PREDICTING AIRCRAFT TIME TO FLY PROFILE ON FINAL APPROACH

Roy Haanen
October 2016
Predicting Aircraft Time to Fly Profile on Final Approach

MASTER OF SCIENCE THESIS

For obtaining the degree of Master of Science in Business Information Systems at the department of Mathematics & Computer Science of Eindhoven University of Technology.

Roy Haanen – 0460519
r.haanen@student.tue.nl

October 2016

Supervisors:

dr. M. Pechenizkiy
Eindhoven University of Technology

dr. ir. J. Vanschoren
Eindhoven University of Technology

dr. R.J. de Almeida e Santos Nogueira
Eindhoven University of Technology

ir. G.B. van Baren
Netherlands Aerospace Centre

ir. R. Rijken
Royal Netherlands Air Force
Abstract

In order to prevent collisions and wake turbulence encounters between aircraft during departure and landing, separation criteria (based on 4 aircraft weight categories) have been defined over 40 years ago by the International Civil Aviation Organization. These criteria lead to over-separations in many instances, degrading an airport’s runway throughput. Eurocontrol has developed advanced wake metrics (RECAT-EU) to set up the European six category wake turbulence separation minima, with a goal to safely support an increase in runway throughput at airports in Europe.

By making improved predictions on an aircraft’s time to fly (T2F) profile during its final approach of a flight, separation between two succeeding aircraft can be managed more accurately by Air Traffic Controllers (ATCo). This helps to use the runway throughput more effectively, reduce delays, while maintaining the required safety level. Optimizing runway throughput will become a necessity in near future since the number of congested airports (operating at 80% or more of their capacity for more than three hours a day) in Europe is expected to grow from 6 (in 2012) to 30 in 2035.

The goal of this project is to make accurate predictions on the time to fly and speed profile of an aircraft for the last 4 Nautical Miles in order for an ATCo to optimize the separation buffer between two succeeding aircraft. Currently a static (initial) model is used to predict the Final Approach Speed profile and time to fly; for each aircraft a reference speed profile is defined for the last 10 Nautical Miles. Prediction accuracy is measured in terms of a Root Mean Squared Error (RMSE) over all predictions.

To be able to assess the success of this research evaluation criteria have been defined, focused on improved performance and ensuring safety. A short analysis is conducted to identify major factors influencing Final Approach Speed (FAS) and their effect on FAS. After merging three distinct datasets containing meteorological data, flight information and actual landing weight of aircraft during final approach an experiment with a (non-optimized) RandomForestRegressor has been performed as benchmark test.

Time Series forecasting using ARIMA appeared not to be an appropriate method to accurately predict T2F and Ground Speed profiles; not only the overall RMSE is bigger compared to the initial model, also variance of predictions is much wider.

Due to good performance and low variance characteristics of ensemble methods, four ensemble algorithms have been selected for comparison together with Gaussian Processes. After optimization of their hyperparameters the BaggingRegressor algorithm of scikit-learn’s library delivered best results, outperforming the initial model significant.
A little over 1% of all flights are having an absolute prediction error of 10 seconds or higher, which could lead to separation infringements. A brief analysis on those affected flights showed the errors were high using any algorithm. It seemed that the fastest or slowest aircraft caused the highest prediction error.

The approach given in this thesis shows the potential for using Machine Learning techniques for predicting T2F profiles for an aircraft in final approach. The data used in this project is available within NLR, the Netherlands Aerospace Center, and can be used in real-time systems supporting an ATCo.
Acknowledgements

The thesis in front of you is the result of my internship within the Aerospace Operations Safety Institute (AOSI) of NLR. NLR is the Netherlands Aerospace Centre for identifying, developing and applying advanced technological knowledge in the area of Aerospace.

Without the effort and dedication of a few great persons this thesis would not exist. I would like express my gratitude to Gerben van Baren for supervising me at NLR during the entire internship. Whenever I needed some guidance I could count on his immediate expert knowledge and professional skills. I would also like to thank Mykola Pechenizkiy for his guidance and reminding me over and over that the business goal is very important. Joaquin Vanschoren, thank you for your time and effort guiding me in my research. Etienne Hunt, thank you for the good time we had together in Amsterdam, it would have been less fun without you.

I have been receiving tremendous support from my wife Inge and my two loving daughters Julia and Isabeau. Thank you Pap, Mam and Inge for supporting me since the moment I was born. Lastly, I want to thank my father and mother in-law, Adri and Henny, who have done so much for me these last years and have enabled me to work on my thesis whenever possible.

Eindhoven, Netherlands

Roy Haanen

October 2016
# Contents

List of Figures x

List of Tables xii

List of Abbreviations xiv

I The problem 1

1 Introduction 2

1.1 Motivation ........................................... 2
1.2 Problem formulation .................................. 3
1.3 Research approach & methodology .................... 4
1.4 Structure of the thesis ................................ 5

2 Description of the problem 7

2.1 Separation delivery .................................... 7
2.2 Predicting Speed/T2F profiles ......................... 11
2.3 Final Approach Speed .................................. 13
2.4 Relevant factors FAS .................................. 14
2.5 Measuring project success ............................ 15
2.6 Research questions review .............................. 19

3 Data Exploration 20

3.1 Data sources .......................................... 20
3.2 Gathering and preparing data .......................... 20
3.3 Analyzing influence relevant factors on FAS ........... 22
3.4 Research questions review .............................. 32

II Predicting Speed / Time to Fly Profile 33

4 Machine Learning methods for aircraft performance prediction 34

4.1 Aircraft performance prediction domain .................. 34
4.2 Traveling time prediction for buses ..................... 35
4.3 What prediction methods to use? ......................... 36
4.4 Research questions review .............................. 37
5 Experiments 38
  5.1 Initial Method error calculations .................................... 38
  5.2 Experiments using Random Forest Regressor ....................... 40
  5.3 Experiments using Time Series Forecasting .......................... 45
  5.4 Experiments using different algorithms ................................ 53
  5.5 Residual analysis ......................................................... 61
  5.6 Research questions review ............................................. 64

III Concluding remarks 66

6 Conclusions 67

7 Recommendations and deployment considerations 70
  7.1 Recommendations .......................................................... 70
  7.2 Considerations for deployment ......................................... 70

IV Appendices 72

A CRISP-DM Methodology 73
B Errors in predictive modeling 75
C Time Series Forecasting 77
D Hyperparameter combinations 80
E Code 82
  E.1 Python code preparing data ............................................ 82
  E.2 Python code predict T2F/Ground Speed ............................ 84
  E.3 R code Time Series forecasts ......................................... 91
  E.4 Python code optimizing hyperparameters ........................... 92

Bibliography 92
# List of Figures

1.1 Wake Vortex Turbulence Illustration [Skybrary, 2016] ........................................... 2
1.2 Currently used speed profiles for five aircraft types [van Baren et al., 2016] ........ 3
1.3 Conceptual model for improving efficiency of runway ........................................... 5

2.1 Compression effect aircraft separation [van Baren et al., 2016] ......................... 9
2.2 Actual spacing between A320 and Dash 8 aircraft [van Baren et al., 2016] .......... 10
2.3 Actual spacing distribution on threshold: current vs. desired [van Baren et al., 2016] ................................................................. 10
2.4 TMA approach segments and speed control [van Baren et al., 2015]. The picture depicts two distinct arrival routes being merged into one single flow before the Final Approach Point ...................................................... 12

3.1 Distribution of ground speed (in kts) for B738 and A332 .................................. 23
3.2 A330 $V_{app}$ calculation table, based on the Quick Reference Handbook (QRH) ................................................................. 23
3.3 A plot of theoretical $V_{app}$ versus the actual (average) IAS for the A332. The red asterisks indicate the expected clusters of data. ...................................................... 24
3.4 A plot of theoretical FAS versus the actual (average) IAS for the B738. The red asterisks indicate the expected clusters of data. ...................................................... 25
3.5 Box and Whisker plot for the average corrected ground speed for the A332 (left) and B738 (right) on Schiphol Airport’s runways. The red dotted lines indicate the mean corrected average ground speed for a runway. The green dotted line indicate the overall mean for all runways ...................................................... 27
3.6 Actual landing weight vs. ground speed A332 ..................................................... 29
3.7 Actual landing weight vs ground speed B738 ..................................................... 30
3.8 Headwind vs. (Weight-corrected) Ground Speed A332 ........................................ 31
3.9 Headwind vs. (Weight-corrected) Ground Speed B738 ........................................ 31

5.1 Initial Model Ground Speed Profiles B738/A332 ............................................... 39
5.2 Bias / Variance vs Number of Trees in RandomForestRegressor .......................... 41
5.3 Time Series decomposition ................................................................................. 46
5.4 Box-plot Time to Fly B738 .................................................................................. 47
5.5 Box-plot Ground Speed B738 .............................................................................. 48
5.6 Example flight Time Series forecast T2F .............................................................. 49
5.7 Example flight Time Series forecast Ground Speed .............................................. 50
5.8 Distribution of forecasted RMSEs (T2F 4NM-0NM) ........................................... 50
5.9 Distribution of flight errors T2F 4NM (B738) ...................................................... 60
5.10 RMSE’s all algorithms T2F 4NM (B738) .............................................................. 60
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.11</td>
<td>Error Distribution BaggingRegressor (B738 T2F 4NM)</td>
<td>61</td>
</tr>
<tr>
<td>5.12</td>
<td>Residual analysis RMSE ≥ 10s (B738 T2F 4NM)</td>
<td>63</td>
</tr>
<tr>
<td>5.13</td>
<td>B738 distribution of T2F over the last 4 NM</td>
<td>63</td>
</tr>
<tr>
<td>A.1</td>
<td>CRISP-DM Methodology [Chapman et al., 2000]</td>
<td>74</td>
</tr>
<tr>
<td>B.1</td>
<td>Bias-Variance trade-off</td>
<td>75</td>
</tr>
<tr>
<td>C.1</td>
<td>Stationary Time Series - Mean</td>
<td>77</td>
</tr>
<tr>
<td>C.2</td>
<td>Stationary Time Series - Variance</td>
<td>78</td>
</tr>
<tr>
<td>C.3</td>
<td>Stationary Time Series - Covariance</td>
<td>78</td>
</tr>
</tbody>
</table>
List of Tables

2.1 ICAO Wake Turbulence Separation Minima [SKYbrary, 2015] .............. 8
2.2 RECAT-EU Separation Minima [Rooseeleer and Treve, 2015] .......... 8

3.1 Dataset description .................................................. 21
3.2 $V_{REF}$ Calculation Table Boeing ................................... 25

5.1 Initial Model results T2F 4NM-0NM .................................. 39
5.2 Initial Model results Ground Speed 4NM-0NM ........................ 40
5.3 Initial Model results T2F 10NM-0NM .................................. 40
5.4 Initial Model results Ground Speed 10NM-0NM ......................... 40
5.5 Random Forest Regressor (default settings) results T2F 4NM-0NM ...... 44
5.6 Random Forest Regressor results T2F 10NM-0NM ....................... 45
5.7 Random Forest Regressor results Ground Speed 4NM-0NM ............ 45
5.8 Random Forest Regressor results Ground Speed 10NM-0NM .......... 46
5.9 Data inputs Time Forecasting ............................................. 47
5.10 Data targets Time Forecasting ............................................ 47
5.11 Time Series Forecasting results T2F 4NM-0NM ......................... 52
5.12 Time Series Forecasting results T2F 10NM-0NM ....................... 53
5.13 Time Series Forecasting results Ground Speed 4NM-0NM ............ 53
5.14 Time Series Forecasting results Ground Speed 10NM-0NM ............ 53
5.15 Hyperparameter grids for all algorithms ................................ 56
5.16 Results all algorithms T2F 4NM ....................................... 58
5.17 Results all algorithms T2F 10NM ....................................... 59
5.18 Results all algorithms Ground Speed 4NM ............................ 59
5.19 Results all algorithms Ground Speed 10NM ........................... 59
5.20 Absolute T2F 4NM flight errors (s) $\geq$ 13s ............................ 62
5.21 Summarized results all algorithms T2F 4NM .......................... 64
5.22 Summarized results all algorithms T2F 10NM ........................ 65
5.23 Summarized results all algorithms Ground Speed 4NM ............... 65
5.24 Summarized results all algorithms Ground Speed 10NM ............. 65

6.1 BaggingRegressor T2F 4NM Results .................................... 68

D.1 Top 5 hyperparameter combinations GaussianProcess .................. 80
D.2 Top 5 hyperparameter combinations AdaBoostRegressor ............... 80
D.3 Top 5 hyperparameter combinations BaggingRegressor .................. 81
D.4 Top 5 hyperparameter combinations GradientBoostingRegressor ......... 81
D.5 Top 5 hyperparameter combinations RandomForestRegressor .......................... 81
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A332</td>
<td>Airbus A330-200</td>
</tr>
<tr>
<td>ADS-B</td>
<td>Automatic Dependent Surveillance-Broadcast</td>
</tr>
<tr>
<td>ALAW</td>
<td>(Actual) Landing Weight</td>
</tr>
<tr>
<td>APP</td>
<td>Approach Controller</td>
</tr>
<tr>
<td>ARIMA</td>
<td>Auto-Regressive Integrated Moving Average</td>
</tr>
<tr>
<td>ATCo</td>
<td>Air Traffic Controller</td>
</tr>
<tr>
<td>B738</td>
<td>Boeing 737-800</td>
</tr>
<tr>
<td>DBS</td>
<td>Distance Based Separation</td>
</tr>
<tr>
<td>FAP</td>
<td>Final Approach Point</td>
</tr>
<tr>
<td>FAS</td>
<td>Final Approach Speed</td>
</tr>
<tr>
<td>FIN</td>
<td>Final Controller</td>
</tr>
<tr>
<td>GS</td>
<td>Ground Speed</td>
</tr>
<tr>
<td>IAF</td>
<td>Initial Approach Fix</td>
</tr>
<tr>
<td>IAS</td>
<td>Indicated Air Speed</td>
</tr>
<tr>
<td>ICAO</td>
<td>International Civil Aviation Organization</td>
</tr>
<tr>
<td>INI</td>
<td>Initial Controller</td>
</tr>
<tr>
<td>INT</td>
<td>Intermediate Controller</td>
</tr>
<tr>
<td>KLM</td>
<td>Royal Dutch Airlines (Koninklijke Luchtvaart Maatschappij)</td>
</tr>
<tr>
<td>kts</td>
<td>Knots</td>
</tr>
<tr>
<td>METAR</td>
<td>Format for reporting weather information</td>
</tr>
<tr>
<td>MRS</td>
<td>Minimum Radar Separation</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Squared Error NLR</td>
</tr>
<tr>
<td>NM</td>
<td>Nautical Mile(s)</td>
</tr>
<tr>
<td>RECAT-EU</td>
<td>European Wake Vortex Re-categorization</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Squared Error</td>
</tr>
<tr>
<td>SESAR</td>
<td>Single European Sky ATM Research</td>
</tr>
<tr>
<td>STAR</td>
<td>Standard Arrival Route</td>
</tr>
<tr>
<td>T2F</td>
<td>Time to Fly</td>
</tr>
<tr>
<td>TAS</td>
<td>True Air Speed</td>
</tr>
<tr>
<td>TBS</td>
<td>Time Based Separation</td>
</tr>
<tr>
<td>TMA</td>
<td>Terminal Maneuvering Area</td>
</tr>
<tr>
<td>$v_{app}$</td>
<td>Approach Speed</td>
</tr>
<tr>
<td>$v_{LS}$</td>
<td>Lowest Selectable Speed</td>
</tr>
<tr>
<td>$v_{REF}$</td>
<td>Reference Landing Speed</td>
</tr>
</tbody>
</table>
Part I

The problem
Introduction

In this thesis a study is performed on the performance of aircraft during final approach. Performance in this context is about the speed profile of the aircraft. By making improved predictions on each aircraft’s speed profile the separation between two succeeding aircraft can be managed more accurately by air traffic controllers. This helps to use the runway capacity more effectively, reduce delays, while maintaining the required safety level.

In Section 1.1 the motivation for this study is provided in more detail. Section 1.2 formulates the problem, which forms the basis for the study and is used throughout this thesis. How the problem is approached is described in 1.3.

1.1 Motivation

Wake Vortex Turbulence\(^1\) (see Figure 1.1 for an illustration) is defined as turbulence which is generated by the passage of an aircraft in flight. It will be generated from the point when the nose landing gear of an aircraft leaves the ground on take-off and will cease to be generated when the nose landing gear touches the ground during landing. Where another aircraft encounters such turbulence, a Wake Vortex Encounter is said to have occurred.

\(^1\)http://www.eurocontrol.int/articles/wake-vortex

Figure 1.1: Wake Vortex Turbulence Illustration [SKYbrary, 2016]
For safe and efficient handling of aircraft approaching a runway for landing, air traffic controllers (ATCo) apply a certain separation distance between two succeeding aircraft. This separation distance should satisfy certain internationally agreed upon minima to protect the aircraft from collision and wake vortex encounter risk.

The actual distance spacing between two aircraft on final approach is however not constant but varies because aircraft decelerate when approaching the runway to their specific landing speed. This landing speed depends (among others) on the aircraft type, weight and wind conditions. Because an ATCo cannot accurately predict the variation in speed profile, the separation distances are applied with a certain buffer. This results in a significant spread in the resulting spacing when the aircraft passes the runway threshold.

In order to enhance runway throughput, there is a need to provide the spacing between aircraft more accurately. For this, it is necessary to improve the aircraft speed profile predictions on intermediate and final approach. Final Approach Speed (FAS, also known as target threshold speed) is the speed (in knots) at which an aircraft travels while it is on the final straight to land at a runway.

1.2 Problem formulation

The research objective of this thesis research is to deliver an improved prediction of an aircraft’s time to fly (T2F) and speed profile on its final approach. Examples of current speed profiles for several airplanes are depicted in Figure 1.2 [van Baren et al., 2015]. Note that the left plot does not consider measures like weather conditions, actual weight of the plane and other important factors that have an effect on the ground speed and, hence, time to fly. The right plot shows the indicated air speed profiles based on actual data in the research of the authors.

![Figure 1.2: Currently used speed profiles for five aircraft types [van Baren et al., 2016]](image-url)
The problem being investigated takes a flight $f$ as input and produces a Ground Speed (GS) or Time to Fly (T2F) profile $p$ as output, taking into account the parameters that (theoretically) have an effect on the speed profile. The flight $f$ originates from either historic radar data or possibly a live datafeed. The output is a profile containing predicted GS/T2F at an interval of 0.5 Nautical Miles (NM) over a range of 0-4NM and 0-10NM from runway threshold.

The main research question is: *given an aircraft about to enter its final approach, what is its predicted speed and time to fly profile from 4 Nautical Miles to runway threshold?*

Based on the problem definition stated above the following (sub) research questions are derived:

1. What are appropriate performance criteria to assess possible solutions in order to secure that risks of inaccurate predictions are acceptable?
2. What are the major factors influencing the speed profile of an aircraft during its final approach?
3. What is the influence of the identified major factors on the speed profile of an aircraft during its final approach?
4. What Machine Learning (ML) techniques can be used to make predictions on the speed and time to fly profile of an aircraft during final approach?
5. How does each of the used ML techniques perform with respect to the defined performance criteria?

### 1.3 Research approach & methodology

The business goal of the research in this thesis is improving efficiency of runway throughput and at the same time securing aviation safety. To accomplish this, improved T2F/Ground Speed predictions on final approach are needed in order for an Air Traffic Controller to optimize the separation buffer between two succeeding aircraft. The approach taken to get improved T2F/GS predictions is shown in Figure 1.3.

In order to be able to make justified statements whether a model is efficient and still safe, appropriate evaluation criteria need to be drafted (answering research question 1) based on identified risks when predictions are inaccurate.

Indexing all relevant factors on Final Approach Speed and their influence on the speed profile is performed next (answering research question 2). This step is necessary in order to have a dataset that resembles the ‘real world’ as good as possible and form a strong foundation to use as input for the relevant Machine Learning techniques. The relevant factors are partly based on previous conducted research in this field [Herrema et al., 2015] and ongoing research within NLR [Hunt, 2016]. It is assumed that the identified major factors influencing aircraft performance also are of major importance in the prediction of the speed and time to fly profiles during final approach. Whether this is the case will be investigated in the data analysis part later on and should provide an answer to research question 3.

---

1. ALAW = Actual Landing Weight of aircraft

---
A study on the existing approaches for predicting times (and speeds) to arrival on fixed positions is conducted. The problem of travel time prediction using ML techniques has been investigated intensively and has many similarities with this thesis’ challenge. The outcome of this part will provide insights for research question 4.

A dataset is assembled containing the features found to be relevant for experiments. Applicable Machine Learning techniques for the problem, as found in the step above, will be used with the generated dataset to make validated statements on accuracy for the predicted speed and time to fly profiles (research question 5). After optimizing hyper-parameters of these techniques (if applicable) and comparing their performance the most optimal model for the problem is chosen. The results are evaluated against the defined performance criteria.

Figure 1.3: Conceptual model for improving efficiency of runway

As a guidance framework, the popular CRISP-DM (CRoss-Industry Standard Process for Data Mining) methodology is used [Chapman et al., 2000]. Appendix A describes briefly what CRISP-DM is and how it is used during this research.

1.4 Structure of the thesis

The remainder of this thesis is set up as follows:

- Part I: the problem
  - Chapter 2 describes the business problem in more detail. It defines the scope of the
research and indicates the current state of work in this research field. Also, chapter 2 addresses the need for speed prediction and will provide answers for research questions 1 and 2. This chapter comprises the business understanding phase of the CRISP-DM model.

- In chapter 3, available data is analyzed in order to measure the influence of the major factors influencing Final Approach Speed (providing an answer for research question 3). With regards to CRISP-DM, data understanding and data preparation phases are covered in this chapter.

• Part II: Predicting Speed / Time to Fly Profile

- Related work in the domain of aircraft performance prediction and a related domain is discussed in chapter 4. Appropriate Machine Learning methods for this research problem are identified, answering research question 4.

- In Chapter 5 the modeling and evaluation phases of CRISP-DM are addressed. It is dedicated to conducting experiments with several methods in order to identify performance of each method, providing an answer for research question 5. Performance is evaluated against the evaluation criteria, specified in Section 2.5.

• Part III: Concluding remarks

- The last chapters take into account considerations for deploying a model and formulates conclusions / future work for this research. The last block in the CRISP-DM model, deployment, is elaborated on here.

If applicable, a chapter closes with a section reviewing the answered research question(s) in that particular chapter.
Description of the problem

2.1 Separation delivery

Airports are classified as congested when they are operating at 80% or more of their capacity for more than three hours a day. The number of congested airports in Europe is expected to grow from 6 (in 2012) to 30 in 2035. Even more airports are experiencing runway capacity as a source of delay or a limitation to business development [Eurocontrol, 2013a].

An important factor limiting the runway throughput are the required separation minima imposed by the ICAO\(^1\) in order to prevent collisions and wake turbulence encounters between aircraft during departure and landing.

2.1.1 Currently used separation schemes

The minimum separation between two aircraft is prescribed by ICAO Procedures for Air Navigation Services — Air Traffic Management, and includes wake turbulence separation criteria. The criteria divide the aircraft into four different wake turbulence weight categories (Heavy, Medium, Light; a fourth sub-category “Super Heavy” was introduced on the introduction of the Airbus A-380). The use of weight categories means in principle that the worst case (i.e. largest weight difference between the vortex generating aircraft and the following encountering aircraft) is safe. For any other combination of aircraft, there may be an over-protection, and hence a potential loss of runway throughput. Table 2.1 displays the ICAO separation minima in Nautical Miles for a combination of aircraft leader/follower.

ICAO’s existing wake vortex separation rules were implemented over 40 years ago. In some respects, they are now outdated and lead to over-separations in many instances. Eurocontrol [Rooseleer and Treve, 2015] has developed advanced wake metrics to set up the European six category wake turbulence separation minima, RECAT-EU. RECAT-EU forms an alternative to the long established ICAO PANS-ATM categories, with a goal to safely support an increase

\(^1\)The International Civil Aviation Organization (ICAO) is a UN specialized agency, established by States in 1944 to manage the administration and governance of the Convention on International Civil Aviation (Chicago Convention).
**Table 2.1:** ICAO Wake Turbulence Separation Minima [SKYbrary, 2015]

<table>
<thead>
<tr>
<th>Preceding aircraft</th>
<th>Following aircraft</th>
<th>Minimum separation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Super</td>
<td>Super</td>
<td>4 NM</td>
</tr>
<tr>
<td></td>
<td>Heavy</td>
<td>6 NM</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>7 NM</td>
</tr>
<tr>
<td></td>
<td>Small</td>
<td>8 NM</td>
</tr>
<tr>
<td>Heavy</td>
<td>Heavy</td>
<td>4 NM</td>
</tr>
<tr>
<td></td>
<td>Large</td>
<td>5 NM</td>
</tr>
<tr>
<td></td>
<td>Small</td>
<td>6 NM</td>
</tr>
<tr>
<td>Large</td>
<td>Small</td>
<td>4 NM</td>
</tr>
</tbody>
</table>

in runway throughput at airports in Europe. In table 2.2 the advanced separation minima are depicted for departure and landing of a leader/follower combination. MRS means Minimum Radar Separation, set at 3 NM in current ICAO provisions².

**Table 2.2:** RECAT-EU Separation Minima [Rooseleer and Treve, 2015]

<table>
<thead>
<tr>
<th>RECAT-EU Scheme</th>
<th>&quot;Super Heavy&quot;</th>
<th>&quot;Upper Heavy&quot;</th>
<th>&quot;Lower Heavy&quot;</th>
<th>&quot;Upper Medium&quot;</th>
<th>&quot;Lower Medium&quot;</th>
<th>&quot;Light&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leader / Follower</td>
<td>&quot;A&quot;</td>
<td>&quot;B&quot;</td>
<td>&quot;C&quot;</td>
<td>&quot;D&quot;</td>
<td>&quot;E&quot;</td>
<td>&quot;F&quot;</td>
</tr>
<tr>
<td>&quot;Super Heavy&quot;</td>
<td>&quot;A&quot;</td>
<td>3 NM</td>
<td>4 NM</td>
<td>5 NM</td>
<td>5 NM</td>
<td>6 NM</td>
</tr>
<tr>
<td>&quot;Upper Heavy&quot;</td>
<td>&quot;B&quot;</td>
<td>3 NM</td>
<td>4 NM</td>
<td>4 NM</td>
<td>5 NM</td>
<td>7 NM</td>
</tr>
<tr>
<td>&quot;Lower Heavy&quot;</td>
<td>&quot;C&quot;</td>
<td>MRS</td>
<td>3 NM</td>
<td>3 NM</td>
<td>4 NM</td>
<td>6 NM</td>
</tr>
<tr>
<td>&quot;Upper Medium&quot;</td>
<td>&quot;D&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5 NM</td>
</tr>
<tr>
<td>&quot;Lower Medium&quot;</td>
<td>&quot;E&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4 NM</td>
</tr>
<tr>
<td>&quot;Light&quot;</td>
<td>&quot;F&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 NM</td>
</tr>
</tbody>
</table>

According to [Rooseleer and Treve, 2015] using RECAT-EU runway throughput benefits can reach 5% or more during peak periods depending on individual airport traffic mix.

In the current practise, the required separation minima are usually managed by the Air Traffic

²Between succeeding aircraft which are established on the same final approach track within 10 NM of the runway end, a reduced separation minimum of 2.5 NM may be applied, provided certain requirements are satisfied, including requirements on runway occupancy time, braking action and radar accuracy.
Controller (ATCo) without using any supporting tools. The ATCo needs to set up separation based on information on prevailing wind conditions, aircraft Wake Turbulence Category and experience.

### 2.1.2 Compression effect

Figure 2.1 displays the effect of compression during final approach. At the threshold (THR, or start of the runway) the separation minima (Final Target Distance) needs to be satisfied. In order to land safely the leader aircraft will decrease its speed at the last 4 NM, which results in a compression of the spacing between the leader and its follower aircraft. To compensate for this compression, a spacing buffer and an initial target distance is required.

To optimize runway throughput and while ensuring safety, the goal is to minimize the spacing buffer at the runway threshold.

![Figure 2.1: Compression effect aircraft separation](van Baren et al., 2016)

Currently, the required spacing buffer is determined based on the experience of the Air Traffic Controller.

### 2.1.3 Actual spacing

Research [van Baren et al., 2016] has been conducted on multiple airports to gather insights in actual distance spacing between two aircraft (Airbus A320 as leader and a Dash 8 as follower are shown below as an example) on a trajectory from 10 NM to threshold. Their results (see Figure 2.2) indicate that distance spacing margin can be up to 1 NM at the threshold. Also differences among airports are clear from this picture, which can be explained by having differences in headwind distribution and separation practices.
The left histogram in Figure 2.3 shows the actual spacing distribution between two random aircraft types on runway threshold. The 0 (red dashed) line represents the optimal spacing. As can be seen most of the aircraft pairs have a spacing higher than the optimal spacing and even some have a lower spacing (negative margin with regards to the separation criteria). Lower spacing does not necessarily mean that there was an unsafe situation; the controller or flight crew can apply visual separation in such cases. The right histogram displays the desired situation, an optimized spacing distribution allowing for better runway throughput and taking into account the separation criteria, but also other factors like weather and aircraft weight.

**Figure 2.2:** Actual spacing between A320 and Dash 8 aircraft [van Baren et al., 2016]

**Figure 2.3:** Actual spacing distribution on threshold: current vs. desired [van Baren et al., 2016]
2.2 Predicting Speed/T2F profiles

2.2.1 Aircraft speed

Several speed measures are used for determining the speed of an aircraft. In this thesis Indicated Air Speed (IAS), True Air Speed (TAS) and Ground Speed (GS) are used. They are described as follows:

- **Indicated Air Speed (IAS)**. IAS is the speed of an aircraft as shown on its pitot static airspeed indicator, calibrated to reflect standard atmosphere adiabatic compressible flow at sea level, uncorrected for airspeed system errors. The IAS speed is displayed in the manuals of the manufacturer.

- **True Air Speed (TAS)**. TAS is the airspeed of an aircraft relative to the undisturbed air. The relation between the TAS and IAS can be described by equation 2.1:

\[
TAS = IAS \sqrt{\frac{\rho_0}{\rho_h}}
\]  
(2.1)

The TAS depends upon air density at sea level (\(\rho_0\)) and the air density at altitude (\(\rho_h\)). Due to the limited altitude of aircraft during the last 2.5NM as used in this sensitivity analysis, the assumption is made that TAS \(\approx\) IAS.

- **Ground Speed**. Ground speed is the combination of the true airspeed vector of the aircraft and the speed vector of wind at aircraft altitude. It is basically the speed of an aircraft relative to the ground, hence ground speed. A headwind component should be subtracted (in case of tailwind, added) to the TAS in order to calculate the ground speed, according to the relation presented in equation 2.2:

\[
GS = TAS - \text{Headwind} \Rightarrow GS \approx IAS - \text{Headwind}
\]  
(2.2)

For all wind related graphs in the subsequent sections, ground speed have been considered instead of IAS.

- **Time to fly (T2F)**. Based on the ground speed in kts (Nautical Miles/Hour), T2F (in hours) can be calculated by dividing the distance to traverse (in NM) by the ground speed.

2.2.2 Speed control

Before reaching a destination airport and entering a Terminal Maneuvering Area (TMA), an aircraft follows a Standard Terminal Arrival Route (STAR). A STAR is a predefined route between the last point of the route filled in the flight plan and the first point of the approach to the airport. Flights coming in from different directions can use different STARs, which are merged into a single flow (see Figure 2.4).

---

3 The pitot static airspeed indicator is part of the pitot-static system. This is a system of pressure-sensitive instruments that is most often used in aviation to determine an aircraft’s airspeed.

4 As the average altitude of aircraft situated 2.5NM of the runway threshold is relatively low (850 feet above sea level in Amsterdam), the error in speed calculation incorporated by applying the assumption is about 1.3%
Figure 2.4: TMA approach segments and speed control [van Baren et al., 2015]. The picture depicts two distinct arrival routes being merged into one single flow before the Final Approach Point.

The figure also shows approach segments indicating the responsible controller for that segment, as well as the Initial Approach Fix (IAF, the point where the initial approach segment of an instrument approach begins) and FAP (Final Approach Point, the point in space where the final approach segment begins on the instrument approach).

The Air Traffic Controllers instruct the aircraft to fly a certain Indicated Air Speed (IAS) at a certain segment of approach. This speed is defined and described in the Aeronautical Information Publication at many airports. Typically speed starts at 210/220 kts on downwind and is reduced to 160 kts until FAP, where the aircraft adopts its Final Approach Speed (FAS). Final Approach Points may deviate per airport. For that reason, the length of the segments where speed is controlled may vary.

Whether speed control is actually applied depends on the traffic density of that airport and other conditions. When applied, there can still be considerable variation in the actual distribution of speed, both IAS and ground speed. Variation in IAS depends on the time when the speed instruction is given by the controller, as well as the time it takes for a Flight Management System to reach the instructed speed. Ground speed is calculated by using the given IAS corrected by the headwind along glide slope.

In the research performed by [van Baren et al., 2015], there is a considerable variation in Final Approach Speed between different aircraft types. Even for a particular aircraft type, variation in FAS may be around 20 kts, due to e.g. difference in actual landing weight, flap settings and weather conditions.

2.2.3 Need for speed prediction

Ground speed profiles of two succeeding aircraft determines how the spacing develops on final approach. The approach controller will set up the initial spacing based on experience, which also includes a certain buffer to account for the compression of the distance spacing.

Due to the variation in Final Approach Speed, as described in the previous section, predicting
the speed profile of an aircraft allows for a more accurate assessment of the required spacing buffer to improve runway throughput while maintaining air safety. By using the time it takes to travel from one fixed point to the next, a time to fly profile can be generated.

2.3 Final Approach Speed

In order to assure optimal balance between aircraft handling qualities (stall margin or controllability/manoeuvrability) and landing distance, aircraft need to have a ‘balanced’ speed which is referred to as the Final Approach Speed (FAS). Boeing (Section 2.3.1) and Airbus (Section 2.3.2), the world’s largest aircraft manufacturers, have different interpretations on how to calculate the Final Approach Speed. How Final Approach Speed is calculated is important since this determines the time to fly profile of a specific aircraft with its environment conditions during the last 4 Nautical Miles.

The FAS is the airspeed starting after the FAP at about 4NM (see Figure 2.4) which should be maintained down to 50 feet over the runway threshold [Flight-Safety-Foundation, 2000]. The FAS finds its origin in the reference landing speed \( V_{REF} \), for Boeing aircraft and Lowest Selectable Speed \( V_{LS} \), for Airbus Aircraft. Both the \( V_{REF} \) (Equation 2.3) and the \( V_{LS} \) (Equation 2.4) are only depending on the stall speed (which is dependent on the Actual Landing Weight\(^5\)) and the configuration (flap setting). Air speed may never drop below these reference values as these constitute an acute stall danger.

\[
V_{REF} = 1.3 \cdot \text{stall speed with selected landing flaps} \tag{2.3}
\]

\[
V_{LS} = 1.23 \cdot \text{stall speed with selected landing flaps} \tag{2.4}
\]

2.3.1 FAS Boeing

For Boeing aircraft, the FAS calculation (Equation 2.5) depends upon the \( V_{REF} \) calculation and the required additional corrections. Under normal conditions (e.g. no failures) the value of the ‘corrections’ term is limited to \( \geq 5 \text{kts} \) and \( \leq 20 \text{kts} \), and mainly depends upon wind conditions (Equation 2.6).

\[
FAS_{Boeing} = V_{REF} + \text{Correction} \tag{2.5}
\]

\[
\text{Correction} = \text{Max}[\left(\frac{1}{2} \cdot (\text{steady headwind component}) + \text{gust value}\right), 5] \tag{2.6}
\]

2.3.2 FAS Airbus

For Airbus aircraft FAS calculation is conducted by means of the so called Ground Speed mini (GS Mini) system. This system is applied to all Airbus fly-by-wire aircraft and automatically assures

\(^5\)The Actual Landing Weight (ALW) is the Actual Zero Fuel Weight of the aircraft plus the reserve fuel that is required.
a minimum ground speed\(^6\) to guarantee a sufficient stall margin. It is due to this system that speed calculations by means of the tables in the AOM are redundant under normal conditions (e.g. no failures).

**\( V_{app} \) Calculation**

The approach speed \( V_{app} \) (Equation 2.7 and Equation 2.8), which is comparable with the \( FAS_{Boeing} \) is a function of the gross weight, configuration and Tower headwind component, and functions as an input for the GS Mini system.

\[
FAS_{Airbus} = V_{app} = V_{LS} + Correction \tag{2.7}
\]

\[
Correction = \text{Max}\left[ \left( \frac{1}{3} \cdot \text{headwind component} \right), 5 \right] \tag{2.8}
\]

\(^*\)The value of the 'Tower headwind component' is limited to \( >0 \text{kts} \) and \( \leq 15 \text{kts} \)

**GS Mini**

In order to assure a sufficient energy level, the aircraft ground speed should never drop below a certain value in the approach, while the winds are changing. It is therefore that the aircraft IAS must vary while descending, in order to cope with gusts/wind changes. The IAS target speed that should be flown is computed by the FMGS (Flight Management and Guidance Computer).

### 2.4 Relevant factors FAS

In order to make predictions on the speed profile during final approach it is necessary to know what factors have an influence on the Final Approach Speed on an aircraft. This section lists the influencing factors, which are a result of internal research [Hunt, 2016] conducted at NLR.

The relevant factors are split into major and minor parameters, based on their expected impact.

**Major factors**

- The Final Approach Speed is primarily based on the reference landing speed (\( V_{REF} \), for Boeing aircraft) and lowest selectable speed (\( V_{LS} \), for Airbus aircraft). Both \( V_{REF} \) and \( V_{LS} \) are dependent on the stall speed of the plane, which is determined by the actual landing weight and the chosen flap settings by the pilot.

- **Aircraft type and manufacturer.** The aircraft type and manufacturer are presumably affecting the FAS, which can be due to the fact that the margins and the methods of calculating the FAS differs per manufacturer and (to a lesser extent) aircraft type.

- **Assigned runway.** Interviews have shown that pilots are inclined to operate with full flaps when the runway length is limited, in order to minimize the required landing distance. It is therefore suspected that there is a correlation between the assigned runway and the FAS. Also the runway condition can be considered as an influencing factor on the FAS, as the pilot is more likely to land with full flaps and minimal FAS in contaminated (e.g. icy) runway conditions.

\(^6\)Ground speed is the horizontal speed of an aircraft relative to the ground, expressed in kts.
DESCRIPTION OF THE PROBLEM

- **Wind.** Due to wind conditions, an additional stall margin for airspeed excursion shall be applied. For Boeings, wind adds a maximum value of 20 knots. For Airbus, this factor is limited to 15 knots.

**Minor factors**

- **A/Throttle speed mode (Boeing aircraft only).** If autothrottle (A/Throttle) speed mode is applied, the airspeed correction factor in addition to $V_{REF}$ is five knots. No additional wind corrections are needed in that case, as the autothrottle automatically corrects for fluctuations caused by the wind.

- **Stability characteristics.** It has been examined from the interviews that there is a significant difference per pilot whether a relatively high FAS with small flap configuration or a low FAS with high flap configuration is favored.

- **Gusty weather.** A smaller flap setting during gusty weather is usually favored by pilots, as it increases the controllability of the aircraft.

- **Assigned Gate.** If the gate is near the runway threshold, the pilot is usually inclined to land with full flaps, minimal landing speed and make advantage of the associated smaller stop distance.

- **Low Visibility.** Lower approach speeds are usually associated with low visibility, as the pilot prefers more time for visual object identification.

- **Fuel Policy.** Differences in fuel efficiency policy per airliner can induce a difference landing flap setting and thus FAS, as the fuel efficiency of the aircraft decreases with flap deflection.

2.5 **Measuring project success**

Within Eurocontrol, a few experiments have started with regards to predicting Time to Fly (T2F) and Ground Speed (GS) profiles on final approach. There are no actual implementations being used in ‘live’ air traffic control systems, making it still an experimental area of research. In accordance with NLR, performance metrics described in sections 2.5.3 and 2.5.4 will be used to assess the success of this research. The metrics make a distinction between predicting T2F and ground speed profiles. In order to select proper performance metrics first the risks involved need to be known. Then, in Section 2.5.2 an explanation is provided why these specific metrics have been selected.

2.5.1 **Levels of risks**

During final approach of aircraft three possible categories of risk can be distinguished:

- Mid-air collision risk
- Runway collision risk
- Wake turbulence encounter risk
For each category the safety impact of making inaccurate predictions is described below.

**Mid-air collision risk**

As far as known, a collision between two aircraft flying under IFR (Instrument Flight Rules\(^7\)), approaching the same runway and under control of a radar controller has never occurred. The probability of such a collision hence is very low, although not necessarily negligible.

The main mechanism to prevent a mid-air collision during final approach is the maintaining of minimum radar separation by air traffic controllers (see Section 2.1.1). Maintaining minimum radar separation will not be affected by implementing possible solutions resulting from this research; there will always be the minimum buffer between two succeeding aircraft in order to guarantee safety.

Should, for any reason, minimum radar separation be infringed and an unsafe situation arise the air traffic controller can use a (normal operating) "go-around" procedure in which the aircraft is ordered to fly back to circuit heights and re-initiate final approach. Please note that this procedure has a negative effect on efficiency.

The risk on mid-air collision will be ignored in the rest of this thesis, under the condition that air traffic controllers will maintain Minimum Radar Separation at all times.

**Runway collision risk**

Runway collision risk involves collision of two aircraft landing consecutively on the same runway. Based on historical data, the probability of a runway collision risk is low, but not negligible. There are two main mechanisms to prevent a runway collision:

1. Air Traffic Controllers should not clear an aircraft to land before a preceding landing aircraft has crossed the runway threshold
2. Pilots should not land if the runway is occupied.

As with the mid-air collision risk, these mechanisms are kept in place in order to assure safety at all times.

**Wake turbulence encounter risk**

ICAO Wake Turbulence separation minima have been developed in order to prevent the risk on Wake turbulence. As long as separation distances between leader/follower combination are guaranteed, this risk is minimized. Accurate time to fly predictions for final approach allow an Air Traffic Controller (ATCo) to optimize spacing between two consecutive aircraft. However, if predictions are not accurate this spacing can vary; it can become larger if time to fly is underpredicted (which is no direct danger for wake encounter) or become smaller if overpredicted. In the case of overprediction the risk on wake encounter increases.

\(^7\)Rules and regulations established by the FAA to govern flight under conditions in which flight by outside visual reference is not safe.
Where the current wake separation minima are considered acceptably safe from a wake encounter risk point of view, an underseparation of 0.5 NM is usually used as a threshold for reporting it as a separation infringement incident. As such, an overprediction of time to fly that corresponds to 0.5 NM can be used as a performance criterion for this research.

2.5.2 Selecting proper performance metrics

This research is about increasing efficiency while maintaining safety. Efficiency-wise a better T2F prediction results in a more optimal runway throughput. However, optimizing the buffer between two succeeding aircraft during their landing also needs to be safe in order to prevent accidents from happening. As described in the previous section certain mechanisms are in place to prevent mid-air and runway collisions.

Predicting either T2F or ground speed for the final approach can deviate in two ways, both having different consequences. A too low prediction/T2F (it takes more time for the aircraft to land than was predicted) does not have direct consequences in terms of safety for that specific aircraft. However, the succeeding aircraft can be affected due to a compressed buffer between both aircraft. In that case the air traffic controller can utilize a buffer (time) to compensate for the unplanned compression.

For a too high prediction/T2F (the aircraft flies faster than predicted during final approach) the risk of wake vortex (by separation infringement) increases. This situation may lead to a go-around, hereby decreasing efficiency of the runway throughput, if the ATCo assesses the situation to be unsafe.

Selecting safety performance metrics

The average speed for the B738 (being the fastest aircraft analyzed in this thesis) over the last 4 NM is approximately 140 kts. Air traffic controllers usually (dependent on the internal rules of the Air Navigation Service Providers) need to report a separation infringement when the separation distance between two succeeding aircraft deviates more than 0.5 NM of the prescribed separation minima (using either ICAO Wake Turbulence Separation Minima or RECAT-EU Separation Minima). Calculating with an average speed of 150 kts, it takes the aircraft 12 seconds to traverse 0.5 NM. In order to build in an extra buffer for faster flying aircraft (in the provided dataset the fastest plane had an average ground speed of 180 kts) a 10 seconds limit has been selected as performance criterion X2 (see Table 2.5.3) over the trajectory of 4NM-0NM. If a flight is over-predicted more than 10 seconds probability for a direct go-around ordered by the ATCo increases, although this probability is not quantifiable in concrete numbers (note that an under-prediction of 10 seconds or more can eventually also result in a go-around). If the outcome of this research will be implemented in future, the number of go-arounds caused by separation infringements or Wake Vortex Encounters should be monitored\(^8\); it is a measure to detect if it is working properly.

The trajectory of 10NM-0NM is included in the analysis for mere benchmark purposes. At the moment of writing this thesis it is not yet clear how this can be used in a practical scenario, nor is it possible to define adequate performance criteria over this trajectory. For that reason results

\(^8\)[Eurocontrol, 2013b] states that approximately 3% of all go-arounds are caused by Wake Vortex Encounters.
for this trajectory will be compared to the currently used static model only. Ground speed predictions suffer the same problem. For both ground speed predictions and time to fly over the trajectory of 10NM-0NM a visualization will be created to show insights in the results.

**Selecting efficiency performance metrics**

In general results of this research are compared to the currently used model in terms of Root Mean Squared Error for predictions on T2F as well as ground speed over both trajectories. Time to fly predictions over the last 4 Nautical Miles are evaluated in an additional manner: the percentage of flights landing 5 seconds before or after predicted time for the 4NM-0NM trajectory. This number is based on an improved performance of the currently used static model. More info on the static model can be found in Section 5.1.

### 2.5.3 Performance metrics T2F prediction

In general the following performance metric is used for predicting Time to Fly over a certain trajectory:

**T2F.I:** Average T2F Root Mean Squared Error (RMSE, in seconds) for the entire dataset compared to the initial (static) model used right now by the Air Navigation Service Providers (ANSPs).

As is described in the previous section, for T2F predictions over the last 4 NM also the following checks are performed:

**T2F.II:** Percentage of flights having a T2F RMSE less than 5 seconds.

**T2F.II-vs:** Percentage of flights having a T2F RMSE less than 5 seconds compared to the initial model.

**T2F.III:** Percentage of flights having a T2F RMSE bigger than 10 seconds.

**T2F.III-vs:** Percentage of flights having a T2F RMSE bigger than 10 seconds compared to the initial model.

In other words, T2F prediction over the trajectory 4NM to 0NM is successful when the average RMSE over all predictions is lower than the static model’s RMSE. Also, in comparison to the static model there are less flights exceeding 10 seconds error and more flights having an error below 5 seconds.

### 2.5.4 Performance metrics GS prediction

For ground speed predictions there is one performance criterion, which is defined as follows:

**GS.I:** Average Ground Speed Root Mean Squared Error (RMSE, in kts) for the entire dataset compared to the initial (static) model used right now by the Air Navigation Service Providers (ANSPs).

---

9 The static model used by ANSPs nowadays does not make use of aircraft and weather specific information such as the actual landing weight and headwind.
2.6 Research questions review

Throughout this chapter, research questions 1 and 2 have been answered:

1. **What are appropriate performance criteria to assess possible solutions in order to secure that risks of inaccurate predictions are acceptable?**
   A distinction has been made between T2F and Ground Speed profile predictions for two trajectories (4NM-0NM and 10NM-0NM). Baseline for the performance criteria is optimizing separation accuracy (with the objective to optimize runway throughput while maintaining safety).

   A general performance metric, applicable for both trajectories and T2F as well as Ground Speed predictions is:
   - Does the new model have a lower overall prediction Root Mean Squared Error in comparison to the initial model?

   For time to fly predictions on the last 4 Nautical Miles, the following extra performance metrics have been defined:
   - What is the percentage of flights having a T2F RMSE less than 5 seconds? How does this number relate to the initial model?
   - What is the percentage of flights having a T2F RMSE larger than 10 seconds (indicating a possible safety issue)? How does this number relate to the initial model?

2. **What are the major factors influencing the speed profile of an aircraft during its final approach?**
   The major factors influencing the Final Approach Speed are:
   - Aircraft type
   - Actual landing weight of the aircraft
   - Flap settings during final approach
   - Wind conditions
   - Assigned runway

   Due to (mostly commercial) reasons in real life actual landing weight is not known by the Air Traffic Controller before final approach. Also, flap settings chosen by the pilot are not available for predicting a speed profile.

   How much each factor influences the speed (and time to fly) profile of an aircraft during final approach will be investigated in the next chapter.
Data Exploration

3.1 Data sources

NLR has available a sufficiently large test data set of aircraft position data (from radar and/or ADS-B\(^1\)) for Schiphol Airport.

For the tests with actual landing weight data a dataset provided by the Royal Dutch Airlines (KLM) has been used. This dataset contains flights of both Boeing and Airbus aircraft types of a period of approximately four years (2012-2016).

Next to flight data, also other data are needed (see Section 2.4). Weather data for the retrieved flights is gathered using an open source website\(^2\). Here it should be noted that the weather information is provided for ground level, no information with regards to wind and temperature at different flight levels is available.

The next section describes a list of all desired data fields and their availability.

3.2 Gathering and preparing data

Table 3.1 shows a list of all data fields desired to use for the speed profile predictions, based on the research conducted on the major factors influencing aircraft performance. The rows marked red represent data fields which are not available, whereas actual landing weight is only available for a part of the records.

The fields \(tdiff\) and \(gpass\) represent the target variables. They are composed of 39 values each, describing the times it took the aircraft to fly over each of the 0.5 Nautical Miles intervals (for the \(tdiff\) variable) and the average ground speed the aircraft had during those intervals.

---

\(^1\)Automatic Dependent Surveillance – Broadcast (ADS-B) is a surveillance technology in which an aircraft determines its position via satellite navigation and periodically broadcasts it, enabling it to be tracked. The information can be received by air traffic control ground stations as a replacement for secondary radar.

\(^2\)http://mesonet.agron.iastate.edu/request/download.phtml
In order to compose the dataset, first the flight dataset was scraped to gather flight information of the year 2015. The flight dataset exports need to be converted before they can be used, for multiple reasons. First of all, data is exported on multiple rows and columns, depending on the number of measurements. This introduces variable length for the flight data points since not all flights are monitored over the same period of time. Measurements are provided every 4 seconds. To be able to compare flights data points need to be available on fixed distances from the threshold, so-called passing gates. Several (MATLAB) scripts are used to calculate the speed and time on the passing gates; the scripts are also used to convert the dataset flights into a dataset in which every row represents a single flight. After this step a (Python) script is used to calculate the remaining Time to Fly (T2f) at the passing gates, find the weekdays of the flights and convert the time into a workable format. As a final step flights containing missing measurements were removed in order to keep the dataset consistent. Altogether the dataset consists of 141 data fields with a total number of 23300 flights.

Table 3.1: Dataset description

<table>
<thead>
<tr>
<th>Data field</th>
<th>Data Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flightnr</td>
<td>Flight identifier</td>
</tr>
<tr>
<td>actype</td>
<td>Aircraft type</td>
</tr>
<tr>
<td>rwy</td>
<td>Runway</td>
</tr>
<tr>
<td>rwyorientation</td>
<td>Runway orientation [deg]</td>
</tr>
<tr>
<td>icaocat</td>
<td>ICAO WTC (1=Super, 2=Heavy, 3=Medium, 4=Light)</td>
</tr>
<tr>
<td>METARtime</td>
<td>Measurement time of closest METAR observation</td>
</tr>
<tr>
<td>METARdrect</td>
<td>METAR wind direction</td>
</tr>
<tr>
<td>METARSknrt</td>
<td>METAR wind speed in kts</td>
</tr>
<tr>
<td>METARGust</td>
<td>METAR gust</td>
</tr>
<tr>
<td>METARVsbym</td>
<td>METAR visibility [NM]</td>
</tr>
<tr>
<td>METARtmpc</td>
<td>METAR temperature [deg celsius]</td>
</tr>
<tr>
<td>METARcsrnd</td>
<td>METAR crosswind [kts]</td>
</tr>
<tr>
<td>METARhwind</td>
<td>METAR headwind [kts]</td>
</tr>
<tr>
<td>icaocatcombi</td>
<td>Combination of leader and follower ICAO WTC</td>
</tr>
<tr>
<td>actypecombi</td>
<td>Combination of leader and follower aircraft type</td>
</tr>
<tr>
<td>icaorefsep</td>
<td>ICAO minimum separation</td>
</tr>
<tr>
<td>recateurefsep</td>
<td>RECATEU minimum separation</td>
</tr>
<tr>
<td>ACin10NM</td>
<td>Number of aircraft within 10 NM (including this one) to indicate traffic density</td>
</tr>
<tr>
<td>weightfactor</td>
<td>Actual landing weight (as a factor of the maximum landing weight) of the aircraft</td>
</tr>
<tr>
<td>flapsetting</td>
<td>Flap setting used during final approach</td>
</tr>
<tr>
<td>autothrottle</td>
<td>Autothrottle setting used or not (Boeing only)</td>
</tr>
<tr>
<td>pilot</td>
<td>Name of the pilot, for his or her landing preferences</td>
</tr>
<tr>
<td>gate</td>
<td>Assigned gate on the airport</td>
</tr>
<tr>
<td>weekday</td>
<td>Day of the week on landing</td>
</tr>
<tr>
<td>landingtime</td>
<td>Time of landing (hours:minutes:seconds)</td>
</tr>
<tr>
<td>t2f</td>
<td>Time to fly till landing at passing gates [19:-0.5:0.0] NM before runway</td>
</tr>
<tr>
<td>tdiff</td>
<td>Time it takes to fly from passing gate to the next [19:-0.5:0.0] NM before runway</td>
</tr>
<tr>
<td>hpass</td>
<td>Height [ft] passing gates [19:-0.5:0.0] NM before runway</td>
</tr>
<tr>
<td>gspass</td>
<td>Ground speed [kts] passing gates [19:-0.5:0.0] NM before runway</td>
</tr>
</tbody>
</table>
Merging flight dataset with the actual landing weights

Actual landing weights are available as a separate dataset for a part of the flight dataset and are merged first in order to create a powerful dataset which contains as much data on the aircraft as possible.
Merging is handled based on the projected landing times and the registration (unique identifier) of the aircraft. Since landing times in both datasets do not match on the second the closest flight is selected and the landing weight is merged with the flight if the unique registration matches. A margin of 1 hour is used, which is considered safe since a unique registration can not land twice on Schiphol within the hour.

3.3 Analyzing influence relevant factors on FAS

Data analysis is useful for multiple reasons. Theory and physical laws are in place (see Section 2.3), but that does not automatically imply that the actual data represents the same behavior. There can be other (unforeseen) factors influencing Final Approach Speed, which are not included in the physical model. Also, there can be parameters which are included in the model but there is no data available for these parameters.

Section 2.4 describes the major and minor factors influencing Final Approach Speed, based on theory and physical models. For the major factors (actual landing weight, aircraft type/-manufacturer, assigned runway and wind conditions) an analysis is conducted to determine the correlation with the speed profile (and hence, time to fly) of an aircraft. The results of this analysis are described in this section.
First, distribution of the ground speed is provided to get a feeling of the data, followed by two plots visualizing the theoretical FAS vs the actual IAS during final approach. Next, analysis of the relevant factors on Final Approach Speed (according to theory) is examined, starting with the influence of a runway.

The analysis is performed for two aircraft types, a Boeing 737-800 and an Airbus A330-200, two very popular aircraft built by different manufacturers and having different characteristics.

3.3.1 Data distribution ground speed

In order to get a better understanding of the data first a plot has been drafted to show more insights in the distribution of individual flights of the B738 and A332. The plot is displayed in Figure 3.1 and shows a histogram of average ground speeds over the last 4 Miles for each aircraft type (the Final Approach Speed). Note that the data contained the entire range of wind conditions, landing weights, etc. There was more data available for the B738, hence the difference in height of the plots.

Judging the distribution in the figure it follows that these two distinct aircraft types overlap each other for a major part of the dataset with regards to ground speed, making it hard to distinguish aircraft type based on only ground speed profiles.
3.3.2 Theoretical FAS ($V_{app}$) vs. Actual IAS (Airbus only)

$V_{app}$ (also known as the Final Approach Speed for the Airbus) can either be determined by the pilot manually (by using figure 3.2) or automatically by the Airbus GS-Mini system\(^3\). The actual IAS was acquired from the flight data.

---

\(^3\)Speed prediction based on the GS-Mini system has not been done, due to missing actual wind information at different altitudes.
To calculate $V_{app}$, first $V_{LS}$ is determined based on Figure 3.2 and the actual weight the pilot can read in the cockpit. As there is no data available regarding the flap settings, all flights are assumed to land with full flaps. The weights (W) listed in this table are matched, by means of interpolating, with the actual landing weights of all flights. Together with the associated headwind data, two scenarios for calculating the theoretical $V_{app}$ are possible, based on equation 2.7 or 2.8.

Since $V_{app}$ is displayed in IAS, the radar data which is measured in ground speed needs to be transformed into IAS. To accomplish this, headwind conditions for a flight needs to be added to the measured (average) ground speed.

Figure 3.3 contains the plot of the variation for every $V_{app}$ value. The red asterisks indicates the expected IAS. Bearing the aircraft $V_{app}$ calculation table in mind, a clustering around these red asterisks was expected, however only a part of data points are situated near the expected value. In order to explain the variance, additional analysis is required.

![Theoretical FAS vs Actual IAS (A332)](image)

**Figure 3.3:** A plot of theoretical $V_{app}$ versus the actual (average) IAS for the A332. The red asterisks indicate the expected clusters of data.

### 3.3.3 Theoretical FAS vs. Actual IAS for the Boeing 737-800

To calculate the FAS for a Boeing aircraft, first $V_{REF}$ is determined based on Table 3.2 and the actual weight. As there is no data available regarding the flap settings, for this analysis all flights are assumed to land with flap setting 30 (the most common setting). The weights listed in this table are matched, by means of interpolating, with the actual landing weights of all flights. Together with the associated headwind data, FAS is calculated based on equations 2.5 and 2.6 (gust information was only given for 4% of the flights and was therefore not taken into account calculating the FAS).

Since FAS (as well as $V_{app}$) is displayed in IAS, radar data needs to be transformed into IAS. This
**Table 3.2: \( V_{REF} \) Calculation Table Boeing**

<table>
<thead>
<tr>
<th>WEIGHT</th>
<th>FLAPS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LBS (x1000)</strong></td>
<td><strong>KGS (x1000)</strong></td>
</tr>
<tr>
<td>180</td>
<td>82</td>
</tr>
<tr>
<td>170</td>
<td>77</td>
</tr>
<tr>
<td>160</td>
<td>73</td>
</tr>
<tr>
<td>150</td>
<td>68</td>
</tr>
<tr>
<td>140</td>
<td>64</td>
</tr>
<tr>
<td>130</td>
<td>59</td>
</tr>
<tr>
<td>120</td>
<td>54</td>
</tr>
<tr>
<td>110</td>
<td>50</td>
</tr>
<tr>
<td>100</td>
<td>45</td>
</tr>
<tr>
<td>90</td>
<td>41</td>
</tr>
</tbody>
</table>

is again accomplished by adding headwind conditions for a flight to the measured (average) ground speed.

**Figure 3.4:** A plot of theoretical FAS versus the actual (average) IAS for the B738. The red asterisks indicate the expected clusters of data.

Figure 3.4 contains the plot of the variation for every FAS value. As for the A332, only a few data points are situated near the expected value and additional analysis is required.
3.3.4 The influence of a specific runway on ground speed

Whether the assigned runway influences the Ground Speed profile is first analyzed. In order to solely measure the effect of the runways, the effect of the actual landing weight and the headwind on the ground speed should be excluded.

Excluding the effect of actual landing weight

Actual landing weight is excluded by first picking a pivot value \( \bar{w} \) (being the mean of all landing weights in this dataset), see equation 3.1. For this pivot value \( \bar{w} \), the corresponding pivot \( V_{LS} \) is determined by using the table in Figure 3.2 and is referred to as \( V_{LS, \bar{w}} \).

\[
W = \{\text{the set of actual landing weights in the dataset}\}
\]

\[
\text{pivot value weights} = \bar{w} = \frac{1}{n} \sum_{i=1}^{n} w_i, w_i \in W
\]  

(3.1)

Next, \( V_{LS} \) of a flight is determined using the same table in Figure 3.2, which is referred to as \( V_{LS,w} \). In order to correct for the variation in actual landing weight of a flight, a correction value \( (C_w) \) is determined (equation 3.2), correcting for the variation of the actual landing weight of any specific flight. The correction value is calculated for all flights in the dataset.

\[
\text{For each flight } w: C_w = (V_{LS, \bar{w}} - V_{LS, w})
\]  

(3.2)

The difference between \( V_{LS} \) of a flight and the pivot value \( V_{LS, \bar{w}} \) determines the correction value for a flight; if an aircraft is heavier than average, the correction value is negative. For aircraft lighter than average a positive correction value is obtained.

Excluding the effect of headwind

Excluding the effect of headwind is handled in a similar way, using equation 3.3 and equation 3.4. In this equation \( \bar{h}w \) is the pivot headwind, being the mean of all \( METARhw \) values in the dataset.

\[
\text{pivot value headwind} = \bar{h}w = \frac{1}{n} \sum_{i=1}^{n} hw_i, hw_i \in HW
\]  

(3.3)

For this pivot value \( \bar{h}w \), the associated correction value (described in section 2.3) is determined by the use of equation 2.6 (Boeing) and 2.8 (Airbus), and is defined as \( c_{\bar{h}w} \).

As with the actual weight correction value, the next step is determining the associated correction value for a given headwind value during the final approach of an aircraft (the \( METARhw \) variable in the data), which is defined as \( c_{hw} \) and is again found by using equation 2.6 (Boeing) and 2.8 (Airbus).

\footnote{For the Boeing 738 \( V_{ref} \) is considered instead of \( V_{LS} \)}

\footnote{For the B738, Table 3.2 is considered instead}
Equation 3.4 is used to determine the (overall) headwind correction value \( (C_{hw}) \) for a flight. For all flights in the dataset this correction value has been calculated in order to exclude the effect of headwind.

\[
C_{hw} = c_{hw} - c_{hw}
\]  

Finally the correction value for actual landing weight and headwind is calculated to retrieve the corrected ground speed (excluding landing weight and headwind effect) for a flight, based on its actual average ground speed over the last 4 NM (equation 3.5).

\[
GS_{corrected} = GS_{actual} + (C_w + C_{hw})
\]  

For both aircraft types B738 and A332, the corrected ground speeds have been plotted against the runways they landed on in Amsterdam. By creating a Box and Whisker plot whether there are important differences between runways with regards to average ground speeds over the last 4 miles. Figure 3.5 contains the created plot.

**Figure 3.5:** Box and Whisker plot for the average corrected ground speed for the A332 (left) and B738 (right) on Schiphol Airport’s runways. The red dotted lines indicate the mean corrected average ground speed for a runway. The green dotted line indicate the overall mean for all runways.

In both plots it can be observed that the average of corrected ground speeds for runway 06, 18C, 18R, 36C and 36R are close to the overall mean (represented by the green dotted line) and therefore appears to have limited influence on the GS. For runway 22, 27 and 24 there is a mismatch between the mean of average corrected ground speeds for those runways and the overall mean:
• Runway 22, being the shortest runway at Schiphol Airport and only being used in a break-off approach procedure\textsuperscript{6}, has a significant lower average ground speed (which can be explained for that reason). This runway is excluded from further analysis.

• A possible explanation for the lower average ground speed of runway 27 can be the fact that runway 27 is usually used in high wind conditions, resulting in lower average ground speed. Its lower average ground speed is not because of the runway characteristics but because of the prevailing wind conditions when this runway is in use. This runway is therefore included in the further analysis.

• Runway 24 is only very limited available in the data of the B738. Due to lack of data this runway is excluded from further analysis.

Conclusion on the influence of a specific runway on ground speed

Based on the analysis performed above and the condition that data for runway 22 and 24 are excluded, a conclusion can be drawn that there is no need to further take into account distinct runways while analyzing the correlation between actual landing weight and headwind on the ground speed. Runway 22 seems to have characteristics that influence average ground speed, which are not further investigated in this research.

3.3.5 Correlation between landing weight and ground speed

Since only the effect of the actual landing weight needs to be considered, other parameters (such as wind conditions) are excluded using the method described in Section 3.3.4. By using the correction value for the headwind a corrected ground speed is calculated for each flight in the dataset, using the following formula (equation 3.6):

\[ GS_{\text{wind-corrected}} = GS_{\text{actual}} + C_{hw} \]  

Figure 3.6 shows a scatter plot of the weight vs the corrected ground speed for the Airbus A330-200.

\textbf{Note:} The x and y-axis in this plot (and the following ones in this chapter) are normalized to be able to compare them to each other. Normalization is performed by the use of equation 3.7.

\[ x' = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \Rightarrow y' = \frac{y - y_{\text{min}}}{y_{\text{max}} - y_{\text{min}}} \]  

\textsuperscript{6}A procedure in which pilots had to break off a landing at a certain runway and divert to runway 22 instead.
In the plot the green dashed line indicates the trend line in the data. As expected and observed, there is a positive relation between the actual landing weight and the ground speed:

**Number of flights A332: 791**
**Correlation factor weight vs ground speed A332: 0.23**
**p-value: <0.001**

![Correlation between landing weight and ground speed (A332)](image)

**Figure 3.6: Actual landing weight vs. ground speed A332**

The effect of actual landing weight on IAS for the Boeing 737-800 is illustrated in Figure 3.7 and has the following characteristics:

**Number of flights B738: 4528**
**Correlation factor weight vs IAS B738: 0.44**
**p-value: <0.001**

---

7The correlation factor is a number ranging between −1 and 1, representing the strength of a linear association between two variables. Closer to 1 indicates a more positive linear correlation (1 being the perfect positive correlation) whereas smaller than 0 and closer to −1 indicates a more negative linear correlation.
3.3.6 Correlation between headwind and ground speed

In order to conduct the analysis on the correlation between headwind and ground speed actual landing weight is excluded by again using the method in Section 3.3.4. The corrected ground speed is calculated by using the correction value for the landing weight for each flight in the dataset, using the following formula (equation 3.8):

\[ GS_{\text{weight-corrected}} = GS_{\text{actual}} + C_w \] (3.8)

Figure 3.8 displays the scatter plot for the effect of headwind on the ground speed for A330-200 flights.

The plot indicates a negative correlation between wind and weight-corrected Ground Speed. This observation is just as expected judging equation 2.2 (a higher headwind results in a lower ground speed).

*Number of flights A332: 791*
*Correlation factor headwind vs. ground speed A332: -0.63.*
*p-value: 3.18 \times 10^{-74}*

For the B738, the plot for the effect of headwind on ground speed is displayed in Figure 3.9.

*Number of flights B738: 4528*
*Correlation factor headwind vs ground speed B738: -0.65.*
*p-value: 0.0*
From the analysis performed in this section there is a correlation between actual landing weight and ground speed, as well as a correlation between wind and ground speed. This is expected since weight and wind are two major factors influencing speed. Based on the correlation factors, it seems that the effect of headwind on ground speed is stronger than the influence of the
actual landing weight. Most likely, this will be confirmed conducting experiments with Machine Learning models later on in this thesis.

Next to correlation, variation in the data is high. This might be an indicator that there are other forces having a huge impact on final approach speed; forces not being the major factors identified. These forces may be found in the factors of which no data is available (like flap settings and assigned gate), but another aspect which can not be found in the data is the fact that there are still human beings flying an aircraft (at least in good weather) and considering how occupied a runway is they still have a a certain degree of freedom in how they handle their final approach. Also it needs to be noted that the METAR (i.e. wind) information available only contains ground level data and is updated every 30 minutes. For better precision METAR data on different flight levels and more precise time would be preferred.

3.4 Research questions review

Throughout this chapter, an answer to research question 2 has been provided:

**What is the influence of the identified major factors on the speed profile of an aircraft during its final approach?**

As acknowledged in Section 2.4, for the identified major factors influencing FAS data is available for actual landing weight, assigned runway and headwind. On Schiphol Airport only runway 22 appears to have an influence on the speed profile. A choice has been made to exclude this runway from further analysis, as is done for runway 24 due to lack of data for that runway.

The influence of actual landing weight and headwind is expressed in a correlation factor, which can range from -1 (perfect negative correlation) to 1 (perfect positive correlation). There is a positive correlation of 0.23 (A332) / 0.44 (B738) between actual landing weight and average ground speed during final approach.

For headwind, correlation is negative and stronger in comparison to actual landing weight: -0.63 (A332) / -0.65 (B738).
Part II

Predicting Speed / Time to Fly Profile
Machine Learning methods for aircraft performance prediction

This chapter describes the approach that has been taken in order to identify what Machine Learning methods are applicable for predicting aircraft performance. First, related work in the domain of airspeed prediction using Machine Learning techniques is discussed followed by related work with regards to traveling time prediction for buses, a similar research field. In the end, an answer to research question 4 is provided.

4.1 Aircraft performance prediction domain

In the aircraft domain several initiatives have been taken to make predictions on the airspeed using Machine Learning techniques. In [Alligier et al., 2015], the authors have applied several techniques to improve aircraft climb prediction in the context of ground-based applications. Using mass and speed intent of the aircraft as key parameters, they face the same problem that these parameters are not available to ground-based trajectory predictors. The authors have built a dataset of historical flights using radar data as well as wind and temperature data at several altitudes. Also flight meta data like departing and arrival data is collected for every flight. Every 15 seconds climb segments are sampled (in total 51 points, of which 11 are used to predict the mass and speed profile). They have applied a Gradient Tree Boosting algorithm, able to handle categorical features and robust to outliers. Comparing the results with the reference speed profiles provided by BADA\(^1\), the reduction of the speed Root Mean Squared Error (RMSE) ranges from 36% to 79%.

The paper of [Herrema et al., 2015] presents a Machine Learning feasibility study for predicting T2F and TAS profile on final approach as a function of meteorological parameters using data on two major European airports. The authors have used methods based on linear regression techniques and on neural networks. Datasets have been composed of radar and weather data (wind and visibility). They have used a supervised learning method to exploit multi-task learning by predicting the segments from 0.5NM to 10NM altogether. RreliefF modeling and Principal

\(^1\)http://www.eurocontrol.int/services/bada
Component Analysis (PCA) have been used to gain information on the importance of the features; ground speed and headwind are the most influencing factors for the variance in T2F and TAS profiles in the last 10NM. The multi-task techniques Lasso and MLP turned out to be the best feasible and most accurate techniques for predicting the TAS and T2F from 8.5NM till 0.5NM and from 4.5NM till 0.5NM. Combining these techniques results in a more robust and accurate ML model.

In [Hrastovec and Solina, 2014] the authors use a Nearest Neighbor algorithm to search for similar flights in a database and predict aircraft performances based on similar flights performed in the past. Their data sources are flight tracks recorded by radars, correlated by flight plans filled in by pilots prior to take off. Weather data is used as a third source. In the performed experiment the algorithm used by the authors in combination with the multi-dimensional database outperforms the existing methods, however apparently this was not satisfactory yet.

De Leege, van Paassen and Mulder [de Leege et al., 2013] are using Generalized Linear Models (GLM) to predict trajectories along one particular landing procedure of a 45 NM length. They are using aircraft type (heavy, medium), aircraft ground speed, altitude over initial point and winds as model inputs. The model predicts with an approximately 5s error on the last 15 nautical miles and 20s error on the 45 miles trajectory.

### 4.2 Traveling time prediction for buses

The context of the problem, predicting time to fly profiles for aircraft in their final approach, very much resembles the problem of traveling time prediction for a bus following a route to an end station. The bus also passes several 'gates' along the route and has interference of external variables like traffic, passengers, etc.

The problem of predicting traveling times of buses using Machine Learning techniques has been researched extensively over the past decades. A variety of models and algorithms have been developed for this purpose.

In [Gal et al., 2015] the authors proposed a prediction for the estimated traveling time of a bus on a specific route considering both historical data and real-time information submitted by buses. First, bus journeys are modeled in segments (bus stops). Then, two prediction methods are used: a method that comes from Queuing Theory (snapshot principle) and a method based on decision trees. Several ensembles of regression trees are compared, in combination with and without the snapshot principle. The combination of methods improves performance based on their experiment.

The authors of [Lee et al., 2012] have developed a new bus travel time prediction framework. The HTTP framework samples a set (using clustering algorithms) of similar trajectories as basis in stead of relying on the (one) best historical trajectory match.

Altinkaya and Zontrul [Altinkaya and Zontul, 2013] have reviewed computational models (based on historical data, statistical, Kalman filtering as well as Machine Learning models) for predicting bus arrival times. Their findings are summarized below:
• Models using historical data require an extensive set of historical data; accuracy largely depends on the similarity between the real-time data and historic traffic patterns.

• Time series models assume that historic traffic patterns will remain the same in the future, changes in relationship between real-time and historic data can have significant influence on accuracy.

• Regression models are able to work properly even when real-time data shows deviating patterns from historical data. An advantage of multilinear regression models is that they reveal less or more important inputs. Multiple studies indicated that regression models are outperformed by other models.

• Kalman filtering algorithms are (in basics) used to provide estimates of the current state of the system, but also for predicting future values. The results of using Kalman filters for predicting bus arrival times is very promising. They provide dynamic travel time estimates, which other models are lacking. Kalman filtering models are effective in predicting traveling time one or two time periods ahead, but their accuracy deteriorates with multiple time steps.

• Artificial Neural Networks (ANNs) have the ability to capture complex non-linear relationship between travel time and independent variables. ANNs can be very useful when it is difficult or even impossible to mathematically formulate the relationship between input and output.

4.3 What prediction methods to use?

In order to choose what algorithms to use for predicting aircraft speed profiles, it is important to know what characteristics they should have. Often, they are classified by their interpretability and predictive power. There are also other ways of classifying them, as is described in [Hastie et al., 2009]. For this research predictive capability is the most important aspect, since accuracy of speed profiles is key. Another facet relevant for the research on aircraft performance is how steady predictions are overall; a low variance in prediction errors is of importance in order to prevent having flights being much slower or (even worse due to increased risk on separation infringement) much faster than predicted.

As stated (Section 4.1) in relevant research within the aircraft performance domain, [Alligier et al., 2015] used a Gradient Tree Boosting algorithm with success whereas [Herrema et al., 2015] successfully studied several methods including Artificial Neural Networks and multiple linear regression techniques. Also Nearest Neighbour algorithms have been used [Hrastovec and Solina, 2014], although not fully satisfactory.

There is a lot of historic flight data available within NLR and there is no reason to assume that behaviour of aircraft will change drastically on short term. Backed up by the conclusions of [Altinkaya and Zontul, 2013] in bus travel time prediction, it makes perfect sense that supervised Machine Learning methods are used to predict speed profiles for new flights.

Random Forests (as part of ensemble methods) are known for being relative simple to use, having low variance en making predictions that can be used as benchmark for other methods. Also, they have never been used in an experiment within the domain of aircraft performance
prediction, providing enough reasons to use them in an experiment in this research. The ensemble ‘family’ contains many more algorithms having the desired characteristic of low variance combined with good performance [Caruana and Niculescu-Mizil, 2006], out of which also Bagging Trees, Gradient Boosting Trees and Adaptive Boosting are selected for comparison in the next chapter.

Each flight in the available dataset consists of 39 points monitoring average speed/height between passing gates and time an aircraft passes a certain gate. This series of data points are listed in time order, using a passing gate as fixed point in time. In that way, a flight can be seen as a Time Series. In other research in the same domain Time Series Forecasting has never been applied as a possible solution for predicting speed profiles. In this thesis an experiment will be devoted to the predictive performance of Time Series Forecasting.

Due to its popularity these days in combination with Time Series data and a hint from an expert, also performance of Gaussian Processes will be evaluated.

4.4 Research questions review

In this chapter research question 4 has been addressed:

**What Machine Learning techniques can be used to make predictions on the speed and time to fly profile of an aircraft during final approach?**

*In theory there are many methods to predict speed profiles of an aircraft. Research in the aircraft speed prediction domain as well as research in a similar domain as traveling times for buses has provided insights in applicable algorithms that deliver proper results. Based on that research and gaps that haven’t been filled yet a choice has been made to focus on multiple algorithms from Ensemble methods, Time Series forecasting and Gaussian Processes in this thesis.*
Experiments

This chapter outlines the conducted experiments with the chosen methods. The dataset used during these experiments is described in Section 3.2. Whenever a subset of this dataset was used, it is noted in the particular experiment.

First, the currently used static model is analyzed to make predictions on time to fly and ground speed. This model is used as reference for the other experiments.

5.1 Initial Method error calculations

Currently a static model (see [van Baren et al., 2016]) is used to predict the Final Approach Speed profile and Time to Fly. For each aircraft a reference speed profile (TAS, in kts) is defined for the last 10 Nautical Miles. Picture 5.1 displays the reference (ground) speed profiles for the B738 and A332. In this thesis the currently used model is referred to as the 'initial model'.

Using the available dataset, the following was performed to obtain the T2F and Ground Speed Root Mean Squared Errors for aircraft type B737-800 and A330-200 for the initial model:

- A flight’s trajectory is split into five distance points (passing gates 10-5NM, 4.5NM, 4NM, 3.5NM and 3-0NM) based on the reference speed profile as depicted in Figure 5.1.

- For each of those points in a flight Ground Speed of the initial model is calculated by subtracting headwind of that flight.

- Since Ground Speed is in kts (NM/h), it is fairly trivial to convert this into a T2F over the point’s distance by dividing the distance (e.g. 5NM over the distance 10-5NM and 0.5NM for the point 4.5NM) by the initial Ground Speed. In order to get this T2F for the 5 distance points in seconds the individual results are divided by 3600.

- T2F for the initial model can then by calculated for the trajectory 4NM-0NM and 10NM-0NM by summing all individual T2F values for the respective trajectory.
Figure 5.1: Initial Model Ground Speed Profiles B738/A332

- Average Ground Speed for the same trajectories is calculated by averaging the average ground speeds for the distance points taking into account the distance (e.g. Average Ground Speed 4NM-0NM = \( \frac{0.5 \times GS_4 + 0.5 \times GS_3.5 + 3 \times GS_3}{4} \))

Now that the T2F and Ground Speed according to the initial model are known and the actual T2F and actual Average Ground Speed over the last 4NM (and 10NM) is available in the data, Mean Squared Errors can be calculated.

The Mean Squared Error is defined as Equation 5.1:

\[
MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2
\]  
\text{(5.1)}

Tables 5.1, 5.2, 5.3 and 5.4 contain the Mean Squared Errors (using Equation 5.1), standard deviation and Root MSE calculated calculated for the initial model using the available dataset for T2F and Ground Speed on the trajectories 4NM-0NM and 10NM-0NM. The two right most columns in the Table 5.1 indicate the evaluation of the performance criteria as defined in Section 2.5.3:\n
Table 5.1: Initial Model results T2F 4NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE (s)</th>
<th>T2F.II</th>
<th>T2F.III</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>34.09</td>
<td>86.64</td>
<td>5.84</td>
<td>64.73%</td>
<td>7.13%</td>
</tr>
<tr>
<td>A332</td>
<td>35.12</td>
<td>62.44</td>
<td>5.93</td>
<td>61.88%</td>
<td>8.25%</td>
</tr>
</tbody>
</table>

\(^1\)Since T2F.I and GS.I are relative to this initial model they are not included here.

\(^2\)4NM-0NM: T2F.II: % of flights with RMSE < 5s / T2F.III: % of flights having RMSE > 10s.
Table 5.2: Initial Model results Ground Speed 4NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE (kts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>82.45</td>
<td>696.39</td>
<td>9.08</td>
</tr>
<tr>
<td>A332</td>
<td>57.88</td>
<td>213.80</td>
<td>7.61</td>
</tr>
</tbody>
</table>

Table 5.3: Initial Model results T2F 10NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>247.98</td>
<td>489.32</td>
<td>15.75</td>
</tr>
<tr>
<td>A332</td>
<td>249.96</td>
<td>424.67</td>
<td>15.81</td>
</tr>
</tbody>
</table>

Table 5.4: Initial Model results Ground Speed 10NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE (kts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>203.34</td>
<td>663.54</td>
<td>14.26</td>
</tr>
<tr>
<td>A332</td>
<td>137.90</td>
<td>260.65</td>
<td>11.74</td>
</tr>
</tbody>
</table>

5.2 Experiments using Random Forest Regressor

Random forests are a combination of decision tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [Breiman, 2001].

As a preparatory step in this experiment two tests have been performed. First, the effect of selecting a different number of trees is investigated with regards to error and bias-variance trade-off (see Appendix B for more info on this trade-off). Next, using an optimal number of trees (optimal with respect to error and required calculation power), analysis for individual features in the dataset has been conducted. Combining individual features with other features to see the effect on the prediction error was done as last.

For this analysis a RandomForestRegressor has been picked as modeling technique for several reasons: A Random Forest performs implicit feature selection and provide a pretty good indicator of feature importance, they are quick to train and Random Forests deliver proper benchmark results.

Note that besides the number of estimators (trees), default settings of a RandomForestRegressor have been used in this experiment. The reason for this is to define benchmark results that can be compared during later experiments.

Followed procedure for calculating Errors, Bias & Variance

Retrieving Bias and Variance are not built in Python functions, manual coding is required. In order to be able to calculate them, resampling with replacement (bootstrapping\textsuperscript{3}) is used. One

\textsuperscript{3}Bootstrap: random selection with replacement of a sample of the size of the data set. On average 63.2% of the data are selected (some with repetitions). The selected examples are used as training set. The remainder out of bag examples (36.8% in average) are used as test set [Efron and Tibshirani, 1994].
bootstrap replicate is the length of the dataset (denoted as $X$) randomly sampled flights. The sampling is with replacement, so some of the flights will be in the bootstrap sample multiple times (forming training set $T_r$) and other flights will not appear at all (forming the test set $T = X \setminus T_r$).

For the analysis 100 bootstrap replicates have been created. For each bootstrap replicate, the RandomForestRegressor has been learned on the training set and a prediction $\hat{f}(x)$ has been computed for each $x \in T$.

For each original data point $x$, we have the actual corresponding value $f(x)$ and a number $k \leq 100$ of predictions $\hat{f}_j(x)$, $j = 1, \ldots, k$. The average prediction $\bar{f}^*$ can now be calculated; Bias is estimated as $(\bar{f}^* - f^*)$. Variance is calculated as $\frac{1}{k-1} \sum_{j=1}^{k} \left( \hat{f}_j(x) - \bar{f}^*(x) \right)^2$.

**Determining optimal number of trees in the Random Forest**

Figure 5.2 shows the effect of the number of trees in a RandomForestRegressor on the Bias and Variance of T2F predictions over the last 4 and 10 Nautical Miles. The dataset of the B738 has been used, consisting of 4590 unique flights. From this plot, it can clearly be seen that increasing the number of trees in the Random Forest decreases variance. Bias, on the other hand, is more or less constant for any number of trees.

Keeping in mind that using more trees also takes more time to make predictions, a choice has been made to use 32 trees for further analysis in this section.

![Figure 5.2: Bias / Variance vs Number of Trees in RandomForestRegressor](image-url)
5.2.1 Experiment setup

Initial Time to Fly prediction using Random Forest Regressor

Using the procedure described above with 32 trees in the Random Forest\(^4\), a couple of tests have been performed to measure Bias, Variance and Errors (MSE/RMSE) for the Boeing 737-800 and Airbus A330-200 aircraft in several scenarios. Multiple targets \(t_{diff}\)\(^5\) (see Table 3.1) and \(g_{\text{pass}}\)\(^6\) for the trajectories 10NM-0NM (20 distinct values) and 4NM-0NM (8 distinct values) have been chosen as targets. The following scenarios for the input feature space have been chosen:

- Actual landing weight only
- Actual landing weight + \(t_{diffs}\)\(^7\)
- Headwind only
- Headwind + \(t_{diffs}\)
- All of the above + all other Meteo information, number of aircraft within 10NM and the runway id.
- All of the above + all other Meteo information, number of aircraft within 10NM and the runway id except actual landing weight since this information is usually not available in a real time situation.

To make T2F/Average Ground Speed predictions for the last 4 NM, the time differences between gates on trajectory 19.5NM-10NM have been used in the feature space. To enable an Air Traffic Controller (ATCo) to make adjustments to an aircraft’s final approach plan, the ATCo needs to plan ahead of time based on ‘only’ this trajectory. Predictions will be better if an aircraft is closer to runway threshold, but there is a trade-off between the ability to still being able to influence final approaches and accuracy of predictions. In consultation with NLR a decision has been made that making a prediction based on available data until 10NM before threshold provides a realistic time window for an ATCo. For predictions on the last 10NM time available information up and till 15NM before runway threshold will be considered.

5.2.2 Results

Tables 5.5-5.8 show a summary of errors, standard deviation of the MSE, squared Bias, variance and the scores on the evaluation criteria for B738 and A332 (as defined in Section 2.5.3). A green marked cell means that particular evaluation criteria has passed, red cells indicate the opposite (in other words, the initial model is better in those cases). Note that only for T2F prediction on the last 4NM T2F.II and T2F.III is included.

---

\(^4\) All other hyperparameters not altered from their default values  
\(^5\) The term \(t_{diff}\) comprehends the time differences between passing gates.  
\(^6\) The term \(g_{\text{pass}}\) comprehends the average ground speed over a 0.5 NM gate distance.  
\(^7\) The number of gates taken into account depends on the to be predicted trajectory (4 NM: 19.5NM-10NM, 10 NM: 19.5NM - 15NM)
The first Table summarizes the results for the trajectory of 4NM to runway threshold, whereas the second one shows the results of the trajectory 10NM-0NM. Each row represents a new scenario, using different features from the feature space (‘All’ indicating the entire feature space).

In this experiment several \( t_{diff} \) values were used as targets. Summed, all \( t_{diff} \) values add up to the T2F over a particular trajectory (4 NM: last 8 \( t_{diff} \) values, 10 NM: last 20 \( t_{diff} \) values). As extra experiment a single target was predicted, being this summed T2F, to check whether there are differences in prediction errors. The outcome of this experiment is that the results are equal: there is no difference in predicting a time to fly profile as independent time differences between gates, or as a single T2F target in terms of error. For Ground Speed predictions the same holds, although in this case not the sum is taken over a specific trajectory; the average over the last 8 \( gspass \) values in case of a 4 NM trajectory (and hence 20 \( gspass \) values for the last 10 NM) result in equal prediction errors compared to multiple \( gspass \) target variables.

For algorithms not capable of predicting multiple targets the above forms an alternative.

### 5.2.3 Conclusions

A couple of conclusions can be drawn from the tables. First of all, taking into account T2F predictions for the last 4 NM:

- Using the entire feature space results in the lowest RMSE, although there is only a marginal difference leaving out the actual landing weight.

- Headwind has more effect on the outcome of predictions compared to actual landing weight, which is positive for future research since actual landing weight is not available during live approaches while headwind is.

- Compared to the initial model and taking into account the entire feature space (or even leaving out the landing weight), this RandomForestRegressor performs better on all criteria.
### Table 5.5: Random Forest Regressor (default settings) results T2F 4NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Features</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>VarBias</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B738 ALW</td>
<td>T2F.I</td>
<td>78.66</td>
<td>125.21</td>
<td>8.87</td>
<td>-28.98%</td>
</tr>
<tr>
<td></td>
<td>T2F.II</td>
<td>78.66</td>
<td>125.21</td>
<td>8.87</td>
<td>-28.98%</td>
</tr>
<tr>
<td></td>
<td>T2F.III</td>
<td>78.66</td>
<td>125.21</td>
<td>8.87</td>
<td>-28.98%</td>
</tr>
<tr>
<td></td>
<td>T2F.II vs T2F.I</td>
<td>6.87</td>
<td>125.21</td>
<td>8.87</td>
<td>-28.98%</td>
</tr>
<tr>
<td></td>
<td>T2F.III vs T2F.I</td>
<td>6.87</td>
<td>125.21</td>
<td>8.87</td>
<td>-28.98%</td>
</tr>
</tbody>
</table>

**VarBias** indicates the percentage of lightings having a RAMSE > 5 seconds.

**Bias** indicates the relative difference between T2F.II compared to initial model's T2F.II (more positive is better).

**Relative difference between overall RMSE compared to initial model's overall RMSE (more negative is better).**
5.3 Experiments using Time Series Forecasting

The entire flight (and hence also the approach) of an aircraft is monitored by radar and by the aircraft itself. Every x seconds latitude, longitude, height and other factors of the aircraft are recorded. This periodic (at equal intervals in time) capturing of data can also be referenced to as a Time Series. By analyzing history in a Time Series, future can be forecasted, under the assumption that historic patterns will continue into the future [Box et al., 2015]. For this research it means that analyzing a certain trajectory of a flight can provide insights in the approach of the aircraft. However, not any data is suitable for this type of analysis.

There are two major components in a Time Series: trend and seasonal variation (see Figure 5.3 for an example of air passengers data).
Table 5.8: Random Forest Regressor results Ground Speed 10NM-0NM

<table>
<thead>
<tr>
<th>AC</th>
<th>Features</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>GS.1</th>
<th>Var</th>
<th>Bias²</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>ALW</td>
<td>313.87</td>
<td>549.74</td>
<td>17.72</td>
<td>24.24%</td>
<td>39.24</td>
<td>275.37</td>
</tr>
<tr>
<td>B738</td>
<td>ALW+gspass</td>
<td>166.72</td>
<td>451.77</td>
<td>12.91</td>
<td>-9.45%</td>
<td>8.55</td>
<td>157.95</td>
</tr>
<tr>
<td>B738</td>
<td>HW</td>
<td>136.90</td>
<td>339.33</td>
<td>11.70</td>
<td>-17.95%</td>
<td>5.30</td>
<td>132.78</td>
</tr>
<tr>
<td>B738</td>
<td>HW+gspass</td>
<td>116.22</td>
<td>263.81</td>
<td>10.78</td>
<td>-24.40%</td>
<td>6.33</td>
<td>112.42</td>
</tr>
<tr>
<td>B738</td>
<td>All-ALW</td>
<td>93.90</td>
<td>281.77</td>
<td>9.69</td>
<td>-32.05%</td>
<td>5.30</td>
<td>89.17</td>
</tr>
<tr>
<td>B738</td>
<td>All</td>
<td>92.38</td>
<td>269.95</td>
<td>9.61</td>
<td>-32.60%</td>
<td>6.07</td>
<td>87.43</td>
</tr>
<tr>
<td>A332</td>
<td>ALW</td>
<td>275.66</td>
<td>380.77</td>
<td>16.60</td>
<td>41.39%</td>
<td>41.43</td>
<td>234.00</td>
</tr>
<tr>
<td>A332</td>
<td>ALW+gspass</td>
<td>170.63</td>
<td>253.64</td>
<td>13.06</td>
<td>11.24%</td>
<td>10.51</td>
<td>160.30</td>
</tr>
<tr>
<td>A332</td>
<td>HW</td>
<td>125.94</td>
<td>208.56</td>
<td>11.22</td>
<td>-4.43%</td>
<td>10.64</td>
<td>109.99</td>
</tr>
<tr>
<td>A332</td>
<td>HW+gspass</td>
<td>110.24</td>
<td>190.23</td>
<td>10.50</td>
<td>-10.59%</td>
<td>6.55</td>
<td>99.51</td>
</tr>
<tr>
<td>A332</td>
<td>All-ALW</td>
<td>84.81</td>
<td>139.08</td>
<td>9.21</td>
<td>-21.58%</td>
<td>6.28</td>
<td>77.94</td>
</tr>
<tr>
<td>A332</td>
<td>All</td>
<td>85.38</td>
<td>145.68</td>
<td>9.24</td>
<td>-21.31%</td>
<td>6.50</td>
<td>78.20</td>
</tr>
</tbody>
</table>

Figure 5.3: Time Series decomposition

The trend forms the smooth long-term direction of a time-series, in the case of the air passengers an upward growth. Seasonal variation is a periodic returning pattern in the data (the peaks of air passengers during summer months, dips after the summer till the end of a year).

Next to trend and seasonal variation, also other cyclic variation (cyclic variation at periods other than one year) and irregular fluctuations (what is left of the data after removing trend and other variations) can be present in Time Series data. Appendix C contains more information on stationarity of Time Series and the ARIMA model, as used in this experiment.

5.3.1 Data

Since time series forecasting utilizes the actual time series for the forecasting, only the data fields as noted in the following two tables (Table 5.9 and 5.10) are used. Data was available on
the trajectory from 19.5 to 1 Nautical Mile from runway threshold, lacking the values for the last mile before threshold. In order to approximate the time to fly and ground speed for the last 2 passing gates (0.5NM and runway threshold, 0NM), $t_{2f}$ and $g_{\text{pass}}$ on the last 4 known gates is averaged.

Table 5.9: Data inputs Time Forecasting

<table>
<thead>
<tr>
<th>Field</th>
<th>Data Description</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{2f}$</td>
<td>T2F from a passing gate to the next [19.5:-0.5:10.0] NM before threshold</td>
<td>s (seconds)</td>
</tr>
<tr>
<td>$g_{\text{pass}}$</td>
<td>Ground speed at passing gates [19.5:-0.5:10.0] NM before threshold</td>
<td>kts (knots)</td>
</tr>
</tbody>
</table>

Table 5.10: Data targets Time Forecasting

<table>
<thead>
<tr>
<th>Field</th>
<th>Data Description</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{2f}$</td>
<td>T2F from a passing gate to the next [9.5:-0.5:0.0] NM before threshold</td>
<td>s (seconds)</td>
</tr>
<tr>
<td>$g_{\text{pass}}$</td>
<td>Ground speed at passing gates [9.5:-0.5:0.0] NM before threshold</td>
<td>kts (knots)</td>
</tr>
</tbody>
</table>

For the experiment data of an Airbus A330-200 (2632 flights) and Boeing B737-800 (4587 flights) are used. In order to get an idea of the data distribution box-plots for the T2F and Ground Speed of the B738 flights have been created, see Figure 5.4 and 5.5.

Figure 5.4: Box-plot Time to Fly B738

In general both plots show a trend in time (getting closer to the runway threshold), but the series in general are non-stationary due to the increasing slope. Also variance and covariance seem to be non-stationary. How this effects the forecasts will be discussed in the following section.
5.3.2 Experiment setup

Since R offers many high-quality Time Series forecasting libraries, a custom R script was used to forecast T2F and Ground Speed. For any single flight the `auto.arima()` function was used to fit the best ARIMA model to the data of that particular flight. After selection of the optimal ARIMA model, a number of forecasts are made using the `forecast.Arima()` function in the forecast library. An advantage of this method is that, next to the forecast itself, it also returns the 80% and 90% lower/upper limits for the prediction intervals.

After forecasting the rest of the flight’s T2F/Ground Speed profile the error of a flight’s forecasts can be calculated as follows:

- **T2F error.** In order to calculate the T2F error of a flight the forecasts over the last 4NM/10NM of the flight are summed. The same is done for the actual flight’s data. By subtracting the actual T2F from the forecasted T2F the error is provided.

- **GS error.** Ground Speed error is calculated by determining the mean of forecasted ground speeds at gates in the last 4NM/10NM and subtract this from the mean over the same trajectory ground speeds in the actual data.

Note that the Root Mean Squared Error for all flight’s forecasts is used for evaluation purposes. Again, this is calculated by applying Equation 5.1 to all individual flight errors and take the root of the result.

Forecasts for 4NM-0NM are performed taking into account three trajectories (19.5NM - 10NM, 19.5NM - 9NM, 19.5NM - 8NM) as input in order to see whether getting closer to the runway.
also delivers better results. Forecasts for 10NM-0NM use information from respectively 19.5NM - 15NM, 19.5NM - 14NM and 19.5NM - 13NM.

5.3.3 Results

Figure 5.6 and 5.7 show an example forecast over the last 4 NM for T2F (prediction error: -11.53 seconds) respectively Ground Speed (RMSE: 24.31 knots). The blue line indicates the forecasts for each passing gate, whereas the blueish marked area is the 80% confidence interval and the grayish marked area represents the 95% interval. The x-axis represents the distance in NM from the runway threshold and the y-axis denotes the number of seconds it takes the aircraft to fly from one gate to the next (so traversing 0.5NM).

![Graph](image)

**Figure 5.6:** Example flight Time Series forecast T2F

Closer to the runway threshold (further in time) forecasts are harder to predict, which is showed by the wider confidence intervals to the right of the graphs.

To get a feeling of the RMSE distribution for the entire dataset Figure 5.8 has been created. Only 4% of all RMSEs are between 0 and 5. A very large part of the forecasts are poorly forecasted, having errors of over 20 seconds.
The following tables contain the results for forecasting Time to Fly (Table 5.11 and 5.12) and Ground Speed (Table 5.13 and 5.14) using the Time Series forecasting method. From the results it is clear that errors decrease when having more information available closer to runway threshold.

The values in cell $T2F,I$ (Table 5.11 only) depict the difference of the forecasted overall RMSE with respect to the RMSE using the static initial model. Clearly the initial model outperforms
this Time Series Forecasting method. See Tables 5.1-5.4 for the RMSEs of the initial model.

Cell \textit{T2F.II.vs} depicts the difference between the Time Series Forecasting model vs the initial model with regards to the percentage of absolute flight errors smaller than 5 seconds. Also on this performance criterion the initial model performs better. Cell \textit{T2F.III.vs} displays the relative difference compared to the initial model for flights having an (absolute) error bigger than 10 seconds.

5.3.4 Conclusions

Time Series forecasting using ARIMA is not an appropriate method to accurately predict T2F and Ground Speed profiles; not only the overall Root Mean Squared Error is bigger compared to the initial model, also variance of predictions is much wider. One possible reason for a high Root Mean Squared Error is the difference in behavior between the trajectory taken as input space and the to be forecasted trajectory.

Although it is not a suitable method, a couple of advantages of Time Series forecasting in the context of aircraft performance prediction are:

- **Only flight time stamps are needed.** No additional information (like weight and meteorological data) are required to make predictions.

- **No need to train.** There is no training involved using this method. Every flight is analyzed separately and forecasts are made on the fly, which makes it a very fast method without the need for expensive hardware for storage or making calculations.
Table 5.11: Time Series Forecasting results T2F 4NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Input MSE (19.5NM-10NM)</th>
<th>MSE (19.5NM-9NM)</th>
<th>MSE (19.5NM-8NM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>848.94% 18.4% 81.57%</td>
<td>332.97% 11.09% 32.08%</td>
<td>29.21% 12.20%</td>
</tr>
<tr>
<td>A332</td>
<td>1203.43% 31.22% 34.69%</td>
<td>365.44% 13.65% 27.18%</td>
<td>23.50% 13.65%</td>
</tr>
<tr>
<td>A332</td>
<td>324.81% 11.99% 24.80%</td>
<td>324.81% 11.99% 22.30%</td>
<td>18.91% 18.91%</td>
</tr>
</tbody>
</table>

B738: 4586 flights / A332: 2632 flights.
### Table 5.12: Time Series Forecasting results T2F 10NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Input</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>T2F.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>19.5NM-15NM</td>
<td>10577.47</td>
<td>72.06</td>
<td>102.85</td>
<td>553.10%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-14NM</td>
<td>9089.84</td>
<td>77.00</td>
<td>95.34</td>
<td>505.43%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-13NM</td>
<td>6392.23</td>
<td>70.68</td>
<td>79.95</td>
<td>407.71%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-15NM</td>
<td>10384.85</td>
<td>74.87</td>
<td>101.91</td>
<td>544.56%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-14NM</td>
<td>9049.00</td>
<td>81.17</td>
<td>95.13</td>
<td>501.68%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-13NM</td>
<td>6969.02</td>
<td>76.65</td>
<td>83.48</td>
<td>428.02%</td>
</tr>
</tbody>
</table>

### Table 5.13: Time Series Forecasting results Ground Speed 4NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Input</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>GS.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>19.5NM-10NM</td>
<td>3521.85</td>
<td>55.91</td>
<td>59.35</td>
<td>553.55%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-9NM</td>
<td>3234.09</td>
<td>54.24</td>
<td>56.87</td>
<td>526.28%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-8NM</td>
<td>1988.10</td>
<td>41.90</td>
<td>44.59</td>
<td>391.04%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-10NM</td>
<td>3612.89</td>
<td>56.62</td>
<td>60.11</td>
<td>690.07%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-9NM</td>
<td>2718.62</td>
<td>50.82</td>
<td>52.14</td>
<td>585.35%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-8NM</td>
<td>1903.11</td>
<td>40.42</td>
<td>43.62</td>
<td>473.42%</td>
</tr>
</tbody>
</table>

### Table 5.14: Time Series Forecasting results Ground Speed 10NM-0NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Input</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>GS.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>19.5NM-15NM</td>
<td>8428.92</td>
<td>84.79</td>
<td>91.81</td>
<td>543.84%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-14NM</td>
<td>7480.60</td>
<td>81.92</td>
<td>86.49</td>
<td>506.54%</td>
</tr>
<tr>
<td>B738</td>
<td>19.5NM-13NM</td>
<td>5785.18</td>
<td>72.91</td>
<td>76.06</td>
<td>433.40%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-15NM</td>
<td>10844.32</td>
<td>97.11</td>
<td>104.14</td>
<td>786.79%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-14NM</td>
<td>9129.48</td>
<td>89.72</td>
<td>95.55</td>
<td>713.66%</td>
</tr>
<tr>
<td>A332</td>
<td>19.5NM-13NM</td>
<td>6540.11</td>
<td>77.12</td>
<td>80.87</td>
<td>588.67%</td>
</tr>
</tbody>
</table>

### 5.4 Experiments using different algorithms

There are many supervised learners available within scikit-learn or other libraries, which might be applicable for the problem described in this thesis. Until now only a Random Forest Regressor has been evaluated. This section aims to evaluate other algorithms in that class of learners in order to minimize the Mean Squared Error. As with the previous experiments prediction for Time to Fly and Ground Speed profiles are conducted for the trajectories 4NM-0NM and 10NM-0NM. First, the experiment setup and chosen algorithms is described followed by the results of the experiments.
5.4.1 Experiment setup

In order to make a justified comparison with the experiment conducted using the RandomForestRegressor the same dataset and method (repeating bootstrap) is used. Since it is most likely that actual landing weight information will not be available in real-time situations, experiments have been performed without the availability of this information. This way the results of these experiments can be used in a real life support system. So, as input the entire feature space is used except actual landing weight.

For the experiment again the power of Python’s library scikit-learn has been utilized. There are a couple of reasons for choosing this library:

1. There are high demands for documentation and providing examples before somebody can contribute
2. The library has a stable number of expert contributors, including Machine Learning and software development experts.
3. Most of the Machine Learning tasks are covered. If a new technique is discovered, it will usually not take long for it to be included in scikit-learn.
4. Personally I already had a basis using Python, which is also the programming language used in sklearn.
5. The interface used is common for many algorithms. Many functions (like `train()` and `predict()`) are used within the entire set of supervised learners, making it fairly easy to evaluate other algorithms without rewriting the code base.

Used regressors

The following regressors (using the names as used in scikit-learn) have been selected from the supervised learning section:

- GaussianProcess [Rasmussen, 2006]
- AdaboostRegressor
- GradientBoostingRegressor
- BaggingRegressor

Besides these different algorithms also the already used RandomForestRegressor is optimized and included for comparison. As is the RandomForestRegressor, the latter three algorithms are also categorized as ensembles: ensemble methods are learning algorithms that construct a set of classifiers (or regressors in this case) and then predict new data points by taking a weighted vote of their predictions [Dietterich, 2000]. Two families of ensemble methods are usually distinguished:

---

8http://scikit-learn.org/stable/
9https://github.com/scikit-learn/scikit-learn/graphs/contributors
In **averaging methods**, the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced. The RandomForestRegressor and BaggingRegressor can be categorized as averaging methods.

By contrast, in **boosting methods**, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensemble. AdaboostRegressor and GradientBoostingRegressor are boosting methods.

AdaboostRegressor, GradientBoostingRegressor and BaggingRegressor are not able to predict more than one outcome variable. Using the results described in Section 5.2.2, in this experiment the single target variable T2F\(^{11}\) and AVG_GSPASS\(^{12}\) (for 4 NM as well as 10 NM) are predicted for all algorithms.

Gaussian processes is a very popular method these days. Based on an expert’s opinion this algorithm is included in the experiment to assess its performance.

### Optimizing hyperparameters

Each of the algorithms selected in the previous section have hyperparameters that should be optimized to obtain the lowest mean squared error and hence the best predictions. This section describes the followed procedure to find the optimal hyperparameters for each regressor.

By calling `estimator.get_params()`, the names and current values for all parameters for a given estimator are provided. There are multiple methods to obtain the most optimal parameter combination for an estimator (regressor in this case). Since the provided dataset is relatively small, an exhaustive search is performed for every estimator. All parameter combinations are considered using a so-called Grid Search. The grid search (provided by GridSearchCV in sklearn) exhaustively generates candidates from a grid of parameter values specified with the `param_grid` parameter. For instance, the following `param_grid`:

```python
param_grid = {'loss': ['linear'], 'n_estimators': [50, 100, 200, 400, 1000], 'learning_rate': [0.1, 0.4, 0.5, 0.7, 1.0]}
```

specifies that a grid should be explored using all cross products of a linear `loss` with `n_estimators` values in [50, 100, 200, 400, 1000] and `learning_rate` values in [0.1, 0.4, 0.5, 0.7, 1.0]. The combination with the lowest Mean Squared Error is returned as best result and will be used for evaluating the algorithm in this chapter. Appendix E.4 contains the Python code used to implement hyperparameter optimization.

For each algorithm the used parameter grid is provided in Table 5.15. The parameters are defined using either the only available options (e.g. `loss`: ['linear', 'square', 'exponential'] for AdaboostRegressor) or a range of possible values (e.g. `n_estimators`: [10, 100, 200, 400, 1000] for the BaggingRegressor), which are based on documentation and examples in scikit-learn.

\(^{11}\)Being the sum of `tdiff` values over the selected trajectory.

\(^{12}\)Being the average of `gspass` values over the selected trajectory.
Table 5.15: Hyperparameter grids for all algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter Grid</th>
</tr>
</thead>
</table>
| GaussianProcess         | theta0: np.logspace(-2,1)  
regr: ['constant', 'linear']  
corr: ['absolute_exponential', 'squared_exponential', 'cubic', 'linear'] |
| AdaboostRegressor       | loss: ['linear', 'square', 'exponential']  
n_estimators: [50, 100, 200, 400, 1000]  
learning_rate: [0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1] |
| GradientBoostingRegressor | loss: ['ls', 'lad', 'huber', 'quantile']  
n_estimators: [50, 100, 200, 400, 1000]  
learning_rate: [0.1, 0.3, 0.5, 0.7, 1]  
max_depth: [1,3,5,7,9,12]  
min_samples_split: [1,2,4,6,8,10]  
min_samples_leaf: [1,2,4,6,8,10]  
warm_start: [True, False] |
| BaggingRegressor        | n_estimators: [10,100,200,400,1000]  
bootstrap: [True, False]  
warm_start: [True, False] |
| RandomForestRegressor   | n_estimators: [16,32,64,128]  
max_features: ['auto', 'sqrt', 'log2']  
max_depth: [3, None]  
min_samples_split: [1,2,4,6,8,10]  
min_samples_leaf: [1,2,4,6,8,10]  
warm_start: [True, False] |

Tables D.1-D.5 display the top 5 hyperparameter combinations for all used algorithms.
5.4.2 Results

Similar to the results described in Section 5.2.2, tables 5.16-5.19 show a summary of errors, standard deviation of the MSE, squared Bias, variance and the scores on the evaluation criteria for B738 and A332 (as defined in Section 2.5.3). Note that only for T2F prediction on the last 4NM T2F.II and T2F.III is included.

The first table (5.16) summarizes the results for the trajectory of 4NM to runway threshold. From this table it can be noted that all algorithms score better than the initial method for all evaluation criteria T2F.I, T2F.II.vs and T2F.III.vs. BaggingRegressor (green cells) delivers best scores for any evaluation criterion and aircraft type, outperforming all other algorithms.
Table 5.16: Results all algorithms T2F 4NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Features</th>
<th>MSE</th>
<th>Vari</th>
<th>T2F.I</th>
<th>T2F.II</th>
<th>T2F.III</th>
<th>MSE%</th>
<th>Vari%</th>
<th>RMSE</th>
<th>T2F.I vs T2F.II</th>
<th>T2F.II vs T2F.III</th>
<th>T2F.I vs T2F.III</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>23.44</td>
<td>0.31</td>
<td>17.05</td>
<td>10.25</td>
<td>5.37</td>
<td>2.63</td>
<td>2.63</td>
<td>3.70</td>
<td>-21.66%</td>
<td>-12.33%</td>
<td>-19.16%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>26.26</td>
<td>1.03</td>
<td>9.22</td>
<td>4.88</td>
<td>2.23</td>
<td>4.37</td>
<td>4.37</td>
<td>5.16</td>
<td>-12.07%</td>
<td>-16.07%</td>
<td>-12.07%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>24.85</td>
<td>1.03</td>
<td>10.25</td>
<td>4.88</td>
<td>2.23</td>
<td>4.37</td>
<td>4.37</td>
<td>5.16</td>
<td>-21.66%</td>
<td>-12.33%</td>
<td>-19.16%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>9.22</td>
<td>1.03</td>
<td>10.25</td>
<td>4.88</td>
<td>2.23</td>
<td>4.37</td>
<td>4.37</td>
<td>5.16</td>
<td>-21.66%</td>
<td>-12.33%</td>
<td>-19.16%</td>
</tr>
</tbody>
</table>

Relative difference between T2F.III compared to initial model's T2F.II (more negative is better)

Percentage of flights having a RMSE < 5 seconds

Relative difference between T2F.II compared to initial model's T2F.II (more positive is better)

Percentage of flights having a RMSE > 10 seconds

Relative difference between overall RMSE compared to initial model's overall RMSE (more negative is better)
Table 5.17: Results all algorithms T2F 10NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>T2F.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>156.77</td>
<td>268.40</td>
<td>12.52</td>
<td>-20.49%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>194.39</td>
<td>340.75</td>
<td>13.94</td>
<td>-11.46%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>170.22</td>
<td>317.91</td>
<td>13.05</td>
<td>-17.15%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>70.66</td>
<td>204.51</td>
<td>8.41</td>
<td>-46.62%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>191.68</td>
<td>294.35</td>
<td>13.84</td>
<td>-12.43%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>199.42</td>
<td>310.23</td>
<td>14.12</td>
<td>-10.68%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>183.33</td>
<td>299.67</td>
<td>13.54</td>
<td>-14.36%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>74.01</td>
<td>182.27</td>
<td>8.60</td>
<td>-45.59%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>122.06</td>
<td>222.34</td>
<td>11.05</td>
<td>-30.12%</td>
</tr>
</tbody>
</table>

Table 5.18: Results all algorithms Ground Speed 4NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>GS.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>49.63</td>
<td>238.37</td>
<td>7.04</td>
<td>-22.42%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>58.40</td>
<td>244.06</td>
<td>7.64</td>
<td>-15.84%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>53.44</td>
<td>313.03</td>
<td>7.31</td>
<td>-19.50%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>20.24</td>
<td>68.00</td>
<td>4.50</td>
<td>-50.45%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>33.95</td>
<td>273.71</td>
<td>5.83</td>
<td>-35.84%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>44.24</td>
<td>74.43</td>
<td>6.65</td>
<td>-12.58%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>49.01</td>
<td>78.04</td>
<td>7.00</td>
<td>-7.98%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>45.20</td>
<td>82.76</td>
<td>6.72</td>
<td>-11.63%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>19.23</td>
<td>50.78</td>
<td>4.38</td>
<td>-42.36%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>29.52</td>
<td>63.28</td>
<td>5.43</td>
<td>-28.59%</td>
</tr>
</tbody>
</table>

Table 5.19: Results all algorithms Ground Speed 10NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>MSE</th>
<th>Std MSE</th>
<th>RMSE</th>
<th>GS.I</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>87.33</td>
<td>185.85</td>
<td>9.35</td>
<td>-34.47%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>107.62</td>
<td>232.88</td>
<td>10.37</td>
<td>-27.25%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>95.59</td>
<td>247.17</td>
<td>9.78</td>
<td>-31.43%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>34.94</td>
<td>92.26</td>
<td>5.91</td>
<td>-58.55%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>60.70</td>
<td>168.13</td>
<td>7.79</td>
<td>-45.36%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>83.18</td>
<td>125.46</td>
<td>9.12</td>
<td>-22.33%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>88.73</td>
<td>138.67</td>
<td>9.42</td>
<td>-19.79%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>83.30</td>
<td>140.92</td>
<td>9.13</td>
<td>-22.28%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>33.89</td>
<td>84.10</td>
<td>5.82</td>
<td>-50.43%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>55.54</td>
<td>104.94</td>
<td>7.45</td>
<td>-36.54%</td>
</tr>
</tbody>
</table>
Based on individual flight errors of all B738 flights in the dataset, a box plot has been drafted for each algorithm, see Figure 5.9. The initial method is the box-plot on the left. Also here it is clear that the BaggingRegressor predicts T2F over the last 4NM with the tightest bounds, making it the best algorithm for this problem of all tested algorithms. Figure 5.10 gives a presentation of the Root Mean Squared Errors of all algorithms for T2F predictions over the last 4 Nautical Miles.

![Figure 5.9: Distribution of flight errors T2F 4NM (B738)](image)

![Figure 5.10: RMSE's all algorithms T2F 4NM (B738)](image)
In Figure 5.11 the error distribution is displayed for the best performing algorithm in order to give some insights in over- and underprediction. The majority of prediction errors are centered around 0 seconds. As can be derived from the plot, there are more flights overpredicted than underpredicted, although differences are not shocking.

![Error Distribution BaggingRegressor (B738 T2F 4NM)](image)

**Figure 5.11:** Error Distribution BaggingRegressor (B738 T2F 4NM)

### 5.5 Residual analysis

As is described in Section 2.5.1, there are certain risks involved when predictions are not accurate. Since this research focuses primarily on T2F predictions over the last 4NM for measuring evaluation criteria, predictions for individual flights are analyzed in further detail in this section in order to identify possible causes for wrong predictions. First, a short analysis is conducted to check what flights are poorly predicted. Are these flights not accurately predicted for all algorithms or is this algorithm specific? Next, possible causes for the less accurate predicted flights are aimed to find.

#### 5.5.1 Less accurate predicted flights

As defined in Section 2.5, a flight’s T2F over the last 4NM can become a risk if the predicted time to fly differs more than 10 seconds compared to its actual time to fly. Table 5.20 displays all (absolute) flight errors of the B738 dataset having a RMSE greater than 13 seconds; the list is sorted by the BaggingRegressor’s errors since this was the best performing algorithm. In order to keep the list clear only flights having an RMSE greater than 13 seconds are displayed. For analysis purposes later in this chapter all errors above 10 seconds are taken into account.
What can be concluded from the table above is that there is a common trend in flight errors. It is not algorithm specific to predict a flight’s T2F less accurate, it happens for any algorithm (at least the ones used in this thesis). Even for the initial model the prediction error is very high. A conclusion that might be drawn from this observation is that there is something ‘strange’ in the flight’s profile itself that leads to a false prediction. The next section will zoom in on this aspect more.

### 5.5.2 Analysis of less accurate predicted flights

Since there might be a relation between flights being predicted less accurate\(^{13}\), this section aims to identify possible causes why the same flights are poorly predicted by all tested algorithms.

Since headwind has a strong correlation with Ground Speed (as identified in Section 3.3.6) there might be a relation as well with poorly predicted flights. Besides headwind there might be a relation between T2F of the trajectory 20NM-15NM and the Final Approach time to fly. Figure 5.12 displays scatter plots of the errors >10s for the combinations headwind/T2F 20NM-15NM versus T2F 20NM-10NM, the last 10 NM and the last 4 NM. In the plots a larger circle indicates a larger error (either negative or positive). Red circles indicate a negative error and blue circles a positive error.

---

\(^{13}\) Considering predicted flights with an error >10 seconds for T2F on the last 4 Nautical Miles, as defined in the evaluation criteria.
In the center plots (using T2F over the last 4 NM) it seems that aircraft flying either very fast or very slow over the last 4 NM result in the largest errors. Figure 5.13 has been added to indicate the distribution of T2F over the last 4 NM. Headwind does not seem to have a relation with the error distribution.
5.6 Research questions review

In this chapter research question 5 has been answered:

**How does each of the used ML techniques perform with respect to the defined performance criteria?**

The following tables (Table 5.21-5.24) show a summary of results for all algorithms for T2F and Ground Speed predictions over the last 4 NM and 10 NM. The RMSE of the initial model is included in these tables for ease of comparison. All features are taken as input except actual landing weight, since this is not available in real-time situations.

From the tables it is clear that all selected algorithms perform better than the initial (static) model; BaggingRegressor performs best in all scenarios.

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>RMSE IM$^a$</th>
<th>RMSE</th>
<th>T2F.I$^b$</th>
<th>T2F.II vs$^c$</th>
<th>T2F.III vs$^d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>5.84</td>
<td>4.84</td>
<td>-17.08%</td>
<td>12.89%</td>
<td>-41.13%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>5.84</td>
<td>5.12</td>
<td>-12.23%</td>
<td>8.76%</td>
<td>-27.44%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>5.84</td>
<td>4.98</td>
<td>-14.63%</td>
<td>12.94%</td>
<td>-31.02%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>5.84</td>
<td>3.04</td>
<td>-47.99%</td>
<td>39.17%</td>
<td>-82.17%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>5.84</td>
<td>3.99</td>
<td>-31.69%</td>
<td>27.36%</td>
<td>-63.42%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>5.93</td>
<td>5.01</td>
<td>-15.44%</td>
<td>11.54%</td>
<td>-42.93%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>5.93</td>
<td>5.27</td>
<td>-11.13%</td>
<td>11.01%</td>
<td>-34.86%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>5.93</td>
<td>5.05</td>
<td>-14.78%</td>
<td>12.87%</td>
<td>-44.83%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>5.93</td>
<td>3.20</td>
<td>-45.98%</td>
<td>43.05%</td>
<td>-80.05%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>5.93</td>
<td>4.13</td>
<td>-30.33%</td>
<td>29.51%</td>
<td>-67.52%</td>
</tr>
</tbody>
</table>

$^a$Initial Model
$^b$Relative difference between overall RMSE compared to initial model’s overall RMSE (more negative is better)
$^c$Relative difference between T2F.II (Percentage of flights having a T2F RMSE less than 5 seconds.) compared to initial model’s T2F.II. More positive is better.
$^d$Relative difference between T2F.III (Percentage of flights having a T2F RMSE bigger than 10 seconds.) compared to initial model’s T2F.III. More negative is better.
### Table 5.22: Summarized results all algorithms T2F 10NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>RMSE IM</th>
<th>RMSE</th>
<th>T2F.I&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>15.75</td>
<td>12.52</td>
<td>-20.49%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>15.75</td>
<td>13.94</td>
<td>-11.46%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>15.75</td>
<td>13.05</td>
<td>-17.15%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>15.75</td>
<td>8.41</td>
<td>-46.62%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>15.75</td>
<td>10.41</td>
<td>-33.88%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>15.81</td>
<td>13.84</td>
<td>-12.43%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>15.81</td>
<td>14.12</td>
<td>-10.68%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>15.81</td>
<td>13.54</td>
<td>-14.36%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>15.81</td>
<td>8.60</td>
<td>-45.59%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>15.81</td>
<td>11.05</td>
<td>-30.12%</td>
</tr>
</tbody>
</table>

<sup>a</sup>Relative difference between overall RMSE compared to initial model’s overall RMSE (more negative is better)

### Table 5.23: Summarized results all algorithms Ground Speed 4NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>RMSE IM</th>
<th>RMSE</th>
<th>GS.I&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>9.08</td>
<td>7.04</td>
<td>-22.42%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>9.08</td>
<td>7.64</td>
<td>-15.84%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>9.08</td>
<td>7.31</td>
<td>-19.50%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>9.08</td>
<td>4.50</td>
<td>-50.45%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>9.08</td>
<td>5.83</td>
<td>-35.84%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>7.61</td>
<td>6.65</td>
<td>-12.58%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>7.61</td>
<td>7.00</td>
<td>-7.98%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>7.61</td>
<td>6.72</td>
<td>-11.63%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>7.61</td>
<td>4.38</td>
<td>-42.36%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>7.61</td>
<td>5.43</td>
<td>-28.59%</td>
</tr>
</tbody>
</table>

<sup>a</sup>Relative difference between overall RMSE compared to initial model’s overall RMSE (more negative is better)

### Table 5.24: Summarized results all algorithms Ground Speed 10NM

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Algorithm</th>
<th>RMSE IM</th>
<th>RMSE</th>
<th>GS.I&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>GaussianProcess</td>
<td>14.26</td>
<td>9.35</td>
<td>-34.47%</td>
</tr>
<tr>
<td>B738</td>
<td>AdaboostRegressor</td>
<td>14.26</td>
<td>10.37</td>
<td>-27.25%</td>
</tr>
<tr>
<td>B738</td>
<td>GradientBoostingRegressor</td>
<td>14.26</td>
<td>9.78</td>
<td>-31.43%</td>
</tr>
<tr>
<td>B738</td>
<td>BaggingRegressor</td>
<td>14.26</td>
<td>5.91</td>
<td>-58.55%</td>
</tr>
<tr>
<td>B738</td>
<td>RandomForestRegressor</td>
<td>14.26</td>
<td>7.79</td>
<td>-45.36%</td>
</tr>
<tr>
<td>A332</td>
<td>GaussianProcess</td>
<td>11.74</td>
<td>9.12</td>
<td>-22.33%</td>
</tr>
<tr>
<td>A332</td>
<td>AdaboostRegressor</td>
<td>11.74</td>
<td>9.42</td>
<td>-19.79%</td>
</tr>
<tr>
<td>A332</td>
<td>GradientBoostingRegressor</td>
<td>11.74</td>
<td>9.13</td>
<td>-22.28%</td>
</tr>
<tr>
<td>A332</td>
<td>BaggingRegressor</td>
<td>11.74</td>
<td>5.82</td>
<td>-50.43%</td>
</tr>
<tr>
<td>A332</td>
<td>RandomForestRegressor</td>
<td>11.74</td>
<td>7.45</td>
<td>-36.54%</td>
</tr>
</tbody>
</table>
Part III

Concluding remarks
Conclusions

In the beginning of this research a main research question has been defined:

*Given an aircraft about to enter its final approach, what is its predicted speed and time to fly (T2F) profile from 4 Nautical Miles to runway threshold?*

In order to formulate an answer to this main research question five sub questions have been derived. The answers to these questions have, combined, led to a targeted manner in predicting T2F (and also Ground Speed) over the last 4 NM using several Machine Learning algorithms.

Below the five sub research questions have been stated and provided with a summarized answer. The results of this research are compared to results using a static (referred to as initial) model in which an aircraft type has a fixed speed at a fixed distance before runway threshold.

1. **What are appropriate performance criteria to assess possible solutions in order to secure that risks of inaccurate predictions are acceptable?**
   
   Baseline for the performance criteria is optimizing separation accuracy (with the objective to optimize runway throughput while maintaining safety).
   
   In general the overall prediction Root Mean Squared Error is calculated and compared to the RMSE of the same dataset using the initial model.

   For T2F predictions over the last 4 NM also the following percentages are calculated and compared to the initial model:
   
   - T2F RMSE less than 5 seconds
   - T2F RMSE larger than 10 seconds

   Whereas the first mentioned percentage is important for performance of the predictor, the latter one is very important safety-wise. Having an absolute prediction error for a flight of over 10 seconds can result in a Wake Vortex Encounter and a "Go-Around procedure".

2. **What are the major factors influencing the speed profile of an aircraft during its final approach?**

   The major factors influencing the Final Approach Speed are: aircraft type, actual landing weight of the aircraft, flap settings during final approach, wind conditions and the assigned
runway. Unfortunately, flap setting information was not available. In real-time situations there is no information on actual landing weight and flap settings.

3. What is the influence of the identified major factors on the speed profile of an aircraft during its final approach?

One runway in the dataset appeared to have an influence on the speed profile of an aircraft. For further analysis flights landing on that runway were removed from the dataset. Correlation between headwind and speed profile is stronger than correlation between actual landing weight and speed profile.

Variation in the data is high. This might be an indicator that there are other forces having a huge impact on final approach speed besides the major factors analyzed. These forces may be found in the factors of which no data is available (like flap settings and assigned gate), but also the role of a pilot flying an aircraft can play a serious role which can not be quantified using this data.

4. What Machine Learning techniques can be used to make predictions on the speed and time to fly profile of an aircraft during final approach?

Research in the aircraft speed prediction domain as well as research in a similar domain as traveling times for buses has provided insights in applicable algorithms that deliver proper results. Based on that research and gaps that haven’t been filled yet a choice has been made to focus on Ensembles, Time Series forecasting and Gaussian Processes.

5. How does each of the used ML techniques perform with respect to the defined performance criteria?

Time Series forecasting using ARIMA predicted T2F profiles significant worse than the initial model.

From the Ensembles category AdaboostRegressor, GradientBoostingRegressor, BaggingRegressor and RandomForestRegressor have been selected. All of these regressors, as well as GaussianProcesses outperformed the initial model, whereas the BaggingRegressor performed best in all scenarios. Table 6.1 shows the results on T2F predictions over the last 4 NM using the BaggingRegressor.

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>RMSE IM</th>
<th>RMSE</th>
<th>T2F.I</th>
<th>T2F.II</th>
<th>T2F.II vs</th>
<th>T2F.III</th>
<th>T2F.III vs</th>
</tr>
</thead>
<tbody>
<tr>
<td>B738</td>
<td>5.84</td>
<td>3.04</td>
<td>-47.99%</td>
<td>90.08%</td>
<td>-47.99%</td>
<td>90.08%</td>
<td>-82.17%</td>
</tr>
<tr>
<td>A332</td>
<td>5.93</td>
<td>3.20</td>
<td>-45.98%</td>
<td>88.52%</td>
<td>-47.99%</td>
<td>90.08%</td>
<td>-80.05%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Model</td>
<td>Relative difference between overall RMSE compared to initial model’s overall RMSE (more negative is better)</td>
<td>Percentage of flights having a RMSE &lt; 5 seconds</td>
<td>Relative difference between T2F.II compared to initial model’s T2F.II (more positive is better)</td>
<td>Percentage of flights having a RMSE &gt; 10 seconds</td>
<td>Relative difference between T2F.III compared to initial model’s T2F.III (more negative is better)</td>
</tr>
</tbody>
</table>

As can be seen by the green cells there are significant improvements on all performance criteria compared to the initial model. Also, from cells in column T2F.III, a little over 1% of all flights are having an absolute prediction error of 10 seconds of higher, which could lead to separation infringements. A brief analysis on those affected flights showed that it did not matter what
algorithm was used, they were always inaccurate predicted. It seemed that the fastest or slowest aircraft caused the highest prediction error, which could indicate a warning for an Air Traffic Controller to pay extra attention while there is still time.

Analysis has been conducted on a dataset of flights landing at Schiphol Airport (Amsterdam, the Netherlands) using aircraft types Boeing 737-800 and Airbus A332-200, two of the most common aircraft types having distinct physical characteristics. As input variables the entire feature space has been used except actual landing weight, since this information is not available in real-time scenarios and would give an too optimistic result.

The approach given in this thesis shows the potential for using Machine Learning techniques for predicting T2F profiles for an aircraft in final approach.
7.1 Recommendations

The research conducted in this thesis is relying heavily on the quality of meteorological data. As stated in Section 3.3.7, provided METAR data does not contain information on various flight levels. It ‘only’ contains data on ground level with a refresh rate of 30 minutes. It would be worth investigating whether connecting more accurate METAR information on various flight levels during final approach would result in more accurate results using the exact same methods as proposed in this thesis.

A logical extension of the research would be to use data from flights at other airports, in stead of only Schiphol Airport. Taking other aircraft types into account can help validate the results of the proposed methods, as well as using flights from other years.

During the experiments GaussianProcess of scikit-learn 0.17 has been used. Since scikit-learn v0.18 has been released a new (more complex) version of the Gaussian Processes implementation has been introduced, based on the work by [Rasmussen, 2006]. This new method is called GaussianProcessRegressor and might be worth an investigation on its performance.

7.2 Considerations for deployment

Deployment of a Machine Learning model can be a project on its own, having all kinds of different challenges, technical and non-technical. In this case safety of people is involved, making it even more challenging.

Whether the to-be deployed system is used as a decision support system for the Air Traffic Controllers or an automated decision making system will be important for the implementation. In both systems the ATCo remains the responsible person for aviation safety, however there is a difference in interpreting the outcome of the system; for a decision support system the ATCo will be advised with certain metrics (speed, height, etc) to order the pilot of an aircraft so there
is an optimal separation buffer at runway threshold. Emphasis here is on *advised* since it is a support system, the ATCo still decides what to do with the information. For an automated decision making system a scenario might be that the pilot of an aircraft will be automatically provided with instructions in order to optimize the separation buffer. In this case the ATCo only handles whenever safety borders are touched or crossed.

Compared to the initial model, results of this research indicate a significant improvement in terms of T2F / Ground Speed prediction making the used techniques more suitable for deployment in live systems. On the other hand, in the outcome of this thesis a little over 1% of all flights have an absolute prediction error of over 10 seconds, indicating possible safety risks during final approach. It would be recommended to conduct more research to either lower the T2F prediction error rate >10 seconds, or make it detectable when a flight will be predicted with a large error before it passes the Final Approach Point in a TMA (see Section 2.2.2).

At the moment there is no justifiable answer available what an acceptable percentage of T2F prediction error >10 seconds of all flights is; more research within NLR / Eurocontrol would be advised in order to make accountable and weighted decisions on the quality of a to be deployed system in the future.

For all experiments performed in Section 5.4 data has been used which is already available and will stay available in the future. This has been done for the sole purpose of practicality; results of this research can be applied directly in an application.
Part IV

Appendices
CRISP-DM Methodology

The CRISP-DM methodology was developed in March 1999 by the CRISP-DM consortium and is still very popular among data miners. It is used as a reference model for data analysts and helps avoid overlooking important steps in the data analysis process. CRISP-DM consists of six phases (see Figure A.1):

1. **Business Understanding**: first, the project’s objectives and requirements are (quantitatively) formulated from a business perspective.

   In this thesis the business understanding step is used to get a better understanding of the actual problem of predicting aircraft performance during final approach. In this phase formulating the appropriate performance criteria is addressed.

2. **Data Understanding**: this phase starts with an initial data collection and is aimed to get more insights on the data.

   The chapter on data exploration is devoted to the data understanding phase, as well as the data preparation phase. What are the relevant factors for predicting speed and time to fly profiles and what is their influence? Also analyzing the distribution of flights in the dataset is covered here.

3. **Data Preparation** is about constructing the final dataset(s) from the initial raw data. During the course of the research datasets have been altered multiple times in order to keep most important features and eliminate features that hamper data mining methods.

4. **Modeling**: in this phase, modeling techniques are selected and applied. If applicable, modeling parameters are calibrated to optimal values. Also the experiment setup is defined. Since some techniques have specific requirements on the form of data, going back to data preparation can be necessary.

---

1 The consortium included NCR Systems Engineering Copenhagen (USA and Denmark), DaimlerChrysler AG (Germany), Integral Solutions – later SPSS Inc. (USA, and OHRA Verzekeringen en Bank Groep B.V. (The Netherlands)

The modeling phase is covered throughout a large part of this thesis, since it includes not only selecting the appropriate Machine Learning techniques but also using and optimizing these techniques to get the best results possible.

5. **Evaluation**: at this stage the business objectives are evaluated against the experiment results. If one or more business objectives are not (sufficiently) covered, it is necessary to adjust the project.

Using the defined performance criteria, three distinct methods (Ensembles, Time Series Forecasting and Gaussian Processes) are evaluated for accuracy and safety considerations.

6. **Deployment**: in the final phase, the model is deployed and will be used by the end user. The tasks for maintenance and monitoring need to be defined in consultation with the end user.

Deployment is not within scope of this research, however considerations for deploying the most optimal model are given in this thesis.

![Figure A.1: CRISP-DM Methodology](Image)
Errors in predictive modeling

Errors in predictive models can be decomposed into three main components: Bias, Variance and Irreducible error (due to noise). The Bias error is used to measure how much on average predicted values deviate from actual values. A high bias error means the model keeps on missing important trends (underfitting). Variance on the other side measures how are the predictions made on same observation different from each other. A high variance model will overfit on the training population and perform badly on any observation beyond training. There is a trade-off between a model’s ability to minimize bias and variance. Figure B.1 shows a visualization of Bias and Variance with respect to predictive models. In this figure the center of the target is a model that perfectly predicts the correct values. Moving away from the center predictions get worse.

*Figure B.1: Bias-Variance trade-off*
In mathematical terms: if we denote the variable we are trying to predict as $Y$ and the feature space as $X$, we may assume that there is a relationship between them being $Y = f(X) + \epsilon$. In this relationship $\epsilon$ represents the error, which is normally distributed with a mean of zero, so $\epsilon \sim N(0, \sigma_\epsilon)$.

By predicting a model $\hat{f}(X)$ of $f(X)$, the expected squared prediction error at a point $x$ is:

$$Err(x) = E[(Y - \hat{f}(x))^2] \quad (B.1)$$

This error can be decomposed into three components:

$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right] + \sigma_\epsilon^2 \quad (B.2)$$

$$Err(x) = Bias^2 + Variance + Irreducible\ Error$$

The Irreducible Error term is the noise that cannot be reduced by any model.
Time Series Forecasting

Stationarity of a Time Series

If future values can be predicted exactly from past values, then a series is deterministic. However, in most cases, this does not hold. Most series are stochastic: the future is only partly determined by the past. If an appropriate model for this random behavior can be found, then the model should enable good forecasts to be computed.

A requirement for building a Time Series model is stationarity. This means that the properties of the underlying model do not change through time. There are three basic criteria for a series to be classified as stationary series:

1. The mean of the series should not be a function of time rather should be a constant. In the image below (Figure C.1) the left hand graph satisfies this condition whereas the graph in red has a time dependent mean.

2. The variance of the series should not be a function of time. Figure C.2 depicts what is (left graph) and what is not a stationary series.

![Figure C.1: Stationary Time Series - Mean](image.png)
3. The covariance of the i-th term and the (i + m)-th term should not be a function of time. In Figure C.3 the spread becomes tighter as the time increases. Hence, the covariance is not constant with time for the ‘red series’.

By running a Dickey-Fuller Test stationarity can be validated. In case a time series is non-stationary, the first requisite is to make it stationary. There are a couple of techniques to achieve this and ARIMA is one of those.

**AR(I)MA Time Series Modeling**

AR(I)MA models are commonly used in time series modeling. In an ARIMA model, AR stands for autoregression; autoregression is a stochastic process used in statistical calculations in which future values are estimated based on a weighted sum of past values. An autoregressive process operates under the premise that past values have an effect on current values. A process considered AR(1) is the first order process, meaning that the current value is based on the immediately preceding value. An AR(2) process has the current value based on the previous two values.
The I stands for integrated, a differencing step which can be applied to reduce the non-stationarity.

Rather than use past values of the forecast variable in a regression, a moving average (MA) model uses past forecast errors in a regression-like model.

Combining differencing with autoregression and a moving average model results in a non-seasonal ARIMA \((p,d,q)\) model, where

- \(p\) = order of the autoregressive part;
- \(d\) = degree of first differencing involved;
- \(q\) = order of the moving average part.

Determining the values for \(p\), \(d\) and \(q\) can be done manually, but also automatically using the `auto.arima()` function in R. The actual implementation for forecasting time to fly and speed profiles is described further in Section 5.3.
Hyperparameter combinations

This appendix contains the top 5 hyperparameter combinations for all used algorithms, for reference purposes.

Table D.1: Top 5 hyperparameter combinations GaussianProcess

<table>
<thead>
<tr>
<th>mean</th>
<th>corr</th>
<th>regr</th>
<th>theta0</th>
</tr>
</thead>
<tbody>
<tr>
<td>28.56</td>
<td>absolute_exponential</td>
<td>linear</td>
<td>0.193069773</td>
</tr>
<tr>
<td>28.86</td>
<td>absolute_exponential</td>
<td>linear</td>
<td>0.167683294</td>
</tr>
<tr>
<td>28.93</td>
<td>squared_exponential</td>
<td>linear</td>
<td>1.59985872</td>
</tr>
<tr>
<td>28.95</td>
<td>linear</td>
<td>linear</td>
<td>4.941713361</td>
</tr>
<tr>
<td>28.95</td>
<td>linear</td>
<td>linear</td>
<td>8.685113738</td>
</tr>
</tbody>
</table>

Table D.2: Top 5 hyperparameter combinations AdaBoostRegressor

<table>
<thead>
<tr>
<th>mean</th>
<th>learning_rate</th>
<th>loss</th>
<th>n_estimators</th>
</tr>
</thead>
<tbody>
<tr>
<td>30.22</td>
<td>0.2</td>
<td>exponential</td>
<td>100</td>
</tr>
<tr>
<td>30.30</td>
<td>0.1</td>
<td>exponential</td>
<td>200</td>
</tr>
<tr>
<td>30.33</td>
<td>0.3</td>
<td>exponential</td>
<td>100</td>
</tr>
<tr>
<td>30.33</td>
<td>0.4</td>
<td>exponential</td>
<td>100</td>
</tr>
<tr>
<td>30.34</td>
<td>0.4</td>
<td>exponential</td>
<td>50</td>
</tr>
</tbody>
</table>
Table D.3: Top 5 hyperparameter combinations BaggingRegressor

<table>
<thead>
<tr>
<th>mean</th>
<th>bootstrap</th>
<th>n_estimators</th>
<th>warm_start</th>
</tr>
</thead>
<tbody>
<tr>
<td>28.69</td>
<td>TRUE</td>
<td>100</td>
<td>TRUE</td>
</tr>
<tr>
<td>28.69</td>
<td>TRUE</td>
<td>100</td>
<td>FALSE</td>
</tr>
<tr>
<td>28.77</td>
<td>TRUE</td>
<td>200</td>
<td>TRUE</td>
</tr>
<tr>
<td>28.77</td>
<td>TRUE</td>
<td>200</td>
<td>FALSE</td>
</tr>
<tr>
<td>28.85</td>
<td>TRUE</td>
<td>1000</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

Table D.4: Top 5 hyperparameter combinations GradientBoostingRegressor

<table>
<thead>
<tr>
<th>mean</th>
<th>lr&lt;sup&gt;a&lt;/sup&gt;</th>
<th>loss</th>
<th>max_depth</th>
<th>msl&lt;sup&gt;b&lt;/sup&gt;</th>
<th>mss&lt;sup&gt;c&lt;/sup&gt;</th>
<th>n_estimators</th>
<th>warm_start</th>
</tr>
</thead>
<tbody>
<tr>
<td>28.80</td>
<td>0.1</td>
<td>lad</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>400</td>
<td>TRUE</td>
</tr>
<tr>
<td>28.93</td>
<td>0.1</td>
<td>lad</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>400</td>
<td>TRUE</td>
</tr>
<tr>
<td>28.94</td>
<td>0.1</td>
<td>ls</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>1000</td>
<td>TRUE</td>
</tr>
<tr>
<td>29.12</td>
<td>0.7</td>
<td>lad</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td>FALSE</td>
</tr>
<tr>
<td>29.15</td>
<td>0.1</td>
<td>lad</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>200</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

<sup>a</sup>Learning rate  
<sup>b</sup>min_sample_leaf  
<sup>c</sup>min_sample_split

Table D.5: Top 5 hyperparameter combinations RandomForestRegressor

<table>
<thead>
<tr>
<th>mean</th>
<th>max_depth</th>
<th>max_feat</th>
<th>min_sample_leaf</th>
<th>min_sample_split</th>
<th>warm_start</th>
</tr>
</thead>
<tbody>
<tr>
<td>32.90</td>
<td>3</td>
<td>auto</td>
<td>1</td>
<td>1</td>
<td>TRUE</td>
</tr>
<tr>
<td>32.90</td>
<td>3</td>
<td>auto</td>
<td>1</td>
<td>1</td>
<td>FALSE</td>
</tr>
<tr>
<td>32.90</td>
<td>3</td>
<td>auto</td>
<td>1</td>
<td>2</td>
<td>TRUE</td>
</tr>
<tr>
<td>32.90</td>
<td>3</td>
<td>auto</td>
<td>1</td>
<td>2</td>
<td>FALSE</td>
</tr>
<tr>
<td>32.90</td>
<td>3</td>
<td>auto</td>
<td>1</td>
<td>4</td>
<td>TRUE</td>
</tr>
</tbody>
</table>
This appendix contains the code used throughout the research. It is split into multiple parts, to begin with the (Python) code used to prepare the dataset, followed by the main functions necessary for making the T2F and Ground Speed predictions (Python and R for Time Series forecasting). Also the code for optimizing hyperparameters is included.

E.1 Python code preparing data

Below the methods used to prepare the data are added.

```python
def matlab2DateTime(matlab_datenum):
    day = dt.datetime.fromordinal(int(matlab_datenum))
    dayfrac = dt.timedelta(days=matlab_datenum%1) - dt.timedelta(days = 366)
    return day + dayfrac

def replaceMatlabDatenums(df, matlab_datenum_columns):
    print("Replacing Matlab datenums...")
    for matlab_datenum_column in matlab_datenum_columns:
        datetimes = []
        matlab_datenums = df[matlab_datenum_column]
        for matlab_datenum in matlab_datenums:
            datetime = matlab2DateTime(matlab_datenum)
            datetimes.append(datetime)
        col_index = df.columns.get_loc(matlab_datenum_column)
        df.drop(matlab_datenum_column, axis=1, inplace=True)
        df.insert(col_index, matlab_datenum_column, datetimes)
```
def insertLandingTimes(df):
    landing_times = []
    lastGateDurations = df['tpass_1'] - df['tpass_1.5']

    for lastGateDuration, tpass_1 in zip(lastGateDurations, tpass_1s):
        landing_time = tpass_1 + lastGateDuration + lastGateDuration
        landing_times.append(landing_time)

    df.insert(3, 'landingtime', landing_times)

def insertFlightWeekdays(df):
    weekdays = []
    date_times = df['tpass_1']

    for date_time in date_times:
        weekday = dt.datetime.weekday(date_time)
        weekdays.append(weekday)

    df.insert(3, 'weekday', weekdays)

def insertT2F(df, tpass__columns, t2f__columns):
    landing_times = df['landingtime']

    for tpass__column, t2f__column in zip(tpass__columns, t2f__columns):
        t2fs = []
        tpass_es = df[tpass__column]

        for tpass_, landing_time in zip(tpass_es, landing_times):
            t2f_datetime = landing_time - tpass_
            t2f_seconds = t2f_datetime.total_seconds()
            t2fs.append(t2f_seconds)

        col_index = df.columns.get_loc(tpass__column)
        #df.drop(tpass__column, axis=1, inplace=True)
        df.insert(col_index, t2f__column, t2fs)

def saveData(df):
    df.to_excel('EHAM-flight-2015-T2F-Dates-LandingTimes-Days.xlsx', index=False, sheet_name='Data')
E.2 Python code predict T2F/Ground Speed

```python
def readCSVData(source):
    print("Reading CSV...")
    df = pd.read_csv(source)
    return df

def getXY(df, feature_names, target_names):
    """Get the X and y out of a DataFrame""
    
    Args:
    df (pandas DataFrame): The DataFrame to extract the X and y from.
    feature_names: an array containing the feature names
    target_names: an array containing the targets

    Returns:
    X: The features of the dataset.
    y: The targets of the dataset.
    """
    features = df[feature_names]
    targets = df[target_names]
    # y is the newly created target column and X the features columns
    y = targets.as_matrix()
    X = features.as_matrix()

    return X, y

def bootstrap(X, y):
    """Generate a training and test dataset for the given samples and labels."
    
    Args:
    X (array of shape [n_samples, n_features]): The samples.
    y (array of shape [n_samples]): The labels.

    Returns:
    X_train (array of shape [n_samples, n_features]): The generated training samples.
    y_train (array of shape [n_samples]): The integer labels for cluster membership of each training sample.
    X_test (array of shape [n_samples, n_features]): The generated test samples.
    y_test (array of shape [n_samples]): The integer labels for cluster membership of each test sample.
    """
    # Bootstrap with replacement
    train_indices = np.random.randint(0, len(X), size=(len(X)))
    X_train = X[train_indices]
    y_train = y[train_indices]
    mask = np.ones(len(train_indices), np.bool)
    mask[train_indices] = 0
    X_test = X[mask]
    y_test = y[mask]
    test_indices = np.transpose(np.nonzero(mask))

    return X_train, y_train, X_test, y_test, test_indices
```
def predictUsingBootstrapRepeats(regressor, X, y, num_bootstraps=100, multivar = True):
    """Make predictions using a regressor and bootstrap repeats using a feature space X on a target set y
    Args:
    regressor: the regressor to be used
    X (array of shape [n_samples, n_features]): The samples.
    y (array of shape [n_samples]): The labels.
    num_bootstraps: the number of bootstrap repeats
    multivar: Multivariate targets (e.g. 8 tdiffs) or 1 single target variable
    Returns:
    measures: a dictionary containing the MSE, Std MSE, RMSE, Bias, Variance and flight errors for the all flights
    """
    y_predictions = [[] for i in range(len(X))]
    y_true = [[] for i in range(len(X))]
    for x in range(0, num_bootstraps):
        # get bootstrapped sets
        X_train, y_train, X_test, y_test, test_indices = bootstrap(X, y)
        regr = regressor
        # train the regressor with the bootstrapped training set
        if (np.array(y_train.shape[1]) == 1):
            regr.fit(X_train, y_train.ravel())
        else:
            regr.fit(X_train, y_train)
        y_pred = regr.predict(X_test)
        # add the predictions to a matrix
        for index, prediction in enumerate(y_pred):
            test_indices = test_indices.ravel()
            y_true[test_indices[index]].append(y_test[index])
            y_predictions[test_indices[index]].append(prediction)
        print ('Finished Bootstrap repeat %d' % x)
        # create occurrences array for the weight of the points
        occurrences = []
        # create array to store all average predictions
        avg_predictions = []
        # create array to store all squared biases
        squared_biases = []
        # create array to store variance of all points
        variances = []
        # create array to store mse in
        mses = []
        # create array to store relative flight errors in
        flight_errors = []
        for index, prediction in enumerate(y_predictions):
            number_of_predictions = len(y_predictions[index])
            occurrences.append(number_of_predictions)
# if there are no predictions for this x, add 0

if not y_predictions[index]:
squared_biases.append(0.0)
avg_predictions.append("nan")
variances.append(0.0)
mses.append(0.0)

else:
    trajectory_predictions = []
    if multivar:
        if y_true[index][0][0] < 50: # TDIFF
            actual_value = np.sum(y_true[index][0])
            for trajectory in prediction:
                trajectory_predictions.append(np.sum(trajectory))
        else: #GSPASS
            actual_value = np.mean(y_true[index][0])
            for trajectory in prediction:
                trajectory_predictions.append(np.mean(trajectory))
    else:
        actual_value = np.mean(y_true[index][0])
        for trajectory in prediction:
            trajectory_predictions.append(np.mean(trajectory))
    avg_prediction = np.mean(trajectory_predictions)
    avg_predictions.append(avg_prediction)
    avg_flight_error = avg_prediction - actual_value
    flight_errors.append(avg_flight_error)
    squared_bias = (avg_prediction - actual_value) ** 2
    squared_biases.append(squared_bias)
    variance = np.var(trajectory_predictions)
    variances.append(variance)
    actual_values = [actual_value] * number_of_predictions
    error = mean_squared_error(actual_values, trajectory_predictions)
    mses.append(error)
    flight_errors = np.array(flight_errors)
    weights = np.array(occurrences) / num_bootstraps
    final_bias = np.dot(squared_biases, weights) / sum(weights)
    final_variance = np.dot(variances, weights) / sum(weights)
    final_mse = np.dot(mses, weights) / sum(weights)
    std_mse = np.std(mses)
    final_rmse = final_mse ** 0.5
    measures = {
        'mse': final_mse, 'std_errors': std_mse, 'rmse': final_rmse,
        'flight_errors': flight_errors,
        'bias': final_bias, 'variance': final_variance}

return measures
```python
def evaluateT2F4NM(flight_errors, evaluation_crit_efficiency=5, evaluation_crit_safety=10):
    """Using the flight errors of the entire dataset, calculate the evaluation performance criteria as defined in the thesis"

    Args:
    flight_errors: Numpy array containing flight errors for each flight in the entire dataset
    evaluation_crit_efficiency: The number of seconds used to evaluate efficiency (default: 5 seconds)
    evaluation_crit_safety: The number of seconds used to evaluate safety (default: 5 seconds)

    Returns:
    performance_criteria: a dictionary containing the T2F.II and T2F.III percentages
    """

t2f_ii = sum([i < evaluation_crit_efficiency for i in np.abs(flight_errors)]) / flight_errors.size

t2f_iii = sum([i > evaluation_crit_safety for i in np.abs(flight_errors)]) / flight_errors.size

# Put all performance percentages into one dictionary:
performance_criteria = {'T2F.II': t2f_ii, 'T2F.III': t2f_iii}

return performance_criteria
```

# START SET VARIABLES

prediction_type = 't2f'
#prediction_type = 'gspass'
#ac = 'b738'
#ac = 'a332'
trajectory = 4 # make predictions over the last 4 NM
#trajectory = 10 # make predictions over the last 10 NM

#algorithm = 'gp' # Gaussian Process
#algorithm = 'ada' # AdaBoost
#algorithm = 'gbr' # GradientBoostingRegressor
#algorithm = 'rf' # RandomForestRegressor
algorithm = 'bagging' # BaggingRegressor

optimize = False # Optimize parameters for the regressors or not

# END SET VARIABLES
```
if prediction_type == 't2f':
    if ac == 'b738':
        df = readCSVData('..../data/EHAM-flights-B738.csv')
    elif ac == 'a332':
        df = readCSVData('..../data/EHAM-flights-A332.csv')

if trajectory == 4:  # T2F 4NM:
    feature_names = ['weekday', 'METARdrc', 'METARsknt', 'METARvbsy',
                     'METARtmpc', 'METARcwnd', 'METARhnd',
                     'Acin10NM', 'tdiff_19.5_19', 'tdiff_19.18.5',
                     'tdiff_18.5_18', 'tdiff_18.17.5',
                     'tdiff_17.5_17', 'tdiff_17.16.5', 'tdiff_16.5_16',
                     'tdiff_16.15.5', 'tdiff_15.5_15',
                     'tdiff_15.14.5', 'tdiff_14.5_14', 'tdiff_14.13.5',
                     'tdiff_13.5_13', 'tdiff_13.12.5',
                     'tdiff_12.5_12', 'tdiff_12.11.5', 'tdiff_11.5_11',
                     'tdiff_11.10.5', 'tdiff_10.5_10']
    #target_names = ['t2f_4']
    target_names = ['tdiff_4.3.5', "tdiff_3.5_3", "tdiff_3.2.5",
                    "tdiff_2.5_2", "tdiff_2_1.5", "tdiff_1.5_1", "tdiff_1_0.5",
                    "tdiff_0.5_0"]

elif trajectory == 10:  # T2F 10NM:
    feature_names = ['weekday', 'METARdrc', 'METARsknt', 'METARvbsy',
                     'METARtmpc', 'METARcwnd', 'METARhnd',
                     'Acin10NM', 'tdiff_19.5_19', 'tdiff_19.18.5',
                     'tdiff_18.5_18', 'tdiff_18.17.5',
                     'tdiff_17.5_17', 'tdiff_17.16.5', 'tdiff_16.5_16',
                     'tdiff_16.15.5', 'tdiff_15.5_15']
    #target_names = ['t2f_10']
    target_names = ['tdiff_10.9.5', "tdiff_9.5_9", "tdiff_9.8.5",
                    "tdiff_8.5_8", "tdiff_8_7.5", "tdiff_7.5_7", "tdiff_7.6_5",
                    "tdiff_6.5_6", "tdiff_6_5.5", "tdiff_5.5_5", "tdiff_5_4.5",
                    "tdiff_4.5_4", "tdiff_4_3.5", "tdiff_3.5_3", "tdiff_3_2.5",
                    "tdiff_2.5_2", "tdiff_2_1.5", "tdiff_1.5_1", "tdiff_1_0.5",
                    "tdiff_0.5_0"]

if prediction_type == 'gspass':
    if ac == 'b738':
        df = readCSVData('..../data/EHAM-flights-B738_incl_gspass.csv')
    elif ac == 'a332':
        df = readCSVData('..../data/EHAM-flights-A332_incl_gspass.csv')

if trajectory == 4:  # GSPASS 4NM:
    feature_names = ['weekday', 'METARdrc', 'METARsknt', 'METARvbsy',
                     'METARtmpc',
                     'METARcwnd', 'METARhnd', 'Acin10NM', "gspass_19.5",
                     "gspass_18.5", "gspass_18", "gspass_17.5",
                     "gspass_17", "gspass_16.5", "gspass_16", "gspass_15.5",
                     "gspass_15", "gspass_14.5",
                     "gspass_14", "gspass_13.5", "gspass_13", "gspass_12.5",
                     "gspass_12", "gspass_11.5",
                     "gspass_11", "gspass_10.5", "gspass_10"]
    #target_names = ['avg_gspass_4']
    target_names = ['gspass_4', "gspass_3.5", "gspass_3", "gspass_2.5",
                    "gspass_2", "gspass_1.5", "gspass_1", "gspass_0.5", "gspass_0"]
```python
elif trajectory == 10:  # GSPASS 10NM:
    feature_names = ['weekday', 'METARdirct', 'METARsknt', 'METARvsby',
                     'METArtmpc',
                     'METARcsnd', 'METARhwind', 'ACin10NM', "gspass_19.5",
                     "gspass_18.5", "gspass_18", "gspass_17.5",
                     "gspass_17", "gspass_16.5", "gspass_16", "gspass_15.5",
                     "gspass_15",
                     "gspass_14", "gspass_13.5", "gspass_13",
                     "gspass_12.5", "gspass_12",
                     "gspass_11.5", "gspass_11",
                     "gspass_10.5", "gspass_10"]

    #target_names = ["avg_gspass_10"]
    target_names = ["gspass_10", "gspass_9.5", "gspass_9", "gspass_8.5",
                    "gspass_8", "gspass_7.5", "gspass_7", "gspass_6.5", "gspass_6",
                    "gspass_5.5", "gspass_5", "gspass_4.5", "gspass_4", "gspass_3.5",
                    "gspass_3", "gspass_2.5", "gspass_2", "gspass_1.5", "gspass_1",
                    "gspass_0.5", "gspass_0"]

    X, y = getXy(df, feature_names, target_names)

    if algorithm == 'gp':
        if optimize:
            parameter_grid = {'theta0': np.logspace(-2, 1), 'regr': ['constant',
                        'linear'],
                        'corr': ['absolute_exponential', 'squared_exponential',
                        'cubic', 'linear']}

            regr = GaussianProcess(verbosetrue, random_start=100)
        best_params = optimizeHyperParameters(algorithm, regr, X, y,
                                             parameter_grid, save_result='true')
        del best_params['MSE']
        else:  # no need to run the optimizer again, here they are for T2F 4NM:
            best_params = {'theta0': 0.193069773, 'regr': 'linear', 'corr':
                           'absolute_exponential'}

        regr = GaussianProcess('best_params, random_start=100)

    elif algorithm == 'ada':
        if optimize:
            parameter_grid = {'loss': ['linear', 'square', 'exponential'],
                              'n_estimators': [50, 100, 200, 400, 1000],
                              'learning_rate': [0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1]}

            regr = ensemble.AdaBoostRegressor(random_state=42)
        best_params = optimizeHyperParameters(algorithm, regr, X, y,
                                              parameter_grid, save_result='true')
        del best_params['MSE']
        else:  # no need to run the optimizer again, here they are for T2F 4NM:
            best_params = {'loss': 'exponential', 'learning_rate': 0.2,
                           'n_estimators': 100}

        regr = ensemble.AdaBoostRegressor(random_state=42, **best_params)

    # the code for T2F 4NM is here
```

elif algorithm == 'gbr':
    if optimize:
        parameter_grid = {'loss': ['ls', 'lad', 'huber', 'quantile'],
                          'n_estimators': [50, 100, 200, 400, 1000],
                          'learning_rate': [0.1, 0.3, 0.5, 0.7, 1],
                          'max_depth':
                          [1, 3, 5, 7, 9, 12],
                          'min_samples_split':
                          [1, 2, 4, 6, 8, 10],
                          'min_samples_leaf': [1, 2, 4, 6, 8, 10],
                          'warm_start': [True, False]
                    }
        regr = ensemble.GradientBoostingRegressor(random_state=42)
        best_params = optimizeHyperParameters(algorithm, regr, X, y,
                                               parameter_grid, save_result=True)
        del best_params['MSE']
    else:
        # no need to run the optimizer again, here they are for T2F 4NM:
        best_params = {'min_samples_leaf': 1, 'learning_rate': 0.1, 'max_depth':
                        1, 'min_samples_split': 0,
                        'loss': 'lad', 'n_estimators': 400, 'warm_start': True}
        regr = ensemble.GradientBoostingRegressor(random_state=42, **best_params)
elif algorithm == 'rf':
    if optimize:
        parameter_grid = {'max_features': ['auto', 'sqrt', 'log2'],
                          'max_depth':
                          [3, None],
                          'min_samples_split': [1, 2, 4, 6, 8, 10],
                          'min_samples_leaf': [1, 2, 4, 6, 8, 10],
                          'warm_start': [True, False]
                    }
        regr = ensemble.RandomForestRegressor(random_state=42)
        best_params = optimizeHyperParameters(algorithm, regr, X, y,
                                               parameter_grid, save_result=True)
        del best_params['MSE']
    else:
        # no need to run the optimizer again, here they are for T2F 4NM:
        best_params = {'n_estimators': 128, 'max_features': 'auto',
                       'min_samples_leaf': 8, 'min_samples_split': 4, 'max_depth': None,
                       'warm_start': True}
        regr = ensemble.RandomForestRegressor(random_state=42, **best_params)
elif algorithm == 'bagging':
    if optimize:
        parameter_grid = {'n_estimators': [10, 100, 200, 400, 1000],
                          'bootstrap':
                          [True, False],
                          'warm_start': [True, False]}
        regr = ensemble.BaggingRegressor(random_state=42)
        best_params = optimizeHyperParameters(algorithm, regr, X, y,
                                               parameter_grid, save_result=True)
        del best_params['MSE']
    else:
        # no need to run the optimizer again, here they are for T2F 4NM:
        best_params = {'n_estimators': 100, 'bootstrap': True, 'warm_start': True}
        regr = ensemble.BaggingRegressor(random_state=42, **best_params)
predictT2F_GS <- function(data, output) {

library(forecast)

# Set the distance (in NM) from threshold from where to be forecasted:
forecast_from_nm = 10
# Calculate MSE from how many NM from threshold:
mse_from_nm = 4

data_points = length(data)
# As input take the tdiffs/gspass values up to the gate to be forecasted:
nr_inputs = data_points - (forecast_from_nm * 2)
inputs = unlist(data[0:nr_inputs]);

# The number of points to be forecasted are the other remaining gates:
nr_targets = forecast_from_nm * 2
targets = unlist(data[nr_targets:data_points])

library(tseries)
y = ts(inputs)

# Fit the best ARIMA model automatically:
fit = auto.arima(y)
# Print a summary of the fitted model:
#summary(fit)

pred = forecast.Arima(fit, h=nr_targets)
# Plot the forecasted points:
plot(forecast(pred))
# Plot the actual points in the same plot:
points(data)

# Point forecasts:
forecasts = pred$mean

library(Metrics)
if (data > 50) { # Ground Speed forecasts
gs_pred = mean(tail(forecasts, mse_from_nm^2))
gs_actual = mean(tail(targets, mse_from_nm^2))
gs_error = gs_pred - gs_actual # Error for this flight
abs_gs_error = abs(gs_error)

cat(paste(gs_error, abs_gs_error, sep=",", file = output, append = T, fill = T)

} else { # T2F forecasts
t2f_pred = sum(tail(forecasts, mse_from_nm^2)); print(t2f_pred)
t2f_actual = sum(tail(targets, mse_from_nm^2)); print(t2f_actual)
print(tail(targets, mse_from_nm^2))
t2f_error = t2f_pred - t2f_actual # Error for this flight
abs_t2f_error = abs(t2f_error)

cat(paste(t2f_error, abs_t2f_error, sep=",", file = output, append = T, fill = T)

}

}
E.4 Python code optimizing hyperparameters

```python
def optimizeHyperParameters(algorithm, regressor, X, y, parameter_grid,
                            save_result=False):
    """Optimize Hyperparameters for a regressor using GridSearchCV on a dataset X
    with targets y using nested resampling

    Args:
    algorithm: a String containing the algorithm name, for file saving purposes
    regressor: regressor for which to optimize hyperparameters
    X (array of shape [n_samples, n_features]): The data to fit.
    y (array of shape [n_samples]): The target variable to try to predict.
    parameter_grid: A dictionary containing the hyperparameters and their
    range of values to test
    save_result: Save the results of the GridSearchCV in a dataframe or not

    Returns:
    best_params: a dictionary containing the best parameters optimizing
    for lowest Mean Squared Error
    """

    # If just a single target flatten the target matrix:
    if (np.array(y.shape[1]) == 1):
        y = y.ravel()
    cv = GridSearchCV(regressor, parameter_grid, n_jobs=-1,
                    scoring='neg_mean_squared_error')
    cv.fit(X, y)
    print("The best parameters are %s with a score of %0.2f" % (cv.best_params_,
                                                            cv.best_score_))
    if save_result:
        search_scores = gridScoresToDataframe(cv.grid_scores_)
        filename = 'hyperparameter_opt_' + algorithm + '.csv'
        search_scores.to_csv(filename)
    return cv.best_params_

def gridScoresToDataframe(grid_scores):
    """Return a panda dataframe containing the mean and hyperparameters
    combinations.

    Args:
    grid_scores (list of named tuples): Contains scores for all parameter
    combinations in param_grid.

    Returns:
    (DataFrame): A dataframe with the mean and hyperparameter combinations.
    """
    scores = []
    for cv_score in grid_scores:
        score = cv_score.parameters
        mean = {'MSE': np.abs(cv_score.mean_validation_score)}
        score.update(mean)
        scores.append(score)
    return pd.DataFrame(scores)
```


