gate assignment has been the main interest in integrating flight delay prediction into the optimization of operations at an airport. Interesting to note is that multiple papers stated that future work should be focused on improving the arrival/departure time prediction models ([19], [63]).

7

Literature Gap and Research Question

This chapter will discuss the observed literature gap in section 7.1, where also the objective of the research will be defined. Section 7.2 will elaborate on this by stating the research question and its sub-questions.

7.1. Literature Gap

The discussed research papers in this literature study have shown that machine learning for flight delay prediction has mainly been applied in classification problems. A few papers treated flight delay as a regression problem. Probabilistic flight delay estimations have only been made through modelling it as a random variable or providing a probability with respect to a flight being delayed or not. While there are methods to obtain probabilistic distribution predictions, these probabilistic methods have never been applied in the context of predicting individual flight delays. Predicting probabilistic distribution for individual flights has not been researched before and is therefore the observed research gap in this literature study.

The objective of this research is to obtain, using machine learning methods, probabilistic flight delay distributions for individual flights. Secondly, the goal is to investigate how these distribution predictions can be made practical by using them for the optimization of operations at an airport.

7.2. Research Question

Based on this research objective the main research question can be defined:

> **Which machine learning model, trained on historical flight schedule and weather data, produced accurate flight delay distribution predictions for individual flights, on a prediction horizon of one day before flight?**

Based on this main research question, the following sub-research questions are defined:

- What kind of ML methods can accurately predict flight delay distributions?
- What methods can be used to compare the performance of a predicted interval?
- What methods can compare the accuracy of a predicted distribution with the actual given point value?
- Which methods can compare different delay distributions with each other and based on which criteria?
- What method can be used to visualize and explain the results of what the algorithm has learned?

Some additional sub-research questions are defined, which are not necessary to answer the main research question, but are useful to have as implementation of flight delay distribution predictions:

- How can probabilistic models be integrated into the Flight-to-Gate assignment?
- How to measure the added value of integrated probabilistic models into the Flight-to-Gate assignment?

8

Proposed Methodology

Now that the objective of this research is defined, the proposed methodology on how to achieve this objectives will be discussed in this chapter. Section 8.1 will discuss what has to be done in order to obtain probabilistic flight distributions and discusses which models will be used. Section 8.2 will elaborate on how flight presence probabilities can be obtained and how they can optimize the flight to gate assignment at airports.

8.1. Obtaining probabilistic flight distributions for individual flights

The first step in this research will be the collecting and merging data. The data will consist of flight schedule data of Schiphol from 2011 till 2019 and weather data provided by KNMI. These two data sources will be merged into one data set, which will be cleaned and missing values will be imputed, or the rows deleted. Outliers will be detected and if needed removed from the dataset.

Once the data is cleaned, exploratory data analysis will be performed on the data set. The purpose of this is to find significant patterns and trends by using statistical methods (e.g. pearson correlation test and significant t-test). These statistical methods are applied to obtain insights and better understanding of the data, instead of blindly fitting a model to the data.

Once the data is explored and statistical trends and patterns are found a model can be trained to predict flight delay. Before actually training the models, different steps will be taken, such as: dealing with the imbalanced dataset (approximately 80% on time and only 20% of flights delayed). Features will be selected and created, those new features should preferably incorporate domain knowledge. Examples are crowdedness at the airport (number of flight arriving/departing per hour) or time shifted weather features.

Categorical features should be encoded (e.g. airline and airport) and some features will need to be transformed trigonometrically to account for periodicity (e.g. hour or day of the week). Features should also be standardized, to be able to compare features which have different units and scale in order to contribute equally to the analysis. More information about these steps can be found in appendix 9.

With the feature set ready, it is time to train the prediction models. Four different models are selected as they represent interesting fundamental differences. The first difference is presented when the 4 models are divided into 2 pairs. The difference between those 2 pairs is how the distributions are obtained, by making n predictions or by predicting the parameters of a distribution. Each pair itself can then be subdivided again as they represent interesting difference as well, which are explained in the text below.

The first model which will be used is a Random Forest algorithm which consists of an ensemble of decision trees. From table 4.1 it was concluded that Random Forest methods have been used widely and are outperforming other algorithms consistently on binary classification of flight delay. On regression tasks they outperformed other models as well (Lee and Malik [39]). In this research the focus lies on obtaining (probabilistic) distributions of flight delay, an ensemble of decision trees will be alternated to predict a distribution of delay values instead of predicting a single value by majority voting or averaging.

The second model which is chosen is a neural network as it comes from a different family of algorithms. Neural networks have seen a significant increase in the last couple of years in their usage for predicting flight delay and have outperformed other models ([67]) with flight delay regression tasks. To obtain a distribution of n predictions, the choice can be made to train n neural networks or use the earlier discussed drop-out method. It is chosen to use the drop-out method to obtain n different predictions, because the training of a single neural networks is already computationally expensive. The training of n different neural networks will become extremely computationally expensive and not favorable anymore to obtain sufficient predictions.

These first two models (Random Forest, Drop-out neural network) create distributions empirically by making n predictions, where the next two models are selected as they predict distributions directly by predicting their parameters. Two models of this type have been chosen in order to compare their individual performance. The underlying assumption and complexity is different for both models.

The third selected model (first of predicting distribution directly) is the Mean Variance Estimation (MVE) method, which uses a neural network to predict the parameters of an assumed distribution. The advantage of this method is that the variance of the estimation is dependent on its input, but with the downside that a target distribution needs to be assumed, which is often not known in real life.

The fourth selected model is the Mixture Density Networks (MDN) method, which extends on the idea of the MVE by not predicting a single distribution, but a mixture of distributions. With use of this model no underlying distribution has to be assumed as theoretically every distribution can be modeled with sufficient Gaussian distributions. This fourth model is selected to research whether this advantage results in an increase in performance of the model.

In order to evaluate and compare the four models equally certain performance criteria have to be used. All 4 method are capable of predicting intervals (PIs) based on their distributions, with a lower (PI_i^L) and upper bound (PI_i^U) of the i_{th} interval. These PIs will be evaluated from both a coverage probability and a width of interval perspective. A combined metric called the coverage-width based criterion (CWC) will be used to evaluate the intervals as is discussed in section 5.1.3.

Other metrics which will used originate from regression problems. From a distribution with an underlying probability the expected value can be obtained. It is interesting to compare this expected value with the actual delay value. The Mean Squared Error (MSE) and Mean Absolute Error (MAE) will be evaluated to compare the performance of the models in this manner (see appendix 9.11 for the formula's).

8.2. Integration flight presence probabilities into Flight-Gate Assignment

The best scoring algorithm will be chosen for the integration with the flight to gate assignment for Schiphol. The selected model will predict arrival and delay distributions for individual flights such that presence probabilities for aircraft at Schiphol can be created. The prediction horizon used for these distribution predictions will be set at one day, because the gate planning itself is made one day in advance. Meaning, that it is not necessary to obtain the predictions earlier and that more accurate weather predictions can be used.

These presence probabilities will be integrated into the flight to gate assignment using the method described in van Schaijk and Visser [63]. The outcome of the optimization problem will be the 'updated gate planning' for Schiphol. In order to compare the original gate planning fairly with the 'updated gate planning', the same constraints of the original optimization algorithm should be run with the additional overlap probability constraint included(van Schaijk and Visser [63]).

As Schiphol's optimization algorithm is unknown and creating a similar model is a thesis on its own, combined with the fact that for a required 5min discretization size the problem cannot be solved on an ordinary laptop anymore, this approach is not a feasible. A workaround could be to consider a simpler model and run it for a part of Schiphol's schedule, but this gives the problem that the comparison between the actual gate schedule of Schiphol and the 'updated gate schedule' is not fair anymore.

Therefore a simple FTGA-model will be developed which will be ran in a controllable environment (number of constraint and gates), such that a fair comparison can be made and solutions are obtained within reasonable time. As input for the FTGA-model, real-life flight data from Schiphol will be used (e.g.scheduled time of arrival and departure, aircraft type). The actual arrival and departure times of flights will be used to evaluate both gate-schedules because this will give a more realistic comparison between the two schedules.

For the comparison of both the schedules, the number of gate conflicts will be recorded, based on when a certain gate schedule will be followed compared with the actual arrival/departure times. Based on whether the counted number of gate conflicts was lower for the 'updated' gate planning or not, it can be concluded whether this approach resulted in an added value for the gate planning. Meaning, that less conflicts occurred during the day of operations itself, resulting in a reduction of workload for the gate planners.

9

Background information on Machine Learning

Flight delay will be evaluated in the context of Machine Learning in this Thesis. But what is Machine Learning actually? The oxford dictionary described it as "a type of artificial intelligence in which computers use huge amount of data to learn how to do tasks rather than being programmed to do them". As the field of Machine Learning comes with it's own nomenclature and concepts, this appendix is dedicated on elaboration on these as background information for the thesis.

9.1. Types of Machine Learning

Within the field of Machine Learning, three different main types can be distinguished. Where the first two are the most well known and the third one is less commonly used. There difference in short can be described as:

1. **Supervised learning**, deals with data that is already labeled by the 'supervisor' and the machines uses these labeled examples to learn from (e.g. picture of a cat or dog)
2. **Unsupervised learning**, deals with the problem where the data is not labeled. The machine needs to find on it's the pattern (e.g. pile of animal photos and fining out who is who)
3. **Reinforcement learning**, deals with problem where no data, but their is an environment to interact with from which the algorithms can learn

As this Thesis will evolve around the prediction of flight delay, where the value or category of delay is known, supervised learning techniques will mainly be used (unsupervised techniques like clustering could be used in prepossessing stage). A distinction can be made within supervised learning for the three main tasks it can perform:

1. **Classification**, deals with problems where the algorithm has to identify to which category an new observation belongs. Classification problems can be binary (2 classes) and multi class (3 or more classes)
2. **Regression**, deals with the problem where the algorithm predicts a value based on a new observation.
3. **Probabilistic**, deal with the problem where the algorithm predicts the uncertainty of the estimations, often by predicting a distribution.

9.2. Machine Learning Algorithms

An ever increasing number of algorithms exists in the field of Machine Learning. This wide variety of algorithms can be reduced to briefly three different families of algorithms:

Classical Machine Learning algorithms

These algorithms have been around the longest time and have a solid foundation in statistical theory. These algorithms span a wide variety of methods, but don't fall under tree based or neural network methods. Classical machine learning algorithms are easier to interpret than neural networks, but often perform less on bigger datasets as neural networks. Examples of classical machine learning algorithms are: Linear Regression, Logistic Regression, Support Vector Machine, Bayesian algorithms and K Nearest Neighbors (KNN).

Tree based algorithms

Tree based methods are one of the most used methods in supervised learning problem and as there name describes, there structure is similar to that of trees. They consist out of different elements: a root, branches, nodes and leaves, see figure 9.1. At each node, two branches split of going to an internal node or a leaf node. To decide which branch is followed a certain condition is checked for a feature value resulting in a 'True' and 'False' branch. The learning process of a decision tree is based on learning simple decision rules inferred from the training data. The leaf node contain the target values, which is obtained by following the rules created by the decision tree.

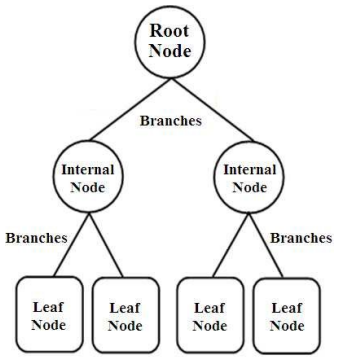


Figure 9.1: An example Decision Tree with its elements labeled. Taken from Sá et al. [54]

By combining multiple decisions trees a random forest method is created, which can be used for both classification and regression tasks. It uses bagging and feature randomness (therefore its name) when building each individual tree. It does to create uncorrelated forest of trees whose prediction combined are more accurate than the prediction of any individual tree. It can be thought of as making a decision based on the knowledge of a group, instead of only the knowledge of an individual. This technique of combining predictions of multiple models is called ensemble and is often used in the setting of tree methods (but not limited to them). Examples of tree based methods are: Decision Trees, Random Forest (ensemble of decision trees) and Extreme Gradient Boosting (XGBoost).

Neural network methods

Neural Network methods derive their name as they loosely model the human brain, by cells connection to one an other to form a complex network. Each neural network has the the same three key element in its architecture, see figure 9.2. First an input layer where feature values are received. Second, a hidden layer consisting out of nodes (cells) with associated weights and biases and an activation function, which maps the input to an output value. Often multiple hidden layers are used, where for 'deep learning' networks, numerous hidden layers are used to capture complex relations. Lastly, the output layer which gives the prediction of the algorithms, which both can be a class or an actual value. Examples of often used neural networks are Deep Learning network, Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) methods.

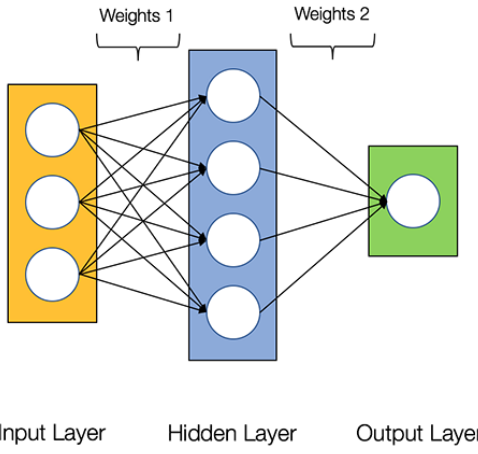


Figure 9.2: Architecture of a Neural Network. Taken from Loy [41]

9.3. Minimizing loss function

Machine learning algorithms learn by using a loss function. This loss function evaluates how well the algorithms predictions compare to the actual values. If the predictions are far off, a large loss numbers will be obtained. With the help of a minimization function, the outcome of the loss functions will be reduced by changing the parameters of the model. This process of changing the parameters to obtain better predictions (lower the loss function) is called 'learning' of the algorithm.

Different types of loss function exist and they can broadly be classified into two major categories based on the task the perform: classification or regression. The most known loss function for regression task is MSE (Mean Squared Error), which takes the mean of squared difference between the actual and predicted value. For classification problems different loss function are needed, the most familiar ones are binary crossentropy for binary classification and categorical crossentropy for multi classification problems.

9.4. Dealing with imbalanced data set

Imbalances in the data set are a problem for machine learning algorithms, especially for classifiers as they will introduce bias in the learning process Rocca [52]. Different methods exists to rebalance the dataset:

- **Undersampling** is a technique which takes random samples from the majority class. The number of random samples is defined by the size of the minority class, resulting in a balanced dataset.
- **Oversampling** is a technique is which some point of the minority class are replicated in order to balance the size of both classes.
- **Generating synthetic data** is a technique in which synthetic data point are created for the minority class. A well known technique is Synthetic Minority Oversampling TEchnique (SMOTE) by Chawla et al. [11]

9.5. Feature engineering

Feature engineering is the process of finding, creating and removing features, such that the most useful features can be used by the model. It is a crucial step in machine learning as unnecessary features decrease the speed of training, decrease the interpretability of the solution and decrease the generalization performance of the model. The process of feature selection can consist out of many step, depending on the quality of the dataset. Examples of these steps are removing features based on missing values or high correlation, removing features which have no predicting power. Other examples are imputing missing values, creating new features based on domain specific knowledge.

9.6. Feature encoding

As computers can only work with numerical values and no categorical inputs are valid. This process of transforming categorical variables into numerical variables such that can by used by the algorithm is called feature encoding. The most common types of encoding are described below:

- **Label encoding** assign numerical values to the classes. The resulting label are for example 0,1,2, etc. Problem with this approach is that although there is no relation between the classes, the algorithm might consider them to be ordered.
- **One-hot-encoding** produces separate columns for each class and the presence of a class will be indicated with a 1, and absence of that class with a 0. Upside of this method is that no order is introduced, but comes with downside of many new columns being generated dependent on the number of classes.
- **Frequency encoding** uses the number of occurrences of a certain class in the dataset as its encoding. The higher the number of occurrences in the dataset, the higher the encoding number.
- **Target mean encoding** uses the target value's in order to encode a feature. For every class the mean value of the target variable over every instant of that class is used as decoding value.
- **Trigonometric encoding** is used for encoding features which have a cyclical nature, for example for hours of the day, as after 24:00, 01:00 comes. Sines as well as cosines are used to encode a feature.

9.7. Features scaling

Feature scaling is needed in the data pre-processing stage to handle highly varying magnitudes or units of features. When feature scaling is not done, a machine learning algorithm tends to weight greater values higher than smaller values, regardless of the unit of values. The two main methods used for feature scaling are standardization and normalization. With normalization the values are being bound between two number, typically [0,1] or [-1,1]. While standardization transform the data to have zero mean and a variance of 1, making the data unitless.

9.8. Training, testing and validating datasets

The obtained dataset serves multiple purposes, as the model will be trained on it, it's performance will be validated and tested on an unseen dataset. The training set is defined as the sample of data used to fit the model. The validations dataset is used to provide an unbiased evaluation of a model fit on the training dataset while tuning the model's hyperparameters. The test dataset will only be used in the last stage, where it will provide an unbiased evaluation of the final model by evaluating its performance on unseen data.

A useful technique for working with a limited dataset is cross-validation, which is a resampling procedure to validate the performance of a trained algorithm before showing it the test data. Cross validating is useful as the performance of a model is checked multiple times, meaning that outliers or other anomalies are moved randomly around. This gives a more realistic view from the model's performance as multiple validations occur and not just one were potentially outliers are unevenly distributed. The training data is divided into k folds, where k-1 folds are used for training and 1 fold for validating the performance. The folds are rotated such that k validations of the model's performance can be obtained as can be seen figure 9.3.

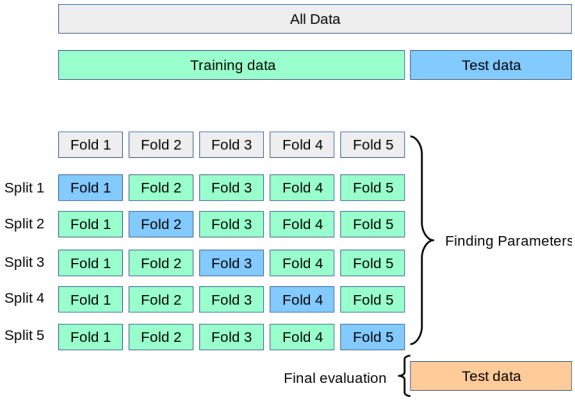


Figure 9.3: Cross validating datasets. Taken from Scikit-learn developers [56]

9.9. Under and over fitting

Poor performance of a model is caused either by underfitting or overfitting of the data. Both are visualized in figure 9.4, as well as a well fitted/robust model. In the case of underfitting, the model cannot capture the complex behaviour of the data, in other words the model is too simple, resulting in poor performance. An example is a linear model that cannot capture non-linear behaviour.

Overfitting occurs when the model is 'too well' fitted to the training data and has the tendency to capture the noise of the dataset instead of its general trends. In the case of overfitting, the model is not generalizing enough for new data, meaning that it doesn't perform adequately on data it has never seen and therfore results in poor performance.

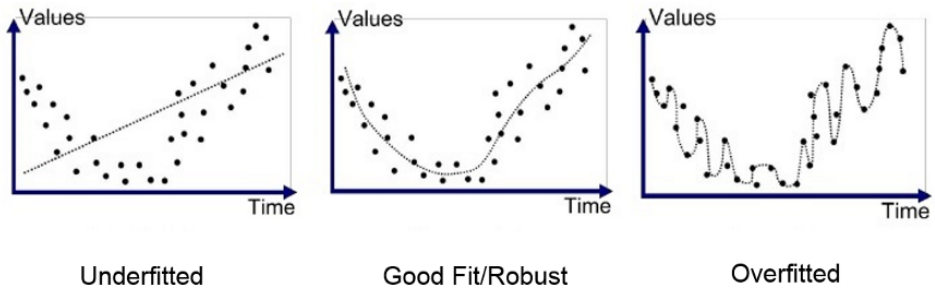


Figure 9.4: Under and over fitting. Taken from Bhande [5]

9.10. Interpretability of solution

An important part of machine learning projects is the interpretability of the solution. A good definition of interpretability is given by Miller [43]: "Interpretability is the degree to which a human can understand the cause of a decision". This is especially relevant for project were decision are made based on the outcomes of a model it is important to be able to explain why a model made a certain decision instead of making decision from a 'black box'.

Different techniques exists to explain and identify the process behind a model prediction's. One of the most used technique is called 'feature importance', which is used to assigns scores to input features of the model which indicate the relative importance of each feature when making a prediction. Feature importance can be applied to both classification as regression problems. Depending on the type of problem and the chosen model, feature importance metrics can obtained via different methods. Examples of methods are: coefficients of linear models, permutation importance, gini importance or mean decrease in impurity for tree models and SHapley Additive exPlanation (SHAP) values.

9.11. Performance metrics for models

In order to compare different models among each other, performance metrics are needed. Dependent on the type of problem different metrics can be used. The most common used metric per type of machine learning problem are discussed below:

Performance metrics for classification

The different metrics used for classification problems can best be explained via a so called confusion matrix. For a general binary classification the confusion matrix in given in table 9.1. Where the definitions of True Positive (TP), False Postive (FP), False Negative(FN) and True Negative (TN) are given, which corresponds to what the model predicted and what the actual class was (True or False).

Table 9.1: Confusion matrix for a general binary classification

		Actual Class	
		Positive (1)	Negative (0)
Predicted Class	Positive (1)	TP	FP
	Negative (0)	FN	TN

Accuracy

Is the number of correct predictions divided by the total number of made predictions or in terms of elements of the confusion matrix, see equation 9.1.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{9.1}$$

Precision

Precision gives a measure of all the positive classes which are predicted, how many are actually positive. It can be taught of as the ability of the model to identify only relevant instances within the dataset. In terms of elements of the confusion matrix it can be formulated as in equation 9.2.

$$Precision = \frac{TP}{TP + FP} \tag{9.2}$$

Recall

Recall gives a measure of all the actual positive classes, how much were predicted correctly. It can be taught of as the ability of a model to find all the relevant instances within the dataset. In terms of elements of the confusion matrix it can be formulated as in equation 9.3.

$$Recall = \frac{TP}{TP + FN} \tag{9.3}$$

F1-score
In order to compare models on a single metric, the F1-score is introduces, which is harmonic mean of precision and recall. It can be computed using equation 9.4.

$$Recall = \frac{2 * Precision * Recall}{Precision + Recall} \tag{9.4}$$

Receiver Operating Characteristic (ROC) curve
The ROC curve plots the true positive rate (TPR) versus the false positive rate (FPR) as a function of the threshold of the model for classifying a positive. The true positive rate is has the same definition as Recall (equation 9.3) and the false positive rate (FPR) is defined as $FPR = \frac{FP}{FP+TN}$. See figure 9.5 for an example of an ROC curve.

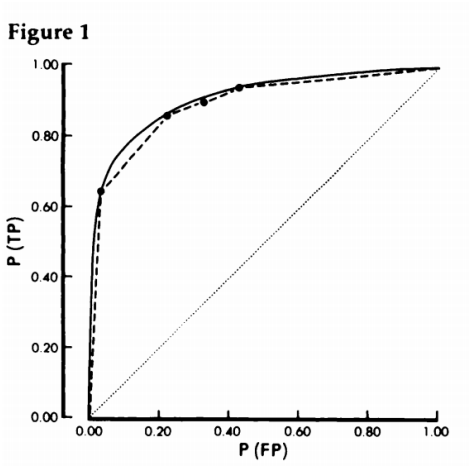


Figure 9.5: Example of an ROC curve. Adapted from Hanley and McNeil [28]

Area under the ROC cuve (AUC)
When the graph under the ROC is integrated, we obtain the performance metric AUC, which can range from 0 to 1. AUC represents the degree of separability the model has, how capable the model is in distinguishing between classes. The higher the AUC score, the better the model is at predicting 0s as 0s and 1s as 1s. Note that when the AUC score is 0.5 (dotted straight line in figure 9.5), the model has no class separation capacity whatsoever.

Performance metrics for regression

Mean Absolute Error (MAE)
The MAE is the average of the absolute difference between the the actual values (y) and the predicted values ($\hat{y}$), also called the residuals. It is a linear score, meaning that all individual difference have the same weight. Mathematically the MAE is given by equation 9.5, where n denotes the number of samples.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{9.5}$$

Mean Squared Error (MSE)
The MSE metric uses the same variables, but not the difference between the actual value and predicted value is squared. It represents the sample standard deviation of the residuals and is mathematically given in equation 9.6.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{9.6}$$

Performance metrics for probabilistic predictions

Based on expected value of distribution
As the probabilistic prediction outputs a distribution, this distribution can be integrated to obtain the expected value. Where for a discrete distribution equation 9.7 can be used and for a continuous distribution equation 9.8. These expected values can be used as the predicted value in the MAE and MSE metrics (see section 9.11).

$$E[X] = \sum_{i=1}^{k} x_i p_i \tag{9.7}$$

$$E[X] = \int_{\mathbb{R}} x f(x) dx \tag{9.8}$$

Negative log likelihood

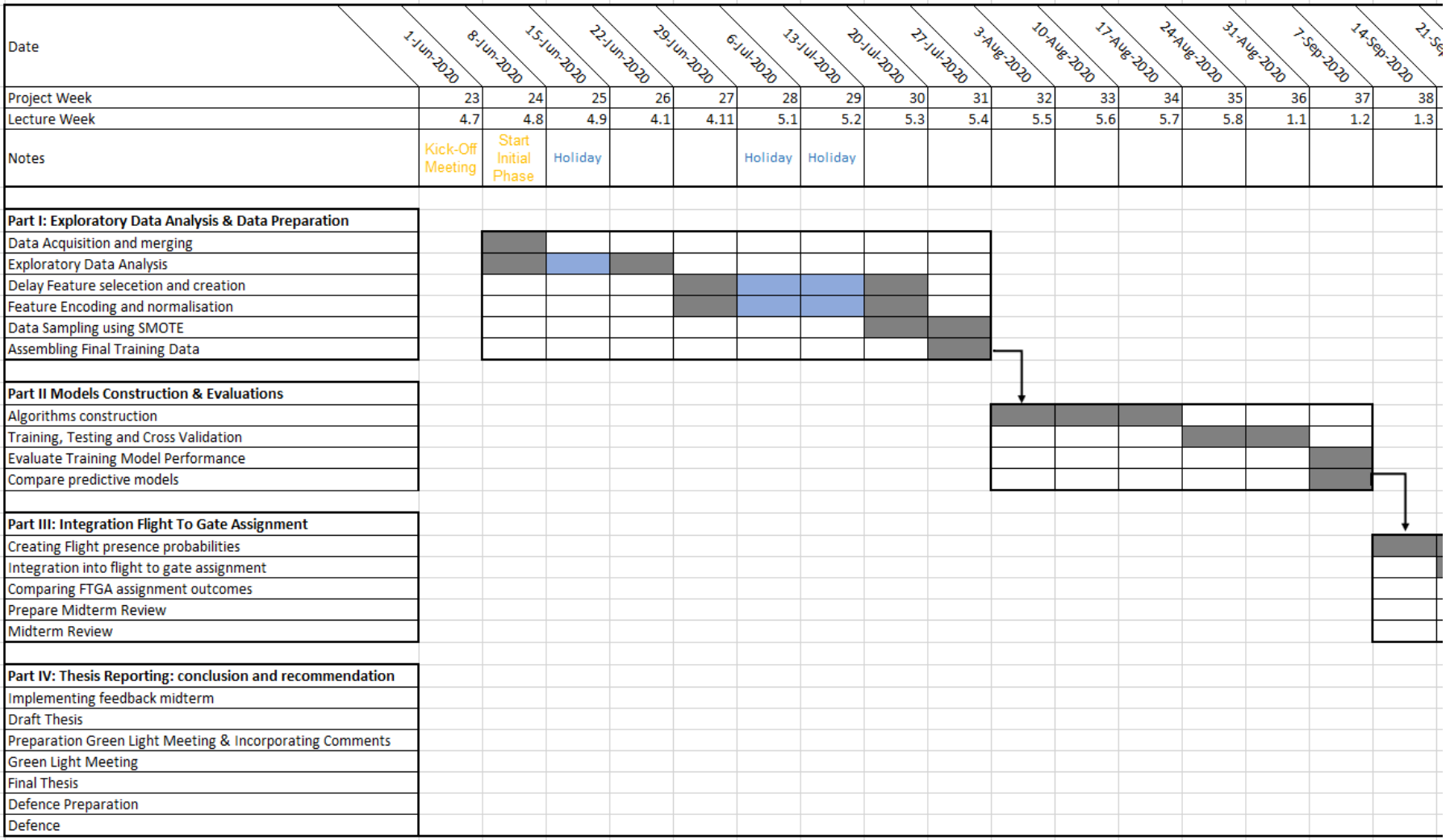
As the MSE method assumes that the standard deviation of the underlying distributions is constant, a different metrics should be considered as well. For this the mean negative log-likelihood comparison can be used as it incorporates the standard deviation to obtain a more nuances comparison between algorithms.

9.12. Hyper parameter tuning

During the training process of a model, its parameters are tuned in order to obtain a low as possible loss function. But before the learning process can begin, model parameter values need to be set, these parameters are the hyperparameters of a model. Examples of hyperparameters for tree based methods are the depth of the tree and amount of trees in a random forest method. For a Neural Network, the number of hidden layers and notes in these layers are hyperparameters. To obtain the best possible results for a model, these hyperparameters need to be tuned as well, as they influence the performance of a model. Two different strategies for optimizing hyperaparameters are: Grid Search and Random Search. Where the difference is that for Grid Search ever combination of a specified subset of hyperparameters is evaluated, which will results in an optimal combination, but has the drawback of being computationally expensive. Random Search select a random subset of the specified hyperparameters, resulting in a decreased processing time, but not guaranteeing the optimal combination.

10

Gantt Chart



Date	1-Jun-2020	8-Jun-2020	15-Jun-2020	22-Jun-2020	29-Jun-2020	6-Jul-2020	13-Jul-2020	20-Jul-2020	27-Jul-2020	3-Aug-2020	10-Aug-2020	17-Aug-2020	24-Aug-2020	31-Aug-2020	7-Sep-2020	14-Sep-2020	21-Sep
Project Week	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	
Lecture Week	4.7	4.8	4.9	4.1	4.11	5.1	5.2	5.3	5.4	5.5	5.6	5.7	5.8	1.1	1.2	1.3	
Notes	Kick-Off Meeting	Start Initial Phase	Holiday			Holiday	Holiday										
Part I: Exploratory Data Analysis & Data Preparation																	
Data Acquisition and merging																	
Exploratory Data Analysis																	
Delay Feature selecetion and creation																	
Feature Encoding and normalisation																	
Data Sampling using SMOTE																	
Assembling Final Training Data																	
Part II Models Construction & Evaluations																	
Algorithms construction																	
Training, Testing and Cross Validation																	
Evaluate Training Model Performance																	
Compare predictive models																	
Part III: Integration Flight To Gate Assignment																	
Creating Flight presence probabilities																	
Integration into flight to gate assignment																	
Comparing FTGA assignment outcomes																	
Prepare Midterm Review																	
Midterm Review																	
Part IV: Thesis Reporting: conclusion and recommendation																	
Implementing feedback midterm																	
Draft Thesis																	
Preparation Green Light Meeting & Incorporating Comments																	
Green Light Meeting																	
Final Thesis																	
Defence Preparation																	
Defence																	

Figure 10.1: Gannt chart part 1

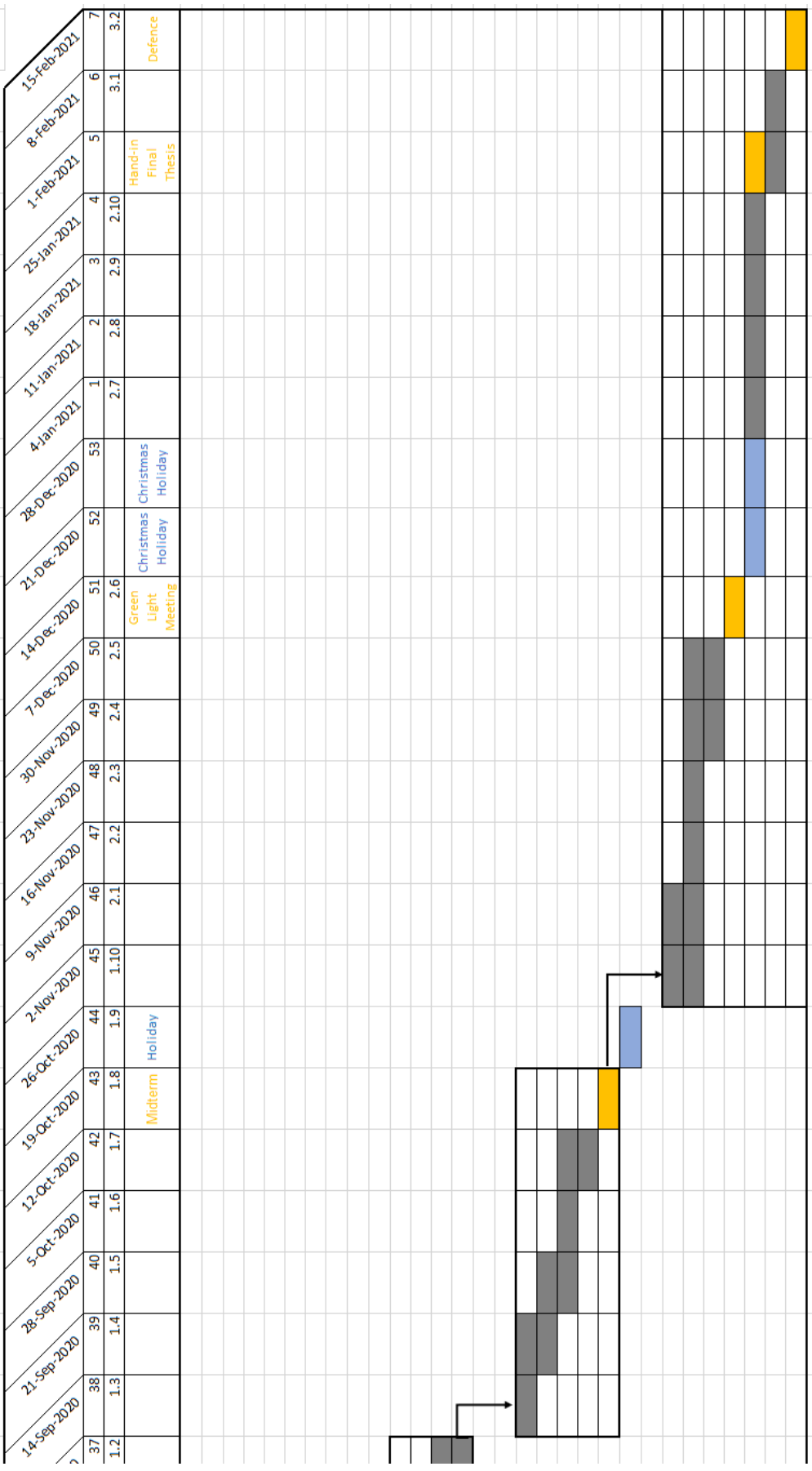


Figure 10.2: Gannt chart part 2

III

Supporting work

1

Negative Log Likelihood loss functions

The Mean Variance Estimator and Mixture Density Network methods do not use a standard loss function such as the MAE or the MSE. This section will treat the 'custom' loss functions for both methods.

1.1. Loss function Mean Variance Estimator

Nix and Andreas [45] proposed the Mean Variance Estimator method in which a Maximum Likelihood Estimation (MLE) loss function was used. Based on the assumption of normally distributed errors around $f(\vec{x}_i)$, in which $\vec{x}_i$ represent the input features of the neural network in vector notation, the target probability density function can be written as in eq. (1.1). In this formula $y(\vec{x}_i)$ represents the predicted mean of the distribution by the neural network and $\sigma^2(\vec{x}_i)$ represents the predicted variance. d_i is the i^{th} sample of the actual data, where NN represent the neural network as input for the loss function.

$$P(d_i|\vec{x}_i, NN) = \frac{1}{\sqrt{2\pi\sigma^2(\vec{x}_i)}} exp\left(-\frac{[d_i - y(\vec{x}_i)]^2}{2\sigma^2(\vec{x}_i)}\right) \tag{1.1}$$

Taking the natural log on both sides results in eq. (1.2), which is called the log likelihood function.

$$ln\,[P(d_i|\vec{x}_i, NN)] = -\frac{1}{2}\ln(2\pi) - \frac{1}{2}\ln[\sigma^2(\vec{x}_i)] - \frac{[d_i - y(\vec{x}_i)]^2}{2\sigma^2(\vec{x}_i)} \tag{1.2}$$

The log likelihood function is maximized, such that the distribution will be found from which it is most likely that the i^{th} sample of actual data came from. As the first term on the right in eq. (1.2) is constant, it can be ignored for maximization. Since loss functions are by definition minimization problems, the negative of the log likelihood function will be minimized. This results in the loss function (C_{MVE}) used for the MVE method, which is given in eq. (1.3) and called the negative log likelihood loss function.

$$C_{MVE} = \frac{1}{2}\sum_i \left(\ln[\sigma^2(\vec{x}_i)] + \frac{[d_i - y(\vec{x}_i)]^2}{\sigma^2(\vec{x}_i)}\right) \tag{1.3}$$

1.2. Loss function Mixture Density Network

Mixture Density Networks (Bishop [7]) extend the idea of the Mean Variance Estimator to predicting a distribution consisting out of multiple normal distributions. The target probability density function of an MDN is a linear combination of normal distributions, also called kernels, given in eq. (1.4).

$$P(d_i|\vec{x}_i) = \sum_{j=1}^{m} \alpha_j(\vec{x}_i)\phi_j(d_i|\vec{x}_i) \tag{1.4}$$

Where m is the number of components in the mixture. The parameters $\alpha_j(\vec{x}_i)$ are called the mixing coefficients, which are functions of the input , which must satisfy that there sum equals 1. The functions $\phi_j(d_i|\vec{x}_i)$ are the conditional density functions of the target d_i for the j^{th} kernel. The loss function is defined in the same way as for the MVE method, by taking the natural log on both sides and writing it as a minimization problem, but now the m components of the mixture need to be taken into account.

2

Features

This supporting chapter will elaborate on the features which were created in this research in more detail. After, the feature selection methods will be discussed with their results for the arrival and departure models.

2.1. Feature creation

The extra created features in this research try to provide the model with more 'domain knowledge' to obtain better probabilistic predictions. The following features were created:

Distance

The feature 'Distance' is included based on the distance between Amsterdam Schiphol airport and the origin/destination airport. It is calculated using the great circle distance, which is approximated with the haversine formula, which assumes that the Earth is a perfect sphere. As input the haversine formula takes the lateral (ϕ_1 and ϕ_2) and longitudinal (λ_1 and λ_2) coordinates of both the airports in radians. The central angle θ between the two points on the earth's sphere is defined in eq. (2.1), in which D is the great circle distance and R is the Earth's radius.

$$\theta = \frac{D}{R} \tag{2.1}$$

The haversine of the central angle θ is defined as in eq. (2.2).

$$hav(\theta) = hav(\phi_2 - \phi_1) + cos(\phi_1) \cdot cos(\phi_2) \cdot hav(\lambda_2 and \lambda_1) \tag{2.2}$$

Rewriting it to the great circle distance D, by combining eq. (2.1) and eq. (2.2) results in:

$$D = 2 \cdot R \cdot arcsin(\sqrt{sin^2(\frac{\phi_2 - \phi_1}{2} + cos(\phi_1) \cdot cos(\phi_2) \cdot sin^2(\frac{\lambda_2 - _1}{2}))} \tag{2.3}$$

The inputs for the lateral and longitudinal coordinates of the airports were obtained from the database of OurAirports [46].

Number of flights last hour Number of flights coming hour

To provide the model with information about the airport's crowdedness at the moment of arrival/departure of a flight, two features were created. These are Number of flights last hour and Number of flights coming hour. Both features count the number of arrival and departure flights which are scheduled in the last hour or in the coming hour. As these features are calculated based on the schedules and not actual arrival/departures, they are known one day in advance. It is assumed that the actual number of arrival/departure equals the schedules values. In reality, some flights are cancelled, which are for the Schiphol data set 1.6 %. Next, there are also flight which arrive/depart more than one hour early or more than one hour late, this is 5.1% of the Schiphol data.

Temperature above 3 degrees
De-icing precautions are needed once the temperature at Schiphol airport drops down 3 degrees. As these precautions take extra time, they could influence the delay of flights. Therefore, a binary features is included, which equals 1 when the temperature is above 3 degrees and equals 0 if the temperature is 3 degrees or less.

Cross gust north-south runways
Amsterdam Schiphol Airport has 3 runways directed in the north-south direction, the Aalsmeerbaan, the Zwanenburglaan and the Polderbaan ([55]). These runways cannot be operated when wind gust speeds components perpendicular on these runways exceeds 30 knots. Closing these 3 runways results in a drastic reduction of the landing/arrival capacity of Schiphol. The feature Cross gust north-south runways ($v_{\perp north-south}$) is defined as the absolute value of the wind gust component perpendicular to these north-south runways. It is calculated using eq. (2.4), where wind direction (ϕ_{wind}) is given in degrees with respect to the true north and v_{gust} is the wind gust magnitude in knots.

$$v_{\perp north-south} = |v_{gust} \cdot \sin(\frac{\phi_{wind} \cdot \pi}{180})| \tag{2.4}$$

Cross gust above 30 knots
The feature Cross gust above 30 knots is a binary feature defined to provide the model directly with information whether the north-south runways are allowed or not allowed to be operated. It equals 1 when the Cross gust north-south runways is above 30 knots and equals 0 when the value is below 30 knots.

2.2. Feature selection

To ensure that feature will be used which have predictive power and limit the number of features to avoid possible computational complexity issues, a feature selection was made. Features were selected with predictive power and features that are irrelevant or redundant for the models were removed. To select the features for the models in the different methods are used, which are discussed in this section. Note that all feature selection methods are only assessed on the train and validation set. This ensures that the test set is never seen and as general as possible for evaluating the models.

2.2.1. Pearson Correlation

Pearson correlation is a method that evaluates the linear correlation between the input features and the target variable and the linear correlations between features themselves. Features which have a high linear correlation with each other can lead to multicollinearity for some machine learning models. Pearson correlation is calculated between two features (x_i and y_i) with formula eq. (2.5). In this formula, n represent the sample size, where $\bar{x}$ and $\bar{y}$ represent the means respectively of the features x and y.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.5}$$

Pearson correlation values range from -1 to 1. A value above 0 implies a linear relation in which if x increases, y increases as well. A value of 0 indicates that a linear equation if found on which all of the samples as x and y lie. Pearson correlation values between -1 and 0 indicate a linear relation in which if x increases, y decreases. A value of -1 indicates a linear equation with a negative coefficient on which all values lie on the exact line. A value of 0 implies that there is no linear correlation between the two features, however, there still can be a non-linear relation.

The Pearson correlation for every feature combination is calculated, which results in a so-called 'Pearson correlation matrix'. Table 2.1 shows a selection of the Pearson correlation matrix, as the whole matrix is too big to fit on paper. In this selection, all features are found with are dropped for having an absolute correlation value of 0.8 or higher. For example, the features 'Seats' and 'Passengers' are correlated with a coefficient of 0.91. It was chosen to drop the feature 'Passengers' as this feature is less likely to be known precisely one day in advance. Based on the Pearson correlation features selection method, the following features were dropped: Month of year (sin), Month of year (cos), Passengers, Dewpoint temperature and Gust wind speed.

	Delay	Month of year (sin)	Month of year (cos)	Day of year (sin)	Day of year (cos)	Seats	Passengers	Temperature	Dewpoint temperature	Wind speed	Gust wind speed
Delay	1.00	-0.01	-0.02	-0.02	-0.01	0.00	0.02	0.03	0.04	0.10	0.11
Month of year sin	-0.01	1.00	0.00	0.96	-0.24	-0.00	-0.02	-0.10	-0.25	0.05	0.05
Month of year cos	-0.02	0.00	1.00	0.23	0.96	0.00	-0.06	-0.81	-0.73	0.14	0.15
Day of year sin	-0.02	0.96	0.23	1.00	-0.01	-0.00	-0.03	-0.29	-0.42	0.09	0.09
Day of year cos	-0.01	-0.24	0.96	-0.01	1.00	0.00	-0.05	-0.76	-0.66	0.13	0.13
Seats	0.00	-0.00	0.00	-0.00	0.00	1.00	0.91	-0.04	-0.00	-0.04	-0.04
Passengers	0.02	-0.02	-0.06	-0.03	-0.05	0.91	1.00	0.01	0.04	-0.05	-0.04
Temperature	0.03	-0.10	-0.81	-0.29	-0.76	-0.04	0.01	1.00	0.87	-0.01	-0.02
Dewpoint temperature	0.04	-0.25	-0.73	-0.42	-0.66	-0.00	0.04	0.87	1.00	-0.09	-0.09
Wind speed	0.10	0.05	0.14	0.09	0.13	-0.04	-0.05	-0.01	-0.09	1.00	0.97
Wind gust speed	0.11	0.05	0.15	0.09	0.13	-0.04	-0.04	-0.02	-0.09	0.97	1.00

Table 2.1: Perason correlation matrix with dropped features

2.2.2. Permutation Importance

As Pearson correlation only looks at the linear correlation between features and the used algorithms (neural networks and random forest) in this research are of non-linear nature a second method for feature selection was used. This second feature selection method is called Permutation Importance and evaluates how important a feature is for making predictions. It shuffles the values of a single feature randomly and evaluates how much the models' performance decreases by shuffling that single features. The decrease in performance is an indication of the importance of those features for making predictions. Is the decrease is significant, then that means that the feature is important. Is there almost no decrease in performance, that indicates that the feature is not important for making predictions. fig. 2.1 display the permutation importance per feature for arrival flights and fig. 2.2 and for departure flights.

For Arrival flights in fig. 2.1 the features Distance and Seats are the most dominate for predictions. For Departure Flights in fig. 2.2, the features Minutes of day, cosine and sine or most important. Followed by the feature Humidity, which was arrival flights the 4th most important. As criteria for feature selection, a threshold score of 10% of the most important feature was selected. This relative method was selected as it can be used independently on the arrival and departure set. Based on the 10% threshold the features: Fright, Temperature above 3 degrees, Schengen and Cross gust above 30 knots were dropped for the arrival flights set. For the departure flights set only the feature Cross gust above 30 knots was dropped.

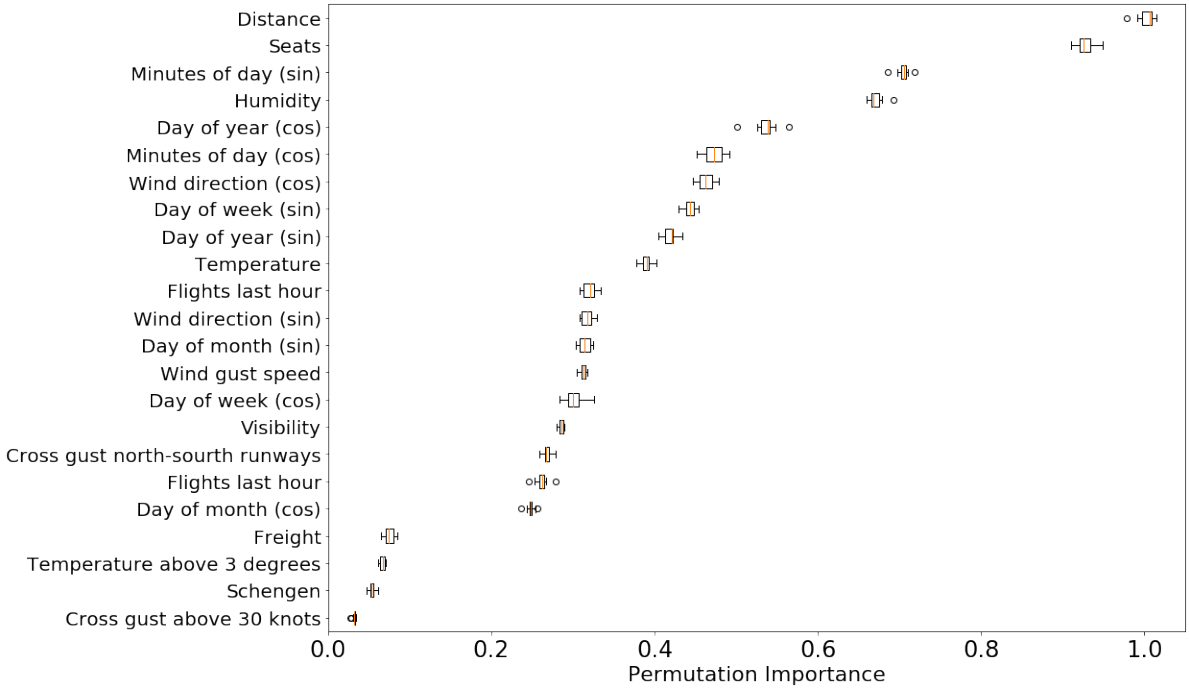


Figure 2.1: Permutation Importance of Arrival Flight features

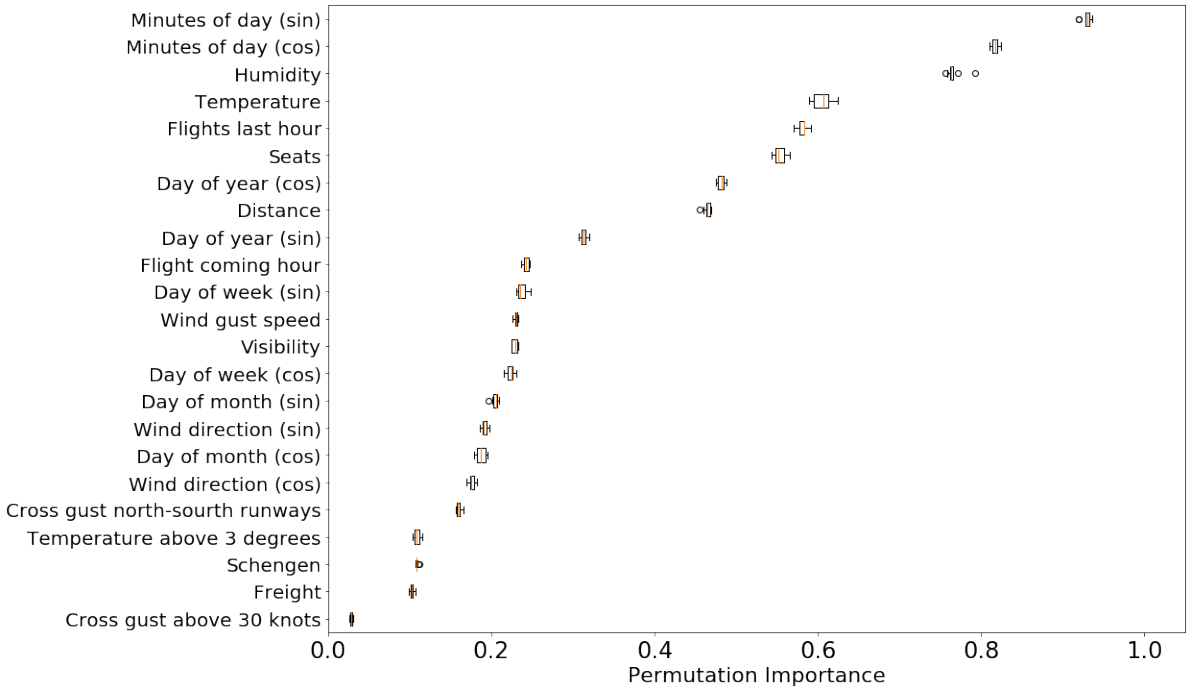


Figure 2.2: Permutation Importance of Departure Flight features

Bibliography

[1] Sachin Abeywardana. Deep Quantile Regression, 2018. URL https://towardsdatascience.com/deep-quantile-regression-c85481548b5a.

[2] Hugo Alonso and António Loureiro. Predicting flight departure delay at porto airport: A preliminary study. In IJCCI 2015 - Proceedings of the 7th International Joint Conference on Computational Intelligence, volume 3, pages 93–98, 2015.

[3] Michael Ball, Cynthia Barnhart, Martin Dresner, Mark Hansen, Kevin Neels, Amedeo Odoni, Everett Peterson, Lance Sherry, Antonio Trani, Bo Zou, Rodrigo Britto, Doug Fearing, Prem Swaroop, Nitish Uman, Vikrant Vaze, and Augusto Voltes. Total Delay Impact Study. Nextor, pages 5–14, 2010.

[4] Loris Belcastro, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Using Scalable Data Mining for Predicting Flight Delays. ACM Transactions on Intelligent Systems and Technology, (January), 2016.

[5] Anup Bhande. What is underfitting and overfitting and how to deal with it. 2018. URL medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal\-with-it-6803a989c76.

[6] Christopher Bishop. Pattern Recognition and Machine Learning. Number 1. 2006.

[7] Christopher M Bishop. Mixture Density Networks. Neural Computing Research Group Report: NCRG/94/004 Aston University, 1994.

[8] Sam Blake. How to generate neural network confidence intervals with Keras. Medium, Hal24K TechBlog, 2019. URL https://www.medium.com/hal24k-techblog/how-to-generate-neural-network-confidence-intervals-with-keras-e4c0b78ebbdf.

[9] Oliver Borchers. A Hitchhiker's Guide to Mixture Density Networks, 2019. URL https://towardsdatascience.com/a-hitchhikers-guide-to-mixture-density-networks-76b435826cca.

[10] Navoneel Chakrabarty. A data mining approach to flight arrival delay prediction for American airlines. IEMECON 2019 - 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference, pages 102–107, 2019.

[11] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE : Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16 (2002), pages 321–357, 2002.

[12] Jun Chen and Meng Li. Chained predictions of flight delay using machine learning. AIAA Scitech 2019 Forum, 2019.

[13] Sun Choi, Young Jin Kim, Simon Briceno, and Dimitri Mavris. Prediction of weather-induced airline delays based on machine learning algorithms. In AIAA/IEEE Digital Avionics Systems Conference - Proceedings, volume 2016-December, 2016.

[14] Sun Choi, Young Jin Kim, Simon Briceno, and Dimitri Mavris. Cost-sensitive prediction of airline delays using machine learning. AIAA/IEEE Digital Avionics Systems Conference - Proceedings, 2017.

[15] Kiranmoy Das, Martin Krzywinski, and Naomi Altman. Quantile regression. Nature Methods, 2019.

[16] Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester. A Modern Introduction to Probability and Statistics. 2005.

[17] Vinayak Deshpande and Mazhar Arikan. The impact of airline flight schedules on flight delays. Manufacturing and Service Operations Management, 2012.

[18] Yi Ding. Predicting flight delay based on multiple linear regression. IOP Conference Series: Earth and Environmental Science, 2017.

[19] Ulrich Dorndorf, Florian Jaehn, and Erwin Pesch. Flight gate assignment and recovery strategies with stochastic arrival and departure times. OR Spectrum, 2017.

[20] B Efron. Bootstap Methods: Another look at the Jackknife. The Annals of Statistics, 1979.

[21] Eurocontrol. European Aviation in 2040. pages 1–91, 2018. URL https://www.eurocontrol.int/publication/long-term-forecast-annual-numbers-ifr-flights-2040.

[22] Eurocontrol. Performance Review Report (PRR 2017). (May), 2018. URL https://www.eurocontrol.int/documents/performance-review-report-prr-2012.

[23] Eurocontrol. Coda Digest 2019, All-Causes Delay and Cancellations to Air Transport in Europe. page 34, 2019. URL https://www.eurocontrol.int/archive_download/all/node/11079.

[24] Eurocontrol. Performance Review Report (PRR 2019). 2020. URL https://www.eurocontrol.int/publication/performance-review-report-prr-2019.

[25] FAA. Flight Delay definition, 2009. URL https://aspmhelp.faa.gov/index.php/Flight_Delay.

[26] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. 33rd International Conference on Machine Learning, ICML 2016, pages 1651–1660, 2016.

[27] Guan Gui, Fan Liu, Jinlong Sun, Jie Yang, Ziqi Zhou, and Dongxu Zhao. Flight delay prediction based on aviation big data and machine learning. IEEE Transactions on Vehicular Technology, 2020.

[28] James Hanley and Barbara McNeil. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. Radiology, pages 29–36, 1982.

[29] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2009.

[30] Simon Haykin. Neural Networks and Learning Machines. 2009.

[31] Yuji Horiguchi, Yukino Baba, Hisashi Kashima, M. Suzuki, H. Kayahara, and J. Maeno. Predicting Fuel Consumption and Flight Delays for Low-Cost Airlines. Innovative Applications of Artificial Intelligence (IAAI) Conference, pages 4686–4693, 2017.

[32] Joint Economic Comittee. Your Flight Has Been Delayed Again: Flight Delay cost passengers, airlines, and the U.S. Economy Billions. Joint Economic Comittee, pages 1–20, 2008.

[33] Sina Khanmohammadi, Chun-an Chou, Harold W Lewis Iii, and Doug Elias. A Systems Approach for Scheduling Aircraft Landings in JFK Airport. 2014.

[34] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F. Atiya. Lower upper bound estimation method for construction of neural network-based prediction intervals. IEEE Transactions on Neural Networks, 2011.

[35] Abbas Khosravi, Saeid Nahavandi, and Senior Member. Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances. (July 2014), 2011.

[36] Sotiris Kotsiantis, Dimitris Kanellopoulos, and Panayiotis Pintelas. Handling imbalanced datasets : A review. Science, 2006.

[37] Tatjana Krstić Simić and Obrad Babić. Airport traffic complexity and environment efficiency metrics for evaluation of ATM measures. Journal of Air Transport Management, pages 260–271, 2015.

[38] Miguel Lambelho, Mihaela Mitici, Simon Pickup, and Alan Marsden. Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions. Journal of Air Transport Management, 2020.

[39] Hanbong Lee and Waqar Malik. Taxi - Out Time Prediction for Departures at Charlotte Airport Using Machine Learning Techniques. 2016.

[40] Loo Hay Lee, Chul Ung Lee, and Yen Ping Tan. A multi-objective genetic algorithm for robust flight scheduling using simulation. European Journal of Operational Research, 2007.

[41] James Loy. How to build your own Neural Network from scratch in Python, 2018. URL https://towardsdatascience.com/how-to-build-your-own-neural-network-from-scratch-in-\protect\@normalcr\relaxpython-68998a08e4f6.

[42] Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems, 2017.

[43] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence, pages 1–38, 2019.

[44] Eric R Mueller, Gano B Chatterji, and Moffett Field. Analysis of Aircraft Arrival and Departure Delay Characteristics. AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical, 2002.

[45] David A. Nix and S. Weigend Andreas. Estimating the Mean and Variance of the Target Probability Distribution. IEEE International Conference on Neural Networks (ICNN'94), 1994.

[46] OurAirports. Open data downloads, 2020.

[47] J. V. Pérez–Rodríguez, J. M. Pérez–Sánchez, and E. Gómez–Déniz. Modelling the asymmetric probabilistic delay of aircraft arrival. Journal of Air Transport Management, pages 90–98, 2017.

[48] Joaquim F. Pinto da Costa, Hugo Alonso, and Jaime S. Cardoso. The unimodal model for the classification of ordinal data. Neural Networks, 2008.

[49] Moschoula Pternea and Ali Haghani. An aircraft-to-gate reassignment framework for dealing with schedule disruptions. Journal of Air Transport Management, 2019.

[50] Nikolas Pyrgiotis, Kerry M. Malone, and Amedeo Odoni. Modelling delay propagation within an airport network. Transportation Research Part C: Emerging Technologies, pages 60–75, 2013.

[51] Juan Jose Rebollo and Hamsa Balakrishnan. Characterization and prediction of air traffic delays. Transportation Research Part C, pages 231–241, 2014.

[52] Baptiste Rocca. Handling imbalanced datasets in machine learning, 2019. URL https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28.

[53] Royal Schiphol Group. Annual Report 2019. 2019.

[54] J. A. S. Sá, A. C. Almeida, B. R. P. Rocha, M. A. S. Mota, J. R. S. Souza, and L. M. Dentel. Lightning Forecast Using Data Mining Techniques On Hourly Evolution Of The Convective Available Potential Energy. 2016.

[55] Schiphol. Flight paths and runway use. URL https://www.schiphol.nl/en/schiphol-as-a-neighbour/page/flight-paths-and-runway-use/.

[56] Scikit-learn developers. Cross-validation: evaluating estimator performance, 2017. URL https://scikit-learn.org/stable/modules/cross_validation.html.

[57] Merve Seker and Nilay Noyan. Stochastic optimization models for the airport gate assignment problem. pages 438–459, 2012.

[58] Durga L. Shrestha and Dimitri P. Solomatine. Machine learning approaches for estimation of prediction interval for the model output. Neural Networks, 2006.

[59] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, pages 1929–1958, 2014.

[60] Alice Sternberg, Jorge Soares, Diego Carvalho, and Eduardo Ogasawara. A Review on Flight Delay Prediction. 2017.

[61] Gabor Takacs. Predicting flight arrival times with a multistage model, 2014.

[62] Yufeng Tu, Michael O Ball, Wolfgang S Jank, T U Yufeng, Michael O B All, and Wolfgang S J Ank. Estimating Flight Departure Delay Distributions — A Statistical Approach With Long-Term Trend and Short-Term Pattern E. 2012.

[63] Oscar R.P. van Schaijk and Hendrikus G. Visser. Robust flight-to-gate assignment using flight presence probabilities. Transportation Planning and Technology, 2017.

[64] Jinn Tsai Wong and Shy Chang Tsai. A survival model for flight delay propagation. Journal of Air Transport Management, pages 5–11, 2012.

[65] Wong J.T Wu, C.L. Delay Propagation Modeling and the Implications in Robust Airline Scheduling. 2007 ATRS World Conference, San Francisco., 2007.

[66] Shangyao Yan and Ching-hui Tang. A heuristic approach for airport gate assignments for stochastic flight delays. pages 547–567, 2007.

[67] Bin Yu, Zhen Guo, and Sean Sobhan Asian. Flight delay prediction for commercial air transport: A deep learning approach. Logistics, Part E, (March), 2019.