The following publication Khan, W. A., Ma, H. L., Chung, S. H., & Wen, X. (2021). Hierarchical integrated machine learning model for predicting flight departure delays and duration in series. Transportation Research Part C: Emerging Technologies, 129, 103225 is available at https://dx.doi.org/10.1016/j.trc.2021.103225

Hierarchical integrated machine learning model for predicting flight departure delays and duration in series

Abstract

Flight delays may propagate through the entire aviation network and are becoming an important research topic. This paper proposes a novel hierarchical integrated machine learning model for predicting flight departure delays and duration in series rather than in parallel to avoid ambiguity in decision making. The paper analyses the proposed model using various machine learning algorithms in combination with different sampling techniques. The highly noisy, unbalanced, dispersed, and skewed historical high dimensional data provided by an international airline operating in Hong Kong was used to demonstrate the practical application of the model. The result shows that for a 4-h forecast horizon, a constructive neural network machine learning algorithm with the Synthetic Minority Over Sampling Technique-Tomek Links (SMOTETomek) sampling technique was able to achieve better average balanced recall accuracies of 65.5%, 61.5%, 59% for classifying delay status and predicting delay duration at thresholds of 60min and 30min, respectively. Similarly, for minority labels, the precision-recall and area under the curve showed that the proposed model achieved better results of 32.44% and 35.14% compared to the parallel model of 26.43% and 21.02% for thresholds of 60min and 30min, respectively. The effect of different sampling techniques, sampling approaches, and estimation mechanisms on prediction performance is also studied.

Keywords: air traffic; aviation; flight delay prediction; high dimensional data; machine learning; sampling techniques.

1. Introduction

1.1 Background and Motivation

The aviation industry is growing rapidly because of the increasing demand for air transportation. In the aviation sector, passenger and cargo demands are increasing at an average rate of 7% and 4.43% each year respectively (IATA, 2019). Flight delays at airports may create an undesirable annoyance for passengers and cargo customers, possibly leading to a change to other means of transportation. Globally, in the year 2018/2019, international airlines contributed to an average of 21.19% flight departure delays (FlightStats, 2019). Such high departure delays may propagate through the entire aviation network (Du et al., 2018) causing economic loss to airlines in terms of having to pay high penalties. Another consequence is flight cancellations causing wastage of time and loss of opportunities (Alderighi and Gaggero, 2018). The problem of flight delays is decreasing passenger demand and simultaneously pressurizing airlines to raise airfares to accommodate the lower demand and increase in block time (Britto et al., 2012). Flight delays cost airlines not only by reallocating resources (Abdelghany et al., 2004) but also by paying higher compensation to passengers for demand sustainability (Hu et al., 2016). High rates of flight delays in the growing aviation industry motivate further study and the need to propose a reliable machine learning prediction model to facilitate airlines in making better-informed decisions.

1.2 Existing models limitation and Research questions

Prediction of flight delays and possible durations in a pre-defined forecast horizon can help airlines in the prompt execution of contingency plans to minimise penalty costs and loss of business opportunities. Flight delay prediction has gained significant research attraction in the last few years. Existing flight delay models have been suggested to predict flight delays above a certain threshold (Belcastro et al., 2016; Kim et al., 2016; Rebollo and Balakrishnan, 2014) from online available data and/or domestically operated flights (Belcastro et al., 2016; Du et al., 2018; Khanmohammadi et al., 2016; Kim et al., 2016; Rebollo and Balakrishnan, 2014; Tu et al., 2008; Yazdi et al., 2017; Yu et al., 2019). These prediction models are limited in scope and further improvements can bring a breakthrough contribution to existing studies. For the current prediction models, a practical application of a single threshold model may not generate sufficient information about delay duration, and implementing multiple threshold models in parallel may result in more than one decision option which may create ambiguities in actual decision making. The majority of studies on domestically operated flights rather than international flights may restrict the applicability of existing models to domestic flights delay prediction and make it unsuitable for international flights delay prediction. The airline operations planning for international flights is considered to be different compared to domestic flights (Wu, 2016). Further, less effort has been devoted to improving flight delay prediction using constructive neural networks (CNN). Inappropriate adjustments of the extensive hyperparameter for machine learning algorithms in complex high dimensional domain problems may cause the algorithm to converge at a suboptimal solution.

To overcome existing limitations, our study aims to address the following research questions: 1) What is the most suitable approach for learning and integrating multiple prediction models for forecasting the flight delay status and possible duration collectively in a hierarchical (or step-by-step) mode? 2) What estimation mechanism is more suitable for highly uncertain historical flight delays? 3) Which pre-processing, transformation and sampling techniques can be used to help to smooth the decision boundaries and improve the prediction accuracy of machine learning methods? 4) Can sampling both flight delay training and testing sets lead to the wrong decision? 5) How can linear dependence of input and hidden units in conventional neural networks and extensive hyperparameter adjustments in machine learning algorithms be eliminated to improve the flight delay prediction? 6) How to determine the number of hidden units in the hidden layers of deep learning?

1.3 Contribution and Novelty

An international airline operating in Hong Kong facing the above problems has been studied. The airline planned to have an integrated flight departure delay and possible duration prediction model embedded in their system for international flights. The high dimensional data of historically operated international flights among various sectors (or airports) for two years have been analysed. Several challenges that limit the applicability of the existing models need considerable attention. The study identified and addressed those challenges as a contribution and novelty to flight delay prediction:

First, in existing studies, in terms of classification, flight delays are predicted above a certain threshold (Belcastro et al., 2016; Kim et al., 2016; Rebollo and Balakrishnan, 2014). Each threshold prediction can be considered as a single model. Adopting a single threshold model may not generate sufficient information about how long the delay duration will last? Implementing multiple models in parallel above certain thresholds may result in more than a unique decision which may create ambiguities in actual decision making. For instance, the same dataset is used to learn the delay prediction models above different thresholds of 30min, 60min, 90min, and many others. In an actual scenario, if the 30min threshold model predicts a delay, the 60min model predicts a delay, and the 90min model predicts no delay, the airline may interpret that the possible delay duration is more than 30 minutes and less than 90 minutes. However, if the 30min threshold model predicts a delay, the 60min model predicts no delay, and the 90min model predicts a delay, then this may create ambiguity, e.g. is the delay greater than 30 minutes and less than 60 minutes or greater than 90 minutes? In such a case, the practical decision making may become difficult because the same dataset has been used to learn models above certain threshold values. Our study addresses this limitation and proposes a classification model, implemented in series rather than in parallel. In series, we propose to predict flight delays above certain thresholds step by step by considering only a portion of the dataset relevant to the threshold learning. For instance, the classifier will predict whether the flight is on-time or a delay will occur? If a delay is predicted, then the classifier will predict the possible delay duration using some initially defined threshold (e.g. 60 min) by considering only the delay dataset. If the delay time predicted is less than a defined threshold (i.e. 60min), then the classifier will further predict the delay time for the next threshold (i.e. 30 min) using a portion of the delay dataset other than an initially defined threshold (i.e. above 60 min). This approach can be applied to any number of thresholds and benefits from avoiding ambiguity in decision making.

Besides, in the existing literature, most of the flight delay models are proposed based on online available data and/or flights operated domestically (Belcastro et al., 2016; Du et al., 2018; Khanmohammadi et al., 2016; Kim et al., 2016; Rebollo and Balakrishnan, 2014; Tu et al., 2008; Yazdi et al., 2017; Yu et al., 2019). For the current study, the historical high dimensional data of actual operated international flights among the various sectors were provided by one of the international airlines operating in Hong Kong. Rebollo and Balakrishnan (2014) found that the routes served by airports comprise essential elements for flight delays. Compared to domestic flights, international flights involve various additional requirements and may face critical issues in airline operations planning such as complex ground operations and enroute operations. The ground operations involve activities at the landside and airside of an airport. The landside activities most often include passenger and baggage check-in, connecting passengers, cargo handling, passenger boarding, etc. The airside operations include disembarkation and embarkation, crew changing, maintenance, re-fueling, loading, and unloading. The landside and airside arrangement time can be one and a half hours and 15 to 20 minutes for domestic flights and up to three hours and one and a half hours to two hours for international flights, respectively. The ground turnaround operations are different for international flights than those performed for domestic flights due to the difference in aircraft types, on-board services, and security requirements. The enroute operations are usually carried out by the cockpit crew and further facilitated by an air traffic controller. Enroute operations are mostly beyond the control of the airlines except they can alter the flight plans in advance. Enroute operations are greatly affected by the geographical location such as inclement weather conditions, the situation at the departure and arrival airport, long and short-haul flights, and many others. Both turnaround operations and enroute operations have a significant effect on flight on-time-performance and profitability (Wu, 2016). The added requirements for a smooth journey in an international flight make it more valuable to study for flight delays. Our study proposes a novel hierarchical integrated model to predict flight departure delays and possible duration in series by considering a case study of an international airline operating in Hong Kong. This contribution mainly addresses research question no.01 in subsection 1.2.

Second, the flight delay task has been considered as either a regression (Tu et al., 2008; Yu et al., 2019) or classification (Belcastro et al., 2016; Kim et al., 2016) process or a combination of both (Rebollo and Balakrishnan, 2014). For regression, the highly skewed and dispersed historical dataset may make it challenging for regressors to correctly predict flight delays, whereas, for classification, the majority of labels belonging to one class may make it challenging for classifiers to correctly classify flight delays above a certain threshold. This study addresses the challenges of both regression and classification estimation mechanisms and recommends a suitable approach for flight delays prediction. The work mainly addresses the research question no.02 in subsection 1.2.

Third, in many cases, simple random over-sampling (Rebollo and Balakrishnan, 2014) and random under-sampling techniques (Belcastro et al., 2016) are recommended to balance the flight delay dataset to make it suitable for classification. In random over-sampling, the chances of overfitting increase because of using duplicate examples of the minority class, while, in random under-sampling, the chances of losing potentially useful data increases because of eliminating examples from the majority class (Batista et al., 2004). The current study applies various over-sampling, under-sampling and their combination techniques to balance the dataset and to make decision boundaries smoother to address research question no.03 highlighted in subsection 1.2. Among various sampling techniques, the one with high prediction accuracy is proposed for flight delay prediction. In existing works, sampling techniques are applied to both training and testing sets of the dataset. Differing from existing works, our study applies balancing techniques to the training set and measures the performance by comparing with the original testing set to address research questions no.04 highlighted in subsection 1.2. Using the original testing set for measuring performance may truly represent the real-world application of the flight delay model.

Fourth, among various machine learning methods, the backpropagation neural network (BPNN), support vector machine (SVM), and random forest (RF) have gained much attention in the airline domain (Evans et al., 2018; Xu et al., 2019). Extensive hyperparameter initializations in machine learning methods may require a lot of trial-and-error experimental work to determine the best optimal structure that has the capability of maximum error reduction. Excessive hyperparameter adjustment along with high dimensional data holding redundant information may cause existing algorithms to converge to a suboptimal solution (Wang et al., 2020). In this study, we propose CNN, named as hyperparameter-free cascade principal component least squares neural network (hyp-free CPCLS), having the characteristics of analytically determining the number of hidden units in hidden layers with no iterative tunning of connection weights. This makes it a novel self-organizing topology structure without involving trial-and-error experimental work. The contribution is that it requires minimal engineering experience and expertise to train the network because of no initialization and adjustment of the hyperparameters. Users do not need to define the number of hyperparameters such as learning rate, connection weights, network topology, the number of hidden layers, and the number of hidden units in each layer. This mainly addresses the research question nos. 05 & 06 highlighted in subsection 1.2. The proposed algorithm eliminates the problem of the hidden unit's coadaptation by generating linearly independent hidden units from the orthogonal linear transformation of the input variables to achieve the best least-squares solution. The results from Single layer BPNN (SL-BPNN), Deep layer BPNN (DL-BPNN), SVM, hyp-free CPCLS, their Ensemble, RF, gradient boosting decision tree (GBDT) and extreme gradient boosting (XGBoost) are analysed to select the most suitable method that has better flight delay prediction capability.

1.4 Major findings and Paper structure

The study addresses the limitation and research questions by the contribution and novelty to existing works. The major findings of the study are:

First, the hierarchical integration of the flight departure delay status and possible delay duration into a single model helps to eliminate ambiguity in decision making. The information flows in one direction (in series) which eliminates the need for implementing multiple models in parallel. The hierarchical integrated model works by generating information about the flight delay status, and if the delay is predicted, the delay duration is predicted at different thresholds. This finding support research question no. 01 and the first contribution.

Second, the nature of uncertainty in flight delay data may avoid the assumption of normality in regression and class balancing in classification. The pre-processing and transformation techniques to improve the highly skewed and dispersed data distribution for the regression mechanism does not have an advantage in improving the performance of the regressors. However, applying various sampling techniques improves the performance of the classifiers by balancing the classes and making decision boundaries smoother. Therefore, the study recommends the classification mechanism as a more suitable approach than regression. The finding addresses research question no.02 and the second contribution.

Third, the selection of sampling techniques depends upon the application area in improving the performance of the estimation methods. The study finds that among eight sampling techniques, the combination of oversampling and undersampling techniques named as Synthetic Minority Over Sampling Technique-Tomek Links (SMOTETomek) helps to achieve better flight delays prediction. The finding addresses research question no.03 and the third contribution.

Fourth, the study finds that the improper application of sampling techniques can lead to a false conclusion. Applying a sampling technique to balance the training set and validating the performance on the original testing set is considered to be a more favorable approach rather than applying a sampling technique to both the training and testing sets. The finding addresses research question no.04 and the third contribution.

Fifth, Orthogonal linear transformation of the input operational parameters and any pre-existing hidden units help to generate linearly independent hidden units, ensuring maximum error reduction at each layer. Similarly, analytically determining the number of hidden units in the hidden layer with no iterative tunning of connection weights along with self-organizing cascade architecture helps in reducing the need for extensive hyperparameter adjustments and human intervention. A comparative study with various machine learning estimation methods demonstrates that hyp-free CPCLS in combination with the

SMOTETomek sampling techniques is capable of handling the flight delay problem more accurately. This finding support research question nos. 05 & 06 and the fourth contribution.

The remaining structure of the paper is organized as follows: Section 2 presents the literature review. Section 3 explains the airline flight delay problem. Section 4 proposes a novel hierarchical integrated model. Section 5 describes machine learning methodologies for predicting flight delays and duration. Section 6 discusses the experimental work with managerial implications and future work. Section 7 concludes the study.

2. Literature review

Recently, flight delays have gained much attention from researchers due to the importance of the growing aviation industry. Controlling flight delays benefits airlines in reducing penalty costs and improving business opportunities. In existing studies, researchers mainly used optimization methods, network analysis, probabilistic models, statistical regression, and machine learning to study flight delays. Among various approaches, machine learning has gained much popularity in the last few years due to its ability to extract useful information from high dimensional data (Khan et al., 2019b, 2019c; LeCun et al., 2015; Tkáč and Verner, 2016).

To avoid flight delay propagation through the network and identify delay sources, Abdelghany et al. (2004) used a classical short path algorithm to project downline delays and generate alerts for crew/aircraft operation breaks. The delays reported due to a ground delay program (GDP), issued because of extreme weather, were investigated. The model showed the significant impact of GDP on total system delays. In the recorded GDP, aircraft and pilot issues appeared to be the main reasons for flight delays. Du et al. (2018) built a delay causality network (DCN) to understand flight delay propagation at the entire system level. The highest flight delay day, the bad weather day, was chosen to present the network analysis. The DCN topological analysis concluded that delay propagation is most likely to occur in the peak travel period. Large airports are more affected by the upstream airports as compared to downstream airports. The heavy air traffic flow indicates that some of the largest airports are helpful in reducing delay propagation. Moreover, the cause of delay propagation cannot be due to a fixed set of airports, as flight delays were also found to be significantly high for connected airport clusters.

The entire delay distribution of flights and the impact of airline policies on flight delays are mainly studied by using probabilistic models and statistical regression techniques. Tu et al. (2008) applied expectation-maximization (EM) by combining it with a genetic algorithm (GA) to estimate flight departure delays. The observed delays were decomposed into three components – seasonal trends, daily propagation patterns and random residuals to understand departure delays. Rather than a point estimate, the model was implemented to estimate the entire delay distribution. Yazdi et al. (2017) studied the linkage between imposing an airline baggage fee (BF) and flight delays. The results from implementing a 3-stage-least-square model (3SLS) concluded that a BF policy directly improves the on-time performance, through improvement in loading efficiencies and airport sorting, and indirectly through lower passenger demand. However, these improvements are highly influenced by the presence of hub airports, and travel types such as business or leisure. It was also concluded that prior implementation of BF (only first checked bag free of charge) resulted in more flight delays compared to the new BF policy (no checked bag free of charge).

The popularity of machine learning to predict flight delays is increasing due to its better learning ability from the available data. Rebollo and Balakrishnan (2014) suggested the random forest (RF) approach to predict departure delays 2-24 hours in the future. In addition to local variables, new network delay variables characterizing the global delay state of the entire system were studied. The effect of varying forecast horizons with a threshold of 60min and the effect of varying thresholds with a time horizon of 2 hours were studied. To improve the performance of BPNN, Khanmohammadi et al. (2016) proposed a new type of multilevel input BPNN for minimizing airport traffic. The study suggested prioritizing arriving flights for landing based on delays. The landing priority of flights is based on the scheduled arrival time and the planned priority needed to change (depending upon airport management strategies) if flight delay is predicted. Belcastro et al. (2016) developed a scalable parallel version of RF to predict the arrival delay of scheduled flights due to weather conditions. A range of experimental work was performed to understand the arrival delay for individual flights by considering different arrival and departure weather conditions, varying delay thresholds and varying target datasets. The delay was classified by using thresholds of 15min, 30min, 45min, 60min and 90min. Kim et al. (2016) implemented the Long Short-Term Memory (LSTM) Recurrent neural network to predict flight arrival and departure delays, using on-time performance and weather data, by adopting a two-stage approach. In the first stage, the daily delay status was predicted. In the second stage, the individual flight delay was predicted from daily delay stage output information. Thresholds of 15min and 30min were used to classify delay output. Yu et al. (2019) employed a deep belief network and support vector machine (DBN-SVM) to predict flight delays by considering both macro and micro level influencing factors. Based on prediction results, among multifactor (macro and micro level), the key factors having the most influence on flight delays were identified as delay propagation, the air route situation, and airport crowdedness.

The application of BPNN is gaining significant interest in improving various operations of airlines, such as fuel estimation (Baklacioglu, 2016; Trani et al., 2004), trajectory prediction (Gallego et al., 2019; Zhang and Mahadevan, 2020), delay prediction (Khanmohammadi et al., 2016), improving customer satisfaction (Lin and Vlachos, 2018), and many others (Chung et al., 2017; Cui and Li, 2017). The benefit of BPNN is that it is theoretically proven to follow a universal approximation theory (Ferrari and Stengel, 2005; Z. Wang et al., 2019) and can approximate any continuous function. However, the initialization and adjustment of connection weights, hidden units, activation function, and learning rate hyperparameters in BPNN have a significant effect on network performance. It is considered that BP learning is generally more time consuming because of the iterative tuning of the connection weights. The iterative tuning may increase the complexity of the network by creating a complex coadaptation among the hyperparameters causing the network to be slow and converge at a local minimum rather than the global minimum (Huang et al., 2006; Krogh and Hertz, 1992; Liew et al., 2016; Srivastava et al., 2014). The optimal BPNN architecture is not always obvious and a lot of trial-and-error experimental work is needed to select the best possible network topology having a significant number of hidden units in the hidden layer. The survey of Y. Wang et al. (2019) on utilization of deep learning to enhance the intelligence level of transportation system concluded the shortcoming of deep learning is that it requires greater engineering experience and expertise to determine the number of hyperparameters, such as the number of hidden layers and number of hidden units in each layer of NNs. Tkáč and Verner (2016) reviewed applications of NNs in business, such as financial analysis, costs monitoring, sales, marketing, decision support, bankruptcy, and many others, summarizing that majority of existing works are focused on determining the number of hidden units and hidden layers using a trial and error approach and there is no guarantee that chosen settings are the best. Initialization and adjustment in the number of hidden units and hidden layers in BPNNs significantly affect the performance of the network (Cranenburgh and Alwosheel, 2019). Among the various means, the most popular way of determining the number of hidden units and hidden layer in BPNN is by trial-and-error experimental work or rule of thumb (Hamad et al., 2017; Xiao et al., 2016). To overcome the problem, various NNs with random weight (NNRW) was proposed by adding randomly generated hidden units in the single layer of the network. However, existing work is focused on a single layer and it is also considered that the randomization of hidden units cannot guarantee the optimal performance of NNs (Cao et al., 2018). How to determine the number of hidden units in hidden layers of deep learning is still an open problem and needs considerable attention. Similarly to BPNN, the application of SVM and RF has also shown considerable improvement in improving airline operations (Evans et al., 2018; Rebollo and Balakrishnan, 2014; Yu et al., 2019). The proper selection of hyperparameters such as the soft margin regularization term and kernel function in SVM, and decision trees in RF, GBDT and XGBoost is important to achieve better network performance. The extensive hyperparameter initialization and adjustment in BPNN, SVM, RF, GBDT and XGBoost need user expertise which may greatly affect algorithms performance and convergence rate.

The contributions of researchers for predicting flight delays are noteworthy. Most existing studies focused on predicting flight delays above a certain threshold from publicly available domestic flights. According to the best of our knowledge, none of the earlier studies suggested flight delay hierarchical integrated prediction models for historical internationally operated flights using a CNN.

3. Problem Explanation

The purpose of this work is to propose a novel model for predicting departure delays and duration in series for an airline. In this study, data were obtained from the major international airline operating in Hong Kong to validate the proposed model. Departure delay occurs when an aircraft takes-off later than the scheduled time due to certain reasons. Before each flight, a flight plan is prepared giving details of various operational parameters needed for the smooth operation of the aircraft. For the selected airline, the flight plan is prepared four hours before each international flight. The flight dispatcher, responsible for preparing the flight plan, obtains information from various functional departments about weather conditions, air traffic flow, aircraft performance, and many other factors for defining the optimal flight trajectory and ensuring smooth flight operation. Flight delays may significantly affect the normal operations of the airline and its great importance during the preparation of the flight plans cannot be denied. The study is focused on predicting airline flight departure delays and possible duration for a four hours forecast horizon from available operational parameters information. This will assist the airline in predicting flight delays four hours in advance of scheduled flights. The airline will be able to make a more informed decision by planning for eliminating flight delays impact. For instance, if the airline predicts a delay of 30 minutes, then they may plan for some other alternatives (for example, another possible route, higher cost index, etc) to reach the arrival airport in a timely manner to avoid aircraft rotation or passenger/load connection delays.

The airline categorizes its departure delays into nine categories, as defined by the International Aviation Transport Authority (IATA). The categories are numbered 1-9 with alphabetic/numerical codes defining the delay reason in each category. The categories explaining various departure delay reasons for the airline are (Eurocontrol, 2020; Wu and Truong, 2014):

1. *Passenger and Baggage:* The codes in this category are used to describe the delay reasons caused by late passenger and improper baggage handling. For instance, reasons reported by the airline are missing check-in passenger, baggage processing or sorting and many others.
2. *Cargo and Mail:* The codes in this category are used to describe the delay reasons caused by inadequate cargo activities and improper mail handling. For instance, inadequate packaging and many others.

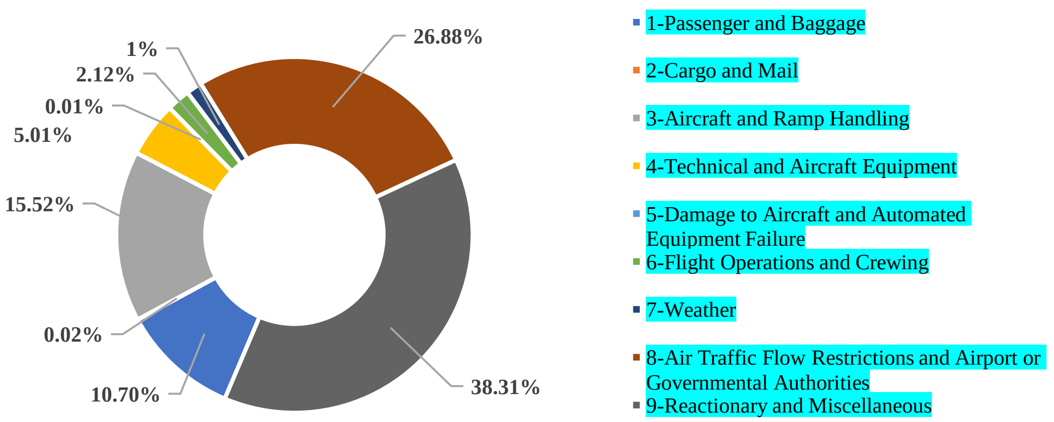


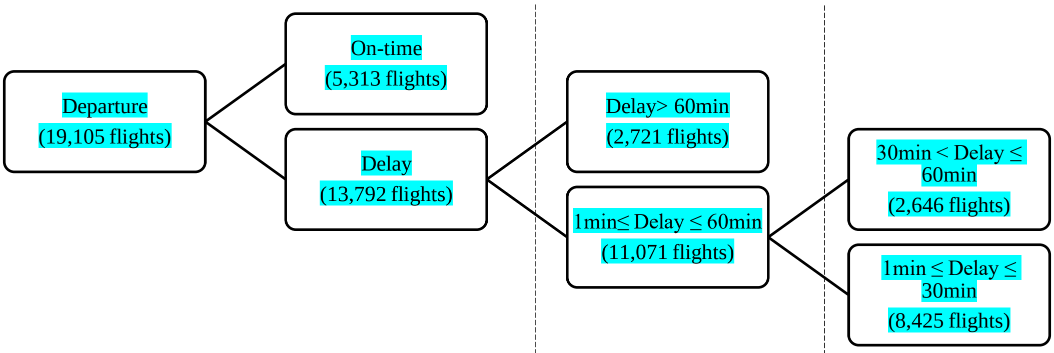
Fig. 1. Percentage contribution of each category causing airline departure delays

3. *Aircraft and Ramp Handling:* The codes in this category are used to describe the delay reasons caused by improper handling of aircraft and ramp/apron area. For instance, reasons reported by the airline are aircraft cleaning, catering late delivery, incorrect load sheet, fuelling and defueling, and many others.
4. *Technical and Aircraft Equipment:* The codes in this category are used to describe the delay reasons caused due to technical issues and lack of aircraft equipment. For instance, reasons reported by the airline are aircraft defects, late release of aircraft from scheduled maintenance and many others.
5. *Damage to Aircraft or Automated Equipment Failure:* The codes in this category are used to describe delay reasons due to damage to the aircraft and automated equipment failure. For instance, the computer system down and many others.
6. *Flight Operations and Crewing:* The codes in this category are used to describe delay reasons caused because of late flight operations and crew scheduling/shortage. For instance, reasons reported by the airline are late completion of the flight plan, late crew boarding and departure, fuel altering and many others.
7. *Weather:* The codes in this category are used to describe delay reasons caused by bad weather conditions. For instance, reasons reported by airlines are ground handling because of adverse weather, alternative route shifting, and many others.
8. *Air Traffic Flow Restriction and Government Authorities:* The codes in this category are used to describe delay reasons caused because of air traffic control restrictions and aircraft or government authorities' requirements. For instance, reasons reported by the airline are a restriction at the destination airport, inadequate airport facilities, runway restriction at the origin airport, mandatory security check-up, airway traffic, and many others.
9. *Reactionary and Miscellaneous:* The codes in this category are used to describe delay reasons for reactionary and miscellaneous reasons. For instance, reasons reported by the airline are late arrival of aircraft, crew rotation, passenger/load connection, check-in error, and many others.

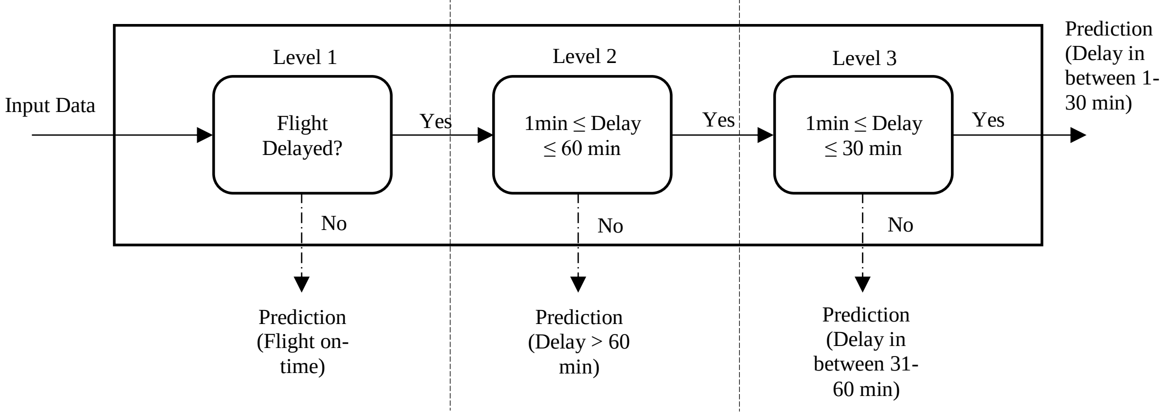
Fig. 1 illustrates the percentage contribution of each category causing airline departure delays. The analysis shows that the highest contributors to departure delays are reactionary and miscellaneous reasons (38.31%), followed by air traffic flow restriction and government authorities' requirements (26.88%). Other categories also make significant contributions to delays with the lowest contribution recorded for cargo and mail mishandling (0.02%), and damage to aircraft or automated equipment failure (0.01%). In many cases, the category that is considered to be the top reason for flight delays is bad weather conditions. However, the analysis shows that bad weather directly contributes to only 1.43% of the total delays. Belcastro et al. (2016) mentioned that the weather is a crucial factor in studying flight delays in that it may adversely affect and may become a source of other delay reasons. The analysis of categories helps in selecting the relevant operational parameters, from two years of flight operational data provided by the airline, for predicting flight departure delays and possible durations. The work benefits in considering flight delays recorded from all categories rather than one or a specific category.

4. Proposed novel hierarchical integrated model

Existing departure delay models above certain threshold values may cause the airline to adopt a single threshold model or more than one threshold model in parallel which may not provide enough information about departure delay duration. The highly dispersed and skewed historical data of internationally operated flights may make it challenging for regressors to truly approximate actual departure delays, whereas, the class imbalance and boundaries overlapping issue may make it challenging for the classifiers to accurately classify the class labels. The uncertainty in historical flights together with extensive hyperparameter adjustment of machine learning estimation methods may cause the algorithm to converge at a suboptimal solution, resulting in low performance of the flight delay model.



(a)



(b)

Fig. 2. (a) Representation of flight departure data used at each hierarchical level, (b) Hierarchical integrated model for predicting flight delay status

To overcome the limitations and facilitate airlines to make informed decisions, we propose a novel hierarchical integrated model. We propose to predict the flight departure delay and duration step-by-step in series (or hierarchical). Instead of training the machine learning algorithm on all datasets, the information relevant for the next threshold is extracted from the dataset. The hierarchical levels can be extended to N number of levels $(level-1, level-2, \ldots, level-N)$ depending upon user requirements. For the sake of simplicity and a better understanding of findings, we plan to predict flight departure delay status and duration at three hierarchical levels: level-1, level-2, and level-3. If the flight is on-time, the proposed model will predict on-time status, however, if the flight experiences a departure delay in the future, the model will predict the possible delay duration. Level-1 is proposed to predict flight delay status, whereas level-2 and level-3 are proposed to predict flight delay duration at thresholds of 60min and 30min, respectively. The three levels are integrated into a single hierarchical model to improve the decision-making process. Fig. 2 illustrates the flight delay hierarchical integrated model and dataset used for each level prediction. The working mechanism of the proposed model can be explained in the following steps:

a) For a given flight, level-1 will check whether the flight will experience a future departure delay? If no, it will indicate that the flight will depart on-time and the model will predict the on-time class label. If yes, the model will predict the delay status and will check the possible delay duration.
b) For a possible departure delay duration, the model will initially check for a longer delay. Level-2 will check whether the delay will be between 1 minute to 60minutes? If no, it indicates that the flight may experience a delay of more than an hour. If yes, it indicates that the delay will be less than or equal to an hour. The model will further hierarchically check for narrowed delay duration to make a better-informed decision.
c) Level-3 will facilitate in identifying the narrow delay duration. The level-3 will check whether the delay will be between 1 minute to 30minutes? If no, it indicates that the flight may experience a delay of between 31 minutes to an hour. If yes, it indicates that the delay will be less than or equal to 30 minutes.

Step (a) or level-1 is a prerequisite because it helps to identify flight status, whereas, steps (b), (c) or levels-2&3 depend on user requirements for certain thresholds. Depending upon airlines needs, the hierarchical integrated model for flight departure delay status and duration can be extended to any number of thresholds and levels to facilitate them in informed decision making.

5. Machine learning methodologies

In the modern competitive era, the popularity of machine learning is increasing because of its ability to make more informed decisions compared to traditional statistical techniques (Kumar et al., 1995; Tkáč and Verner, 2016). The range of applications, not limited to, includes connected flights buffer time estimation (Chung et al., 2017), traffic prediction (Cui et al., 2020), aircraft boarding prediction (Schultz and Reitmann, 2019), trajectory prediction (Khan et al., 2021), enhancing the intelligence level of the transportation system (Y. Wang et al., 2019) and many others. Various types of machine learning are supervised, unsupervised, and reinforcement.

To achieve the objective of predicting flight departure delays, the supervised machine learning approach is adopted. The dataset, provided by the airline, contains both the input operational parameters and the desired output of the flight departure delay. Using any single algorithm may bias the prediction. Different learning algorithms, having the ability for both regression and classification, are tested to select a model having better prediction performance. The algorithms (or estimation methods) tested are:

5.1 Backpropagation Neural Network

BPNN is a type of feedforward neural network (FNN) that does not create a cycle or loop (Hecht-Nielsen, 1989). All the information flows in the forward direction and the concept originates from human brain neuron functioning (Baklacioglu, 2016). It consists of processing elements (or hidden units) interconnected by channels known as connection weights. It learns by adjusting the connection weights between hidden units and attributes.

5.2 Novel hyperparameter-free Cascade Principal Component Least Squares Neural Network

Among the early attempts, the Cascade correlation learning algorithm (CasCor) was proposed to address the learning issues of the fixed topology BPNN (Fahlman and Lebiere, 1990). CasCor is a type of CNN and works by adding hidden units one by one to the network. The benefit of CasCor is that it does not need trial and error work to find the network architecture and experimental work concluded that the learning speed is faster than BPNN. Due to the growing interest in CNNs, researchers are making continuous efforts to improve the existing CasCor (Huang et al., 2012; Nayyeri et al., 2018; Qiao et al., 2016). To improve the performance and convergence of CasCor and its variant, an algorithm named Cascade Principal Component Least Squares Neural Network (CPCLS) was proposed (Khan et al., 2019a). CPCLS analytically calculates connection weights rather than iterative tuning and improves the existing cascade architecture by adding linearly independent multiple hidden units, rather than one by one, having the capability of maximum error reduction. This may avoid generating redundant hidden units and converges smoothly.

For a given training dataset (x_i, y_i) with N samples, where input units $x_i \in R^n, i = 1,2, \dots \dots, l$ and output unit $y \in R^l$, such that $y_i \in \{1,0\}$, CPCLS define an only one hyperparameter, i.e. $h_i, i = 1,2, \dots \dots, k$, the number of hidden units to be generated in each hidden layer. There can be multiple hidden units in each layer such that $k \le l$. CPCLS is Initialized with the number of h (N^h) in first hidden layer H. For input connection weight w, it orthogonally linear transforms x into linearly independent h by eigen decomposition of x covariance square matrix S:

$$S = \frac{1}{N-1}(\boldsymbol{x} - \overline{\boldsymbol{x}})^T(\boldsymbol{x} - \overline{\boldsymbol{x}}) \tag{1}$$

The eigenvalues λ are determined and those having the highest value corresponding to the eigenvector. The selected N^h eigenvector are considered as w:

$$|S - \lambda I| = 0 \tag{2}$$

$$(S - \lambda I)w = 0 \tag{3}$$

h is determined by taking nonlinear activation of the product of x and w with added bias b:

$$h = g(w^T x + b) \tag{4}$$

Generating non-redundant and linearly independent h by orthogonal linear transformation assures the maximum error reduction capability of H. The H explaining maximum variance in the dataset becomes more linear in the relationship with y. This facilitates in calculating output connection weight β by ordinary least squares:

$$\beta = (h^T h)^{-1} h^T y \tag{5}$$

The $\hat{y}$ is determined by linearly transferring h through β:

$$\hat{y} = \beta^T h \quad (6)$$

The network error E is determined and the algorithm is stopped if $E < e$, else another H is added in the network, defining new $N^{h\prime}$ such that $k \leq l$. Where e is a predefined error. For the proceeding H_n, it receives w from all x and pre-existing H_{n-1}. To avoid linear dependencies among H, only the newly added H_n is connected to y and diminishes the previous connection of pre-existing H_{n-1} to y. The pre-existing H_{n-1} becomes part of x, such that:

$$x = (x, H_{n-1}) \quad (7)$$

$$N^h = N^h + N^{h\prime} \quad (8)$$

where N^h is the amount of h for the proceeding H_n. The steps (1) to (6) are repeated and $\hat{y}$ is predicted until $E < e$.

The CPCLS can generate non-redundant h and H, which ensure maximum error reduction with smooth convergence. The generation of multiple h and H makes CPCLS a deep learning method. The network typology is determined by self-organizing h and H rather than fixed defining. This requires human intervention to determine the number of hidden units h in hidden layers H by trial-and-error experimental work. We propose novel hyp-free CPCLS to eliminate the need for initialization and adjustments of hyperparameter by trial-and-error experimental work. The number of hidden units in each hidden layer can be determined by sorting λ from largest to smallest values:

$$\lambda = \lambda_1 > \lambda_2 > \lambda_3 > \cdots > \lambda_n \quad (9)$$

Calculate the percentage variance $V(\%)$ explained by each λ:

$$V(\%) = \frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i} * 100\% \quad (10)$$

Calculate the cumulative percentage variance $CV(\%)$, such that:

$$CV(\%)_i = \sum_{j=1}^{i} V(\%)_j \quad (11)$$

Assign the number of hidden units N^h in the hidden layer such that $CV(\%)$ is less than 99.99%:

$$\begin{gathered} Let\ initial\ N^h = 0 \\ while\ CV(\%)_i < 99.99\% \\ N^h = N^h + 1 \end{gathered} \quad (12)$$

It is recommended to keep the number of hidden units in each layer such that $CV(\%)_{i-1} < CV(\%)_i$ and stop when $CV(\%)_i \approx CV(\%)_{i+1}$. The latter condition implies that the last few λ may explain lesser $V(\%)$ which may not be helpful in generating hidden units' parameter w that can extract useful information from the dataset and will increase network complexity. The value of $CV(\%)$ become constant at 100% for each sorted λ. In that case, $CV(\%)_i \approx CV(\%)_{i+1}$ and hence $V(\%)_i \approx V(\%)_{i+1}$. Therefore, for better generalization performance, the number of hidden units in each layer should be selected that has $CV(\%)_i < 99.99\%$. The characteristics of hyp-free CPCLS is that it has a self-organizing topology network and can determine the number of non-redundant hidden units in each hidden layer with no iterative tunning of connection weights. This makes it a novel approach with no initialization and adjustment of hyperparameters. The CPCLS and hyp-free CPCLS algorithms are shown in Appendix A. In literature, the application of CNN for predicting flight delay and duration is hardly explored. The advantages of hyp-free CPCLS motivate us to study for flight delay and duration prediction.

5.3 Support Vector Machine

The support vector machine (SVM) objective is to define the decision boundary (hyperplane) with the best separate cases of different class labels (Cortes and Vapnik, 1995; Evans et al., 2018). The optimal hyperplane is the one having maximum distance from the nearest data points (also known as support vectors). SVM may easily and correctly classify the linearly separable cases into different labels by finding a hyperplane that maximizes the margin, however, for linearly non-separable cases, finding a hyperplane that classifies all cases to their label might become a difficult task. SVM addresses the issue of linearly non-separable cases by introducing the concept of the soft margin and kernel trick. Various forms of the kernel are linear, polynomial, radial bias function, and sigmoid.

5.4 Averaging/voting Ensemble Learning

Table 1
Statistical analysis and distribution test of the train:test dataset

Data Split	N	on-time	Delay	Mean	Stdev	Min	Q1	Med	Q3	Max	IQR	Skew	Kur	KS Test
	No.	No.	No.	min	min	min	min	min	min	min	min	min	min	Sig.
Train	9552	2654	6898	34.08	84.85	0	0	11	34	2115	34	9.48	144.03	0.996
Test	9553	2659	6894	33.55	84.14	0	0	11	35	1998	35	9.58	144.71	
Original dataset	19105	5313	13792	33.81	84.50	0	0	11	35	2115	35	9.53	144.34	---

Ensemble learning is a method that combines multiple classifiers/regressors to generate results with better prediction and less variability (Dietterich, 2002). The voting and averaging ensembles technique simply train different classifiers/regressors and combine the prediction through voting in classification and averaging in regression.

5.5 Random Forest Ensemble Learning

Random forest (RF) is an ensemble technique and consists of many decision trees (Breiman, 2001; Rebollo and Balakrishnan, 2014). The input to each decision tree is the sampled data from the given dataset. RF collects the prediction results of each tree and chooses the most voted result (for classification) or average result (for regression) as the final prediction result.

5.6 Gradient Boosting Decision Tree and Extreme Gradient Boosting Ensemble Learning

Gradient Boosting Decision Tree (GBDT) and Extreme Gradient Boosting (XGBoost) are ensemble tree methods and follow the principle of gradient boosting (Chen and Guestrin, 2016; Friedman, 2001). Trees are added sequentially and trained by a gradient optimization algorithm to correct prediction errors made by the prior trees. This results in creating a strong classifier from the number of weak classifiers.

6. Numerical experimental work

In this paper, we consider the data obtained from the international airline operating in Hong Kong as a case study. In supervised learning, two main types of commonly known estimation mechanisms are regression and classification. For regression, the flight delay output is predicted in a continuous form. For classification, the flight delay output is classified in a binary form. In actual applications, the collected data are highly noisy, unbalanced, dispersed, and skewed which may greatly affect machine learning algorithms prediction capability. The highly skewed and dispersed data may make it challenging for the regressors to predict the flight delay, whereas, highly unbalanced and noisy data may make it challenging for the classifiers to classify flight delays. To study and overcome these challenges, both types of estimation mechanisms are employed.

Prior to applying estimation methods, the dataset was normalized to reduce the magnitude of the data to a common scale, helping to give equal weight importance to each attribute in the dataset. The SL-BPNN, DL-BPNN, SVM, hyp-free CPCLS, their Ensemble, RF, GBDT and XGBoost estimation methods were applied for both regression and classification estimation mechanisms. The ensembles (Subsection 5.4) refer to averaging (for regression) or voting (for classification) results of SL-BPNN, DL-BPNN, SVM, and hyp-free CPCLS to check whether their combined prediction can improve results. All the numerical experimental work was carried out in Anaconda Spyder Python v3.2.6 programming language. The BPNN, SVM,

Table 2
Data attributes for predicting flight departure delay

Type	Attributes	Variables
Airport	Origin, destination, alternative	34 binary variables
Aircraft details	Type	10 binary variables
	Registration	107 binary variables
Flight schedule	Departure (month, day, hour, minutes)	4 continuous variables
	Arrival (month, day, hour, minutes)	4 continuous variables
	Flight duration (hour, minutes)	2 continuous variables
	Week day	1 continuous variable
Weather	Wind Speed	1 continuous variable
	Wind direction	1 continuous variable and 2 binary variables
	Atmospheric pressure	1 continuous variable
	Outside air temperature	1 continuous variable
	Temperature deviation (ground and air)	2 continuous variables
Air traffic	Altitude (initial and final)	2 continuous variables
Runway configuration	Runway Direction	1 continuous variable and 6 binary variables
	Runway Surface	33 binary variables
Flight operation	Ramp weight	1 continuous variable
	Speed	1 continuous variable
	Engine Performance	1 continuous variable
	Distance	1 continuous variable

Table 3
Estimation techniques prediction results and absolute error

		Mean	Stdev	Q1	Med	Q3	Max	IQR	Skew	Kur	R
Prediction (min)	SL-BPNN	38.58	34.42	4.24	34.18	62.03	182.29	57.78	0.68	-0.23	0.01
	DL-BPNN	35.70	15.18	24.76	33.70	45.33	98.32	20.57	0.53	0.10	0.16
	SVM	37.84	25.57	20.31	35.77	51.94	319.60	31.62	1.37	6.65	0.15
	hyp-free CPCLS	33.95	21.43	18.57	30.46	45.79	221.17	27.22	1.21	3.28	0.19
	Ensembles	34.83	15.59	24.57	33.68	43.75	160.00	19.18	0.82	2.68	0.15
	RF	34.48	26.87	20.94	26.19	38.83	472.84	17.88	4.73	39.64	0.16
	GBDT	34.32	24.23	21.93	27.73	38.87	718.15	16.94	6.86	105.73	0.16
	XGBoost	34.80	34.05	17.78	25.57	40.19	892.56	22.41	6.30	85.97	0.17
Absolute Error (min)	SL-BPNN	47.16	78.20	12.18	32.80	60.16	1931.29	47.98	10.17	165.62	---
	DL-BPNN	38.22	73.77	13.80	25.88	41.67	1915.53	27.87	11.75	203.04	---
	SVM	39.31	74.68	12.59	26.38	44.49	1927.16	31.90	11.42	195.64	---
	hyp-free CPCLS	36.37	74.40	10.77	22.26	39.86	1905.90	29.09	11.50	196.45	---
	Ensembles	37.26	74.31	13.49	25.51	38.94	1919.97	25.44	11.73	201.37	---
	RF	36.60	75.75	12.07	20.67	35.25	1963.25	23.19	11.15	190.26	---
	GBDT	36.42	75.38	13.41	21.39	34.94	1951.44	21.53	11.37	193.88	---
	XGBoost	36.57	77.00	10.43	19.04	35.68	1929.80	25.24	10.41	168.61	---

RF, and GBDT were optimized using the scikit-learn module and XGBoost was optimized using the XGBoost module. The stochastic gradient descent learning algorithm with a sigmoid activation function in the hidden layer was used to train BPNN. Among different trials and error experimental work, SL-BPNN with 20 hidden units and DL-BPNN with 25 hidden units in the first layer and 10 hidden units in the second layer was considered as a best-fixed topology network. For SVM, the best combination of C, γ hyperparameters with kernel activation function was searched using grid search scikit-learn python tool, for instance, $C\epsilon\, \{0.005, 0.05, 0.5, 1, 5, 10, 20, 50, 100, 200, 300, 500, 800, 1000\}$ and $\gamma\ \epsilon\ \{1/l, 1/(l * x.var(\quad))\}$. The grid search returned $C = 300$ and $\gamma = 1/l$ as the best optimal hyperparameters, with greater accuracy. For RF, the number of trees was set to 100Nos. with criterion was set to entropy having greater accuracy. For GBDT and XGBoost, the number of trees of 100Nos. and the learning rate of 0.01 was selected as the best optimal hyperparameters, with greater accuracy. Like BPNN, the sigmoid activation function was used in the hidden layers of hyp-free CPCLS. For both estimation mechanisms, the dataset was split 50:50 for training and testing of the estimation methods. Table 1 shows the descriptive statistics of train and test data split. The N, Mean, Stdev, Min, Q1, Med, Q3, Max, IQR, Skew, Kur, and KS Test refers to a number of examples, mean, standard deviation, minimum, first quartile, median, third quartile, maximum, interquartile range, skewness, kurtosis, and two-sample Kolmogorov-Smirnov test. The statistical analysis and KS tests in the table show that both training and testing datasets are from the same distribution. The KS Test having a significance value of 0.996 greater than the significance level of 0.05 recommends accepting the null hypothesis by explaining that the distribution of the training and testing dataset is the same. This gives insight that both are representative of the original dataset. The training dataset has all relevant examples for the effective mapping of input to outputs. Similarly, the testing dataset also has all relevant examples for evaluating the model performance. To ensure better comparison, the same split dataset was used for the estimation methods.

This section is organized as follows: Subsection 6.1 describes the historical data provided by the international airline for predicting departure flight delays and possible duration. Subsection 6.2 presents and discusses the prediction results of estimation methods applied for both estimation mechanisms.

6.1 Data source and pre-processing

The historical data provided by the airline for flight delay prediction comprises 19,105 international passenger and cargo flights. The actual flights were performed over two years, from April 2015 to March 2017, covering eight international OD (or sectors) airports. In total, 107 widebody aircraft (Airbus A330-300 and Boeing 747-400/747-800/777-300) were operated. The data contain information for individual flights in terms of airports, aircraft, flight scheduled dates and times, weather information, runway configuration, air traffic control, and flight operational details. Table 2 provides information about the data attributes used for predicting departure delays. For continuous variables, the data were normalized in the range [0,1], whereas for the categorical variables, one hot encoding pre-processing technique was applied to create a binary vector for each category. The attributes were selected based on information provided by the airline and the importance to each delay reason category as classified in Fig. 1.

6.2 Departure delays prediction

In this subsection, the departure delay prediction results are presented for both regression and classification estimation mechanisms. Various challenges in the historical departure delays were studied, and possible solutions were recommended to select the best optimal model. Subsection 6.2.1 concerns predicting delays by considering the task as regression, and Subsection 6.2.2 concerns predicting delays by considering the task as a classification problem.

6.2.1 Delay prediction as a regression problem

For the regression task, the objective is to minimize the difference between actual and predicted delays. Mean absolute error (MAE), an objective function, is calculated by taking the mean absolute difference between actual and predicted delays. The objective function is expressed as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{13}$$

The measurement scale of MAE is identical as y. The y is flight delay variable measured in minutes (min). For each flight, the historical data with the value of zero (min) means flight departed on-time and values greater than zero means departure delay faced by that flight.

Various statistical analyses were performed to get a better insight about the estimation methods, in order to recommend an estimation method capable of truly approximating actual flight delays. Table 3 shows the descriptive statistics of estimated flight delays and their absolute error. The statistics in Tables 1 & 3 help to understand the central tendency, dispersion, and spread of the actual flight delay and the results generated by the estimation methods. The R value refers to the correlation relationship between actual and predicted flight delays. The actual flight delay (Table 1) mean value of 33.55 min with Stdev 84.14min, skew 9.58min, and kur 144.71min indicates that the data is highly dispersed, right-skewed, and leptokurtic. Although the mean values of the estimation methods, for instance, SL-BPNN = 38.58min, DL-BPNN = 35.70, SVM = 37.84min, hyp-free CPCLS = 33.95min, Ensembles = 34.83min, RF = 34.48min, GBDT = 34.32min and XGBoost = 34.80 are much closer to the actual flight delay mean value, but the Stdev, skew and kur indicate that the distribution is clustered and symmetrical at the centre. The hyp-free CPCLS showed a better linear trend with R=0.19, whereas SL-BPNN showed a worse linear trend with R= 0.01. For quartiles, more specifically med (2nd quartile), the SL-BPNN, DL-BPNN, SVM, hyp-free CPCLS, Ensembles, RF, GBDT and XGBoost estimated mean flight delays of 34.18 min, 33.70 min, 35.77 min, 30.46 min, 33.68 min, 26.19min, 27.73min and 25.57min respectively do not truly represent the actual flight delay mean value of 11.00 min. The maximum departure flight delay approaches 1998.00 min in actual delay, whereas, for estimation methods, it approaches a maximum of 892.56 min. Similarly, the Stdev, skew, and kur of absolute error comparisons indicate that for all the estimation methods the error distributions are highly dispersed and skewed to the right side with leptokurtic peak. Preferably, in the regression estimation mechanism, the estimation method should be able to truly approximate actual flight delay in all quartiles. In our study, the MAE of 47.16min, 38.22 min, 39.31min, 36.37min, 37.26min, 36.60min, 36.42min and 36.57min with Stdev 78.20min, 73.77min, 74.68min, 74.40min, 74.31min, 75.75min, 75.38min and 77.00min for SL-BPNN, DL-BPNN, SVM, hyp-free CPCLS, Ensemble, RF, GBDT and XGBoost from the actual delay implies the regression mechanism maybe not appropriate for predicting flight delays.

To investigate the reasons, the distribution and normality of the actual flight delays were studied. The one sample KS normality test and Quantile-Quantile plot (Q-Q plot) were performed to check the normality of the actual flight departure delay dataset. The one sample KS normality test rejects the null hypothesis of the normal population distribution because of the p-value < 0.05 (level of significance). The Q-Q plot showed that flight delay data points are not along the diagonal of the line which does not fulfil the assumption of normality. To overcome the above normality assumption limitations, various pre-processing and transformation techniques were tested to convert the non-normal distribution into a normal distribution. In pre-processing, the extreme values, long tails, outliers, and noisy data were removed to improve the performance of the estimation methods. In transformation, various techniques were employed to improve the distribution by taking the square root or logarithmic. The pre-processing and transformation techniques were tested individually and collectively. However, no significant improvement in the distribution of dataset and minimization of the objective function was found.

The descriptive and graphical statistical analyses of the actual flight delays, estimation methods for predicted flight delays, and their absolute errors imply that the regression mechanism may not be a suitable approach when the historical flight dataset is highly dispersed and positively skewed. Efforts to improve the data distribution and estimation methods performance by various pre-processing and transformation techniques do not show significant support in minimizing the objective function.

6.2.2 Delay prediction as a classification problem

To compare the estimation mechanisms, the same historical highly dispersed and positively skewed dataset used for regression is applied to evaluate the performance of the classification task. The regression flight delays continuous variable is converted to a binary variable by labelling. The zero-minute values are labelled as on-time and value greater than zero minutes are labelled as delays. The objective is to improve the prediction accuracy of the classifiers by correct classifying labels. Other than simple accuracy measurement, which may lead to the wrong conclusion, confusion matrix and classification report performance indicators were also used to evaluate the performance of the classifiers.

Table 4
Estimation techniques classification prediction result with the original dataset

Dataset type	Estimation Technique	Model Accuracy (%)		Label	Classification Report (%)			Confusion Matrix (Nos.)	
		Train	Test		Precision	Recall	F1	On-time	delay
Original Dataset	SL-BPNN	72.21	72.16	On-time	0	0	0	0	2659
				delay	72	100	84	0	6894
	DL-BPNN	73.51	72.35	On-time	51	14	22	374	2285
				delay	74	95	83	356	6538
	SVM	74.39	72.35	On-time	52	10	17	263	2396
				delay	74	96	83	245	6649
	hyp-free CPCLS	73.36	72.62	On-time	53	17	25	439	2220
				delay	75	94	83	396	6498
	Ensembles	73.44	72.29	On-time	52	6	10	147	2512
				delay	73	98	84	135	6759
	RF	77.95	72.65	On-time	58	6	11	163	2496
				delay	73	98	84	116	6778
	GBDT	74.96	73.12	On-time	59	12	19	309	2350
				delay	74	97	84	218	6676
	XGBoost	74.67	72.94	On-time	57	12	19	306	2353
				delay	74	97	84	232	6662

6.2.2.1 Delay prediction results with the original dataset

Table 4 summarizes the results generated by the estimation methods. The table shows model accuracy, label classification report, and confusion matrix. The estimation methods were able to achieve an average model accuracy of 72.56% with a standard deviation of 0.34%. The average recall of 96.88% for the delay class and 9.63% for the on-time class in the classification report shows that the estimation methods predict the delay class at higher accuracy as compared to the on-time class. Predicting the delay class at high frequency compared to the on-time class may make the model unsuitable for actual applications because the model will suggest most often that the flight will experience departure delay. This problem can be easily understood from the confusion matrix. In the actual original dataset, among 9553 flights from the testing dataset, 2659 flights belong to the on-time class and 6894 flights belong to the delay class. The confusion matrix shows that the estimation methods predict a delay class more often compared to an on-time class.

Most often, it is desired to get a balanced, highest recall and precision percentage. The objective of this study is to get a balanced recall for both classes, with better precision. The results with the original dataset show that model prediction is unbalanced. The in-depth study shows that the main reason for this is due to unbalanced datasets. Among the 19105 flights, 72% belong to the delay class and 28% belong to the on-time class. The majority of datasets belonging to the delay class causes the machine learning estimation methods to learn concepts related to the delay class and ignore the on-time class. The class imbalance may make it very challenging for the estimation methods to accurately classify class labels. To overcome this challenge, various sampling techniques are recommended to balance the classes and remove noisy, overlapping data on the decision boundaries.

6.2.2.2 Sampling techniques for class imbalance and decision boundaries overlapping

The issue of class imbalance and overlapping may significantly affect the performance of machine learning classifiers. Class imbalance can be defined as the training set belonging heavily to one class compared to another class. Class Overlapping can be defined as one class in a training set occupying a large space of another class. For the sake of simplicity, the class in training set occupying a large space is named a majority class and another a minority class. Class imbalance and overlapping may cause the classifier to learn concepts related to the majority class and dominate the minority class. This may create the problem of low classifier accuracy by wrongly predicting a minority class. Class distribution and overlapping can be improved by collecting more datasets that approximately represent both classes equally. In a real scenario, collecting data is costly and involves stakeholder interests. In such a scenario, collecting additional data may not be feasible and there is a need to apply alternative machine learning approaches. A possible alternative strategy can include sampling the training set to improve class distribution and avoid overlapping. Sampling techniques include under-sampling, over-sampling and hybrid (combination) approaches. The major objective of sampling techniques is to improve the class distribution decision boundary by removing noisy, overlapping, and borderline samples. The most popular and widely used techniques are:

Under-Sampling Techniques (US)

a. Random Under-Sampling (RUS): RUS balances the datasets by randomly eliminating examples from the majority class to bring them equal to the minority class. This technique only applies to the majority class. In RUS, the chances

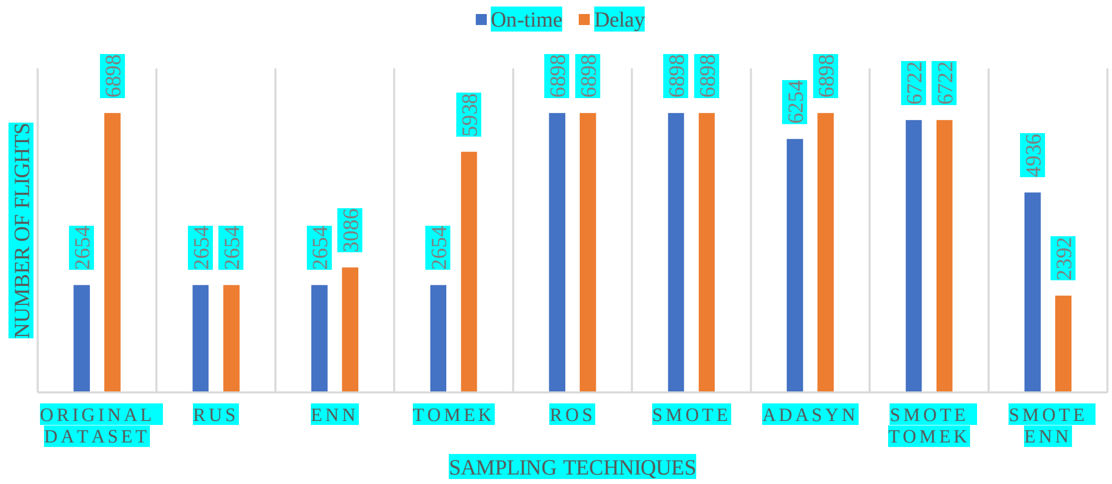


Fig. 3. Improving class distribution and decision boundaries by applying various sampling techniques to the training set

of losing potentially useful data increase because of eliminating examples from the majority class (Batista et al., 2004).

b. Edited Nearest Neighbours (ENN): ENN edit examples by removing them from the majority that does not agree with their neighbours. The examples are removed based on the predefined nearest neighbour K (typically $K = 3$ in most cases). An example will be removed from the dataset if a number of neighbours from another class are predominant (Wilson, 1972). This helps to make the decision boundary smoother by editing noisy and close border examples (Wilson and Martinez, 2000).

c. Tomek Links (Tomek): Tomek can be considered as an under sampling method and a data cleaning method, eliminates noisy examples from the majority class and borderline examples from both classes. Let x^{maj} be an example of the majority class, x^{min} be an example of the minority class and $d(x^{maj}, x^{min})$ is the distance between both examples. The pair (x^{maj}, x^{min}) is called Tomek if there is no case x^{m}, such that $d(x^{maj}, x^{m}) < d(x^{maj}, x^{min})$ or $d(x^{min}, x^{m}) < d(x^{maj}, x^{min})$. If Tomek is formed between both examples, then either one of these examples is noisy or both are on borderline. Besides helping in eliminating noisy examples, this helps to clean overlap between classes and makes the decision boundary smoother (Tomek, 1976).

Over-Sampling Techniques (OS)

a. Random Over-Sampling (ROS): ROS balances the dataset by randomly adding examples to the minority class. The dataset is balanced by randomly replicating/duplicating examples from the minority class to bring them equal to the majority class. This technique is only applied to the minority class. Contrary to the RUS, in ROS, the chances of overfitting increases because of using duplicate examples of the minority class (Batista et al., 2004).

b. Synthetic Minority Over Sampling Technique (SMOTE): SMOTE oversamples the minority class by creating "synthetic" examples rather than replicating or duplicating to avoid overfitting. SMOTE works by taking the difference between the minority sample x^{min} and a randomly selected K-nearest neighbour x^{KNN} (i.e.: $diff = x^{min} - x^{KNN}$). Multiplying the difference with a random number between 0 to 1 and adding to the minority sample under consideration (i.e.: $New\ sample = x^{min} + rand(0,1) * diff$), generates a new random synthetic sample between the minority sample and its K-nearest neighbour (Chawla et al., 2002).

c. Adaptive Synthetic (ADASYN): This concept of ADASYN is similar to SMOTE. ADASYN also oversamples the minority class by creating "synthetic" samples with some improvement compared to SMOTE. In SMOTE, an equal number of synthetic samples is generated for each minority sample, whereas, ADASYN gives weight to minority samples that are closer to the majority class and harder to learn. The more minority samples closer to the majority class and harder to learn, the more it will generate synthetic samples for those samples (He et al., 2008).

Combination of under and over-sampling Techniques (UOS)

a. SMOTETomek: Batista et al. (2003) explained that oversampling techniques can balance the class distribution but cannot solve the problem of skewed class distributions. The class cluster (decision boundary) may not be well defined since some majority class examples may be occupying minority class space. Another issue can be the generating of artificial synthetic samples that may introduce minority examples too deeply in the majority class space. Batista et al. (2003) proposed applying Tomek to the oversampled SMOTE examples as a data cleaning

Table 5
Estimation methods classification prediction result for sampled datasets

Sampled Dataset	Estimation Method	Model Accuracy (%)		Label	Classification Report (%)			Confusion Matrix (Nos.)	
		Train	Test		Precision	Recall	F1	On-time	delay
RUS	SL-BPNN	63.33	62.06	On-time	39	63	48	1664	995
				delay	81	62	70	2629	4265
	DL-BPNN	65.27	61.15	On-time	38	66	49	1751	908
				delay	82	59	69	2803	4091
	SVM	63.77	57.33	On-time	37	76	50	2025	634
				delay	84	50	63	3442	3452
	hyp-free CPCLS	67.28	64.15	On-time	41	68	51	1795	864
				delay	83	63	72	2561	4333
	Ensembles	65.09	60.67	On-time	38	68	49	1812	847
				delay	82	58	68	2910	3984
	RF	76.52	62.80	On-time	41	75	53	1989	670
				delay	86	58	69	2883	4011
	GBDT	68.97	62.89	On-time	41	72	52	1922	737
				delay	85	59	70	2808	4086
	XGBoost	80.32	60.05	On-time	39	77	52	2052	607
				delay	86	53	66	3209	3685
ENN	SL-BPNN	73.67	60.58	On-time	38	68	49	1797	862
				delay	82	58	68	2903	3991
	DL-BPNN	77.14	60.64	On-time	38	68	49	1807	852
				delay	82	58	68	2908	3986
	SVM	75.73	60.28	On-time	39	72	50	1914	745
				delay	84	56	67	3049	3845
	hyp-free CPCLS	77.18	61.62	On-time	39	70	50	1867	792
				delay	84	58	69	2874	4020
	Ensembles	75.64	60.79	On-time	39	70	50	1849	810
				delay	83	57	68	2935	3959
	RF	75.48	63.12	On-time	40	68	50	1796	863
				delay	83	61	71	2660	4234
	GBDT	80.76	59.92	On-time	39	77	52	2058	601
				delay	86	53	66	3228	3666
	XGBoost	68.11	64.17	On-time	41	68	51	1810	849
				delay	84	63	72	2573	4321
Tomek	SL-BPNN	70.13	72.17	On-time	50	10	16	257	2402
				delay	73	96	83	256	6638
	DL-BPNN	78.27	69.40	On-time	45	42	44	1126	1533
				delay	78	80	79	1390	5504
	SVM	75.23	71.98	On-time	49	30	37	795	1864
				delay	77	88	82	812	6082
	hyp-free CPCLS	71.71	72.14	On-time	50	31	38	813	1846
				delay	77	88	82	815	6079
	Ensembles	73.28	72.78	On-time	52	23	32	620	2039
				delay	76	92	83	561	6333
	RF	81.12	72.87	On-time	52	29	37	771	1888
				delay	77	90	83	703	6191
	GBDT	73.63	72.97	On-time	53	22	32	596	2063
				delay	76	92	83	519	6375
	XGBoost	73.20	72.34	On-time	51	27	36	730	1929
				delay	76	90	82	713	6181
ROS	SL-BPNN	63.90	61.00	On-time	38	65	48	1734	925
				delay	82	59	69	2800	4094
	DL-BPNN	66.15	61.88	On-time	39	65	49	1741	918
				delay	82	61	70	2723	4171
	SVM	63.40	57.33	On-time	37	76	50	2025	634
				delay	84	50	63	3442	3452
	hyp-free CPCLS	67.27	64.21	On-time	41	67	51	1772	887
				delay	83	63	72	2532	4362
	Ensembles	64.84	60.45	On-time	38	68	49	1821	838
				delay	83	57	68	2940	3954
	RF	66.92	63.10	On-time	41	70	51	1863	796
				delay	84	60	70	2729	4165
	GBDT	68.64	63.64	On-time	41	72	53	1921	738
				delay	85	60	71	2735	4159
	XGBoost	68.15	63.30	On-time	41	72	52	1918	741
				delay	85	60	70	2764	4130

method. The basic idea is to oversample the minority class by SMOTE and then remove the noisy majority class and

Table 5
Continued

Sampled Dataset	Estimation Method	Model Accuracy (%)		Label	Classification Report (%)			Confusion Matrix (Nos.)	
		Train	Test		Precision	Recall	F1	On-time	delay
SMOTE	SL-BPNN	63.64	62.12	On-time	39	63	48	1676	983
				delay	81	62	70	2635	4259
	DL-BPNN	70.47	62.67	On-time	39	64	49	1695	964
				delay	82	62	71	2602	4292
	SVM	63.06	63.35	On-time	39	58	47	1544	1115
				delay	80	65	72	2386	4508
	hyp-free CPCLS	67.53	65.07	On-time	42	66	51	1757	902
				delay	83	65	73	2435	4459
	Ensembles	64.07	62.81	On-time	39	62	48	1651	1008
				delay	81	63	71	2544	4350
	RF	68.14	65.35	On-time	42	64	51	1696	963
				delay	83	66	73	2347	4547
	GBDT	70.48	64.48	On-time	42	68	51	1798	861
				delay	84	63	72	2532	4362
	XGBoost	70.19	64.03	On-time	41	68	51	1815	844
				delay	84	62	71	2592	4302
ADASYN	SL-BPNN	61.42	62.78	On-time	39	60	47	1596	1063
				delay	81	64	71	2492	4402
	DL-BPNN	63.42	63.20	On-time	39	60	48	1596	1063
				delay	81	64	72	2452	4442
	SVM	61.07	63.35	On-time	39	58	47	1544	1115
				delay	80	65	72	2386	4508
	hyp-free CPCLS	65.83	65.85	On-time	42	63	51	1683	976
				delay	83	67	74	2286	4608
	Ensembles	61.70	63.33	On-time	39	59	47	1581	1078
				delay	81	65	72	2425	4469
	RF	66.84	66.62	On-time	43	60	50	1585	1074
				delay	82	69	75	2114	4780
	GBDT	69.70	66.63	On-time	43	59	50	1572	1087
				delay	82	70	75	2101	4793
	XGBoost	68.11	64.17	On-time	41	68	51	1810	849
				delay	84	63	72	2573	4321
SMOTETomek	SL-BPNN	64.20	62.17	On-time	39	63	48	1670	989
				delay	81	62	70	2624	4270
	DL-BPNN	72.25	62.67	On-time	39	61	48	1626	1033
				delay	81	63	71	2533	4361
	SVM	63.40	63.35	On-time	39	58	47	1544	1115
				delay	80	65	72	2386	4508
	hyp-free CPCLS	***67.90***	***65.23***	On-time	***42***	***66***	***52***	***1766***	***893***
				delay	***83***	***65***	***73***	***2429***	***4465***
	Ensembles	64.72	62.75	On-time	39	62	48	1647	1012
				delay	81	63	71	2546	4348
	RF	66.12	64.26	On-time	40	60	48	1586	1073
				delay	81	66	73	2341	4553
	GBDT	71.19	64.29	On-time	41	67	51	1777	882
				delay	83	63	72	2529	4365
	XGBoost	68.11	64.17	On-time	41	68	51	1810	849
				delay	84	63	72	2573	4321
SMOTEENN	SL-BPNN	80.36	53.41	On-time	36	83	50	2199	460
				delay	86	42	57	3990	2904
	DL-BPNN	82.46	54.72	On-time	36	79	49	2111	548
				delay	85	45	59	3777	3117
	SVM	78.34	49.60	On-time	34	86	49	2274	385
				delay	86	36	51	4429	2465
	hyp-free CPCLS	89.62	60.26	On-time	39	72	50	1906	753
				delay	84	56	67	3043	3851
	Ensembles	80.89	53.39	On-time	36	83	50	2205	454
				delay	86	42	57	3998	2896
	RF	81.79	54.40	On-time	36	83	50	2211	448
				delay	87	43	58	3908	2986
	GBDT	84.95	52.95	On-time	36	87	51	2319	340
				delay	89	40	55	4155	2739
	XGBoost	83.88	58.05	On-time	38	80	51	2120	539
				delay	86	50	63	3468	3426

borderline majority and minority classes by Tomek to make the decision boundary smoother.

b. SMOTEENN: The working method of SMOTEENN is similar to SMOTETomek. The basic difference between Tomek and ENN is in the sampling mechanism. Tomek removes noisy majority class examples and borderline examples from both cases, whereas, ENN removes examples misclassified by the nearest neighbours. It is considered that ENN tends to remove more examples from both classes by doing more in-depth cleaning compared to Tomek (Batista et al., 2004).

6.2.2.3 Delay prediction results with a sampled dataset

To improve the prediction accuracy of the estimation methods, the sampling techniques mentioned in section 6.2.2.2 are applied to the original training set. Fig. 3 illustrates the original training dataset and sampled training dataset generated by each sampling technique. The sampled data were used to train the estimation methods and the original testing set was used to validate the performance. For actual applications, exact knowledge about which sampling technique will perform better is unknown. Detailed experimental work was carried to select the best suitable combination of sampling technique and estimation method so as to achieve better prediction performance.

Table 5 summarizes the results generated by the estimation methods using various sampling techniques. In-depth analysis of the table demonstrates that the hyp-free CPCLS classifier with the SMOTETomek sampling technique was able to achieve better prediction performance compared to the others. The recall accuracy for the delay was 65% and for on-time 66%. The precision accuracy for the delay was 83% and for on-time 42%. However, the recall and precision accuracies of hyp-free CPCLS_SMOTE were also the same as the hyp-free CPCLS_SMOTETomek, suggesting that the SMOTE sampling technique may be also helpful in improving prediction accuracy. In such cases, other performance metrics were studied to select the best sampling technique. Referring to the f1 score and confusion matrix, the hyp-free CPCLS_SMOTETomek overall performance is better than hyp-free CPCLS_SMOTE. The model average f1 score demonstrated that hyp-free CPCLS_SMOTETomek was able to achieve accuracies of 62.5%, slightly better than hyp-free CPCLS_SMOTE at 62%. Similarly, the confusion matrix shows that hyp-free CPCLS_SMOTETomek predicted 1766 flights for TP and 4465 flights for TN comparatively better than hyp-free CPCLS_SMOTE for 1757 flights for TP and 4459 flights for TN. An interesting fact, for both sampling techniques, was that the classifier is the same (i.e. hyp-free CPCLS). Moreover, the hyp-free CPCLS was able to achieve learning much faster compared to SL-BPNN, DL-BPNN, SVM, RF, GBDT and XGBoost. The hyp-free CPCLS was able to get balanced recall accuracy in 0.73s compared to SL-BPNN of 7.62s, DL-BPNN of 17.14s, SVM of 82.20s, RF of 0.79s, GBDT of 10.31s and XGBoost of 1.03s.

Furthermore, it can be seen that the prediction capability of the estimation methods is considerably lower with Tomek compared to other sampling techniques. The Tomek sampling technique generated an average recall of 89.5% for the delay class and 26.75% for the on-time class. The recall accuracies were lower than other sampling techniques but considerably better than the results predicted by the original dataset (Table 4). The selection of any sampling technique based on prior judgment may not be quite optimal. Different combinations of estimation methods and sampling techniques results help to select the hyp-free CPCLS_SMOTETomek technique having better prediction capability. In comparison with each sampling technique, the performance of the hyp-free CPCLS classifier is found to be superior to other estimation methods such as SL-BPNN, DL-BPNN, SVM, Ensembles, RF, GBDT and XGBoost. The experimental work gives insight that generating linearly independent and non-redundant hidden units in each hidden layer with no iterative tuning of connection weights helps to improve the predictive performance of the network. Unlike BPNN (SL and DL), hyp-free CPCLS make sure that each hidden unit generated in the hidden layer is orthogonal in relationship to other hidden units in the same layer. Similarly, analytically calculating connection weights and self-organizing topology reduces the complexity of the network which facilitates improving the learning process.

The comparison of hyp-free CPCLS_SMOTETomek was also performed with Cost-Sensitive Weighted RF (CSWRF) and Nested SVM (NSVM) to demonstrate the effectiveness. Table 6 summarizes the results generated by the CSWRF and NSVM. CSWRF achieved recall accuracy of 64% for the delay class and 61% for the on-time class. For NSVM, the number of folds was set to $k = 5$ in the outer loop and $k = 3$ in the inner loop. The best combination of C, γ hyperparameters (mentioned in Section 6) was searched using grid search. Two types of experimental work were performed. Without applying sampling techniques, NSVM achieved recall accuracy of 95.2% with Stdev 1.30% for the delay class and 14.8% with Stdev 3.49% for

Table 6
CSWRF and NSVM prediction results

Estimation Technique	Model Accuracy (%)		Label	Classification Report (%)		
	Train	Test		Precision	Recall	F1
CSWRM	63.72	63.29	On-time delay	40 81	61 64	48 72
NSVM	72.82 (0.23)	72.88 (0.81)	On-time delay	54.8 (3.03) 74.4 (1.14)	14.8 (3.49) 95.2 (1.30)	22.8 (4.44) 83.6 (0.55)
NSVM with SMOTETomek	79.98 (0.26)	67.3 (1.02)	On-time delay	41.4 (1.14) 77.2 (1.10)	41.6 (2.19) 77.2 (1.10)	41.2 (1.64) 77.6 (0.89)

the on-time class. By applying the SMOTETomek sampling technique, NSVM achieved recall accuracy of 77.2% with Stdev 1.10% for the delay class and 41.6% with Stdev 2.19% for the on-time class. The downside of NSVM is that it took 3688s and 11348s to train the model without and with the sampling technique. The comparison of results in Table 6 with results in Table 5 demonstrates that hyp-free CPCLS_SMOTETomek has better and balanced recall accuracy of 65% for the delay class and 66% for the on-time class compared to both CSWRF and NSVM.

In Table-4, the better classifier hyp-free CPCLS was able to achieve recall accuracy of 94% for the delay class and 17% for the on-time class which gives insight that the classifier overtrained the delay class by learning noisy, overlapping, and borderline examples. The comparison of results in Table-4 and Table-5 provides intuition that the application of sampling techniques helps to remove noisy, overlapping, borderline examples, improve class decision boundaries, and class distribution which results in better generalization performance and avoids overfitting compared to the original unbalance and noisy data. More specifically, the application of SMOTETomek facilitated in oversampling the minority class by SMOTE, and then removed the noisy majority class and borderline majority and minority classes by Tomek to make the decision boundary smoother. The results in Table-5 shows that hyp-free CPCLS-SMOTETomek predicted delay and on-time classes at better and balance recall accuracy of 65% and 66% compared to an imbalanced dataset (Table-4) of 94% and 17%, respectively. This shows that the model with hyp-free CPCLS-SMOTETomek does not suffer from overfitting and has better and balance generalization performance.

Unlike existing works, the sampling techniques in the current study are applied to the training set, and performance is evaluated on the original testing set. Training on synthetic sampled examples and evaluating an original testing set may help to study the application of estimation methods on actual application examples. Applying sampling techniques to both training and testing sets may lead to inaccurate findings. The sampling techniques applied to the testing set will remove noisy and overlapping examples by improving the class distribution but may not truly represent the actual flight delay distribution. A new set of experiments was carried out by applying sampling techniques to both the training and testing sets to see any difference in improvement. The work was carried out by considering the hyp-free CPCLS classifier because of its superior performance compared to other estimation methods. Table 7 shows the best results predicted by hyp-free CPCLS with SMOTETomek and SMOTEENN, using two different types of sampling approaches. For the sake of simplicity, approach-one refers to sampling techniques applied to only the training set and approach-two refers to sampling techniques applied to both the training and testing set. Compared to the results of approach-one shown in Table 5, the sampling techniques with approach-two show better results. Table 7 summarizes the results of both approaches. For approach-two, SMOTEENN generated the highest improvement in prediction accuracy. The approach-two recalls of 73% and 86% compared to approach-one recalls of 56% and 72% for the delay and on-time class labels respectively, indicate a significant difference in prediction results. The comparison of SMOTETomek is also important because for our recommended approach (Table 5) its prediction accuracy is superior. Like SMOTEENN, the prediction accuracy of SMOTETomek increased by using approach-two. The approach-two recalls of 79% and 70% compared to approach-one recalls of 65% and 66% for delay and on-time class labels respectively, showing the significant difference in prediction results. The results in Table 7 demonstrate that a wrong application of sampling techniques on both the training and testing sets may lead to an inaccurate conclusion, which may create the problem of poor accuracy in real-world applications.

Experimental work using a different combination of sampling techniques, sampling approaches, and estimation methods suggests that the hyp-free CPCLS classifier with SMOTETomek sampling techniques applied to the training set (approach-one) shows reliable results. The prediction of the departure flights in a future four-hour time horizon as being on-time or delayed is of major importance to airlines, however, as knowing the delay duration can help airlines to make informed decisions in order to avoid or minimize delays. In the following section, we explain a novel hierarchical integrated machine learning model for predicting the flight delay and its duration at a certain threshold in the series.

6.2.2.4 Hierarchical integrated model prediction results

Table 7
Hyp-free CPCLS classifier prediction results by using two types of sampling approaches

Dataset type	Sampling technique applied to only training dataset						Sampling technique applied to both training and testing dataset					
	Model Accuracy (%)		Label	Classification Report (%)			Model Accuracy (%)		Label	Classification Report (%)		
	Train	Test		Precision	Recall	F1	Train	Test		Precision	Recall	F1
SMOTETomek	67.90	65.23	On-time	42	66	52	78.44	74.42	On-time	77	70	73
			delay	83	65	73			delay	72	79	76
SMOTEENN	89.62	60.26	On-time	39	72	50	90.65	82.33	On-time	88	86	87
			delay	84	56	67			delay	71	73	72

Table 8
Hierarchical integrated model prediction results

Estimation Technique	Model Accuracy (%)		Label	Classification Report (%)			Confusion Matrix (Nos.)	
	Train	Test		Precision	Recall	F1	On-time	delay
Departure Delay (Level-1)	67.90	65.23	On-time	42	66	52	1766	893
			delay	83	65	73	2429	4465
Threshold 60 min (Level-2)	68.17	62.95	1-60min	87	64	74	3563	2021
			>60min	28	59	38	534	778
Threshold 30 min (Level-3)	64.99	60.21	1-30min	82	61	70	2583	1644
			31-60min	31	57	41	559	750

Following the study approach of section 6.2.2.3, the departure delay duration at a certain threshold, in series, was studied using the hyp-free CPCLS classifier with SMOTETomek sampling techniques applied to the training set. For a better understanding of the hierarchical integrated model, we propose to predict departure delay and duration at three levels. Level-1 is the prediction model proposed in section 6.2.2.3 (Delay prediction results with sampled dataset). Level-2 and level-3 are models proposed at thresholds of 60min and 30min respectively, to predict the flight departure duration. The three levels are integrated into one hierarchical model. Unlike existing works, the predictions at 60min and 30min thresholds are based on the relevant hierarchical dataset. For level-2, the binary classes are labelled as predicting a departure delay at duration 1-60min (TP) and greater than 60min (TN). For level-3, the binary classes are labelled as predicting a departure delay at duration 1-30min (TP) and 31-60min (TN). For each proceeding threshold, the dataset relevant to that specific threshold was used for model training, and Fig. 2(a) illustrates this concept. For level-2, the dataset greater than zero minutes (excluding the on-time dataset) was used for training and testing the classifier. For level-3, the dataset greater than zero and less than 60 minutes (excluding level-1 on-time and level-2 delay greater than 60min dataset) was used for the training and testing classifier.

Fig. 2(b) and Table 8 summarize the hierarchical integrated model implementation and prediction results respectively. In the table, the results of level-1, discussed in section 6.2.2.3, are imported from Table 5. The hyp-free CPCLS classifier, with the SMOTETomek sampling technique applied to the training set, was used to classify binary classes for level-2 and level-3. The model predicted testing accuracy for level-2 and level-3 are approximately near to level-1. The levels testing accuracies indicates that the classifier prediction capabilities increase by selecting a higher threshold and vice versa. In level-2 and level-3, recalls of 64% for the 1-60min class label and 59% for >60min class label, and the recall of 61% for the 1-30min class label and 57% for 31-60min class label respectively indicate the balanced prediction. In terms of model learning time, the hyp-free CPCLS classifier required 0.73s for level-1, 0.62s for level-2, and 0.51s for level-3. The recall accuracy and learning time are useful indicators to understand the scalability of the hyp-free CPCLS classifier. Scalability is defined as the effect of an increase in training size on the computational performance of the classifier. It is important to have the same effect of training size on both accuracy and learning time. Comparative study show that level-1 (19,105 flights) achieved average accuracy of 65.5% in 0.73s, level-2 (13,792 flights) achieved average accuracy of 61.5% in 0.62s, and level-3 (11,071 flights) achieved average accuracy of 59% in 0.51s using same hyperparameters of hyp-free CPCLS. For all three levels, results are consistent. For instance, comparing results from small to large data set, accuracy is improving, and training time is increasing.

6.2.2.5 Factors influencing flight delay

Impact factor analysis was performed to identify attributes (or factors) that have a great influence on the prediction performance. The mutual information (bits) evaluation method was adopted to determine the most influencing factor that highly contributes to the flight delay. Fig. 4 illustrates the impact factor analysis of prediction. The attributes are ranked according to their increase in mutual information. The top five influencing factors that contribute to flight delays are distance travelled, aircraft registration, ramp weight, cruise initial altitude, and aircraft type. Figs. 5 and 6 show the analysis of influencing factors highly contributing to flight delays. The dataset consists of eight sectors covering short-range flights and long-range flights on long-haul routes. The two sectors having a distance of 3373Nautical Mile (NM) and 3454NM are abbreviated as SD3373 and SD3454. Both sectors belong to short-range flights with an average flight time of 433min duration. Where SD3373, for example, means sector distance (SD) of 3373NM. The other six sectors having a distance of 4693NM, 4762NM, 5537NM, 5632NM, 6584NM, and 6665NM are abbreviated as SD4693, SD4762, SD5537, SD5632, SD6584, and SD6665. These six sectors belong to long-range flights with an average flight time of 696min duration.

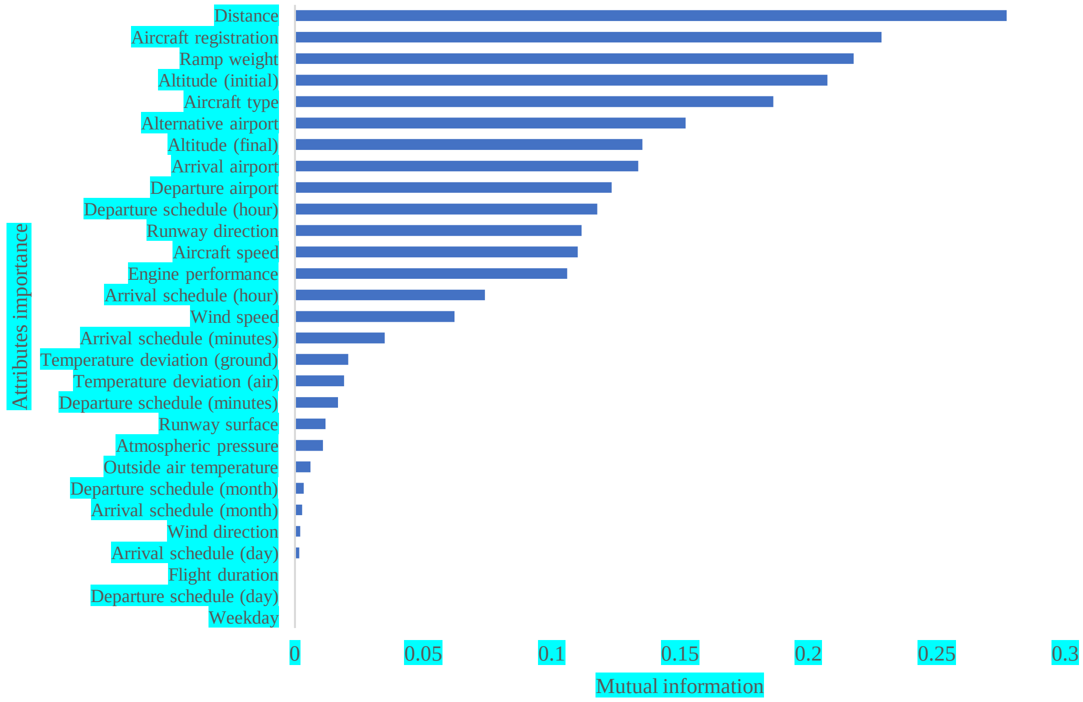


Fig. 4. Impact factor analysis

Fig. 5 summarizes the influence of distance, ramp weight, and cruise altitude on the flight delay. The horizontal axis, primary vertical axis, and secondary vertical axis refer to the sector, weight in kilogram (Kg) and altitude in feet (ft), and delay time (min), respectively. The analysis illustrates that the short-range sectors experience less average delay time compared to long-range sectors. The short-range sectors 3373NM and 3454NM reported average delay time of 26min and 16min, respectively. The long-range sectors 4693NM, 4762NM, 5537NM, 5632NM, 6584NM, and 6665NM reported average delay time of 51min, 49min, 33min, 21min, 27min, and 34min, respectively. The analysis in the figure shows that short-range sectors are operated with less ramp weight and high altitude compared to long-range sectors. This provides intuition that operating flights at high altitudes experience less delay compared to operating flights at low altitudes. For instance, high altitudes may be less crowded compared to low altitudes. Similarly, ramp weight leads to the intuition that flights with high weight may experience high delays because of the complex ground and enroute operations compared to flights with less weight.

Fig. 6 summarizes the influence of aircraft registration and aircraft type on the flight delay. The horizontal axis shows aircraft and sector, and the vertical axis is about delay time (min). The aircraft registration and aircraft type subcategories are merged

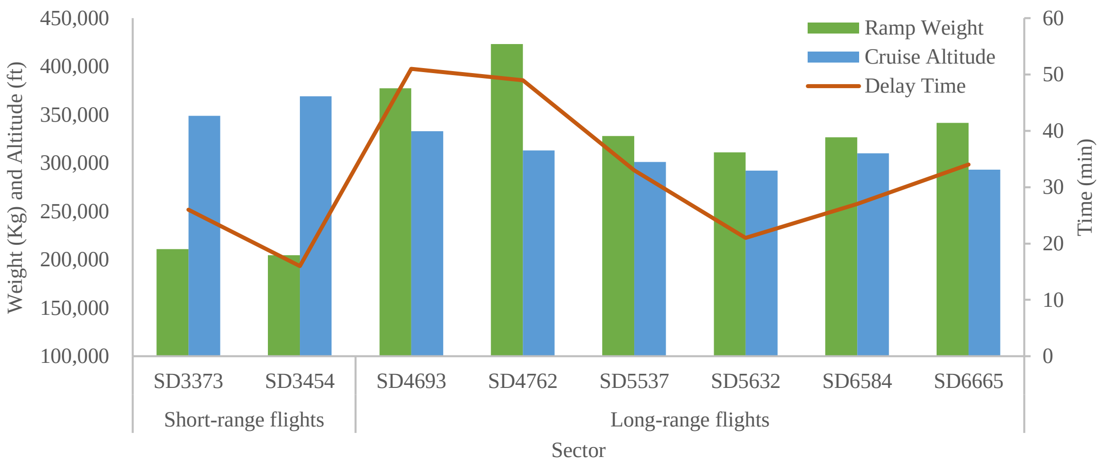


Fig. 5. Distance, weight, and altitude influencing flight delays

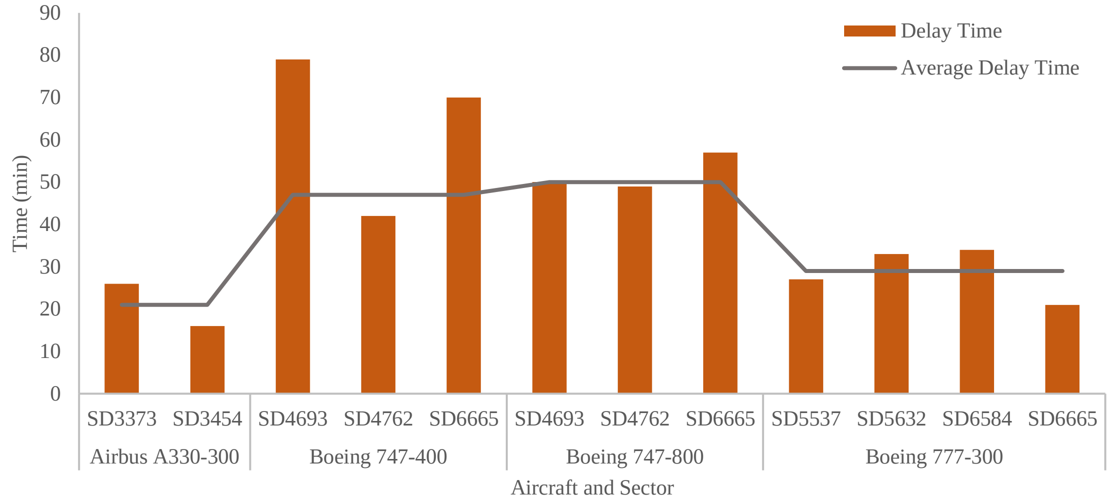


Fig. 6. Aircraft registration and type influencing flight delay

into one common category named aircraft size. The dataset consists of widebody aircraft of Airbus A330-300 and Boeing 747-400/747-800/777-300. Airbus A330-300 was mainly operated on short-range sectors, whereas the Boeing 747-400/747-800/777-300 were mainly operated on several long-range sectors. The analysis in the figure shows that the low passenger carrier Airbus A330-300 experiences an average delay of 21min compared to Boeing 747-400/747-800/777-300 which experiences an average delay of 35.83min. The three sizes of Boeing aircraft, Boeing 747-400, Boeing 747-800, and Boeing 777-300 were operated alternatively on long-range six sectors. The Boeing 747-400 operated in SD4693, SD4762, and SD6665 experience an average delay of 47min, Boeing 747-800 operated in SD4693, SD4762, and SD6665 experience an average delay of 50min, and Boeing 777-300 operated in SD5537, SD5632, SD6584, and SD6665 experience an average delay of 29min. Further analysis was performed to study the relation of aircraft engine performance on flight delays. The existing airline measures aircraft engine performance in percentage compared to new aircraft in the fleet. Airbus A330-300 and Boeing 747-400/747-800/777-300 having an average engine performance of 4.92%, and -9.95 experience an average delay of 21min, and 36 min, respectively. This gives intuition that aircraft having efficient engines may experience comparatively less delay because of less non-schedule maintenance and fewer breakdowns of parts compared to aircraft having inefficient engines.

Other than five influencing factors, results show that alternative airport, altitude (final), arrival airport, departure airport, departure schedule (hour), runway direction, aircraft speed, arrival schedule (hour), and wind speed are also the factors that highly contribute to flight delay prediction. However, because of the data confidentiality agreement, international airline demands not to disclose information about attributes in detail.

6.2.2.6 Comparison of Hierarchical integrated (series) model with parallel model and multiclass classification scheme

In this subsection, two types of comparisons are performed to demonstrate the effectiveness of the hierarchical integrated (series) model. In the first comparison, the series model is compared with the parallel model, whereas, in the second comparison, the series model is compared to the multiclass classification scheme to understand the improvement in threshold prediction.

Predicting flight delay and duration in a hierarchical series of steps can be considered a more novel approach, above a certain threshold, than implementing multiple prediction models in parallel. The experiment *(first comparison)* was conducted to demonstrate the effectiveness of a series prediction model compared to parallel prediction models. Fig. 7 shows the precision-recall curve in order to understand the performance of both models. The precision-recall (PR) curve is an important tool to evaluate the performance of models dealing with imbalanced classification problems having minority class. The objective is to improve the PR curve and select a model having a larger area under the curve (AUC). Step (a) or Level (1) is similar for both series and parallel models. This means that classification of flight departure status as delay or on-time is the same because both models are using the same data and hence no comparison is needed. However, for threshold prediction, the comparison can be performed because both models have a different method of extracting information from the dataset. The figure illustrates the comparison for thresholds of 60min and 30min for both series and parallel models. The figure shows that the AUC for the series model of 32.44% and 35.14% is better compared to the parallel model 26.43% and 21.02% for thresholds of 60min and 30min respectively. The results demonstrate that the series model facilitates improving the PR curve for threshold minority class prediction. This makes the series model a favourable approach for flight delays and duration prediction rather than a parallel model.

Table 9
Multiclass classification prediction results

Model Accuracy (%)		Label	Classification Report (%)			Confusion Matrix (Nos.)			
Train	**Test**		**Precision**	**Recall**	**F1**	**On-time**	**1-30min**	**31-60min**	**>60min**
43.95	41.76	On-time	46	8	13	200	2252	207	0
		1-30min	46	79	58	172	3308	718	0
		31-60min	25	36	29	30	820	481	0
		>60min	0	0	0	30	788	547	0

For comparison of the series model with the multiclass classification scheme *(second comparison)*, the output variable was encoded with multiple labels. For instance, the flights with no delay were labelled as "on-time", flights with delay 1 to 30 minutes were labelled as "1-30min", flights with delay 31 to 60 minutes were labelled as "31-60min" and flights with delay greater than 60 minutes were labelled as ">60min". This results in a total of four labels for multiclass classification prediction. Table 9 summarizes the results for multiclass classification prediction. Comparison of Table 8 and Table 9 shows that the series model (involving binary labels) has better recall accuracy of 66%, 61%, 57%, and 59% compared to the multiclass model recall accuracy of 8%, 79%, 36% and 0% for labels on-time, 1-30min, 31-60min, and >60min, respectively. The classification report and confusion matrix give insight in that the prediction results of the multiclass model are imbalanced. The reason is that labels are imbalanced, and the majority class is learned more compared to the minority class. For that reason, the majority class label (1-30min) shows higher recall but lower precision. The comparison demonstrates that the series model has improved and balanced the recall accuracy compared to multiclass classification. This makes the series model a more favourable approach compared to the multiclass model.

6.2.2.7 Prediction of delay category

Fig. 1 shows that categories such as passenger and baggage handling, aircraft and ramp handling, air traffic flow restriction and government authority, and reactionary and miscellaneous are the main reason for airline departure delay. In terms of percentage, the categories reported in total 91.41% to departure delay making them considerably important for further study. The better results of hyp-free CPCLS_SMOTETomek in predicting flight departure delay status and duration motivates us to check the performance of the method in the prediction of delay category. Table 10 summarizes the result obtained by hyp-free CPCLS_SMOTETomek in predicting delay categories. The results show that the impact of air traffic flow restriction and government authority on delay is higher compared to other categories. Hyp-free CPCLS_SMOTETomek predicted air traffic flow restriction and government authority at the recall of 62% and F1 at a higher percentage of 47% as a most influencing delay reason. For a smooth and safe flight, the airlines have less control over the restrictions imposed by air traffic control and government authority which mainly contribute to long delays.

6.2.2.8 Managerial implications and future work

The proposed model serves three main purposes. First, the better results of the proposed hierarchical integrated model demonstrate that the prediction of flight delay and duration in series helps is improving the accuracy of the model. This makes it a better decision tool for airlines to initially forecast flight delay and then possible duration so as to plan for a contingency strategy. The application of major international airline's historical data further validates the performance of the proposed hierarchical integrated model. Second, the model can work as an alternative in applications where the regression and multiclass classification estimation mechanism cannot generate the best results. The regression mechanism is useful in getting information in continuous form. However, highly noisy, unbalanced, dispersed, and skewed data may make it difficult for regression to generate desired results. Similarly, class imbalance and overlapping might make it difficult for a multiclass classification scheme to accurately classify class labels. The use of binary labels in the hierarchical integrated model along with the usage of data sampling techniques makes it the best alternative approach to regression and multiclass classification. Third, the work considered flight delays data caused by all delay categories rather than considering one or a specific category delay. This facilitates that the hierarchical integrated model can be embedded with airlines existing information system by

Table 10
Delay category prediction results

Delay Category	Model Accuracy (%)		Label	Classification Report (%)		
	Train	**Test**		**Precision**	**Recall**	**F1**
Passenger and Baggage Handling	70.87	61.39	No delay	91	59	71
			delay	29	74	42
Aircraft and Ramp Handling	61.78	59.09	No delay	83	59	69
			delay	30	60	40
Air Traffic Flow Restriction and Government Authority	67.15	64.99	No delay	84	66	74
			delay	37	62	47
Reactionary and Miscellaneous	73.03	72.16	No delay	88	76	81
			delay	37	57	45

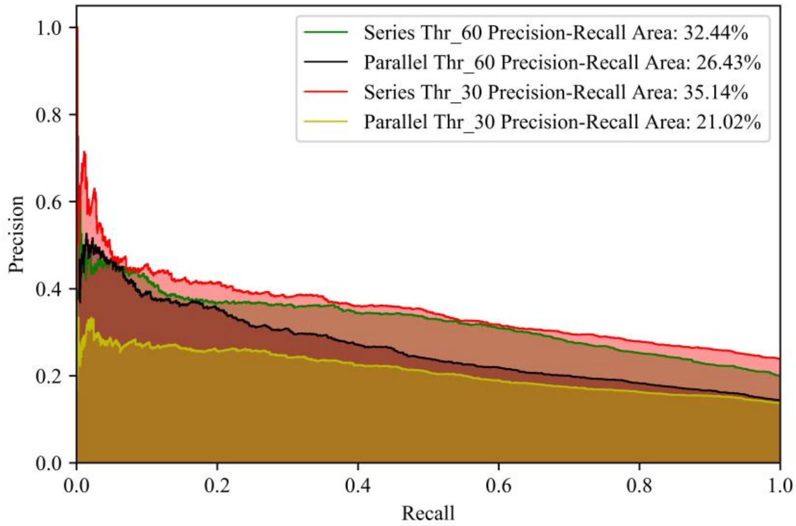


Fig. 7. Precision-recall curve for series model and parallel model

taking into consideration possible flight delays, such as changing the normal parameters to reach arrival airport on-time to improve the overall traveling experience and customer satisfaction.

The comparison of the hierarchical integrated model with the parallel model and multiclass classification scheme demonstrates its effectiveness. However, the average balanced recall accuracies of 65.5%, 61.5%, and 59% for delay status and duration, and 63.25% for delay categories need considerable attention for improvement in the future. The main objective and scope of the current study were to propose a novel hierarchical integrated model and prediction method to predict flight delay status and duration in series to improve the decision-making process. The current study used a dataset having information about the total delay and the category that highly contributes to the total delay on a particular day for each flight instance. Generally speaking, in the real scenario, the reason for the flight delay can be from one subcategory or a combination of many subcategories. Similarly, each subcategory may contribute differently to the flight delay duration. Future work is to obtain information about subcategories along with their respective delay time for each flight instance. The idea is to predict the flight delay status and duration for each category and analyze the importance of the subcategories in each category to enhance the prediction accuracy and further improve the decision-making process. Moreover, in the current work, the historical data was highly noisy, unbalanced, dispersed, and skewed making prediction tasks more challenging. In the future, efforts will be made to obtain more flight delay data and information about attributes such as crew member allocation, air traffic restrictions at arrival and departure airport, immigration mandatory security, and aircraft rotation may facilitate minimizing the problem of class overlapping and improve prediction accuracy.

Machine learning is growing significantly and has gained much attraction in recent years in a wide range of applications due to the advent of big data. Big data enables machine learning to discover more hidden patterns and facilitate improving the predictive power of algorithms. However, big data presents a major challenge of model scalability for machine learning. In this work, we made an effort to address the scalability problem, however, in applications of big data (having gigabytes of data with millions of examples) the stated learning process might be not as scalable and needs further investigation. In the future, challenging work can be to obtain flight delay big data and recommend machine learning algorithms to improve scalability performance with the increasing training dataset. It is well-known that the training dataset plays a significant role in prediction accuracy. Sufficient historical big data and accurate extraction of data contribute to model performance. To ensure that the key attributes are available when needed for training, it is thus required to establish a database for storing historical big data from every aspect of aircraft operation. Thus, the training dataset must be updated regularly so that existing big data and new instance are involved in studying and measuring scalability.

One of the promising but challenging future works is to further explore the influencing factors contributing to flight departure delay. In current work, due to confidentiality constraints, the impact factor analysis cannot be explored in detail. In the future, one of the possible works is to use a hierarchical integrated model to predict flight departure delays using online available or public data sources. The Impact factor analysis can be performed to identify key influencing factors and to compare the results with the currently identified key influencing factors. By this approach, the detailed information about key influencing factors can be explored in the future.

7. Conclusions

This paper proposed a novel hierarchical integrated model for predicting flight departure delay status and duration in series rather than parallel to avoid ambiguity in decision making. The proposed model performance was demonstrated by obtaining historical high dimensional data from the international airline operating in Hong Kong. The highly disperse, right-skewed, noisy, and unbalanced data made it challenging for estimation mechanisms to truly approximate flight departure delays. Our findings show that the proposed model is the best alternative in applications where regression and multiclass classification estimation mechanisms cannot perform. Various sets of experimental work and comparison among SL-BPNN, DL-BPNN, SVM, hyp-free CPCLS, their Ensembles, RF, GBDT and XGBoost estimation methods along with various sampling techniques was performed to investigate the flight delay problem. The statistical analysis of the regression estimation mechanism shows that SL-BPNN, DL-BPNN, SVM, hyp-free CPCLS, Ensembles, RF, GBDT and XGBoost achieved a mean absolute error of 47.16min, 38.22min, 39.31min, 36.37min, 37.26min, 36.60min, 36.42min and 36.57min respectively. Using various pre-processing and transformation techniques does not benefit from improving the regression estimation. Similarly, multiclass classification mechanisms showed an imbalance recall accuracy of 8%, 79%, 36%, and 0% for labels on-time, 1-30min, 31-60min, and >60min, respectively. The results of both regression and multiclass classification show that both estimation mechanisms may not be a suitable approach when the historical flight dataset is highly dispersed, positively skewed with overlapping class decision boundaries.

For the proposed model, the results show that the hyp-free CPCLS machine learning algorithm with the SMOTETomek sampling technique achieved a better-balanced average recall accuracy of 65.5%, 61.5%, 59% for classifying delay status and predicting delay duration hierarchically at thresholds of 60min and 30min, respectively. In a comparison of the proposed model with the parallel model, the result shows that the proposed model was able to predict minority labels more accurately. The area under the precision-recall curve shows that the proposed model achieved a better result of 32.44% and 35.14% compared to the parallel model 26.43% and 21.02% for thresholds of 60min and 30min respectively.

Acknowledgment

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administration Region, China (UGC/FDS14/E04/19).

Appendix A. Pseudocode of CPCLS and hyp-free CPCLS algorithms

CPCLS	**hyp-free CPCLS**
Given a training set $\{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbf{R}^{\boldsymbol{n}}, y_i \in \mathbf{R}, i = 1, \cdots, N\}$ and nonlinear activation function.	Given a training set $\{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbf{R}^{\boldsymbol{n}}, y_i \in \mathbf{R}, i = 1, \cdots, N\}$ and nonlinear activation function.
// Step 1	*// Step 1*
***Initialisation*: Define the number of hidden units in the first hidden layer N^h and proceeding hidden layers $N^{h\prime}$, and expected learning accuracy e	***Initialisation***: Define expected learning accuracy e
// Step 2	*// Step 2*
Learning process: **while** $\|E\| > e$ **do**	***Learning process***: **while** $\|E\| > e$ **do**
	a) let $N^h = 0$
a) calculate the covariance matrix S according to (1)	b) calculate the covariance matrix S according to (1)
b) calculate eigenvalue λ and corresponding eigenvector (or input connection weight) w according to (2) and (3)	c) calculate eigenvalue λ and corresponding eigenvector (or input connection weight) w according to (2) and (3)
	d) Sort λ from largest to smallest value according to (9)
	e) Calculate the percentage variance $V(\%)$ and cumulative percentage variance $CV(\%)$ according to (10) and (11)

	f) select N^h according to (12), such that:
	while $CV(\%)_i < 99.99\%$ **do**
	$N^h = N^h + 1$
	end while

c) calculate user-defined hidden unit h by taking nonlinear activation of the product of x and w with added bias b according to (4)	g) calculate selected hidden unit h by taking nonlinear activation of the product of x and w with added bias b according to (4)
d) calculate the output connection weight β according to (5)	h) calculate the output connection weight β according to (5)

e) predict the target output $\hat{y}$ according to (6) f) calculate E: $E = y - \hat{y}$ g) stack pre-existing hidden units with x according to (7) h) increase the number of hidden units N^h by $N^{h\prime}$ according to (8) **end while**	i) predict the target output $\hat{y}$ according to (6) j) calculate E: $E = y - \hat{y}$ k) stack pre-existing hidden units with x according to (7) **end while**

References

Abdelghany, K.F., S. Shah, S., Raina, S., Abdelghany, A.F., 2004. A model for projecting flight delays during irregular operation conditions. J. Air Transp. Manag. 10, 385–394. https://doi.org/10.1016/j.jairtraman.2004.06.008

Alderighi, M., Gaggero, A.A., 2018. Flight cancellations and airline alliances: Empirical evidence from Europe. Transp. Res. Part E Logist. Transp. Rev. 116, 90–101. https://doi.org/10.1016/j.tre.2018.05.008

Baklacioglu, T., 2016. Modeling the fuel flow-rate of transport aircraft during flight phases using genetic algorithm-optimized neural networks. Aerosp. Sci. Technol. 49, 52–62. https://doi.org/10.1016/j.ast.2015.11.031

Batista, G.E., Bazzan, A.L.C., Monard, M.C., 2003. Balancing Training Data for Automated Annotation of Keywords: a Case Study., in: WOB. pp. 10–18.

Batista, G.E., Prati, R.C., Monard, M.C., 2004. A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor. Newsl. 6, 20–29.

Belcastro, L., Marozzo, F., Talia, D., Trunfio, P., 2016. Using Scalable Data Mining for Predicting Flight Delays. ACM Trans. Intell. Syst. Technol. 8, 1–20. https://doi.org/10.1145/2888402

Breiman, L., 2001. Random Forests. Mach. Learn. 45, 5–32. https://doi.org/10.1023/A:1010933404324

Britto, R., Dresner, M., Voltes, A., 2012. The impact of flight delays on passenger demand and societal welfare. Transp. Res. Part E Logist. Transp. Rev. 48, 460–469. https://doi.org/10.1016/j.tre.2011.10.009

Cao, W., Wang, X., Ming, Z., Gao, J., 2018. A review on neural networks with random weights. Neurocomputing 275, 278–287.

Chawla, N. V, Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. 16, 321–357.

Chen, T., Guestrin, C., 2016. XGBoost: a scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 785–794. https://doi.org/10.1145/2939672.2939785

Chung, S.H., Ma, H.L., Chan, H.K., 2017. Cascading delay risk of airline workforce deployments with crew pairing and schedule optimization. Risk Anal. 37, 1443–1458. https://doi.org/10.1111/risa.12746

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297.

Cranenburgh, S. V., Alwosheel, A., 2019. An artificial neural network based approach to investigate travellers' decision rules. Transp. Res. Part C Emerg. Technol. 98, 152–166. https://doi.org/10.1016/j.trc.2018.11.014

Cui, Q., Li, Y., 2017. Airline efficiency measures under CNG2020 strategy: An application of a Dynamic By-production model. Transp. Res. Part A Policy Pract. 106, 130–143.

Cui, Z., Ke, R., Pu, Z., Ma, X., Wang, Y., 2020. Learning traffic as a graph: A gated graph wavelet recurrent neural network for network-scale traffic prediction. Transp. Res. Part C Emerg. Technol. 115, 102620. https://doi.org/10.1016/j.trc.2020.102620

Dietterich, T.G., 2002. Ensemble learning. Handb. brain theory neural networks 2, 110–125.

Du, W.-B., Zhang, M.-Y., Zhang, Y., Cao, X.-B., Zhang, J., 2018. Delay causality network in air transport systems. Transp. Res. Part E Logist. Transp. Rev. 118, 466–476. https://doi.org/10.1016/j.tre.2018.08.014

Eurocontrol, 2020. All causes delay and cancellations to air transport in Europe [WWW Document]. URL https://www.eurocontrol.int/sites/default/files/2021-02/eurocontrol-coda-digest-q3-2020.pdf (accessed 3.15.21).

Evans, A.D., Lee, P., Sridhar, B., 2018. Predicting the operational acceptance of airborne flight reroute requests using data mining. Transp. Res. Part C Emerg. Technol. 96, 270–289. https://doi.org/10.1016/j.trc.2018.09.024

Fahlman, S.E., Lebiere, C., 1990. The cascade-correlation learning architecture, in: Lippmann, R.P., Moody, J.E., Touretzky, D.S. (Eds.), Advances in Neural Information Processing Systems. Morgan Kaufmann, Denver, pp. 524–532.

Ferrari, S., Stengel, R.F., 2005. Smooth function approximation using neural networks. IEEE Trans. Neural Networks 16, 24–38. https://doi.org/10.1109/TNN.2004.836233

FlightStats, 2019. Airport on-time perfromance reports [WWW Document]. URL https://www.flightstats.com/company/monthly-performance-reports/airports/ (accessed 6.20.19).

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Ann. Stat. 1189–1232.

Gallego, C.E.V., Gómez Comendador, V.F., Amaro Carmona, M.A., Arnaldo Valdés, R.M., Sáez Nieto, F.J., García Martínez, M., 2019. A machine learning approach to air traffic interdependency modelling and its application to trajectory prediction. Transp. Res. Part C Emerg. Technol. 107, 356–386. https://doi.org/10.1016/j.trc.2019.08.015

Hamad, K., Ali Khalil, M., Shanableh, A., 2017. Modeling roadway traffic noise in a hot climate using artificial neural networks. Transp. Res. Part D Transp. Environ. 53, 161–177. https://doi.org/10.1016/j.trd.2017.04.014

He, H., Bai, Y., Garcia, E.A., Li, S., 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). IEEE, pp. 1322–1328.

Hecht-Nielsen, R., 1989. Theory of the backpropagation neural network, in: International Joint Conference on Neural Networks. IEEE, Washington DC, pp. 593–605 vol.1. https://doi.org/10.1109/IJCNN.1989.118638

Hu, Y., Song, Y., Zhao, K., Xu, B., 2016. Integrated recovery of aircraft and passengers after airline operation disruption based on a GRASP algorithm. Transp. Res. Part E Logist. Transp. Rev. 87, 97–112. https://doi.org/10.1016/j.tre.2016.01.002

Huang, G.-B., Zhu, Q.-Y., Siew, C.-K., 2006. Extreme learning machine: theory and applications. Neurocomputing 70, 489–501.

Huang, G., Song, S., Wu, C., 2012. Orthogonal least squares algorithm for training cascade neural networks. IEEE Trans. Circuits Syst. I Regul. Pap. 59, 2629–2637.

IATA, 2019. Industry facts and statistics [WWW Document]. Online. URL https://www.iata.org/pressroom/facts_figures/fact_sheets/Pages/index.aspx (accessed 8.5.19).

Khan, W.A., Chung, S.-H., Ma, H.-L., Liu, S.Q., Chan, C.Y., 2019a. A novel self-organizing constructive neural network for estimating aircraft trip fuel consumption. Transp. Res. Part E Logist. Transp. Rev. 132, 72–96. https://doi.org/10.1016/j.tre.2019.10.005

Khan, W.A., Chung, S.H., Awan, M.U., Wen, X., 2019b. Machine learning facilitated business intelligence (Part I): Neural networks learning algorithms and applications. Ind. Manag. Data Syst. 120, 164–195. https://doi.org/10.1108/IMDS-07-2019-0361

Khan, W.A., Chung, S.H., Awan, M.U., Wen, X., 2019c. Machine learning facilitated business intelligence (Part II): Neural networks optimization techniques and applications. Ind. Manag. Data Syst. 120, 128–163. https://doi.org/10.1108/IMDS-06-2019-0351

Khan, W.A., Ma, H.-L., Ouyang, X., Mo, D.Y., 2021. Prediction of aircraft trajectory and the associated fuel consumption using covariance bidirectional extreme learning machines. Transp. Res. Part E Logist. Transp. Rev. 145, 102189. https://doi.org/10.1016/j.tre.2020.102189

Khanmohammadi, S., Tutun, S., Kucuk, Y., 2016. A new multilevel input layer artificial neural network for predicting flight delays at JFK airport. Procedia Comput. Sci. 95, 237–244. https://doi.org/10.1016/j.procs.2016.09.321

Kim, Y.J., Choi, S., Briceno, S., Mavris, D., 2016. A deep learning approach to flight delay prediction, in: 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC). IEEE, pp. 1–6.

Krogh, A., Hertz, J.A., 1992. A simple weight decay can improve generalization, in: Hanson, S.J., Cowan, J.D., Giles, C.L. (Eds.), Advances in Neural Information Processing Systems. Morgan Kaufmann, Denver, pp. 950–957. https://doi.org/https://dl.acm.org/citation.cfm?id=2987033

Kumar, A., Rao, V.R., Soni, H., 1995. An empirical comparison of neural network and logistic regression models. Mark. Lett. 6, 251–263.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539

Liew, S.S., Khalil-Hani, M., Bakhteri, R., 2016. An optimized second order stochastic learning algorithm for neural network training. Neurocomputing 186, 74–89.

Lin, Z., Vlachos, I., 2018. An advanced analytical framework for improving customer satisfaction: A case of air passengers. Transp. Res. Part E Logist. Transp. Rev. 114, 185–195. https://doi.org/10.1016/j.tre.2018.04.003

Nayyeri, M., Yazdi, H.S., Maskooki, A., Rouhani, M., 2018. Universal approximation by using the correntropy objective function. IEEE Trans. neural networks Learn. Syst. 29, 4515–4521.

Qiao, J., Li, F., Han, H., Li, W., 2016. Constructive algorithm for fully connected cascade feedforward neural networks. Neurocomputing 182, 154–164. https://doi.org/10.1016/j.neucom.2015.12.003

Rebollo, J.J., Balakrishnan, H., 2014. Characterization and prediction of air traffic delays. Transp. Res. Part C Emerg. Technol. 44, 231–241. https://doi.org/10.1016/j.trc.2014.04.007

Schultz, M., Reitmann, S., 2019. Machine learning approach to predict aircraft boarding. Transp. Res. Part C Emerg. Technol. 98, 391–408. https://doi.org/10.1016/j.trc.2018.09.007

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1929–1958. https://doi.org/10.1214/12-AOS1000

Tkáč, M., Verner, R., 2016. Artificial neural networks in business: Two decades of research. Appl. Soft Comput. 38, 788–804. https://doi.org/10.1016/j.asoc.2015.09.040

Tomek, I., 1976. Two modifications of CNN. IEEE Trans. Syst. Man Cybern. 6, 769–772.

Trani, A., Wing-Ho, F., Schilling, G., Baik, H., Seshadri, A., 2004. A neural network model to estimate aircraft fuel consumption, in: AIAA 4th Aviation Technology, Integration and Operations (ATIO) Forum. American Institute of Aeronautics and Astronautics, Reston, Virigina, p. 6401. https://doi.org/10.2514/6.2004-6401

Tu, Y., Ball, M.O., Jank, W.S., 2008. Estimating Flight Departure Delay Distributions—A Statistical Approach With Long-Term Trend and Short-Term Pattern. J. Am. Stat. Assoc. 103, 112–125. https://doi.org/10.1198/016214507000000257

Wang, Y., Zhang, D., Liu, Y., Dai, B., Lee, L.H., 2019. Enhancing transportation systems via deep learning: A survey. Transp. Res. Part C Emerg. Technol. 99, 144–163. https://doi.org/10.1016/j.trc.2018.12.004

Wang, Z., Khan, W.A., Ma, H.-L., Wen, X., 2020. Cascade neural network algorithm with analytical connection weights determination for modelling operations and energy applications. Int. J. Prod. Res. 1–18. https://doi.org/10.1080/00207543.2020.1764656

Wang, Z., Ma, H., Chen, H., Yan, B., Chu, X., 2019. Performance degradation assessment of rolling bearing based on convolutional neural network and deep long-short term memory network. Int. J. Prod. Res. 1–13. https://doi.org/10.1080/00207543.2019.1636325

Wilson, D.L., 1972. Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans. Syst. Man. Cybern. 408–421.

Wilson, D.R., Martinez, T.R., 2000. Reduction techniques for instance-based learning algorithms. Mach. Learn. 38, 257–286.

Wu, C.L., 2016. Airline operations and delay management: insights from airline economics, networks and strategic schedule planning. Routledge.

Wu, C.L., Truong, T., 2014. Improving the IATA delay data coding system for enhanced data analytics. J. Air Transp. Manag. 40, 78–85. https://doi.org/10.1016/j.jairtraman.2014.06.001

Xiao, G., Juan, Z., Zhang, C., 2016. Detecting trip purposes from smartphone-based travel surveys with artificial neural networks and particle swarm optimization. Transp. Res. Part C Emerg. Technol. 71, 447–463. https://doi.org/10.1016/j.trc.2016.08.008

Xu, S., Chan, H.K., Zhang, T., 2019. Forecasting the demand of the aviation industry using hybrid time series SARIMA-SVR approach. Transp. Res. Part E Logist. Transp. Rev. 122, 169–180. https://doi.org/10.1016/j.tre.2018.12.005

Yazdi, A.A., Dutta, P., Steven, A.B., 2017. Airline baggage fees and flight delays: A floor wax and dessert topping? Transp. Res. Part E Logist. Transp. Rev. 104, 83–96. https://doi.org/10.1016/j.tre.2017.06.002

Yu, B., Guo, Z., Asian, S., Wang, H., Chen, G., 2019. Flight delay prediction for commercial air transport: A deep learning approach. Transp. Res. Part E Logist. Transp. Rev. 125, 203–221. https://doi.org/10.1016/j.tre.2019.03.013

Zhang, X., Mahadevan, S., 2020. Bayesian neural networks for flight trajectory prediction and safety assessment. Decis. Support Syst. 131, 113246.