A Boosted Tree Framework for Runway Occupancy and Exit Prediction

Dario Martinez, Seddik Belkoura and Samuel Cristobal
Innaxis Research and Foundation

Floris Herrema
Delft University of Technology

Philipp Wächter
Austro Control

***Abstract*—This paper presents a machine learning algorithm trained to predict the actual runway occupancy times at Vienna airport. Runway occupancy times are usually estimated by individual studies on previous operations or analytical methods. However, due to the uncertainty of the actual operations, wide safety margins need to be applied by air traffic controllers. Finding an acceptable compromise between the maximisation of the throughput and minimisation of the risk may improve runway utilisation. In the future, machine learning models could boost controllers' confidence by giving more accurate predictions on expected runway occupancy times, resulting in a smaller buffer that ultimately increases capacity without compromising safety levels.**

The training of the machine learning model is focused on runway 34 of Vienna airport. Different predictive models compute the expected runway occupancy time and expected exit at different distances from threshold. Features are engineered based on meteorological conditions, sequence of flights, aircraft trajectories and flight plan. At the end of the paper, a list of the relevance importance of precursors is presented exactly at the threshold and 2NM ahead of it.

***Keywords*—Runway occupancy, prediction, machine learning, lightGBM.**

I. Introduction

In the past, the most commonly used approach to increase airport capacity was to modify infrastructure, e.g. additional runway or terminals. However, this approach is difficult and costly to implement. On the other hand, recent developments in Artificial Intelligence and Machine Learning (ML) have proven to be more efficient in optimizing systems through data analytics. A representative indicator of airport performance is the Runway Utilization (RU), i.e. the sum of the Runway Occupancy Times (ROTs) of all the landing flights divided by the total time of use of the runway. For each landing flight, the ROT is defined as the time interval from the instant the flight surpasses the threshold to the instant it vacates the runway completely. Therefore, the RU is maximised by minimising the aggregated ROTs. Hence, the obvious approach to the problem is to study whether a decrease of the ROT is feasible [1]–[6]. From previous works, two drawbacks in the RU studies can be extracted: (a) all previous models try to manually classify aircrafts into groups and fail to differentiate performance characteristics; and (b) as claimed by Koenig [4] in the 70's, a strategy based in minimising the ROT can be highly influenced by pilot/airline motivations, e.g. a pilot may spend more time on the runway for a better exit . This adds the complexity of non-trivial and most likely unknown incentives of airlines and pilots.

In recent years, airport control has turned towards another type of solution for RU optimisation: controlling the time between consecutive landing flights. Time Based Separation (TBS) [7]–[10] is a new operating procedure for separating aircraft by time. While this effectively introduces a more robust and resilient strategy, it still raises some issues due to separation being predefined by weather conditions and aircraft types. In fact, the ROT might depend on several other factors unknown to the ATCO. In this context, Machine Learning can be a powerful tool in accounting for more potential factors, i.e. features, and measuring their impact on ROT predictions, i.e. precursors.

In fact, the risk associated with decreased separation between flights is highly dependent on the risk assessment of the controlled. This becomes particularly important in High Intensity Runway Operations (HIRO) periods where the amount of stress on the controllers and the higher level of risks might directly impact their optimisation of runway throughput. Currently, no human support system assists the arrival manager (AMAN) and departure manager (DMAN) on predicting runway exits and ROTs.

We propose the first steps towards a predictive analytics tool that could help controllers deal with such complex and high risk assessment tasks. This tool specifically identifies the precursors causing higher than average ROTs, mostly by assessing the chances of a flight not taking the procedural exit. This paper presents a novel data-based approach: by using predictive modelling techniques, key predictor variables can be extracted from a set of observations. It is important to note that the predictive model only uses information available before the threshold to make the prediction and whole flight information to train the model.

This paper falls under the scope of the H2020 "SafeClouds.eu" project that aims to increase safety levels by using big data. This paper uses the SafeClouds platform called DataBeacon. DataBeacon enables safe data sharing using de-identification techniques called Secure Data Frames or SDFs. Because of this, the model was trained using a rich variety of data sources and the future plans include expanding over more datasets - thus the relevance of "SafeClouds.eu" and DataBeacon.

The paper is organised as follows: Section II provides a de-

TABLE I
AIP EXIT TAXIWAY DESIGN

Aircraft ICAO Category	Procedural exit taxiway code	Percentage
Heavy	B4, B5	10%
Medium	B5, B7	80%
Light	B7, B9	10%

scription of the key characteristics of Vienna Airport (LOWW) and the data available. In Section III, the predictive model is presented with an efficient and powerful classification boosted tree along with the problem tackled. In Section IV, variables and potential precursors are explored and then extracted using feature engineering techniques. These features are then used in a statistical learning approach to predict the runway exit in with a binary classification and ROT with a regression in Section V, with a post-analysis of the results. Finally, conclusions and directions for future work are presented in Section VI.

II. VIENNA AIRPORT CHARACTERISTICS

The factors influencing actual runway occupancy times are sometimes not obvious and difficult to assess, even for experts. For example, some pilots familiar with an airport may take a further exit in order to be closer to the assigned gate, especially if they are already delayed. In order to successfully build a machine learning model to predict ROT, it needs to be tuned specifically for the target airport and its own characteristics such as topology, layout, design of the runways and exits. In our case, the model covers Vienna airport (LOWW), which has 2 runways used in both directions (R11/R29 and R16/R34 - see Fig. 1). Runway R34 will be the main focus as it concentrates 46% of the traffic. Fortunately, because of the orientation of this runway, certain aspects such as the distance to the gate mentioned before are not that relevant. Future studies will include runway R29 and test whether there is a difference between the influence of the assigned gate in both runways.

Landing distance and ROT are known to be influenced by airport characteristics such as runway length, slope, exit taxiway angle with respect to the runway [4]. However, with this unique airport, those parameters can be considered as the same for all the landing aircrafts. From a statistical point, the variables remain constantly distributed for every sample in the dataset. Landing distance and ROT are also not perfectly correlated. For example, the same landing distance can correspond to different ROTs because of different break profiles and velocities. The study does not take into account specific engineering aircraft technologies such as ”brake-to-vacate”, which could influence the ROT. In general terms, those flights will be considered outliers. In spite of this, some particular characteristics of the airport configuration are relevant and will be confirmed by statistical analysis later on:

ATCOs use a mixed-operation configuration in R34 (i.e. both departure and arrivals on the same runway) in HIRO situations. Obviously, the risk level of a higher runway utilisation in such conditions might be higher because of the reduced

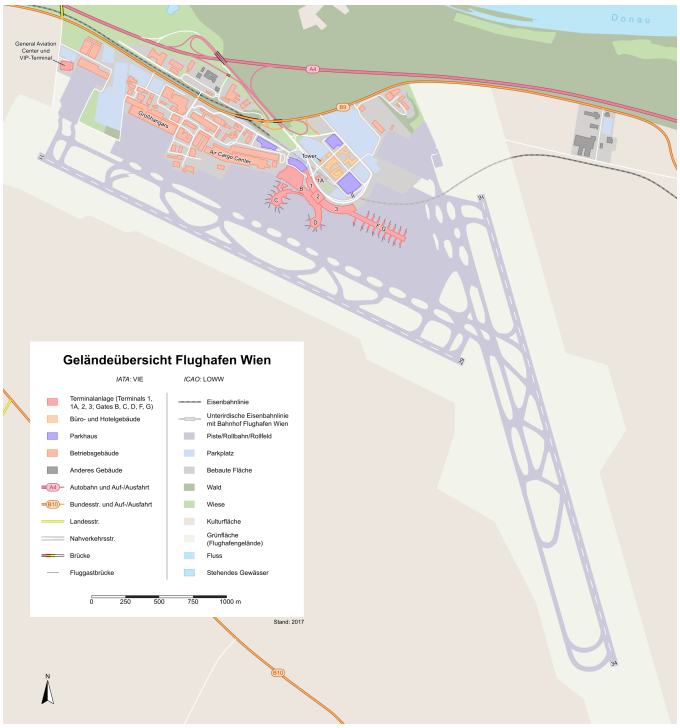

Figure 1. LOWW Runway Design

buffer between planes. Therefore, airport configuration should be included as a potential precursor (or a proxy of it, as the configuration was not available in the data and needed to be engineered - see Section IV for more details).

The Aeronautical Information Publication (AIP) of LOWW stipulates that ”To minimise the runway occupancy time, pilots should make use of the following procedures: (a) In general, an exit taxiway should be planned which is used after landing under normal circumstances. Missing an earlier exit taxiway and continuing slowly to the next exit taxiway should be avoided; and (b) If possible, the runway should be vacated via the defined exit taxiway for each aircraft category”. In other words, there are advised, but not obligatory, exits depending on the aircraft category (Heavy, Light, Medium, etc). One particular problem is the imbalanced distribution of categories for arrivals at LOWW airport, which is estimated at 80% Medium, 10% Heavy and 10% Light.

III. PROBLEM ASSESSMENT

To develop a reliable pattern recognition algorithm, not only does the exit and/or the runway occupancy time of a aircraft need to be predicted, but a precursors analysis must also be performed in order to ensure that the ATCOs have relevant, reliable and detailed information before the actual landing when it is still useful. Experts from AUSTROCONTROL (ATCOs of the Vienna airport) have mentioned the potential benefits, stress-wise and risk-wise, of ATCOs getting access to robust predictions of a flight not taking the recommended exit. These predictions can cover aircraft spending additional time

on the runway and help identify situations that could force a "go-around" if the runway is not vacated as expected.

Taking into consideration the ATCOs requirements and goals, the problem can be summarised into two research questions:

- With what accuracy can we predict flights bypassing the exit taxiway defined in the AIP when the aircraft is 2NM away from threshold?
- What is the error related to the forecast of the runway occupancy time (from threshold to complete exit taxiway) when the landing aircraft is 2NM away from threshold?

From the Machine Learning perspective, the first question corresponds to a binary classification problem, i.e. '0' for the cases in which the aircraft took the defined exit and '1' for the cases in which it didn't. The second question corresponds to a regression problem, i.e. the 'prediction' of a continuous value.

The instant in which the prediction is made has been initially set to 2NM before threshold. As stated by ATCO experts, 2NM before threshold represents an approximation of their time-window of focus. Indeed, ATCOs usually focus their attention on the flights close to landing and do not rely as much on previous information. The controllers involved in SafeClouds.eu have decided that 2NM should be the reference distance; just enough time to react but narrow enough of a window to reduce the number of possible unwanted alerts.

In the dataset, 74% of the flights exit through the expected taxiway. Therefore, 26% of the flights behave "abnormally" by not taking the expected exit. This means that the data is "imbalanced". As an initial study, the Machine Learning problem presented in this section has been tackled as a balanced problem, further work is envisaged using specific imbalanced-class mathematical machinery.

IV. METHODOLOGY

A. Data

The data was provided by AUSTROCONTROL and covers the whole 2015 and segments of 2014 and 2016. It contains:

A radar track database - a concatenation of 4D trajectories for flights defined by their callsign, date, aircraft registration and ICAO category. Each trajectory is defined as $P_i^* = [t_i^*, X_i^*, Y_i^*, Z_i^*, R_i^*, V_i^*]$ where each line of P_i accounts for a new timestamp t_i; (X_i, Y_i, Z_i) stands for the latitude, longitude (in degrees) and Flight Level of the flight; V_i records the velocity of the flight and R_i is a boolean that values 1 whenever the position of the aircraft is between the runway threshold and an exit taxiway. The resulting matrix is of size $(n_i, 6)$, n_i being the number of observations of that particular flight i. The overall radar track database is therefore a concatenation of all the P_i of size $(\sum_i n_i, 6)$.

An airport information database which returns the callsign and date of the flights along with some planned information as the Estimated Time of Arrival (ETA), departure and arriving airports, runway and gates assigned.

Different meteorological databases such as METAR, SNOWTAM or WMA. The information can be redundant among the different datasets, but of relevance, wind speed, visibility and QNH were available.

Radar Track and Airport information databases are merged together using callsigns and dates as unique keys, which filters the data as such: (a) flight trajectories spotted by the radar but not landing or departing from the airport are automatically discarded; (b) Matching uncertainties (e.g. redundant callsign over a same day) are also discarded. Note that, in this deliverable, only landings are taken into consideration and only those that correspond to the studied runway. This final part of the data consists of 59.369 flights.

The parameter Z_i is an altitude with respect runway. This altitude is seen as a "flight level" that is normally below the transition altitude, and as a result depends on an isobaric pressure reference that may change. Also, the distance to arrival is absent from the data, therefore making it difficult to identify the point of prediction (2NM) as described in our forecasting problem. Thus, it is necessary to format and complete the original data. The dataset incorporates the following modifications:

Distance from arrival The vector of distances of the flight (at each time stamp) D_i^* from the reference point T, which indicates the threshold of runway 34, is added. The threshold T coordinates are defined as $(X_{thres}, Y_{thres}, Z_{thres})$ as (48.092222, 16.596667, 597). For each trajectory point, D_i is calculated as the great-circle distance from T, using the Haversine formula:

$$A = sin^2(\frac{X_i - X_{thres}}{2}) + cos(Y_i)cos(Y_{thres})sin^2(\frac{Y_i - Y_{thres}}{2})$$

$$D = 2.H.atan^2(\sqrt{a}, \sqrt{1a})$$

with coordinates in radian and $H = R + E_{LOWW}$, R being the mean radius of the earth and $E_{LOWW} = 597ft$ being the elevation of the runway 34.

Flight Level to Altitude Merging the data has also allowed to link all flights with a specific atmospheric pressure measured and reported in the METAR as the QNH (Query: Nautical Height). It is possible to approximate the altitude from the flight level using the following formula:

$$Z_i(NM) = Z_i(FL) * 100 + + \frac{288.15}{0.0065 * 0.3048}(\frac{QNH}{1013.25}^{\frac{0.0065*287}{9.81}} - 1)$$

Angle to runway The angle, with respect with direction of the runway, has been computed. It shows the variation in trajectory with respect to the direction of the runway and indirectly to the ILS. The angle, in radians, is calculated with respect to the threshold coordinates as:

$$\phi_i = arctan(\frac{X_i - X_{thres}}{Y_i - Y_{thres}})$$

Energy The energy of the flight might also seem a value interesting to manufacture. However, the kinetic energy of a flight depends on its mass. One solution around the problem is to compute the amount of energy per unit of mass. Another column has been added:

$$KE_i = Z_i - E_{LOWW} + \frac{V_i^2}{2.g})$$

Z_i being in NM and g being the acceleration of gravity on the surface of the earth at sea level.

B. Feature Engineering

Table II presents the most promising list of features extracted from a preliminary data mining exercise. Note that, though not all of them are going to be relevant precursors, it is necessary to calculate them from the raw data in order to assess their relevance. Table II presents both engineered features and available data within the databases. Also, please note that some features are static information related to the flights and sometimes to the weather; others are extracted from dynamic characteristics of the flights (trajectories time series, distances and angles, etc.). Static information does not convey problems and consists of unique information per flight (e.g. type of aircraft, state of the runway, etc.). Dynamic information, such as position, velocity, etc. are much more complex to understand physically in a model and to process in a learning algorithm as the amount of observations per flight is too high (>100), therefore increasing over-fitting (curse of dimensionality [11]). Unless specific dynamic models are used (such as state space models, LTSM etc.), a strategy is needed to handle all the input time series. A two-step approach has been used:

1) Time-Series are sub-sampled. This is the process of reducing the number of samples of each series without losing significant information for the case study. In this particular case, only the information at $[10, 9.5, \ldots, 2.5, 2]NM$ has been considered instead of the whole time series using the already computed value D_i. The sub-sampling granularity normally reduces the computational requisites of the pre-processing steps as the expense of losing some information. In our case, similar results are expected independent of the granularity of the sampling.

2) Then features extraction is performed based on a preliminary data inspection and on domain-experts (pilots and ATCOs) experience. The extraction was also performed using a non-supervised clustering of the time series in order to recognise patterns. However, the results of the clustering were not satisfactory, partially due to mediocre tuning of the hyper-parameters.

In Table II, the features that have been extracted from the data or engineered are presented. An additional column differentiates between original features (raw information presented the dataset) or engineered features (calculated during the analysis). Notice that the model is being trained with 26 engineered features and only 4 "original" features . This is intended as, apart from reducing the dimensionality of the time-series, the nature of the machine learning algorithm used also influences that decision. While a powerful enough algorithm can identify how using QNH to transform the Flight Level into altitudes helps improve predictions, many processing steps are too complex for an algorithm to automatically learn the patterns without some guidance. Note that some studies like to use data reduction techniques, eg. Principal Component Analysis (PCA), for those situations. However, the features extracted using this methodology are not easily interpretable and not useful in the precursor analysis.

The final dataset defined to the learning algorithm is of shape $(59369, 34)$, with 59.369 being the number of flights and 34 the number of features used to train the model.

C. Machine learning algorithm

Gradient Boosting (GB) Frameworks (also known as Gradient Boosting Machines, GBM) [12] are powerful techniques for building predictive models. They select an arbitrary differentiable loss as the objective function and uses an additive model of many weak learners - typically regression trees - to minimize this loss. The parameters of the additional decision trees are tuned by a gradient descent algorithm. This methodology has gained popularity recently alongside continuous development of the well-established Random Forest algorithm [13]. The main advantage of using GBMs over other ML algorithms is that the model is iteratively trained. For each new round, the model uses data samples that were "difficult" to learn in previous iterations. This is perfect behaviour for this particular problem in addressing the 26% of flights that are not taking the AIP exit.

There are two iwdely-used GBMs in the data science community: XGBoost [14] and LightGBM [15], [16]. The former is very popular among Kaggle community and has been used for many competitions. The latter is a newcomer with several improved features and has already been applied successfully to fields like genomics [17] or acoustics [18]. All these methods can be used both as a classifiers or as regressors. Specifically, LightGBM:

- Uses histogram based algorithms, which aggregates continuous features into discrete bins to speed up training and reduce memory usage.
- Grows the tree leaf-wise, which can reduce even more the loss than a level-wise algorithm.

They are very similar in practice and when comparing results of using Random Forest, XGBoost and LightGBM. However, LightGBM had slightly better overall classification accuracy. As a result, LightGBM was chosen as the Machine Learning algorithm (for both classification and regression) used in all the presented experiments.

D. Cross-validation

When evaluating the performance of a binary classification using Machine Learning algorithms, classical performance metrics, such as plain accuracy, are usually not enough. False positives and false negatives are usually overlooked when only accuracy is used to understand the classifier. In this case study, a higher rate of false-negatives has arguably bigger consequences than a higher rate of false-positives. A false positive alert would only trigger the attention of the controller to a trivial situation (potentially distracting her from a riskier situation requesting attention). However, false negatives might

TABLE II
POTENTIAL PRECURSORS, O/E STANDS FOR ORIGINAL/ENGINEERED

Name	Source	O/E	Description
aircraftRegistration_x	Radar track	O	Registration tail of the aircraft
ICAOCategory_x	Radar track	O	ICAO category of the aircraft, i.e. Light, Medium or Heavy
arrivalGate	Airport information	O	Expected Gate in the Flight Plan
mixedOperation	Airport information	E	Computed from airport information. Mixed Operation is fixed as True when at least one departure and one arrival are detected in the runway in the preceding 15 minutes.
groundVisibility_m	METAR	O	Visibility measured by METAR
isRunwayWet	METAR	E	Binary feature indicating if there is information on abnormal friction on runways R88, R34 or R16 in the METAR report.
windGust	METAR	E	Binary feature indicating if there is a wind gust value in the METAR report or not.
windSpeed_trans_kts	METAR	E	Wind values reported by METAR are directed to the true north, a simple trigonometric correction has been applied to extract the perpendicular projection of the wind with respect to the runway: $v_{wind}.cos(\alpha_{wind} - 250)$ being v_{wind} the reported wind speed (kts) and α_{wind} the reported wind direction (degrees).
windSpeed_parallel_kts	METAR	E	Wind values reported by METAR are directed to the true north, a simple trigonometric correction has been applied to extract the parallel projection of the wind with respect to the runway: $v_{wind}.sin(\alpha_{wind} - 250)$ being v_{wind} the reported wind speed (kts) and α_{wind} the reported wind direction (degrees).
badVisibility	METAR	E	Binary Feature indicating if cloud layer opacity reported by METAR is higher than 5 okt and if the cloud layer height is less than 1500 ft.
Throughput	Airport information	E	Based on Estimated Times of Arrivals available on the airport information database, the calculated estimated throughput of arrivals within the hour (i.e. 30 minutes before to 30 minutes after the ETA of the observed flight)
aircraftRegistration_y	Airport information	E	The following aircraft is extracted from the airport information, based on the ETAs. See discussion on the possible noise this introduce.
ICAOCategory_y	Airport information	E	The following aircraft category (i.e Light, Medium or Heavy)
Distance_next_flight	Radar track	E	The following flight distance from arrival when the observed aircraft is at distance of prediction (e.g. 2NM)
velocity_next_flight	Radar track	E	The following flight velocity when the observed aircraft is at distance of prediction (e.g. 2NM)
number_breaks	Radar track	E	The number of significant decelerations that the aircraft performed during landing phase up until the distance of prediction. The threshold for significance has been chosen manually.
max_break	Radar track	E	The value of the highest deceleration.
max_break_distance	Radar track	E	The distance from arrival at which the highest deceleration happened.
number_acceleration	Radar track	E	The number significant accelerations the aircraft performed during landing phase up until the distance of prediction.
max_acceleration	Radar track	E	The value of the highest acceleration.
max_acceleration_distance	Radar track	E	The distance from arrival at which the highest acceleration happened.
mean_slope	Radar track	E	The mean slope of speed during landing phase over the subsampled trajectory.
large_angle	Radar track	E	The angle between aircraft position at 10NM and the direction of the runway.
adherence_distance	Radar track	E	The distance at which the aircraft angle with respect to the direction of the runway is inferior to 1
num_flat_intervals	Radar track	E	The number of 0.5NM intervals in which the flight remained at the same altitude.
last_energy	Radar track	E	Normalised Energy at 2NM.
mean_slope_energy	Radar track	E	Slope of energy over the subsampled trajectory.
last_speed	Radar track	E	Speed at the moment of prediction.
last_FL	Radar track	E	Altitude (feet) at the moment of prediction.
last_angle	Radar track	E	Angle of the aircraft position with respect the the runway at the moment of prediction.
delay	Radar track + Airport information	E	Delay is approximated computing the difference between the moment of the prediction and the ETA. Positive values indicate that the aircraft is already behind schedule (i.e. actual delay).
diff_speed	Radar track	E	Difference between speed at the moment of prediction and 0.5NM before.
diff_alt	Radar track	E	Difference between altitude at the moment of prediction and 0.5NM before.

be riskier as the algorithm falsely leads the ATCOs to expect normal behaviour from the incoming plane, therefore augmenting the risk of the situation (e.g. if the following flight has been given the authorisation to fly closer with the expectation of the first plane to behave normally). To take such information into account, we compute the Receive Operating Characteristics (ROC) curves: ROC curves are created by plotting the true positive rate against the false positive rate. The most widely used metric for performance is Area Under the Curve (AUC) which equals the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. In other words, the higher the AUC, the better the classifier [19].

Although these algorithms have been trained with adequate validation functions to measure how the Machine Learning model fit training data, its stability needs to be validated. This is based on the assessment of how well the learner will generalise an independent/unseen dataset in the future, otherwise known as cross-validation. Cross-Validation is crucial to avoid problems such as overfitting (the model contains more parameters that can be justified by the data) or selection bias (the selection of training data not being properly randomised). Cross-validation can be performed using different strategies. In thiscase, the K-Fold Cross-Validation was used: the data is divided in k subsets and the holdout method is repeated k times. Each time, one of the k subsets is used as the test dataset and the other k-1 subsets are used as the train dataset. The error estimation is averaged over all k trials to calculate the total effectiveness of the model. This methodology is particularly useful because it reduces bias by using most of the data for fitting. It also reduces the variance because most of the data is also used in the test dataset. For the experiments, the number of folds K were fixed at 8.

In the dataset, and due to the problem with the imbalance in the flights taking the AIP exit, the algorithm will tend to predict the output of the most numerous class. To contrast this statistical effect, the stratified form of K-Fold cross validation was used where all the folds are made by preserving the percentage of samples for each class. This methodology, in combination with the Gradient Boosting framework, will almost completely tackle the problem of having imbalanced classes.

V. Results

A. Binary classification case

The results of the classification return an accuracy of 0.772 and an AUC of 0.784. These results are represented as a confusion matrix in Figure 3, where 92% of the flights shown taking the expected exit (i.e. stipulated in the AIP) are classified correctly. However, it also shows that other flights are only correctly classified 34% of times. We strongly believe that very little improvement can be achieved through mathematical optimisation (i.e. improving the tuning of the algorithms or the validation strategies). The test performed by the authors (e.g. change of number of estimates of the trees, improve the number of folds of the validation, etc.) did not improve the prediction in a significant way. This implies that there is information about the flight that is not accounted for and relevant to the exit deviation from the AIP.

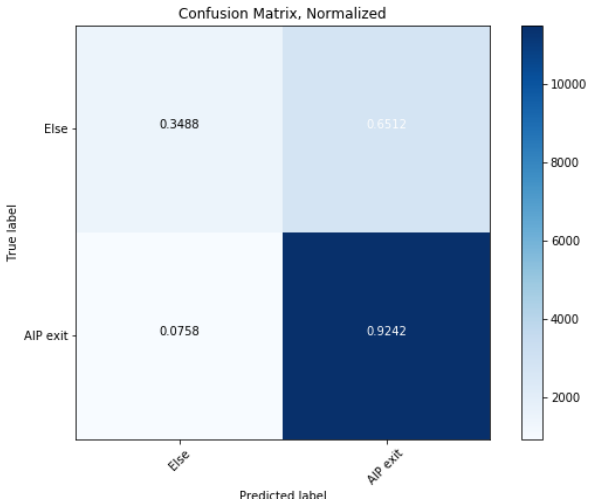

Figure 2. Normalised Confusion matrix of the classification

Figure 3 offers a look at the relevant features that have been used by the lightGBM algorithm to perform the classification. Delay can be seen as the most important value. A delayed flight might be more prone to take an exit close to its assigned gate even though it is not the one stipulated by the AIP. Hence the fact that the gate assigned is the second in line followed by the type of aircraft. One potential reason is that ATCOs grant pilots later exits if traffic situation permits and it reduces the taxi-time needed for the aircraft concerned. The alignment of the flight with the ILS glide slope as the velocity of the aircraft remains important features. Of course, the velocity of the following flight (i.e. next to land) is important. AUSTROCONTROL ATCOs confirmed the importance of such a feature as depending on the distance between both flights and their respective velocities. If needed, they can ask the pilot to exit early, if possible. Radio communication is absent from accessible data, yet this particular feature might account for situations in which ATCOs might contact the pilot to request an early exit. (This does not account for ATCO experiences or nature, such as being more cautious than normal for example.)

Several questions are present at this stage: What information can we be missing? Is it present in the data at hand at least in an approximated way? Is it present later in time?

The same analysis was performed with all the data until reaching threshold to find out whether something specific happens between 2NM and threshold. The results were improved very slightly (AUC of 0.793) with the speed at threshold being registered as the most important feature for classification (see Fig. 4). Although such a tool (i.e. prediction at threshold) would be useless for ATCOs, it shows that, except for value of the speed at threshold, few things change from a prediction at 2NM. Actually, Figure 5 shows how the performances hardly

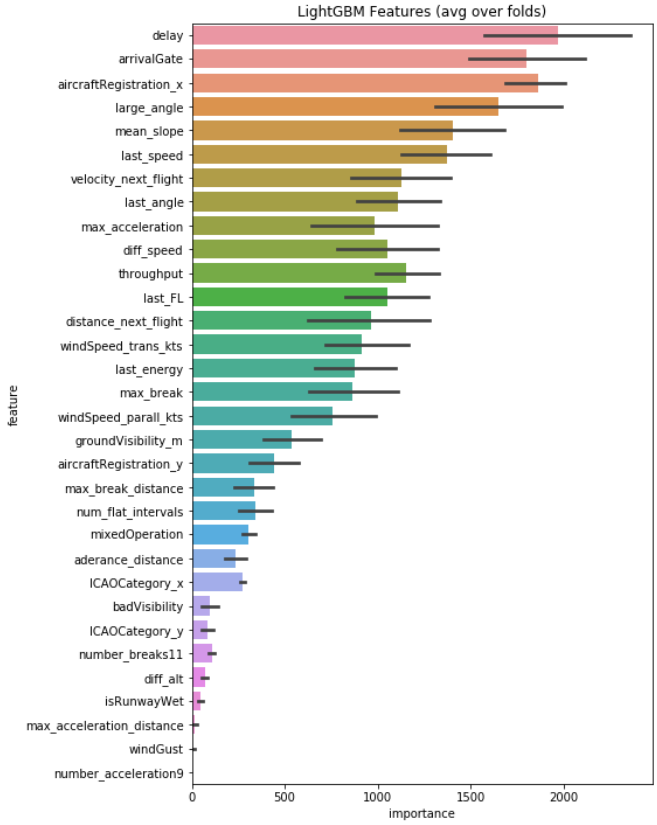

Figure 3. Feature importance when the prediction is performed at 2NM from threshold.

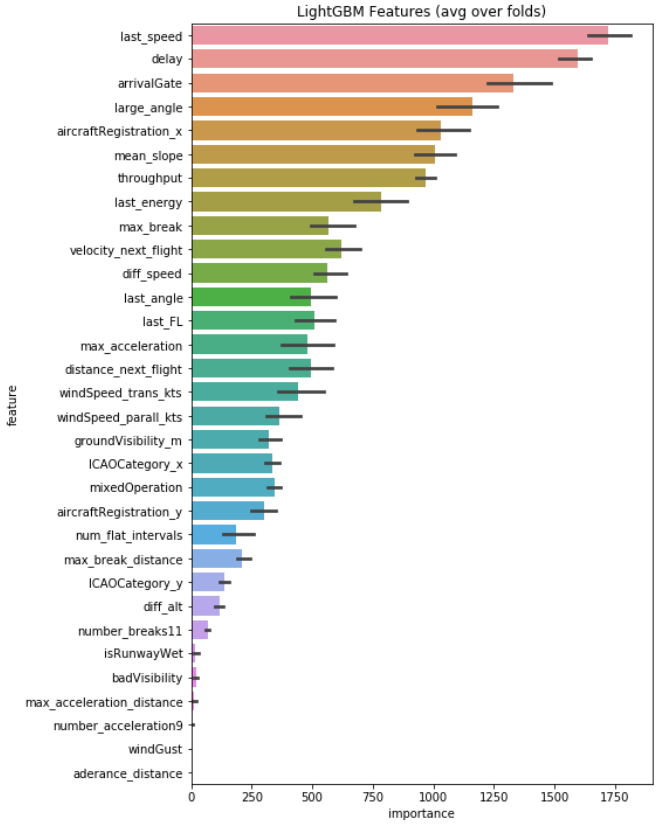

Figure 4. Feature Importance when prediction performed at threshold.

decrease even when predictions are made 5NM from threshold.

Such behaviour is two-fold: (a) the several static features (i.e. does not change during the course of a flight, such as the aircraft registration number) are highly ranked in the feature importance of the classification; (b) the values of the features (e.g. last speed) vary depending on distance of prediction. In this scenario, variability across all flights may stay equal given the low effect of said values on prediction, therefore providing an equally satisfactory classification. To push this further, all the features were discarded that have been extracted from dynamic parameters of the flight, restraining the model to a set of much reduced features (11 static features present in Table 2). The result of AUC of 0.718 is surprising. In other words, the addition of dynamic features only improves the prediction by 9%. Figure 7. shows how the model is still able to predict 96% of the flights following the AIP, but reduced its precision in the other class of flights by identifying only 14% of them in comparison to 35% found in Figure 2. The effect of dynamics features is much more appreciable this way but still too negligible to fully cover unexpected cases. Please refer to the discussion section for possible ways to improve this prediction.

B. Continuous case

Here, we wanted to directly forecast the runway Occupancy Time of the flight using the same features as before. light-

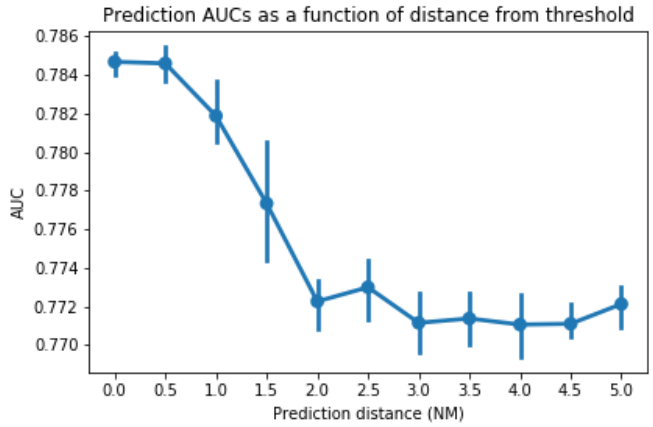

Figure 5. AUC as a function of prediction distance.

GBM and other tree decision-based algorithms also propose a solution for that. The forecast of a continuous variable involved performing not only the splitting of each feature, but a regression for each new sub-sample created. Based on that, a prediction has been obtained with a mean absolute error of 8 seconds. Taking into account that the mean ROT in LOWW is of 49.66 seconds with a standard deviation of 14.4, the results can seem satisfactory at first. Figure 8. shows how slightly more than 80% of the ROTs are predicted with less than 14

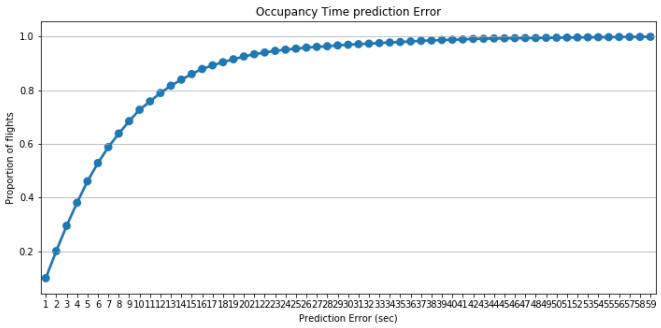

Figure 6. Cumulative Probability of Occupancy time prediction Error

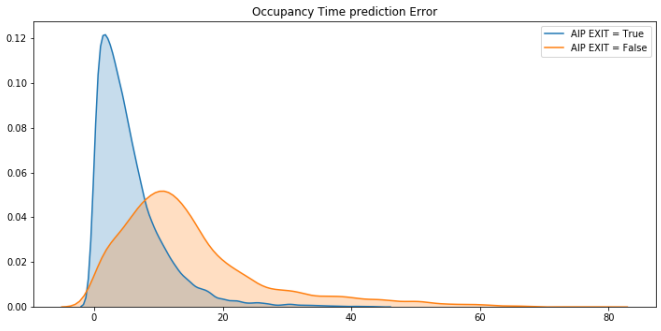

Figure 7. Absolute Error distributions

seconds of error, which is the standard deviation of the real ROT.

However, a closer look shows that this model suffers from the same limitations as the binary classification presented before. In post-analysis, dividing the flights into those that actually followed the AIP and those that didn't, the MAE passes from 5.7 seconds to 15 seconds for, respectively, a mean ROT of 45 and 62 seconds. In other words, the ratio error to value passes from 12.5% to 24%, underlining once again how the models struggle to correctly identify the flights not following the norm.

VI. Discussion

Within this contribution we pave the way towards a machine learning tool that would provide ATCOs a more accurate forecast of the behaviour of the landing flights. In the future, such a tool could help ATCOs confidently reduce the separation between consecutive flights and maximise the runway occupancy while keeping or improving its safety.

The first step consists of a quantitative assessment of the knowledge present in the data available to ATCOs. Based on their operational data, we have been able to train a model that accurately predicts (96%) when a flight will take an expected exit. However, the model can only spot 36% of flights that behave against the AIP specifications. When the same problem is formulated in a continuous way, the model is able to predict the ROT of the landing flight with an average absolute error of 8 seconds, which is reduced to an average absolute error of 5.7 seconds when the flight behaves as expected by the AIP and 15 seconds otherwise.

In light of the results, three improvements are to be developed further:

First, considering that the landing procedures are threefold : a) the flare manoeuvre, b) the point of the main gear touchdown to the point where the nosewheel touches down, and c) the ground braking distance and roll. ROT depends on the aircraft braking capability and on the pilots technique and preference, but ATCOs have no information regarding braking technique. The next step in SafeClouds.eu is to take advantage of the presence of Flight Data Monitoring information in order to complete the features with specific information regarding the pilot braking plans (e.g. AUTOBREAK status, whether spoilers have been armed before 2NM, etc.). If such information helps improve the prediction, the next step would be to find a proxy for these elements (e.g. if weight, available in FDM, is an important parameter for the model, data mining techniques can infer them from trajectories [20]). If such proxy is not possible, this will emphasize the need of more data sharing to the ATM/ATC community.

Second, it is still possible to improve the model presented in this contribution. We suspect a considerable amount of noise to be present in the data. For example, the filtering done (i.e. matching irregularities) might cause information such as the identification of the next landing flight to be spurious. In addition, the actual tool uses the estimated time of arrivals to spot the next landing flight. While such a solution helped limit the computer requirements of the pre-processing steps, such a gain comes at a cost as next flight identification might be erroneous. A searching algorithm should be added in order to ensure the next landing flight in real-time.

Finally, the data processed here do not reflect the reality of the operational environment of the ATCOs. ATCOs can request early exits from the pilots through radio communication without any recording of the event. The presented work has tried to create features that would identify such events (e.g. sudden and strong deceleration), however, the results are mitigated. FDM presents a specific feature which states the status of the radio communication that may help in that sense, but from the ATC perspective, new features should be found in order to better cover those cases.

These results and the rest of SafeClouds.eu project will hopefully pave the way towards a complete understanding of runway occupancy precursors and how to forecast them using available data. More airports are joining SafeClouds.eu to help improve the tool presented in this study. Such tools might be a good addition to the TBS protocol already tested and validated in several airports in Europe, such as Heathrow.

References

[1] R Horonjeff, RC Grassi, RR Read, and G Ahlborn. A mathematical model for locating exit taxiways. *Institute of Transportation and Traffic Engineering, University of California, Berkeley*, 1959.

[2] William E Weiss and JN Barrer. Analysis of runway occupancy time and separation data collected at la guardia, boston, and newark airports. Technical report, MITRE CORP MCLEAN VA, 1984.

[3] Terry A Ruhl. Empirical analysis of runway occupancy with applications to exit taxiway location and automated exit guidance. *Transportation Research Record*, (1257), 1990.

[4] Steven E Koenig. Analysis of runway occupancy times at major airports. Technical report, MITRE CORP MCLEAN VA METREK DIV, 1978.

[5] Byung Kim, Antonio Trani, Xiaoling Gu, and Caoyuan Zhong. Computer simulation model for airplane landing-performance prediction. *Transportation Research Record: Journal of the Transportation Research Board*, (1562):53–62, 1996.

[6] Antonio A Trani, AG Hobeika, BJ Kim, V Nunna, and C Zhong. Runway exit designs for capacity improvement demonstrations. phase 2: computer model development. 1992.

[7] Charles Morris, John Peters, and Peter Choroba. Validation of the time based separation concept at london heathrow airport. In *10th USA/Europe ATM R&D Seminar, Chicago*, volume 10, 2013.

[8] Hartmut Helmke, Ronny Hann, Maria Uebbing-Rumke, Daniel Müller, and Dennis Wittkowski. Time-based arrival management for dual threshold operation and continous descent approaches. In *8th USA/Europe ATM Seminar, Napa, CA, USA*, 2009.

[9] Milan Janic. Toward time-based separation rules for landing aircraft. *Transportation Research Record: Journal of the Transportation Research Board*, (2052):79–89, 2008.

[10] Vincent Treve Goce Nikolovski and Floris Herrema. Local tbs delay reduction effect on global network operations. In *2018 International Conference on Research in Air Transportation, Barcelona, Spain*, 2018.

[11] David L Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000):32, 2000.

[12] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[13] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[14] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[15] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.

[16] Jie Zhu, Ying Shan, JC Mao, Dong Yu, Holakou Rahmanian, and Yi Zhang. Deep embedding forest: Forest-based serving with deep embedding features. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1703–1711. ACM, 2017.

[17] Dehua Wang, Yang Zhang, and Yi Zhao. Lightgbm: an effective mirna classification method in breast cancer patients. In *Proceedings of the 2017 International Conference on Computational Biology and Bioinformatics*, pages 7–11. ACM, 2017.

[18] Eduardo Fonseca, Rong Gong, Dmitry Bogdanov, Olga Slizovskaia, Emilia Gómez Gutiérrez, and Xavier Serra. Acoustic scene classification by ensembling gradient boosting machine and convolutional neural networks. In *Virtanen T, Mesaros A, Heittola T, Diment A, Vincent E, Benetos E, Martinez B, editors. Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017); 2017 Nov 16; Munich, Germany. Tampere (Finland): Tampere University of Technology; 2017. p. 37-41.* Tampere University of Technology, 2017.

[19] Jin Huang and Charles X Ling. Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering*, 17(3):299–310, 2005.

[20] Junzi Sun, Joost Ellerbroek, and Jacco M Hoekstra. Aircraft initial mass estimation using bayesian inference method. *Transportation Research Part C: Emerging Technologies*, 90:59–73, 2018.